3rd ACM SIGSPATIAL International Workshop on Location-Based Social Networks (LBSN 2011)

November 1, 2011 - Chicago, Illinois, USA

Held in conjunction with the 19th ACM SIGSPATIAL GIS 2011

General Co-Chairs:

Christian S. Jensen , Aarhus University, Denmark

Wang-Chien Lee, Pennsylvania State University, USA

Program Co-Chairs:

Yu Zheng, Microsoft Research Asia, China

Mohamed F. Mokbel, University of Minnesota, USA

Conference ID - 2011-41080
ACM ISBN 978-1-4503-1033-8/11/11

1.1 Program Committee:

Sponsors of ACM SIGSPATIAL GIS 2011

Google™ esri®

Microsoft® NAVTEQ™

McCormick
Northwestern Engineering

Table of Contents

Space-Time Dynamics of Topics in Streaming Text

Alexei Pozdnoukhov
National Centre for Geocomputation
National University of Ireland
Maynooth, Co. Kildare, Ireland
Alexei.Pozdnoukhov@nuim.ie

Christian Kaiser
National Centre for Geocomputation
National University of Ireland
Maynooth, Co. Kildare, Ireland
Christian.Kaiser@nuim.ie

ABSTRACT

Human-generated textual data streams from services such as Twitter increasingly become geo-referenced. The spatial resolution of their coverage improves quickly, making them a promising instrument for sensing various aspects of evolution and dynamics of social systems. This work explores space-time structures of the topical content of short textual messages in a stream available from Twitter in Ireland. It uses a streaming Latent Dirichlet Allocation topic model trained with an incremental variational Bayes method. The posterior probabilities of the discovered topics are post-processed with a spatial kernel density and subjected to comparative analysis. The identified prevailing topics are often found to be spatially contiguous. We apply Markov-modulated non-homogeneous Poisson processes to quantify a proportion of novelty in the observed abnormal patterns. A combined use of these techniques allows for real-time analysis of the temporal evolution and spatial variability of population's response to various stimuli such as large scale sportive, political or cultural events.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications—*data mining, mining methods and algorithms, interactive data exploration and discovery*

General Terms

ALGORITHMS

1. INTRODUCTION

Geo-referenced user-generated text streams is a promising source of data for use in a variety of applications ranging from geomarketing to real-time disaster management. The space-time dynamics of topics present in such streams originate from a mixture of complicated processes. A comprehensive analysis of these processes would need to bring together research in social networks, spatial interaction models, thorough understanding of human mobility, advanced models of individual behaviours guiding the way people use electronic media and, finally, natural language models for semantic content extraction and representation.

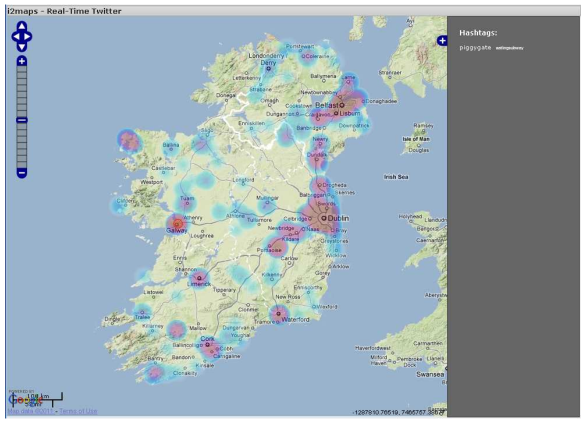


Figure 1: A public interface of our real-time Twitter analysis system showing a total intensity heat map and a cloud of trending hashtags.

1.1 Space-time topic modelling perspective

This paper investigates space-time dynamics of topics found in the data feed available from a micro-blogging service of Twitter[1]. Twitter is a unique datasource in that it provides access to the *location-enabled content* of messages and person-to-person communications. The logs of time-stamped and geo-referenced text messages can be used to quantify the intensities of communication activities aggregated over a particular area and time period [22], while the content can be helpful to identify semantics behind the observed events. We assume a casuality between events (with the real-world ones each having some temporal and spatial extent) and a response in Twitter, a system that we consider here in a role of a measurement instrument. A central modelling tool we use is a content analysis with probabilistic topic models [6]. We expect to see abnormal levels in the number of conversations, their times and locations as a reaction to various stimuli that start discussions on various topics. Local political, sportive or cultural events may cause localised change in activity patterns whereas a national holiday or a serious natural disaster will likely have a much more widespread effect

[1]`http://www.twitter.com`

which at the same time is less adherent to spatial variations. These effects interfere and it is through the topic modelling we are going to carry out a semantically rich high-fidelity analysis to segregate inputs of these different sources in a data-driven and interpretable way.

1.2 Human behavior perspective

There are several factors that bring in extra dimensions of complexity. The major processes that define space-time dynamics of topics in textual streams is the temporal evolution of users' interest to the external stimuli such as news or large-scale events the users are engaged in [15]; and endogeneous processes inherent to human behaviours, such as periodicity of daily routines on one hand and the so-called burstiness [22, 27, 14, 20] on the other. Prior to be extracted for an analysis, these initial responses are subject to the influence of mechanisms typical to social media that proliferates information spread over networks causing phenomena such as multiple re-tweet waves and feedback loops [14].

1.3 Physical movement perspective

Another influencing factor is the physical movement of users that generate texts. Human mobility patterns and their relation to the spatial spread of individual social networks are a subject of intense studies nowadays [5]. A baseline process to keep in mind is an everyday home-to-work commute which account for over 90% of the overall predictability of travel behaviours [26]. At the same time, an adherence to truncated Levy flight type of movements [26] gives evidence that some induviduals cover distances of the orders of magnitudes larger in their typical trips than a majority of other users. Combined with a similar scaling properties with respect to tweeting activities, one may encounter a significant bias introduced by one single "hyper-active" user. Moreover, there seems to be a correlation between movement and texting activities which existence, to the best of our knowledge, has received relatively limited scientific verification [17], but is however evident from the common sense: people often tend to share their experiences by more active blogging or tweeting when they travel on business, for leisure, or during and right after attending various events.

1.4 Spatial community structures

Why would one hope to find spatial patterns in this seemingly messy and chaotic system? There is sufficient empirical evidence of spatially contigious community structures in the way people communicate with each other in their daily lives [28, 23, 9]. Both for landline and mobile phones networks and a variety of temporal scales, it was observed that people tend to communicate with spatially proximate counterparts. Moreover, this pattern is sustainable throught the course of the day [28]: we just switch from the community of co-workers in the office hours to the one composed of friends and family members in the evening. This implies that there may be distinct spatial patterns in information spread processes over social networks. The dispertion channels may appear to be spatially enclosed giving rise to localised regions where particular topics are discussed at increased intensity. People tend to share their views with someone who can give a response, and physical proximity aspect seems to act as a glue in these processes. For the events significant enough to start discussion topics and where no physical movement is involved, we expect to find localised response patterns that show regularities within typical geographical extent of a social network in the given locality. On the other hand, the presense of an untypical mobility pattern common to many users would most definitely break typical structures and may act as an indicator of spatially localised event such as a large-scale festival attracting visitors country-wide. In this paper we experimentally verify these various hypotheses.

1.5 Contributions of this work

We have concentrated our work around probabilistic topic modelling. We argue that understanding the semantic content of what makes a reason for travel or abnormal tweeting intensities is central to the analysis of its space-time dynamics. Topic modelling focuses on the cause of the processes while the spatial spread and temporal dynamics observed via Twitter is a consequence shaped by a variety of factors mentioned above. It is both a purely data-driven approach which avoids making unnessesary assumptions on pre-defining the Regions of Interest to monitore [17] and is a natural way to enhance the interpretability of such analysis for decision-making. We also apply the Markov-modulated nonhomogenious Poisson process models to rigourously quantify levels of abnormality in behaviours associated with various events.

The contributions of our study can be summarized as follows:

- probabilistic topic models are applied to geo-referenced streaming text data (Sections 2, 5);
- spatio-temporal regularities in the topics are examined with a comparative kernel density analysis (Sections 3 and 5). A pathway to enhance the scalability of the method for high-volume and high resolution processing is outlined;
- a rigorous quantitative analysis based on nonhomogeneous Poisson processes is introduced to describe the relative abnormal activity levels associated with the events (Section 4 and 5);
- it is shown that topics found in streams are related to various events and inhere typical spatial patterns. The particularly interesting cases (a spatially local and a global event causing spatially heterogeneous response) are described in detail in Section 5.3.

To conclude the paper, we outline the use of the presented knowledge discovery approach and discusss various issues encountered in the course of this research in Section 6. We now follow on to the description of the topic model we used in this study.

2. TOPIC MODELS

A baseline and powerful topic model is Latent Dirichlet Allocation (LDA) [6]. It is a Bayesian probabilistic model of documents that assumes a collection of K "topics" defined as a multinomial distribution over the given vocabulary. Topics are assumed to have been drawn from a Dirichlet distribution, $\beta_k \sim Dirichlet(\eta)$. Documents d are assumed to be generated from the topics with the following generative process. First a distribution is drawn over topics $\theta_d \sim Dirichlet(\alpha)$. Then, for each word i in the document, a topic index $z_{di} \in \{1, ...K\}$ is drawn from the topic weights

Table 1: An example of LDA analysis of a small corpus of documents

Sample Text Documents	Topic,[γ] ($K_{max} = 2$)	Topic,[γ] ($K_{max} = 3$)
Windy day today, light rain or drizzle in places. #metoffice	**2**, [0.12, 0.88]	**2**, [0.06, 0.89, 0.05]
Going to the Zoo today? See you there. Hope there's no rain.	**2**, [0.07, 0.93]	**1**, [0.91, 0.05, 0.04]
Rain forecasted for Dublin. Stayed home to watch the game with friends.	**1**, [0.93, 0.07]	**3**, [0.04, 0.04, 0.92]
Didn't see giraffes... Mommy said they don't like rain :(	**2**, [0.06, 0.94]	**2**, [0.38, 0.58, 0.04]
Gooooaaaal! Come on, Dublin!	**1**, [0.87, 0.13]	**3**, [0.11, 0.11, 0.78]

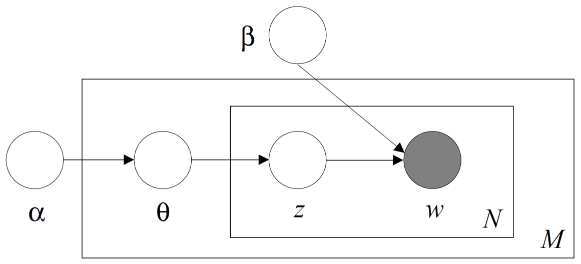


Figure 2: Graphical representation of the LDA model [6].

$z_{di} \sim \theta_d$ and finally the observed word $w_{di} \sim \theta_{z_{di}}$ is drawn from the selected topic. This genrative process is presented as a graphical model in Figure 2.

A purpose of LDA is to analyse a corpus of documents. Posterior distribution of the topics β, topic proportions θ, and topic assignments z conditioned on the documents in the corpus have to be examined to reveal a latent structure in the document collection that can be used for text data exploration. Unfortunately, this posterior can not be computed directly. It is usually approximated using Markov Chain Monte Carlo (MCMC) methods or variational inference. Developing scalable approximate inference methods for topic models is an active area of research as both classes of methods present significant computational challenges in the face of massive data sets.

2.1 Online LDA

With the growth of popularity of social media, analysis of streaming text becomes an area of increasing interest. There are two evident ways in which current developments are targeted. These are the analysis of massive text corpora and the analysis of streaming text. In the latter task, variational Bayes approaches get serious advantage over MCMC methods [10]. Instead of a costly MCMC procedure, variational methods optimize a simplified parametric distribution to be close by the Kullback-Leibler divergence to the posterior. An idea developed in [10] is to minimize it with a stochastic gradient descent over a variational parameter λ parametrizing topic distributions β, and γ defining per-document topic weights. Each update step is made once a new document, or a mini-batch of documents (useful to reduce noise) arrives. The structure of the update follows an Expectation-Maximization [8] procedure for updating λ, with an additional tracking update

$$\lambda \leftarrow (1 - \rho_t)\lambda + \rho_t\hat{\lambda}, \tag{1}$$

where $\hat{\lambda}$ is a result of an EM iteration and $\rho_t = (\tau_0 + t)^{-k}$ with $\tau_0 \geq 0$ and $k \in (0.5\ 1]$ define the rate at which λ is forgotten. It both guarantees convergence and can also be used to a some extent to accomodate temporal drifts in the set of topics being discussed. This is a baseline topic model, and more advanced model for tracking temporally-varying content have been developed recently (for example, [11]).

2.2 A toy example

To illustrate the functioning of LDA and give intuitions for experiments below, we present here a simple example. Table 1 presents a corpus of short text messages where one can see several topics are mentioned, and "weather", "zoo", "sports" amongst others. LDA finds a generative model likely to have produced these texts. Each text is generated from a distribution over topics and therefore composed of several ones with different ratios as defined by γ. The number of topics K_{max} is a user-defined parameter. At $K_{max} = 2$ LDA has found the "Weather" and "Sports" topics, while at $K_{max} = 3$ one finds that "Weather" topic can be decomposed into "Weather" and "Zoo" as shown in Table 1.

Each topic is represented by a distribution over words parametrized by λ. For example, the 5 most probable words in topics found with $K_{max} = 3$ are {rain (0.199), today (0.127), light (0.113), day (0.112), places (0.112), ...} $\sim$ "Weather", {see (0.189), rain (0.12), going (0.094), zoo (0.094), giraffes (0.045), ...} $\sim$ "Zoo", and {dublin (0.219), home (0.109), watch (0.109), friends (0.109), game (0.109), ...} $\sim$ "Sports".

It is through the analyis of words one can reveal semantic behind the topics, give them names and associate with particular events. We hypothetize that large-scale public events generate noticable response in a form of topics that are characterized by particular space-time regularities.

3. KERNEL DENSITY ESTIMATES

The outputs of LDA, i.e. a set of geo-referenced texts labelled by the most probable topic discussed therein, has to be subjected to a point pattern analysis technique in order to discover spatial regularities in their distribution such as the presence of clustering [24]. There are powerful scan statistic techniques for discovering clusters in space-time point processes [16], which are however computationally expensive.

We have chosen to use a kernel density estimate which is a well-established non-parametric statistical technique [25], due its computational effeciency and scalability for processing streaming data, and due to the fact that it provides a way to relate point patterns to continuous areas if a detected topic is caused by an event with a distinct spatial extent. For a set of points $\{\mathbf{x}_1, \ldots \mathbf{x}_n\}$ the estimate one uses is

$$\lambda(\mathbf{x}) = \sum_{i=1}^{n} \frac{1}{\sigma^2} K_\sigma(\mathbf{x}, \mathbf{x}_i), \tag{2}$$

where $K_\sigma(\mathbf{x}, \mathbf{x}_i)$ is a kernel function decaying with distance $\|\mathbf{x} - \mathbf{x}_i\|$, such as a Gaussian RBF, σ is the spatial bandwidth characterizing the rate of decay and its squared value in the

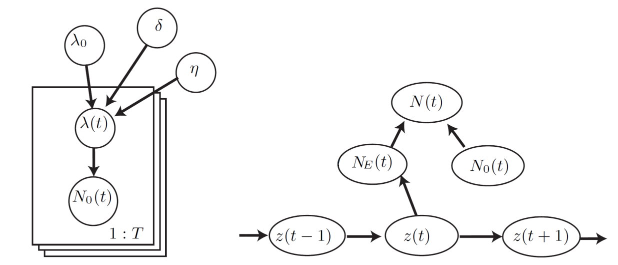


Figure 3: Graphical model representation for the Markov-modulated Poisson process [12].

denominator appears since we are interested in the estimates of intensity of points per unit area. The scalability of the method is achieved due to the linearity of the estimate (3) in terms of data samples that allows straightforward extensions suitable for parallel implementation [13].

3.1 Comparing densities

To identify a spatial pattern in the difference of two processes one has to compare two kernel densities. A useful observation is that the variance of the square root of the density estimate, $\sqrt{\lambda}$ does not depend on the true intensity [7]. We thus use the following value to estimate a difference between two estimates, $\lambda_i(\mathbf{x})$ and $\lambda_j(\mathbf{x})$:

$$\delta(\mathbf{x}) = \frac{\sqrt{\lambda_i(\mathbf{x})} - \sqrt{\lambda_j(\mathbf{x})}}{\sqrt{2\sigma^2 S^2 n_i n_j/(n_i + n_j)}}, \qquad (3)$$

where S^2 is a normalizing constant for the kernel, $S = \int K(x,x')^2 dx$ and n_i and n_j is a number of points in the sets i and j which are being compared. The difference is considered to be significant at the level of $|\delta(\mathbf{x})| = 2$ [7]. This technique allows us to identify spatial areas where the relative intensity of tweets on a particular topic is significantly higher compared to the overall tweeting activity.

4. NOVELTY QUANTIFICATION

An aggregated number of count events, such as the number of tweets or the number of users within an area identified by a KDE test can be compared to the usually observed values to quantify the properties of the observed events, such as, for example, event's attendance. An approach that allows one to avoid introducing an ad-hoc threshold for defining what extra amount of counts makes given moment untypical is required for this task. The other advantage we seek for is to resolve the "chicken-and-egg" problem, i.e. the method should be applicable without knowing which parts of the given time series of count data do not contain novel or untypical components. This has to be learned directly from data with minimum prior information.

4.1 Markov-modulated nonhomogeneous Poisson processes

To quantify a volume of abnormal tweets or users within a topic in a geographic area identified as described in Section 3, we follow [12] and train a Markov-modulated time varying Poisson process model (MMPP). Under this model, one assumes that the number of observed counts are due to periodic routine $N_0(t)$ and additive novel processes $N_E(t)$:

$$N(t) = N_0(t) + N_E(t). \qquad (4)$$

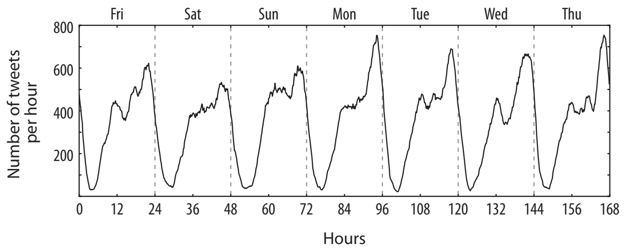


Figure 4: Temporal profile of the number of tweets per hour in a typical week.

A number of events in the periodic component is assumed to follow a Poisson distribution $P(N,\lambda) \sim e^{-\lambda}\lambda^N$ with a temporal profile defined by a time-varying rate $\lambda(t)$. To account for different profiles typical for weekday/weekend, one decomposes $\lambda(t) = \lambda_0 \delta_{d(t)} \eta_{d(t),h(t)}$ where $\delta_{d(t)}$ accounts for the effect of the type of the day (defined with index $d(t) \in \{1,\ldots,D\}$ where D is a total number of different day types), and $\eta_{j,i}$ accounts for the influence of time period i given day of the type j. The coefficients measure the influence of a particular temporal instant relative to the base rate λ_0.

An increased number of counts is accounted for with the term $N_E(t)$:

$$N_E(t) \sim \begin{cases} 0, & \text{if} \quad z(t) = 0, \\ P(N,\gamma(t)), & \text{if} \quad z(t) = 1, \end{cases} \qquad (5)$$

where an indicator variable $z(t) = \{0,1\}$ shows the presence of an event at time t. A Markov structure is assumed for $z(t)$, with a transition probability matrix

$$M_z = \begin{bmatrix} 1 - z_0 & z_1 \\ z_0 & 1 - z_1 \end{bmatrix}. \qquad (6)$$

Parameters in the transition probability matrix thus have the following meaning: $1/z_0$ is an expected length of time period between the events and $1/z_1$ is a duration of an event once it has started.

A graphical model for $N_0(t)$ and $N_E(t)$ is presented in Figure 3. It is trained using Markov chain Monte Carlo (MCMC) approach with each iteration of linear complexity in the length of the time series. It is convenient to use a typical time-length such as a 24-hour interval of the data to simplify inference with MCMC, implying that in applications a type of the day required to sample $\delta_{d(t)}$ and $\eta_{d(t),h(t)}$ is known in advance with certainty.

5. EXPERIMENTS

5.1 Data crawler

Twitter allows people to express their opinions through sending short messages of 160 characters. Each user can define his profile and messages ("tweets" or "statuses") as being public. The user has also the option to define a location that can vary in time. The location can be given as a pair of coordinates, or the name of the locality in a form of a free text. Each user has also an associated time zone, usually defined by the nearest city within this time zone according to the *tz* database [3]. Twitter makes available a sample of these statuses through a streaming API [2]. This streaming API allows filtering the messages in different ways whereas a common approach for studying particular areas is to define a geographical bounding box.

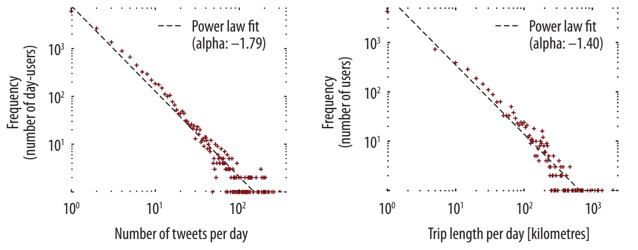


Figure 5: Statistics of tweeting users: both the number of tweets and the distance travelled per day follow power laws with exponents $\alpha = -1.79$ and $\alpha = -1.4$ correspondingly.

Beside the short message, a tweet contains information about the user who sent the message, such as the name, a description, the number of followers and friends, etc. A tweet can also refer to another user using a @<username> syntax, or contain a "hashtag" for making a link to a topic (#<topic>). There can be several hashtags describing the same event or a topic as hashtags are user-generated and is of arbitrary, often wrong or slang-like, spelling.

5.1.1 Geocoding

We log the tweets from the stream filtered by location as defined by a bounding box around the whole island of Ireland. Each message is passed through a geocoding filter. This filter extracts the coordinates of the user if those are available. For further enhancing the rate of geocoded messages, we have used a geocoding routine for user locations given as named geographical entities. It is based on the Yahoo GeoPlanet data [1] and our in-house geocoding engine based on the Sphinx search engine [4]. A simple verification of the speed of each displacement helps filter out errors caused by ambiguities in place's names. Tweets that receive no location as a result of the geocoding procedure are being discarded. The crawler source code is available online[2].

5.1.2 Descriptive statistics

Communication logs provide rich sources of data for understanding fundamental patterns of human activity. The analysis of mobile phone networks [21, 26] and instant messanging communication services [18] have shown particular scaling laws that we investigate here on our dataset. We usually collect about 70'000 messages each week (approx. 7'000 to 10'000 a day). A typical temporal profile of tweeting intensity is presented in Figure 4 with a fitted power law $p(n) \sim n^{-\alpha}$. The intensity of tweeting and the distances traveled per day by each user over the period of study is presented in Figure 5. This is an observation to keep in mind during the analysis of space-time topic dynamics: there are users that travel for up to a thousand of kilometers and others that post hundreeds of tweets a day on a regular basis.

5.2 Analysis scheme

We followed a common analysis scheme in our experiments. The sequence of steps we undertaken uses the methods described in Sections 2-4. We have considered each tweet as one document and trained topic models with $K_{max} = 10, 20, 50, 100$. We have used a held-out perplexity, defined as $\exp(-\sum_d^M \log p(\mathbf{w}_d)/\sum_d^M N_d)$ for a set of M documents of length N_d to asess the quality of topic distribution fit.

[2]http://ncg.nuim.ie/i2maps/

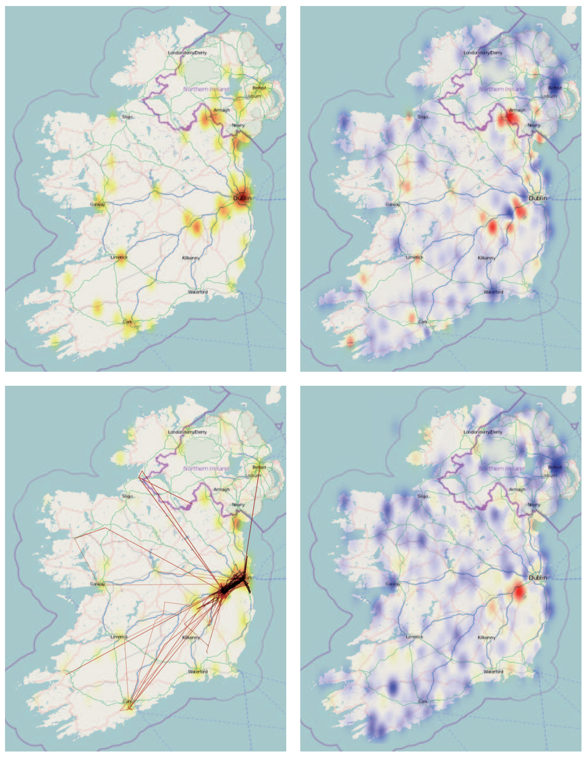


Figure 6: Absolute (left) and relative (right) spatial intensities of the Oxegen topic tweets before (top) and during the festival (bottom). The change in the pattern is due to physical movement: the trajectories of the users on the day before the festival are shown in bottom left.

We then classified all tweets according to the topic of the highest weight. Semantics behind the topics was identified by examining the sets of high-impact dictionary terms.

Absolute and relative kernel density estimates were computed for the point patterns of tweet locations to compare each selected topic subclass vs. the whole dataset of mean user activity. For particular localised events we observed the "hot spots" and selected their geographical extent according to the threshold $|\delta(\mathbf{x})| = 2$ of the estimated difference in spatial densities.

An MMPP model was trained on the time series of counts (overall or specific to a particular locality area) in order to quantify the volume of tweets present in the stream as a response to unusual event. For local estimates we computed aggregates for the number of tweets and users whithin the "hot spot" areas for a relevant temporal period. Below we demonstrate some particularly illustrative findings of this analysis.

5.3 Case study highlights

5.3.1 Oxegen

Oxegen is the largest Irish open air music festival[3]. In 2011 it run for 3 days of 8th-10th of July and its line-up with world-wide famous performers attracted many visitors from all over the country. It was a major topic observed in a Twitter stream in all our experiments throught that time

[3]http://www.oxegen.ie

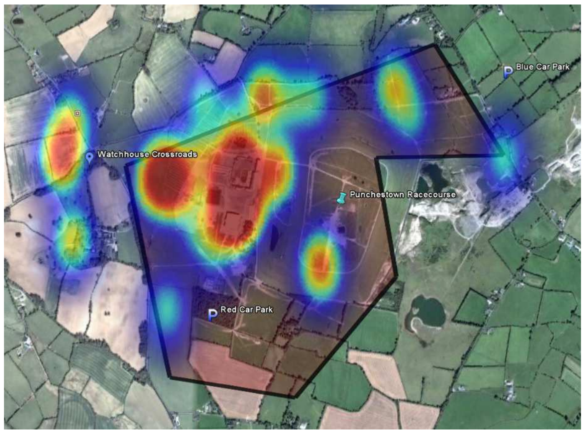


Figure 7: A close-up view on the Oxegen festival area: the relative spatial intensity of tweets from the "oxegen" topic is overlaid on the sattelite imagery. Festival boundaries as given on the official website of the event are shown as a polygon.

period. Figure 6 presents a relative density of the "Oxegen" topic obtained with a KDE estimate with spatial bandwidth $\sigma = 1km$ for the tweets recordered during 8^{th}-10^{th} of July. The hot spot on the map matches the location of the festival.

A fine-scale estimate with $\sigma = 50m$ within the festival area is shown in Figure 7. It delineates the physical boundaries of the festival area and its facilities such as parking lots with surprising precision. One can even reasonably assume a misalighnment of the official boundaries (shown as a polygon in the Figure 7) with the actual ones at its Northern side.

Note that the absolute intensity of the topic does not give a unique indication of the festival location as this was still a major discussion topic in some other places. Sample tweets from users not attending the festival illustrate the point[4]. A temporal profile of the topic intensity during and after the festival is presented in Figure 10, showing temporal decay in post-festival discussions.

Finally, we tracked an evolution of the words in the topic corresponding to Oxegen. Figure 8 presents the words in the topic on Friday as a tag cloud where the size of the word corresponds to its weight λ in a topic. One can notice considerable weigths of "hope" and "rain". The word tag cloud for the last, key and most attended day of the festival, Sunday, is presented on the right of Figure 8. The actual weather appeared to be dry and the festival was a huge success.

5.3.2 *Harry Potter movies weekend*

The screening of the final episode of the Harry Potter sequel[5] began on July 15th, and the premiere was held at the same time all over Ireland. The movie has beaten box office records for the day of premiere. We have trained a topic model for the time period of screening and estimated the popularity of the Harry Potter topic for the time period spanning the previous and following weeks. One can notice a growing interest which culminates on the evening preceeding the premiere (Figure 10). The spatial pattern of the topic

[4]"Have to keep changing the radio station cause all I keep hearing is 'oxegen oxegen oxegen' gonna be a #boring weekend @ home ha!"; "Keeva said she'd ring me from Oxegen for Beyonce, bet she's too drunk to remember"

[5]`http://harrypotter.warnerbros.com`

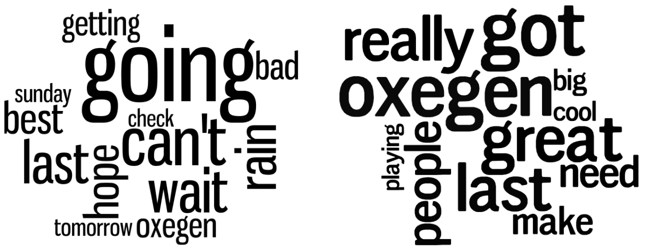


Figure 8: The cloud of the most probable words in the "Oxegen" topic the day before the festival (left) and on its last day (right). Text size is proportional to the word's weight.

popularity shows no particualr hot spots and roughly follow the density of population and the distribtion of cinemas in largest cities (Figure 9).

A hot spot in top right of Figure 9 corresponds to a discussion duirng the pre-screening of the movie at the Trafalgar Square in London, UK, on Thursday the 7th of July. Surprisingly, the response to this event is only visible in Cork and went unnoticed elsewhere. We hypotethise that this is the local community structures [28] that significantly predefine spatial spread of the response.

5.3.3 *Multiple event discovery*

An MMPP model highlighted a significant unexplained variance of tweets on the weekend of Harry Potter movies. To our surprise the posterior probability of abnormal events presence was high but the "Harry Potter" topic alone was not accounting for the increased volumes, especially on Sunday, July the 17^{th}. We have considered other major topics discovered during these days. They included The Open UK golf tournament ("Golf"), Ulster finals of GAA sports, and the broadcast of the final episode of The Apprentice TV show ("BBC"). The contribution of these events explains 85% of the extreme component of Twitter activity during that weekend (Figure 11), and it's spatial pattern indicates various hot spots in Nothern Ireland. The intensity of the "Harry Potter", "Golf" and "BBC" topics are presented in Figure 10.

6. DISCUSSION AND CONCLUSIONS

There is vast potential to enhance the resolution of the analysis of geo-referenced social media streams both in terms of time-space granularity so as in terms of understanding the variability of their content. The analysis scheme we introduced in this paper helps revealing high-fidelity patterns in the spatial and temporal variation of the response of Twitter users to particular events. In this work we have advanced the state-of-the-art with a purely data-driven and rigorous quantitative approach where we avoided introducing pre-defined thresholds to describe the level of "novelty" of the observed event both in space and time. Our approach explores this issue in a purely data-driven way, while, for example, the categorization proposed for geo-social events in [17] imposes constraints on the type of processes it can accomodate.

A powerful machine learning framework of probabilistic topic modelling was applied to identify trending topics in the Twitter streams. This approach improves over simple filtering by hashtags or keywords which is a baseline

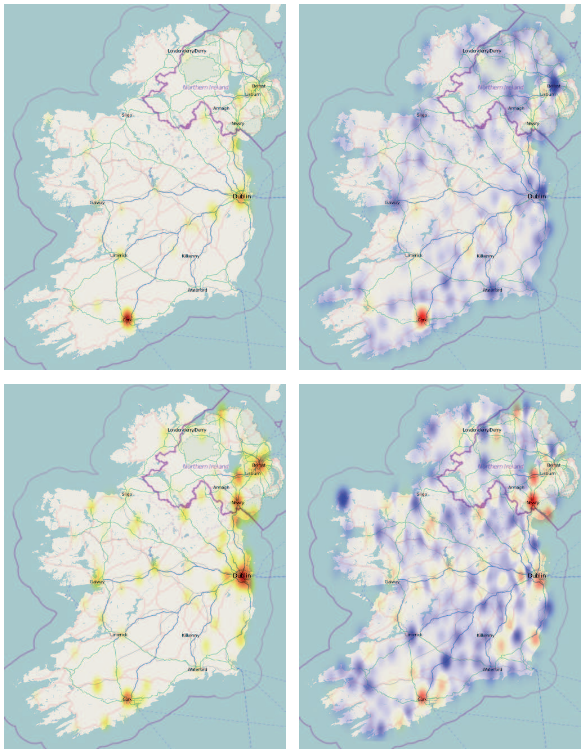


Figure 9: Absolute (left) and relative (right) spatial intensities of the Harry Potter topic tweets during pre-screening (top) and world premiere (bottom).

level of analysis of tweets. Generally, event tracking in online communities is a popular research field [19], and topic models are being extended to explicitly accomodate temporal trends [11]. A prominent extension following this work would be an implicit integration of space-time parameters at the level of topic modelling. One can investigate modelling schemes when two documents are assumed to be more likely to share a topic if besides co-occurencies of words they are proximate in space-time, or movement patterns of the users preceding the publishing of posts are similar.

The amount of abnormal tweets was quantified with an MMPP model and related to the events discovered through topic modelling (Figure 11). This is a major advancement over our previous work [22], where the reasons behind the abnormal levels of activities could not be explained due to the lack of access to the content. However, we were able to explain this variability post factum, and more work is needed to model the evolution of topics in real time. This will require approaches better accounting for intrinsic nature of human behaviour. For example, we have noticed that an event can cause increased level of activity with a distinct event-related topic, however, its intensity fades out and is replaced by a prolonged period of a general chatter.

We have observed different mechanisms leading to the space-time hot spots of topics discussed in Twitter. The first type is caused by physical movement of users and is typical for local event such as a festival, while the second one is due to the increased intensity with which a topic is discussed within a local community [28] as a response to an important event. What's more, empirical evidence suggests that an event causing this effect can both be of local or external origin. Strict quantification and modelling of these

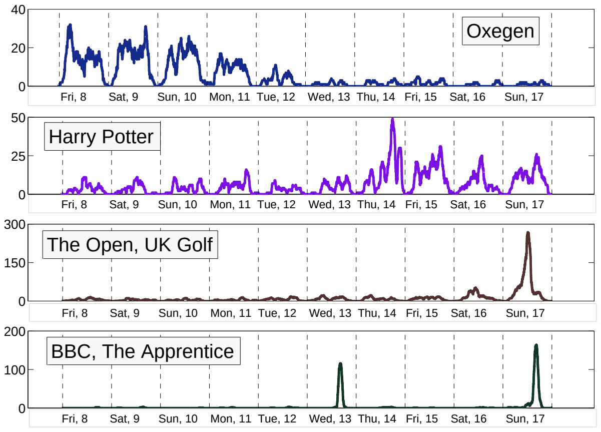


Figure 10: Temporal profile of the number of tweets in major LDA topics.

mechanisms as an important direction of our future work.

Finally, we note that our analysis was carried out for the "population" of Twitter which can hardly be considered as a representative for a total country population. It is heavily biased towards younger age groups and spatial statistics approaches has to be applied to adjust the produced estimates based on census data. We consider this to be a necessary step in our future work which would allow us to uncover further possibilities available through the analysis of geo-referenced social media streams.

7. ACKNOWLEDGMENTS

Research presented in this paper was funded by a Strategic Research Cluster grant (07/SRC/I1168) and Stokes Lectureship award by Science Foundation Ireland under the National Development Plan, ERA-NET Complexity project COSMIC and the Marie-Curie ITN Geocrowd.

8. REFERENCES

[1] `http://developer.yahoo.com/geo/geoplanet/data/` (last accessed 18.07.2011).

[2] `http://dev.twitter.com/docs/streaming-api` (last accessed 18.07.2011).

[3] `http://en.wikipedia.org/wiki/List_of_tz_database_time_zones` (last accessed 18.07.2011).

[4] `http://sphinxsearch.com/` (last accessed 18.07.2011).

[5] K. W. Axhausen. Social networks, mobility biographies, and travel: survey challenges. *Environment and Planning B: Planning and Design*, 35:981–996, 2008.

[6] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, 2003.

[7] A. W. Bowman and A. Azzalini. *Applied Smoothing Techniques for Data Analysis: The Kernel Approach with S-Plus Illustrations.* Oxford Statistical Science Series. Oxford University Press, USA, Nov. 1997.

[8] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1–38, 1977.

[9] P. Expert, T. S. Evans, V. D. Blondel, and R. Lambiotte. Uncovering space-independent

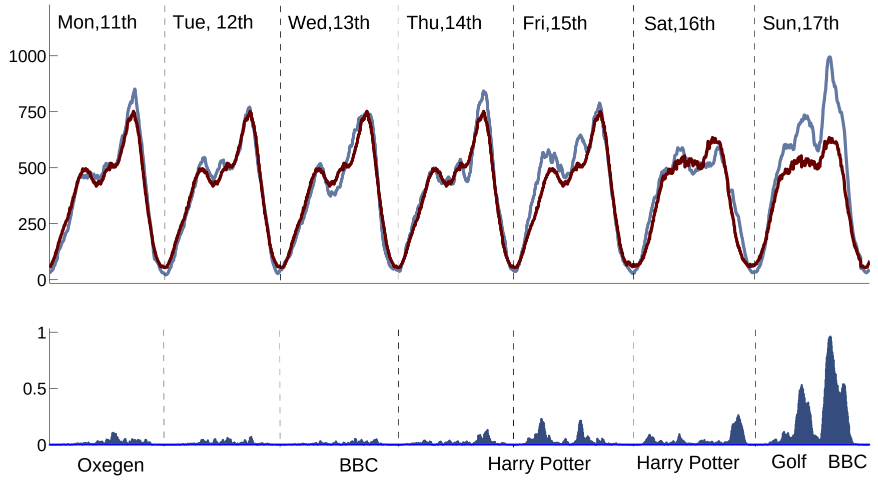


Figure 11: Top: temporal variation of the amount of tweets (light line) and its MMPP-predicted normal level (dark line). Bottom: posterior MMPP probability of abnormal event and a prevailing LDA topic.

communities in spatial networks. *Proceedings of the National Academy of Sciences*, 108(19):7663–7668, May 2011.

[10] M. D. Hoffman, D. M. Blei, and F. Bach. Online learning for latent dirichlet allocation. In *NIPS*. MIT Press, 2010.

[11] L. Hong, D. Yin, J. Guo, and B. Davison. Tracking trends: Incorporating term volume into temporal topic models. In *ACM SIGKDD: Knowledge Discovery and Data Mining*, San Diego, CA, USA, 2011.

[12] A. Ihler, J. Hutchins, and P. Smyth. Adaptive event detection with time-varying poisson processes. In *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 207–216, New York, NY, USA, 2006. ACM.

[13] C. Kaiser and A. Pozdnoukhov. Enabling real-time city sensing with kernel stream oracles and mapreduce, June 2011. The First Workshop on Pervasive Urban Applications (PURBA).

[14] M. Karsai, M. Kivelä, R. K. Pan, K. Kaski, J. Kertész, A.-L. Barabási, and J. Saramäki. Small but slow world: How network topology and burstiness slow down spreading. *Phys. Rev. E*, 83(2):025102, Feb 2011.

[15] J. Kleinberg. Temporal dynamics of On-Line information streams. In *Data Stream Management: Processing High-Speed Data Streams*, 2005.

[16] M. Kulldorff. A spatial scan statistic. *Communications in Statistics - Theory and Methods*, 26(6):1481–1496, 1997.

[17] R. Lee and K. Sumiya. Measuring geographical regularities of crowd behaviors for twitter-based geo-social event detection. In *Proceedings of the 2nd ACM SIGSPATIAL International Workshop on Location Based Social Networks*, LBSN '10, pages 1–10, New York, NY, USA, 2010. ACM.

[18] J. Leskovec and E. Horvitz. Planetary-scale views on a large instant-messaging network. In *WWW '08: Proceeding of the 17th international conference on World Wide Web*, pages 915–924, New York, NY, USA, 2008. ACM.

[19] C. X. Lin, B. Zhao, Q. Mei, and J. Han. Pet: a statistical model for popular events tracking in social communities. In *Knowledge Discovery and Data Mining*, pages 929–938, 2010.

[20] R. D. Malmgren, D. B. Stouffer, A. E. Motter, and L. A. N. Amaral. A poissonian explanation for heavy tails in e-mail communication. *Proceedings of the National Academy of Sciences*, 105(47):18153–18158, Nov. 2008.

[21] J.-P. Onnela, J. Saramäki, J. Hyvönen, G. Szabó, M. A. de Menezes, K. Kaski, A.-L. Barabási, and J. Kertész. Analysis of a large-scale weighted network of one-to-one human communication. *New Journal of Physics*, 9(6):179, 2007.

[22] A. Pozdnoukhov and F. Walsh. Exploratory novelty identification in human activity data streams. In *Proceedings of the ACM SIGSPATIAL International Workshop on GeoStreaming*, IWGS '10, pages 59–62, New York, NY, USA, 2010. ACM.

[23] C. Ratti, S. Sobolevsky, F. Calabrese, C. Andris, J. Reades, M. Martino, R. Claxton, and S. H. Strogatz. Redrawing the map of great britain from a network of human interactions. *PLoS ONE*, 5(12):e14248, 12 2010.

[24] B. D. Ripley. *Spatial Statistics*. Wiley-Interscience, 1981.

[25] B. W. Silverman. *Density Estimation for Statistics and Data Analysis*. Chapman and Hall/CRC, 1 edition, April 1986.

[26] C. Song, Z. Qu, N. Blumm, and A.-L. Barabási. Limits of predictability in human mobility. *Science*, 327(5968):1018–1021, Feb. 2010.

[27] A. Vázquez, J. G. Oliveira, Z. Dezsö, K. I. Goh, I. Kondor, and A. L. Barabási. Modeling bursts and heavy tails in human dynamics. *Physical Review E*, 73(3):036127+, Mar. 2006.

[28] F. Walsh and A. Pozdnoukhov. Spatial structure and dynamics of urban communities, June 2011. The First Workshop on Pervasive Urban Applications (PURBA).

Extracting Urban Patterns from Location-based Social Networks

Laura Ferrari, Alberto Rosi, Marco Mamei, Franco Zambonelli
Dipartimento di Scienze e Metodi dell'Ingegneria
University of Modena and Reggio Emilia, Italy
name.surname@unimore.it

ABSTRACT

Social networks attract lots of new users every day and absorb from them information about events and facts happening in the real world. The exploitation of this information can help identifying mobility patterns that occur in an urban environment as well as produce services to take advantage of social commonalities between people. In this paper we set out to address the problem of extracting urban patterns from fragments of multiple and sparse people life traces, as they emerge from the participation to social networks. To investigate this challenging task, we analyzed 13 millions Twitter posts (3 GB) of data in New York. Then we test upon this data a probabilistic topic models approach to automatically extract urban patterns from location-based social network data. We find that the extracted patterns can identify hotspots in the city, and recognize a number of major crowd behaviors that recur over time and space in the urban scenario.

Categories and Subject Descriptors

G.3 [**Probability and statistics**]: Time series analysis; H.3.3 [**Information Search and Retrieval**]: Retrieval Models; I.5.2 [**Design Methodology**]: Pattern Analysis

General Terms

Algorithms, Human Factors, Measurements, Experimentation, Performance, Verification

Keywords

social dynamics, spatio-temporal data mining, information retrieval in location-based social networks, semantic meaning and knowledge discovery from location-related data

1. INTRODUCTION

Thanks to advancement in mobile technologies, PDAs and smart-phones are the most rapidly growing technologies in the world [8]. With the assent of the owner, such devices silently observe people going through their lives, record the places they visit, the information they share, the people with whom they interact. Finally, the information produced can be globally shared in social networks, under the shape of a Facebook post, a Twitter tweet, a Flickr geolocalized picture, etc.

ACM LBSN '11, November 1, 2011. Chicago, IL, USA
Copyright ©2011 ACM ISBN 978-1-4503-1033-8/11/11...$10.00

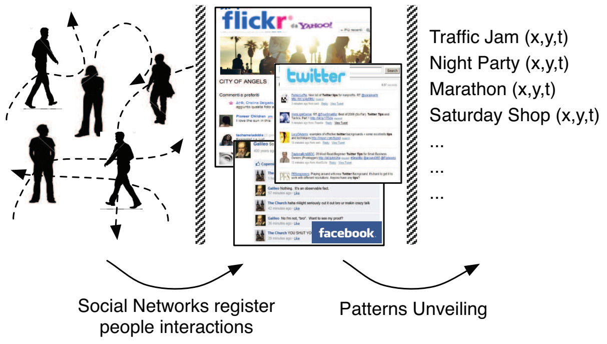


Figure 1: From people interactions to events

Social data represents a still under-explored treasure of information, especially when it is further enriched with the location dimension (e.g., Facebook Places, Geolocalized Twitter Post or Flickr Pictures). Beside sparse in the geographical space, incomplete in timespan and fragmented on millions of users, social data expresses a myriad of "life traces" describing, by social network conventions, the way in which people live and interact in their own city. Such vision is exemplified in Figure 1. On the left side, people produce traces during their own lives; such traces intersect when people participate, create or, are subjected to, events that are happening in the same time and place in the urban environment.

It is important to remark that if from one side social networks are effective in registering and capturing single user moods and comments, they are not conceived for performing analysis at a crowd level. Despite the sparsity of social networks data, in our view it is possible to extract from distinct traces higher level information such as people similarities, recurrent behaviors, or inference on upcoming events.

In this paper we set out to investigate the possibility of extracting urban patterns from location-based social networks data. In particular, we explore the extent that recurrent patterns in an urban environment can be computed if we use only the spatio-temporal feature of social network data. To do so, we apply a topic model-based approach on a large dataset of Twitter posts. We want to show that our proposal, based on a probabilistic topic model approach, can successfully discover high-level patterns from location-based social networks. By applying our approach, we make two main contributions:

1. we show that a probabilistic topic model technique can be successfully applied not only to a "user-centric" scenario, but also to a "city-centric" ones. In fact, in previous state-of-the-art works, topic models have been applied to extract patterns and routines behaviors from a fairly large and accurate log of individual users behaviors [5, 6]. Here we propose a topic model based approach able to infer high-level patterns from data representing the life of a large set of users but in an extremely sparse and spotty way.

2. we show that our approach can be successfully applied to the sparse data coming from location-based social networks. We tested our approach on a large (3 GB) collection of geo-localized Twitter posts, created over the city of New York from June 2010 to June 2011. We show that our approach can recognize and extract, in an unsupervised manner, recurrent behaviors and high-level patterns that recur over both the space and time dimensions. Such results can find a natural application in grouping together people with similar interests and usage of public spaces, as well as in tracking or predicting events that have a strong relation with the urban environment.

The remainder of this paper is organized as follows. In Section 2, we discuss related work in the area of pattern recognition of geo-localized data. In Section 3 we describe our model for adapting social mobility traces to topic models. In Section 4, we discuss, evaluate and validate results. In Section 5, we describe the possible applications that are enabled by the automatic extraction of urban patterns. Section 6 concludes and identifies areas for future research.

2. RELATED WORK

In recent years, many approaches have been proposed both for *(i)* extracting hotspots and *(ii)* identifying spatio-temporal patterns from geo-localized data.

Extracting hotspots.
The CitySense project (*http://www.citysense.com*) uses GPS and WiFi data to cluster people whereabouts and discover hotspots of activity in the city area. In a similar work based on extremely large anonymized mobility data coming from Telecom operators, authors were able to identify the most visited areas by tourists during the day and the typical time of the visit (see for example [3], [7]). Another group of works is based on photo-sharing sites which contain billions of publicly accessible images taken virtually every where on earth. These photos are annotated with various forms of information including geo-location that implicitly identifies the location of the user. Researchers have been able to analyze a global collection of geo-referenced photographs, and evaluate them on nearly 35 million images taken from Flickr with the goal of identifying hot spots (and also tourist routine behaviors) [9, 12, 14] . From a complementary perspective, the problem of finding boundaries for vague regions corresponding to human-centered areas of the city looking at Web query logs was also studied in [17, 19].

Identifying spatio-temporal patterns.
Several researches have been developed to identify recurrent patterns in mobility data. In particular, some works rely on the identification of frequent sequential patterns to analyze spatio-temporal datasets. For example, in [10] the presence of frequent sequential patterns are used to find recurrent mobility patterns. Eagle and Pentland [4], use Principal Component Analysis (PCA) to identify the main components structuring daily human behavior. The main components of the human activities, which are the top eigenvectors of the PCA decomposition are termed *eigenbehaviors*. Similarly, the work presented in [18] compares different data mining techniques to extract patterns from mobility data. In particular, they found Principal Component Analysis (PCA) and Independent Component Analysis (ICA) best suited to the task of identifying daily patterns.

A recent group of works is based on topic models, which are powerful tools initially developed to characterize text documents [2] and are at the base of our approach. In [5] authors propose the use of probabilistic topic models to capture human routines from cell tower connections. They were able to understand both individual behaviors and interactions. Similarly, in [16] authors propose a Latent Social Theme Dirichlet Allocation to automatically discover high-order temporal social patterns from very noisy and sparse location data. They apply the proposed framework to a real-world noisy dataset collected over 1.5 years, and show that useful and interesting patterns can be computed. In [6] topic models are applied on GPS signals obtained from the Google Latitude application. They used grid-based algorithms to extract significant locations that are either automatic and manually labeled. Locations are analyzed in a broader resolution than in [5, 16], involving places such as pub, cinema, disco, etc. Topic models are then used in a similar way as proposed in [5].

Our approach mainly deviates from the current state of the art works in two relevant aspects:

1. **Single-to-crowd analysis.** The focus of our analysis doesn't concern user-centric behavior (above works in literature extract routine patterns of individual users) but rather we mine for daily routines describing common patterns of the city as a whole. This kind of patterns extractions support the novel classes of applications shown in section 5.

2. **Complete-to-sparse datasets.** Works in literature applied topic models to datasets that are very hard to obtain (mainly for privacy purposes), since they are built on complete traces over the 24h. Our approach adapts topic models to work on a large set of scattered mobility traces as they emerge from location-based social networks (e.g., Twitter, Foursquare, Brightkite);

most of the time such data locates a huge number of users only for a tiny fraction of the day.

3. A MODEL FOR REPRESENTING SOCIAL MOBILITY TRACES

In this section, we present the mechanisms at the core of our approach to extract patterns and routine behaviors from location-based social networks.

In extreme summary, our approach is based on three steps:

1. **Crowd Detection:** given a selected geographical area of analysis, we detect the most crowded areas at any given hour of our dataset. Crowd detection is based on social networks life traces, in our case we bank on the number of Twitter posts sent by users. Obviously such "measure" of crowd distribution could be replaced or combined with other location-based social networks, such as the number of picture taken in that area (considering the Flickr database), the number of Facebook posts sent from there, etc.
2. **Data preparation:** starting from the above results, we translate such data in a form that is suitable for topic models. In particular, we transformed each day in our dataset in a complete collection of *crowd-footprints* (as detailed in section 3.2) each indicating the time and the areas of the city that are most crowded. Such collection of *crowd-footprints* is the input data structure for urban patterns recognition.
3. **Urban patterns recognition:** on the above crowd aggregated data, we run a topic models algorithm to discover recurrent behaviors (i.e.,*topics*, as detailed in section 3.3) over space and time. As depicted in Section 5, such crowd behaviors could help in track or predicting events that have a strong relation with an urban environment.

In the following of this section we detail the above three steps.

3.1 Crowd Detection

The first step of our approach is to find what are the most crowded areas in the city at a given time slot. Our approach applies to a dataset of geolocalized posts from many users of a social network site spanning several days. The approach works as follow:

1. We divided each day into a $H = 12$ time-slots lasting 2 hours each. In general the greater the number of time-slots, the finer the routine behaviors being extracted. From our experiments we found that for our dataset 12 time-slots provide the most meaningful routine behaviors.
2. For each day d and time-slot h, we cluster location-based social networks data (in our case Twitter posts) with the aim of finding the most visited locations. In our experiments we used the EM algorithm [1] to perform this computation.

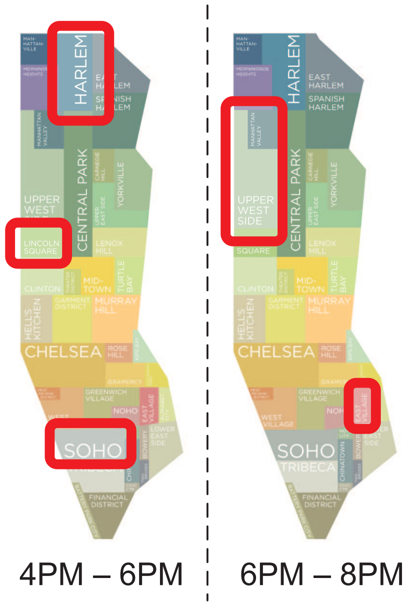


Figure 2: Crowd-footprint in a city. This figure shows the concentration of people in the city at different times.

3. We associate each cluster to the zip-code defined area it belongs to (see Fig. 2). Despite this step is not functional to topics extraction, it helps in characterizing an urban area in terms of the kind of activities being performed there.

This procedure identifies where and when the city is most crowded and how the crowd shifts across the city. This kind of detection focuses on city-centered information and disregards individual user behaviors.

3.2 Data preparation

In the second step, from the above identified footprints, we created for each day in our dataset a representation that is suitable for topic models.
Here we define a $crowd-footprint_x$ as a 2-tuple consisting of a time π_x and a place l_x where the $crowd-footprint_x$ is experienced: $crowd-footprint_x =< \pi_x, l_x >$.

More in details, we constructed each $crowd-footprint$ in the following way:

- starting time π_x is mapped into H time-intervals representing the hour of the day;
- a dictionary of Z zip codes is constructed and the nearest zip code of l_x is used.

This method of construction will theoretically yield a maximum number of *crowd-footprints* of $W = H * Z$ if all combinations of π_x and l_x are observed in the data.

In summary, each day is translated into a sequence of *crowd-footprints*. For example, a day is translated into the following sequence: $0-10001, 0-10004, 1-10008, 2-10010$,etc. thus representing that the crowd is located in zip 10001 and 10004 from 0am-2am, in zip 10008 from 2am-4am etc. The set of all days represents the input data structure for the next step.

3.3 Urban patterns recognition

In our methodology we adopted a probabilistic topic model technique [1] to identify the routine behaviors with which people move and cluster across the city. Topic models are powerful tools initially developed to characterize text documents, but can be extended to other collections of discrete data (e.g., mobility data). They are probabilistic generative models that can be used to explain multinomial observations by unsupervised learning. Formally, the entity termed word is the basic unit of discrete data defined to be an item from a vocabulary of size V. A document is a sequence of N words. A corpus is a collection of D documents. As above mentioned, in this paper a word is represented by a place and a time slot (i.e., $crowd-footprint$), while a document is a day of the city. There are K latent topics (i.e. routines) in the model, where K is defined by the user.

Among other topic modeling algorithms, in this paper we apply Latent Dirichlet Allocation (LDA) [2]. LDA has two main characteristics that make it suitable to our pattern discovery task. On the one hand, it is an unsupervised approach: it does not require to define classes (i.e. topics) *a priori* and it does not require difficult-to-be-acquired labeled data. On the other hand, topics represent meaningful probabilistic distributions over words and documents. This allows to analyze and understand the routine behavior they stand for. As shown in Section 2, LDA has been applied in this scenario only to "user-centric" data, where patterns and routine behaviors have been extracted from a fairly large and accurate log of individual users behaviors [5, 6]. One important contribution of this paper is to show an approach to apply LDA on data coming from location-based social networks, that represent the life of a large set of users but in an extremely sparse and spotty way.

More in detail, LDA is based on the Bayesian network depicted in Figure 3. A word w is the basic unit of data, representing $crowd-footprint$ at a given *time-label*. A set of N words defines a day of the city (i.e. a document). The city taken into consideration has a dataset consisting of D documents. Each day is viewed as a mixture of topics z, where topics are distributions over words (i.e., each topic can be represented by the list of words associated to the probability $p(w|z)$). For each day i, the probability of a word w_{ij} is given by $p(w_{ij}) = \sum_{t=1}^{K} p(w_{ij}|z_{ik})p(z_{ik})$, where K is the number of topics. $p(w_{ij}|z_{ik})$ and $p(z_{ik})$ are assumed to have multinomial distributions. Mixture parameters are assumed to have Dirichlet distributions with hyperparameters α and β respectively. LDA uses the EM-algorithms to learn the model parameters. In our implementation we use the library LingPipe (http://alias-i.com/lingpipe/) to perform these computations. Once the model is trained, Bayesian deduction allows to extract the topics best describing the routines of a given day (rank z on the basis of $p(d|z)$).

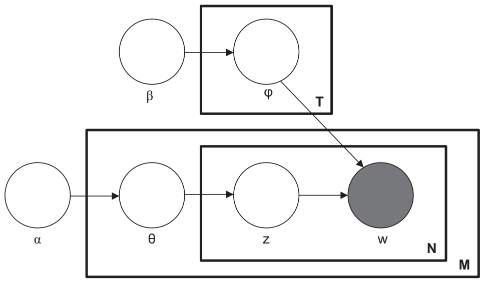


Figure 3: Plate notation representing the LDA model. α is the parameter of the uniform Dirichlet prior on the per-document topic distributions. β is the parameter of the uniform Dirichlet prior on the per-topic word distribution. θ_i is the topic distribution for document (day) i, ϕ_j is the topic distribution for word j, z_{ij} is the topic for the j-th word in document i, and w_{ij} is the specific word. The w_{ij} are the only observable variables.

4. EXPERIMENTS

In this section, we describe some experiments we conduced to evaluate the effectiveness of the proposed approach. First, we describe and motivate the adopted experiment setup, then we illustrate and discuss the obtained early results. Finally, we analyze the accuracy of the extracted topics.

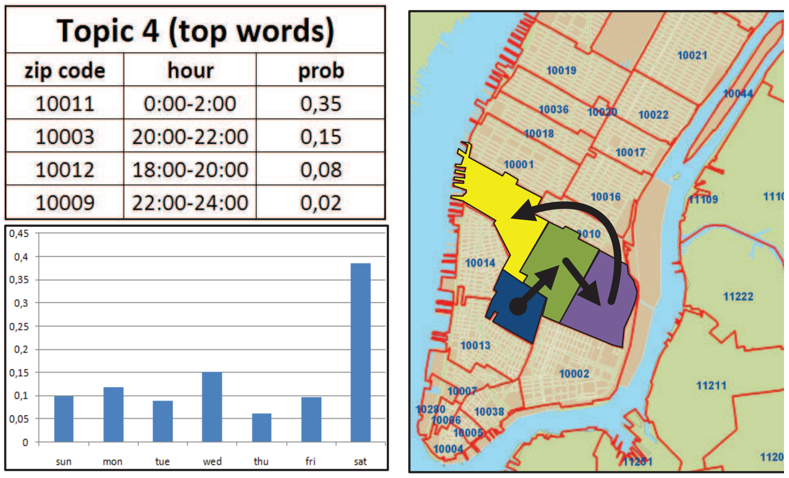
4.1 Experiment Setup

The first aspect to consider to experiment with the proposed topic-based discovery is to select a dataset of geo-localized social network posts on which to ground the analysis. As the majority of social networks provide public API to access their data, the fundamental question is whether it is possible to have data critical mass to conduct meaningful analysis.

Among several candidates, Flickr and Twitter are those systems offering best access modality to a large number of geo-localized data. However, in our experience even Flickr and Twitter, despite the terabytes of data they produce daily, provide enough geo-localized data only on few selected cities. Accordingly, In this paper we used a dataset of mobility traces collected with Twitter over the city of New York (NY, USA) to test the effectiveness of the proposed approach. In particular, the dataset consists of all geo-referenced Twitter over a period of 1 year centered over Manhattan.

Despite many social sources haven't yet reached a mass of data enabling for robust information extraction, our strong belief is that in the next future social networks will be subjected to explosive adoption rates, rapidly reaching the threshold for being exploited in future pervasive applications and services.

Before staring the experiment, we run some clean-up on the data. Since we are interested on mobility patterns of people, we removed all those Twitter users that post always from the same coordinates, since they are likely to be services in which the geo-tags are associated to the service's premises and not to a real moving user. After this, we were left with 13 millions tweet with an average of 30.000 tweets per day.



Topic 4 (top words)

zip code	hour	prob
10011	0:00-2:00	0,35
10003	20:00-22:00	0,15
10012	18:00-20:00	0,08
10009	22:00-24:00	0,02

Figure 4: Routine behavior in Manhattan.

4.2 Results and Discussion

We applied the proposed mechanism to Twitter data with the aim of identifying hotspots in the city life, or rather crowd behaviors that recur over time and space in an urban scenario.

In particular, for these experiments we instantiated the LDA model setting $K = 30$ topics. In general the greater the number of topics, the finer the routine behaviors being extracted. The estimation of the optimal number of topics is an active research challenge and some mechanisms have been proposed to guide this choice [2, 5].

In a first experiment, we try to understand if the extracted patterns are meaningful and coherent with the expected city life. The LDA model successfully revealed some trends characteristic of the city. In Fig. 4 and Fig. 5, we illustrate two exemplary topics.

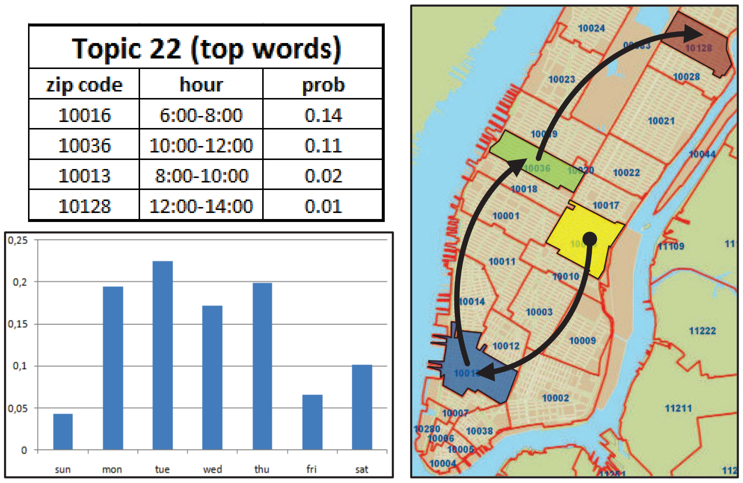
Topic 4 represents a typical weekend activity centered on Saturdays. It shows an hotspot centered around Greenwich village, East village and Nolita districts in Manhattan, characterizing the nightlife. The topic is represented by the table at the top left of the figure. It comprises several zip codes associated with a time period and a probability of that area begin one of the most crowded areas in the city. The topic is also represented in the map on the right of the figure, where zip codes are highlighted and arrows are drawn to illustrate how the concentration of people (i.e., crowd-footprint) shifts from an area to the other. Finally, at the bottom-left of the figure, we present the distribution of the topic over the days of the week. From this, we can see that this is a typical Saturdays' pattern.

Given the lack of accurate ground-truth information, and the fact that topics cluster data in classes that are not defined a priori, it is difficult to provide sound measures on the accuracy of the obtained results. However, the fact that the identified district from the hotspot are well-known areas of the city of night-life entertainment, partially validate the obtained results.

From a similar analysis, Topic 22 represents one possible week day in which the focus of activity is scattered all-over Manhattan ranging from Gramercy, Chelsea, Soho and East Harlem. Many other topics, involving other districts, describe other possible configurations of weekday activities. This is in-line and compatible with the mixed structure of Manhattan districts, but missing groundtruth data it is difficult to validate results further.

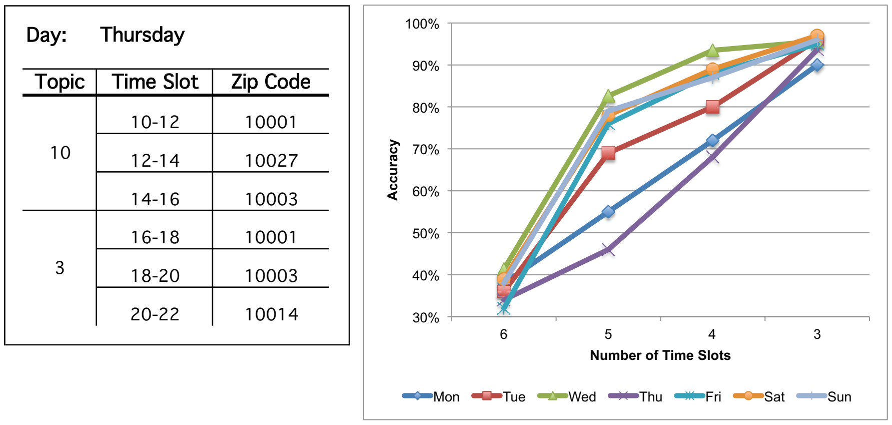
It is worth emphasizing that the represented topics do not correspond to the actual movement of people. People in an area might be different from the the people in another area. Topics represent only where there is a concentration of people and how this concentration moves across the city. The identity of individual is totally disregarded. This is very important in our opinion in that this kind of topic analysis completely preserves individual privacy and only illustrates aggregated information. The extensive study on protecting privacy in location-based social networks (e.g., [15, 20]) further motivates our topic models-based approach.

In summary, this first experiment illustrates that the LDA model applied to Twitter data successfully reveals different types of patterns, by assigning characteristic trends to various topics with a probability measure (p(w|z) and p(d|z)). In addition, based on such method, we are able to answer to several interesting questions such as “Are there specific patterns occurring on weekends versus weekdays?” or “How do the topics characterize the days in the dataset?”. The above results illustrate also one of the key advantages of LDA compared to other clustering mechanisms (e.g., k-means). While most other clustering algorithms group together days that are similar for the whole 24 hours, LDA can cluster days that are similar only in a given time interval. For example, LDA can cluster the days for a specific night-life activity, even if those days have very different signatures in the morning. Other clustering mechanisms would not be able to identify that cluster since they consider whole days only [5].



Topic 22 (top words)		
zip code	hour	prob
10016	6:00-8:00	0.14
10036	10:00-12:00	0.11
10013	8:00-10:00	0.02
10128	12:00-14:00	0.01

Figure 5: Routine behavior in Manhattan.



Day: Thursday

Topic	Time Slot	Zip Code
10	10-12	10001
	12-14	10027
	14-16	10003
3	16-18	10001
	18-20	10003
	20-22	10014

Figure 6: Topic Accuracy

4.3 Topics Accuracy

The purpose of this section is to perform a topic accuracy analysis, i.e., we investigate the ability for topics to recognize the event, or the pattern that they actually describe.

In particular, we employ our system to test the assumption that each day of the week, from Monday to Sunday, differs from the others for a peculiar distribution of crowd dislocations, around city areas and hours of the day. To perform such test, we divided our 12 months dataset in a 9 months training set and a 3 months testing set.

Starting from the training set, we run our system to provide, for each day of the week, the list of topics that better describe them. From these topics, we compose the sequence of visited areas that most likely (from a probabilistic point of view) describes each day. To better clarify this operation, we show in the left part of Figure 6 an example of a typical "Thursday" pattern. The figure illustrates that a typical "Thursday" can be described by topics 10 and 3. For each time slot, topics indicate which is the area (i.e., zip code) people are likely to get together. For example, the figure shows that in a typical "Thursday" from 10 am to 12 pm, a number of people is clustered in zip code 10001. Then from 12 pm to 2 pm, people cluster in zip code 10027, and so on.

Then, we tried to verify whether the extracted topics can actually describe a given day in general. We take the testing set into consideration. We compare each day of the week with the associated topic extracted before. For example, for each Thursday, we take the Thursday topic into account, and we verified whether the movements patterns of that particular day are in line with those described by the topic (i.e., whether the concentration of people in that Thursday, for a given time slot, happens to be in the zip code indicated by the Thursday topic).

In particular, we verified whether the particular day and the associated topic are in line considering 3 time slots (the specific day and the topic have to be in line for at least 6 hours - 3 slots of 2 hours each), up to 6 time slots (the specific day and the topic have to be in line for at least 12 hours - 6 slots of 2 hours each).

Results are illustrated in Figure 6. For a given number of time slots the average accuracy of the topic description of that day is reported. For example, an accuracy of 40% for 6 time slots on Mondays means that, only 40% of Mondays in the testing set is described by the Monday topic for 6 time slots (12 hours - 6 slots of 2 hours each).

It is rather natural to see that the average accuracy is inverse proportional to the number of time slots being considered. The more the extent of the day we are tying to describe the less accurate we are because of the inherent variability in each individual day. Nevertheless, the graph shows a fairly good stability in the patterns being discovered as we are able to describe any given day over 3 time slots with accuracy greater than 90%.

As a side consideration, we should say that the limited employment of geolocalization over social networks posts (only the 3% of generated data is geolocalized) is still limiting the real effectiveness of our system. In fact, despite obtained daily patterns were all different, frequently they share each other multiple common sequences of "time slot - zip code". That general likeness between days of the week, in the case of matching based on few time-slots, leads our system on an higher level of false positive results (around 40% for matching based on 3 time slots).

5. APPLICATIONS

The presented topic-extraction mechanism and, more in general, the study of human-generated patterns could find a natural application in discovering social commonalities among people, as well as to track and predict events that have a strong correlation with a urban environment in terms of space and time.

Marketing and advertisement are natural domains for this kind of technologies [11]. Services built on top of the proposed topic extraction mechanism could group together people with similar interests and usage patterns of public spaces. For example, they could provide customized recommendations on *where to go*, and *what to see*, to people going to visit tourist places for the first time. In this context, patterns of visit could be inferred from past crowd experiences, and once a new user has been labeled with a tourist pattern (e.g. she has visited in sequence a list of popular places described by a topic), an applications could suggest her the next location to be visited, or a restaurant to have dinner [13, 21].

From a complementary perspective, once a sequence of events (or in general human activities) have been recognized as a characteristic topic for the city, we can use this information to make predictions about future events. In particular, once early events indicate the initial fulfillment of a topic, we can predict the next topic evolution. On this basis, two types of social inference could be performed:

- *Direct prediction*. This kind of prediction is related to the fact that a topic is about to happening. In this context, typical examples concern the prediction of urban dynamics as traffic occurrences and crowd displacement. A congestion on some side roads in most of the case will lead to choke up the contiguous main highways. Once a "traffic intensification" pattern is delineated, police could be dispatched in advance to cover the places that the topic predicts as the next ones for traffic diffusion. Similarly, on the happening of mass events (e.g., concerts, fairs, sport events, etc.), anomalous crowd behaviors could be caught and corrected by arranging the flow of people to avoid possible upcoming dangerous situations.
- *Indirect prediction*. This kind of prediction is related to the fact that an expected topic is not happening. This is about detection anomalous and rare event, for example in the case of disaster response. Usually a natural or human driven disaster modifies daily routines bringing people to different and anomalous behaviors. These behaviors (a growing number of phone calls, different car routes, traffic redirection, etc.) differ from typical daily patterns and are easily recognizable. If an emergency would take the ICT networks to the point of collapse, variations in patterns will result even more evident.

These application domains further motivate the proposed topic-extraction mechanism. In addition, since our mechanism can be applied to several location-based social network data, applications tailoring specific networks could be realized. For example, topics coming from photo-sharing sites will support more naturally tourist-oriented applications, while topics coming from Twitter our Foursquare are more in line with understanding the everyday activity of the city.

In conclusion, for the study of human-generated patterns via topic-extraction mechanisms we envision two promising areas of applicability. The first sees the use of such mechanisms as a standalone system for urban management, with the purpose of performing the above described social analysis and studies over an urban environment. The latter concerns the integration of topic-extraction mechanisms in future pervasive services where the view on urban facts and events, and crowd behaviors, will enrich pervasive applications with a further level of context awareness (i.e., "social-awareness").

6. CONCLUSIONS

In this paper, we presented an approach for the automatic extraction of urban patterns and recurrent behaviors from location-based social networks. In particular, we adopt a probabilistic topic model, Latent Dirichlet Allocation (LDA) to identify the routine behaviors with which people move and cluster across the city. Results illustrates that meaningful patterns about city routines can be discovered.

Beside further investigate the proposed application scenarios, and further refine our approach, our future work will proceed in two main directions. On the one hand we will try to apply the proposed approach to other kind of data from social networks. This is important to better evaluate the generality of the proposed approach. In fact cross-validation across multiple data sources can validate the extracted topics also without ground truth information. On the other hand, we will compare the proposed LDA approach with other data mining techniques to extract patterns from geo-localized data. The goal of this study is to determine if combining some different mechanisms could lead to better results and precisions in the recognition of patterns.

Acknowledgements: Work supported by the SAPERE (*Self-Aware Pervasive Service Ecosystems*) project (EU FP7-FET, Contract No. 256874).

7. REFERENCES

[1] C. Bishop. *Pattern Recognition and Machine Learning*. Springer Verlag, 2006.

[2] D. Blei, A. Ng, and M. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3(1):993–1022, 2003.

[3] F. Calabrese, J. Reades, and C. Ratti. Eigenplaces: analysing cities using the space-time structure of the mobile phone network. *IEEE Pervasive Computing*, 9(1):78–84, 2010.

[4] N. Eagle and A. Pentland. Eigenbehaviors: Identifying structure in routine. *Behavioral Ecology and Sociobiology*, 63(7):1057–1066, 2009.

[5] K. Farrahi and D. Gatica-Perez. Discovering routines from large-scale human locations using probabilistic topic models. *ACM Transactions on Intelligent Systems and Technology*, 2(1), 2011.

[6] L. Ferrari and M. Mamei. Discovering daily routines from google latitude with topic models. In *IEEE International Workshop on Context Modeling and Reasoning*, 2011.

[7] F. Girardin, J. Blat, F. Calabrese, F. D. Fiore, and C. Ratti. Digital footprinting: Uncovering tourists with user-generated content. *IEEE Pervasive Computing*, 7(4):36–43, 2008.

[8] R. Heeks. Ict4d 2.0: The nextphase of applying ict for international development. *IEEE Computer*, 41(6):26–33, 2008.

[9] L. Hollenstein and R. Purves. Exploring place through user-generated content: using flickr to describe city cores. In *JOSIS*, number 1, pages 21–48, 2010.

[10] Y. Huang, L. Zhang, and P. Zhang. A framework for mining sequential patterns from spatio-temporal event data sets. *IEEE Transactions on Knowledge and Data Engineering*, 20(4):433–448, 2008.

[11] J. Krumm. Ubiquitous advertising: The killer application for the 21st century. *IEEE Pervasive Computing*, 10(1):66–73, 2011.

[12] D. Leung and S. Newsam. Proximate sensing: Inferring what-is-where from georeferenced photo collections. In *IEEE Conference on Computer Vision and Pattern Recognition*, San Francisco (CA),USA, 2010.

[13] K. W.-T. Leung, D. L. Lee, and W.-C. Lee. Personalized web search with location preferences. In *IEEE International Conference On Data Engineering*, Long Beach (CA),USA, 2010.

[14] M. Mamei, A. Rosi, and F. Zambonelli. Automatic analysis of geotagged photos for intelligent tourist services. In *IEEE Intelligent Environment*, Kuala Lumpur, Malaysia, 2010.

[15] M. Mano and Y. Ishikawa. Anonymizing user location and profile information for privacy-aware mobile services. In *Proceedings of 2nd ACM SIGSPATIAL International Workshop on Location Based Social Networks*, San Jose (CA),USA, 2010.

[16] D. Phung, B. Adams, and S. Venkatesh. Computable social patterns from sparse sensor data. In *First international workshop on Location and the Web*, 2008.

[17] M. Sharifzadeh, C. Shahabi, and C. A. Knoblock. Learning approximate thematic maps from labeled geospatial data. In *Proceedings of International Workshop on Next Generation Geospatial Information*, Boston (MA),USA, 2003.

[18] S. Sigg, S. Haseloff, and K. David. An alignment approach for context prediction tasks in ubicomp environments. *IEEE Pervasive Computing*, 9(4):90–97, 2010.

[19] F. Twaroch, C. Jones, and A. Abdelmoty. Acquisition of a vernacular gazetteer from web sources. In *LocWeb*, Beijing, China, 2008.

[20] C. R. Vicente, D. Freni, C. Bettini, and C. S. Jensen. Location-related privacy in geo-social networks. *IEEE Internet Computing*, 15(3):20–27, 2011, to appear.

[21] V. W. Zheng, Y. Zheng, X. Xie, and Q. Yang. Collaborative location and activity recommendations with gps history data. In *Proceedings of 19th International World Wide Web Conference*, Raleigh (NC),USA, 2010.

Towards an online detection of pedestrian flocks in urban canyons by smoothed spatio-temporal clustering of GPS trajectories

Martin Wirz
Wearable Computing Lab.
ETH Zurich
wirzma@ife.ee.ethz.ch

Mikkel Baun Kjærgaard
Wearable Computing Lab.
ETH Zurich
mikkelk@ife.ee.ethz.ch

Sebastian Feese
Wearable Computing Lab.
ETH Zurich
feese@ife.ee.ethz.ch

Pablo Schläpfer
Wearable Computing Lab.
ETH Zurich
schlaepfer@ife.ee.ethz.ch

Daniel Roggen
Wearable Computing Lab.
ETH Zurich
roggen@ife.ee.ethz.ch

Gerhard Tröster
Wearable Computing Lab.
ETH Zurich
troester@ife.ee.ethz.ch

ABSTRACT

Detecting pedestrians moving together through public spaces can provide relevant information for many location-based social applications. In this work we present an online method to detect such pedestrian flocks by spatio-temporal clustering of location trajectories. Compared to prior work, our method provides increased robustness against the influence of noisy and missing GPS data often encountered in urban environments. To assess the performance of the method, we record GPS trajectories from ten subjects walking through a city. The data set contains various flock formations and corresponding ground truth information is available. With this data set, we can evaluate the accuracy of our method to detect flocks. Results show that we can detect flocks and their members with an accuracy of 91.3%. We evaluate the influence of noisy and missing location data on the detection accuracy and show that the introduced filtering heuristics provides increased detection accuracy in such realistic situations.

Keywords

Collective Behavior Sensing, Spatio-temporal Clustering, Flock Detection

1. MOTIVATION

The increasingly availability of positioning and tracking technologies in personal, mobile devices has triggered an explosion in the amount of location data being collected. This data captures our personal behaviors and can give insight into our daily routines [6]. At a larger scale, this data can help understanding interaction patterns of people, give insight into the dynamics of groups [4] and even provide information about how a society functions [24, 19].

ACM LBSN '11, November 1, 2011. Chicago, IL, USA
Copyright ©2011 ACM ISBN 978-1-4503-1033-8/11/11...$10.00

Not only can we nowadays collect data from a large number of people, such data is increasingly available instantaneously thanks to the communication capabilities of mobile phones. Real-time location information is already used in many location based services such as Google Latitude[1]. Having instant access to location information from a massive amount of people might be used for a real-time understanding of the behavior of crowds. Thinking about urban spaces, areas with high crowd densities, the flow of pedestrians through a city or the grouping behavior of people might be inferred. For a correct unveiling of such patterns and to provide situational awareness, not solely the behavior of people individually should be considered but an understanding of the behavior of the collective is required. This calls for methods that can identify relevant behavior patterns of the collective from real-time location information provided by individuals.

One collective behavior pattern which can often be observed among pedestrians moving through urban spaces is the *flocking* behavior. It describes a number of people walking together as a compact, coherent group for some time. Such a group is often referred to as a pedestrian *flock* [3]. Different approaches have been developed and investigated to automatically detect such flocks from location trajectories [9, 23, 14, 17].

Prior work has focused on animals and vehicles in open space conditions or evaluated the methods on artificial data sets. Evaluating the performance of flock methods under the influence of noisy or even missing data samples as encountered in urban environments and extending these methods to provide robustness against these influences has not been addressed so far. The problem is that in urban spaces like cities where high-rise buildings are blocking the view to the sky (so called urban canyons), GPS-provided location information often suffers from inaccuracy or sporadically missing data [16]. Hence, understanding this influence and providing robustness against noisy and missing data becomes important.

[1]http://latitude.google.com

In this work, we go beyond state-of-the-art approaches. We i) present an online flock detection method that provides increased robustness against the influence of noisy and missing GPS data obtained from mobile phones and, ii) evaluate the flock detection accuracy under the influence of noisy, unreliable and missing data samples. For this, we collect GPS trajectories from ten subjects walking around a city. This data set contains various flock formations and corresponding ground truth information is available.

The reminder of the paper is organized as follows: We start by introducing a formal notation of a pedestrian flock and present a summary of related work. We continue by describing the flock detection algorithm in detail. Next, we introduce the data set. We then present the results from our evaluation before we conclude our work at the end.

2. THE NOTION OF FLOCK

In this section, we give a formal description of a pedestrian flock as used throughout this work. The description is closely related to the problem statement given by Kalnis et al. [14] which defines *moving clusters*. Our notion extends on this definition to better capture the notion of a pedestrian flock.

Similar to Kalnis et al., we assume a set of entities $E = \{e_1, e_2, \dots e_n\}$ that move in space. Each entity e_i possesses a trajectory $T_i = \{T_i^1, T_i^2, \dots T_i^m\}$ that comprises of a sequence of consecutive time-space tuples $T_i^j = (t_k, L_i^{t_k})$. Each of these tuples T_i^j consists of location information $L_i^{t_k}$ and an associated time stamp t_i. We assume a periodic sampling rate and synchronized trajectories. In contrast to Kalnis et al., however, location data can be missing from an entity's trajectory at any given time. Kalnis et al. introduce the terminology *snapshot* $S(t_k) = \{e_i \in E : L_i^{t_k} \neq \emptyset\}$, as the set of entities and their locations at time t_k. For each snapshot $S(t_k)$, spatially collocated entities can be detected to infer groups of entities which are close to each other. C_i is a cluster if all entities of C_i meet the clustering algorithm's specific constraints. The outcome is a set of clusters at a given time $\Gamma(t_k) = \{C_1^{t_k}, C_2^{t_k}, \dots C_m^{t_k}\}$.

With this, the definition of moving clusters is given by:

Definition 1. *Let $G = \{C_\alpha^{t_1}, C_\alpha^{t_2}, \dots C_\alpha^{t_p}\}$ be a sequence of clusters such that for each i $(1 \leq i < p)$, $t_{i+1} - t_i \leq \delta t$ and $\alpha \in \Gamma(t_i)$. Then, G is a moving cluster, with respect to an integrity threshold θ $(0 < \theta \leq 1)$, if the similarity function f fulfills the condition $f\left(C_\alpha^{t_i}, C_\alpha^{t_{i+1}}\right) \geq \theta, \forall i : 1 \leq i < k$*

δt is application specific and limits the maximal time interval between the re-occurrence of a clusters in the sequence. f is a function that describes how similar two clusters are with respect to each other and θ is a threshold above with two cluster are considered to be similar enough to be the same. With this definition of *moving cluster*, we can formulate our definition of a *pedestrian flock* that is used in this work.

Definition 2. *A pedestrian flock F is a moving cluster that exists for the duration $t \geq \tau$ and consists of at least $n \geq \nu$ entities where τ and ν are application specific.*

3. RELATED WORK

There is an increasing amount of literature focusing on describing, sensing and detecting pedestrian collective behavior patterns. Dodge et al. [3] proposed a conceptual framework for classifying movement patterns of moving objects (such as pedestrians) by consolidating categorizations provided by other researchers. The proposed framework follows a top-down approach. In contrast, Wirz et al. present in [25] a bottom-up approach to describe collective behavior patterns of pedestrians. This classification is inspired by the field of collective robotics [20] where robots are brought to exhibit specific collective behaviors by programming simple interaction rules.

To monitor and infer collective behavior patterns from pedestrians, a variety of different sensor modalities have been considered and investigated. Vision based approaches relying on video footage provided by ambient video surveillance systems have been developed to detect small group structures of crowds [7] and to infer pedestrian groups [21]. Such approaches face the challenge of occlusion and require a complete coverage of the space of interest.

Call-data records (CDRs) obtained from mobile network operators have been used to study crowd movements and collective behavior patterns of pedestrians in urban spaces [8, 22]. CDRs have the advantage of automatically being collected by the network operators but have a crude spatial resolution. Mobile phones themselves are becoming location aware and can communicate this information to a server [2]. This enables the collection of users' movement trajectories in a resolution and at a scale that was not possible until recently and allows for detecting the flocking pattern we are aiming to uncover.

In this work, we focus on the detection of the flocking pattern of pedestrians. Related work can be found in the field of mining moving object databases and can be classified into research on clustering moving objects [11, 14, 18], detecting convoys from trajectories [12, 13] and querying flock patterns. Methods for querying flock patterns from trajectory data bases were introduced in [1] and later extended in [9]. It is shown in [9] that discovering the duration of the flock lasting the longest is an NP-hard problem. As a result, approximation algorithms are presented. In [23], Vieira et al. propose a polynomial time solution to the flock querying problem with a predefined time duration. A further characteristics of their method is the online capability. All these querying methods have the peculiarity that they are designed to retrieve flocks of disc-like shape. In contrast, methods relying on spatial clustering algorithms are able to detect flocks of arbitrary shape. We see this as an advantage for detecting realistic pedestrian flocks.

In the field of clustering moving objects, various algorithms such as DBSCAN [5] or DJ-Cluster [26] have been presented to identify dense areas at a given point in time. In [14], Kalnis et al. perform DBSCAN clustering for every time step on location trajectories. Clusters that have been found for two or more consecutive time instances are joined and considered as a moving cluster. A constraint is given that clusters can be joined only if the number of common objects among them are above a predefined threshold. The method pre-

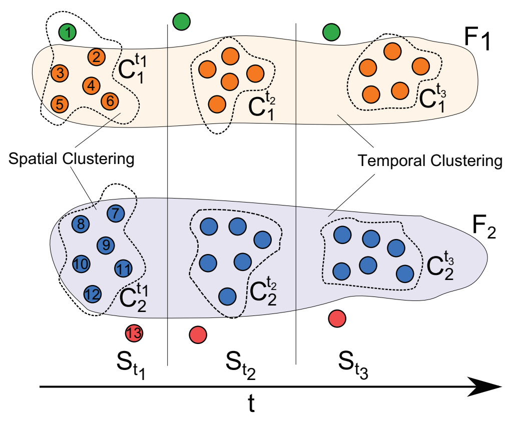


Figure 1: Schematical visualization of the flock detection algorithm.

sented in this work is conceptually similar to the algorithm of [14] on detecting moving clusters. However, we extend the approach by introducing filtering heuristics to address robustness against noisy and missing data.

Recent work has also been looking beyond detecting flocks solely based on location information. Laube et al. [17] have proposed an approach to group entities with similar behaviors by considering the object's motion properties such as speed, change of speed or motion azimut. They present methods to detect several spatio-temporal patterns, including *flock*, *leadership*, *convergence*.

In summary, prior work has primarily focused on the evaluation of the computational complexity of the algorithm. Evaluation is performed with data sets including vehicles and animals moving in open space conditions with limited ground truth information and with synthetic data sets. Therefore, as also identified by [9], methods are needed that address the influence of noisy and missing data with respect to the flock detection accuracy. We propose an algorithm that provide increased robustness against these influences. We recorded a real-world data set with known ground truth with which we can investigate these influences in detail.

4. FLOCK DETECTION ALGORITHM

The basic concept behind our algorithm is a clustering in space and time of a set of location trajectories. A schematical illustration of the flock detection principle is presented in Figure 1. First, at a given time step, the corresponding snapshot is obtained and the entities are spatially clustered based on their location information. In Figure 1, two clusters are detected at each time step. These clusters represent spatially collocated groups of entities. Flocks, however, are groups of entities that stay together over time. Hence, the algorithm then performs a temporal clustering: Clusters that exist for several successive time steps are combined into flocks by respecting a set of rules which are outlined in this section. As a result, two flocks are detected in the example presented in Figure 1.

4.1 Step 1: Spatial Clustering

At each time step, the clustering algorithm uses location information (longitude and latitude) from all entities of the corresponding snapshot to group collocated entities together using an adaption of the DensityJoin-Cluster (DJ-Cluster) algorithm [26]. The basic idea of DJ-Cluster is to have a set of points p in a 2D-plane. DJ-Cluster first calculates the neighborhood $N(p)$ for each point p. The neighborhood $N(p)$ consists of all points around p within a threshold radius R. After a neighborhood $N(p)$ is created, the DJ-Cluster algorithm checks if $N(p)$ consists of at least minPts points. If so, it seeks to *density join* it to existing clusters. A neighborhood is *density joinable* to a cluster if they share at least one common point. Our implementation computes the neighborhood $N(p)$ for each entity with a valid location information p and tries to density-join it to an existing cluster. If no cluster can be found, a new cluster is created. The advantage of spatial clustering using DJ-Cluster as opposed to e.g. [9, 23] is that clusters of arbitrary shapes can be detected. The pseudo-code description of the algorithm is given in Algorithm 1 and Algorithm 2. The outcome of this step is a set is a set of clusters containing collocated entities.

Algorithm 1 spatialClustering(timestamp) <minPts>

```
1:  $clusters := \emptyset$
2:  $S :=$ all points of current timestep
3:  for all points $p$ of $S$ do
4:      $N :=$ neighborhood$(p, S)$
5:      if $|N| \geq minPts$ then
6:          if $N$ is density joinable existing clusters then
7:              merge $N$ and all the density-joinable clusters
8:          else
9:              $clusters \leftarrow N$  # Add $N$ as a new cluster
10:         end if
11:     end if
12: end for
```

Algorithm 2 neighborhood(p, S) <R>

```
1: $N := \emptyset$
2: $N := p$
3: for points $q \neq p$ in $S$ do
4:     d = distance$(p, q)$
5:     if d $\leq$ R then
6:         $N := N \cup q$
7:     end if
8: end for
9: return $N$
```

4.2 Step 2: Temporal Clustering

Flocks are now formed by detecting spatial clusters that recur over multiple time steps. We are aiming at an online detection of pedestrian flocks. This imposes certain restrictions on the design of our algorithm: At a given point in time, our method has to detect the flocks without possessing knowledge of future steps and only based on information from the present and the past. Additionally, the method is not allowed to alter results from previous steps. We define three states a flock can be in at any given time t_i:

- *Active flock*: A flock that is present at t_i and existed for a time duration of at least τ.

- *Potential flock*: A flock that is present at t_i and existed for a time duration of less than τ.
- *Dissolved flock*: A flock that occurred previously and has existed for a time duration of at least τ but is not present anymore at t_i.

The algorithm to detect flocks by combining spatial clusters is an adaptation of the time-based clustering algorithm proposed by Kang et al. [15]. We adapt this approach in the following way: The algorithm holds a list of all *active flocks* and *potential flocks*. At each time step, spatial clustering is performed. Then, the algorithm assigns all entities of a spatial cluster to that flock (potential or active) that is most similar to the spatial cluster. If no suitable flock can be found, i.e. the similarity between the spatial cluster and all flocks (potential or active) is smaller than a given threshold, the members of the spatial cluster become members of a new *potential flock*. We use the Jaccard measure [10] to calculate the similarity between a cluster and flocks with respect to their members:

$$\text{Similarity} = \frac{|\text{Clustermembers}| \cap |\text{Flockmembers}|}{|\text{Clustermembers}| \cup |\text{Flockmembers}|}$$

The Jaccard measures can be used to determine the similarity of of two sets and is the same measure that has been used in [14] to infer moving cluster.

As soon as a *potential flock* has persisted for longer than τ, it is promoted and becomes an *active flock*. However, if no new clusters are added to a *potential flock* for longer than δt, this *potential flock* will be discarded and its members will be marked as unassigned for the period where they belonged to that *potential flock*.

In contrast, if no new cluster is added to an *active flock* for longer than δt, the *active flock* is recognized as a *dissolved flock* from this point onwards no new clusters can be appended anymore. The pseudo-code description of the temporal clustering is given in Algorithm 3.

Algorithm 3 Flocking Algorithm

```
 1: $act\_flocks := \emptyset$          # initialize set of active flocks
 2: $pot\_flocks := \emptyset$          # initialize set of potential flocks
 3: $dis\_flocks := \emptyset$          # initialize set of dissolved flocks
 4: for all timesteps do
 5:   for $pot\_flock$ in $pot\_flocks$ do
 6:     if $pot\_flock.duration() \geq \tau$ then
 7:       $act\_flocks := act\_flocks \cup pot\_flock$
 8:       $pot\_flocks := pot\_flocks \setminus pot\_flock$
 9:     end if
10:     if $pot\_flock.lastseen() \geq \delta t$ then
11:       $pot\_flocks := pot\_flocks \setminus pot\_flock$
12:     end if
13:   end for
14:   for $act\_flocks$ in $act\_flocks$ do
15:     if $act\_flock.lastseen() \geq \delta t$ then
16:       $act\_flocks := act\_flocks \setminus act\_flock$
17:       $dis\_flocks := dis\_flocks \cup act\_flock$
18:     end if
19:   end for
20:   $clusters$ := spatialClustering(timestamp)
21:   for $cluster$ in $clusters$ do
22:     compute similarity to each $flock$ in ($act\_flocks \cup$
        $pot\_flocks$)
23:     assign cluster to $flock$ in ($act\_flocks \cup pot\_flocks$)
        with highest similarity
24:   end for
25: end for
```

4.3 Filtering Heuristics

Location information provided by mobile devices may be unreliable, inaccurate or even missing for some period of time. Our method should not lose track of a flock in such situations and should provide increased robustness against such influences.

As we aim for an online detection of flocks, our method has to fulfill the causality principle and hence only information from the past and present can be considered. A simple interpolation of the trajectory data between the current and the last seen location information is no option in our method as this would require us to alter information from the past. Hence, we implement the following filtering heuristics to address this issue.

If a location estimation sample is missing or if the accuracy falls below a threshold $th_{LocationEstimAcc}$, instead of not assigning the entity to a cluster for this time step, it gets assigned to the same flock as its peers with which it was clustered previously. If no location information from an entity is available for more then τ' time steps, the entity gets disregarded from this time onwards until location information is again available. We use the same value for τ' as for τ.

Further, we do not ask a flock to be recognized in two successive time steps but within δt. With this, a flock is not immediately assigned as a *dissolved flock* if the cluster does not get detect at a clustering stage or if the similarity between successive time steps falls under the threshold.

5. DATA SET

To be able to perform an evaluation of the algorithm's flock detection accuracy, we conducted an experiment in an urban environment to obtain a data set. The experiment was conducted according to a predefined script and monitored with video cameras and manual annotation in order to obtain a ground truth against which the algorithm's performance can be evaluated.

The experiment took place in the city center of Zurich with 13 participants. All of them were equipped with Google Nexus One smart phones running a dedicated sensor logging software to record GPS location information (latitude, longitude and positioning accuracy).

During the experiment, three participants were selected to be group leaders and the others were divided into three groups of size 3, 3 and 4. Each group was guided by a previously instructed leader. The groups followed their leaders

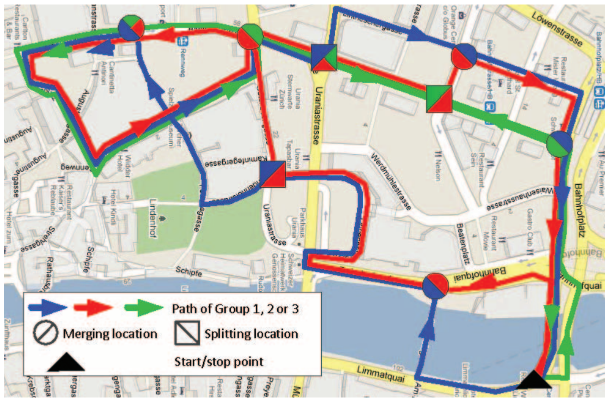


Figure 2: The map indicates paths of the three groups including merging and splitting points.

along predefined routes which were chosen such that the three groups would meet, merge and split several times (5 merging and 4 splitting points). In order to obtain recordings of natural behavior, we did not instruct the subjects on how to move or behave and only told them to put the mobile phones into their trouser pockets and to always follow and stay close to their leader (where 'close' was not further specified but was observed to be around $2m$). The timing of important events like an encounter of two groups was manually annotated by the respective group leaders. Figure 2 shows the three routes on a map highlighting locations where the groups merged and split. For evaluation, we only considered the data from ten participants, ignoring the data from the group leaders.

GPS locations were sampled every 5 seconds and the experiment lasted for 32 minutes which resulted in a data set of $3'297$ valid location points. Together with the location information, the mobile phones provided an accuracy of the location estimation. This accuracy is given in meters and most often defines the radius of 95% confidence circle[2]. In our data set, we observed that during 13.5% of the time, the device was unable to obtain GPS location information which was flagged as 'missing data'. The histogram together with the cumulative distribution of the device-estimated GPS accuracy shown in Figure 3 reveal that approximately 95% of the samples have an accuracy better than 24 meters. For further processing, the data streams from all devices were synchronized so that all devices share the same time stamps.

6. EVALUATION

In this section, we are going to evaluate various aspects of our algorithms. We first investigate the flock detection accuracy on our data set before we analyze the influence of noisy and missing data on the detection accuracy.

6.1 Flock Detection Accuracy

By evaluating the flock detection accuracy, we are interested in two metrics: i) How accurately is the algorithm able to infer the correct number of flocks present at any point in time, and ii) how well can the individual subjects be assigned to the correct flock. We first investigate the influence of R and θ on these metrics. Throughout this analysis, the following parameter set was kept fix: $\tau = 20s$, $\delta t = 10s$

[2]Could not be verified for Android devices but indications are given.

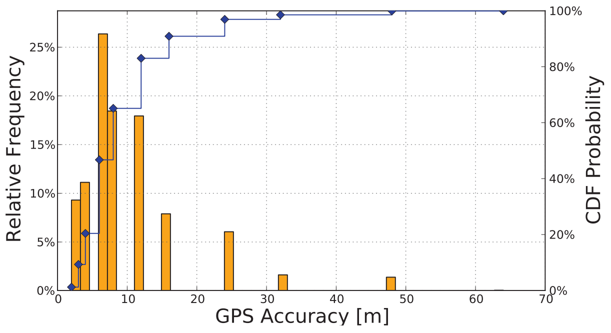


Figure 3: Histogram and cumulative distribution function of GPS accuracy over the complete data set. The accuracy value defines the radius of 95% confidence circle.

6.1.1 *Detecting the Number of Flocks*

Of interest is how accurately the algorithm is able to infer the correct number of flocks present at a point in time. To evaluate this, at each time step, the number of flocks identified by the algorithm is compared to the ground truth.

The Number of Flocks Detection Accuracy (NFDA) is the number of time steps with a correct estimation of the flock count divided by all time steps. Figure 4 shows the obtained accuracies for varying R and θ. We achieved the highest NFDA value for $R = 21$ and $\theta = 0.4$. With these parameters, in 78.5% of the time, the correct number of flocks was detected (65.9% without missing data smoothing).

6.1.2 *Flock membership assignment*

We examine how well the individual subjects are assigned to the correct flock. To obtain a meaningful evidence, we calculate the following measure: At each time step t, we identify the number of subjects that have been assigned to the correct flock, $s_{cor}(t)$, which is divided by the total number of subjects s_{tot} to obtain $a(t)$, the fraction of correctly identified subjects.

This allows to compute the Flock Assignment Accuracy FAA by averaging $a(t)$ over all time steps T:

$$\text{FAA} = \frac{\sum_t a(t)}{T} = \frac{\sum_t s_{cor}(t)}{s_{tot} \cdot T}$$

Figure 4 shows the obtained FAA values for varying R and θ. With $R = 21$ and $\theta = 0.4$ we achieved the highest FAA value of of 91.3% (74.0% without missing data smoothing).

6.1.3 *Detailed considerations*

In the following, we set $R = 21m$ and $\theta = 0.4$. The achieved results are summarized in Table 1 and explained in more detail in this section. Figure 5 illustrates the group formations

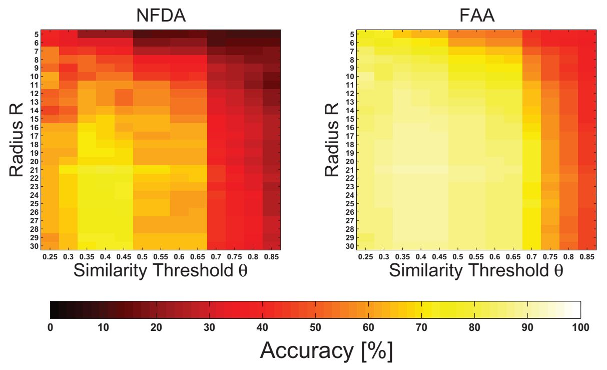


Figure 4: NFDA and FAA values by varying R and θ. Best configuration is achieved with $R = 21m$ and $\theta = 0.4$.

	Detection Accuracy	
	NFDA	FAA
Data Set 1		
filtered	78.5%	91.3%
non-filtered	65.9%	74.0%

Table 1: Summary of classification accuracy results for $R = 21$ and $\theta = 0.4$

over time for the whole duration of the experiment and the achieved flock recognition results. The illustrations show time steps on the horizontal axis and the individual subjects along the vertical axis. The colors represent detected flocks. All the subjects assigned to a flock are colored identically. The black outlines in the figure represent the ground truth information. Subjects should be recognized as being in the same flock, i.e. colored the same, while being within the same black outline. Areas shown in white indicate time steps in which the algorithm considered the subject as not belonging to any flock. We will provide a detailed evaluation and use circled numbers (e.g. ①, ②) to point out instructive occurrences.

There are two reasons for not achieving higher NFDA values: The first problem was the incorrect recognition of one single flock in situations where two flocks were in close proximity for a longer time duration. E.g. in ①, all subjects are recognized as one single flock, although two flocks were present but collocated. The combination of the two flocks into one is caused by the fact that one group was walking directly in front of the other for longer than τ'. In ②, the two flocks were walking the same path on different sides of the street.

The second issue is the problem of clustering range and latency. In the time steps marked with ③, the algorithm assigns all subjects to one single flock before two flocks actually merge. In ④, on the other hand, two flocks are still assigned to the same flock, although the groups already split. The reason for under- and overfill is that the algorithm uses the subject's proximity as a measure for clustering. Right before two groups meet (or right after they split up), they are only few meters apart. Because of this low proximity, the algorithm adds both groups' subjects to one large flock. As investigated previously, a smaller R and a larger τ mitigate these issues but resulted in a lower overall accuracy due to noisy data. There are situation where subjects are considered as belonging to no flock (indicated in white, e.g. at ⑤). This happens when either the individual subjects do not get assigned to a cluster during the spatial clustering phase (e.g. because the location estimation is incorrect) or is considered as an unrelated subject by the filtering heuristic.

6.2 Influence of noisy or missing GPS information

In our data set, 13.5% of the expected GPS samples were missing. Hence, we are interested in understanding the robustness of our algorithms with respect to the absence of GPS samples. To do so, we additionally removed between 0% and 100% of the GPS samples from our data set and rerun the flock detection algorithm. In the first case, we removed the GPS samples with a random uniform distribution. In the other case, to be truthful to environmental influences encountered in urban canyons, we removed samples with an accuracy value larger than a threshold (i.e. we first removed all samples with an accuracy of $64m$, then of $48m$, etc.). Figure 6 shows the obtained NFDA values and Figure 7 shows FAA values. Besides showing the performance of the algorithm with missing data smoothing filtering heuristic (green cirlces), we also evaluated the situation where subjects with no GPS information are randomly assigned to a flock (blue squares) and where subjects are discarded for the time steps where no location information is present (red triangles). It can be seen that NFDA and FAA drop for all cases the more data is absent. However, the missing data smoothing filtering significantly outperforms the other strategies. The plots also reveal that NFDA and FAA values are lower for corresponding values of missing data in the case of data removal based on accuracy values. This is because not just members of a flock are missing but the whole flock as all members suffer from a similar acuracy value.

7. CONCLUSION

We demonstrate in this work that pedestrians forming flocks in urban spaces can be detected online by means of GPS location information in a robust manner. We evaluated the presented approach with a real-world urban-scale data set. This is a unique characteristic of this work. Past work surveyed here only analyzed algorithmic complexity of their proposed methods, but did not have experimental data comprising ground truth information. Also, the problem of noisy or missing data has not been addressed so far but is a typical characteristic of urban environments. Using of-the-shelf mobile phones for data recording, the correct number of flocks was identified in 78.5% of the time and we achieved an accuracy of 91.3% to detect the correct flock members. Due to missing data smoothing filtering, the method achieves high accuracy even in situations with low GPS-signal quality. We observed that the algorithm suffers from under- and overfill at flock configuration changes. In a future work, we aim to address this issue by incorporating additional behavioral information besides the location of subjects, such as

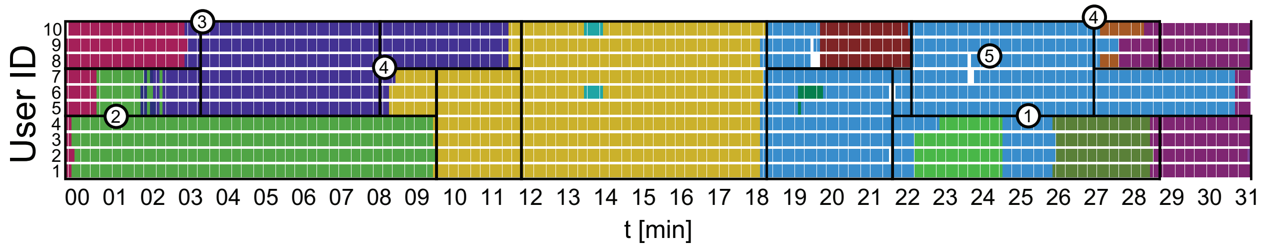
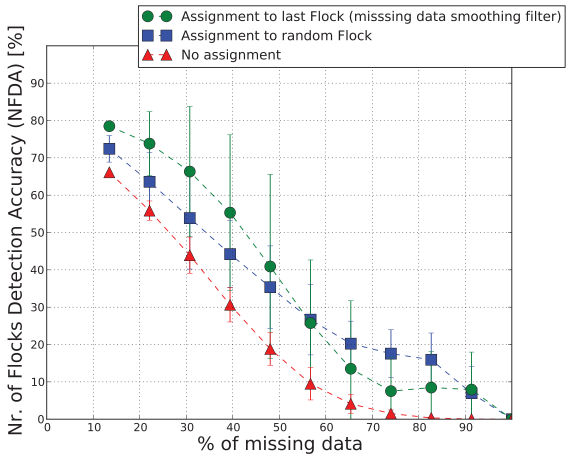
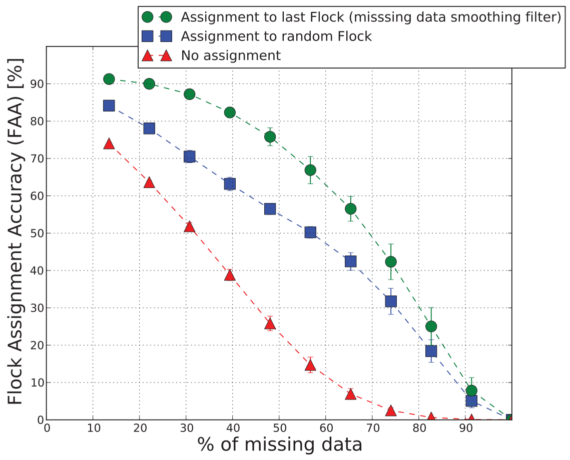


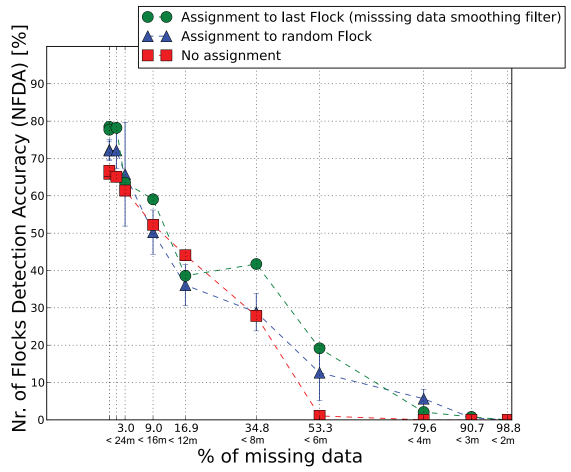
Figure 5: Flock recognition results for the whole duration of the experiment. The colors represent detected flocks and subjects assigned to the same flock are colored the same. Subjects should be recognized as being in the same flock, i.e. colored the same, while being within the same black outline.



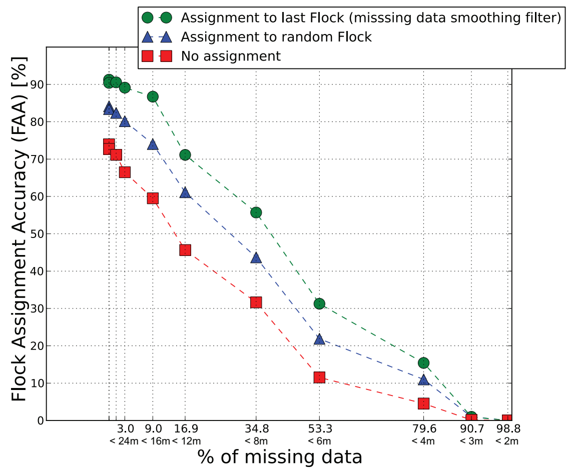
(a) Samples are removed with a random uniform distribution.



(a) Samples are removed with a random uniform distribution.



(b) Samples are removed in a burst-like scheme.



(b) Samples are removed in a burst-like scheme.

Figure 6: Number of Detected Flocks Accuracy (NDFA) by the absence of GPS samples.

Figure 7: Flock Assignment Accuracy (FAA) by the absence of GPS samples

motion attributes. In [17, 25], speed and speed variation was proposed to infer crowd patterns and in particular if people walk in groups/flocks. Additionally, a multi-modal approach considering WiFi scans together with accelerometer and compass information could lead to an increased detection accuracy and could be used at locations with missing GPS information like indoor venues. The experimental methodology used in this work allows for a detailed characterization of the method, yet we find this not to adversely influence the behavior of the participants. However, in a next step, performance and robustness should be assessed on datasets comprising unscripted flock formations in daily situations. With the obtained results that pedestrian flocks can be detected using GPS trajectories even in challenging situations such as urban canyons. Together with other elements (e.g. clogging, queuing) this is an enabling technology supporting situational awareness of collective pedestrian behavior many future location based social services can profit from.

8. ACKNOWLEDGMENTS

This work is supported under the FP7 ICT Future Enabling Technologies Programme under grant agreement No 231288 (SOCIONICAL). We thank Google for providing the phones.

9. ADDITIONAL AUTHORS

10. REFERENCES

[1] M. Benkert, J. Gudmundsson, F. Hübner, and T. Wolle. Reporting flock patterns. *Algorithms–ESA 2006*, pages 660–671, 2006.

[2] T. Choudhury, G. Borriello, S. Consolvo, et al. The mobile sensing platform: An embedded activity recognition system. *IEEE Pervasive Computing*, 7:32–41, 2008.

[3] S. Dodge, R. Weibel, and A. Lautenschütz. Towards a taxonomy of movement patterns. *Information Visualization*, 7(3):240–252, 2008.

[4] N. Eagle and A. (Sandy) Pentland. Reality mining: sensing complex social systems. *Personal Ubiquitous Comput.*, 10:255–268, March 2006.

[5] M. Ester, H. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the 2nd International Conference on Knowledge Discovery and Data mining*, volume 1996, pages 226–231. Portland: AAAI Press, 1996.

[6] K. Farrahi and D. Gatica-Perez. What did you do today?: discovering daily routines from large-scale mobile data. In *Proceeding of the 16th ACM international conference on Multimedia*, pages 849–852. ACM, 2008.

[7] W. Ge, R. Collins, and B. Ruback. Automatically detecting the small group structure of a crowd. In *Applications of Computer Vision (WACV), 2009 Workshop on*, pages 1 –8, dec. 2009.

[8] F. Girardin, A. Vaccari, A. Gerber, and C. Ratti. Quantifying urban attractiveness from the distribution and density of digital footprints. *Journal of Spatial Data Infrastructure Research*, 4:175–200, 2009.

[9] J. Gudmundsson and M. van Kreveld. Computing longest duration flocks in trajectory data. In *Proceedings of the 14th annual ACM international symposium on Advances in geographic information systems*, GIS '06, pages 35–42, New York, NY, USA, 2006. ACM.

[10] P. Jaccard. The distribution of the flora in the alpine zone. *New Phytologist*, 11(2):37–50, 1912.

[11] C. Jensen, D. Lin, and B. Ooi. Continuous clustering of moving objects. *IEEE transactions on knowledge and data engineering*, pages 1161–1174, 2007.

[12] H. Jeung, H. T. Shen, and X. Zhou. Convoy queries in spatio-temporal databases. *Data Engineering, International Conference on*, 0:1457–1459, 2008.

[13] H. Jeung, M. Yiu, X. Zhou, C. Jensen, and H. Shen. Discovery of convoys in trajectory databases. *Proceedings of the VLDB Endowment*, 1(1):1068–1080, 2008.

[14] P. Kalnis, N. Mamoulis, and S. Bakiras. On discovering moving clusters in spatio-temporal data. *Advances in Spatial and Temporal Databases*, pages 364–381, 2005.

[15] J. Kang, W. Welbourne, B. Stewart, and G. Borriello. Extracting places from traces of locations. *ACM SIGMOBILE Mobile Computing and Communications Review*, 9(3):58–68, 2005.

[16] M. Kjærgaard, H. Blunck, T. Godsk, T. Toftkjær, D. Christensen, and K. Grønbæk. Indoor positioning using gps revisited. In P. Floréen, A. Krüger, and M. Spasojevic, editors, *Pervasive Computing*, volume 6030 of *Lecture Notes in Computer Science*, pages 38–56. Springer Berlin / Heidelberg, 2010.

[17] P. Laube, M. Kreveld, and S. Imfeld. Finding remo: Detecting relative motion patterns in geospatial lifelines. In *Developments in Spatial Data Handling*, pages 201–215. Springer Berlin Heidelberg, 2005.

[18] Y. Li, J. Han, and J. Yang. Clustering moving objects. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 617–622. ACM, 2004.

[19] B. Liu, F. Calabrese, A. Vaccari, C. Ratti, et al. Towards the SocioScope: an Information System for the the Study of Social Dynamics through Digital Traces. In *Int. Conference on Advances in Geographic Information Systems*, volume 9, pages 52–61, 2009.

[20] M. Mataric. Designing emergent behaviors: From local interactions to collective intelligence. In *From Animals to Animats 2: Int. Conf. on Simulation of Adaptive Behavior*, pages 423–441, 1993.

[21] S. Pellegrini, A. Ess, and L. Van Gool. Improving data association by joint modeling of pedestrian trajectories and groupings. *Computer Vision–ECCV*, 2010.

[22] A. Vaccari, L. Liu, A. Biderman, C. Ratti, F. Pereira, J. Oliveirinha, and A. Gerber. A holistic framework for the study of urban traces and the profiling of urban processes and dynamics. In *Int. Conference on Intelligent Transportation Systems*. IEEE, 2009.

[23] M. Vieira, P. Bakalov, and V. Tsotras. On-line discovery of flock patterns in spatio-temporal data. In *Int. Conference on Advances in Geographic Information Systems*. ACM, 2009.

[24] A. Wesolowski and N. Eagle. Parameterizing the dynamics of slums. In *AAAI Spring Symposium on Artificial Intelligence for Development (AI-D)*, 2010.

[25] M. Wirz, D. Roggen, and G. Troster. Decentralized detection of group formations from wearable acceleration sensors. *Computational Science and Engineering, IEEE International Conference on*, 4:952–959, 2009.

[26] C. Zhou, D. Frankowski, P. Ludford, S. Shekhar, and L. Terveen. Discovering personal gazetteers: an interactive clustering approach. In *Int. workshop on Geographic information systems*. ACM, 2004.

Identification of Live News Events using Twitter [*]

Alan Jackoway
University of Maryland
jackoway@umd.edu

Hanan Samet
University of Maryland
hjs@cs.umd.edu

Jagan Sankaranarayanan
University of Maryland
jagan@cs.umd.edu

ABSTRACT

Twitter presents a source of information that cannot easily be obtained anywhere else. However, though many posts on Twitter reveal up-to-the-minute information about events in the world or interesting sentiments, far more posts are of no interest to the general audience. A method to determine which Twitter users are posting reliable information and which posts are interesting is presented. Using this information a search through a large, online news corpus is conducted to discover future events before they occur along with information about the location of the event. These events can be identified with a high degree of accuracy by verifying that an event found in one news article is found in other similar news articles, since any event interesting to a general audience will likely have more than one news story written about it. Twitter posts near the time of the event can then be identified as interesting if they match the event in terms of keywords or location. This method enables the discovery of interesting posts about current and future events and helps in the identification of reliable users.

Categories and Subject Descriptors

H.3 [**Research Paper**]: Systems and Services

General Terms

Algorithms

Keywords

tense identification, future tense, event detection, geotagging, Twitter

1. INTRODUCTION

Though Twitter presents a massive source of information on current events, it is an incredibly noisy medium, so automatically selecting which posts (i.e., Tweets) are reliable and interesting for a general audience can be very difficult. Many users post information that is only interesting to their individual group of followers, post spam, or post incorrect information. Furthermore, some users could be very reliable on one topic while being completely unreliable about another topic. Our goal is to help identify which users post reliable Tweets and which Tweets are interesting.

We accomplish this by identifying and predicting future events as well as where they will be happening. Events are discovered using a constantly updating corpus of news articles, called NewsStand and described fully in [28]. After news articles have been downloaded and grouped into clusters based on their story, events can be discovered as described in Section 5. Using information about the events, we extract relevant and reliable information from posts on Twitter. This is aided by making use of a large and constantly updated corpus of news articles. The prediction capability is achieved by using the future events to define new features (such as keywords and location) related to the events which can be subsequently extracted so that Twitter traffic (i.e., postings) about the events can be identified. Our work is also useful in identifying which Twitter postings are associated with real time events. Moreover, it aids us in identifying Twitter posters who are posting on real events and, of equal importance, which posters can be deemed reliable on a particular issue. We can also begin to determine a geographic region with which a user is familiar by finding out where the events that a user posts about are occurring. The more reliable posts about events in a certain area, the more likely future posts about that area are to be reliable. This is all in the spirit that there are millions of people posting on Twitter, and as we cannot follow all of them, we would like to have some measure of their reliability and credibility. This work is a temporal extension of our prior work [23] which attempted to determine the spatial locations associated with posts on Twitter despite the absence of location information in the Tweets.

The rest of this paper is organized as follows. Section 2 provides a brief introduction to Twitter, which is expanded with regard to news in Section 3, while Section 4 reviews the geotagging process. Section 5 describes our tense identification methods. Section 6 contains results of an experimental evaluation of these methods, while concluding remarks are drawn in Section 7.

2. TWITTER OVERVIEW

Twitter is a social networking website that has expanded greatly over the last few years. Twitter users post messages (termed *Tweets*), which can be seen by all users who choose to "follow" them. One user can become another user's "follower," in which case everything posted by the second user will be viewable by the first user (though there is a way to block users from reading posts if desired). Each Tweet is at most 140 characters, allowing Tweets to be posted via text message as well as using the Twitter website. As of April 2010, Twitter had 105 million registered users, with 300,000

[*]This work was supported in part by the National Science Foundation under Grants IIS-10-18475, IIS-09-48548, IIS-08-12377, CCF-08-30618, and IIS-07-13501.

more registering each day. These users were posting about 55 million Tweets per day [30]. Twitter's enormous popularity could be due to its simplicity—by limiting users to 140 character-long messages, it means that posting or reading a Tweet is guaranteed to not take very long. Additionally, Twitter is available on a number of platforms, including the website, SMS messaging, and a multitude of clients for both phones and computers. As a social network, Twitter's popularity increases because it encourages users to follow many people and get as many people as possible to follow and read their Tweets.

Because of the enormous number of Tweets posted per day on a huge variety of topics, Twitter can be an ideal source for finding out up-to-date information on events. However, there are a few major obstacles to extracting this information from Twitter. First, because of the character limit on posts, Tweets do not have as much information as a post on a blog or website, which can make it difficult to assess their value. Posts with links (which ordinarily would provide quite a bit of information because of the hyperlink) are made more difficult to analyze because they are almost always "shortened" by passing them through a forwarding service specifically designed to map real URLs to 10-20 character links. Another confounding factor in finding useful information on Twitter is the prevalence of spam messages. Many users posting spam attempt to make their messages look as legitimate and interesting as possible to catch unsuspecting eyes. Twitter has taken major steps to reduce spam with some success. In February 2009, spam messages accounted for about 5.5% of Tweets. In the summer of 2009, spam neared 10%, but as of February 2010, spam has been reduced to just over 1% of Twitter's posts [4].

The most pressing problem with finding interesting and reliable Tweets is the volume of Tweets that are not interesting to a general audience. Twitter does not have a specific topic or goal like some websites, but rather it encourages people to post about anything they like. As a result, many Tweets are brief thoughts of a user, often interesting only to people who know that user personally. Though this is not a problem for Twitter—as long as the people following a user are interested in what he or she tweets, Nevertheless, Twitter remains useful as a social network, although it can be quite troubling for those trying to extract Tweets that would be interesting to an audience other than a users' followers. This paper provides a way to identify which users are reliable by using the fact that we can verify their posting on particular events on account of having discovered the actual events independently by looking at news articles from NewsStand [28], a constantly updated corpus of online news, similar to Google News. By verifying users based on their comments on events that are known to be happening in the world, we can determine what users are knowledgeable about certain events or geographic regions.

3. BREAKING NEWS USING TWITTER

As we pointed out, Twitter is an electronic medium that allows a large user populace to communicate with each other simultaneously. We recently demonstrated a system called TwitterStand [23] that uses Tweets posted by users in Twitter to capture breaking news faster than conventional news aggregators (e.g., Google News, Yahoo! news). In fact, a constant criticism of conventional news aggregators is that they are slow in responding to breaking news [6]. When important news occurs, it takes quite a while for news aggregators to prominently display it. This shortcoming of news aggregators is not surprising as they themselves do not produce the content, but rather simply crawl and index news sources that produce them. The news sources first have to produce the news content, which usually passes through an editorial pipeline after which they are crawled and indexed by the news aggregators. This process takes time, which means that there is an unavoidable time lag between when the event occurs and when the news aggregators can display them. Also, most news aggregators only show a few important stories, where importance is usually assigned based on how many different news sources contribute to (i.e., report on) a certain *news topic*, which is obtained by *clustering* news articles so that similar articles from different sources are associated with the same news topic. This means that the news topic should contain enough news articles before it is prominently displayed, which further adds to this perceived time lag with conventional news aggregators.

A news aggregator using Tweets (e.g., TwitterStand) attempts to capture late breaking news entirely using Tweets written by the users of Twitter. The result is analogous to a distributed news wire service, where the identities of the contributors/reporters (i.e., analogous to news sources) are not known in advance and there may be many of them. Tweets are not sent according to a schedule and there are no reporters being assigned to cover stories. The challenge becomes how to separate the news Tweets from non-news Tweets, which is a hard problem. What makes Twitter attractive for capturing breaking news is that there is very little lag between the time that an event happens, or is first reported in the news media, and the time at which it is the subject of a posting on Twitter. The data is "pushed" by the content providers (i.e., people who send Tweets) and is delivered nearly instantaneously to the content consumers (i.e., people who receive Tweets and TwitterStand). In contrast, conventional news aggregators must constantly poll the content providers for updates with web spiders, which means that there could be a significant time lag between the time the news is published and is first picked up by the news aggregators. Thus we see that from the point of view of speed (i.e., the ability to generate a scoop), Twitter provides news aggregators such as TwitterStand an edge over conventional news aggregators such as Google News.

However, there are still issues with news aggregators that rely on Tweets for news gathering. In particular, such aggregators must deal with the inherent unreliability of the information carried by Tweets. As Tweets undergo no editorial control there is very little one can do to assure reliability of the information that they carry. Moreover, there is very little in the way of holding users accountable for publishing misinformation. In fact, Twitter users often indulge in misinformation campaigns. For example, consider the rumor campaign on Twitter reporting on the death of the actor Johnny Depp in a car accident [15] in January 2010. This means that what we perceive as news Tweets in Twitter could very well be part of a deliberate rumor campaign. So, there is a need to examine ways to instill trust and reliability in the news gathered by the way of Tweets, which is the goal of this paper. Please note that this issue of trust and reliability does not still diminish the utility of systems like TwitterStand. Even with these problems, the basic intent of the majority of Twitter users is not to indulge in misinformation campaigns, but one always has to be wary of the unreliability of this medium.

In general, assuring trust and reliability of news obtained from Twitter is a hard problem. A simple idea that we explore in this paper is to combine the reliability of conventional news media with the swiftness of Twitter for certain kinds of breaking news. In particular, we are interested in exploring a synergy between conventional news sources and Twitter when it comes to events that are prescheduled. In other words, we are not interested in unexpected events (e.g., Earthquake in Haiti), but in events that we already know are going to take place in the future, in a certain time window. In this regard, our ability to recognize future events in conventional news (as described in section 5) comes in handy. Given that we know that a certain news event is going to occur at a certain time, can we now use Tweets posted in Twitter to provide live updates as the event is progressing. For example, considering that we know from processing conventional news that Superbowl 2011 will happen on Feb 6, 2011, can we use Twitter to post live updates as the game is progressing? The proposed system combines the reliability of conventional news media with the speed as Twitter. The assurance that the Superbowl will happen on Feb 6, 2011 is obtained from conventional news sources, while the swiftness of Tweets is used to provide real time updates as the game is taking place.

The setup of our system is as follows. We constantly process conventional news sources so that we can pick out any news topic n containing mentions of a future event. This is done under the auspices of our earlier system called NewsStand [28], which is a system that brings news reading experience to a map interface. Note that the extraction of future events from news articles is described in Section 5. A news topic n containing mention of a future event is denoted by its feature vector $\boldsymbol{TFV_n}$, which is the set of words or phrases that can be used to describe the news topic. The feature vectors are obtained using the TF-IDF [20] measure. Furthermore, each news topic n is associated with a time window $[t_s, t_e]$, which is the time period during which an event related to n will occur. Finally, the geographic region n_g serving as the focus of n is obtained through geotagging [11, 28] of the news articles associated with n. Furthermore, assuming that the current time is t, let N denote a set of news topics such that the time window associated with the events in N contains t (a "time window," defined by a start and end time, is just a period of time). In other words, N denotes the set of news topics that are happening at moment t. We refer to N as the set of active news topics, which is constantly updated as news topics are added and removed from it. Let F be the set of feature vector terms obtained by pooling the feature vectors of all the news topics in N. Now, the set of keywords F forms the input of the *track* API of Twitter, which takes up to 200,000 keywords as input, and obtains Tweets in Twitter containing one or more of the keywords in F.

Our input is a stream of Tweets from Twitter track API containing one or more keywords in common with one or more news topics in N. As most of the Tweets obtained using this method from Twitter are not related to news, we first apply a coarse filtering on incoming Tweets, which classifies incoming tweets as either *junk* or *news*, where junk tweets have a good chance of not being related to the news and hence, are discarded, while the news tweets have a good chance of being related to news. Note that our goal is not to completely get rid of noise, which may not even be possible given the uncertain boundary between news and noise, but instead to find a way of discarding tweets that clearly cannot be news. So, the goal here is to throw away as many tweets as possible without losing many news tweets. Note that we are not really worried about misclassifying a small percentage of *news* Tweets as junk and not even processing them. This is due to the incredibly high amounts of Tweet data that is available to us from Twitter. However, our aim is to ensure that we provide a very good quality output, one that does not contain too many noise Tweets. For the purpose of separating junk Tweets from news Tweets, we use a naive Bayes classifier [16] that is trained on a training corpus of tweets that have already been marked as either news or junk. Interested readers are referred to [23] for a brief review of the classifier used to separate out news Tweets. At this stage, the Tweets, which are still noisy, have a higher percentage of news Tweets, which is good enough for our purposes.

We now have to associate an input Tweet t from Twitter with a news topic n in N, or possibly discard it. For this purpose, we maintain an active set N of news topics, such that each news topic n in N is denoted by its *feature vector* $\boldsymbol{TFV_n}$. When an input news Tweet t is obtained, we first represent t by its feature vector representation $\boldsymbol{TFV_t}$ using the TF-IDF measure. We then use a variant of the *cosine similarity measure* [26] to compute the distance between t and a candidate news topic n in N, which is defined as follows:

$$\delta(t, n) = \frac{\boldsymbol{TFV_t} \bullet \boldsymbol{TFV_n}}{||\boldsymbol{TFV_t}||\,||\boldsymbol{TFV_n}||}$$

where $\boldsymbol{TFV_t}$, $\boldsymbol{TFV_n}$ are the feature vectors of t and n, respectively. t is considered a news update of the closest news topic n in N as long as $\delta(t, n) \leq \epsilon$, where ϵ is a pre-specified constant. In addition to a distance based constraint, we also stipulated that there be at least γ features between t and n, where γ is a small constant. If no such news topic exists in N, then we simply discard the Tweet t and proceed to the next Tweet in the input stream.

To expedite the search for a news topic n in N that is nearest to t as well as within a distance of ϵ from t, we maintain an inverted index on the feature vectors of the news topics in N. That is, for each feature f in $\boldsymbol{TFV_n}$ of a news topic n in N, the index stores pointers to all news topics in N that contain f. We use this index to reduce the number of distance computations required for associating a Tweet t with a news topic in N. When a new Tweet t is encountered, we only compute the distances to those news topics in N that have at least one feature in common with t. This optimization enables our algorithm to minimize the number of distance computations necessary for associating a Tweet with a news topic. Tweets that are updates to a news topic in N are directly posted to the user interface. The result is that users can see live updates of ongoing events, which is now possible due to our understanding of future event references in conventional news.

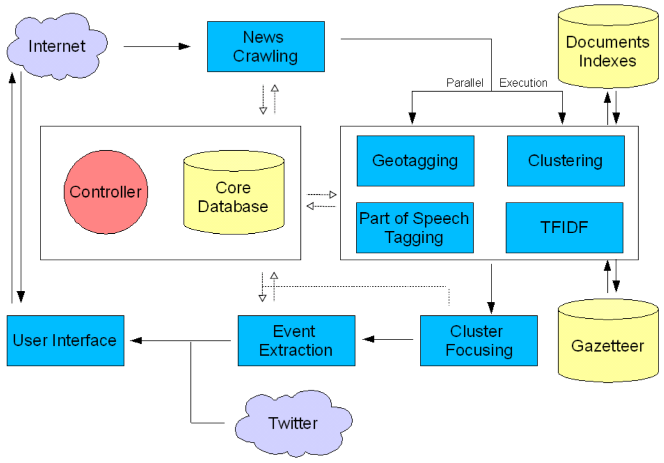


Figure 1: A high level overview diagram of NewsStand [28] with future events added to the system. News articles move through the system, which is orchestrated by the controller.

4. REVIEW OF GEOTAGGING

Geotagging [1] is the process of identifying locations in text and assigning them latitude/longitude coordinate values. Once text has been assigned coordinate values in this way, common spatial queries can be applied to it, including both feature-based and location-based queries. This allows a search to find both where something referenced in the text is happening and what texts discuss a certain place. Geotagging systems have been constructed for a variety of domains such as blogs [31], encyclopedia articles [7], news articles [5, 28], Twitter messages [23], and more.

The textual references to geographic locations identified by geotagging are known as *toponyms* [8]. Since most toponyms are somewhat ambiguous (e.g., over 60 places around the world are named "Paris"), simple textual matching is not sufficient to identify the latitude/longitude of a textual entity. Moreover, there is also the issue of toponym recognition (e.g., "Jordan" can refer to a person as well as a county). The system we used for geotagging news articles combined of data in the geonames database [29] with the location of the publisher of the news article while inferring a reader's local lexicon [10, 11, 17]. There are many other geotagging systems as well [1, 8, 12, 18, 28].

The geotagging used in the system we describe is fully explained in [9, 11], and results in geotagging locations in news stories with a good degree of accuracy (precision of at least 0.800). These locations are used to determine the areas about which a Twitter user is knowledgeable, so the accuracy of the geotagger is important to ensuring the accuracy of our determinations about Twitter users.

5. IDENTIFICATION OF FUTURE EVENTS

The ability to identify future events using a large news corpus is the crux of our system for finding reliable Twitter posts. Future events, along with their keywords and locations, are used to find Twitter posts that could be interesting to a general audience and would certainly be interesting to those who are following the event in question. Once events are identified, we can find Twitter posts around the time of the event and, based on information gleaned from a news database, determine which posts are most likely to be reliable. This helps to identify which users on Twitter can be trusted to post about certain geographical regions or types of events. The process of identifying dates and times in news stories has also been attempted by both Mani [14] and Schilder [24]. Both used quite different approaches from ours and achieved similar results. Mani's work used a few starting rules (similar to the tense identifiers described in Section 5.2 and Section 5.3), and then used machine learning techniques to improve tense identification. Schilder, who worked with German text, determined "semantic attributes" related to the date expressions found in the text. These attributes included the text itself (e.g. "on Monday"), the published date and time of the article, and words like "last," "next," and so on. By contrast, our methods for identifying tense are focused on using the verbs surrounding the date to identify tense.

By adding the ability to identify future events and integrating this service with Twitter, we can extend our previous works NewsStand [22, 28] and TwitterStand [23] by adding a temporal component to their spatial displays. The resulting system is diagrammed in Figure 1 and a screenshot of the resulting application is presented in Figure 2. A longer description of the resulting application is presented in Section 7.

The identification of future events proceeds as follows. First, news articles are collected from many news sources across the Internet. Their text is extracted, cleaned, and tagged by geotagging, named entity recognition, and part of speech tagging. Articles are clustered at this time by a process described in [28]. The resulting clusters of articles are all about the same story, so a "cluster" is just a collection of articles about the same news story. Keywords are determined for each article using a standard TFIDF analysis. Next, the text is scanned for any word that could possibly indicate a date (as in [14]). Each of these potential dates must be identified as future or non-future. To achieve this, a tense detector processes each sentence with a potential date and marks the date as future or non-future. This yields a list of future (and past) dates for the article. These dates, coupled with the story around them, are termed *events* for an article.

Since the process by which future dates are determined is imperfect, the process of clustering the articles is used to improve accuracy. In particular, once a cluster's size exceeds a certain threshold value (determined by the accuracy of tense identification and the desired accuracy of the resulting events), the cluster is scanned to find dates that are reported in a suitable number of the cluster's articles (this number must also be determined by desired accuracy of the result). Clustering assists the identification process because it helps to eliminate incorrectly identified events. Since events are only reported if they appear in a certain percentage of the cluster's articles, it is very unlikely that the tense of an event can be reported incorrectly. Nevertheless, for this to happen, the event would have to be erroneously identified in a large number of articles, which is very unlikely given the accuracy of the tense identifiers described later in this section. If a future event appears in enough articles in a cluster, then the system reports that a future event for the cluster is happening, and information such as keywords (identified by TFIDF) and location (identified by geotagging) can be used to determine the nature of the event.

Once a future event has been identified by finding it in articles, it can be used to find Twitter posts about the event. In particular, associated with each article is a list of keywords that are determined using a TFIDF analysis. These keywords are then compared between articles in the cluster so that if a keyword occurs in enough articles, then it is designated as a keyword for the cluster itself. Now, when one of the future dates for the cluster approaches, Twitter traffic that contains keywords associated with the cluster is likely to be about the event. This is effective because the clustering identifies keywords that pertain to the article, and future event identification is used to indicate when people are most likely to be discussing an article (really a cluster topic) on Twitter. Additionally, each article is run through a geotagging process, giving locations at which the event may occur or locations that are involved in the event (such as a sports event between two cities). Using Twitter's information about the location of its posters (or inferring it as in [23]) can also help identify traffic about an event that is legitimate, as users closer the the event geographically may be more likely to discuss it.

5.1 Tense Identification

To determine whether a date in a sentence corresponds to a future date is not always a simple task, and to date, little research has been done on the topic (though part-of-speech taggers have greatly improved, determining the tense of verbs well and some work has been done on automatically learning tenses of verbs [3, 13, 19], but these efforts have only been applied to past tense constructions). Some dates in text, like May 19, 2010 or 2010-05-19 or even "next Friday" can be identified unambiguously, many dates cannot be uniquely determined. For example, consider the difference between the sentences, "He appears in court Tuesday" and, "He appears to have fled on Tuesday." The first occurrence of "Tuesday" is clearly a future occurrence, while the second occurrence is clearly a past occurrence. In order to distinguish between examples such as these, we devised four different tense identification methods, termed *tense identifiers.* The first two methods are very simplistic and are presented here only to be used as a baseline to evaluate the last two methods, which are more complex and more effective. It should be noted that the last two methods require part of speech tagging to be applied to the news articles, while the first two methods do not. In systems handling huge numbers of articles at a high speed, the part of speech tagging requirement may be an obstacle, but (as reported in Section 6) the last two methods performed so much better than the first two that it is probably worth applying part of speech tagging to at least the sentences that contain date words for most systems.

5.2 Naive with "will" Method

The first method, termed the *naive with "will" method*, starts by searching for the word "will", the most common word marker of the future tense in the English language, and tags all of its occurrences as "future." Next, sentences containing a pronoun (identified by a list) or a proper noun (which can be identified by capitalization or TFIDF) followed immediately by a verb in the past tense, are tagged as "past." Past tense verbs were determined with the aid of a short list of common irregular past tense verbs (had, was, were, etc.) and any word ending in "-ed." Otherwise, the sentence was tagged as "not future" (but not necessarily past).

When this method determined that sentences were in the future tense, it was generally correct, as the word "will" is a fairly reliable marker of tense. However, in news articles, the future is often indicated by a present tense construction, such as the sentence "She hopes to resolve the matter on her July 6 court date." The fact that the "present" is constantly changing and "becoming the past", makes it nearly impossible to write a news story about the present, and thus it is usually the case that present tense verbs are used to indicate a future event in news articles.

5.3 Naive with "-s" Method

To address the issue of misinterpretation of "present tense" constructions, by being unable to classify a sentence like "She hopes to resolve the matter on her July 6 court date" as future, the naive method was modified to create a second method termed the *naive with "-s" method*. It functions the same way as the naive with "will" method with an additional check at the end for the occurrence of a a pronoun or proper noun being immediately followed by a word ending in "-s," in which case the sentence and all dates occurring in it are identified as "future." This was found to be effective as news stories are often written in third person, and the third person singular ending "-s" for present tense verbs can therefore be used to indicate a future event. As described in the Experimental Results section below, the naive with "-s" method had a substantially better recall value but a far worse precision value in comparison to the naive with "will" method. The reason for the dramatic fall in the precision value is the fact that most plural nouns (and many other words) also end in "-s," so sentences that contained a proper noun or a pronoun followed by a plural noun were mistakenly identified as "future." However, the advance in recall was significant enough that this method is worth considering for systems that cannot apply part of speech tagging.

It should be clear that both this method and the naive with "will" method (referred collectively as the *naive methods*) are inadequate for our purpose. First, neither method attempts to definitively identify the parts of speech of the words in the sentence. Aside from a small list of verbs, these naive methods attempt to infer which words are verbs and their tense by looking for pronouns and proper nouns followed by words with verb-like endings. While they work well for basic constructions such as "he was there Thursday," "she performs Wednesday," "they scored seven runs last Tuesday," etc., they are ineffective for more complicated constructions where the verb is split from the subject such as in the sentence "Jed York, president and CEO of the San Francisco 49ers, announced plans for the new stadium on Wednesday." Another problem with the two naive methods is that in news articles, future events are often described using infinitives (e.g. "The report, expected to arrive on Tuesday..."). Finally, some sentences have multiple cases, as in "He said Thursday that he will be playing in Sunday's game" In this example, the date word "Thursday" is in the past tense, while the date word "Sunday" refers to the future.

5.4 Verbs Method

The third tense identification method, termed the *verbs method*, addressed the major issues with the naive methods. In particular, in this method, each article is processed with a part of speech tagger, which identifies both verbs and their cases. The result is that all verbs are tagged, eliminating the need to infer which words were verbs. Additionally, sentences were analyzed at a phrase level instead of at sentence level, so that sentences with multiple tenses can be analyzed correctly.

The algorithm works as follows. Once the part of speech tagging process is done, all date words in the article are identified. Next, the tense of each date word is determined by analyzing the sentence in which it occurs. Each such sentence is decomposed into phrases (using punctuation), so that only the part of the sentence nearest the date word is used in the date word's identification. Initially, all verbs in the phrase containing the date word (including helping verbs) are assigned a score. The scores were determined heuristically and seem to work well in the systems that will be described. Both the verbs method of tense identification and the next tense identifier used the same scores.

High positive scores (+10) are assigned to verbs tagged as being in the future tense, which also included the modal verb "will". The high score represents the frequency with which these verbs are found in future constructions. In the sentences that were identified manually, every sentence containing the word "will" was at least partially in the future tense (some had a clause in future tense and a clause in past).

Verbs tagged as being in the past tense are assigned a score of -2. Past tense verbs are a good, but not absolute (as is the case of "will"), indicator of a past construction. Most sentences with past tense verbs in the sample set were past tense, but past tense verbs also occurred in appositive phrases, such as "The team, which won last week, plays tomorrow" in which the past tense "won" does not make the sentence past tense. Constructions like "it is expected to arrive Friday" also include past tense verbs despite being in future tense. However, in our corpus, past tense verbs were highly correlated with past constructions, so a score of -2 is assigned.

Verbs tagged as being in the present tense are assigned a score of +1. The rationale for this assignment in terms of the weakness of the evidence of the future action (i.e., assigning a -2 score for past events vis-a-vis assigning +1 to present events) was based on our experimental observation that the correlation of past tense verbs with past sentences was much higher than the correlation of present tense verbs were with future events. That is, in our corpus, present tense verbs were more correlated with future constructions than they were with past, but that correlation was not as strong as the correlation between past tense verbs and past constructions.

Most modal and helping verbs (such as "can," "be," and "may" but NOT "will") are assigned a score of 0. In our corpus, there was no clear correlation between modal verbs

Table 1: Comparative Results of Tense Identification

Identifier	Naive with "will"	Naive with "-s"	Verbs	Phrases
Correct Future	33	63	65	78
Number Future	97	97	97	97
Correct Past	240	189	240	237
Number Past	253	253	253	253
Incorrect	77 (22.0%)	98 (28.0%)	45 (12.9%)	35 (9.7%)
Precision (Future)	70.2%	49.2%	86.7%	84.8%
Recall (Future)	34.0%	49.2%	67.0%	80.4%
Precision (Past)	78.9%	84.8%	87.0%	91.5%
Recall (Past)	94.8%	74.7%	94.8%	93.7%

and tense.

Infinitive verbs, like present tense verbs are given a score of +1 since they often indicate future events, especially in news sources. They can be in all three tenses and are best determined by the verbs around them. For example, consider the three sentences, due to [27], which discusses the tense of infinitives at length:

- "I expect John to win the race" (future, determined by the present tense verb "expect"),
- "The president is believed to be guilty" (present, determined by present "is" with past participle "believed"),
- "I remember John to be the smartest" (past, determined by the present tense verb "remember").

The tense identifier accumulates all the scores for the verbs in the phrase containing the date word. If the phrase is assigned a non-zero score, then the score and its tense interpretation are returned and the process terminates. Otherwise (i.e., the phrase containing the date word has a score of 0), adjacent phrases are processed and assigned scores to be added to the score for the original phrase. Once a non-zero result is found or the method runs out of phrases, then the score is returned. A more formal description of the algorithm is given below, where scoreVerbs is a method that adds up the scores of all verbs in a given phrase.

Algorithm 1 The verbs method of tense identification

```
Break the sentence $s$ into phrases $p_i$.
Let $d$ be such that the date word is in phrase $p_d$.
Let $score \leftarrow 0$
Let $window \leftarrow 0$
while $score = 0$ do
    for $i \leftarrow d - window$ to $d + window$ do
        $score \leftarrow score + \text{scoreVerbs}(p_i)$
    end for
    $window \leftarrow window + 1$
end while
Return $score$
```

As an example, consider two applications of this method to the sentence "He will speak Tuesday night about the storm, which destroyed many houses in the Midwest on Friday." When the method is applied to the word "Tuesday," the first phrase is analyzed for a total score of 11 (10 for "will" and 1 for "speak"). This score is non-zero, so "Tuesday" is declared to be in the future tense. When the method is applied to "Friday," the first phrase analyzed only has the past tense "destroyed," so a score of -2 is returned and "Friday" is declared to be in the past tense.

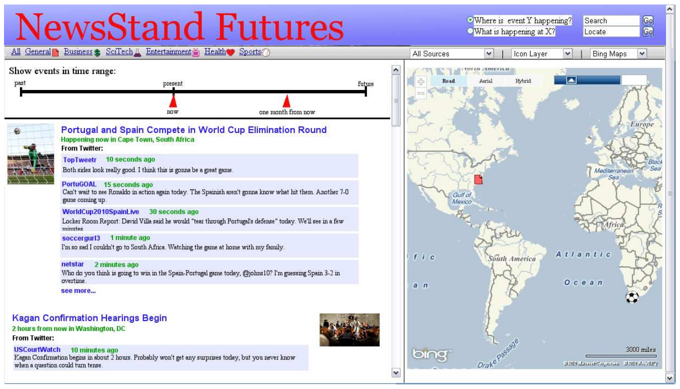
Observe that the verbs tense identification method makes two major improvements over both of the naive methods. First, by examining phrases instead of sentences, it eliminates the main source of erroneously-identified future events in the two naive methods, which is caused by sentences in which a date word appears in one part in one tense, while another part of the sentence is in a different tense. For example, in the above sentence involving the "storm," the naive with "will" method identifies "Friday" as a future word because of the presence of the word "will," but the verbs method correctly identifies the more proximate use of "destroyed" as determining the tense. Second, by using the output of a part of speech tagger, it eliminates errors caused by poorly guessing which words in the sentence are verbs. Because there are many verbs with past tenses not ending in "-ed" and there are many sentence constructions other than "proper noun or pronoun followed immediately by verb" this can greatly improve output. For example, neither naive method can handle a sentence as simple as "Smith, a former running back, arrived on Tuesday" because the verb is split from the proper noun, while the verbs method will easily identify "arrived" as a verb and Tuesday as being in the past.

5.5 Phrases Method

The fourth and final tense identification method, termed the *phrases method*, is very similar to the verbs method, except that it is also aware of verb phrases. In particular, verbs occurring within two words of each other are treated as a single phrase, and the phrase is assigned a score of -2, +1, or +10, much like a verb in the verbs method. The score assigned to a phrase is simply the score of the first word in the phrase. For example, in the sentence, "the storm is expected to arrive Tuesday" the verbs method would assign it a score of 0 (1 for the present "is," -2 for the past participle "expected," and 1 for the infinitive "arrive"). However, the phrases method assigns the phrase "is expected to arrive" a score of 1 since this is the score for the leading word in the phrase. The result is that the phrases method correctly identifies "Tuesday" as a future occurrence, while the verbs method does not. As another example, consider the sentence "But a change of plea hearing is scheduled for Wednesday in federal court." The verbs method identifies the word "Wednesday" as being in the past because the past tense tagging of "scheduled" (-2) outweighs the present tense word "is" (1). On the other hand, the phrases method by being aware of verb phrases can recognize that "is" is the determining word for the tense, and correctly identifies that "Wednesday" is in the future.

6. EXPERIMENTAL RESULTS

We evaluated the different tense identification methods by manually tagging 350 different dates from news as being future dates or not future dates, and running the tense identifiers on them. In this sample set, 97 dates were in the future, and the remaining 253 were in the past (generally there is no present tense in news stories, as the news cannot be read at the same time it is written). We report precision and recall values for both the determination that an event was occurring in the future and the determination that an event was not occurring in the future. In addition,



Portugal and Spain Compete in World Cup Elimination Round
Happening now in Cape Town, South Africa
From Twitter:
TopTweetr 10 seconds ago
Both sides look really good. I think this is gonna be a great game.
PortuGOAL 15 seconds ago
Can't wait to see Ronaldo in action again today. The Spainish aren't gonna know what hit them. Another 7-0 game coming up.
WorldCup2010SpainLive 30 seconds ago
Locker Room Report: David Villa said he would "tear through Portugal's defense" today. We'll see in a few minutes.
soccergurl3 1 minute ago
I'm so sad I couldn't go to South Africa. Watching the game at home with my family.
netstar 2 minutes ago
Who do you think is going to win in the Spain-Portugal game today, @johns10? I'm guessing Spain 3-2 in overtime.
see more...

Kagan Confirmation Hearings Begin
2 hours from now in Washington, DC
From Twitter:
USCourtWatch 10 minutes ago
Kagan Confirmation begins in about 2 hours. Probably won't get any surprises today, but you never know when a question could turn tense.

Figure 2: Screenshot of Spatio-Temporal Twitter Demo

we report correct and incorrect numbers for both future and non-future events. so as to show how often the events were correctly identified. Unlike the analyses by [14] and [24], we do not report accuracy on "TIMEX" expressions; we only report the results of the complete temporal identification of dates in news. The results can be found in Table 1.

In terms of the number of correct identifications, the phrases method did slightly better than the verbs method (i.e., the method that only used verb tags but was not aware of verb phrases). For future events, the recall score of the phrases method was also better, although its precision was slightly worse. Both naive methods are very bad, and should only be used as a baseline for the purpose of a comparison to the other two methods. The naive with "-s" method has very bad precision but decent recall, while the naive with "will" method has very bad recall with decent precision. Although, in the context of a large news database, precision should be more important (because there are enough articles to determine the future event in another source if it is missed in one), the increase in recall was sufficiently great when using the phrases method that the outcome of its use was superior for determining future events in the corpus as a whole.

Experiments to evaluate the quality of the breaking news obtained from combining Twitter with our future detection module were performed over a two week period starting Oct 10, 2010. In particular, our event detecting module identified 680 events in the two week period. Recall, that each event is represented by a set of feature vectors, which form the input for the *track* API of Twitter. In total we obtained about 96.5 million Tweets, which we continuously processed for breaking news. We discarded all Tweets containing less than 3 words as not being of sufficient length to convey meaningful information. Next, after passing the remaining Tweets through our classifier that classifies Tweets into *spam* or *news*, we discarded 92.7 million Tweets as spam. Although this may seem too draconian, such a measure is necessary to ensure that our output is of a high quality. We obtained 3.8 million Tweets, which were classified as news Tweets by the classifier. Now, we tried to associate these Tweets with an ongoing event, using the technique described in Section 3. Recall, that a Tweet t is considered part of the breaking news if the distance between an event n and t is less than ϵ, when both of the events are represented in terms of their feature vectors as well as when both t and n contain γ features in common between them. In our experiments, we set ϵ to be 0.5 and γ to be 3. Our system identified 76,098 Tweets as breaking news. In order to evaluate the quality of the output, we chose 100 events at random and chose up to 100 Tweets for each of the events. A human evaluator determined if a Tweet t associated with news event n actually belonged to n. The output was computed in terms of precision, which in this case is defined as the number of incorrect Tweets associated with a news event n. Note that computing the recall of our system is not interesting as we discard a large percentage of the input Tweets. However, we do argue that such an aggressive approach to noise reduction is probably the only option available to us given that most of the Tweets are not related to news. The precision of our method was found to be 93.80%, which means that the output of our system is of a high quality, which has always been our goal. Moreover, almost all of the errors occurred in Tweets associated with events drawn from relatively *local* events, which were not of broad interest to the users of Twitter. A simple way to eliminate these errors would be to discard events drawn from news clusters, which have relatively few news articles in them.

7. CONCLUDING REMARKS

Although the precision of our event detection methods is too low to definitively mark future events on an article-by-article basis, future events can be identified with a high degree of accuracy by comparing the classification results with other articles in the cluster. News stories about future events both in the near future (a few days) and the more distant future (a few months) were routinely identified. The resulting articles can be used as a way of determining what is currently happening in the world or as a seed to find reliable information on Twitter.

Once events are identified along with location and keywords, it is fairly simple to filter Twitter traffic around the time or place of the event based on those keywords. This identifies posts about the event, helping indicate which posts are reliable and which users are knowledgeable about certain areas or events. The resulting system allows a display of news that is both spatial and temporal, with input on ongoing events provided by Twitter. A screenshot of this application, based on our NewsStand [28] and TwitterStand [28] applications is provided in Figure 2.

Our system is designed to answer the following three questions "What is happening at X?" (a location-based query [2]), "Where is topic T or article Y happening?" (a feature-based query as in spatial data mining [2]), and "When is a topic T or article Y happening?". Consequently, our user interface has two selection modes corresponding to the "What" and "Where" modes and a slider that allows selection of news articles or topics based on the "When" criterion. The controls for the "What" and "Where" are radio buttons on the top right of the screenshot in Figure 2, while the slider on top left corner of Figure 2 allows filtering of news topics based on the "When" criterion. The users can interact with the map using *pan* and *zoom* to retrieve additional news articles. As users pan and zoom on the map, the map is constantly updated to retrieve new topics for the viewing window, thus keeping the window filled with topics, regardless of position or zoom level. A given view of the map attempts to produce a summary of the news topics in the view, providing a mixture of topic significance and geographic spread of the topics. Users interested in a smaller or larger geographic region than the map shows can zoom in or out to retrieve more topics involving that region. A slider provides dynamic control in restricting article based on the reference to the future event they contain. In particular, our system improves on both NewsStand and TwitterStand in the sense that it exposes the user to the temporal aspect of events in news. Moreover, for news stories in our news interface that belong to the set of active stories, we constantly post updates as we find them in the form of Tweets posted in Twitter.

Future work involves using more advanced techniques from computational linguistics to increase the rate of correct tense identification. In particular, improvements to the analysis of verb phrases and modifications of the scoring parameters for sentences could be very helpful. Also, combining our techniques for tense identification with those of [14] or [24] could be productive. In addition, the linking of articles to future events allows the creation of a temporal news network, providing insight into how a story is reported before and after it occurs, and making it easier to determine how a story compared with its expectations. A temporal news network could also provide insights into the causes of events by providing a clear sequence. Placing this sequence on a map as in Figure 2 enables easy knowledge discovery about the temporal and spatial relationship between events. This could be aided by making use of spatial browsers and libraries [21, 25]. Finally, since the methods described were more effective at finding past events than future events, past events could also be cataloged. This could help to both identify reliable Twitter traffic (by looking through posts around the time and place of the event) and to support queries about what was happening on a given day. For example, if enough news stories were captured and analyzed, one could identify a number of events that were happening on a given day in any country, state, or even major city. By interacting with Twitter, historical researchers could find out how a group of people reacted to a given event (or set of events, type of event, etc.) with a high degree of accuracy. Additionally, finding past events as well as future events would enhance the usefulness of any application dedicated to helping people learn about events. By making it easy to compare news coverage to Twitter posts about an event, our system offers both up-to-the-minute information and valuable insight into past events.

8. REFERENCES

[1] E. Amitay, N. Har'El, R. Sivan, and A. Soffer. Web-a-Where: Geotagging web content. In *SIGIR*, pp. 273–280, Sheffield, UK, July 2004.

[2] W. G. Aref and H. Samet. Efficient processing of window queries in the pyramid data structure. In *PODS*, pp. 265–272, Nashville, TN, April 1990.

[3] J. L. Bybee and D. I. Slobin. Rules and schemas in the development and use of the English past tense. *Language*, 58(2):265–289, 1982.

[4] A. Chowdhury. State of Twitter spam. `http://blog.twitter.com/2010/03/state-of-twitter-spam.html`, March 2010.

[5] E. Garbin and I. Mani. Disambiguating toponyms in news. In *HLT/EMNLP*, pp. 363–370, Vancouver, Canada, October 2005.

[6] M. Helft. At Google, slow growth in news site. `http://www.nytimes.com/2008/06/24/technology/24google.html`, June 2008.

[7] W. Kienreich, M. Granitzer, and M. Lux. Geospatial anchoring of encyclopedia articles. In *InfoVis*, pp. 211–215, London, July 2006.

[8] J. L. Leidner. *Toponym Resolution in Text: Annotation, Evaluation and Applications of Spatial Grounding of Place Names.* PhD thesis, University of Edinburgh, Edinburgh, Scotland, 2007.

[9] M. D. Lieberman and H. Samet. Multifaceted toponym recognition for streaming news. In *SIGIR*, pp. 843–852, Beijing, China, July 2011.

[10] M. D. Lieberman, H. Samet, and J. Sankaranarayanan. Geotagging: Using proximity, sibling, and prominence clues to understand comma groups. In *GIR*, Zurich, Switzerland, February 2010.

[11] M. D. Lieberman, H. Samet, and J. Sankaranarayanan. Geotagging with local lexicons to build indexes for textually-specified spatial data. In *ICDE*, pp. 201–212, Long Beach, CA, March 2010.

[12] M. D. Lieberman, H. Samet, J. Sankaranarayanan, and J. Sperling. STEWARD: architecture of a spatio-textual search engine. In *ACM GIS* , pp. 186–193, Seattle, WA, Nov. 2007.

[13] C. X. Ling. Learning the past tense of English verbs: The symbolic pattern associator vs. connectionist models. *J. of Arti. Intll. Res.*, 1:209–229, 1994.

[14] I. Mani and G. Wilson. Robust temporal processing of news. In *ACL*, pp. 69–76, Morristown, NJ, USA, 2000.

[15] R. Mansfield. Johnny Depp alive after twitter hoax. `http://news.sky.com/home/technology/article/15535854`, January 2010.

[16] T. M. Mitchell. *Machine Learning.* McGraw-Hill, New York, NY, 1997.

[17] G. Quercini, H. Samet, J. Sankaranarayanan, and M. D. Lieberman. Determining the spatial reader scopes of news sources using local lexicons. In *ACM GIS*, pp. 43–52, San Jose, CA, Nov. 2010.

[18] E. Rauch, M. Bukatin, and K. Baker. A confidence-based framework for disambiguating geographic terms. In *HLT-NAACL*, pp. 50–54, Edmonton, Canada, May 2003.

[19] D. E. Rumelhart and J. L. McClelland. *On Learning the Past Tenses of English Verbs.* MIT Press, Cambridge, MA, USA, 1986.

[20] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Inf. Proc. & Mngt.*, 24(5):513–523, 1988.

[21] H. Samet, H. Alborzi, F. Brabec, C. Esperança, G. R. Hjaltason, F. Morgan, and E. Tanin. Use of the SAND spatial browser for digital government applications. *CACM*, 46(1):63–66, January 2003.

[22] H. Samet, B. E. Teitler, M. D. Adelfio, and M. D. Lieberman. Adapting a map query interface for a gesturing touch screen interface. In *WWW (Companion Volume)*, pp. 257–260, Hyderabad, India, March-April 2011.

[23] J. Sankaranarayanan, H. Samet, B. Teitler, M. Lieberman, and J. Sperling. Twitterstand: News in tweets. In *ACM GIS*, pp. 42–51, Seattle, WA, November 2009.

[24] F. Schilder and C. Habel. From temporal expressions to temporal information: semantic tagging of news messages. In *Proc. on Temporal and Spatial Inf. Proc.*, pp. 1–8, Morristown, NJ, USA, 2001.

[25] C. A. Shaffer, H. Samet, and R. C. Nelson. QUILT: a geographic information system based on quadtrees. *Int. J. of Geo. Inf. Sys.*, 4(2):103–131, April–June 1990.

[26] M. Steinbach, G. Karypis, and V. Kumar. A comparison of document clustering techniques. In *KDD Workshop on Text Mining*, pp. 1–20, Boston, MA, Aug. 2000.

[27] T. Stowell. The tense of infinitives. *Linguistic Inquiry*, 13(2):561–570, Summer 1982.

[28] B. E. Teitler, M. D. Lieberman, D. Panozzo, J. Sankaranarayanan, H. Samet, and J. Sperling. Newsstand: A new view on news. In *ACM GIS*, pp. 144–153, Irvine, CA, November 2008.

[29] M. Wick and B. Vatant. The geonames geographical database. `http://www.geonames.org/`.

[30] J. Yarow. Twitter finally reveals all its secret stats. http://www.businessinsider.com/twitter-stats-2010-4, April 2010.

[31] N. Yasuda, T. Hirao, J. Suzuki, and H. Isozaki. Identifying bloggers' residential areas. In *AAAI-CAAW*, Palo Alto, CA, March 2006.

Discovering Personalized Routes from Trajectories

Kai-Ping Chang[†], Ling-Yin Wei[†], Mi-Yeh Yeh[§], Wen-Chih Peng[†]
[†]Department of Computer Science, National Chiao Tung University, Hsinchu, Taiwan
[§]Institute of Information Science, Academia Sinica, Taipei, Taiwan
[†]{xud9wu,lywei.edu,wcpeng}@gmail.com
[§]miyen@iis.sinica.edu.tw

ABSTRACT

Most people usually drive their familiar routes to work and are concerned about the traffic on their way to work. If a driver's preferred route is known, the traffic congestion information on his/her way to work will be reported in time. However, the current navigation systems focus on planning the shortest path or the fastest path from a given start point to a given destination point. In this paper, we present a novel personalized route planning framework that considers user movement behaviors. The proposed framework comprises two components, familiar road network construction and route planning. In the first component, we mine familiar road segments from a driver's historical trajectory dataset, and construct a familiar road network. For the second component, we propose an efficient route planning algorithm to generate the top-k familiar routes given a start point and a destination point. We evaluate the performance of our algorithm using a real dataset, and compare our algorithm with an existing approach in terms of effectiveness and efficiency.

1. INTRODUCTION

With increasingly prevalent GPS devices, such as GPS loggers, smart phones, and GPS navigation devices, many location-based services are being developed. One popular location-based service is the navigation service. On Google map, a user issues a source and a destination, and then the shortest route from the source to the destination is derived. Most navigation services and research work have elaborated on how to efficiently derive the shortest route [5][4][18][16]. By exploring real-time traffic information, several research efforts have been devoted to discovering the fastest route [7][9][17]. In this paper, we study the problem of discovering personalized routes from a set of individual trajectories. Given a source and a destination, an ideal personalized route from the source to the destination passes through the road segments that a user frequently traverses.

Discovering personalized routes brings several benefits. The first benefit is to help drivers get useful traffic information in time. In general, people usually drive their familiar routes to work, and they are concerned about the real-time traffic along the routes. Once the personalized routes are planned, one could only retrieve real-time traffic information along those routes [20]. If a navigation system only plans the shortest/fastest route, it would waste the network bandwidth to retrieve useless real-time traffic information on the road segments since this user will follow his/her personalized routes. Moreover, people follow some regular routes for specific purposes, and these regular routes are usually not the shortest/fastest routes, but somewhat infer the activities of users. For example, given a source S and a destination D in Figure 1(a), a user usually follows Route 1 to stop by one specific drive-through restaurant, even if Route 2 is faster/shorter than Route 1. Given a personalized route, more intelligent location-aware services are designed to facilitate users' activities. For example, in the previous scenario, the drive-through restaurant could be informed in advance to prepare the user's food before his/her arrival.

In this paper, given a source, a destination and a rank threshold k, we discover the top-k personalized routes, in which the source and the destination are connected via road segments mined from a user's trajectories. To explore personalized routes, the study in [13] uses the concept of greedy algorithms to compute and create a least cost personal route. However, they do not consider the frequency of road segments that a user usually travels along, and the efficiency of deriving personalized routes is not very good. In addition, in [1], the purpose is to discover the most popular route from all users' trajectories. If a set of individual trajectories is given, the problem in [1] is the same as our problem with $k = 1$. In [1], the authors explore intersections to form a transfer network and discover the most popular route in the transfer network. However, the discovered route reveals a coarse path represented by intersections. For example, in Figure 1(b), although the discovered driving path $S \rightarrow D$ is known, we have no idea how to specify the specific routes from S to D if there are multiple routes between S and D.

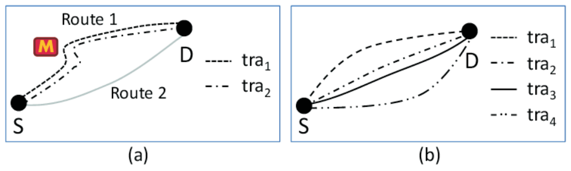


Figure 1: Examples of personalized routes.

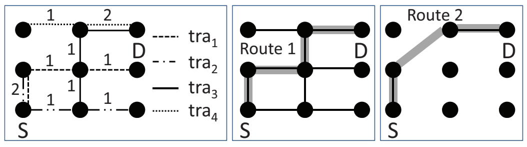


Figure 2: The impact of familiar road segments on discovering routes.

Furthermore, a transfer network is generated according to the intersections of the trajectories. It is likely to construct a transfer network by various trajectories from many users. However the number of intersections generated from individual trajectories may be few. Consequently, with such a rough transfer network, the work would discover a rough route so that a driver has no idea about detailed routes.

Given the source and the destination, one solution to derive personalized routes is to cluster individual trajectories that traverse from the source to the destination. However, if historical trajectories do not traverse from the source to the destination, such a clustering approach cannot find any personalized routes. Consequently, we have developed a pattern-aware personalized routing framework. In our framework, we first transform a user's trajectories into sequences of road segments, and extract familiar road segments that the user frequently traverses instead of extracting intersection points among trajectories. These familiar road segments are viewed as anchor points that indicate familiarity degrees from a user's trajectories. Given the source, the destination, and rank-threshold k, our goal is to derive the top-k routes that connect some of these familiar road segments. Note that the top-k routes mean that these k routes have the top-k highest scores, where the score function is designed in our paper to reflect not only the personalized preferences but also the length of a route. The challenges facing our framework are (1) how to determine the number of familiar segments; (2) how to formulate the score function for routes; and (3) how to efficiently derive the top-k routes via familiar road segments. For instance, given a source S and a destination D in Figure 2, Figure 2(a) shows four trajectories passing the area from one user and a weight on a road segment here represents the number of distinct trajectories traversing that segment. As shown in Figure 2(b), if we extract all road segments that have been traversed by the user, it will be costly to search routes from S to D. If we search routes based on the road segments selected by a certain degree of familiarity in Figure 2(c), the performance would be improved, but the discovered Route 2 is coarser than Route 1 discovered by the graph in Figure 2(b).

To overcome the above three issues, the proposed framework is comprised of two components, *familiar road network construction*, and *route planning*. First, we discover the familiar road segments that a user frequently traverses, and generate a familiar road network. With the familiar road network, we develop a score function for a route by considering a user's familiarity degree and the length of the route, and propose algorithms to discover the top-k personalized routes. We evaluate the performance of our framework using a real dataset generated by a user over a period of four months in Taiwan. In addition, we also compare our algorithm with an existing approach and extensive experiments are conducted to demonstrate the effectiveness and the efficiency of our framework.

The contributions of this paper are summarized as follows:

- We propose a pattern-aware personalized routing framework to derive the top-k routes.
- We formulate a route score function to reflect not only the personalized preferences but also the length of a route.
- We evaluate our framework using a real trajectory dataset, and the results demonstrate the effectiveness and the efficiency of our framework.

The remainder of this paper is organized as follows. Section 2 presents the problem definition. Section 3 introduces our proposed framework. In Section 4, we evaluate the performance of our framework. Section 5 introduces related work. Section 6 concludes this paper.

2. PRELIMINARY

In this section, we introduce some terms used in this paper and define the problem.

DEFINITION 1 (ROAD NETWORK). *A road network is a directed graph $G = (V, E)$, where V comprises intersections and road endpoints, and E is a set of road segments. Each road segment R in E is represented by $R : R.s \rightarrow R.e$ where $R.s$ and $R.e$ are R's endpoints and $R.s, R.e \in V$.*

In this paper, the collected dateset consists of raw trajectories. A raw trajectory is a sequence of GPS points, i.e., $T : p_1 \rightarrow p_2 \rightarrow \cdots \rightarrow p_n$. Each GPS point has latitude and longitude coordinates. Before using these raw trajectories, with a road network, we first map each GPS point of raw trajectories onto a road segment by searching for its closest road segment. However, although GPS data has spatial bias error, we can overcome this problem by the developed approaches [11][21]. After that, each raw trajectory is transformed into a sequence of road segments, which is formally defined as follows.

DEFINITION 2 (TRAJECTORY). *Given a road network $G = (V, E)$, a trajectory is represented by a sequence of road segments $T : R_1 \rightarrow R_2 \rightarrow \cdots \rightarrow R_n$ where $R_i \in E$ for $1 \leq i \leq n$.*

The problem of this paper is that given a source and a destination, we discover the top-k personalized routes from individual trajectories. The personalized route should pass through the road segments that a user is familiar with. Since our personalized routes are derived for navigation, the definition of routes is defined as follows:

DEFINITION 3 (ROUTE). *A route is a sequence of road segments, $P = R_1 \rightarrow R_2 \rightarrow \cdots \rightarrow R_n$, where $R_i.e$=$R_{i+1}.s$ for $1 \leq i < n$.*

3. PATTERN-AWARE PERSONALIZED ROUTING FRAMEWORK

In this section, we propose a framework of Pattern-aware Personalized rouTing (abbreviated as PPT). The framework

Table 1: An example of a trajectory dataset

TID	A sequence of familiar road segments
T_1	$R_1 \to R_3 \to R_4 \to R_7$
T_2	$R_2 \to R_1 \to R_4 \to R_8$
T_3	$R_4 \to R_7 \to R_8$
T_4	$R_8 \to R_9 \to R_6 \to R_2$
T_5	$R_3 \to R_6 \to R_5 \to R_4$
T_6	$R_4 \to R_8 \to R_6$

consists of two components: *familiar road network construction* and *route planning*. In the first component, we mine familiar road segments from one user's trajectory dataset and assign scores to the familiar road segments. These familiar road segments are used to construct a familiar road network. Second, we design an algorithm to discover on-line the top-k personalized routes from a familiar road network.

3.1 Familiar Road Network Construction

The component is performed in an off-line manner. We first discover a set of familiar road segments from one user's trajectory dataset. These familiar segments are viewed as some anchor points and will be utilized to derive personalized routes. Then, by representing each trajectory as a sequence of familiar road segments, we generate a trajectory profile of the user. In order to recognize the degree of familiarity of familiar road segments, we give a familiarity score to each familiar road segment.

To extract familiar road segments, we first define the frequency of a road segment as the number of distinct trajectories that pass through it. Then, we rank road segments in terms of their frequencies in decreasing order. Then, we only select the first θ road segments as familiar road segments. These terms are formally defined as follows.

Definition 4 (Frequency). *Given a trajectory dataset D and a road segment r, the frequency of the road segment r is defined as $f(r) = |\{T|r \in T, T \in D\}|$.*

Definition 5 (Familiar road segment). *Given a set of road segments and a rank-threshold θ, the road segments are sorted in decreasing order of frequency and the top-θ road segments are called familiar road segments.*

Once we have extracted the familiar road segments from the historical trajectories, we compute the familiarity score for each familiar road segment. In the beginning, we use the frequency of a road segment as its score. However, if there exists a road segment that has a very large number of trajectories passing through it, its score will be too large. Therefore, we need to perform normalization to derive scores for each familiar road segment. The highest frequency and the lowest frequency are denoted as f_{max} and f_{min}, respectively. The familiarity score function is formulated below.

Definition 6 (Familiarity score). *Given a set of familiar road segments $\mathcal{R}$ and a familiarity road segment R, a familiar score of R is defined as*

$$S(R) = \frac{f(R) - f_{min} + 1}{f_{max} - f_{min} + 1}$$

where $f_{max} = \max_{R\in\mathcal{R}}\{f(R)\}$ and $f_{min} = \min_{R\in\mathcal{R}}\{f(R)\}$.

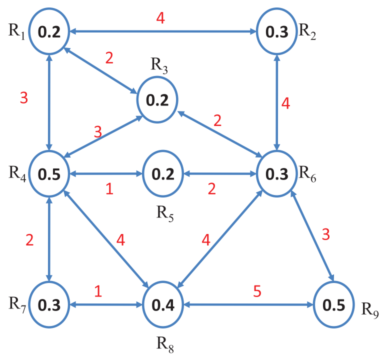


Figure 3: A familiar road network.

Given a set of familiar road segments $\mathcal{R}$=$\{R_1$,R_2,...,$R_n\}$, we construct a weighted directed familiar road network, denoted as $G(V, E)$, where each vertex represents a familiar road segment in $\mathcal{R}$, and a directed edge $< R_i, R_j >$ exists if there is at least one trajectory that traverses from R_i to R_j. The weight of each edge stands for the Euclidean distance between the two centers of the two road segments. It plays an important role in a familiar road network, because the weight is used to balance the familiarity degree and the length of a route.

Given a set of familiar road segments $\mathcal{R}$=$\{R_1$,R_2,...,$R_9\}$, and a trajectory dataset in Table 1, we build up connections between any two segments by scanning the trajectory dataset. We scan each trajectory and record the connected road segments. For example, R_1 and R_3 are connected in a familiar road network by scanning T_1. After identifying the connections among all familiar road segments, the familiar road network is depicted in Figure 3.

With the familiar road network, given a source and a destination, we first map them onto two familiar road segments (R_s,R_d) in the graph. Then, our problem is to *find the top-k personalized routes from R_s to R_d*. Note that each route will have its route score. The route score is defined as follows:

The definition of a route is described as below.

Definition 7 (Route score). *Given a route P=R_1→...→R_n, the score of the route is defined as*

$$S(P) = \frac{\sum_{i=1}^{n} S(R_i)}{\sum_{k=1}^{n-1} distance(R_k, R_{k+1})}$$

where $distance(R_k, R_{k+1})$ is the Euclidean distance between the center of R_k and the center of R_{k+1}.

Note the route score takes both the familiar scores of road segments and the length of routes into consideration. For example, in Figure 3, if a route P is $R_1 \to R_2 \to R_6 \to R_9$, the score of the route $S(P) = \frac{1.3}{11} = 0.118$.

3.2 Route Planning

When a user issues a query, routing planning is performed in an on-line manner. The input of the query includes the source S, the destination D, and a rank-threshold k. We propose a naïve method and an efficient algorithm to discover the top-k personalized routes.

First, when a query is issued, we map the source S and the destination D onto two road segments in the constructed

Algorithm 1: Backtracking Algorithm

```
Input: A familiar road network $G(V, E)$, the source
       $S.R$, the destination $D.R$, and a rank-threshold
       $k$.
Output: Top-$k$ routes.
1  Results ← null /* A set of routes */
2  route ← null
3  set = DR($S.R$,route,Result)
4  foreach routes $p_i \in set$ do
5      sum ← 0, length ← 0
6      foreach segment $R_j \in p_i$ do
7          sum ← sum + $S(R_j)$
8          length ← length + distance($R_j, R_{j+1}$)
9      end
10     $S(p_i)$ ← sum/length
11 end
12 Routes ← max-k($S(p_i)$)
13 return Routes
```

Algorithm 2: Deriving Routes(DR)

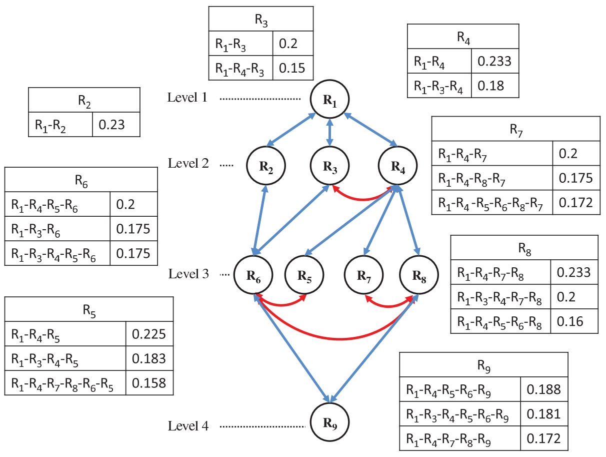
```
Input: R, route and Result
Output: Result
1  if $R \notin route$ then
2      append R to route
3      if $R == D.R$ then
4          Result ← Result ⋃ route
5      else
6          for each neighbor $N$ of $R$ do
7              Backtracking(N,route,Result)
8          end
9  end
10 return Result
```

familiar road network. Both S and D are represented by latitude and longitude coordinates. With a given S and D, we need to allocate their corresponding familiar road segments in the familiar road network. If S (or D) does not locate on any familiar road segment, we will find the closest familiar road segment for S (or D). If there exist multiple familiar road segments closest to S (or D), we choose the segment whose score is highest to stand for S (or D). The reason is that a score of a familiar road segment is higher and the user is more familiar with the road segment. Note that we use the road segments $S.R$ and $D.R$ to represent S and D, respectively.

3.2.1 Backtracking Algorithm

Once we have a familiar road network, a naïve BackTracking algorithm (abbreviated as BT) is proposed to derive the top-k personalized routes. Algorithm BT starts from $S.R$ and stores the routes in a set. Then, algorithm BT will expand its routes until $D.R$ is reached. The routes in the set are candidates for the top-k personalized routes. Algorithm BT will have a route set that includes all possible routes from $S.R$ to $D.R$ and we compute the scores of the routes in the set. According to the scores of the routes, we will rank these routes in decreasing order. Finally, the top-k routes are selected. Algorithm BT is detailed in Algorithm 1.

For example, given the graph $G(V, E)$ in Figure 3, the source is R_1 and the destination is R_9. First, we find the neighbors of R_1, i.e., R_2, R_3 and R_4. Then we use the backtracking algorithm to search for all the neighbors of R_2, R_3 and R_4. A route will be generated once R_9 is reached. After searching, we derive 23 possible routes from R_1 to R_9. Moreover, we compute the score of each route and the top-1 route is $R_1 \to R_4 \to R_5 \to R_6 \to R_9$.



R_3	
R_1-R_3	0.2
R_1-R_4-R_3	0.15

R_4	
R_1-R_4	0.233
R_1-R_3-R_4	0.18

R_2	
R_1-R_2	0.23

R_7	
R_1-R_4-R_7	0.2
R_1-R_4-R_8-R_7	0.175
R_1-R_4-R_5-R_6-R_8-R_7	0.172

R_6	
R_1-R_4-R_5-R_6	0.2
R_1-R_3-R_6	0.175
R_1-R_3-R_4-R_5-R_6	0.175

R_8	
R_1-R_4-R_7-R_8	0.233
R_1-R_3-R_4-R_7-R_8	0.2
R_1-R_4-R_5-R_6-R_8	0.16

R_5	
R_1-R_4-R_5	0.225
R_1-R_3-R_4-R_5	0.183
R_1-R_4-R_7-R_8-R_6-R_5	0.158

R_9	
R_1-R_4-R_5-R_6-R_9	0.188
R_1-R_3-R_4-R_5-R_6-R_9	0.181
R_1-R_4-R_7-R_8-R_9	0.172

Figure 4: An running example of algorithm ER.

However, the discovered routes are rough since they are represented by familiar road segments and may not be consecutive in a real road network. In order to give users detailed routes, we further search for the shortest/fastest path between any two consecutive familiar road segments in a route that are discontinuous in a road network. Note that, R_i and R_j are discontinuous in a road network if $R_i.e$ and $R_j.s$ are not the same. After that, each discovered route is represented by consecutive road segments in a road network. Finally, we return such refined routes to the user.

3.2.2 Efficient Routing Algorithm

Algorithm BT finds all the routes from the source to the destination. Since the response time of deriving the top-k routes should be shorter, algorithm BT is not suitable. Thus, we propose an Efficient Routing algorithm (abbreviated as ER).

This algorithm is developed based on the concept of branch-and-bound search. While searching for results, we store the top-k' subroutes and their route scores in a table for each vertex in the graph. Note that the table for a familiar road segment R_i is denoted as $T(R_i)$. Then we expand the stored subroutes to derive the top-k personalized routes. Specifically, we let a source be in level 1 and start at that level. When going to some vertices in the next level, we will use the routes stored in the current table to compute all possible routes from the current vertex to some vertex in the next level. Then we save the top-k' subroutes and their scores in the table for the visited vertex in the next level. If there exist more than two neighboring vertices (i.e., two vertices that are connected in the graph) in the next level, we expand the current subroutes to visit these neighbors and renew their tables. To find the routes from layer i to layer j, where $i > j$, we use the tables of level i to update each layer from layer i to layer 1. With these new tables, we could derive routes by updating the routing sets from layer 1 to layer i. We repeat this procedure until all levels are visited.

We illustrate the algorithm ER with a running example in Figure 4 as follows. Given a graph in Figure 3, a source R_1, a destination R_9, $k = 3$ and $k' = 3$, we start at R_1 in level 1. Then, we visit its children R_2, R_3 and R_4, respectively. For each child, we save the routes from R_1 to itself into $T(R_2)$, $T(R_3)$ and $T(R_4)$, respectively. As shown in Figure 4, for instance, the current discovered route from R_1 to R_3 is $R_1 \rightarrow R_3$. However, $T(R_3)$ and $T(R_4)$ are neighbors in level 2, so we renew table $T(R_3)$ considering $R_3 \rightarrow R_4$. Similarly, we also renew table $T(R_4)$ considering $R_4 \rightarrow R_3$. Hence, according to $T(R_3)$, there are two possible routes from R_1 to R_3. After that, we search level 3 by using the routes stored in $T(R_2)$, T_{R_3}, and $T(R_4)$. However, if the number of possible routes is over k', we only keep the top-k' routes in the table of a vertex. The algorithm will stop when level 4 is reached. Finally, we provide the top-k' routes stored in $T(R_9)$ to the user. Similarly, if two road segments are not connected, we will perform the shortest/fastest path between these two segments.

After that, ER explores approximate top-k routes since it only keeps the top-k' subroutes at each searching step. For example, given $k = 2$, assume that the top-2 subroutes (p_1, p_2) are saved in $T(R_1)$ and $S(p_1)$=5.689/1389=0.00409 and $S(p_2)$=4.689/1155=0.00405. When road segment R_2 whose score is 0.8 is visited (i.e., p_1 and p_2 are appended R_2), $S(p_1)$=6.489/1500=0.00432 and $S(p_2)$=5.489/1266=0.00433 if the distance between R_1 and R_2 is 111. In the beginning, $S(p_1) > S(p_2)$. After visiting R_2, $S(p_2) > S(p_1)$. If $k' = 1$, $T(R_1)$ would only keep p_1. Then, p_2 with a score higher than p_1 would not be found.

4. EXPERIMENTAL RESULTS

In this section, we evaluate the performance of the proposed algorithm using a real trajectory dataset. The real trajectory dataset that consists of 5,294 raw trajectories and 122,747 GPS points was collected from one user over a period of four months in Kaohsiung, Taiwan. We compare our algorithm with the existing approach MPR [1] in terms of discovering routes and efficiency. Since familiar road network construction, which is executed off-line, is preprocessing for on-line routing, we evaluate the efficiency of the algorithms by on-line query time. Moreover, we study the influence of the parameters, the rank-threshold k and the number of familiar road segments θ, on performance. In the experiments, the default setting is that $k = 5$ and $\theta = 100$. In addition, all algorithms are implemented in Java, and the experiments are conducted on Windows with an Intel Core 2 CPU (2.93GHz) and 2.0GB Memory.

4.1 Performance Comparison

We first compare our discovered routes with the results derived from the existing approaches, the shortest path and the algorithm MPR [1]. Figure 5(a) shows a user's trajectory dataset with the red degree representing the frequency of a road segment according to the dataset. Given a source S and a destination D, the top-1 route discovered by each approach is demonstrated in Figure 5. Figure 5(b) shows the shortest path from the source S to the destination D. However, we can observe that in Figure 5(a), the shortest path does not pass through many familiar road segments. It is possible that the user does not like to traverse these roads or he/she

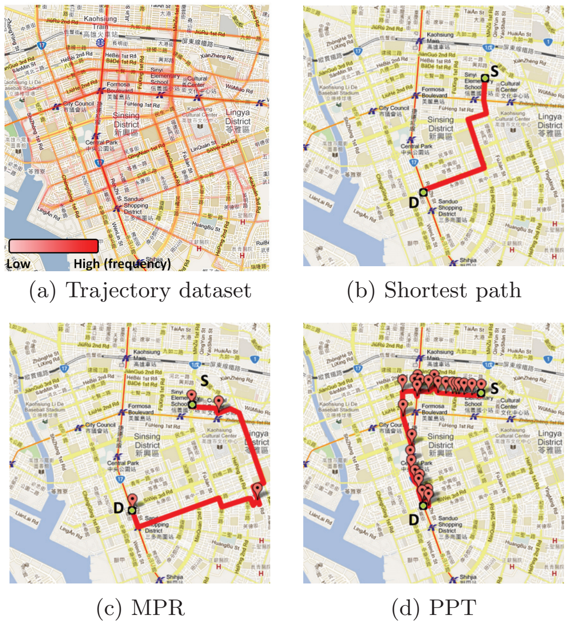


Figure 5: Top-1 route discovered using different approaches.

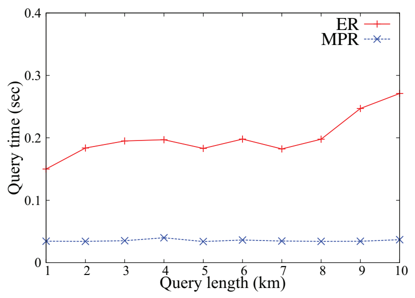
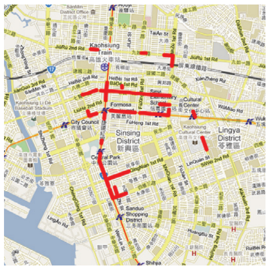
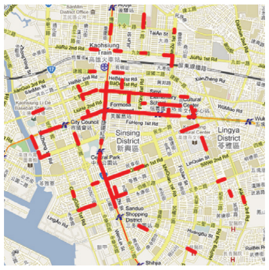


Figure 6: Efficiency comparison.

is not familiar with the streets. The route discovered by MPR only contains four points, marked by points in Figure 5(c), so the route is rough. A user may still have no idea about driving from S to D if the route only consists of four points. We therefore apply the shortest path approach to fill the missing parts between any two consecutive points, and the refined route is shown in Figure 5(c). As shown in Figure 5(c), the red line indicates the post-filled route by applying the shortest path. In addition, the length of the route discovered by MPR is longer and does not favor the user's familiar road segments. Figure 5(d) shows our discovered route which obviously favors the road segments with higher frequency even if the length is slightly longer than the length of the shortest path.

Moreover, we compare our algorithm *ER* with the algorithm MPR in terms of query time. As shown in Figure 6, we evaluate the query time for our algorithm ER and MPR with varying query lengths. Query length here means the Euclidean distance between a source and a destination. In Figure 6, the query time of ER slightly increases as the


(a) θ=50


(b) θ=100

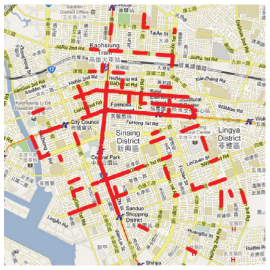
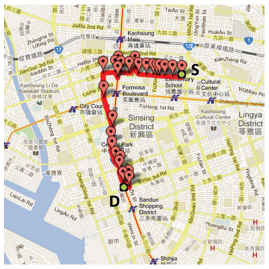
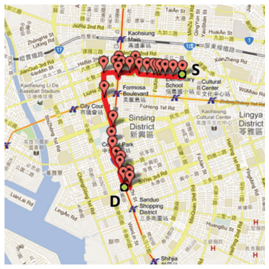

(c) θ=150

Figure 7: Familiar road segments by different θ.


(a) θ=50


(b) θ=100

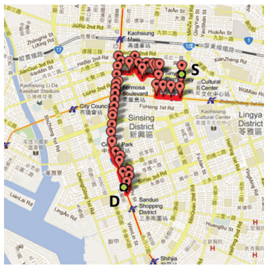

(c) θ=150

Figure 8: Top-1 route by different θ.

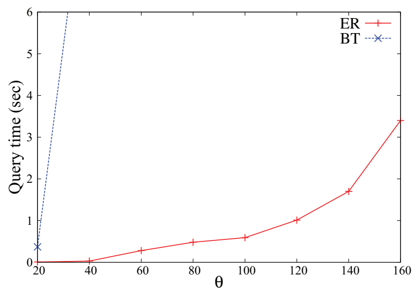


Figure 9: Efficiency evaluation with different θ.

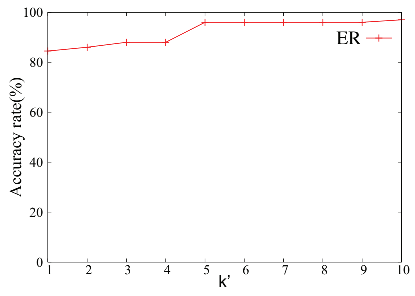


Figure 10: Effectiveness evaluation with different k'.

query length increases. Although MPR is faster than ER, the query time of ER is still less than 0.3 seconds.

4.2 Parameter Study

4.2.1 *Effect on θ*

According to the real trajectory dataset in Figure 5(a), Figure 7 shows the familiar road segments generated by different θ. In Figure 8, given a source S and a destination D, the top-1 route is discovered with respect to θ. The discovered routes are similar when $\theta = 50$, $\theta = 100$ and $\theta = 150$. However, the route in Figure 7(c) is slightly longer than the routes in Figure 7(a) and Figure 7(b). The reason is that a route could favor more familiar road segments with a larger θ without the length of the route being not increased too much.

In addition, we evaluate the efficiency of our proposed algorithms (BT and ER) with different θ. As shown in Figure 9, the query time of ER slightly increases when θ is increased from 20 to 120. Then, the query time of ER is dramatically increased when θ is greater than 120. However, the query time of the naïve algorithm *BT* is markedly slower than the query time of ER. Therefore, Figure 8 and Figure 9 could advise us to set a proper θ to derive effective personalized routes without high computation costs.

4.2.2 *Effect on k*

We evaluate the accuracy rate of ER with different k. Given a rank-threshold $k = 1$, we first use BT to derive top-1 route. With using ER, we record the top-k' local optimal routes in each step while searching for the top-1 route. In the experiments, we ran ER to discover the top-1 route with varying k' from 1 to 10. We ran 90 different queries (denoted as Q) for the experiments and averaged the results. The

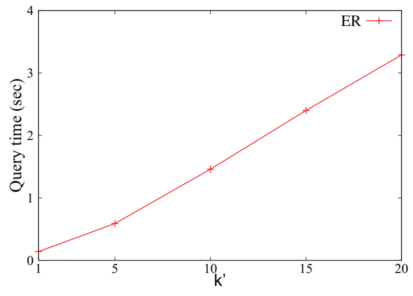


Figure 11: Efficiency evaluation with different k'.

accuracy rate is defined as $\Sigma_{q \in Q} c(q)/|Q|$ where $c(q) = 1$ if $p^q_{BT} = p^q_{ER}$ and $c(q) = 0$ if $p^q_{BT} \neq p^q_{ER}$. Note that p^q_{BT} (or p^q_{ER}) represents the top-1 route discovered by BT (or ER) according to the query q. As show in Figure 10, when k' is 1, the accuracy rate of ER is 0.85. While k' is between 2 and 5, the accuracy rate of ER is 0.88. The accuracy rate of ER reaches 0.97 when $k' = 5$. Figure 10 demonstrates that the accuracy rate of ER increases when k' increases.

In addition, we also evaluate the effect of k' on query time in Figure 11. The query time of ER dramatically increases as k' increases. However, although a larger k' induces more results for a user, the user may prefer the significant ones instead of a large volume of results. When $k' = 5$, the accuracy rate of ER is close to 0.9 and the query time of ER is less than 1 second.

5. RELATED WORK

5.1 Route planning

5.1.1 *Shortest path*

There are many famous algorithms to find the shortest path, such as Dijkstra [5] and A* algorithm [4]. Dijkstra is a single-source shortest path search on a weighted graph, and the time complexity is $O(m+nlogn)$. In [18] the authors propose a landmark-based method for point-to-point distance estimation in a large network. The authors consider that the existing algorithms are not suitable for large networks because the data is too large to be computed. Therefore, the method is proposed to solve this problem. However, the authors in [18] select landmarks at random that are not efficient, while in [16] the authors provide some methods to select landmarks rather than selecting them at random, to increase efficiency.

5.1.2 *Fastest path*

Time-dependent shortest path search is also a popular issue [7] [20] [9]. [7] acquires traveling time functions by mining speed patterns, but the work does not describe how to produce speed patterns in detail. Therefore, [20] describes how to produce speed patterns by mining historical trajectory data specifically, and propose a method named the *spatio-temporal weighted method* to accurately estimate the traffic status (i.e., the driving speed). In [9], it is mentioned how to change a speed pattern into the traveling time function. Given a start-off time and road segment, we can know how much time it takes to pass the segment by the traveling time functions. The minimum cost path from a source point to the destination point means a path with the smallest travel time among all possible paths on a road network. [17] presents *PathMon* to find the minimum cost path, which is an efficient system for monitoring min-cost paths in dynamic road networks. It consists of a *query scope index*(QSI) and a *partial path computation algorithm*(PPCA).

5.1.3 *Popular route*

In [1], a framework is proposed to recommend popular routes from historical trajectories, so in some respects, the work is similar to ours. First, in order to retrieve a transfer network from the user's trajectory data, they propose a Coherence Expanding algorithm. Second, they use the Absorbing Markov Chain model to acquire a reasonable transfer probability for each node in the transfer network. Third, to discover the MPR from a transfer network, they design a Maximum Probability Product algorithm that is based on the popularity indicators in a breadth-first way. The work in [22] proposes a framework to mine k interesting regions with the highest score, which are acquired by using that user's travel experiences, and location interests have a mutual relationship; however, they focus on the interesting regions rather than trajectory recommendations. The work in [19] extracts frequent regions by giving each region an interest score, and then analyzes historical trajectories to find the most interesting route.

5.1.4 *Personalized route*

The goal of navigation services is to compute routes between the source and the destination. However, the navigation result usually only considers the shortest path, even if a familiar path exists. Therefore, a system called MyRoute [15] has been developed. MyRoute is based on a priori knowledge of familiar routes and landmarks to create user specific routes, in order to reduce route complexity. However, the drawback is that users need to manually provided the familiar landmarks. Therefore, the authors in Going My Way [3] proposed a user-aware route planner. It can identify the landmarks automatically from the personal historical GPS log data without needing the user to manually input landmarks. Going My Way can provide directions based on personal landmarks rather than street names and intersections. However, even if the idea can reduce the complexity of the routing recommended by the navigation system, it would not be intuitive for many people. Moreover, for a place the user has never been to, the system would have no assistance for the user. In [13], authors describe and evaluate a case-based route planning system. First they collect user profiles, then, when the user inputs the start and end locations, they use the greedy concept to compute and create a least cost personal route and return it to the user, but the drawback is that they do not consider the concept of frequency. For example, the user has passed through a road before, but maybe he is not familiar with the path, because maybe he just passed along that road once. On the other hand, the work in [2] studies a new problem of searching trajectories by location, in order to find the k Best Connected Trajectories (k-BCT). For instance, if users query five locations with longitude and latitude, then the framework returns k trajectories that are the closest to the five locations.

5.2 Trajectory pattern mining

There are many approaches which have elaborated on mining trajectory patterns, such as the stay-point-based approach [22] [10], the density-based approach [12] [8], and the grid-based approach [14] [6]. In general, trajectory patterns are sequences of regions in which a user usually appears. There exists an important issue in trajectory pattern mining which is how to define a hot region, a basic unit of trajectory patterns. In [22] and [10], a stay point stands for a geographic region where the user stayed for a while. Compared to a raw GPS point, each stay point has a particular semantic meaning such as home, company, etc. In [12] and [8], these works have explored the density concept, which identifies hot regions as those that contain a sufficient number of data points. These hot regions are the basic units in trajectory patterns. Given a set of trajectories, prior works in [12] and [8] have directly employed density-based clustering algorithms (e.g., DBSCAN) to identify hot regions. In [6], the whole space is divided into grids, and the density in a grid is defined as the number of trajectories that cross it. Grids that have higher densities form a compact region and are regarded as a hot region.

6. CONCLUSIONS

In this paper, we propose a pattern-aware personalized routing framework for personal route planning. We first mine familiar road segments and construct a familiar road network from the historical trajectories generated by a user. Given a source, a destination and a rank-threshold k, we propose a naïve backtracking algorithm and an efficient routing algorithm to derive the top-k personalized routes from the familiar road network. To derive ranked routes, we formulate the route score function by considering a user's familiarity degree and the length of a route. We have conducted experiments using a real trajectories collected from a user. The experimental results demonstrate the performance of our framework and show that our efficient routing algorithm could derive the top-k personalized routes that approximate the real top-k personalized routes. In the future, we will consider time information for familiar road network construction, and propose a more effective routing algorithm for a large familiar road network.

7. ACKNOWLEDGMENTS

Wen-Chih Peng was supported in part by the Industrial Technology Research Institute, Project No. 100-EC-17-A-05-01-0626.

8. REFERENCES

[1] Z. Chen, H. T. Shen, and X. Zhou. Discovering popular routes from trajectories. In *Proc. of ICDE*, pages 900–911, 2011.

[2] Z. Chen, H. T. Shen, X. Zhou, Y. Zheng, and X. Xie. Searching trajectories by locations - an efficiency study. In *Proc. of ACM SIGMOD*, pages 255–266, 2010.

[3] J. Chung and C. Schmandt. Going my way: a user-aware route planner. In *Proc. of CHI*, pages 1899–1902, 2009.

[4] R. Dechter and J. Pearl. Generalized best-first search strategies and the optimality of a*. *J. ACM*, 32(3):505–536, 1985.

[5] E. W. Dijkstra, W. Heise, A. J. Perlis, and K. Samelson. Algol sub-committee report - extensions. *Commun. ACM*, 2(9):24, 1959.

[6] F. Giannotti, M. Nanni, F. Pinelli, and D. Pedreschi. Trajectory Pattern Mining. In *Proc. of ACM SIGKDD*, pages 330–339, 2007.

[7] H. Gonzalez, J. Han, X. Li, M. Myslinska, and J. P. Sondag. Adaptive fastest path computation on a road network: A traffic mining approach. In *Proc. of VLDB*, pages 794–805, 2007.

[8] H. Jeung, Q. Liu, H. T. Shen, and X. Zhou. A hybrid prediction model for moving objects. In *Prof. of ICDE*, pages 70–79, 2008.

[9] E. Kanoulas, Y. Du, T. Xia, and D. Zhang. Finding fastest paths on a road network with speed patterns. In *Proc. of ICDE*, page 10, 2006.

[10] Q. Li, Y. Zheng, X. Xie, Y. Chen, W. Liu, and W.-Y. Ma. Mining user similarity based on location history. In *Proc. of ACM SIGSPATIAL GIS*, page 34, 2008.

[11] Y. Lou, C. Zhang, Y. Zheng, X. Xie, W. Wang, and Y. Huang. Map-matching for low-sampling-rate gps trajectories. In *Proc. of ACM SIGSPATIAL GIS*, pages 352–361, 2009.

[12] N. Mamoulis, H. Cao, G. Kollios, M. Hadjieleftheriou, Y. Tao, and D. W. Cheung. Mining indexing and querying historical spatiotemporal data. In *Proc. of ACM SIGKDD*, pages 236–245, 2004.

[13] L. McGinty and B. Smyth. Personalized route planning: A case-based approach. In *EWCBR*, pages 431–442, 2000.

[14] A. Monreale, F. Pinelli, R. Trasarti, and F. Giannotti. Wherenext: a location predictor on trajectory pattern mining. In *Proc. of ACM SIGKDD*, pages 637–646, 2009.

[15] K. Patel, M. Y. Chen, I. E. Smith, and J. A. Landay. Personalizing routes. In *Proc. of ACM UIST*, pages 187–190, 2006.

[16] M. Potamias, F. Bonchi, C. Castillo, and A. Gionis. Fast shortest path distance estimation in large networks. In *Proc. of ACM CIKM*, pages 867–876, 2009.

[17] Y. Tian, K. C. K. Lee, and W.-C. Lee. Monitoring minimum cost paths on road networks. In *Proc. of ACM SIGSPATIAL GIS*, pages 217–226, 2009.

[18] M. V. Vieira, B. M. Fonseca, R. Damazio, P. B. Golgher, D. de Castro Reis, and B. A. Ribeiro-Neto. Efficient search ranking in social networks. In *Proc. of ACM CIKM*, pages 563–572, 2007.

[19] L. Y. Wei, W. C. Peng, B. C. Chen, and T. W. Lin. Pats: A framework of pattern-aware trajectory search. In *Proc. of MDM*, pages 372–377, 2010.

[20] L.-Y. Wei, W.-C. Peng, C.-S. Lin, and C.-H. Jung. Exploring spatio-temporal features for traffic estimation on road networks. In *Proc. of SSTD*, pages 399–404, 2009.

[21] J. Yuan, Y. Zheng, C. Zhang, X. Xie, and G. Sun. An interactive-voting based map matching algorithm. In *Proc. of MDM*, pages 43–52, 2010.

[22] Y. Zheng, L. Zhang, X. Xie, and W.-Y. Ma. Mining interesting locations and travel sequences from gps trajectories. In *Proc. of WWW*, pages 791–800, 2009.

Sensing Urban Mobility with Taxi Flow

Marco Veloso
Centro de Informática e Sistemas da Universidade de Coimbra, Portugal
Escola Superior de Tecnologia e Gestão de Oliveira do Hospital, Portugal
mveloso@dei.uc.pt

Santi Phithakkitnukoon
Culture Lab, School of Computing Science, Newcastle University, United Kingdom
SENSE*able* City Lab, Massachusetts Institute of Technology, Cambridge, MA, USA
santi@mit.edu

Carlos Bento
Centro de Informática e Sistemas da Universidade de Coimbra, Portugal
bento@dei.uc.pt

ABSTRACT

The analysis of taxi flow can help better understand the urban mobility. In this work, we analyze 177,169 taxi trips collected in Lisbon, Portugal, to explore the relationships between pick-up and drop-off locations; the behavior between the previous drop-off to the following pick-up; and the impact of area type in taxi services. We also carry out the analysis of predictability of taxi trips given history of taxi flow in time and space.

Categories and Subject Descriptors

I.5.2. Patter Recognition: Pattern analysis

General Terms

Algorithms.

Keywords

Urban mobility, spatiotemporal analysis, taxi-GPS traces, naïve Bayesian classifier.

1. INTRODUCTION

With the development of pervasive technologies (e.g. Global Positioning System, Global System for Mobile Communications), new services have been provided, such as Location-Based Social Networks (LBSN), allowing people to share geo-referenced information. Points of Interest (POI) are always something useful to know when travelling in new cities and LBSN can help us finding relevant locations according to our profile. Nevertheless, urban areas are experiencing a growth in size and population, and with that growth there is also an increase of attractive places. The constant movement of people demands for changes in the transportation systems, to improve public transportation modes (e.g. bus, metro, train) in order to meet citizens needs and therefore reduce the use of individual means of transport.

To optimize the public transportation network it is essential to understand what drives the common citizen. Retrieving data from the traditional public transportation (e.g. bus, train, metro) can provide a relevant database of samples and general passengers' movement. However, does not always provide the exact origin and destination for each passenger, since these transportation modes rely on pre-designated stops and paths, and usually the ticket validation is only performed on the pick-up. The taxi service can be a way to retrieve large dataset of information with a higher precision when we focus the origin and destination of each trip. It can pick-up the passengers right where they are standing, and then drop-off them precisely in the desirable destination, without being bounded to a pre-determined path.

Our on-going work is focused on the analysis of taxi-GPS traces acquired in the city of Lisbon, Portugal, to better understand urban mobility. The contribution of this work lies on the following two aspects: spatiotemporal analysis and study of predictability of taxi trips. For the former, we analyze taxi traces to identify the relationships between pick-up and drop-off locations; characterize the scenario between taxi services (i.e. what happens between the latest drop-off and next pick-up) in order to improve taxi profit; and explore the value of Points of Interest in analysis of taxi flow. For the latter, we explore the possibility of predicting the next destination given hour of the day, day of the week, weather condition, and area type.

2. RELATED WORK

Although our on-going work is relatively recent, this research field is well known, with several publications and achievements.

Liu et al. [1] classify taxi drivers into the top and standard drivers according to their income. Based on 3,000 taxi drivers, they observe that top drivers have the special proportion of operation zones, with an optimal balance between taxi travel demand and fluid traffic conditions, while ordinary drivers operate in fixed spots with few variations. Ziebart et al. [2] present a decision modeling framework for probabilistic reasoning from observed context-sensitive actions. Based on 25 taxi drivers, the model is able to make decisions regarding intersections, route, and destination prediction given partially traveled routes. Yuan et al. [3] propose the T-Drive system that relies on an historical GPS dataset generated by over 33,000 taxis in a period of three months, to present the algorithm to compute the fastest path for a given destination and departure time. Chang et al. [4] propose a four-step approach for mining historical data in order to predict demand distributions considering time, weather, and taxi location. They show that different clustering methods have different performances on distinct data distributions. Phithakkitnukoon et al. [5] present a model to predict the number of vacant taxis for a given area of the city using a naïve Bayesian classier with developed error-based learning algorithm and mechanism for detecting adequacy of historical data. With 150 taxi drivers, they achieve an overall error rate of less than one taxi per 1x1 km^2 area. Yuan et al. [6] built a probabilistic recommender for both the taxi drivers and passenger, considering the knowledge of passengers' mobility patterns and taxi drivers' pick-up behaviors, from a dataset of 12,000 taxis. Zheng et al. [7] detected flawed urban planning using GPS trajectories of 30,000 taxis. As result of

ACM LBSN '11, November 1, 2011. Chicago, IL, USA
Copyright © 2011 ACM ISBN 978-1-4503-1033-8/11/11...$10.00

the study, the authors present the region pairs with traffic problems and the linking structure among these regions Qi et al. [8] investigates the relationship between regional pick-up and drop-off characteristics of taxi passengers and the social function of city regions. They develop a simple classification method to recognize regions' social function which can be break in Scenic Spots, Entertainment Districts and Train/Coach Stations.

3. SPATIOTEMPORAL ANALYSIS

The exploratory analysis is a useful tool to identify emerging patterns and obtain a better understanding of the variables that model the system. In this section, we will describe the source dataset, previous work and explore the following aspects: spatial relationships between pick-up and drop-off locations; analysis of the movement of taxis between services and the impact of area type characterized by Points of Interest (POI) to the taxi service.

3.1 Dataset

For the present study we use a database with more than 10 million taxi-GPS samples from August through December in 2009, collected in Lisbon, Portugal by GeoTaxi. For study purposes, only pick-up and drop-off locations and timestamps are considered, which correspond to 177,169 distinct trips. A data cleaning process was applied, removing trips with less than 200m and more than 30km. Data was collected from 217 distinct taxis, which account for nearly 15% of taxis in Lisbon area.

Weather conditions for the period under study were retrieved from Weather Underground and a collection of 10,954 Points Of Interest, grouped into eight categories (Services 16.96%, Recreation 14.78%, Education 20.84%, Shopping 4.65%, Police 2.81%, Health facilities 6.58%, Transportation 2.28%, Accommodation 1.81%), was provided by Sapo Maps.

The area of study encompasses the Lisbon council that consists of 53 parishes, an area of around 110 km^2, and a population of 800,000 habitants. The city downtown is the central area, which includes the oldest and smallest parishes with greatest population density, touristic, historic and commercial areas, and the interface for several public transportation services (bus, metro, train and ferry). Moving from the city center there are larger area parishes with lower population density, which are characterized by residential areas surrounding business areas and major infrastructures (e.g. airport, industrial facilities). For the analysis, we model the Lisbon map with grids of 0.5x0.5 km^2.

3.2 Spatial relationships between pick-up and drop-off locations

In order to understand how the pick-up and drop-off location areas relate we compute the number of trips between every two possible locations. The result is shown in figure 1 where the thickness of the line represents this intensity (A, City downtown; B, Airport; C, Train Station; D, Train Station; E, Ferry dock; F, City center; G, Univ. Campus; H, Commercial Area; I, Residential).

Strong relations can be observed in links B-C, D-E, D-A, A-F, and F-B. All those locations are characterized by some public transportation modality. B is the access to the airport, C and D are trains stations, E is a ferry dock, A and F are bus stops zones. From this observation, we hypothesize that the taxi service is often used as a bridge between public transportation modalities. It is also important to point out that the locations A, C and F (some of the frequent pick-up or drop-off locations) give access to services and commercial areas.

3.3 Downtime analysis

Another possibility is that the taxi drivers may want to improve their income by targeting the above-mentioned locations. To better understand that we need to consider what happens in between services (i.e. downtime – time spent looking for next pick-up).

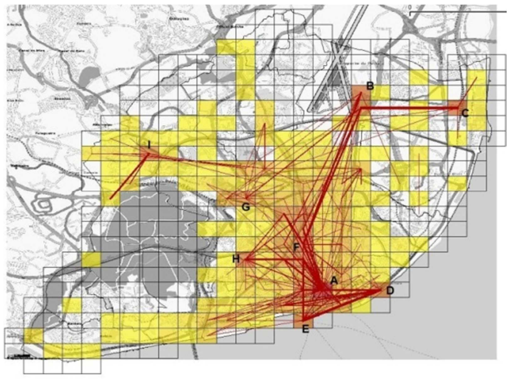


Figure 1. How strongly connected locations are, according to taxi services.

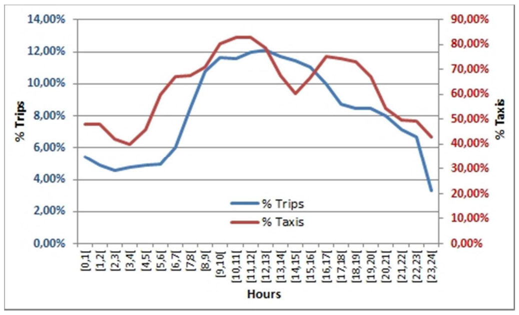


Figure 2 Distribution of taxi trips throughout the day (blue) and number of taxis in service (red).

Figure 2 shows the variation in trips made by and the number of taxis in service throughout the day. Figure 3 shows the average time spent and distance traveled during downtime. In the early AM hours (12 a.m. to 7 a.m.), due to the low amount of taxis in service, the average downtime and distance traveled searching for new passengers are relatively high. The average downtime remains almost constant during 10 a.m. to 10 p.m. There is a sudden drop in downtime at 10 p.m. but a rise of distance traveled. The lower number of taxis in service as well as potential passengers during this late hour presumably causes longer time spent searching for pick-up. Both distance traveled and downtime appear to follow exponential distributions – as shown in figure 4.

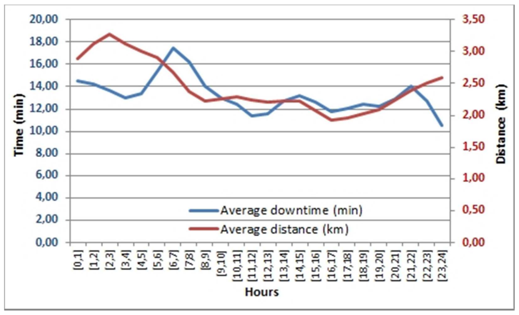


Figure 3. Average downtime (blue) and distance traveled.

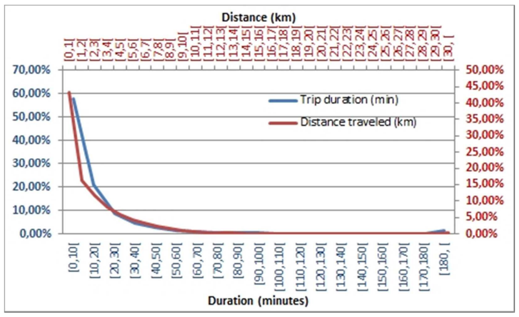


Figure 4. Distribution of distance traveled (red) and time spent (blue) during downtime.

In figure 5, we can see the relationship between the distance traveled during downtime, and the resulting service distance with corresponding average income. A higher distance traveled during downtime does not guarantee a more profitable service.

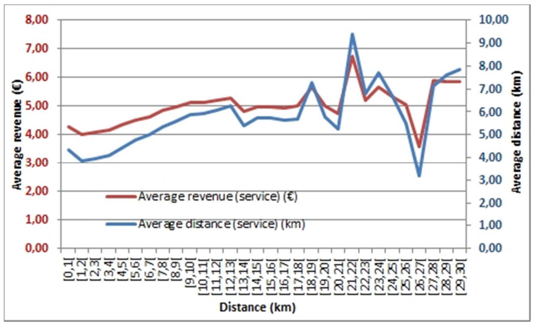


Figure 5. Variation in service distance (blue) and income.

Figure 6 presents the areas with high (red) and low (yellow) average distance traveled when taxis search for new pick-ups and the relationship between the previous drop-off locations and the following pick-up locations. The areas away from the city center show higher average distances traveled between services, whereas in downtown the distances traveled are relatively smaller. By the same token, strong relationships between adjacent locations are observed in urban areas while in suburban areas strong links are observed between distant locations. This appears to us that after a drop-off in suburban area, a taxi driver typically heads to locations with higher probability of picking up new passengers (e.g. airport, city center) even if it means to travel a higher distance to the next pick-up location.

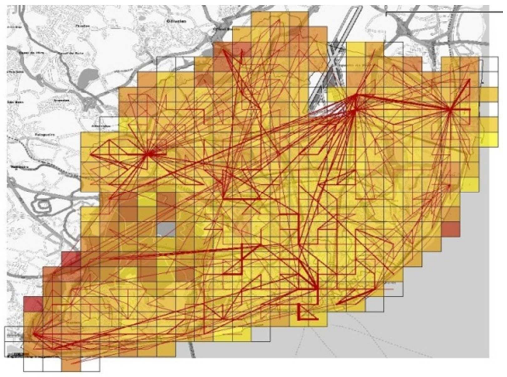


Figure 6. Spatial distribution according to the average distance traveled during downtime (red corresponds to high value) and the relationship between previous drop-off and next pick-up location (line thickness represents strength).

As an overall observation, in order to improve the profit, it is preferable for a taxi driver to wait for passengers in locations related with main public transportation terminals, and not travel great distances to the next pick-up location, unless to return to the aforementioned locations. If the drop-off location coincides with a public transportation terminal it is preferable to wait for new passengers in that location.

3.4 Impact of area type characterized by POIs to the taxi service

By embedding spatial profile like Points of Interest (POIs) onto the map, we can further observe taxi dynamics according to the area characteristic. Figure 7 shows the map with POIs that are grouped in eight different categories, and figure 10 shows the distribution of these categories. One can observe that Education facilities (e.g. kindergarten, university, etc.), Recreation (bar, restaurant, etc.) and Services (e.g. bank, etc.) are the dominant POI categories (which account for over 70%).

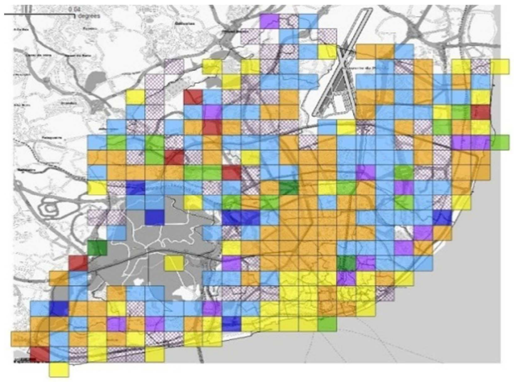


Figure 7. Predominant POI category on each location (colors correspond to classification performed in figure 9).

This POI map allows us to further explore the origin-destination area type. Figure 8 shows that Services and Recreation are the most frequent drop-off area types – independent of the pick-up location. Transportation is the most likely drop-off area if the pick-up is Shopping, Transportation, Health, or Education. Education, as the most likely drop-off area, is mainly connected to pick-up locations from Health area. This observation is an indication that the area type can be a possible predictive attribute to consider when looking for taxi demand.

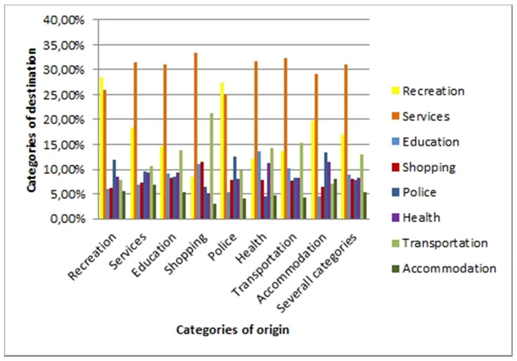


Figure 8. Origin-destination distribution according to area type characterized by POI categories.

4. PREDICTABILITY ANALYSIS

Contrary to other public transportation modes, the taxi movement dynamically adapts to the flow and the need of the city. In previous work [6] we carried out a spatiotemporal analysis of trips made by taxis and found that day of the week, time of the day, and weather condition are promising features in predicting taxi volume. In this work, we aim to explore the predictability of taxis given the current drop-off location. We have observed that area type characterize by POI can potentially be used here along with other aforementioned features used in the previous work. Here we apply a simple probabilistic approach.

We apply a naïve Bayesian classifier for our study of the predictability. The classifier simply applies the Bayes' theorem with independence assumption [9]. The objective is to compute the likelihood of each possible grid cell destination (*Y*) given the hour of the day (*T*), day of the week (*D*), weather condition (*W*) and area type (*I*). The conditional probability can be formulated as follows:

$$P(Y = y_i | T, D, W, I) = \frac{P(Y = y_i) P(T, D, W, I | Y = y_i)}{P(T, D, W, I)} \tag{1}$$

where T = {1, 2, …, 24}, D = {Sunday, …, Saturday}, W = {Sunny, Cloudy, Rainy}, and I = {Services, Recreation, Education, Shopping, Police, Health, Transportation, Accommodation}. The prediction is based on the *maximum a posteriori probability* (MAP) decision rule:

$$\begin{aligned} y_{MAP} &= \arg\max_{y_i \in Y} P(Y = y_i | T, D, W, I) \\ &= \arg\max_{y_i \in Y} P(Y = y_i) P(T, D, W, I | Y = y_i) \\ &= \arg\max_{y_i \in Y} P(Y = y_i) \\ &\quad \prod_i P(T | Y = y_i) P(D | Y = y_i)\, P(W | Y = y_i) P(I | Y = y_i) \end{aligned} \tag{2}$$

Based on 5-folds cross validation, we are able to predict (for each pick-up) the next destination at about 5%. To further investigate on this predictability aspect, we examine the *predicted list* [10] – the list of the most likely destinations where the top of the list contains more likely destinations than the ones lower on the list.

Figure 9 shows that accuracy rate varying with the length of the predicted list. The list must grow to about 90 possible destinations in order to predict the correct destination at a high accuracy (70%). This reflects the randomness of the taxi flow in the city and the fact the variables could not be independent. Although the previous work [6] has shown a promising result in predicting vacant taxi volume at a large scale, predicting taxi individually shown in this work here appears to be rather difficult and not suitable for simple approaches.

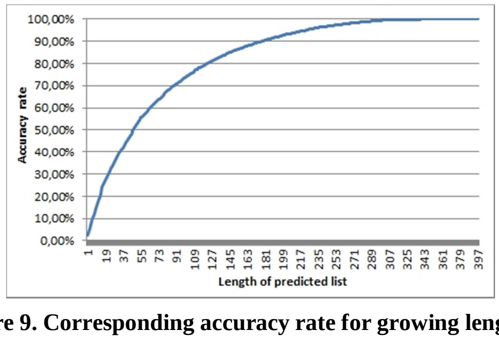


Figure 9. Corresponding accuracy rate for growing length of the predicted list.

If we modify the objective to compute the likelihood of each possible area type destination given the same variable (hour of the day, day of the week, weather conditions and area type) we are able to predict the next area type destination at about 47%, an expected improvement since we reduced the class domain.

5. CONCLUSIONS

By analyzing the taxi-GPS traces from Lisbon, Portugal, we are able to identify the link between pick-up and drop-off locations, the behavior during downtime – time spent searching for next pick-ups – and the impact of area type characterized by POIs to the taxi service. We observed strong links between public transportation terminals and taxis tend to avoid making long trips to suburban areas for pick-up. We also verified that Transportation is the most likely drop-off area if the pick-up is Shopping, Transportation, Health, or Education, and Education is the most likely drop-off area if the pick-up Health area.

Our predictability analysis shows that individual taxi trips are relatively random. With Bayesian approach given time of the day, day of the week, weather condition, area type, and the current pick-up location, only 5% of all trips are predictable. Being able to accurately predict taxi flow is important and a challenging problem, which we will address it further in our future work. Other topics for our future studies include the commuting pattern between multimodality as suggest by the exploratory analysis.

6. REFERENCES

[1] Liu, L., Andris, C., Bidderman, A., Ratti, C.: Revealing taxi drivers mobility intelligence through his trace. Movement-Aware Applications for Sustainable Mobility: Technologies and Approaches, (2010), 105-120.

[2] Ziebart, B.D., Maas, A.L., Dey, A.K., Bagnell, J.A.: Navigate like a cabbie: probabilistic reasoning from observed context-aware behavior. In: UbiComp '08: Proc. of the 10th Int. Conf. on Ubiquitous computing, New York, USA, ACM (2008), 322-331.

[3] Yuan, J., Zheng, Y., Zhang, C., Xie, W., Xie, X., Huang, Y.:T-Drive: Driving Directions Based on Taxi Trajectories, in Proc. ACM SIGSPATIAL GIS 2010, Association for Computing Machinery, Inc. 1 (2010), 99-108.

[4] Chang, H., Tai, Y., Hsu, J.Y.: Context-aware taxi demand hotspots prediction. Int. J. Bus. Intell. Data Min. 5(1) (2010).

[5] Phithakkitnukoon, S., Veloso, M., Bento, C., Biderman, A., Ratti, C.: Taxi-Aware Map: Identifying and predicting vacant taxis in the city. In Proc. AmI 2010, First Int. Joint Conf. on Ambient Intelligence (2010), 86-95.

[6] Yuan, j., Zheng, Y., Zhang, L., Xie, X., Sun, G,: Where to Find My Next Passenger? 13th ACM Int. Conf. on Ubiquitous Computing (UbiComp 2011), China (2011).

[7] Zheng, Y., Liu, Y., Yuan, J., Xie, X.: Urban Computing with Taxicabs. 13th ACM Int. Conference on Ubiquitous Computing (UbiComp 2011), China (2011).

[8] Qi, G., Li, X., Li, S., Pan, G., Wang, Z., Zhang, D., Measuring Social Functions of City Regions from Large-scale Taxi Behaviors. In PerCom- Workshops 2011, pp. 21-25, Seattle, USA, (2011).

[9] Mitchell, T.M.: Machine Learning. McGraw-Hill, New York, (1997).

[10] Phithakkitnukoon, S. and Dantu, R.: CPL: Enhancing Mobile Phone Functionality by Call Predicted List. The 3rd Int. Workshop on MObile and NEtworking Technologies for social applications (MONET'08), pp. 571-581 (2008).

Collaborative Activity Recognition via Check-in History

Defu Lian[1, 2] Xing Xie[2]
[1]School of Computer Science and Technology, University of Science and Technology of China
[2]Microsoft Research Asia, Building 2, No. 5 Dan Ling Street, Haidian District, Beijing 100080,China
liandefu@mail.ustc.edu.cn, xingx@microsoft.com

ABSTRACT

With the growing number of smartphones and increasing interest of location-based social network, check-in becomes more and more popular. Check-in means a user has visited a location, e.g., a Point of Interest (POI). The category of the POI implies the activities which can be conducted. In this paper, we are trying to discover the categories of the POIs in which users are being located (i.e., activities) based on GPS reading, time, user identification and other contextual information. However, in the real world, a single user's data is often insufficient for training individual activity recognition model due to limited check-ins each day. Thus we study how to collaboratively use similar users' check-in histories to train Conditional Random Fields (CRF) to provide better activity recognition for each user. We leverage k-Nearest Neighbors (kNN) and Hierarchical Agglomerative Clustering (HAC) for clustering similar users and learn a separated CRF for each cluster on the histories of its users. As for similarity, the first metric involves linear combination of three types of user factors attained by matrix decomposition on User-Activity, User-Temporal and User-Transition matrices. The second metric between two clusters can be the cosine similarity between weights of CRF corresponding to these two clusters. By the initial experiment on real world check-in data from Dianping, we show that it is possible to improve the classifier performance through collaboration and that the first similarity metric is not good to find the real neighbors.

Categories and Subject Descriptors

I.5.2 [**Pattern Recognition**]: Design Methodology - *Classifier design and evaluation*; I.2.6 **[Artificial Intelligence]**: Learning - *Knowledge acquisition*.

General Terms

Algorithms, Design, Experimentation

Keywords

Location-Based Services, Location-Based Social Network, Activity Recognition, Location Naming, Place Annotation, Point of Interest, Check-in

1. INTRODUCTION

With the increasing availability of smart phone, diverse means of localization, and growing interest of social network, a combination of these techniques—Location-Based Social Network, e.g., Foursquare[1], Jiepang[2], Facebook Place[3], and Dianping[4], has attracted plenty of attention. These LBSNs allow users to establish cyber relationship, issue comments, upload photos, share their tips and experiences, and synchronize with other social network to expect more social interactions. These functions are built on user's manually check-in at their real locations. However, check-in activities are troubling and weird since users need take out phones and wait for receiving GPS and select one of nearby Points of Interest, thus sometimes users are not willing to check-in. If phones can automatically check-in at users' real location after users start the LBSN application, all the above concerns can be eliminated. Nevertheless, this problem is quite challenging due to the large number of Points of Interest (POIs) and the high density of GPS. Thus in this paper, we take a step back to consider the **Activity Recognition**, which predicts activities which a user is performing given the location, time, identification and check-in history of the user (Because the category of the POI implies the activities which can be performed, we don't differentiate them in this paper). Accurate activity recognition can be useful in two aspects: it assists to check-in at the accurate POI just as query classification can help to refine the ranking of documents; check-in at only activity or category is one way to satisfy the growing need of privacy preservation.

Activity recognition is in nature with the sequential property, which means the current activity depends previous activity based on first order Markov property, and has been modeled by Conditional Random Field [2][3][4][6]. However, since users don't exhibit fully consistent check-in patterns, it is inappropriate to treat each user equally by mixing their histories in the training data. On the contrary, to create an activity recognition model for each user is also inappropriate, since the success of training such model relies on having sufficient check-in data from each user, which is difficult to satisfy in the real world due to his limited check-ins each day. However, some users may behave similarly given the similar contexts. Therefore, in order to try alleviating the data sparseness problem, we resort to collaborative technique, which groups similar users' data for training. [10] has first proposed collaborative activity recognition problem in user-dependent aspect model and reported that collaborative activity recognition outperforms other models and overcome the data sparseness problem to a certain extent compared with model trained with a single user history. [5][9] proposed the similar problem and explicitly modeled user similarity and weighted their contributions to other users by their similarity. However, they all didn't address the activity relation learning problem under the

[1] www.foursquare.com

[2] www.jiepang.com

[3] www.facebook.com/places

[4] www.dianping.com

collaboration, which is our concern. When relation learning is applied, the data sparseness will be server for that transition weights between activities are also included. Therefore, in this paper, we consider how to integrate CRF with user factors.

The most challenging problem of collaborative activity recognition lies in how to formulize model with user factors. We take the similar approach as [5] to only consider CRF as a black box. After computing the similarity between users, we can find k Nearest Neighbors given a user and alternatively employ Hierarchical Agglomerative Clustering (HAC) to cluster users and then mix their histories for training a separate CRF model. As for similarity, [7] determines the similarity between users based on maximal travel match between their category-based location histories. [5] proposed three types of similarity functions—physical, lifestyle, and sensor data similarity and build a separate model for each similarity function. However, their similarity measurements cannot be directly applied to our problem. Thus we try two approaches to model user similarity. First, based on users' histories, we define User-Activity, User-Temporal and User-Transition matrices and then get the user factors by matrix decomposition on them and then compute the similarity between users based on these factors. Second, we first train CRF model for each user and weights of features are considered as their profiles and then we can define user similarity between their profiles. When many users will be merged into a cluster, this cluster will be considered as a new user to substitute or represent for these users.

Our contributions are summarized as follow:

- We first address the collaborative relation learning problem and propose the user similarity based modeling solution.
- By conducting initial experiment on the real world check-in data from Dianping, we show that it is possible to improve the classifier performance through collaboration and that the first similarity metric is not good to find the real neighbors.

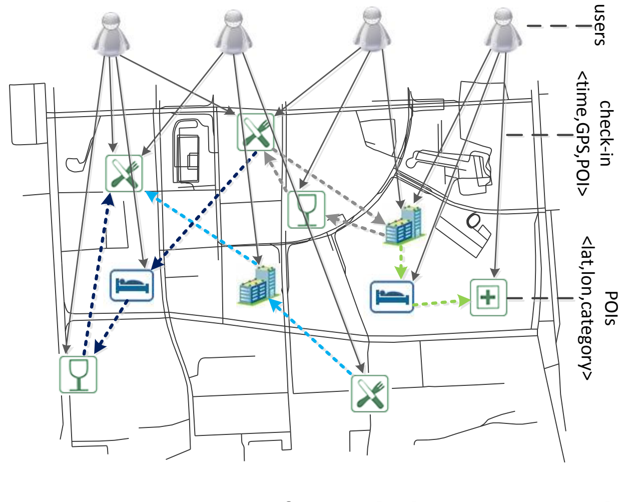


Figure 1 Representation of user check-in in LBSN. The category of Check-in POI implies activities which can be conducted.

2. DEFINITION

2.1 Check-in History

Figure 1 illustrates the representation of fragments of check-in histories from 4 users. We denote U and P as the set of users and POIs. Users and POIs are connected through a set of check-ins $Cs = \{c = \langle u, t, g, p\rangle | u \in U, t \in T, g \in G, p \in P\}$, where T and G are a set of time stamps and GPS readings respectively when user u check-in at POI p. The POI p is represented by a quadruple $p = \langle name, lat, lon, categroy\rangle$. Let check-in history of the user u $Cs_u = \{c_1, c_2, \dots, c_{N_u}\}, where\ c_i \in Cs\ and\ c_i.t < c_{i+1}.t$. N_u indicates the number of check-ins of user u. POIs are connected if they are traveled consecutively by the same user. Edge e can be represented by $e = \langle p_1, p_2, \{\langle u, n\rangle\}\rangle$ which means that user u has visited p_1 and p_2 consecutively n times. Since we conduct activity recognition in the paper, only the category attribute of the POI is taken into account and thus each check-in c is transferred into an activity $ac = \langle u, t, g, a\rangle|_{u=c.u \wedge t=c.t \wedge g=c.g \wedge a=c.p.category}$. The meaning of As_u is similar to Cs_u except that the category has replaced the POI. Based on As_u, we can build the profile of the user u. Alternatively, CRF model can be trained and its weight values can be considered as the profile of u. When similar users are merged as U^c, their histories are also merged and denoted as $As_{U^c} = \{As_u | u \in U^c\}$. User subset U^c is considered as a new "user" u'. Single user u can also be considered as user subset U^c only with one element.

2.2 Weight-Based Similarity

Assuming that $\{As_{U^c} | U^c \subset U\}$ is available, where U is the user set, CRF can be applied and then the weight of CRF can be considered as their profiles. Because we have fixed number of features, the weight vector of CRF doesn't change with length of sequence and thus all weight vectors from trained CRF are with the same length. Given two users $U^C{}_1\ and\ U^C{}_2$, we train CRF on their activity histories and get two weight vectors $\boldsymbol{w}_1, \boldsymbol{w}_2$ and define the similarity between these two clusters as the cosine similarity:

$$sim^1(U^C{}_1, U^C{}_2) = \frac{\boldsymbol{w}_1 \cdot \boldsymbol{w}_2}{\|\boldsymbol{w}_1\| \|\boldsymbol{w}_2\|}$$

2.3 Factor-Based Similarity

To facilitate the extraction of Factor-Based user similarity, we first extracted three matrices—User-Activity, User-Temporal, User-Transition from $\{As_{U^c} | U^c \subset U\}$. Assume that there are k user subset $\{U^c{}_1, U^c{}_2, \dots, U^c{}_k\}$ and we have m activities to form the activity set A and thus to compose the $m \times m$ activity relation set $Atran$. Time information is discretized into temporal set T_d with the time discretization function $d(\cdot)$. In the User-Activity matrix, the weight between user subset $U^c{}_i$ and activity A_j is $w_{i,j}{}^1 = \sum_{u \in U^c{}_i} \left|\{ac | ac \in As_u \ \wedge\ ac.a = A_j\}\right|$. In the User-Temporal matrix, the weight between user subset $U^c{}_i$ and temporal index T_d^j is $w_{i,j}{}^2 = \sum_{u \in U^c{}_i} \left|\{ac | ac \in As_u \ \wedge\ d(ac.t) = T_d^j\}\right|$. In the User-Transition matrix, the weight of user subset $U^c{}_i$ and activity transition $Atran_j$ is $w_{i,j}{}^3 = \sum_{u \in U^c{}_i} \sum_{m=2}^{m=N_u} I(ac_m.a \to ac_{m-1}.a = Atran_j)$, where $I(*)$ is a boolean function indicating whether $*$ is true or false. After performing matrix decomposition on these three matrices, we get three cluster factors $q^1_{U^c}, q^2_{U^c}, q^3_{U^c}$ for each user subset U^c. Factor-Based similarity is defined as follow:

$$sim^2(U^c{}_1, U^c{}_2) = (1 - \alpha - \beta) q^1_{U^c{}_1} \cdot q^1_{U^c{}_2} + \alpha q^2_{U^c{}_1} \cdot q^2_{U^c{}_2} + \beta q^3_{U^c{}_1} \cdot q^3_{U^c{}_2}$$

3. CLUSTERING ALGORITHM

K-NN and HAC are applied in our algorithm to group similar users. As for k-NN, we find k nearest neighbors given a user u and train the CRF on k+1 users' activity history mixture and treat this trained model as u personalized model. The optimal k value for each user can be determined by cross validation or by observation of the variation of similarity value. When the number of check-ins of a user is small, his profile cannot be accurately modeled thus all the other users should be resorted to. When the number of user's check-ins is large, his profile can be accurately modeled by his own data and thus there is no need to resort to other users.

Because HAC is a hierarchical structure for storing the clustering results, we can retrieval all the clusters to which user *u* belongs in the hierarchy and train the CRF model for each of such clusters. As a result, given a input sequence, inferring activity sequence can be based on from overall model which means mixing all data from all users (top level) to single users model (bottom level). Therefore, similar to k-NN, HAC can choose the optimal user grouping means flexibly. The only difference is that the most two similar users are merged as a new "user" for each step until all users belong to a cluster.

4. EVALUTION

4.1 Data Description

Sina Weibo, a microblog service in China, provides APIs for third parties to publish information on its platform. Dianping is one such third party which allows Dianping's users to check-in their locations and publish them on Sina Weibo as microblogs. Each check-in record, appearing as a microblog on Sina Weibo, contains user identification, check-in time, a short URL linking to the homepage of check-in POI and a text status message. In our experiment, we have crawled check-in records located in Beijing from Dianping from Jan 7th 2011 to June 11th 2011 and extracted check-in history of 83 active users with at least 30days histories and segment their histories by day, for that check-in activity is usually daily periodic. For each user, we split the 70% of his data in chronological order as training data, and the other 30% as test data. Figure 2 shows the category distribution of this dataset. It depicts that restaurant occupies the largest portion among all the 7 categories.

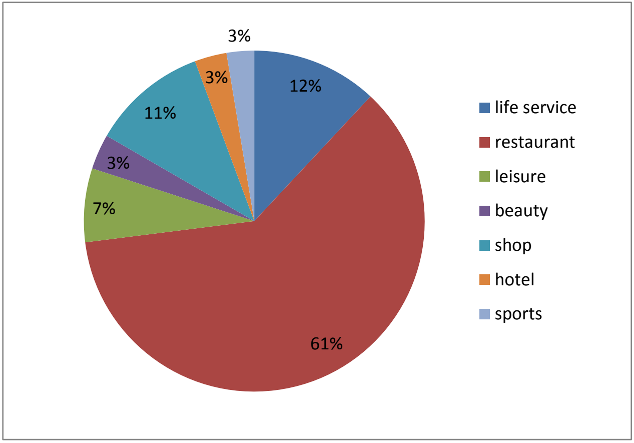


Figure 2 The category distribution of the dataset

4.2 Result and Discussion

As for evaluation metric of CRF classifiers, we use the weighted F1score (W-F1), which is computed by averaging F1 of each category based on its percentage. F1 considers the precision and recall of the test simultaneously [1].

$$F1 = \frac{2 \times precision \times recall}{precision + recall}$$

Weighted precision (W-Precision) and weighted recall (W-Recall) have the similar meanings as weighted F1. For example, if we label all the instances with the restaurant, we can get 61% weighted recall, 37% weighted precision and 46.2% weighted F1.

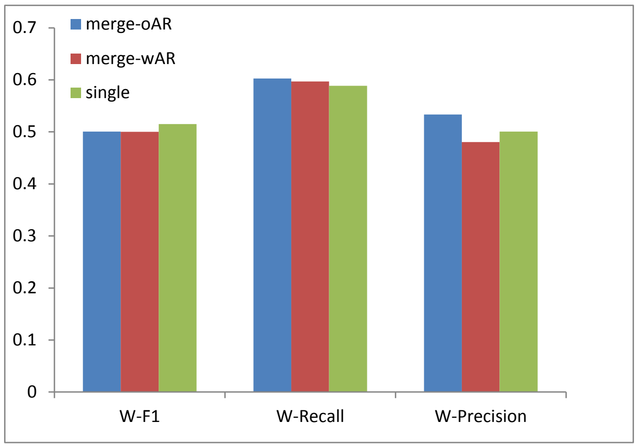


Figure 3 Performance comparison between merge and single model and comparison between with (wAR) and without (oAR) activity relation (AR).

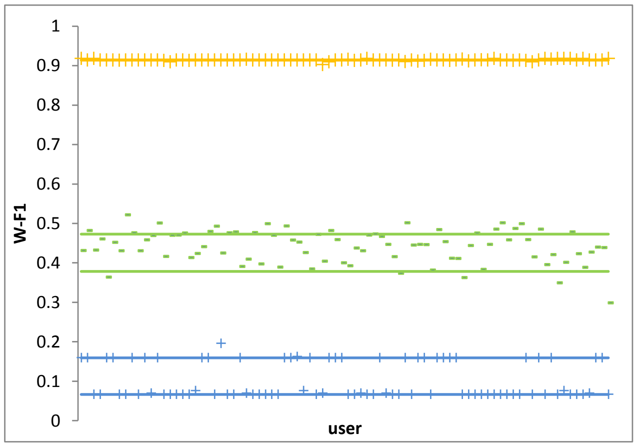


Figure 4 Performance of "Brute-Force" testing. Each of three colors corresponds to each tested user. Each of three users is combined with each of other users to train his model and test on his data and show its performance as one point. Each color has two horizontal lines (the yellow ones are too close to each other). The top one and the bottom are the testing result of the single model and the merge model, respectively.

As we know, different users usually have different visiting patterns, and Figure 3 illustrates that model trained on single user history (**Single** model) is better at performance of testing on his own test set than that trained on all uses' history by treating each user equally (**Merge** model). Nevertheless, it is possible for some users to behave similarly. In order to validate this assumption, we conduct "Brute-Force" testing. We first ranked the users by the number of their check-ins and their performance tested on their own data with the **Merge** model and then selected three users in the top, the middle and the bottom position and assigned to them yellow, green and blue color respectively. For each of these three users, we mix the history of his and each of all users but him and then learn CRF on the mixture and test on his own test set. The result is shown in Figure 4. Each user is associated with two lines, the top of which is the performance tested with **Single** model and the bottom of which is the performance tested with **Merge** model

(Two lines of the yellow group are too close to each other to discriminate). For the blue user, only one of the other users is similar to him due to his few historical data. For the yellow one, its performance almost reaches the ultimate so that it's difficult to make more improvement. For the green user, its single model can be improved by many neighbors, which are shown as dots above top green line. Therefore, overall performance can be improved if we can find real similar users.

Next, we try three similar functions to verify their influence on the performance. These three similarity functions are computed based on user-activity, user-temporal and user-transition matrices. In each of these three matrices, users are represented its row vectors and similarity between two users are defined as the cosine similarity between their vectors. The result is shown in Figure 5. It depicts that we didn't find good similarity functions even for the blue user with lots of good neighbors. Although these similarity functions don't take effect, they have the different impact on the performance. User-transition is more effective than the other two. That is probably because it preserves the sequence property of CRF thus contains more information than activity itself. We have discussed that CRF don't help to improve the performance compared with non-sequence model, therefore, neighbors determined by user-transition probably aren't necessary similar to each other. User-Temporal is the worst similarity function. After we remove all but temporal features in CRF model, all test instances are predicted as restaurant. Although this observation may result from skew category distribution, it also illustrates an important fact that temporal information doesn't discriminate the restaurant from other activities.

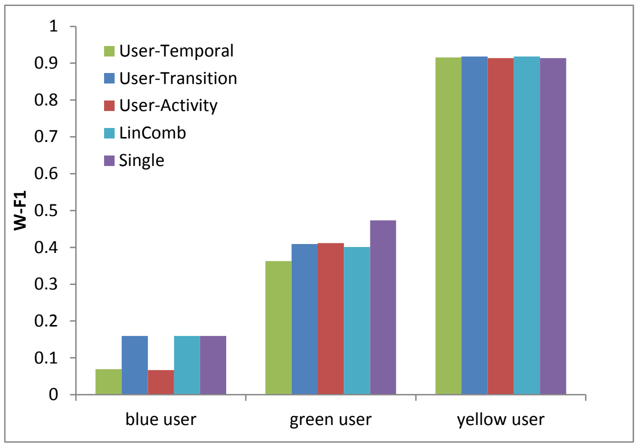


Figure 5 Performance comparison between three similarity functions and linear combination (LinComb) of these three similarities and single model.

Besides, according to our observation, location evidence in CRF model plays an important part in activity recognition. In other words, where the users are located take effect in recognize different activities. That is probably because location is usually dominated by some activities and number of activities which can be performed in a special location is limited. However, number of location evidences depends on the discretization granularity to gps and thus it is easy to suffer from the data sparseness problem. So the study of the impact of location evidences on user similarity will be placed in future work.

The last assumption which we would like to verify is whether sequential learning (e.g., CRF) can make an improvement for activity recognition in the check-in application. Figure 3 shows the comparison result. It shows that the existence of activity relation doesn't have the large influence on the performance. We think that there are two reasons for this observation. First, from Figure 2, the category distribution is over skew, thus the decision heavily depends on the activities with large portion, i.e., restaurant (Skew category distribution results from that Dianping started from restaurant review services). Second, check-in depends on users' feeling and their availability, so they probably forget or weren't willing to check-in when they have been to a place. However, in the first order CRF model, current activity only depends on the previous activity which may be not available in the check-in application.

5. CONCLUSION

In this paper, we first address the collaborative activity relation leaning problem in the activity recognition field and propose the user similarity based solution. By the initial experiment on real world check-in data from Dianping, we discover that it is possible to make an improvement when grouping real similar users and that the factor-based similarity function doesn't take effect in finding the real neighbors. Another two conclusions from the experiments lie in that CRF does not have the large influence on activity recognition on this Dianping dataset and temporal information is not effective to differentiate the restaurant from other activities.

6. REFERENCES

[1] Han, J., Kamber, M. and Pei, J. 2011. *Data mining: concepts and techniques*. Morgan Kaufmann Pub.

[2] Liao, L., Fox, D. and Kautz, H. 2005. Location-based activity recognition using relational Markov networks. In *Proceedings of IJCAI '05*. 773-778.

[3] Liao, L., Fox, D. and Kautz, H. 2005. Location-Based Activity Recognition. In *Proceedings of NIPS '05*. 787-794.

[4] Liao, L. and Fox, D. and Kautz, H. 2007. Extracting places and activities from gps traces using hierarchical conditional random fields. *The International Journal of Robotics Research*. 26, 1 (2007), 487-506.

[5] Lane, N. D., Xu, Y., Lu, H., Hu, S.H., Choudhury,T., Campbell, A.T., Zhao, F. 2011. Enabling Large-scale Human Activity Inference on Smartphones using Community Similarity Networks. In *Proceedings of Ubicomp '11*.

[6] Vail, D. L. and Veloso, M. M. and Lafferty, J. D. 2007. Conditional random fields for activity recognition. In *Proceedings of AAMAS '07.1-8*

[7] Xiao, X. and Zheng, Y. and Luo, Q. and Xie, X. 2010. Finding similar users using category-based location history. In *Proceedings of GIS '10*.442-445.

[8] Zheng, Y., Li, Q., Chen, Y., Xie, X. and Ma, W.Y. 2008 Understanding mobility based on GPS data. In *Proceedings of Ubicomp '08*. 312-321.

[9] Zhuang, J. F., Mei, T., Choi, C. H., Xu, Y-Q, Li, S. P. 2011. When Recommendation Meets Mobile: Contextual and Personalized Recommendation On The Go. In *Proceedings of Ubicomp '11*.

[10] Zheng, V.W., Yang, Q. 2011. User-dependent Aspect Model for Collaborative Activity Recognition. In *Proceedings of IJCAI '11*.

Storing Routes in Socio-Spatial Networks and Supporting Social-Based Route Recommendation

Yerach Doytsher
Technion
Haifa, Israel
doytsher@technion.ac.il

Ben Galon*
Technion
Haifa, Israel
bgalon@technion.ac.il

Yaron Kanza*
Technion
Haifa, Israel
kanza@cs.technion.ac.il

ABSTRACT

Cellular phones and GPS-based navigation systems allow recording the location history of users, to find places the users frequently visit and routes along which the users frequently travel. This provides associations between users and geographic entities. Considering these associations as edges that connect users of a social network to geographical entities on a spatial network yields an integrated socio-spatial network. Queries over a socio-spatial network glean information on users, in correspondence with their location history, and retrieve geographical entities in association with the users who frequently visit these entities.

In this paper we present a graph model for socio-spatial networks that store information on frequently traveled routes. We present a query language that consists of graph traversal operations, aiming at facilitating the formulation of queries, and we show how queries over the network can be evaluated efficiently. We also show how *social-based route recommendation* can be implemented using our query language. We describe an implementation of the suggested model over a graph-based database system and provide an experimental evaluation, to illustrate the effectiveness of our model.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications—*Spatial databases and GIS*

General Terms

Experimentation, Performance

Keywords

Geographic information systems, social networks, geosocial networking, socio-spatial networks, spatial query language, recommendation systems, route search, graph database

*The work of these authors was partially supported by the Israeli Ministry of Science and Technology, Grant 3/6472.

1. INTRODUCTION

Recently, devices that allow tracing the location of people, such as cellular phones and GPS-based navigation systems, have become prevalent. Continuously recording the locations provided by such devices yields the location history of users, and in particular, it allows discovering frequently visited places (stay points) and frequently traveled routes between stay points.

The location history of users is essentially a log of location-time records that are associated to users. Each record specifies that the user has been in the recorded location at the recorded time. Such records can be analyzed to produce *life patterns*. Basically, a life pattern synopsizes the orderliness of visits in different geographic entities. It specifies the frequency of visits of a certain user in a specific location or the frequency of her travel along a certain route. For example, a life pattern P of a user Alice may specify that Alice visits the American University at Washington DC every workday from 9 AM to 5 PM. Another life pattern may indicate that Alice travels via Massachusetts Ave from her apartment near Mitchel Park to the campus of the American University in the morning on every workday.

Since in real life users typically do not behave according to strict patterns (*e.g.*, Alice may skip a day or two of her studies due to various reasons), there should be some flexibility in life patterns and the level of flexibility should be represented. To do so, a *confidence* value is assigned to each pattern. Intuitively, the confidence value of a pattern indicates the likelihood of the pattern to hold in the time frame it relates to. That is, a confidence of 0.9 for the pattern P indicates that in an ordinary working day there is a chance of 90% that Alice will arrive at the American University. Extracting life patters of users from location loggings has been studied by Ye *et al.* [13] and others (see [3, 10, 15]). Thus, we assume in this paper that their techniques are being employed for the generation of the life patterns.

Life patterns can enrich a *social network* by connecting it to a *spatial network*. A social network is a graph where the nodes represent users and the edges represent relationships between users. A spatial network is a graph where the nodes represent geographical entities and the edges represent roads that connect adjacent entities. A *socio-spatial network* is created by combining a social network with a spatial network, and more specifically, by connecting the nodes of the two graphs using *life-pattern edges*, so that each user is connected to the places she frequently visits.

To maintain information on frequently traveled routes, the socio-spatial network should represent routes as nodes. Each route should be related to users who frequently travel along it and should be associated with the road fragments it comprises. Storing such information enables answering queries such as *"find the routes that friends of Alice travel to get to the American University"* or *"find friends of Alice who frequently travel in the morning, along a route that goes near her home and arrives at the American University"*.

In order to exploit socio-spatial graphs, it should be possible to easily and efficiently query the data. A suitable query language should be designed to cope with the structure of the data being a graph, and it should be adapted to graphs in which the nodes are divided into groups—some nodes represent users, some nodes represent geographical entities such as buildings or road segments, and other nodes represent sets of entities, *e.g.,* routes are sets of road segments. Note that in such graph there are several types of edges—edges that represent relationships between users, edges that represent adjacency between geographical entities, life-pattern edges that associate user to geographical entities, and edges that connect routes to the road segments they comprise, and to their start and end locations. The query language should be able to discern between the different types of nodes and to distinguish between different types of edges, according to the query. It should also provide the ability to express spatial conditions such as finding objects in a certain area.

Edges in the social network may indicate similarity among users. This similarity may relate to travel behavior that is extracted from the routes users have chosen, or it may be defined according to travel preferences that are formulated using a query over the socio-spatial graph. In either case, the network can be used for *social-based recommendation* of routes. In a social-based route recommendation, a user u provides source and target locations and the goal is to find routes from the source to the target, that are frequently traveled by users who are related to u in the social network. Our model and proposed query language can facilitate social-based recommendation by providing a generic framework to define the similarity among users and the ranking of the recommended routes.

In this paper we present a model that supports the representation of frequently-traveled routes in a socio-spatial network. We provide a query language that is based on graph-traversal and ordering operations to expedite the formulation of queries over the network. We show how the query language can be used to define similarity among users according to the routes they travel and we describe the evaluation of social-based route recommendation over a socio-spatial network.

This paper continues a work that was presented in [2]. The previous work has introduced an initial solution for integration of a social network and a spatial network. The main differences between this paper and [2] are the followings. *(1)* The model that was suggested in [2] is not designed to maintain information on routes, and is in general, a simpler and less expressive model than the model presented in this paper. *(2)* A query language that was presented in [2] is based on set operations and not on graph traversal. It is, thus, inexpressive enough for queries over routes that are built from road segments. *(3)* The topic of route recommendation has not been addressed in the previous work.

The paper is organized as follows. In Section 2 we formally present our framework. The operators of the query language are presented in Section 3. Using the query language for social-based route recommendation is described in Section 4. An implementation of our approach over a graph-based database system is illustrated in Section 5. In Section 6 we provide an experimental evaluation of our implementation. Finally, in Section 7 we discuss related work and in Section 8 we conclude.

2. FRAMEWORK

In this section we present our framework. We formally define two types of networks—social network and spatial network—and we define a *spatio-social network* that combines the two networks by connecting their nodes using life-pattern edges. We explain how frequently-traveled routes are represented in the network.

Social Network. A social network is a graph whose nodes (also called *users*) represent real-world people and whose edges represent relationships (typically, similarity or friendship relationships) between people. Each user has attributes specifying personal properties of the person it represents. Name and hobbies are examples of personal properties.

Formally, a social network is an undirected labeled graph $N_{social} = (U, F)$, where U is a set of nodes (users) and $F \subseteq U \times U \times L$ is a set of labeled edges. An edge $(u_1, u_2, l) \in F$ is called a *social edge* between u_1 and u_2 and the label l indicates the type of relationship among the users.

Spatial Network. The spatial dataset consists of *basic* geographic entities and *complex* geographic entities. Buildings and roads are instances of basic entities. Routes, which are sets of road segments, and *spatial groups*, which represent collections of geographical entities, are complex entities. A neighborhood is an example of a spatial group since it includes buildings. Note that a complex geographical entity can include other complex entities, *e.g.,* a city includes neighborhoods.

A building is represented as a polygon. A road segment is represented as a polygonal line (polyline) that starts and ends in a junction point. In our model, junction points only occur on the ending points of road segments (*i.e.,* there cannot be a junction inside a road segment), and the intersection of two road segments is always in a junction point.

A *spatial network* is a labeled graph whose nodes represent basic and complex geographic entities—buildings, road segments, routes and spatial groups. The edges represent connections between entities and the labels specify the types of the connections. Edges with the label `include` represent an inclusion of an entity in another entity. A spatial group can include spatial groups and buildings. There are edges with an `include` label from routes to the road segments they comprise. The road segments of a route are numbered and the numbers appear as part of the label. Edges labeled by `include` are directed edges from the entity that includes to the one that is included. Routes also have directed edges with labels `start` and `end` to the locations (buildings or spatial groups) where the route starts and ends, respectively. There is an edge labeled by `touch` between road segments that share a junction. Edges labeled by `leadto` connect road segments to buildings—each road segment is connected to the buildings it leads to.

Formally, a spatial network is a directed labeled graph $N_{spatial} = (O, R, L)$, where O is a collection of geo-spatial objects, representing basic and complex real-world geographical entities, $R \subseteq O \times O$ is a set of edges, and L is a function

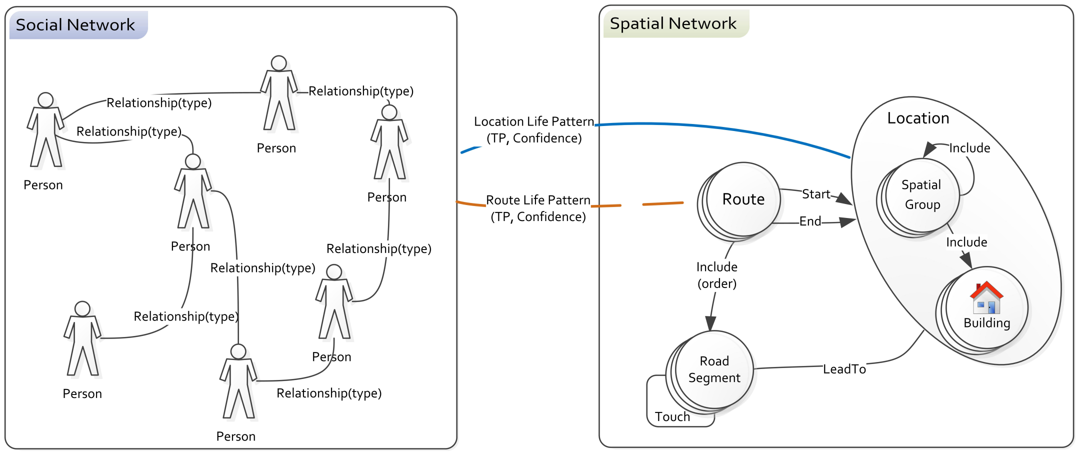


Figure 1: An illustration of a socio-spatial network.

that maps each edge of R to a label, as described above. Objects of O have spatial attributes such as location, and they may have non-spatial attributes such as name or address.

Time Patterns. We use *time patterns* to represent the frequency of repeating events. We chose days as the basic time unit on which time patterns are constructed. Accordingly, we express for a repeating event the hours during the day when the event occurs and the time frame to which the repetition refers. We generate a time pattern as a 4-tuple of *(1) denominator* that expresses the size of the cycle to which the repetition refers, *(2) numerator* that counts the days with respect to the denominator, *(3) start time*, and *(4) end time*. For example, for an event that occurs once a month from 10 AM till 10 PM, the numerator is 1, the denominator is month, the start time is 10 AM and the end time is 10 PM. For an event that occurs 3 times a week, the numerator is 3 and the denominator is week. When the denominator is week, sometimes we specify the exact days during the week when the event occurs. For instance, $[2, 3, 4, 5, 6]$ as a numerator when the denominator is week, refers to the workdays. Formally, a time pattern has the form $TP = (num, denum, st, et)$ where num is the numerator, $denom$ is the denominator, st is the start time and et is the end time during the day.

We define a partial order over time patterns, denoted $TP_1 \le TP_2$, in the obvious way. Given a time pattern $TP_1 = (num_1, denum_1, st_1, et_1)$ and a time pattern $TP_2 = (num_2, denum_2, st_2, et_2)$, we consider r to be the ratio of $denum_2$ and $denum_1$, *e.g.,* for day and week the ratio is 7, for month and year the ratio is 12, etc. Then, $TP_1 \le TP_2$ if $num_1 \le r \cdot num_2$, $st_1 \ge st_2$ and $et_1 \le et_2$. For example, an event that occurs 2 times a week from 13 PM to 16 PM is consider greater than an event that occurs 6 times a month from 14 PM till 15 PM. In such case, we say that the first pattern *entails* the second pattern. We use entailment to define conditions over time patterns.

Life Pattern. Associations between users and geographic entities are expressed by *life patterns*. A life pattern specifies that a certain individual visits a certain geographical entity, or travels along a certain route, at a specified frequency.

Given a set of users U, a life pattern is a 4-tuple $P = (u, e, t, c)$, where $u \in U$ is a user, e is a geographical entity or a route, t is a time pattern, and $0 \le c \le 1$ is a *confidence* value. Intuitively, the confidence c indicates the likelihood of the event defined by the life pattern to hold. It is the percentage of cases in the log of location-time records that support the life pattern. For example, a confidence value 0.9 in a time pattern that defines visits of Alice at the American University in workdays will indicate that only in 90% of the workdays Alice indeed arrives at the University.

Life patterns can be extracted from logs of location-time records, using data mining techniques. For details, see the work of Ye *et al.* [13].

Socio-spatial Network (SSN). A *socio-spatial network* (SSN) is a graph that integrates a social network and a spatial network by connecting the users of the social network to geographic entities and routes, using life-pattern edges.

Let $N_{social} = (U, F)$ be a social network and $N_{spatial} = (O, R, L)$ be a spatial network. Then, the socio-spatial network of N_{social} and $N_{spatial}$ is a graph of the form $SSN = ((U, F), (O, R, L), P)$, where (U, F) is the given social network, (O, R, L) is the given spatial network, and P is a set of life patterns with respect to U and O. We refer to the set P of life patterns as the *bridge* between N_{social} and $N_{spatial}$. In our model, the labels we assign to life patterns that connect users to routes are different from the labels on life patterns that connect users to other spatial entities. This is done to facilitate traversal from users to their routes. An illustration of a socio-spatial network is depicted in Figure 1.

3. QUERY LANGUAGE

To easily and effectively query a socio-spatial network, we propose a query language that is based on graph-traversal operators and basic set operators. We present a set of operators that are the building blocks of the proposed language, and we provide examples to illustrate the expressibility of the language.

In a socio-spatial network there are five different types of nodes—users, buildings, road segments, routes and spatial groups. Typically, operators should be evaluated over a set of nodes that are all of the same type. We refer to a set of nodes where all the nodes are of the same type as *homogeneous*. We refer to two homogeneous sets that have the same type of nodes as *corresponding sets*.

Throughout this section we assume that the SSN has the form $SSN = (N_{social}, N_{spatial}, P)$, and we use the notation of the previous section.

The query language has the form of an algebra over graphs. We refer to it by the name SSGT (Socio-Spatial Graph Traversal). Next, we present the operators of the language.

Select. The *select operator* receives a homogeneous set of nodes and a condition, and it returns all the nodes of the set, that satisfy the condition. It may receive a network, instead of a set, and in such case, the condition is applied to all the nodes of the network. The operator has the option of specifying an ordering attribute, and then, the result is sorted according to this attribute, in an ascending or descending order. The operator is written in the form

`Select(`S`, `C`, [`A` asc | desc])`

where $S \subseteq U$ or $S \subseteq O$ is a homogeneous set of nodes, C is a condition and A is an order-by attribute. (The square brackets refer to optional parameters. To simplify the notation, when an optional parameter is not provided, it is replaced by a `null` value.) The condition C is with respect to the attributes of the nodes, and it may include `and`, `or`, `not` and `like`, with syntax and semantics similar to those of SQL. For example, in a social network N_{social} where users have an academic-status attribute, the following query

`Select(`N_{social}`, status like '%undergraduate student%', null)`

will return all the people in the social network whose status is of an undergraduate student. The query

`Select(`N_{social}`, status like '%undergraduate student%', name asc)`

will return the students sorted in an ascending order, by their names.

Selection of spatial objects can be performed based on spatial conditions. The following two predicates are included in the language for spatial traversal and can be applied when S is a set of spatial objects, *i.e.,* $S \subseteq O$.

1. The `InBoundingBox`(p_1, p_2) predicate receives two points p_1 and p_2 and returns true for objects that are inside the bounding box defined by p_1 and p_2.

2. The `near`(S', d) receives a set S' of spatial objects and a distance parameter d. It returns true for objects that are inside the area defined by a buffer of size d around the objects of S'.

For example, given a spatial network $N_{spatial}$, the query

`Select(`$N_{spatial}$`, near(Select(`$N_{spatial}$`, type='Hospital', null), 100), null)`

will return buildings near hospitals, *i.e.,* whose distance from a hospital does not exceed 100 meters.

Move. Traversal of the graph is, essentially, conducted using the *move operator*. The move operator receives a homogeneous set of nodes and a label ℓ, and it returns all the nodes that are reachable from the given nodes by following an edge with label ℓ. The move operator has the following optional parameters: a condition, an order-by attribute and a scoring formula. When a condition is supplied, only the edges that satisfy the condition are crossed. When an order-by attribute is provided, the answer is returned sorted according to the values of this attribute. The scoring formula is calculated with respect to the attributes of the traversed edges and the computed values are attached as attribute to the returned nodes.

The operator has the form

`Move(`S`, `l`, [`C`], [`A` asc | desc], [`f_s`])`

where $S \subseteq U$ or $S \subseteq O$ is a homogeneous set of nodes, l is a label, C is a condition, A is an order-by attribute and f_s is a scoring function. We sometimes shorten the expression to be `Move(`S`, `l`)`, when all the optional parameters are absent. For example, starting with the social network N_{social}, the following query

`Students` $\leftarrow$ `Select(`N_{social}`, status like '%student%')`
`Move(Students, Route)`

returns all the routes that are frequently traveled by students. It starts by finding student nodes and it assigns these nodes to the `Students` variable. Then, it follows edges with a `Route` label to nodes that represent routes. The query

`Move(Students, Route, confidence>0.5, confidence asc, `$f_{overlaps([1],week,5am,10am)}$`(time-pattern))`

returns the routes that are frequently traveled by students, where the confidence of the life pattern exceeds 0.5. The result is returned ordered by the confidence values of life patterns. The value of the function $f_{overlaps([1],week,5am,10am)}$ is attached to the result. (Function $f_{overlaps([1],week,5am,10am)}$ finds the percentage of the overlap of the time pattern of the followed edge with the time pattern `([1], week, 5am, 10am)`. This can be used by a following operator to rank the results so that routes that are travel during the time defined by the given pattern get precedence over routes that are traveled at different times.)

Note that to simplify the presentation of the expression, we divided it into subexpressions and we assigned the subexpression that defines students to a node-set variable **`Students`**. Node-set variables can be defined and then utilized in expressions that follow the assignment. An unfolding, where each variable is replaced by the expression it represents, recursively, can eliminate the variables and produce a single expression that expresses the query.

Hyper Move. The *hyper-move operator* generalizes the move operator by examining more than one edge when conducting the move. In a hyper-move, the input includes the parameters of move, and in addition, a minimal-support value in the range 0 to 1. So, given a homogeneous set of nodes S of size $|S|$, a label ℓ, and a minimal-support value $0 \leq \nu \leq 1$, the result of a hyper move comprises the nodes n that satisfy the following condition: for at least $\nu \cdot |S|$ nodes in S there is an edge labeled by ℓ from them to n. When the input includes a condition C, we require that from at least $\nu \cdot |S|$ nodes of S we could get to n by following an edge that satisfies C and has a label ℓ. An order-by attribute and a scoring function can be added in the same fashion as for the `Move` operator.

The operator is written in the following form.

`HyperMove(`S`, `l`, `ν`, [`C`], [`A` asc | desc], [`f_s`])`

where ν is the minimal-support value and all the other parameters are similar to those of Move. For example, starting with the social network N_{social}, the following query

```
WineLovers ← Select($N_{social}$, hobby like '%wine%')
HyperMove(WineLovers, Route, 0.2, null, null,
null)
```

finds people who love wine (according to their hobby attribute in the social network) and then finds routes that are frequently traveled by at least 20% of these people. Such query can be used to discover wine routes.

Union, Intersect, Difference. The set operators *union*, *intersect* and *difference* are applied over corresponding sets of nodes in an ordinary fashion, under set semantics, *i.e.*, without repetitions of nodes.
Union(S_1, S_2) $= \{v \mid v \in S_1 \text{ or } v \in S_2\}$;
Intersect(S_1, S_2) $= \{v \mid v \in S_1 \text{ and } v \in S_2\}$;
Difference(S_1, S_2) $= \{v \mid v \in S_1 \text{ and } v \notin S_2\}$.

Subqueries. Some queries express complex conditions that require nested loops. To support formulation of such queries, we include in the language the ability to use subqueries as part of the selection condition of the Select operator. A subquery receives a node of the network as a parameter and it returns a set of nodes. We use the contains predicate to test whether a given set of objects is contained in the result of the subquery or contains it. The following example illustrates the use of subqueries to find routes that go via two specified addresses. The expressions

```
Address1 ← Select($N_{spatial}$, address like 'Old
Georgetown Rd. 9200, Washington DC')
Address2 ← Select($N_{spatial}$, address like
'Pennsylvania Ave. 900, Washington DC')
```

assign to Address1 and Address2 the nodes that represent the geographical entities at the provided addresses.

```
BuildingsInRoute(r) = Union(Union(Move(Move(r,
include),leadto), Move(r,start)), Move(r,end))
```

defines a BuildingsInRoute subquery, which for a given route finds the buildings via which the route goes. By Move(r, include) the query finds the road segments of the route r, and Move(Move(r, include),leadto) returns the buildings that these segments lead to. These buildings are being united with the buildings at the start and end locations of r, using the Union operator. Given the definition of the subquery, the answer is routes r for which BuildingsInRoute(r) contains the locations Address1 and Address2, as formulated in the following expression.

```
Select($N_{spatial}$, type=route and BuildingsInRoute(r)
contains Address1 and BuildingsInRoute(r) contains
Address2)
```

4. ROUTE RECOMMENDATION

We show now how our data model and query language can be utilized to implement social-based route recommendation. In a social-based route recommendation, a user u provides two locations l_1 and l_2, and the goal is to find routes that are frequently traveled by users who are related to u in the social network. This is of particular importance when connections between users indicate similarity in travel preferences. Thus, it is useful to generate a social network from routes in a way that reflects travel behavior.

4.1 Generating a Social Network from Routes

There are different ways to generate a social network from routes. A general framework for such task is by using the query language SSGT. We formulate a query Q over the spatial network and the recorded routes, to defines the connections among the users. This is done by constructing a query Q that for each user u returns a set $Q(u)$ of users, and considering the network in which each u is connected to the users of $Q(u)$.

The following example illustrates a query that can be used to define relationships based on recorded routes. *Connect to u users u' that satisfy the following condition: at least 20 percent of the road segments of the routes of u are frequently traveled by u'.* First, define a subquery that for a user u' returns the road segments traveled by u'.

```
Segments(u') = Move(Move({u'}, 'Route Life
Patten'), 'include')
```

Next, use the subquery Segments with the following predicate: overlaps(S_1, S_2, p) that is satisfied when the overlap between S_1 and S_2 (*i.e.*, the ratio of their intersection to their union) exceeds p.

```
Select($N_{social}$, overlaps(Segments(u'), Segments(u),
0.2), null)
```

A simpler definition that only requires u' to have a route that goes via at least 20 percent of the road segments traveled by u can be formulated as follows. First, find the routes of u.

```
u-Routes ← Move({u}, 'Route Life Patten')
```

Secondly, find the road segments on routes of u.

```
u-Road-Segments ← Move(u-Routes, 'include')
```

Thirdly, find routes that go by at least 20 percent of the segments of u-Road-Segments.

```
OverlappingRoutes ← HyperMove(u-Road-Segments,
'include', 0.2, null, null, null)
```

Finally, find the users of the discovered routes.

```
RelatedUsers ← Move(OverlappingRoutes, 'Route
Life Patten')
```

Note that when the users who are related to u are computed by a query, we can consider the social network as a virtual network and compute it in real time when we need to find the users who are related to u. Since the computation is based on graph traversal, this can be done efficiently. The benefits of real time computation of relationships are: *(1)* we do not need to store the entire network and update it, and *(2)* when relationships are computed, it is with respect to all the newly available data, and we do not need to rely on stale computations.

4.2 Computing a Recommendation

Given a user u and two locations l_1 and l_2, we can compute the route recommendation in the following two ways.

1. *From users to routes (u2r).* In this approach, first we find the users U that are related to u in the social network. Secondly, we find routes that are traveled by users of U. Thirdly, we examine all the routes and return those that go via locations l_1 and l_2.

2. *From routes to users (r2u)*. In this approach, we start by finding all the routes that go via l_1 and l_2. This can be done with the assistance of a spatial data structure. Secondly, we examine each route whether the user that frequently traverses it is related to u, and if not, we discard it.

The computation of recommendation according to these two approaches can be done using SSGT queries. Next, we illustrate this. First, consider computation in the u2r approach. We start with a set that comprises only u, and we move on edges with label 'similar' to users of N_{social} that are similar to u. Then, we move on life-pattern edges to the routes of these users.

`RotueOfSimilarUsers` $\leftarrow$ `Move(Move(`$\{u\}$`, 'similar'), 'Route Life Pattern')`

Next, we define a subquery that for a given route r returns the buildings via which r goes.

`BuildingsNearRoute(`r`) = Union(Union(Move(`r`,'leadto'), Move(`r`, 'start')), Move(`r`, 'end'))`

Now, we get

`RecommendedRoutes` $\leftarrow$ `Select(RotueOfSimilarUsers, BuildingsNearRoute(`r`) contains` $\{l_1\}$ `and BuildingsNearRoute(`r`) contains` $\{l_2\}$`)`

Next, we present computation of route recommendation in the r2u approach. First, we find the routes in the relevant area.

`RoutesInAera` $\leftarrow$ `Select(`$N_{spatial}$`, type=Route and InBoundingBox(`l_1`,` l_2`))`

Secondly, we define a subquery `BuildingsNearRoute(`r`)` that returns buildings near a given route, in the same way as we did in the u2r approach. We find the routes among the routes in the area, that go via the given locations.

`ViaLocations` $\leftarrow$ `Select(RoutesInAera, BuildingsNearRoute(`r`) contains` $\{l_1\}$ `and BuildingsNearRoute(`r`) contains` $\{l_2\}$`)`

We compute `RotueOfSimilarUsers` exactly as in the u2r approach. Finally, the result is provided by intersecting the routes of similar users with routes that go via the specified locations.

`RecommendedRoutes` $\leftarrow$ `Intersection(ViaLocations, RotueOfSimilarUsers)`

5. IMPLEMENTATION

We implemented the proposed model and the SSGT query language. Our goal was to experimentally test the effectiveness of the approach, when storing and querying socio-spatial data.

We used a *graph database management system (GDBMS)* to store the socio-spatial network. A GDBMS is designed to store and manage data whose model is a graph. Consequently, such system provides a natural storage for socio-spatial networks. The system has a storage engine that provides an easy mechanism to store the elements of the graph and access them. This includes, nodes with their attributes, and labeled edges, either as directed or as undirected edges. In our implementation, the labels on edges are being utilized as ordinary labels and also being employed to store time patterns and confidence values of life-pattern edges.

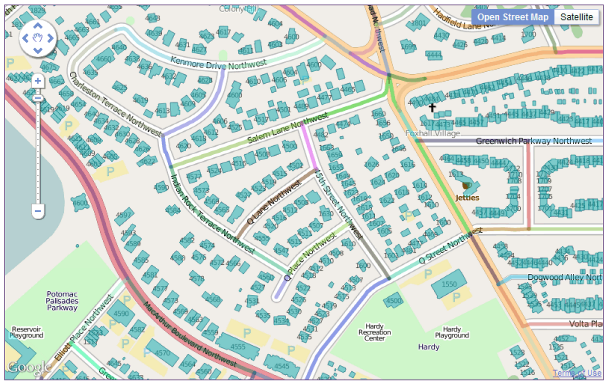


Figure 2: A fragment of a map of Washington DC where road segments are depicted in different colors.

In our implementation, we used Neo4j Version 1.4[1]—an open source graph database-management system, written in Java. It is a transactional system that provides persistence by managing data on the disk. An indexing mechanism is used for efficient retrieval of the nodes.

Two spatial indexes were defined for the elements of the spatial network. One index was defined for buildings and road segments. A second index was defined for routes. This separation was done so that when the number of routes is relatively small, they could be retrieved efficiently regardless of the number of buildings and road segments in the area. More importantly, when the number of routes is large, the retrieval of buildings or road segments should not be affected by this. That, improves the scalability of the system. The spatial indexes are implemented as R-trees.

The system uses non-spatial indexes for retrieval of nodes based on their type and other attributes. Thus, Selection is implemented as retrieval using an index. The `Move` operator is implemented as a direct retrieval using the edges of the graph, since they are stored in the system natively. The `HyperMove` combines graph traversal, as in the implementation of `Move`, with counting to guarantee that for nodes of the result, the percentage of nodes with appropriate edge to them exceeds the minimal support value. The implementation of the set operations is being done in the traditional way.

6. EXPERIMENTS

In this section we present our experiments with the implementation of the query language. The experiments were conducted on a computer with Intel Core 2 Duo T7100 CPU (1.80GHz) having 4GB of RAM, using a Windows 7 Professional operating system.

6.1 Datasets

We conducted our experiments over data that is partially real and partially synthetic. For the geographic network, we used real data of roads and buildings in Washington DC. The data was taken from the Open Street Map project and we analyzed it to partition the roads into road segments, by

[1]See `http://neo4j.org/`.

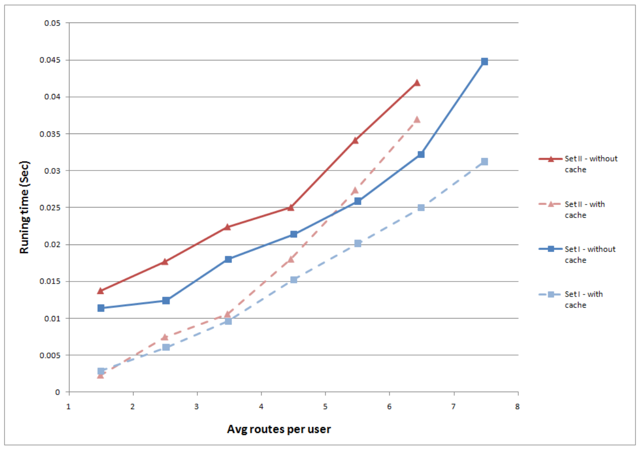


Figure 3: The time it takes to generate a social network from routes.

dividing roads in intersection points. A fragment of the network is depicted in Figure 2. The users of the social network were generated syntactically. The routes were generated by arbitrarily choosing buildings on the map, finding the route between them and assigning them to users. This way, we generated different datasets in which the number of routes per user varied.

We generated two spatial datasets, constructed from areas of different sizes, based on buildings and roads of Washington DC. For each dataset, we assigned different numbers of routes to the users. The routes share many road fragments because in this area routes are dense.

	Buildings	Road segments	Routes	Routes per user (avg.)
Spatial Set I	659	112	929	0.48
			2888	1.50
			4853	2.51
			6722	3.48
			8694	4.50
			10631	5.51
			12526	6.49
			14433	7.47
Spatial Set II	4783	997	960	0.50
			2868	1.49
			4810	2.49
			6697	3.47
			8608	4.46
			10547	5.46
			12414	6.43

Table 1: Two spatial datasets with different sizes and varying numbers of routes, for a social network of 1931 users.

6.2 Results

The first set of experiments we present examines the computation of similarity among users based on shared road segments, as described in Section 4.1. The results are presented in Figure 3. It presents the time (in seconds) it takes to compute the similar users of a given user, as a function of the average number of routes per user. This shows that similarity can be computed in real time, however, computation time is affected by the number of routes per user.

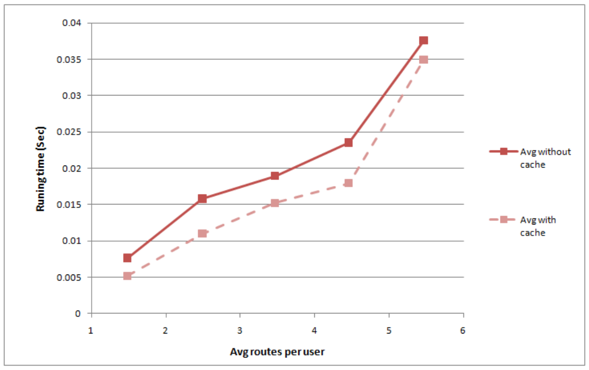


Figure 4: The time it takes to compute a route recommendation in the user-to-route (u2r) approach.

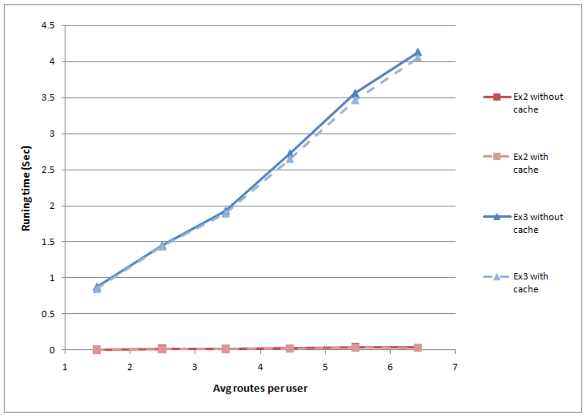


Figure 5: Comparison of the times it takes to compute a route recommendation in the route-to-user (r2u) approach, with and without cache (the upper lines) versus computation in the user-to-route approach (u2r), with and without cache (lower lines).

We used the caching mechanism of Neo4j to test the effect of the cache on the performances. We tested each query for two cases. In the first case, the cache is empty at the beginning of the test (the system has been shutdown and restarted, before the test). In the second experiment, the results are measured after the system has run for a while and the queries were executed several times. The results are illustrated in Figure 3.

The second experiment is a computation of a route recommendation over a social network, where the edges indicate route-preference similarity. We consider a computation as described in Section 4.2. In Figure 4 we show the times of computing a route recommendation using the user-to-route (u2r) approach. (Times are in seconds and we show them as a function of the average number of routes per user.) We can see that the computation is efficient and barely affected by the use of a cache. Running time is increasing linearly as a function of the number of routes per user.

In a third experiment, we compared the u2r approach with the route-to-user (r2u) when computing a route recommendation. The results are presented in Figure 5. We can see that u2r is significantly more efficient than r2u. The reason to these results is that spatial indexes are not as selective as user similarity, and hence, u2r filters routes earlier than r2u and provides a more efficient computation.

7. RELATED WORK

Several papers studied the problem of how to analyze GPS logs to discover significant locations or frequently visited locations. Ye et al. [13] presented algorithms for discovering life patterns from GPS tracks. Zheng et al. [15] provided algorithms for analyzing GPS tracks in order to find significant points. Alvares et al. [1] showed how to generate from GPS tracks a sequence of stopping points and how to learn about the travels between them from existing geographical information. Lee et al. [10] introduced algorithms for discovering travel patterns and Giannotti et al. [3] showed how to find patterns from repeated visits in certain locations, even when the information refers to different individuals.

Some papers have studied the problem of finding similarity among users based on their visits in significant places or according to shared parts of their traveled routes [9, 11]. Hung et al. [6] showed how to categorize people and create communities based on the routes of people. Jeung et al. [7] presented methods for discovering people who travel in a convoy, from their location tracing. Some papers dealt with finding semantic information from GPS traces [5, 12].

Karimi et al. [8] described a framework for sharing navigation experience where route recommendation is done by human users. Yuan et al. [14] presented a navigation system that is based on the driving history of taxi drivers. Other papers have also presented methods for finding a route between two locations based on driving history of either professional drivers or ordinary drivers [16, 4]. However, their work did not consider the social aspect of the recommendation and cannot be easily adapted to take advantage of knowledge about similarity among users.

8. CONCLUSIONS

In this paper, we introduced a data model, namely socio-spatial network, that facilitates integration of social networks with spatial data, for the case where the spatial data includes information on frequently traveled routes. Various applications, such as city and transportation planning, can benefit from such integration, and in particular from enriching information on frequently traveled routes with knowledge about relationships among users, such as similarity. We presented a query language that is based on graph traversal, and we showed how queries can be used for easily implementing social-based route recommendation. Future work includes testing various definitions of similarity and their effect on the quality of the route recommendation.

9. REFERENCES

[1] L. O. Alvares, V. Bogorny, B. Kuijpers, J. A. F. de Macedo, B. Moelans, and A. Vaisman. A model for enriching trajectories with semantic geographical information. In *Proceedings of the 15th annual ACM international symposium on Advances in geographic information systems - GIS '07*, 2007.

[2] Y. Doytsher, B. Galon, and Y. Kanza. Querying geo-social data by bridging spatial networks and social networks. In *Proceedings of the 2nd ACM SIGSPATIAL International Workshop on Location Based Social Networks - LBSN '10*, 2010.

[3] F. Giannotti, M. Nanni, F. Pinelli, and D. Pedreschi. Trajectory pattern mining. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2007.

[4] H. Gonzalez, J. Han, X. Li, M. Myslinska, and J. Sondag. Adaptive fastest path computation on a road network: A traffic mining approach. In *Proceedings of the 33rd international conference on Very large data bases - VLDB '07*, 2007.

[5] B. Guc, M. May, Y. Saygin, and C. Körner. Semantic annotation of GPS trajectories. In *11th AGILE International Conference on Geographic Information Science*, 2008.

[6] C.-C. Hung, C.-W. Chang, and W.-C. Peng. Mining trajectory profiles for discovering user communities. In *Proceedings of the 2009 International Workshop on Location Based Social Networks, LBSN 2009*, 2009.

[7] H. Jeung, M. L. Yiu, X. Zhou, C. S. Jensen, and H. T. Shen. Discovery of Convoys in Trajectory Databases. *Proceedings of the VLDB Endowment*, 1(1), Feb. 2010.

[8] H. Karimi, B. Zimmerman, A. Ozcelik, and D. Roongpiboonsopit. SoNavNet: a framework for social navigation networks. In *Proceedings of the 2009 International Workshop on Location Based Social Networks*, 2009.

[9] J. Lee, J. Han, and K. Whang. Trajectory clustering: a partition-and-group framework. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, 2007.

[10] J. W. Lee, P. O. Hyun, and K. H. Ryu. Temporal moving pattern mining for location-based service. *Journal of Systems and Software*, 73(3):481–490, 2004.

[11] Q. Li, Y. Zheng, X. Xie, Y. Chen, W. Liu, and W. Ma. Mining user similarity based on location history. In *Proceedings of the 16th ACM SIGSPATIAL international conference on Advances in geographic information systems*, 2008.

[12] K. Xie, K. Deng, and X. Zhou. From trajectories to activities: a spatio-temporal join approach. In *Proceedings of the International Workshop on Location Based Social Networks - LBSN '09*, 2009.

[13] Y. Ye, Y. Zheng, Y. Chen, J. Feng, and X. Xie. Mining Individual Life Pattern Based on Location History. In *2009 Tenth International Conference on Mobile Data Management: Systems, Services and Middleware*, May 2009.

[14] J. Yuan, Y. Zheng, C. Zhang, W. Xie, X. Xie, G. Sun, and Y. Huang. T-drive: driving directions based on taxi trajectories. In *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems - GIS '10*, 2010.

[15] Y. Zheng, L. Zhang, X. Xie, and W. Ma. Mining interesting locations and travel sequences from GPS trajectories. In *Proceedings of the 18th International Conference on World Wide Web, WWW 2009*, 2009.

[16] B. D. Ziebart, A. L. Maas, A. K. Dey, and J. A. Bagnell. Navigate like a cabbie: Probabilistic reasoning from observed context-aware behavior. In *Proceedings of the 10th international conference on Ubiquitous computing*, 2008.

Tag Recommendation for Georeferenced Photos

Ana Silva
ana.da.silva@ist.utl.pt

Bruno Martins
bruno.g.martins@ist.utl.pt

Instituto Superior Técnico, INESC-ID
Av. Professor Cavaco Silva
2744-016 Porto Salvo, Portugal

ABSTRACT

This paper presents methods for annotating georeferenced photos with descriptive tags, exploring the annotations for other georeferenced photos which are available at online repositories like Flickr. Specifically, by using the geospatial coordinates associated to the photo which we want to annotate, we start by collecting the photos from an online repository which were taken from nearby locations. Next, and for each tag associated to the collected photos, we compute a set of relevance estimators with basis on factors such as the tag frequency, the geospatial proximity of the photo, the image content similarity, and the number of different users employing the tag. The multiple estimators can then be combined through supervised learning to rank methods such as RankBoost or AdaRank, or through unsupervised rank aggregation methods well-known in the information retrieval literature, namely the CombSUM or the CombMNZ approaches. The most relevant tags are finally suggested. Experimental results with a collection of photos collected from Flickr attest for the adequacy of the proposed approaches.

Categories and Subject Descriptors

H.2.8 [**Database Applications**]: Spatial Databases and GIS; H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Retrieval models, Selection process*

General Terms

Algorithms, Experimentation

Keywords

Tag Recommendation, Geographic Information Retrieval, Rank Aggregation, Supervised Learning to Rank

1. INTRODUCTION

The rapid popularization of digital cameras and smart mobile phones has lead to an explosive growth of consumer photo collections, including collections of photos that have been automatically georeferenced through the GPS devices that are coupled to many of the modern photo collecting devices. Moreover, photo repositories such as Flickr[1] and Panoramio[2] are nowadays used by millions of persons, as a means to store and share these photo collections.

Although the content-based paradigm of image retrieval has been researched for over a decade [37], it has not been widely regarded as a success. To date, keyword-based search remains the preferred method for users to specify queries over large image repositories. Given this preference over keyword-based queries, textual annotations for the photos (e.g., descriptive tags) are very important to enable efficient and effective search in these vast collections of images. However, the current situation at photo-sharing websites like Flickr or Panoramio is that photos are usually noisily or sparsely tagged, and thus they cannot be effectively retrieved. Previous studies have, for instance, indicated that roughly 50% of the tags from landmark photos stored in Flickr are incorrect [18]. We also have that humans are able to annotate concepts that are not directly captured in the images themselves, making image annotation a subjective process. Thus, an important research challenge is related to the development of automatic methods capable of suggesting tags that are indeed relevant to the visual contents of a given photo, providing the users with an easy input mechanism and expecting a low mental effort associated to the choice of new tags to add. In the related literature, this problem is often referred to as automated image annotation.

In this paper, we present automated methods for annotating georeferenced photos with descriptive tags, exploring the redundancy over the large volume of annotations available at online repositories with other georeferenced photos. The general idea is to aggregate multiple sources of tag relevance, in a way that avoids the noise present in Flickr tags. Specifically, by using the geospatial coordinates associated to the photo which we want to annotate, we start by collecting photos from the Flickr online repository, taken from nearby locations. Next, and for each tag associated to the collected photos, we compute a set of relevance estimators with basis on factors such as the tag frequency, the geospatial proximity of the photo, the image content similarity, and the number of different users employing the tag. The multiple relevance estimators can then be combined through supervised learning to rank methods such as RankBoost or

[1]`http://www.flickr.com/`
[2]`http://www.panoramio.com/`

AdaRank, or through unsupervised rank aggregation methods well-known in the information retrieval literature, such as the CombSUM or the CombMNZ approaches. The tags are then ranked according to the combined scores, and the most relevant tags are finally suggested to the user. Experiments with a dataset of 500 georeferenced photos attest for the adequacy of the proposed methods, showing precision values of 0,410 and 0,261, respectively when suggesting a total of 5 and 10 tags with the approach based on the Coordinate Ascent learning to rank algorithm, or precision values of 0,392 and 0,249 when suggesting a total of 5 and 10 tags with the ComnMNZ rank aggregation method.

The rest of this paper is organized as follows: Section 2 describes fundamental concepts and related work. Section 3 presents the proposed method, detailing the computation of the different relevance estimators and their aggregation with supervised learning to rank approaches, as well as with the CombSUM and CombMNZ rank aggregation methods. Section 4 describes the experimental validation of the proposed method, presenting the test data, the evaluation metrics, and the obtained results. Finally, Section 5 summarizes our conclusions and points directions for future work.

2. CONCEPTS AND RELATED WORK

Automatic image annotation has been a hot topic among multimedia researchers of late. Several tag recommendation systems, particularly for photos shared on repositories like Flickr or Panoramio, have for instance been described in the related literature. These systems are capable of suggesting tags related to a given image according to criteria such as visual content analysis, collaborative filtering or personalization strategies. Authors like Lindstaedt et al. provide an overview on the current state-of-the-art [26].

Candidate tags can for instance be produced by analyzing the correlation between an image and several well-defined concepts. This can be made by relying on classifiers trained to detect concepts like *ocean* or *sunset* over the images, through the usage of similarity metrics based on visual descriptors [6, 3], or by relying on probabilistic methods that infer the probability that a particular concept can be associated to a given image [9, 27]. However, most of these methods rely on training databases that only cover a fixed number of concepts, thus limiting the expressiveness of such tag recommendation systems. More recently, a number of attempts have been made to extract candidate tags from the folksonomic information present at photo-sharing Websites.

Previous studies such as those of Sigurbjörnsson and van Zwol [35] or Garg and Weber [13] have described tag recommendation systems capable of extending the number of tags already assigned to the images that are stored on photo-sharing sites, relying on tag co-occurrence statistics computed with basis on the initial tags provided by a user. Other authors have experimented with methods based on association rule mining for collective tag recommendation [15]. If many images with tags T_1 (i.e., high coverage) are typically also annotated with tags T_2 (i.e., high confidence), then a new image with tags T_1 can also be annotated with tags T_2. Through one of these approaches, it is possible to annotate images with tags corresponding to thin-grained concepts, as long as the corresponding tags are commonly employed by different users for annotating different photos.

Other studies have also explored tag co-occurrence together with content similarity or image concept classifiers, leveraging on the notion that the conceptual distance between text labels is also reflected on the visual similarity between images [38]. For instance Lindstaedt et al. have proposed a tag recommendation method that essentially consists of two complementary steps, in which the first uses a supervised classifier to annotate images with a controlled concept vocabulary, and the second propagates user-contributed tags along visually similar images [25]. The high-level concepts mined through visual classifiers are this way merged together with the thin-grained tags. On the other hand, authors like Kucuktunc et al. [22] or Gul Sevil et al. [34] have proposed systems for recommending additional tags which work by retrieving images that are annotated with the tags and, according to the visual similarity between the image to be annotated and the retrieved images, weight the tags associated with the retrieved images and finally present users with the tags having the highest weights. Other authors still have proposed the usage of graph-based tag recommendation techniques, including variations of the PageRank random walking algorithm that explore the notion that popular users, resources and tags can reinforce each other [17, 27].

Although capable of achieving interesting results, the above methods do not explore a particular feature that is nowadays quite common, namely the proximity between the GPS coordinates associated to the places where the photos were taken. The previous works that are perhaps closest to ours are those of Moxley et al. [31, 19] and Naaman and Nair [32], where the authors also proposed methods leveraging on the geospatial context together with other similarity cues for weighting the candidate tags. These previous works used somewhat heuristic methods for integrating the geospatial context information into a global estimator or tag relevance (e.g., a linear combination of different similarity features, including geospatial proximity, where the individual weights were set empirically [31]). In our work, we propose to use either supervised learning to rank methods for combining the different estimators of tag relevance [28, 23], or unsupervised rank aggregation methods with a strong theoretical foundation on voting theory [4].

Specifically with regard to supervised learning to rank (L2R), we have that both Tie-Yan Liu [28] and Hang Li [23] have presented good surveys on the subject, categorizing the existing algorithms into three groups, according to their input representation and optimization objectives:

- **Pointwise approach** - The L2R task is seen as either a regression or a classification problem. Given feature vectors of each single resource (e.g., each tag) from the data for the input space, the relevance degree of each of those individual resources is predicted with the learned scoring functions. Through these scores we can sort resources and produce the final ranked list. Several pointwise methods have been proposed, including Multi-class Classification for Ranking (McRank) [24].
- **Pairwise approach** - The L2R task is seen as a binary classification problem for pairs of resources, since

the relevance degree can be regarded as a binary value telling which ordering of the resources is better. Given feature vectors of pairs of resources from the data for the input space, the relevance degree of each of those resources can be predicted with scoring functions which try to minimize the average number of misclassified pairs of resources. Several different pairwise methods have been proposed in the literature, including SVM*rank* [16], RankNet [2] or RankBoost [11].

- **Listwise approach** - The L2R task is addressed in a way that takes into account an entire set of resources, associated with a query, as instances. These methods train a ranking function through the minimization of a listwise loss function defined on the predicted list and the ground truth list. Given feature vectors of a list of resources of the data for the input space, the relevance degree of each of those resources can be predicted with scoring functions which try to directly optimize the value of a particular information retrieval evaluation metric, averaged over all queries in the training data [28]. Several different listwise methods have also been proposed, including SVM*map* [40], AdaRank [39] or Coordinate Ascent [30].

In this paper, we made experiments with the application of state-of-the-art supervised learning to rank algorithms available through the RankLib[3] open-source Java package, namely RankBoost, AdaRank and Coordinate Ascent. We modeled the task of selecting relevant tags to associate to photos as a learning to rank problem where we represent the associations between tags and photos as feature vectors containing multiple relevance estimators, and use training data to learn a ranking model capable of sorting tags in relation to a given photo in a way that optimizes the Mean Average Precision evaluation metric.

Concerning rank aggregation, previous studies have reported experiments with methods that take their inspiration on the voting protocols proposed in the area of statistics and in the social sciences. Given M voters (e.g., the different estimators of relevance) and N objects (e.g., the tags), we can see each voter as returning an ordered list of the N objects according to their own preferences. From these M ordered lists, the problem of rank aggregation concerns with finding a single consensus list which optimally combines the M rankings. There are different methods for addressing the problem which, according to Julien Ah-Pine [1], can be divided into two large families:

- **Positional methods** - For each object, we consider the preferences (i.e., the scores) given by each voter, aggregating them through some particular technique and finally re-ranking objects using the aggregated preferences. The first positional method was proposed by Borda, but linear and non-linear combinations of preferences, such as their arithmetic mean or the triangular norm, are also frequently used [10, 1].
- **Majoritarian methods** - Pairwise comparison matrices are computed for the objects, mostly based upon the aggregation of order relations using association criteria such as Condorcet's criterion, or distance criteria such as Kendall's distance. Many other majoritarian methods have also recently been proposed, using Markov chain models [7] or techniques from multicriteria decision theory [8].

[3]`http://www.cs.umass.edu/~vdang/ranklib.html`

Fox and Shaw [10] defined several rank aggregation techniques (e.g., CombSUM and CombMNZ) which have been the object of much IR research since. Both these methods have been used in the experiments reported on this paper.

Flickr data has recently been used in a wide range of geospatial applications [5], given the abundance of direct links between geospatial coordinates (i.e., the locations where photos were taken, automatically acquired through GPS devices in cameras or mobile phones, or manually associated by the authors), calendar dates (i.e., the moments when the photos were acquired) and textual descriptions (i.e., titles, descriptions and tags associated to the photos). Through this paper, we provide additional contributes in this specific area, effectively showing how the different information elements provided by Flickr can be combined to solve the challenging problem of automatic image annotation.

3. RETRIEVING RELEVANT TAGS FROM NEARBY SIMILAR PHOTOS

The methods proposed in this paper for the automatic annotation of georeferenced photos with descriptive tags use the Flickr API[4] as a way to collect candidate tags that are known to be used in the description of photos taken from nearby locations. The general methodology involves four steps, namely (i) collecting the tags from the photos at nearby locations, (ii) computing a set of relevance estimators between the original photo and each candidate tag, (iii) using learning to rank or rank aggregation methods to assign a score to each candidate, and (iv) returning the top ranked tags as the suggested image annotations — see Figure 1.

In the first step, a query is made to the Flickr API in order to obtain all tags associated to the photos whose distance towards the candidate photo is less than k kilometers. In our tests, the k parameter was set to the value of 5 kilometers (i.e., the default value for the Flickr API). A maximum of 500 photos, sorted according to their distance towards the candidate photo, are collected in this step. Although a larger number of photos could be collected, this value is the parameter from the proposed methodology that has the strongest impact on computational performance. In the envisioned applications for the proposed method, giving tag suggestions nearly in real-time is of particular importance.

In the second step, and for each of the collected tags, we compute a set of relevance estimators. The proposed approaches for estimating the relevance of a given tag as an annotation for a given image use numeric scores related to tag usage, geospatial proximity, and visual content similarity. The general intuition behind the different estimators is that photos taken from nearby locations are often similar, and thus we can effectively estimate tag relevance by ac-

[4]`http://www.flickr.net/services/api/`

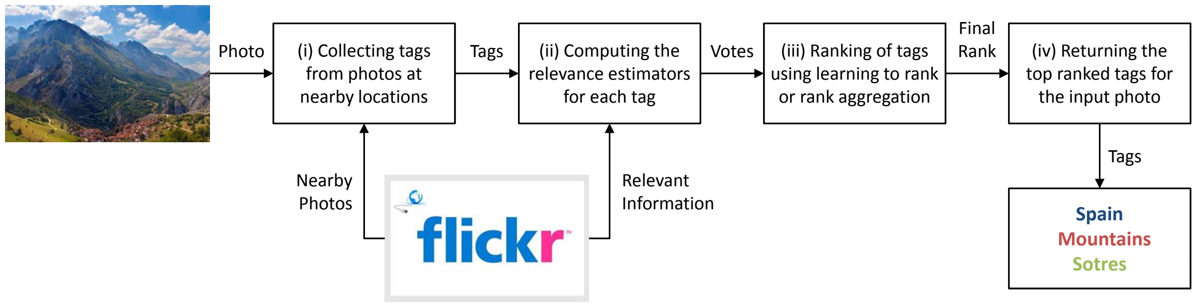


Photo
(i) Collecting tags from photos at nearby locations
Tags
(ii) Computing the relevance estimators for each tag
Votes
(iii) Ranking of tags using learning to rank or rank aggregation
Final Rank
(iv) Returning the top ranked tags for the input photo
Nearby Photos
flickr
Relevant Information
Tags
Spain
Mountains
Sotres

Figure 1: Ranking of tags using learning to rank or rank aggregation.

cumulating evidences from these neighbors. The following estimators were considered in our experiments:

1. Number of nearby photos using the tag, under the assumption that frequently used tags are more likely to be considered relevant annotations.
2. Number of different Flickr users employing the tag for describing nearby photos, under the assumption that tags that are used by a larger portion of the community of Flickr users are more likely to be relevant. If many different persons label nearby photos with the same tags, then these tags are likely to reflect objective aspects of the visual content.
3. Number of Web visits made for the nearby photos using the tag, under the assumption that photos with more visits are more representative of the location and also easier to retrieve through tags. Thus, the tag under evaluation should also be better.
4. The average geospatial distance, expressed in kilometers, between the coordinates of the nearby photos that use the tag and the coordinates of the target photo. The general intuition is that tags associated to photos that were taken closer are more likely to be considered relevant annotations. To compute the geospatial distances, we rely on the great circle method[5].
5. The minimum geospatial distance, in kilometers, between the coordinates of the nearby photos that use the tag and the coordinates of the target photo.
6. The average visual distance between the photo that we wish to annotate and the set of photos taken from nearby locations that are associated to the corresponding tag, under the assumption that some preference should be given to tags that are associated to many photos that are visually similar.
7. The minimum visual distance between the photo that we wish to annotate and the photos taken from nearby locations that are associated to the corresponding tag.

Visual distances are computed according to the naïve method given by Rafael Santos[6], which is based on local color features. The algorithm starts by normalizing the photo dimensions to 300×300 pixels, afterwards computing 25×3 color features corresponding to the average of the RGB values on 25 regions with 30×30 pixels, uniformly sampled from the image. To calculate the similarity between two images A and B, we take the 25 regions from each image, compute the Euclidean distance between the RGB values for the regions, and accumulate the distance values.

The numeric values corresponding to the different estimators of tag relevance are also normalized to a value between zero and one. The relevance estimators corresponding to geospatial and visual distances d are first converted to similarity scores, according to the formula $\frac{1}{1+d}$. The final normalization is then made according to the min-max procedure, which is given by Equation 1.

$$Normalized\_Value = \frac{Value - min}{max - min} \quad (1)$$

Each relevance estimator returns a ranking of candidate tags and, in the third step, we use either (a) supervised learning to rank algorithm or (b) rank aggregation methods to combine the multiple ranking scores.

Concerning the supervised learning to rank methods, we experimented with the usage of the state-of-the-art learning to rank algorithms available through the RankLib open-source Java package, namely:

- RankBoost - This is a pairwise boosting technique for ranking, in which the final learned model is essentially a linear combination of weak rankers [11]. Training proceeds in rounds, starting with all the pairs of photos being assigned to an equal weight. At each round, the learner selects the weak ranker that achieves the smallest pairwise loss on the training data, with respect to the current weight distribution. Pairs that are correctly ranked have their weight decreased and those that are incorrectly ranked have their weight increased, so that the learner will focus more on the hard samples in the next round. The weak rankers can theoretically be of any type, but they are most commonly chosen as a binary function with a single feature and a threshold. This function assigns a score of 1 to a document of which feature value exceeds the threshold, and 0 otherwise. In our experiments, the number of threshold candidates was set to 50 and the number of rounds to train was set to 1000.

[5]`http://en.wikipedia.org/wiki/Great-circle_distance`
[6]`http://www.lac.inpe.br/JIPCookbook/6050-howto-compareimages.jsp`

- AdaRank - The idea here is similar to that of RankBoost, except that AdaRank is a listwise approach [39]. Hence, this method directly maximizes any desired IR metric (e.g., MAP) computed over ranked lists of results, whereas the objective in RankBoost is simply to minimize the pairwise loss. In our experiments, the number of rounds to train was set to 1000.

- Coordinate Ascent - This is a state-of-the-art listwise method, proposed by Metzler and Croft, which uses coordinate ascent to optimize the ranking model parameters [30]. Coordinate ascent is a well-known technique for unconstrained optimization, which optimizes multivariate objective functions by sequentially doing optimization in one dimension at a time. The method cycles through each parameter and optimizes over it, while fixing all the other parameters.

Concerning rank aggregation methods, we experimented with the usage of the CombSUM and the CombMNZ algorithms originally proposed by Fox and Shaw [10], as well as with the Borda voting protocol [4].

Both CombSUM and CombMNZ use normalized sums for the different features. We specifically have that the CombSUM score of a tag t, for a given photo P, is the the sum of the normalized scores received by the tag in each of the k individual rankings, and is given by Equation 2.

$$CombSUM(t,P) = \sum_{j=1}^{k} score_j(t,P) \tag{2}$$

Similarly, the CombMNZ score of a tag t for a given photo P is defined by Equation 3, where r_e corresponds to the number of non-zero similarities between t and P.

$$CombMNZ(t,P) = CombSUM(t,P) \times r_e \tag{3}$$

In the Borda voting protocol, each position in a ranked list j has a different score $\alpha_j(t,P) = m - position(t,P)$, where m is the number of candidate tags and $position(t,P)$ is the position of the tag in the ranked list for P. A combined relevance score is produced for each tag by aggregating the scores from the individual ranked lists. A ranking of tags is then made with basis on the resulting aggregation scores.

$$Borda(t,P) = \sum_{j=1}^{k} \alpha_j(t,P) \tag{4}$$

In the fourth and final step of the proposed methodology, the top ranked tags obtained with the combination of the relevance estimators are suggested to the user. In the experiments reported in Section 4, we analyze the quality of the recommended tags when considering different cut-off positions (i.e., precision at the top 5 and 10 tags).

4. EXPERIMENTAL VALIDATION

The validation of the proposed method was based on photos selected from the IM2GPS dataset[7], which consists in a set of georeferenced photos collected from Flickr. We randomly selected 500 georeferenced photos from the IM2GPS dataset.

[7]`http://graphics.cs.cmu.edu/projects/im2gps/`

Relevance Estimator	MAP	P@5	P@10
Estimator 1	*0,389*	*0,357*	*0,221*
Estimator 2	*0,394*	*0,356*	*0,215*
Estimator 3	*0,377*	*0,352*	*0,209*
Estimator 4	*0,050*	*0,064*	*0,051*
Estimator 5	*0,219*	*0,235*	*0,181*
Estimator 6	*0,030*	*0,036*	*0,026*
Estimator 7	*0,038*	*0,049*	*0,037*

Table 1: Results for different relevance estimators.

These photos have, on average, 6 tags and 120 neighbor photos in Flickr, in a range of 5 kilometers.

For each training photo, a query is made to the Flickr API in order to obtain all tags associated to photos whose distance towards the candidate photo is less than 5 kilometers. A maximum of 500 photos, sorted according to their distance towards to the candidate photo, are collected from Flickr. Both the original photos and their neighboring photos, together with the corresponding metadata, are stored in a PostgreSQL database. Having the data in a relational database facilitated the execution of the evaluation tests.

In order to evaluate the quality of the results, metrics such as Precision at rank position K (P@K) and Mean Average Precision (MAP) were used [29]. The performance metric of Precision at rank position K (P@K), for a ranked list of tags, is defined as follows:

$$P@k(p) = \frac{\#(relevant\ tags\ in\ the\ top\ k)}{k} \tag{5}$$

For a photo p, the $P@k$ metric only considers the top-ranked tags as relevant and computes the fraction of such tags in the top-k elements of the ranked list.

The Mean of the Average Precision (MAP) metric is defined as the mean value of the Average Precision (AP) over all the test photos. The AP is defined as follows:

$$AP(p) = \frac{\sum_{k=1}^{m} P@k(p) \cdot I_k}{\#(relevant\ tags)} \tag{6}$$

In the formula, m is the total number of tags associated with photo p, and I_k is the binary judgment that represents the relevance or non-relevance of the tag at the k-th position relative to the photo p.

Table 1 presents the results obtained for each of the individual relevance estimators, and Table 2 presents the obtained results when using unsupervised rank aggregation methods with different sets of relevance estimators and when returning up to 10 tags for each photo.

The results from Table 2 show that the quality of the recommended tags is improved when using the CombMNZ method to combine the relevance estimators, in comparison to the individual relevance estimators. The best performing method uses CombMNZ to combine 4 different relevance estimators, namely the number of nearby photos using the tag, the number of different Flickr users employing the tag for describing nearby photos, the number of Web visits made for the

	Borda			CombSUM			CombMNZ		
Relevance Estimators	**MAP**	**P@5**	**P@10**	**MAP**	**P@5**	**P@10**	**MAP**	**P@5**	**P@10**
E1, E2	*0,400*	*0,369*	*0,224*	*0,403*	*0,369*	*0,224*	*0,403*	*0,371*	*0,224*
E1, E3	*0,392*	*0,368*	*0,221*	*0,390*	*0,363*	*0,217*	*0,390*	*0,362*	*0,218*
E1, E5	*0,381*	*0,372*	*0,241*	*0,438*	*0,388*	*0,251*	*0,440*	*0,387*	*0,253*
E1, E7	*0,102*	*0,124*	*0,093*	*0,314*	*0,293*	*0,182*	*0,314*	*0,293*	*0,182*
E2, E3	*0,394*	*0,367*	*0,217*	*0,401*	*0,371*	*0,220*	*0,401*	*0,371*	*0,220*
E2, E5	*0,351*	*0,351*	*0,221*	*0,424*	*0,379*	*0,233*	*0,428*	*0,383*	*0,238*
E2, E7	*0,102*	*0,116*	*0,090*	*0,318*	*0,294*	*0,181*	*0,318*	*0,295*	*0,181*
E3, E5	*0,378*	*0,367*	*0,239*	*0,437*	*0,393*	*0,248*	*0,418*	*0,363*	*0,239*
E3, E7	*0,169*	*0,093*	*0,062*	*0,303*	*0,287*	*0,174*	*0,303*	*0,287*	*0,174*
E1, E2, E3	*0,400*	*0,374*	*0,224*	*0,401*	*0,368*	*0,223*	*0,403*	*0,371*	*0,225*
E1, E2, E3, E4	*0,203*	*0,243*	*0,181*	*0,406*	*0,371*	*0,225*	*0,412*	*0,375*	*0,233*
E1, E2, E3, E5	**0,405**	**0,383**	**0,234**	**0,439**	**0,390**	**0,253**	**0,436**	**0,392**	**0,249**
E1, E2, E3, E4, E6	*0,171*	*0,205*	*0,148*	*0,329*	*0,303*	*0,181*	*0,341*	*0,315*	*0,191*
E1, E2, E3, E4, E7	*0,182*	*0,217*	*0,155*	*0,322*	*0,297*	*0,183*	*0,331*	*0,307*	*0,191*
E1, E2, E3, E5, E6	*0,248*	*0,256*	*0,184*	*0,361*	*0,323*	*0,206*	*0,358*	*0,328*	*0,204*
E1, E2, E3, E5, E7	*0,254*	*0,269*	*0,189*	*0,347*	*0,313*	*0,204*	*0,344*	*0,315*	*0,202*
E1, E2, E3, E4, E5, E6, E7	*0,181*	*0,210*	*0,154*	*0,322*	*0,292*	*0,180*	*0,338*	*0,310*	*0,196*

Table 2: Results for different rank aggregation methods when combining different sets of relevance estimators.

nearby photos using the tag, and the minimum distance between the coordinates of the nearby photos that use the tag and the coordinates of the target photo. The estimators related to content-based image similarity achieved the worse results, although we should point that the considered technique for measuring image similarity was relatively naïve. For future work, we would like to experiment with more advanced methods for measuring image similarity.

In a separate experiment, we evaluated the quality of the tag recommendations produced with supervised learning to rank methods, using a leave-one-out cross-validation methodology with five folds. The set of 500 photos is first randomly split into five sets, each with 100 photos. Four of the sets are used to train the supervised ranking models, which are then evaluated over the remaining photos. The averaged scores from the five cross-validation experiments are finally used as the evaluation result. Table 3 shows the obtained results, for each of the considered learning to rank algorithms and for three different sets of features, namely (a) the entire set of relevance estimators described in Section 3, (b) the set that obtained the best results with the CombMNZ rank aggregation method, and (c) an extended set of features that also considered the CombMNZ score corresponding to the aggregation of all features, as an individual feature in the supervised L2R model.

The feature vectors for all tags that are relevant to the photo are considered when training the ranking models. As for the non-relevant feature vectors used to train the learning to rank models, and in order to avoid high imbalancements between the number of relevant and non-relevant tags, we only consider a maximum of 56 non-relevant feature vectors corresponding to tags collected from neighboring photos. Twenty-eight of these feature vectors are chosen at random from the tags associated to neighboring photos, and the other twenty-eight correspond to the non-relevant tags that are hard to discriminate with basis on the considered features, namely tags having the highest values in terms of the seven ranking features that were considered, and that are nonetheless still considered non-relevant.

The results from Table 3 show that learning to rank algorithms achieve significantly better results when compared with the unsupervised rank aggregation methods, effectively leveraging on the entire set of features. The best results were obtained with the Coordinate Ascent listwise learning to rank method, considering the extended set of features. Figure 2 illustrates the obtained results for four different example photos, for the best performing method.

In conclusion, our initial experiments showed that effective tag recommendation for georeferenced photos can be achieved by combining multiple relevance estimators, computed for tags collected from photos taken at nearby locations and stored at online photo repositories such as Flickr.

5. CONCLUSIONS AND FUTURE WORK

This paper presented automated methods for annotating georeferenced photos with descriptive tags, exploring the annotations available at online repositories of georeferenced photos through supervised learning to rank or unsupervised rank aggregation methods that combine different estimators of tag relevance. Experimental results attest for the adequacy of the proposed methods, showing precision values of 0,410 and 0,261, respectively when suggesting a total of 5 and 10 tags with the approach based on the Coordinate Ascent learning to rank algorithm, or precision values of 0,392 and 0,249 when suggesting a total of 5 and 10 tags with the CombMNZ rank aggregation method.

Despite the interesting results, there are many interesting challenges for future work. A particularly interesting idea is related to the usage of more and better features for estimating tag relevancy, including network centrality measures computed for the tags with basis on the associations existing

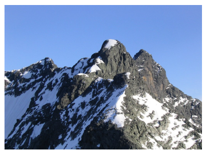
			
Sydney Australia Opera House Opera House	Tibet Kailash Kora Kang Rinpoche China	London England UK City United Kingdom	Jotunheimen Norway Mountains Hurrungane Climbing

Figure 2: Top five tags recommended for four different example photos.

	RankBoost			AdaRank			Coordinate Ascent		
Estimators	**MAP**	**P@5**	**P@10**	**MAP**	**P@5**	**P@10**	**MAP**	**P@5**	**P@10**
All	*0,319*	*0,218*	*0,153*	*0,448*	*0,289*	*0,199*	*0,634*	*0,409*	*0,261*
Best set	*0,311*	*0,212*	*0,146*	*0,562*	*0,364*	*0,225*	*0,635*	*0,410*	*0,261*
Extended set	*0,597*	*0,397*	*0,244*	*0,448*	*0,289*	*0,199*	*0,638*	*0,418*	*0,262*

Table 3: Results for the different learning to rank methods.

between them [17, 27], more advanced content-based image similarity metrics, and metrics derived from latent topics associated to nearby photos [21].

For photos that are not automatically georeferenced, it would be interesting to combine the proposed approach together with one of the previously proposed methods for estimating the geospatial coordinates of new photographs, which often explore the principle that visually similar images are likely to correspond to nearby locations [19, 14, 5].

It would also be interesting to experiment with better unsupervised methods for combining the different relevance estimators. Recent works in the area of information retrieval have described several advanced unsupervised learning to rank methods, capable of outperforming the popular CombSUM and CombMNZ approaches [10]. In IR, rank aggregation is currently a very hot topic of research and, for future work, we would for instance like to experiment with the ULARA algorithm proposed by Klementiev et al. [20].

Finally, we would also like to experiment with the usage of relational learning to rank methods [33], capable of exploring the similarity between tags together with the relevance estimators for each image-tag pair, through a metric such as the Flickr distance [38]. We are particularly interested in extending recently proposed ensemble learning methods [36, 12] for performing relational learning to rank.

Acknowledgements

This work was partially supported by the Fundação para a Ciência e a Tecnologia (FCT), through project grant PTDC/EIA-EIA/109840/2009 (SInteliGIS). We would like to thank our colleagues Pável Calado, Rui Candeias, Ivo Anastácio and Catarina Moreira for their valuable input in the many discussions that we had during the development of this work.

6. REFERENCES

[1] J. Ah-Pine. On data fusion in information retrieval using different aggregation operators. *Web Intelligence and Agent Systems*, 9(1), 2011.

[2] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender. Learning to rank using gradient descent. In *Proceedings of the 22nd international conference on Machine learning*, 2005.

[3] C. B.Yang, M. Dong, and J. Hua. Region-based image annotation using asymmetrical support vector machine-based multiple-instance learning. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2006.

[4] V. Conitzer. *Computational aspects of preference aggregation*. PhD thesis, Carnegie Mellon University, 2006.

[5] D. J. Crandall, L. Backstrom, D. Huttenlocher, and J. Kleinberg. Mapping the world's photos. In *Proceedings of the 18th international conference on World Wide Web*, 2009.

[6] C. Cusano, G. Ciocca, and R. Schettini. Image annotation using SVM. *Internet Imaging V*, 2003.

[7] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar. Rank aggregation methods for the web. In *Proceedings of the 10th International Conference on World Wide Web*, 2001.

[8] M. Farah and D. Vanderpooten. An outranking approach for rank aggregation in information retrieval. In *Proceedings of the 30th Annual Anternational ACM SIGIR Conference on Research and Development in Information Retrieval*, 2007.

[9] S. L. Feng, R. Manmatha, and V. Lavrenko. Multiple bernoulli relevance models for image and video annotation. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2004.

[10] E. A. Fox and J. A. Shaw. Combination of multiple searches. In *Proceedings of the 2nd Text Retrieval Conference*, 1994.

[11] Y. Freund, R. Iyer, R. E. Schapire, and Y. Singer. An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research*, 4, December 2003.

[12] Y. Ganjisaffar. *Tree Ensembles for Learning to Rank*. PhD thesis, University of California, Irvine, 2011.

[13] N. Garg and I. Weber. Personalized, interactive tag recommendation for Flickr. In *Proceedings of the 2008 ACM Conference on Recommender Systems*, 2008.

[14] J. Hays and A. Efros. IM2GPS: Estimating geographic information from a single image. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2008.

[15] P. Heymann, D. Ramage, and H. Garcia-Molina. Social tag prediction. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2008.

[16] T. Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the 9th ACM Conference on Knowledge Discovery and Data Mining*, 2002.

[17] R. Jäschke, L. Marinho, A. Hotho, L. Schmidt-Thieme, and G. Stumme. Tag recommendations in folksonomies. In *Proceedings of the 11th European Conference on Principles and Practice of Knowledge Discovery in Databases*, 2007.

[18] L. Kennedy, S.-F. Chang, and I. Kozintsev. To search or to label?: Predicting the performance of search-based automatic image classifiers. In *Proceedings of the 8th ACM International Workshop on Multimedia Information Retrieval*, 2006.

[19] J. Kleban, E. Moxley, J. Xu, and B. S. Manjunath. Global annotation on georeferenced photographs. In *Proceeding of the ACM International Conference on Image and Video Retrieval*, 2009.

[20] A. Klementiev, D. Roth, K. Small, and I. Titov. Unsupervised rank aggregation with domain-specific expertise. In *Proceedings of the 21st International Joint Conference on Artifical Intelligence*, 2009.

[21] R. Krestel, P. Fankhauser, and W. Nejdl. Latent Dirichlet allocation for tag recommendation. In *Proceedings of the 3rd ACM Conference on Recommender Systems*, 2009.

[22] O. Kucuktunc, S. Sevil, A. Tosun, H. Zitouni, P. Duygulu, and F. Can. Tag suggestr: Automatic photo tag expansion using visual information for photo sharing websites. In *International Conference on Semantic and Digital Media Technologies*, 2008.

[23] H. Li. *Learning to Rank for Information Retrieval and Natural Language Processing*. Morgan & Claypool Publishers, 2011.

[24] P. Li, C. Burges, and Q. Wu. Learning to rank using multiple classification and gradient boosting. In *Proceedings of the 21st Conference on Advances in Neural Information Processing Systems*, 2008.

[25] S. Lindstaedt, R. Mörzinger, R. Sorschag, V. Pammer, and G. Thallinger. Automatic image annotation using visual content and folksonomies. *Multimedia Tools and Applications*, 42(1), 2009.

[26] S. Lindstaedt, V. Pammer, R. Mörzinger, R. Kern, H. Mülner, and C. Wagner. Recommending tags for pictures based on text, visual content and user context. In *Proceedings of the 3rd International Conference on Internet and Web Applications and Services*, 2008.

[27] J. Liu, B. Wang, H. Lu, and S. Ma. A graph-based image annotation framework. *Pattern Recognition Letters*, 29(4), 2007.

[28] T. Liu. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval*, 3(3), 2009.

[29] C. D. Manning, P. Raghavan, and H. Schtze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[30] D. Metzler and W. B. Croft. Linear feature-based models for information retrieval. *Information Retrieval*, 10, 2007.

[31] E. Moxley, J. Kleban, and B. S. Manjunath. SpiritTagger: A geo-aware tag suggestion tool mined from Flickr. In *Proceeding of the 1st ACM International Conference on Multimedia Information Retrieval*, 2008.

[32] M. Naaman and R. Nair. Zonetag's collaborative tag suggestions: What is this person doing in my phone? *IEEE MultiMedia*, 15(3), 2008.

[33] T. Qin, T.-Y. Liu, X.-D. Zhang, D.-S. Wang, W.-Y. Xiong, and H. Li. Learning to rank relational objects and its application to web search. In *Proceeding of the 17th International Conference on World Wide Web*, 2008.

[34] S. G. Sevil, O. Kucuktunc, P. Duygulu, and F. Can. Automatic tag expansion using visual similarity for photo sharing websites. *Multimedia Tools Applications*, 49(1), 2010.

[35] B. Sigurbjörnsson and R. van Zwol. Flickr tag recommendation based on collective knowledge. In *Proceedings of the 17th International World Wide Web Conference*, 2008.

[36] D. Sorokina, R. Caruana, and M. Riedewald. Additive groves of regression trees. In *Proceedings of the 18th European Conference on Machine Learning*, 2007.

[37] R. C. Veltkamp and M. Tanase. Content-based image retrieval systems: A survey. Technical report, Department of Computing Science, Utrecht University, 2002.

[38] L. Wu, X.-S. Hua, N. Yu, W.-Y. Ma, and S. Li. Flickr distance. In *Proceeding of the 16th ACM international conference on Multimedia*, 2008.

[39] J. Xu and H. Li. AdaRank: A boosting algorithm for information retrieval. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, 2007.

[40] Y. Yue, T. Finley, F. Radlinski, and T. Joachims. A support vector method for optimizing average precision. In *Proceedings of the 30th ACM SIGIR international Conference on Research and Development in Information Retrieval*, 2007.

Spatial-Social Network Visualization for Exploratory Data Analysis

Wei Luo
GeoVISTA Center
Department of Geography,
Penn State University
302 Walker Building
University Park, PA, U.S
wul132@psu.edu

Alan M. MacEachren
GeoVISTA Center
Department of Geography,
Penn State University
302 Walker Building
University Park, PA, U.S
maceachren@psu.edu

Peifeng Yin
PDA Group
Department of CSE,
Penn State University
337 IST Building
University Park, PA, U.S
pzy102@cse.psu.edu

Frank Hardisty
GeoVISTA Center
Department of Geography,
Penn State University
302 Walker Building
University Park, PA, U.S
Hardisty@psu.edu

ABSTRACT

There has been considerable interest in applying social network analysis methods to geographically embedded networks such as population migration and international trade. However, research is hampered by a lack of support for exploratory spatial-social network analysis in integrated tools. To bridge the gap, this research introduces a spatial-social network visualization tool, the *GeoSocialApp*, that supports the exploration of spatial-social networks among network, geographical, and attribute spaces. It also supports exploration of network attributes from community-level (clustering) to individual-level (network node measures). Using an international trade case study, this research shows that mixed methods — computational and visual — can enable discovery of complex patterns in large spatial-social network datasets in an effective and efficient way.

Categories and Subject Descriptors

H.5.2 [**User Interfaces**]: Graphical user interfaces (GUI)

General Terms

Management, Measurement, Design

Keywords

Spatial-Social Network, International Trade, Geovisualization, Geovisual Analytics

1. INTRODUCTION

Social network analysis is used to understand how relationships among actors (i.e., individuals, groups, or other social collectives) within a network affect the behaviors or attributes of themselves and other actors [18]. A growing number of researchers realize the potential application of this analysis method to geographically embedded networks and flows such as population migration [17], and international trade driven by globalization [11]. This increasing realization has called for development of integrated spatial-social tools to support spatially intuitive thinking in traditional social sciences [7].

Many network statistical algorithms exist to detect important individuals, relationships, and clusters, but it is difficult to understand their strictly quantitative outputs. Integrating network statistics with visualization methods has been effective in enabling users to make sense of data [16]. Thus, integrating visualization into spatial-social analysis systems can facilitate sensemaking and discovery of features such as distributions, patterns, and trends over social and spatial space.

The integration of computational methods into effective, interactive visualization in the context of geo-spatial environment gives birth to a new field: geovisual analytics. This new field aims to enable insights on large, complex data containing a geospatial component [1]. This paper introduces a novel geovisual analytics tool, the *GeoSocialApp*, designed to discover previously hidden patterns in complex datasets and to facilitate insight gain. The tool consists of three major components, each of which performs a specific task and can coordinate with other components to facilitate the insight gaining process. These major components include (a) network space, (b) geographic space, and (c) attribute space.

This paper is organized as follows. Section 2 reviews related literature. Section 3 presents an overview of the methods implemented. Section 4 presents the case study application of the tool. Section 5 discusses the limitations of the case study and tool and outlines the future work.

2. RELATED WORK

Current spatial-social network visualization tools can be classified into two major groups: the first group focuses on the spatializing network structures; the second group focuses on the combination of spatial analysis and social network analysis.

The first group considers spatial-social network visualization from a spatial perspective. For example, Guo [8] proposes an integrated interactive visualization framework, in order to effectively discover and visualize major flow patterns and multivariate relations from county-to-county migration data in the U.S. Wood et al.[19] propose an origins and destinations (OD) map to preserve all origin and destination locations of the spatial layout. Overall, this group develops new visualization and interaction techniques to synthesize, visualize, and discover patterns from very large spatial-social networks, but it lacks

ACM LBSN '11, November 1, 2011. Chicago, IL, USA
Copyright © 2011 ACM ISBN 978-1-4503-1033-8/11/11...$10.00

network analysis methods, which have the potential to gain deeper insight for users.

The second group of methods combines geospatial and network analysis with visualization (e.g., Sentinel Visualizer). Davis et al [5] introduce an integrated tool, connecting the Organizational Risk Analysis tool with Geospatial Information (Ora-GI) that allows the integration of geospatial data into the analysis of relational information. Both tools handle location-based spatial network relationships overlaid on a map without supporting network relationships among geographical units (e.g., countries and states). In addition, those tools focus on mathematical and statistical analysis, rather than visual data exploration.

Overall, the above work illustrates a trend to develop new visualization techniques to integrate social networks into a spatial framework, but current work has two major gaps: spatial-social visualization tools do not integrate network analysis methods to enable users to gain deep insight and no existing spatial-social visualization tool deals with network relationships among geographical units. This research aims to fill these gaps.

3. METHODS

GeoSocialApp incorporates network analysis and visualization in a geospatial framework. It is implemented using the GeoViz Toolkit (GVT), a component-based environment for developing multi-view, coordinated geovisualization applications [9].

The network component is based on the Java Universal Network/Graph (JUNG) Framework. JUNG is a JAVA API that provides a framework for modeling, analysis and visualization of relational data [15]. With JUNG, we implement two network views in GVT to facilitate visual exploration of patterns in social space and spatial space simultaneously, a Node-Link and a Dendrogram view (Figure 1). Below, we introduce the views, plus several other GVT components used in GeoSocialApp.

3.1 Node-Link View

The node-link view has two parts: the layout and individual-level network measures. The layout part supports the following layouts: KKL, FRL, Spring, Circle and Self-Organizing Map (SOM). Each layout is suitable for different network structures. The KKL layout implements the Kamada-Kawai algorithm [13], and the FRL implements the Fruchterman-Reingold force-directed algorithm [6]. The Spring layout models the edge as a spring and uses Hooke’s law to determine the position of nodes in the visualization [10]. The KKL, FRL and Spring tend to place connected nodes together while unconnected components tend to be positioned far from each other. Thus, this kind of layout is suitable for a graph with strong communities and weak inter-community links. The circle layout organizes all graph nodes on a circle. It helps users find critical graph nodes in an intuitive way, by pointing out on first sight those nodes with many connections. The Circle layout visualization is suitable for social, computer, and other networks having critical nodes with high degree and complex among node connections. Finally the SOM implements a self-organizing map layout algorithm based on Meyer's self-organizing graph methods. SOM is closely related to force-based graph layout [14], but it is more efficient for large-scale graphs with an optimization algorithm.

Individual-level network measures include:

- Degree: the number of links connected directly by others.
- Betweenness: the extent to which a particular node lies between other nodes.
- Clustering coefficient: a measure of degree to which nodes in a graph tend to cluster together.

3.2 Dendrogram View

The dendrogram view implements the convergence of the iterated correlations (CONCOR) algorithm [3] to group actors with similar position in a single network or multiple social networks together. Actors in similar network positions are considered to have similar social influence in the relational network. The relation of each node with other nodes is represented as binary values (1=connected and 0=disconnected). Thus a node is associated with a binary string, which is treated as the sample data of that node. Based on these “sample data”, CONCOR computes the Pearson product-moment correlation coefficients for every node pair; higher coefficients indicate a more similar position. Through iterative computation, these values will converge on either 1 or -1; the former indicates a strong similarity between positions of two nodes while the latter suggests a high difference. Nodes with a coefficient of 1 are classified into the same group, indicating their highly similar positions in the social network. Hierarchical structure can be achieved by running CONCOR on each subgroup.

The dendrogram view provides two layout methods to visualize the hierarchical structure of CONCOR results: a tree layout and a balloon layout (Figure 1). The tree layout positions child nodes under their common ancestors and organizes the graph in a hierarchical way. Also, an equivalent balloon view can be obtained from the tree by placing each node’s children in its enclosing circle [4]. These two views are good at displaying hierarchical structure of the graph, and are especially adaptive for displaying clustering results as in our tool. There is a slider bar in the view to control the level of the dendrogram result.

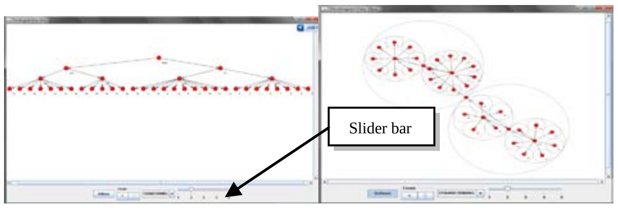


Figure 1. Dendrogram View

3.3 Components in GVT

Existing components in GVT used in this study include the choropleth map view, the histogram view, and the star plot view. The choropleth map view is used here for visual exploration in geographical space. The histogram depicts the distribution of univariate data. The star plot view is used to represent multiple variables (Figure 2). The number of variables corresponds to the number of rays emanating from the center of the star plot, and the length of each ray is proportional to the value it represents. For this research, basic network node measures calculated in the node-link view are imported into the histogram and star plot views, in order to explore the distribution of individual measures and the relationship among multiple measures.

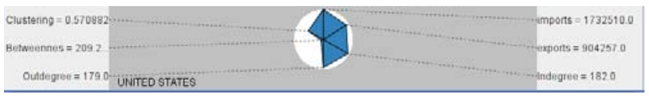


Figure 2. Star Plot View

3.4 Network-GVT View Coordination

The GeoSocialApp allows users to explore spatial-social networks in multiple spaces by supporting coordination among network views and GVT views, with each indicating different perspectives on the same datasets. The dendrogram view and nodelink view allow users to explore networks from the community-level (clusters) to the individual-level (node measures). The choropleth map view gives users an impression about the geographical positions. In addition, each node in the network views represents a geographical unit (i.e., states, countries) in the choropleth map; thus, the linked network views and geographical views allow the explicit interaction between geographical space and social space. Other components in GVT such as the histogram and the star plot allow users to explore the distribution of individual measures and the relationships among multiple measures. Therefore, coordination among network views and other GVT views allows users to explore spatial-social network data in network space, geographical space and attribute space, facilitating the insight gaining process.

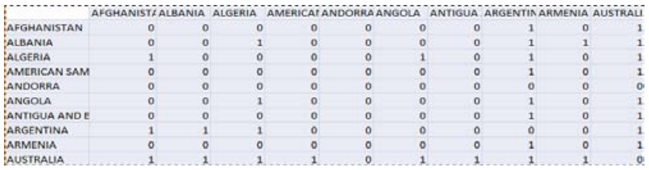
4. WORLD TRADE NETWORK ANALYSIS: A CASE STUDY

We apply GeoSocialApp to world trade data analysis to demonstrate the potential of the methods to support economic research and policy-making. When using a network approach to study international trade, each country is typically considered to be a node of the network. International trade is usually measured by the monetary value of exports and imports between countries, so trading relationships are analogous to links in a network of country nodes. The focus here is on network structure, thus relationships, rather than the monetary values.

4.1 Data Preprocessing

We use the import and export data in current U.S. dollars among 192 countries in 2005 as a case study test of this tool. These data were extracted from the CorrelatesOfWar (COW) Database [2]. Countries are the nodes of the network and a link between two countries represents a trading relationship. We organize the data in matrix form with columns as exporting countries and rows as importing countries. As an illustration, Table 1 is the binary matrix for the first 10 countries in our data; "1" represents trade between countries, "0" represents no trade.

Table 1. Partial Binary Matrix for 0% Threshold in 2005



	AFGHANIST/	ALBANIA	ALGERIA	AMERICA	ANDORRA	ANGOLA	ANTIGUA	ARGENTIN	ARMENIA	AUSTRALI
AFGHANISTAN	0	0	0	0	0	0	0	1	0	1
ALBANIA	0	0	1	0	0	0	0	1	1	1
ALGERIA	1	0	0	0	0	1	0	1	0	1
AMERICAN SAM	0	0	0	0	0	0	0	1	0	1
ANDORRA	0	0	0	0	0	0	0	0	0	0
ANGOLA	0	0	1	0	0	0	0	1	0	1
ANTIGUA AND E	0	0	0	0	0	0	0	1	0	1
ARGENTINA	1	1	1	0	0	0	0	0	0	1
ARMENIA	0	0	0	0	0	0	0	1	0	1
AUSTRALIA	1	1	1	1	0	1	1	1	1	0

4.2 Spatial-Social Network Analysis and Visualization

After converting international trade into the network format, we use the GeoSocialApp to explore it. Initially, we use the dendrogram view to divide the network data into two groups (Figure 3). After highlighting one group (yellow nodes in network views and blue outlines in map view), we find most countries in the highlighted group are economic periphery countries (i.e., most countries in Central America and Africa) and most countries in the other group are core and semi-periphery countries (i.e., North America and European Union). The bivariate choropleth map uses two variables: total values of exports and total values of imports for each country. The bivariate colors reinforce this classification: economically less-important countries are indicated by light green, whereas other, more important countries are indicated by dark green. The node-link view in Figure 3 also indicates the core and periphery relationship with the highlighted group on the periphery of the international trade network, because the KKL layout tends to put connected nodes together and separate unconnected components. From the exploration of the dendrogram, bivariate choropleth, and node-link views, we find that global trade is hierarchical with a core-periphery structure at higher levels of trade; this result matches previous research [12].

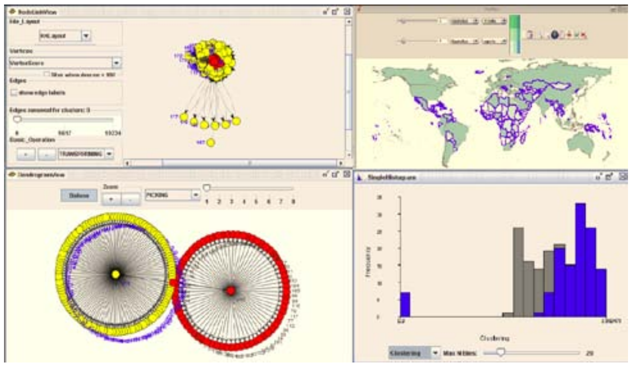


Figure 3. Screenshot with Node-Link, Dendrogram, Bivariate Choropleth Map and Histrogram Views. Large Pictures Available at: http://www.geovista.psu.edu/GeoSocialApp

Second, we are interested in the relationship between basic network node measures for countries and their positions in the world trade network. From the histogram view in Figure 3, we see that periphery countries (highlighted in blue) have a high cluster coefficient. This indicates that periphery countries tend to have international trade among each other, whereas a low coefficient indicates that more developed countries tend to have more trade partners than less developed countries. A few periphery country outliers do not follow the distribution of most other periphery countries. We continue to explore the underlying reason behind those outliers. In Figure 4, we highlight those outliers that have only one trade partner (one export link and one import link). One trade partner results in cluster coefficient of zero. Therefore, those countries are outliers that do not violate the result we find in Figure 3. The result is also supported by the star plot view (Figure 5). In this view, we represent six variables: imports, exports, in-degree, out-degree, betweenness, and clustering coefficient. All 192 countries are ranked from highest to lowest according to the first variable: import. It is obvious that most countries with a high clustering coefficient (long ray from the center of the stars) have a low value with other five variables. The negative relationship implies that rich countries may benefit more from more diversified trade partners and small economies may benefit more from concentrated trade partners. The result is also supported by Kali et al. [11]: the number of trading partners and the concentration of trade are both positively correlated with growth across all countries, but the former is greater for rich countries and the latter is concentrated in poor countries.

Results found using the spatial-social network visualization tool correspond to previous research [11-12]. Most past research has used traditional network measures and computational models, but their results are not easy for non-experts to understand. Our tool can give direct visual representation of patterns to a broader audience, facilitating insight development.

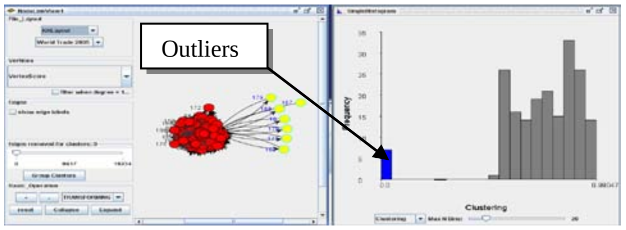


Figure 4.

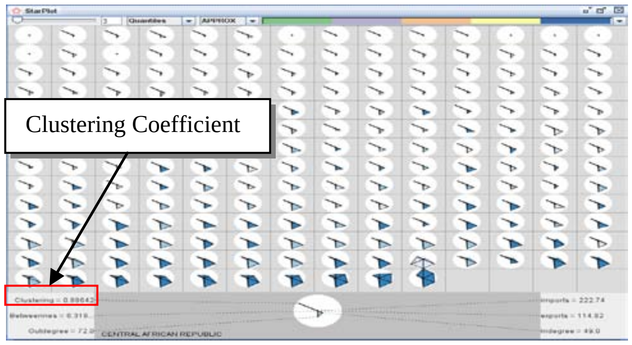


Figure 5. Star Plot View

5. CONCLUSIONS

We introduce the GeoSocialApp that supports exploration of spatial-social networks among multiple spaces: geographical, network, and attribute space. It also supports the exploration of network attributes from community-level (positional clustering) to individual-level (network node measures). The brief case study presented illustrates that this tool can facilitate an insight gaining process about spatial-social networks underlying international trade. GeoSocialApp has been designed as general ones suited to any data aggregated to enumeration units. Thus the system is applicable not just to country-level data as shown here but can be used with data for individuals (e.g., data about social or other connections between individuals aggregated to census blocks in a city). Two possible extensions of this work in terms of application of the tools to trade analysis can be explored in the future: (a) increasing threshold values from zero monetary units enables us to understand the sensitivity of various topological characteristics of the network to differing trade magnitudes; (b) exploring multiple years of data can enable us to understand international trade over time.

6. ACKNOWLEDGMENTS

This material is based, in part, upon work supported by the U.S. Department of Homeland Security under Award #: 2009-ST-061-CI0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the U.S. Department of Homeland Security; a grant from the Gates Foundation also provided partial support.

REFERENCES

[1] Andrienko, G., Andrienko, N., Keim, D., MacEachren, A. M. and Wrobel, S. 2011. Challenging Problems of Geospatial Visual Analytics (editorial introduction). *Journal of Visual Languages & Computing*, 22, 4, 251-256.

[2] Barbieri, K., Keshk, O. M. G. and Pollins, B. 2008. Correlates of war project trade data set codebook, Version 2.0. .

[3] Breiger, R., Boorman, S. and Arabie, P. 1975. An algorithm for clustering relational data with applications to social network analysis and comparison with multidimensional scaling. *Journal of Mathematical Psychology*, 12, 3, 328-383.

[4] Carriere, J. and Kazman, R. 1995. *Research report: Interacting with huge hierarchies: beyond cone trees*. In *Proceedings of the Proc. IEEE Information Visualization' 95* (1995).

[5] Davis, G., Olson, J. and Carley, K. 2008. *OraGIS and Loom: Spatial and temporal extensions to the ORA Analysis Platform*. Carnegie Mellon University.

[6] Fruchterman, T. and Reingold, E. 1991. Graph drawing by force-directed placement. *Software: Practice and Experience*, 21, 11, 1129-1164.

[7] Goodchild, M. and Janelle, D. 2010. Toward critical spatial thinking in the social sciences and humanities. *GeoJournal*, 75, 1, 3-13.

[8] Guo, D. 2009. Flow mapping and multivariate visualization of large spatial interaction data. *Visualization and Computer Graphics, IEEE Transactions on*, 15, 6, 1041-1048.

[9] Hardisty, F. and Robinson, A. 2010. The geoviz toolkit: using component-oriented coordination methods for geographic visualization and analysis. *International Journal of Geographical Information Science*, 25, 191-210.

[10] Herman, I., Melançon, G. and Marshall, M. 2000. Graph visualization and navigation in information visualization: A survey. *IEEE Transactions on Visualization and Computer Graphics*, 6, 1, 24-43.

[11] Kali, R., Méndez, F. and Reyes, J. 2007. Trade structure and economic growth. *Journal of International Trade Economic Development*, 16, 2, 245-269.

[12] Kali, R. and Reyes, J. 2007. The architecture of globalization: a network approach to international economic integration. *Journal of International Business Studies*, 38, 4, 595-620.

[13] Kamada, T. and Kawai, S. 1989. An algorithm for drawing general undirected graphs. *Information processing letters*, 31, 12, 7-15.

[14] Meyer, B. 1998. *Self-organizing graphs—a neural network perspective of graph layout*. In *Proceedings of the Graph Drawing. 6th International Symposium, GD' 98. Proceedings* (Berlin, Germany, 1998). Springer-Verlag.

[15] O'Madadhain, J., Fisher, D., Smyth, P., White, S. and Boey, Y. 2005. Analysis and visualization of network data using JUNG. *Journal of Statistical Software*, 10, 1-35.

[16] Perer, A. and Shneiderman, B. 2008. *Integrating statistics and visualization: case studies of gaining clarity during exploratory data analysis*. In *Proceedings of the Proc. SIGCHI conference on Human factors in computing systems* (2008). ACM.

[17] Tobler, W. R. 1987. Experiments in migration mapping by computer. *Cartography and Geographic Information Science*, 14, 2, 155-163.

[18] Valente, T. 2010. *Social Networks and Health: Models, Methods, and Applications*. Oxford Univ Pr.

[19] Wood, J., Dykes, J. and Slingsby, A. 2010. Visualisation of origins, destinations and flows with OD maps. *The Cartographic Journal*, 47, 2, 117-129.

User Association Analysis of Locales on Location Based Social Networks

Josh Jia-Ching Ying[1], Wang-Chien Lee[2], Mao Ye [2], Ching-Yu Chen[1] and Vincent S. Tseng[3]
[1, 3] Department of Computer Science & Information Engineering, National Cheng Kung University, Taiwan, ROC
[2] Department of Computer Science & Engineering Pennsylvania State University, PA 16802, USA
[1]{jashying, tom76925}@gmail.com, [2]{wlee, mxy177}@cse.psu.edu, [3]tsengsm@mail.ncku.edu.tw

ABSTRACT

In recent years, location-based social networks (LBSNs) have received high attention. While this new breed of social networks is nascent, there is no large-scale analysis conducted to investigate the associations among users in locales of the network. In this paper, we propose four locale based metrics, including *Locale Clustering Coefficient*, *Inward Locale Transitivity*, *Locale Assortativity Coefficient*, and *Locale Assortability Coefficient* to make association analysis on EveryTrail, a popular LBSN specialized on sharing trips. Based on the analysis result, we observe that people who share more trajectories will get more attention by other users, and people who are popular will connect to the people who are also popular.

Categories and Subject Descriptors

H.2.8 [Database Management]: Database Applications – *Data Mining, Spatial Databases and GIS*

General Terms

Measurement, Experimentation.

Keywords

Social Network Analysis, Locale Based Metrics, Clustering Coefficient, Assortativity Coefficient, Inward Transitivity.

1. INTRODUCTION

With the rapid growth and fierce competition in the market of Web 2.0, many web service providers have deployed various social networking services to allow users to share their digital content and personal views/information with each other, e.g., YouTube, Flickr and Blogger facilitate their users to share video clips, photos and articles, respectively. These services are very useful since users who have similar interests can share their information or experiences. In recent years, a new breed of social networking services, called *location-based social networks (LBSNs),* have emerged. Thanks to the advances in mobile computing and wireless networking technologies, users of LBSNs can track and share location-related information with each other on the move. By adding this new dimension of features, LBSNs bring their users from the virtual world back into real lives and allow the real-life experiences be shared in the virtual world in a very convenient fashion. Several LBSNs sites allow their users to share not only visited locations but also trajectories, e.g., Bikely [1] and EveryTrail [2]. Notice that a trajectory, which typically consists of a sequence of spatio-temporal points (in form of latitude, longitude, and time) as shown in Figure 1, captures a user's physical moving behavior in real world. Thus, the physical behaviors of users can be extracted from user trajectories to facilitate many advanced location based services.

ACM LBSN '11, November 1, 2011. Chicago, IL, USA
Copyright © 2011 ACM ISBN 978-1-4503-1033-8/11/11...$10.00

With the rapid development of the social networking services and the need to better understanding their users, many network analytic metrics adopted from graph theory, such as diameter, graph density, and maximum size of component, have been adopted for analyzing basic properties of social networks [6][7][10]. Additionally, many social network metrics, such as clustering coefficient [3] [13] and assortativity coefficient [7] [9] have been used to measure the *association* between users in a social network. Notice that *association analysis* is important for enhancing social network services. For example, by analyzing factors that significantly affect the social behavior of users, friend recommendation service of a social network can be improved and many other services, e.g., community discovery and activity recommendation, can be enriched. Moreover, some metrics are gifted with unique meaning. For example, while clustering coefficient is originally proposed to measure how *clustered* the neighborhood of a node is, it also can measure the *transitivity* of a user in a social network, i.e., if many friends of a user make friend with each other via the user, the user have a strong transitivity for her friends.

Only recently, the issues of analyzing LBSNs are discussed in the literature. However, existing studies mostly focus only on analyzing properties of *the whole network* by employing traditional graph theoretical metrics to analyze the LBSNs. Some prior works [3][10][12] have proposed to use the *distance between two users* by modifying clustering coefficient into a new metric, called *geographic clustering coefficient*, in order to quantify how tightly nodes in the neighborhood of a node (as measured by distance) is connected in an LBSN. However, this measure does not capture the impact of *geographic locales* (or *locales* in short)

Figure 1. An example of GPS trajectory.

and activities on the users in the social network.

In this paper, we aim to address this deficiency by proposing a number of new social network analytic metrics based on geographic locales. Here we informally refer to locales as *the geographic regions of particular interests where users of LBSNs have activities*. Examples of geographic locales range from business districts, city vicinity, country boundaries to continentals. In this paper, we mainly consider city vicinity even though our analysis and proposed concepts are applicable to other fine-grained or coarse-grained locales. As mentioned earlier, a geographic trajectory typically consists of a sequence of geographic points (represented as <latitude, longitude>) tagged with timestamps. Thus, the locales of a user are easy to extract from her trajectories. For example, if a user uploads many trajectories in New York City, we could conclude that the user has a lot of activities in New York City. In this paper, we are interested in developing some locale-based analytical metrics for analyzing the social association among those who have activities in a number of selected cities.

To analyze the social association in terms of locales in an LBSN, we collect user data including social links and trajectories from EveryTrail, a trip-sharing and social networking website. On EveryTrail, users can upload GPS logs, travelogues and photos regarding their trips. Users can also comment on other users' trips too. Every registered user has her own profile, which shows a brief introduction about the user and all trips created by the user. Users can view other users' profiles and *follow* those whom they are interested in. If a user follows somebody, she will get notified when her *followees* upload new trips or do something special on the website. In contrast to a followee, we call a user following somebody a *follower*. Such a "follow-up" relationship among users forms a directed social network. For a given locale, we group the users into *inside users*, i.e., those who have activities inside the locale, and *outside users*, i.e., those who do not have activities inside the locale. Accordingly, we propose a series of novel locale-based metrics for social association analysis. First, we propose the *locale clustering coefficient* to analyze the clustering effect of users in a given locale. Also, we propose *inward locale transitivity* to analyze the transitivity between inside users and outside users of a given locale. Then we propose a new metric, called *locale assortativity coefficient,* to analyze the correlation between the "in-degrees" of followers and "out-degree" of their followees within a locale. Finally, to measure the correlation between the "in-degrees" of followers and "in-degree" of their followees within a locale, we propose a new metric, named *locale assortability coefficient,* to exam whether popular users follow other popular users.

Geographic locales such as city vicinities naturally play an important role in LBSNs. While LBSNs have received a lot of attention in recent years, to the best knowledge of the authors, tools for analyzing association of users in locales have not been explored. In this study, we have a number of contributions and findings as follows:

- We propose the *locale clustering coefficient* to analyze the clustering effect of users in a given locale. Our analysis result shows that New York City and San Francisco have very strong localized locale clustering coefficient. While there are four cities with 0 locale clustering coefficient since the personal social links of users and their neighbors in these cities are pretty much star-like. Besides, the locale clustering coefficient seems to be increasing proportional to the number of trips in city, which is probably because that rich activities attract more users to follow each other.
- We propose the inward locale transitivity to examine the degree of transitivity from outside users to inside users of a given locale. Our analysis on Everytrail data shows that users active in New York and those active in San Francisco have higher inward transitivity than users active in other cities because these two cities not only have many more active users but also many popular landscapes or scenic places that attract a lot of people.
- We propose the locale assortativity coefficient to analyze the correlation, in terms of node degrees, between users involved in the follow-up relationship in a given locale. Based on our analysis, we observe significant assortativity between inside users and outside users for each locale.
- We propose the locale assortability coefficient to examine whether popular users follow other popular users. Using the locale assortability coefficient, we find that groups in New York and San Francisco gather all kinds of people and most of them want to follow popular users.
- We use a real dataset collected from EveryTrail to exam our proposal. It is worth noting that our locale based association analysis show similar result for users in the locales of San Francisco and New York City but not for users in other cities.

The remaining of this paper is organized as follows. We briefly review the related work in Section 2 and describe the EveryTrail dataset and the result of several basic analyses on the dataset in Section 3. We then introduce the proposed locale based metrics and analytical results in Section 4. Finally, the conclusions and future work are summarized in Section 5.

2. RELATED WORK

Many studies have discussed the associations between users based on their movement behaviors [5][14]. In [5], Li *et al.* propose a hierarchical structure to describe users' movement behaviors and calculate the users' similarity based on the hierarchical structures. In addition to the GPS similarity, Ying *et al.* [14] also exploit the users' similarity based on the semantic meaning of their movement. However, these studies analyze users' association based on users' movement regardless of the structure of social network.

Several studies have performed analysis on on-line social network and telecom call graphs. Graph theoretical analysis [6][7][10] and association based analysis [4][7][11] are two hot topics in this research domain. In [6], Mislove *et al.* use many graph metrics such as path lengths, diameter, graph density, etc. to measure different social networks in Flickr, YouTube, Live Journal, and Orkut. Their results confirm the power-law, small-world, and scale-free properties of online social network. In [7], Nanavati *et al.* use graph density as a metric to understand the shape of call graphs. The results provide business insights to telecommunication carriers. In [10], Opsahl *et al.* argue that links in a network often have strength which can be operated as weights. However the three common centrality metrics used in the paper, including degree, closeness, and betweenness, are for network with non-weight edges. They combine weight features with

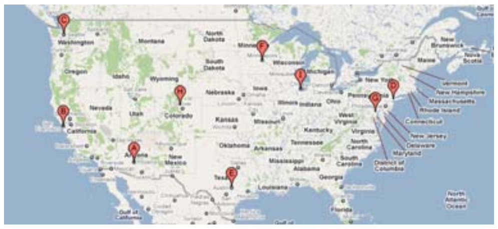


Figure 2. Nine cities on the map.

conventional centrality metric to demonstrate the benefits by applying them to Freeman's EIES dataset.

The graph theoretical analysis can help us understand the whole structure of a whole network. Moreover, it is also important to understand the association between users in a network. For example, clustering coefficient is a very common coefficient for examining the association between friend and friend's friend. In [4][7][11], the authors use clustering coefficient to see the association of the association in the social network. In [11], Scellato *et al.* show the impact of geographic distance on the probability of a connection, i.e., social link. They propose a new metric geographic clustering coefficient which is modified from the conventional clustering coefficient. They give each triangle a weight which is computed by geographic distance. Therefore they can emphasize link with short geographic distance. The result shows that location-based social networks connect their neighborhoods much tighter than other on-line social networks, such as twitter. In [4], Li *et al.* use a metric similar to assortativity coefficient. They calculated the Spearman's rank correlation coefficient between a user's number of friends and the average number of friends for that user's friends. The result confirms that popular users tend to associate with users who are also popular. In [7], Nanavati *et al.* use assortativity coefficient with four variants to understand whether users in different region will have consistent behavior or not. The result shows that different regions exhibit consistent behavior in certain level. However, there is no research considering the effect of locales in metrics to analyze location-based social networks.

3. EVERYTRAIL DATASET

To analyze LBSNs, we have crawled the EveryTrail website to collect a dataset of users and their shared information (including profiles, trajectories, social links, etc). Fortunately, users in Everytail are numbered and named by consecutive numeric IDs. Thus, we are able to scan all users in Everytail to download their profiles and shared data. As a result our data can be considered a complete snapshot of EveryTrail (dated in February 2011), which is good for social network analysis. In this section, we first provide a general overview of how Every Trail looks like by performing basic social network analysis.

Table 1. Basic statistics of trips per user.

	cities	*# of trips*	*# of users*	*Avg. # of trips per user*	*Max # of trips per user*
A	Phoenix	389	102	3.81	36
B	San Francisco	355	207	1.71	16
C	Seattle	340	145	2.34	23
D	New York	333	146	2.28	29
E	Austin	293	133	2.2	15
F	Minneapolis	277	70	3.96	100
G	Baltimore	266	18	14.78	241
H	Boulder	180	93	1.94	10
I	Chicago	153	94	1.63	7

The data collected were created by users from September 08, 2006 to February 11, 2011. We retrieved 256,378 profiles of users who signed up on EveryTrail before February 11, 2011. We extract 2,586 trips information in total for 9 cities, including Seattle, San Francisco, Phoenix, New York, Minneapolis, Chicago, Boulder, Baltimore and Austin as shown in Figure 2. Table 1 lists some basic statistics of our dataset, in city vicinity level, including the number of trips, the number of users, the average number of trips per user and the maximum number of trips per user. This table facilitates us to compare the different characteristics in different geographic locales as discussed below.

3.1 Statistics of Users and Trips

First, we like to see whether in different cities, trips are usually provided by a small potion of the inside people or not. We also want to see how many trips each user usually has in different cities. As shown in Table 1, we find that trips are usually created by different people. Notice that each user has about one to three trips in average, but if we look at the statistics carefully, we find that the average number of trips per user in Baltimore is about 15. That is because even though there are 266 trips in Baltimore, one particular user has 241 trips. Similarly, there are only 277 trips in Minneapolis, but there is a user who has 100 trips. In summary, we find in this data that while most users have one to three trips in each different city, there are still some users have extremely large amount trips in one city. These users are often doing daily life pattern logging.

Table 2. Activities in each city.

	Seattle	***San Francisco***	***Phoenix***	***New York***	***Chicago***	***Boulder***	***Austin***	***Baltimore***	***Minneapolis***
Flying	3%	1%	0%	1%	2%	1%	0%	1%	0%
Motorcycling	2%	3%	0%	1%	2%	0%	1%	0%	0%
Sightseeing	1%	7%	1%	5%	2%	0%	0%	1%	1%
Other	3%	6%	1%	5%	6%	2%	1%	2%	6%
Hiking	2%	11%	10%	0%	7%	**31%**	14%	1%	4%
Biking	**37%**	**34%**	**71%**	**25%**	**29%**	*26%*	**45%**	1%	**54%**
Sailing	1%	2%	0%	0%	0%	0%	0%	1%	0%
Driving	2%	5%	1%	2%	9%	3%	2%	0%	0%
Running	2%	8%	2%	*25%*	10%	15%	12%	0%	*19%*
Boating	1%	0%	0%	0%	8%	0%	0%	0%	0%
Walking	*35%*	*14%*	2%	18%	*20%*	7%	8%	**91%**	10%
Unknown	10%	8%	13%	18%	4%	16%	16%	3%	7%

3.2 Statistics of Activities

The GPS traces in EveryTrail log different activity types, such as motorcycling, biking, sailing and so on. Since different cities have their unique geographical features, the activities in different cities could be different. Thus, we next look into the proportion of different activities in each city and compare the difference between them. According to the activity tags on trips, we categorize activities into 10 types. Notice that the EveryTrail website provides not only predefined activity terms to choose for but also user-input activity terms to describe trips. In order to obtain accurate counts of the trip activities, we manually group trip activities and get the statistics as shown in Table 2.

As we can see in the Table 2, biking is the top trip activity (numbers shown in bold) for most of the examined cities. However, there are some exceptions. For example, in New York, running trips have the same proportion as biking trips; and in Boulder, hiking trips have a higher proportion than biking trips. This phenomenon is due the culture of the people there and the geographical nature of New York and Boulder. New York is a big and crowded city, people like to run or joggle in the park; and Boulder is located near the foothills of the Rocky Mountains, hiking is naturally a favorable activity there. We next look at the second largest proportion of activity types in each city (numbers shown in red and italic). In Seattle, San Francisco and Chicago, the second largest proportion of activity types is walking. In New York and Minneapolis, the second largest proportion of activity types is running. We find that biking, running and walking are all similar exercises in some way. Therefore, we conclude that people usually want to upload their sporting trails in these nine cities.

Because trips in Baltimore and Minneapolis are mostly contributed by one person, the activities distributions are biased in these two cities. We can see that recorded trips mainly consist of walking trips in Baltimore and more than half of the trips in Minneapolis are biking trips. As a result, the data of these two cities better describe particular users' daily life patterns instead of the cities themselves.

According to basic statistics of EveryTrail dataset, we observe fluctuant values among different locales. That means the users' behaviors in various locales are very different. Intuitively, the users' social behaviors may also get affected by locale properties. Thus, it motivates us to provide some locale based metrics to analyze the location based social networks.

4. LOCALE BASED METRICS

To our best knowledge, existing social network analysis metrics do not take into account the effect of locales. However, we do observe that all the graph theoretical metrics show fluctuate values in different cities. That means there are some potential effects related to locales. Therefore, it is necessary to provide some metrics to reflect the effect of locales for analyzing a location based social network. In this section, we propose four different locale based metrics, including *Locale Clustering Coefficient*, *Inward Locale Transitivity, Locale Assortativity Coefficient*, and *Locale Assortability Coefficient*. Among them, the locale clustering coefficient is used to understand the clustering effect of users in a locale and the inward locake transitivity is for measuring the transitivity between users inside and outside a locale. On the other hand, the locale assortativity coefficient and the location assortability are used, in different ways, to measure the correlation between the node degrees of users within a locale and that of their followees in the locale. Similar to Inward transitivity, we also propose inward assortativity and inward assortability for locales.

Table 3. Locale clustering coefficients.

Cities	LCC (Global)	LCC (Localized)
Austin	0.07	0.005
Seattle	0.056	0.025
Phoenix	0.094	0.018
New York	0.032	0.225
San Francisco	0.107	0.284
Chicago	0.002	0
Boulder	0.017	0
Baltimore	0.069	0
Minneapolis	0	0

4.1 Locale Clustering Coefficient

Before introducing the locale clustering coefficient, we first describe the conventional clustering coefficient [3][13]. The clustering coefficient measures the degree of transitivity of a node (i.e., a user in this paper) in a network. Consider a *triad* of three nodes/users A, B and C in a social network. The triad is transitive if when there is a link from A to B (denoted as A→B) and one from B to C (denoted as B→ C), then there is also a link from A to C (denoted as A→C). If the A→B and B→C links both exist, but A is not connected to C, then the triad is intransitive. It has been shown that social relations have a tendency toward transitive, i.e., given two nodes in a network sharing the same neighbor node, there is a heightened probability that these two nodes will also be connected. Thus, we can say that if many neighbors of a node are connected to each other, then the node has a high degree of transitivity, which can be measured by the clustering coefficient as defined below.

$$C_i = \frac{\text{number of triangles connected to node } i}{\text{number of triples centered on node } i}, \qquad (1)$$

where node i is being measured, triple is a path of 3 nodes, and triangle is the complete graph of 3 nodes.

Notice that clustering coefficient is associated with an arbitrary node. We are interested in measuring the *locale clustering coefficient (LCC)* as the average clustering coefficient of users in a locale. To obtain this measure, a simple approach is to calculate the clustering coefficient for each user in the global network and then average the clustering coefficients of users who have activities in the locale of interest. We apply this approach on EveryTrial data to obtain the average locale clustering coefficient for each of the cities under examination. The result, denoted as *LCC (Global),* is shown in Table 3. Notice that we label it as Global because the clustering coefficients are derived from the global network instead of the subnetworks in the locales. Also notice that the average clustering coefficient for the whole social network of EveryTrail is 0.058. The result shows that the average LCC (Global) of users who have activities in San Francisco, Phoenix, Austin, and Baltimore are higher than the average cluster coefficients of the whole network, indicating that users active in these cities form tighter clusters than users in the rest of cities. Since the neighbors considered in calculating clustering

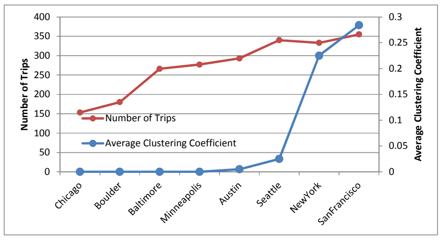


Figure 3. number of trips and conventional clustering coefficient

coefficient here may spread all over the network, we propose a *localized LCC* which takes into account only neighbors within the locale. In other words, we calculate this LCC by considering the localized social network of users within the locale. Accordingly, we also show the obtained average clustering coefficient, denoted as LCC (Localized), for each city in Table 3. It's interesting to find that New York City and San Francisco have very strong localized locale clustering coefficient, much higher than that of the whole network, indicating high clustering effect among users in these two cities. Moreover, there are four cities with 0 in their average localized LCCs, i.e., the structures of the personal social networks for users in these cities are pretty much star-like instead of forming clusters. Another interesting finding is that the localized LCC seems to be increasing proportional to the number of trips (see Figure 3). This is probably because the more activities (i.e., trips) related to a city tends to attract more users to 'follow' each other.

4.2 Inward Locale Transitivity

In addition to the clustering effect within a locale, we are also interested in measuring the transitivity between users outside a locale (called *outside users* of the locale) towards users inside a locale (called *inside users* of the locale). We argue that users who are interesting in a city may not have ever been to the city and thus may follow the inside users of the city. We wonder what kinds of followers users who have activities in a certain locale will attract. Therefore, we consider links from outside users to inside users and called them as "Inward-Pairs". As Figure 4 shows, B is an inside user and A is an outside user. The link "A follows B" is an inward-pair links. For completeness, we also define the links from inside users to outside users as "Outward-Pairs" and the links between inside users as "Inner-Pairs". Table 4 shows the numbers of Inward-Pairs, Outward-Pairs and Inner-Pairs for each city. We can see that comparing to the whole number of users on EveryTrail, each number of users in these nine cities is relative small. It is interesting to find that the number of Inward-Pairs is always higher than the number of Outward-Pairs in each group and the number of Inner-Pairs is always the smallest in each group.

We aim to measure the Inward transitivity between inside users and outside users since it may be an indicator of outside interests

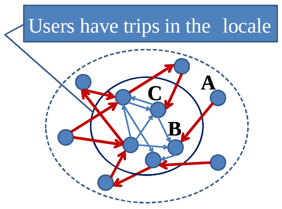


Figure 4. Inward-pairs, outward-pairs and inner-pairs.

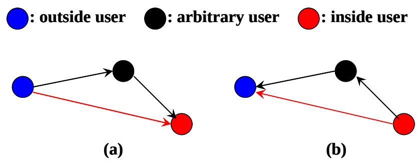


Figure 5. Illustration of inward transitivity.

in a locale. Thus, we model the Inward-Transitivity as follows.

$$T_{\text{inward}}(i) = \frac{\#\text{ of Inward - Pairs from node } k \text{ to node } j}{\#\text{ of triples from node } k \text{ to node } j \text{ centered on node } i} \quad (2)$$

where node k is an outside node and node j is an inside node.

Figure 5(a) illustrates the notion of inward transitivity. For an arbitrary user (i.e., black node in Figure 5), we could calculate the rate of her outside followers to a given locale (i.e., blue node in Figure 5) to follow her followees inside the locale (i.e., red node in Figure 5). The inward-transitivity reflects the ability of a user to relate outside friends to his followees in a locale. Similarly, we also model the Outward-Transitivity as follows.

$$T_{\text{outward}}(i) = \frac{\#\text{ of Outward - Pair from node } j \text{ to node } k}{\#\text{ of triples from node } j \text{ to node } k \text{ centered on node } i} \quad (3)$$

where node k is an outside node and node j is an inside node.

The notion of outward-transitivity is illustrated in Figure 5(b). Similarly, for an arbitrary user, we can calculate the rate for her inside followers to follow her outside followees. The outward-transitivity reflects *indirectly* the ability of a user to relate outside friends to his followees in a locale since we assume that a user is likely to review the profiles and trips of her followers as well.

Figure 6 shows the measured result of inward-transitivity (we also show that of outward transitivity as a secondary indicator) to study outside users' interests in the examined locales. The measured inward-transitivity values are very small since there are only 18 to 207 users in each group. However we still can compare them among different groups. First, from the *inward-transitivity,* we can see that the New York and San Francisco user groups have much higher inward-transitivity than other groups. This is because that these two groups not only have much more users but also very famous scenic places and landmarks that attracts a lot of outside people to follow the inside users. Since the selected cities have more active users and trips than other cities, outside users have more motivations to follow inside users. Thus, inward-transitivity may serve as the strength of outside interests in locales. For example, consider New York and Seattle, which have similar

Table 4. statistics of link.

Cities	**# of In-Pairs**	**# of Out-Pairs**	**# of Inner-Pairs**	**# of Users**
San Francisco	2,507	519	51	207
New York	2,556	827	30	146
Seattle	1,136	128	10	145
Austin	603	88	11	133
Phoenix	366	40	33	102
Chicago	570	15	2	94
Boulder	234	75	8	93
Minneapolis	31	4	0	70
Baltimore	23	11	0	18
All Users				256,378

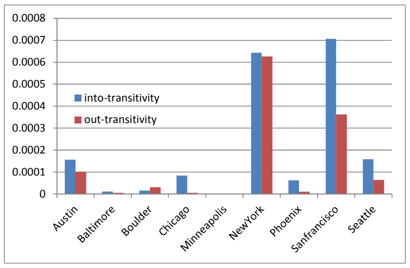


Figure 6. Inward locale transitivity.

numbers of users, i.e., 146 and 145, respectively. However, the inward-transitivity of New York is much higher than that of Seattle. Therefore, we can say that the users in this network tend to be more interested in New York than Seattle. The outward-transitivity, while usually lower than the inward-transitivity, provides a secondary indication of the strength of the (reversed) transitivity inward a locale.

4.3 Locale Assortativity Coefficient

Before introducing the locale assortativity coefficient, we first describe the conventional assortativity coefficient [7] [9]. The assortativity coefficient is a metric to measure the degree of connectivity association, which refers to the preference for a node to get connected with other nodes that are similar or different (i.e., correlated) in certain way. Notice that the correlation can be established in many different ways. In this paper, we examine the assortativity in terms of *node degree* and first calculate the assortativity coefficient for a directed network as defined in [7] (See Equation (4) below).

$$r = \frac{\sum_{jk} jk(e_{jk} - q_j^{in} q_k^{out})}{\sigma_{in}\sigma_{out}}, \qquad (4)$$

where e_{jk} is the probability that a randomly chosen directed edge leads into a node of in-degree j and out of a node of out-degree k, q_j^{in}, q_k^{out} are normalized distribution of in-degree j and out-degree k and σ_{in} and σ_{out} are the standard deviations of the distribution q_j^{in} and q_k^{out}, repectively.

Notice that the above definition is based on the "number of followers" and "number of followees" associated with edges in the network, i.e., pairs of users. It aims to see if users who have a lot of followers would follow those users who have a lot of followees. For every user, the number of her followers is the in-degree and the number of followees is the out-degree. Thus, it measures the correlation between the in-degree of beginning node and the out-degree of ending node.

Table 5. Locale Assortatitivity Coefficient.

*means significant

Cities	LAC (Global)	LAC (Localized)	IAL (Absolute)	IAL (Relative)
Austin	0.941*	1*	0.124 *	0.159*
SanFrancisco	0.353*	0.524*	0.183*	0.275*
Baltimore	no edge	no edge	0.074*	0.052*
Boulder	0.612	1*	0.097*	0.121*
Chicago	0	0	0.092*	0.091*
Minneapolis	no edge	no edge	0.0004	0.011*
NewYork	0.376	0.573*	0.214*	0.335*
Phoenix	0.238	0.318	0.052*	0.086*
Seattle	0.607	0.882*	0.124*	0.181*
All Users	0.159			

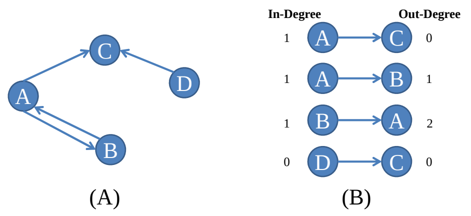


Figure 7. an example of calculating of assortativity.

Accordingly, the assortativity coefficient for the whole network is calculated by picking up each edge in the network and use Pearson correlation coefficient to compute the in-degree of the vertex leads into the edge (i.e., the beginning node) and the out-degree of the vertex leads out of the edge (i.e., the ending node). For example, given a graph like Figure 7(A) we can pick up all edges as Figure 7 (B) shows. Take the edge "A→C" as an example, the node A got 1 as the in-degree because node A has an edge coming from B and node C got 0 as the out-degree because node C points to no one. By listing all the in-degree values and out-degree values as Figure 7(B) shows, we can calculate their Pearson correlation coefficient. Next, we consider only the edges within the locale of interest to obtain the locale assortativity coefficient (LAC). The result for cities under examination is shown under LAC (Global) in Table 5. Since the assortativity coefficient is calculated based on correlation coefficient, we use hypothesis test to exam whether this coefficient is significant or not. The null hypothesis is that there is no association between nodes' connectivity in this network. We reject the null hypothesis when the p-value is less than 0.05. As the Table 5 shows, the assortatitivity coefficient of the whole dataset is 0.159 which is not significant, that means in this network the connectivity association is probably weak. We can see that only Austin and San Francisco groups have significant LAC (Global) values because the number of edges in inner group is very small in other groups. Therefore the result above the significant level should be very high.

Obviously, the LAC (Global) is obtained regardless the incoming and outgoing edges are within the locale or not. Thus, similar to our exercise for locale cluster coefficient, we also calculate the *localized LAC* and show the result under the label LAC (localized) in Table 5. We can see that, in most groups, LAC (localized) is higher than LAC (Global). It's interesting to find that most of cities have very strong localized locale assortativity coefficient, much higher than that of the whole network, indicating high connectivity effect among users in these cities.

Additionally, we believe that it is important to measure the

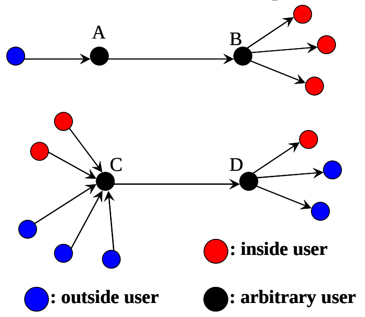


Figure 8. example of connectivity association.

connectivity association between inside users and outside users of the locale. Therefore, we propose a new measure, called *Inward Assortativity for Locale (IAL)*, which aims to measure, for each edge in the network, the average correlation between its incoming edges from outside users (for a locale) and its outgoing edges to inside users .

To obtain the IAL, we adopt two different approaches to measure the nodes' degree: (1) Absolute degree – for a given edge in the network, let *in-degree* be the incoming edges from outside users and *out-degree* be the outgoing edges to inside users. For example, in Figure 8 the *in-degree* and *out-degree* values of the edge "C→D" are 3 and 1, respectively, since there are 3 outside users' edges pointing to node C and node D has an edge pointing to inside users. (2) Relative degrees – let *in-ratio* be *the in-degree divided by the total number of incoming edges* and *out-ratio* be *the out-degree divided by the total number of outgoing edges.* For example in Figure 8 the *in-ratio* and *out-ratio* values of the edge "C→D" equals 3/5 and 1/3, respectively. The reason we need to consider relative degrees is because of the possible overweight of the degrees in some cases. For example, in Figure 8, the in-degree and out-degree values of the edge "A→B" are 1 and 3 respectively and the those values of the edge "C→D" are 3 and 1, respectively. We get the correlation coefficient of these both cases as -1, i.e., negative correlation. However if we use relative degrees, the in-ratio and out-ratio values of the edge "A→B" are 1 and 1 and those values of the edge "C→D" are 3/5 and 1/3, respectively. We get the correlation coefficient of these two pair as 1 which is positive correlation instead. We separate users into different group based on locales and perform our proposed IAL (absolute) and IAL (relative) on the EveryTrail data, the values of IAL (absolute) and IAL (relative) of each locale are given in Table 5.

As the Table 5 shows, both the IAL (absolute) and IAL (relative) suggest that there is significant positive correlation between incoming edges from outside users and outgoing edges to inside users for each city. Only the result of IAL (absolute) for Minneapolis is not significant. Besides, the New York group and San Francisco group gets the highest IAL in both absolute and relative degrees measures. If we compare IAL (absolute) and IAL (relative), in almost every group the relative degrees approach is higher than absolute degrees approach. This result shows that all these groups contain a lot of pairs with over weighted result, which may be related to the locale feature, e.g., the New York and San Francisco groups have much difference between absolute and relative approaches and the Baltimore group has relative degrees smaller than absolute degrees.

4.4 Locale Assortability Coefficient

In addition to the locale assortativity coefficient defined along the basic idea in [7], we also would like to observe whether popular users in the network follow the users who are also popular (or the users who are not so popular). Different from Section 4.3, for an edge in the network, here we are interested in the correlation between in-degree of node leading into the edge and the in-degree of node leads out of the edge. This new metric, *Assortability Coefficient*, is defined as follow:

$$r = \frac{\sum_{jk} jk(e_{jk} - q_j q_k)}{\sigma_j \sigma_k}, \qquad (5)$$

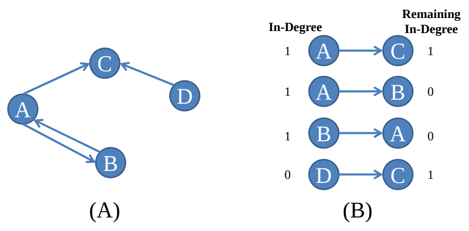


Figure 9. an example of calculating of assortability.

where e_{jk} is the probability that a randomly chosen directed edge leads into a node of in-degree j and out of a node of in-degree k, σ_j, σ_k is the standard deviation of the distribution q_j, q_k.

To calculate the assortability coefficient of the whole network, we can pick up all edges in the network and use Pearson correlation coefficient to compute the in-degree of the vertex leads into the edge and remaining in-degree of the vertex leads out of the edge. The remaining in-degree means the number of in-degree of the node minus one, which represents the current edge point to it. For example, given a graph as shown in Figure 9(A), we can pick up all edges as Figure 9(B) illustrates. Take the edge "A→C" as example, the node A got 1 in-degree because node A has an edge coming from node B and node C gets 1 as the remaining in-degree because node C has only one edge coming from node D. By listing all the in-degree values and remaining in-degree values as shown in Figure 9(B), we can calculate their Pearson correlation coefficient.

The assortability coefficient of the whole network is 0.158 which is significant but indicates in this network only a small number of popular users follow other popular users. Since we are interested in the assocrtability coefficient for a locale, we again consider only the edges within the locale of interest to obtain the *locale assortability coefficient* (and denote it as *LABC*). The result for cities under examination is shown under LABC (Global) in Table 6. We can see that assortability coefficient of whole dataset is 0.158 which is not significant, that means in this network the connectivity association is probably weak. We can see that only New York and San Francisco groups have significant LABC (Global) values because the number of edges in inner group is very small in other groups. Therefore the result above the significant level should be very high.

Obviously, the LABC (Global) is obtained regardless the incoming edges under consideration are within the locale or not. Thus, similar to our previous exercise, we also calculate the localized LABC and show the result under LABC (localized) in Table 6. It's interesting to find that Austin, New York City and San Francisco have very strong localized locale assortability coefficient, much higher than that of the whole network, indicating high assorting effect among users in these two cities.

Finally, we propose a new measure, called Inward Assortability for Locale (IABL), for edges in the network, to measure the average correlation between incoming edges from outside users to the beginning node and incoming edges from inside users to the ending nodes. Again, we consider the effect of overweight in absolute in-degrees and thus adopt both the absolute degree and the relative degree approaches in calculating the IABL.

To obtain the IABL, we adopt two different approaches to measure the nodes' degree: (1) Absolute degree – for a given edge in the network, let *outside-in-degree* be the incoming edges from

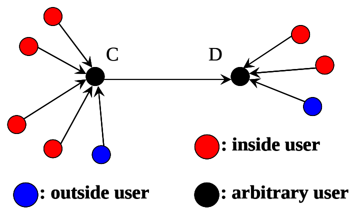


Figure 10. Illustration of Inward Assortability for Locale.

outside users and *inside-in-degree* be the incoming edges from inside users. For example, in Figure 10 the *outside-in-degree* and *inside-in-degree* values of the edge "C→D" are 1 and 2, respectively, since there is an outside users' edges pointing to node C and there are two inside users' edges pointing to node D. (2) Relative degrees – let *outside-in-ratio* be *outside-in-degree divided by the total number of incoming edges* and *inside-in-ratio* be the *inside-in-degree divided by the total number of incoming edges*. For example in Figure 8 the *outside-in-ratio* and *inside-in-ratio* values of the edge "C→D" equals 1/5 and 2/3, respectively. If we separate users into different group based on locales and perform our proposed LABL (absolute) and IABL (relative) on whole network, the values of IABL (absolute) and IABL (relative) of each locale are given in Table 6.

As the Table 6 shows, both the IABL (absolute) and IABL (relative) suggest that there is significant positive correlation between incoming edges from outside users and incoming edges from inside users for each city. Only the result of IABL (absolute) for Minneapolis is not significant. Besides, the New York group and San Francisco group gets the highest IABL in both absolute and relative degrees measures. If we compare IABL (absolute) and IABL (relative), in almost every group the relative degrees approach is higher than absolute degrees approach. This result shows that all these groups contain a lot of pairs with over weighted result, which may be related to the locale feature, e.g., the New York and San Francisco groups have much difference between absolute and relative approaches and the Baltimore group has relative degrees smaller than absolute degrees.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we propose a series of locale based metrics, including locale clustering coefficient, inward locale transitivity, locale assortativity coefficient, and locale assortability coefficient, to support association analysis of users in a location-based social network. We conduct an analysis using these metrics and find that New York City and San Francisco have very strong localized locale clustering coefficients, which indicate high clustering effect among users in these two cities. We also observe that users having activities in cities with more trajectories receive attention from other users since this kind of cities has both many more active users and popular landscapes or scenic places to attract other users. We also confirm that people who are popular usually connect to the people who are also popular. Moreover, either conventional association analysis or locale based association analysis show similar results regarding users who have activities in San Francisco and New York City. As for the future work, we plan to further investigate other locale based metrics to enhance the analysis on LBSNs.

Table 6. Locale Assortability Coefficient.

*means significant

Cities	LABC (Global)	LABC (Localized)	IABL (Absolute)	IABL (Relative)
Austin	0.325	0.961*	0.133*	0.076*
San Francisco	0.356*	0.589*	0.207*	0.207*
Baltimore	no edge	no edge	0.075*	0.027*
Boulder	0.460	0	0.102*	0.063*
Chicago	0.707	0	0.093*	0.034*
Minneapolis	no edge	no edge	0.0004	0.016*
New York	0.562*	0.603*	0.258*	0.251*
Phoenix	0.254	0.251	0.053*	0.051*
Seattle	0.350	0.236	0.130*	0.094*
All Users	0.158			

6. ACKNOWLEDGMENTS

This research was supported by National Science Council, Taiwan, R.O.C. under grant no. NSC100-2631-H-006-002 and NSC100-2218-E-006-001.

7. REFERENCES

[1] Bikely: http://www.bikely.com/.
[2] EveryTrail: http://www.everytrail.com/.
[3] S. Boccalettia, V. Latorab, Y. Morenod, M. Chavez, D.-U. Hwang. Complex networks: Structure and dynamics. Physics Reports-Review Section of Physics Letters, vol. 424, pp. 175-308, Feb 2006.
[4] N. Li, G. Chen. Analysis of a Location-based Social Network. in *CSE*, 2009.
[5] Q. Li , Y. Zheng , X. Xie , Y. Chen , W. Liu , W.-Y. Ma, Mining user similarity based on location history, in *ACM GIS*, 2008.
[6] A. Mislove, M. Marcon, K. P. Gunnadi, P. Druschel, B. Bhattacharjee. Measurement and Analysis of Online Social Networks. in *IMC*, 2007.
[7] A. A. Nanavati, S. Gurumurthy, G. Das, D. Chakraborty, K. Dasgupta, S. Mukherjea, A. Joshi. On the Structural Properties of Massive Telecom Call Graphs: Findings and Implications. in *CIKM*, 2006.
[8] M. E. J. Newman. Assortative Mixing in Networks. Physical Review Letter, vol. 89, 2002.
[9] M. E. J. Newman. Mixing patterns in networks. Physical Review E, vol. 67, Feb 2003.
[10] T. Opsahla, F. Agneessens, J. Skvoretzc. Node centrality in weighted networks: Generalizing degree and shortest paths. Social Networks, vol. 32, pp. 245-251, July 2010.
[11] S. Scellato, C. Mascolo, M. Musolesi, V. Latora. Distance Matters: Geo-social Metrics for Online Social Networks. in *WOSN*, 2010.
[12] S. Scellato, A. Noulas, R. Lambiotte, C. Mascolo. Socio-spatial Properties of Online Location-based Social Networks. in *ICWSM*, 2011.
[13] D. J. Watts, S. H. Strogatz. Collective dynamics of 'small-world' networks. Nature, vol. 393, p. 3, June 1998.
[14] J. J.-C. Ying, E. H.-C. Lu, W.-C. Lee, T.-C. Weng, V. S. Tseng. Mining User Similarity from Semantic Trajectories. In *LBSN* , 2010.

Crowd-based Urban Characterization: Extracting Crowd Behavioral Patterns in Urban Areas from Twitter

Shoko Wakamiya
University of Hyogo
Hyogo, Japan
ne11n002@stshse.u-hyogo.ac.jp

Ryong Lee
National Institute of Information and Communications Technology
Nara, Japan
ryonglee@nict.go.jp

Kazutoshi Sumiya
University of Hyogo
Hyogo, Japan
sumiya@shse.u-hyogo.ac.jp

ABSTRACT

The advent of location-based social networking sites provides an open sharing space of crowd-sourced lifelogs that can be regarded as a novel source to monitor massive crowds' lifestyles in the real world. In this paper, we challenge to analyze urban characteristics in terms of crowd behavior by utilizing the crowd lifelogs in urban area. In order to collect crowd behavioral data, we utilize Twitter where enormous numbers of geo-tagged crowd's micro lifelogs can be easily acquired. We model the crowd behavior on the social network sites as a feature, which will be used to derive crowd-based urban characteristics. Based on this crowd behavior feature, we analyze significant crowd behavioral patterns for extracting urban characteristics. In the experiment, we actually conduct the urban characterization over the crowd behavioral patterns using a large number of geo-tagged tweets found in Japan from Twitter and report a comparison result with map-based observation of cities as an evaluation.

Categories and Subject Descriptors

J.4 [Social and Behavioral Sciences]: Sociology; H.1.2 [User/ Machine Systems]: Human factors processing; I.5.2 [Design Methodology]: Pattern analysis; H.2.8 [Database Applications]: Spatial databases and GIS

General Terms

Human Factors, Experimentation, Measurement

Keywords

Urban characteristics, Crowd behavior, Location-based social network sites

1. INTRODUCTION

The advent of location-based microblogging sites such as Twitter [1], Foursquare [2] and Gowalla [3] enables us to share our daily activities as well as our thoughts in an instant way. This kind of trend is explosively increasing due to the fast spread of smartphone and wireless network technologies. At the moment, in terms of geographic analytics such as landmark analysis and urban characterization, such social networking sites provide us a novel source capable of monitoring crowds' lifestyles in the real world through the massive local users' lifelogs.

In this work, we attempt to extract urban characteristics from the crowd-sourced data using Twitter. In fact, characterizing urban space is a critical issue relevant to our every activity and decision making in our living space. For instance, when we need to look into a place for moving in an unknown city, it takes an effort to quickly know the living environment drawing the image of the city in mind such as 'Is it safe and convenient for living?' or 'How far is the school for children?' Furthermore, in many practical geo-business or geo-politics, we have to know the overall feature of local area as early as possible. However, surveying such characteristics of urban space usually takes a huge cost and time involving off-line on-the-spot observation or gathering information by the questionnaire method. So far, we haven't acquired quick and massive scale investigation due to the extensibility problem. Instead, we only had to depend on the general conception of urban areas usually formed by the media or self-experience or statistics from national census data. For example, urban characteristics have been surveyed from limited aspects such as appearance of the space and landmarks.

Compared to those limited methodologies in conventional urban characterization research, location-based social networking sites have many significant advantages and benefits; first, there are enormous populations around the world who are voluntarily publishing their daily activities, particularly, in urban space. Second, the crowd-sourced data have various information from trivial travel diary to crowd's seasonal movement trajectory by analysis. These heterogeneous types of data shared over the social network sites can help us conduct various quantitative and qualitative research studies and realize practical systems. Obviously, the unprecedented scale of crowd lifelogs in urban space will promise many challenging and beneficial issues in the near future.

In this paper, we approach modeling of crowd behavior observable using location-based social networking sites. In general, crowd behavior in urban area is a critical factor to be considered when we conduct urban analytics. For instance, in residential districts, commuters expectedly become more active in morning and evening hours. On the other hand, at night in a bedroom town, crowd behavior would be much less active. Behind this speculation, we can easily imagine the strong relationship between the real space and the activities of citizen. Particularly, in a city space that we usually develop for our convenience, our activities such as commuting, working, shopping, educating, etc. will feature urban areas letting us know how we use the real space. Therefore, with crowd's daily lifelogs over social networking sites, we can conduct urban characterization by observing crowd behavior and finding significant behavioral patterns to know the usage types of urban space by crowds as depicted in Figure 1. Because crowd behavior observed from social networking sites is quite dynamic, we can expect to find more useful urban characteristics based on crowds in urban space. We primarily focus on extracting urban characteristics in terms of crowd behavior in the real world that can be largely available by exploiting geo-tagged tweets from Twitter where enormous numbers of geo-tagged crowd's micro lifelogs can be easily acquired. Specifically, we compute crowd behavior feature based on periodic occurrence pattern of geo-tagged tweets for a geographic region and monitor crowd behavior focusing on temporal changes in the region. We also examine significant crowd behavioral patterns for reasoning urban characteristics. Finally, we experimentally extract geographic crowd behavioral patterns from a large number of actually gathered geo-tagged tweets in Japan and report a comparison with real socio-geographic features of each region as an evaluation.

The remainder of this paper is organized as follows: Section 2 presents our challenge to make the most of the location-based social networking sites for extracting socio-geographic characteristics which significantly represent how people use their living urban spaces and surveys related work. Section 3 describes an overall process focusing on estimation of normal crowd behavioral patterns and clustering urban areas where similar patterns of crowd behavior. Section 4 illustrates the experiment that we conducted to extract crowd behavioral patterns as urban characteristics with a huge volume of data gathered from Twitter. Finally, section 5 concludes this paper with a brief description of future work.

2. CHARACTERIZING URBAN AREAS USING SOCIAL NETWORKS

2.1 Urban Characterization based on Crowd Behavior Extracted from Social Network Sites

In this section, we explain what we mean by the term 'urban characteristic' in this paper which will be explored from social network sites. Especially, we describe what kinds of contexts we could obtain from the crowd-sourced social network sites.

Generally, in urban space, there are a lot of facilities for living such as housings, transportation, offices, schools, parks and shopping centers. In such complex space, we can easily observe some daily routines such as commuting, working, eating and drinking, and relaxing at home by exploiting crowd-sourced lifelogs over social networking sites as illustrated in Figure 1. In addition, these crowd activities let us know roles or functions of the urban space; an urban area ob-

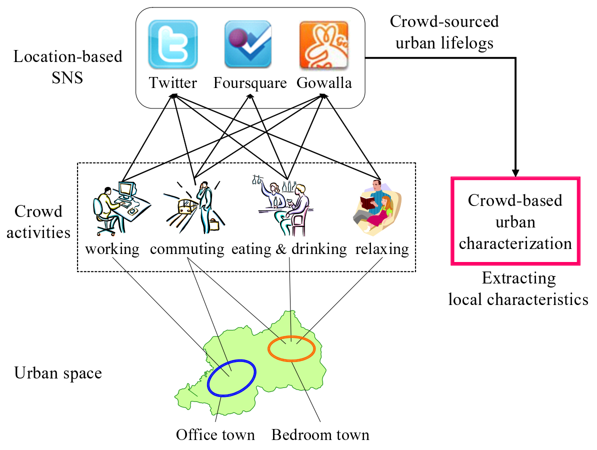


Figure 1: Research model for urban characterization based on crowd behavior from location-based social network sites

served crowds commuting and working would be estimated an office town. On the other hand, if crowds commuting, eating and drinking, and relaxing at home are monitored in the urban area, we might regard there as a bedroom town. Like this, we often capture the image of urban space by means of crowd behavior.

In fact, geographical characteristics have been researched so far from various perspectives by physical geographic shape or diverse objects such as streets or landmarks, or by cultural and structural aspects such as residential, commercial, and industrial districts. These two different views have been well studied in many research fields. Kevin A. Lynch's seminal contribution in his book titled "The Image of the City" [12] defined five fundamental elements of a city; paths, edges, districts, nodes, and landmarks. Based on these elements, Lynch thought that we could characterize our living space within the appearance of a city to imagine ourselves living and working there as shown on the left side in Figure 2. In another remarkable work describing a way to extract geographic characteristics, Tezuka et al. [15] extracted geographic objects and their roles frequently mentioned on the Web contents which can be regarded as a mirror of the crowds' minds to the real world as shown in the middle of Figure 2.

These two different types of urban characteristics seem to be obviously useful to grasp the image of the city. However, they are focusing on static elements of the city and out of touch to a dynamic element of the city; "crowds" living there which is a critical factor for observing urban characteristics. For instance, in the case of the recent big earthquake and tsunami in Japan, although lots of static characteristics in Fukushima prefecture remain unchanged, but the image of the city based on crowds living there was drastically changed because of nuclear accident induced by the horrible natural disasters.

Therefore, we distinguishably attempt to derive a new kind of social geographic characteristics based on crowd, especially in terms of their behavior, by utilizing the location-based social network sites. Needless to say, on such sites, there are plenty of crowd-sourced data dynamically reflecting the real world, especially involving crowd behavior in local areas. In addition, crowd behavioral logs would be difficult to grasp crowd activity by analyzing GPS-based lo-

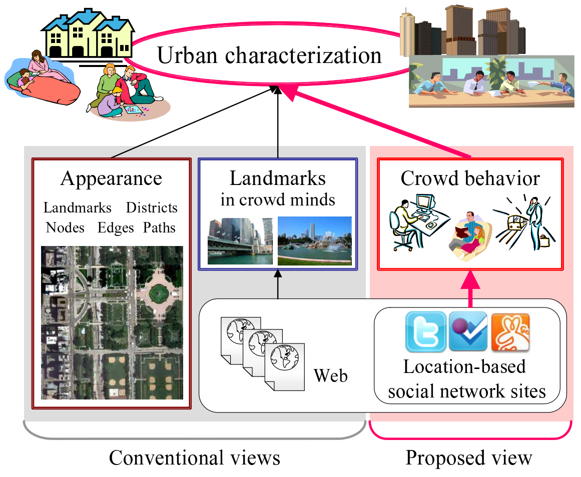


Figure 2: Three views of urban characteristics

cation data only, since the location data has been regularly monitored regardless of crowd activity. Furthermore, it is not easy to obtain the GPS-based location data of massive crowd due to the privacy issue and cost problem. On the other hand, geo-tagged microblogs over social networking sites can be easily collected since these are shared in the open spaces. Therefore, we can enough utilize the free data to extract the image of cities based on crowd activity in real world.

2.2 Related Work

Recently, social networking sites are regarded as a novel source which enables us to acquire lots of daily crowd lifelogs almost freely, while we still have to be aware of the privacy concerns on the individual data. However, in terms of social and individual benefits from the new open sharing space, it must be a critical issue to be researched and explored in the academic field as well as the business world.

As for geographic query processing based social knowledge, Zheng et al. [17] showed a method to mine knowledge using the location data based on GPS and users' comments at various locations to answer two typical socio-geographic questions: If we want to do something, where shall we go? If we visit some place, what can we do there? By modeling the location-activity relation into a matrix, the former question is answered by activity recommendation given location query, and the latter one is solved by location recommendation given activity query. In this work, authors focused on determining the correlation between locations and crowd activities, while we are focusing on regular crowd activities on locations.

Some researches focusing on cooperation with Twitter for some natural incidents in real world have been introduced. De Longueville et al. [8] analyzed the temporal, spatial and social dynamics of Twitter activity during a major forest fire event in the South of France in July 2009. Sakaki et al. [14] constructed an earthquake reporting system in Japan using tweets which are posted from each Twitter user regarded as a sensor. In this method, tweets which are reporting the occurrence of an earthquake are extracted by using hashtags. Furthermore, in our previous work [9][10], we proposed a method for discovering social events based on user movement histories and their activity from massive micro-bloggers over Twitter.

3. EXTRACTING URBAN AREA CHARACTERISTICS BASED ON CROWD BEHAVIOR OVER TWITTER

3.1 Extracting Crowd Behavior from Location-based Social Network Sites

First of all, we need to gather geo-tagged tweets from Twitter to observe crowd behavior in the real world. However, it takes a considerable amount of efforts to acquire a significant number of geo-tagged tweets because of certain practical limitations: first, Twitter's open API [4] solely supports the simplest near-by search by means of the specification of a center location and a radius. Furthermore, each query can only obtain a maximum of 1,500 tweets for a past week. Therefore, in order to overcome these restrictions and perform periodic monitoring of any user-specified regions, we utilized a geographical microblog gathering system we developed [10] which can monitor crowd behavior for a specific region of any size by automatically partitioned regions depending on the density of tweets as depicted in Figure 3.

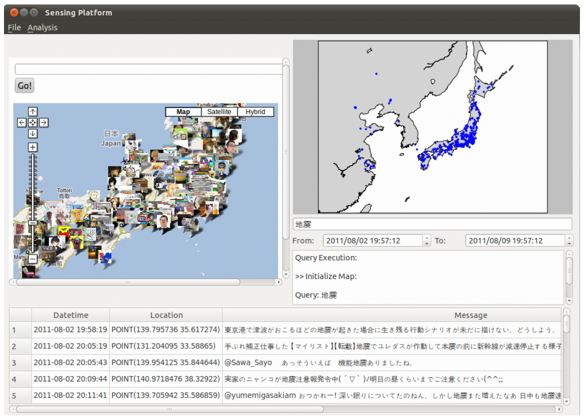
The location information of geo-tagged tweets can be received either in raw text form or in very precise location coordinates. Hence, in the case of the former textual style, we needed to perform geo-coding to identify the exact coordinates by translating place names into the corresponding exact locations. We were able to solve the problem easily by using another mash-up service with Google Map's geocoding service [5]. We directly transferred the place names to this conversion service and received the precise coordinates. Subsequently, we could accurately determine when and where each tweet was written. For instance, a map on the right side of Figure 3 shows geo-tagged tweets published around Japan for a specific time period.

In fact, geo-tagged tweets have various attributes such as pictures used for user icons, devices utilized for posting the tweets, the number of followers (people who receive the user's tweets), the number of following (someone on Twitter), hashtags, and URLs to external media, etc. as well as textual message, location information, user ID and posted time. In this work, however, we refer to location information, user ID and posted time since it would be enough for constructing primitive crowd behavior feature.

3.2 Setting Urban Areas by Socio-geographic Boundaries

Next, in order to decide geographic urban areas for computing crowd behavior to extract urban characteristics, we have to partition a given region into sub-areas by examining crowd behavior in those areas. As for how to partition the region, however, there are several different space-partitioning methods; for instance, administrative area based one, grid-based one, and cluster-based one. Hence, we should select the most appropriate method for our goal.

Administrative area based method is to split a target region into prefectures and municipalities by administrative boundaries; limits or borders of a geographical area under the jurisdiction of some governmental or managerial entity as shown in Figure 4 (a). Therefore, it is difficult to figure out if crowd behavior areas are relevant or almost dependent



地震

From: 2011/08/02 19:57:12 To: 2011/08/09 19:57:12

Query Execution:

>> Initialize Map:

Query: 地震

	Datetime	Location	Message
1	2011-08-02 19:58:19	POINT(139.795736 35.617274)	東京港で津波がおこるほどの地震が起きた場合に生き残る行動シナリオが未だに描けない。どうしよう。
2	2011-08-02 20:05:19	POINT(131.204095 33.58865)	手ぶれ補正仕事した【マイリスト】【転載】地震でユレダスが作動して本震の前に新幹線が減速停止する様子
3	2011-08-02 20:05:43	POINT(139.954125 35.844644)	@Sawa_Sayo　あっそういえば　機能地震ありましたね。
4	2011-08-02 20:09:44	POINT(140.9718476 38.32922)	実家のニャンコが地震注意報発令中(´▽`)ﾉ明日の昼くらいまでご注意ください(^^;;
5	2011-08-02 20:11:41	POINT(139.705942 35.586859)	@yumemigasakiam おつかれー! 深い眠りについてたのねん。しかし地震また増えたなあ 日中も地震速

Figure 3: Geo-tagged lifelogs acquired by the geographical microblog gathering system

on administrative areas, because crowd behavior often easily cross over the administrative boundaries. Accordingly, we consider that this method might be inadequate for examining crowd behavior. Next, as for grid-based method, it is difficult to decide the adequate cell size because a grid is formed by a lot of equally-sized cells as shown in Figure 4 (b). In addition, it would consume considerable unnecessary costs for observing crowd behavior since it does not consider nonuniform distribution of crowds.

On the other hand, in the case of a cluster-based space partition method, it can reflect the geographical distribution of crowds based on location information included in geo-tagged tweets and deal with heterogeneous regions differently as shown in Figure 4 (c). As a result of this approach, small urban areas of densely populated areas appeared around major cities; for example, Tokyo, Nagoya, and Osaka in Japan. In contrast, large urban areas of sparsely populated areas are spread over the other suburban areas or surrounding sea. Thus, we can effectively establish the appropriate socio-geographic boundaries for the target region and partition into urban areas by referring to the actual geographic crowd behavior.

For these reasons, in this paper, we selected the cluster-based space partition method for analyzing crowd behavior. Especially, in our experiment described later, we adopted the K-means clustering method [7] based on the geographical frequency of geo-tagged tweets. In addition, we depicted a voronoi diagram [11] using the center points (latitude, longitude) of the K-means results and regarded the formed rectangular regions as a set of urban areas.

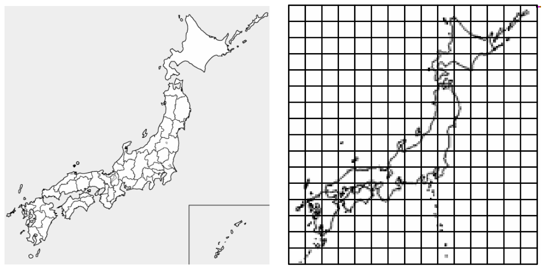
3.3 Extracting Crowd Behavioral Patterns

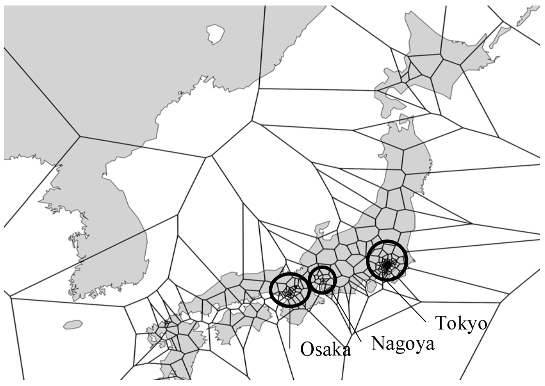
Under the assumption that it would be possible to grasp urban characteristics using geo-tagged tweets obtained from Twitter, we present a method to summarize ordinary or usual local crowd behavior in a succinct representation. Specifically, we first generate a crowd behavior vector in each urban area for modeling crowd behavior. For this, we define three primitive behavior features computable by geo-tagged tweets as follows:

$\#Tweets$: The total number of tweets occurring inside of an urban area.

$\#Crowd$: The number of distinct Twitter users found in an urban area. This feature is computed with user IDs attached to geo-tagged tweets. Naturally, the in-equality $\#Crowd \leq \#Tweets$ is valid since any individual user can write one or more tweets during a specific period of time.

$\#MovCrowd$: The number of moving users in an urban area. To be more precise, moving crowd is a set of users who posted two or more tweets at different locations in the area. This feature is also computed with user IDs of geo-tagged tweets. Obviously, the in-equality $\#MovCrowd \leq \#Crowd$ is valid. This feature is the most dynamic parameter which can explicitly figure out the temporal usage of space.


(a) Administrative area　　(b) Grid



(c) Cluster

Figure 4: Three ways of space partitioning to monitor crowd activities

Crowd behavior vector for an urban area r_i is represented based on these three features; $cbv(r_i) = <\#Tweets, \#Crowd, \#MovCrowd>$. In general, crowd behavior would vary according to time slot of a day. Therefore, in order to conduct periodic monitoring of crowd behavior features, we should split a day into certain time period. In this paper, we empirically set 6-hour by splitting a day into four equal time slots: morning (M, 06:00–12:00), afternoon (A, 12:00–18:00), evening (E, 18:00–24:00), and night (N, 24:00–06:00). By this, a feature is represented as follows: $feature = (feature_M, feature_A, feature_E, feature_N)$.

In addition, since we measure the crowd behavior features using a long-term dataset, it is natural that each crowd behavior feature in a specific time slot has variation values as shown in three graphs in dotted line frame of Figure 5. When a monitoring period is set 30 days, for example, $\#Tweets$ is represented by $\#Tweets_M = (\#Tweets_{M1}, ..., \#Tweets_{M30})$, $\#Tweets_A = (\#Tweets_{A1}, ..., \#Tweets_{A30})$, $\#Tweets_E = (\#Tweets_{E1}, ..., \#Tweets_{E30})$, $\#Tweets_N$

$= (\#Tweets_{N1}, ..., \#Tweets_{N30})$.

In order to summarize crowd behavior features, we deal with all the features in a statistical manner by using a boxplot [13], which is primarily used for explicitly visualizing a data distribution. For a temporally changing data during a time slot, boxplot-based summarization generates a bounding range that consists of five values; maximum, upper quartile, median, lower quartile, and minimum. We depict three values out of five values; upper quartile, median, and lower quartile Accordingly, summarized crowd behavior features are represented as follows: $boxplot(feature_p) = (max, u\_quartile, median, l\_quartile, min)$, where *boxplot* is a function calculating a distribution of training data, $feature$ is a variable meaning one of crowd behavior features, and p is a variable meaning a time slot; M, A, E, or N as shown in the second frame from the top in Figure 5.

Subsequently, we define characteristics of urban areas by patterns of crowd behavior in the areas. Therefore, we respectively estimate crowd behavioral patterns in urban areas using three features of crowd behavior vectors. Although each crowd behavior feature is composed by multiple values, we depict relative temporal changes of the feature. Specifically, we compute the differences between two consecutive periods of time; from morning to afternoon (A–M), from afternoon to evening (E–A), and from evening to night (N–E) to denote urban characteristics. For the sake of simplification, we created temporal changes by considering the median values of crowd behavior features only as shown in the three graphs in the second frame from the bottom in Figure 5. By this, we can observe normal crowd behavioral patterns respectively in each urban area.

3.4 Categorizing Urban Areas based on Crowd Behavioral Patterns

On the basis of crowd behavioral patterns in urban areas, we characterize urban areas. For this, we first classify urban areas where observed similar patterns of crowd behavior, that is, urban characteristics based on crowd. That is because we could consider such areas would provide significant similar roles and functions for crowds living and utilizing there. Then, we characterize urban areas by reasoning crowd behavioral patterns.

In order to look for urban areas causing similar crowd behavioral patterns, we should compare with crowd behavioral patterns in urban areas. In general, the scale of crowd behavior would be different depending on crowds in urban areas; periodic occurrence of geo-tagged tweets or the absolute number of crowds are different in each urban space. However, two or more regions can be similar in terms of an increasing and decreasing tendency, for example, from night to morning there can be increasingly crowded places such as a large station. Therefore, we can extract urban characteristics utilizing patterns. However, because the ranges of each region can be different and their comparison can be complicated, we must consider a much simpler expansion.

Accordingly, we find out common patterns in the crowd behavioral patterns. Since each crowd behavioral pattern is established by a series of the temporal changes of three crowd behavior features, the maximum of possible combinations would be large ($= 3^9$) and the computational cost to find out the common full-size or partial patterns would be unbearable. In order to reduce such cost, we applied a frequent itemset mining algorithm [16] for statistically looking

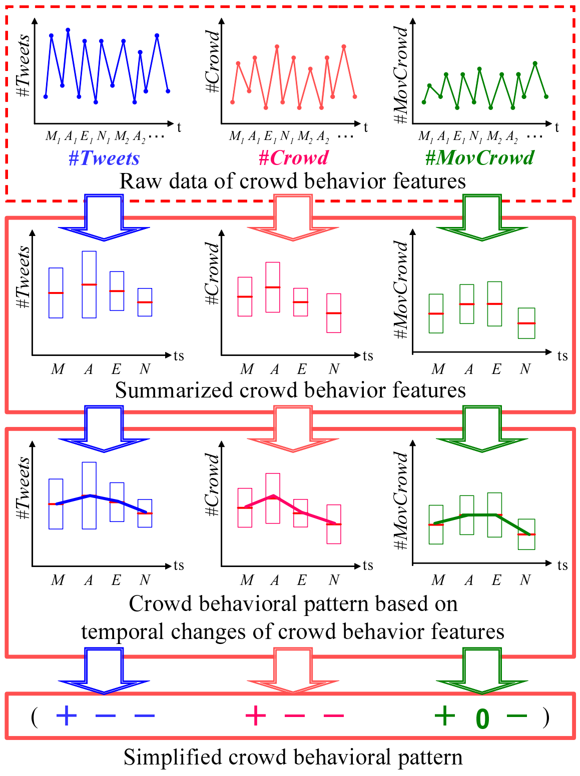

#Tweets
#Crowd
#MovCrowd
$M_1 A_1 E_1 N_1 M_2 A_2 \cdots$
t
Raw data of crowd behavior features
ts
M A E N
Summarized crowd behavior features
Crowd behavioral pattern based on temporal changes of crowd behavior features
(+ – – + – – + 0 –)
Simplified crowd behavioral pattern

Figure 5: Extracting crowd behavioral patterns based on temporal changes of crowd behavior features

for them to the full size of the pattern.

Again, to observe the change, we kept it simple by only looking up the method of change by transforming the representation to a symbol list such as (+, –, 0), which respectively increases from morning to afternoon, decreases from afternoon to evening, and indicates no change from evening to night. Indeed, the temporal changes dynamically can signify crowd behavior so that there can be many combinations such as (+, –, –) and (+, 0, –), as illustrated at the bottom of Figure 5.

4. EXPERIMENT

In this section, we describe our experiment to extract the characteristic patterns of crowd which was modeled in the section. Interestingly, we were able to gather agent deal of geo-tagged tweets from Twitter in the geographic range of Japan. We computed the aforementioned geographic regularities through the indirective indicators and extracted their common changing patterns. Lastly, we successfully confirmed that our experimentation to utilize the social network based crowd behavior as an estimator for urban characterization was achieved by comparing with each geographic regions feature on a map.

4.1 Data Collection

First, we obtained geo-tagged tweets from Twitter for one month between Jul. 1st, 2010 and Jul. 31st, 2010 around Japan with the latitude range [30.004609:45.767523] and the longitude range [116.27921:148.381348] using our geographical microblog gathering system as shown in Figure 3. As a

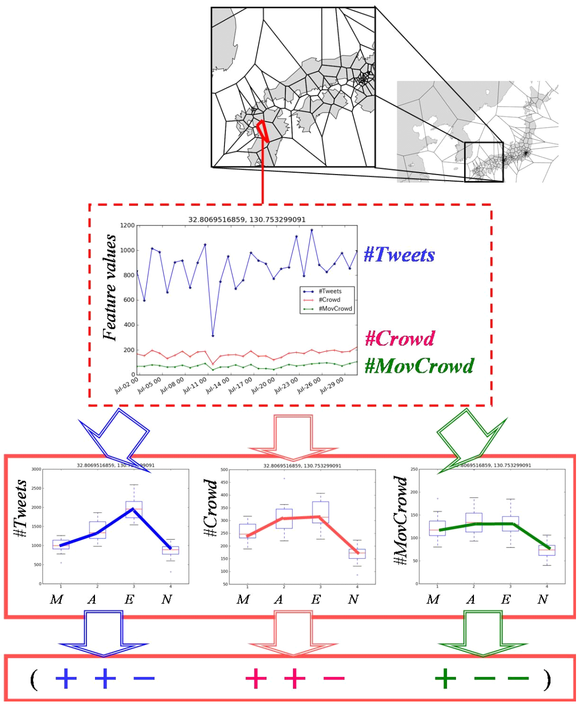


Figure 6: An example of crowd behavioral pattern using real dataset

result, we could acquire 11,632,750 geo-tagged tweets from 211,361 distinct users.

Next, we constructed socio-geographic boundaries using the obtained crowd behavior and divided the target space into 300 urban areas by a K-means clustering method and a voronoi diagram, here the value of K empirically set 300 as shown in Figure 4 (c).

4.2 Reasoning Urban Characteristics

We analyzed crowd behavior in 300 urban areas respectively based on three features; $\#Tweets$, $\#Crowd$, and $\#MovCrowd$, for every 6-h time slot for the period from the training period. Figure 6 shows an example of a crowd behavioral pattern established by temporal changes of crowd behavior features monitored periodically for an urban area.

We obtained 300 crowd behavioral patterns from temporal changes of crowd behavior features for the same number of areas. Then, in order to find common patterns, we applied a frequent itemset mining algorithm [16], statistically constructing it to full size: 9 items. In this experiment, we extracted 8 types of common patterns whose occurrence ratios were over a threshold (here, set at 2 %), as shown in Table 1. This covered 86.3 % of all 300 urban areas. We finally extracted 4 types of common patterns based on the partial similarity of 8 patterns.

On the basis of these frequent patterns, we clustered urban areas and empirically called them "bedroom towns," "office towns," "nightlife towns," and "multifunctional town" as shown in Table 2. We explain these four types in detail as follows.

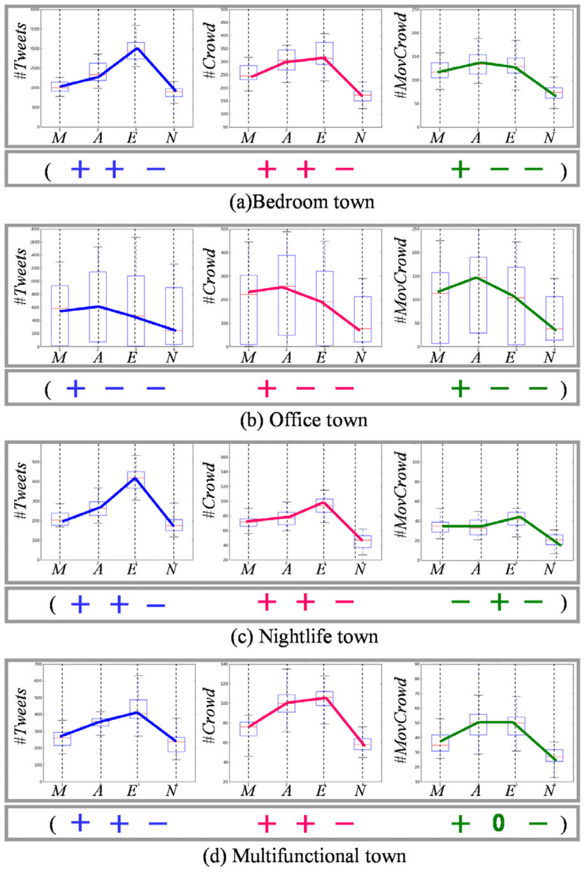
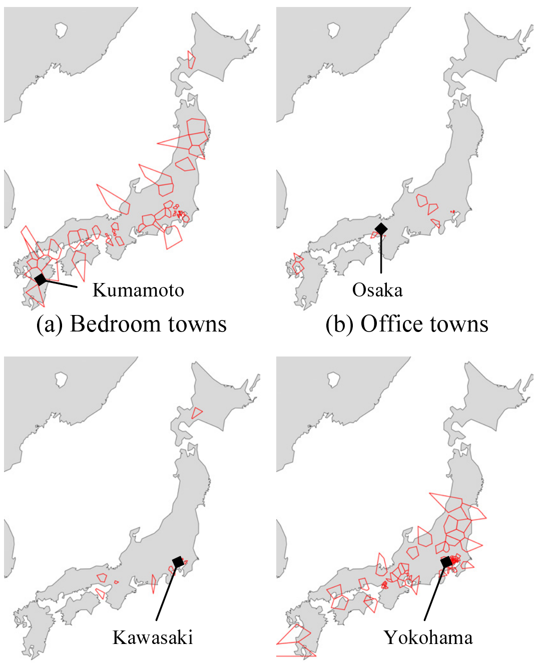


Figure 7: Examples of significant crowd behavioral patterns

(a) Bedroom towns: As shown in Table 2 (a), both the number of tweets and the number of crowd continue to increase from the morning to evening. On the other hand, the number of moving crowds is increasing from the morning to the afternoon, but is decreasing from afternoon to evening. In short, more people exist in and come to the areas, but they do not move actively there in the evening; the crowd behavioral pattern may be caused by crowds returning home after work or school. Therefore, we called the urban areas "bedroom towns," where active crowds aggregate until the evening and then calm down at night.

(b) Office towns: As shown in Table 2 (b), all three features; the number of tweets, the number of crowd, and the number of moving crowd, mostly are increasing from morning to afternoon, and are decreasing from afternoon to night. In other words, more people exist in come to the areas and move actively there in the afternoon, and then leave there in the evening; the behavioral pattern can be regarded as that of crowds coming to work. Therefore, we named the areas " office towns," where active crowds gather in the afternoon and disperse since the evening.

(c) Nightlife towns: Table 2 (c) shows both the number of tweets and the number of crowd continue to increase from morning to evening. On the other hand, the number of moving crowd is decreasing from morning to afternoon, but is increasing from afternoon to evening. That is, more people exist in and come to these areas in the evening. However, they do not move actively



(a) Bedroom towns (b) Office towns

(c) Nightlife towns (d) Multifunctional towns

Figure 8: Geographic distribution of characterized urban areas

there before the evening and then become active in the evening. The crowd behavioral pattern may be caused by crowds who come in the evening and originally staying there for the newcomer. Therefore, we called the areas "nightlife towns," where crowds calm down before the evening and active crowds aggregate since the evening.

(d) Multifunctional towns: As shown in Table 2 (d), all three features; the number of tweets, the number of crowds, and the number of moving crowds continue to increase from morning to evening, and then these are decreasing from afternoon to night. In other words, more people exist in and come to the areas, move active there until the evening, and then leave there at night; the behavioral pattern would be caused by crowds with various purposes. Therefore, we called the areas "multifunctional towns," where active crowds aggregate almost all day long, and that provide various functions and roles to support the lifestyles of many crowds.

In addition, we showed distributions of urban areas where were clustered into the four urban area types on a map depicted respectively in Figure 8. According to this figure, urban areas which are characterized as (a) Bedroom towns or (d) Multifunctional towns are dispersed all around the Japan. In contrast, (b) Office towns or (c) Nightlife towns concentrated on the major cities in Japan, such as Tokyo, Nagoya, and Osaka.

We also showed examples of these four types using the four urban areas where shown by diamond marks on the maps in Figure 8. Lastly, in order to confirm the correctness of the characterization above, we looked into satellite images of the Earth's surface provided by Google Earth [6] and got interesting results as depicted in Figure 9. In the case of the area

Table 1: Relative change patterns extraction by the frequent itemset mining algorithm

pattern	*#Tweets*			*#Crowd*			*#MovCrowd*			occurrence ratio
	A-M	*E-A*	*N-E*	*A-M*	*E-A*	*N-E*	*A-M*	*E-A*	*N-E*	
2	+	+	−	+	+	−	+	−	−	20.0
8					0					2.7

(a) Bedroom town

pattern	*#Tweets*			*#Crowd*			*#MovCrowd*			occurrence ratio
	A-M	*E-A*	*N-E*	*A-M*	*E-A*	*N-E*	*A-M*	*E-A*	*N-E*	
4	+	+	−	+	−	−	+	−	−	6.3
7		−								3.0

(b) Ofiice town

pattern	*#Tweets*			*#Crowd*			*#MovCrowd*			occurrence ratio
	A-M	*E-A*	*N-E*	*A-M*	*E-A*	*N-E*	*A-M*	*E-A*	*N-E*	
5	+	+	−	+	+	−	−	+	−	3.7
6							0			3.3

(c) Nightlife town

pattern	*#Tweets*			*#Crowd*			*#MovCrowd*			occurrence ratio
	A-M	*E-A*	*N-E*	*A-M*	*E-A*	*N-E*	*A-M*	*E-A*	*N-E*	
1	+	+	−	+	+	−	+	+	−	38.0
3								0		9.3

(d) Multifunctional town

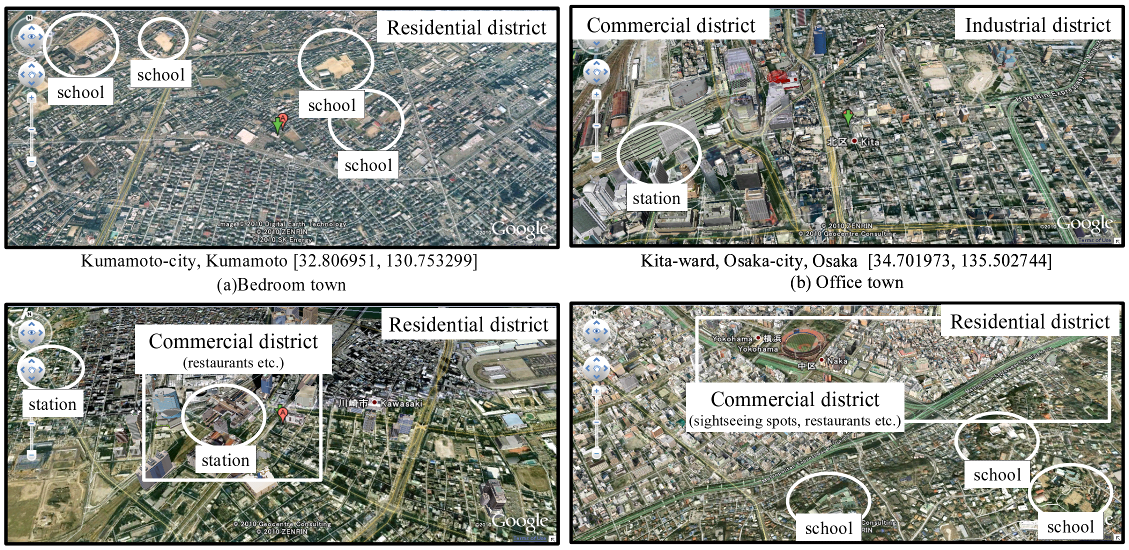
Table 2: Common change patterns used for clustering urban areas

pattern	*#Tweets*			*#Crowd*			*#MovCrowd*			occurrence ratio
	A-M	*E-A*	*N-E*	*A-M*	*E-A*	*N-E*	*A-M*	*E-A*	*N-E*	
1	+	+	−	+	+	−	+	+	−	38.0
2	+	+	−	+	+	−	+	−	−	20.0
3	+	+	−	+	+	−	+	0	−	9.3
4	+	+	−	+	−	−	+	−	−	6.3
5	+	+	−	+	+	−	−	+	−	3.7
6	+	+	−	+	+	−	0	+	−	3.3
7	+	−	−	+	−	−	+	−	−	3.0
8	+	+	−	+	0	−	+	−	−	2.7

of Kumamoto-city as shown in Figure 9 (a), we can see an extensive residential district and four schools. Actually, the sceneries like this area could be often found in the bedroom town. As for the area of Kita-ward, Osaka-city as shown in Figure 9 (b), we can find lots of office buildings around a big station, and both commercial and industrial districts. These huge components like buildings are especially confirmed in commercial districts where lots of high office buildings are standing. In the case of the area of Kawasaki-city as shown in Figure 9 (c), we are able to see a station and a large restaurant district. In many cases, the areas surrounding stations are for eating and drinking places opening late like bars. As for the area of Yokohama-city as shown in Figure 9 (d), we can find various types of districts as sightseeing spots, restaurant streets, a park, a stadium, a garden, and some schools. These areas would provide multiple functions for living, education, etc.

5. CONCLUSION

In this paper, we proposed a crowd-based urban characterization method based on crowd-sourced data obtained from social networking sites. In our proposed method, we collected geo-tagged tweets from Twitter as crowd's behavioral logs in real world and modeled crowd behavior in terms of primitive features extracted from Twitter. Then, we estimated geographic crowd behavioral patterns and classified urban areas by common behavioral patterns. Furthermore,



Kumamoto-city, Kumamoto [32.806951, 130.753299]
(a)Bedroom town

Kita-ward, Osaka-city, Osaka [34.701973, 135.502744]
(b) Office town

Kawasaki-city, Kanagawa [35.530512, 139.699034]
(c) Nightlife town

Yokohama-city, Kanagawa [35.438838, 139.641048]
(d) Multifuctional town

Figure 9: Comparing Google Earth [6] aerial photographs of characterized urban areas

on the basis of our proposed method, we conducted the experiment using massive geo-tagged tweets and categorized urban areas into four types; “bedroom towns,” “office towns,” “nightlife towns,” and “multifunctional towns.”

In the future work, we will extend our work by using textual messages of tweets to support the validity of our proposed method. Furthermore, we will investigate further crowd behavior patterns affected by various social and natural events.

6. ACKNOWLEDGMENTS

This research was supported in part by the Microsoft Research IJARC Core Project.

7. REFERENCES

[1] Twitter: http://twitter.com/.

[2] Foursquare: https://foursquare.com/.

[3] Gowalla: http://gowalla.com/.

[4] Twitter Open API: http://apiwiki.twitter.com/Twitter-Search-API-Method%3A-search.

[5] Google Geocoding API: http://code.google.com/intl/ja/apis/maps/documentation/geocoding/.

[6] Google Earth: http://www.google.com/earth/index.html.

[7] M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars. Computational Geometry: Algorithms and Applications (Third edition). Springer-Verlag, 2008.

[8] B. De Longueville, R. S. Smith, and G. Luraschi. "OMG, from here, I can see the flames!": a use case of mining location based social networks to acquire spatio-temporal data on forest fires. In Proceedings of the 2009 International Workshop on Location Based Social Networks, LBSN ’09, pages 73–80, New York, NY, USA, 2009. ACM.

[9] R. Lee and K. Sumiya. Measuring geographical regularities of crowd behaviors for Twitter-based geo-social event detection. In Proceedings of the 2nd ACM SIGSPATIAL International Workshop on Location Based Social Networks, LBSN ’10, pages 1–10, New York, NY, USA, 2010. ACM.

[10] R. Lee, S. Wakamiya, and K. Sumiya. Discovery of unusual regional social activities using geo-tagged microblogs. World Wide Web (WWW) Special Issue on Mobile Services on the Web, 14(4):321–349, March 2011.

[11] S. P. Lloyd. Least squares quantization in PCM. IEEE Transactions on Information Theory, 28(2):129–137, March 1982.

[12] K. Lynch. The Image of the City. The MIT Press, 1960.

[13] R. Mcgill, J. W. Tukey, and W. A. Larsen. Variations of box plots. The American Statistician, 32(1):12–16, February 1978.

[14] T. Sakaki, M. Okazaki, and Y. Matsuo. Earthquake shakes Twitter users: real-time event detection by social sensors. In Proceedings of the 19th international conference on World wide web, WWW ’10, pages 851–860, New York, NY, USA, 2010. ACM.

[15] T. Tezuka and K. T. and. Landmark extraction: a Web mining approach. spatial information theory. COSIT, 3693:379–396, 2005.

[16] J. X. Yu, Z. Li, and G. Liu. A data mining proxy approach for efficient frequent itemset mining. The VLDB Journal, 17:947–970, July 2008.

[17] V. W. Zheng, Y. Zheng, X. Xie, and Q. Yang. Collaborative location and activity recommendations with GPS history data. In Proceedings of the 19th international conference on World wide web, WWW ’10, pages 1029–1038, New York, NY, USA, 2010. ACM.

Towards Trajectory-Based Experience Sharing in a City

Byoungjip Kim[1], Youngki Lee[1], SangJeong Lee[1], Yunseok Rhee[2], Junehwa Song[1]

[1]Dept. of Computer Science
Korea Advanced Institute of Science and Technology (KAIST)
{bjkim, youngki, peterlee, junesong}@nclab.kaist.ac.kr

[2]Dept. of Digital Information Engineering
Hankuk University of Foreign Studies
rheeys@hufs.ac.kr

ABSTRACT

As location-aware mobile devices such as smartphones have now become prevalent, people are able to easily record their trajectories in daily lives. Such personal trajectories are a very promising means to share their daily life experiences, since important contextual information such as significant locations and activities can be extracted from the raw trajectories. In this paper, we propose MetroScope, a *trajectory-based real-time and on-the-go experience sharing system* in a metropolitan city. MetroScope allows people to share their daily life experiences through trajectories, and enables them to refer to other people's diverse and interesting experiences in a city. Eventually, MetroScope aims to satisfy users' ever-changing interest in their social environments and enrich their life experiences in a city. To achieve real-time, on-the-go, and personalized recommendation, we propose an approach of *monitoring activity patterns over people's location streams*.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications – *Spatial databases and GIS*. H.3.5 [**Information Storage and Retrieval**]: On-line Information Services – *Data sharing*.

General Terms

Design, Experimentation, Human Factors.

Keywords

Location-based social networks, Experience sharing, Activity pattern monitoring.

1. INTRODUCTION

Social networking applications such as Facebook [2] and Twitter [5] have become one of the most important Web services, significantly changing the way people interact and consume news from each other. Recently, as location-aware mobile devices such as smartphones and tablet PCs have become prevalent, *location-based social networking applications* such as Foursqure [3], EveryTrail [1], and GeoLife [19] are emerging. In such location-based social networking applications, people want to share their location information and obtain reference information from other users' travel experiences. Many researchers expect that such location-based social networking applications will significantly enrich our daily lives by expanding the huge benefits of online social networks into our physical world.

Currently, the most popular types of location-based social networking applications are *location sharing applications* such as Foursquare [3] that allow people to simply share their current locations with their friends. However, in many cases, people want to share not only separate locations but also *whole experiences* in their daily lives. For example, while touring a city, people usually visit a sequence of places and want to share their experiences comprised of many activities happening at various places. More important, some people may want to recommend their significant traveling or dating experiences not only to friends but also to many other people. There are only a few *trajectory sharing applications* such as EveryTrail [1] and ShareMyRoutes [4]. However, these existing trajectory sharing applications are still insufficient to satisfy the abovementioned scenarios, mainly because they (1) share only raw GPS trajectories and (2) require people to search interesting trajectories through Web browsers by themselves. Even worse is that since the trajectory search function allows only coarse-grained search such as at the city level, it takes significant amount of time to find an interesting trajectory among many candidates, especially for on-the-go people.

In this paper, we propose MetroScope, a *trajectory-based real-time and on-the-go experience sharing system* in a metropolitan city. MetroScope allows people to share their daily life experiences such as hiking, touring, and dating by uploading their trajectories recorded by GPS- or Wi-Fi-enabled mobile devices to a sharing server. Basically, a trajectory is a series of location points with timestamps, but it can include additional contents such as photos or notes taken at some interesting locations. In MetroScope, trajectory-based experience sharing is performed through two main operations: *experience export* and *experience recommendation*. Experience export is a set of procedures that make a recorded personal trajectory public. Experience recommendation is a process to help people find other's relevant and interesting experiences.

To achieve real-time, on-the-go, and personalized recommendation, we propose the approach of *monitoring activity patterns over people's location streams*. The basic idea is that people will highly likely be interested in some experiences of other people who show similar activity patterns to them. For example, a young woman who is dating with her boyfriend (e.g., eating at a restaurant and watching a movie at a cinema) would be more interested in other nice and new dating courses rather than a city tour experience for foreigners (e.g., visiting imperial palace and memorial hall). More specifically, MetroScope automatically extracts semantic information such as significant locations and activities from the uploaded trajectories, and generates a recommendation rule based on the extracted information. Basically, a recommendation rule is a combination of people's activity pattern and profile. Then, MetroScope finds people who match the recommendation rule, and recommends the associated experience to the matched people.

There are two main challenges in comprehensively realizing MetroScope. First, we should accurately extract significant locations or activities from trajectories provided by users. However, it requires sophisticated processing, since a raw trajectory usually includes many errors and it is often hard to

ACM LBSN '11, November 1, 2011. Chicago, IL, USA

Copyright © 2011 ACM ISBN 978-1-4503-1033-8/11/11...$10.00

accurately determine the visited places or real activities. Fortunately, many research efforts have been devoted to mining locations and activities from trajectories [6][18][14]. Basically, we exploit existing solutions to implement MetroScope. Second, we should efficiently monitor activity patterns from location streams to recommend relevant and interesting experiences to people in a real-time manner. However, such continuous monitoring of location streams is very challenging since there are a huge number of people to monitor and a large number of monitoring rules to execute in the system. Therefore, we argue that it is important to develop scalable activity pattern monitoring techniques such as continuous query indexing and complex event processing.

The rest of the paper is organized as follows. Section 2 briefly discusses some related work. Section 3 presents a system overview of MetroScope. Finally, Section 4 concludes this paper.

2. RELATED WORK

Location Mining from Trajectories. Identifying significant locations such as visit places from a trajectory is one of the most required techniques, and has been studied by many researchers. The main approach to this problem is to apply clustering algorithms such as k-means and density-based clustering to trajectories. Ashbrook et al. [6] proposed a variant of k-means clustering in order to cluster GPS trajectories taken over an extended period of time into meaningful locations at multiple scales. Zheng et al. [18] proposed Tree-Based Hierarchical Graph (TBHG) for modeling multiple users' location histories. Using a density-based clustering algorithm over multiple users' trajectories, they hierarchically cluster location points into some geospatial regions. Thus, the similar stay points from various users would be assigned to the same clusters on different levels.

Activity Mining from Trajectories. Identifying activities from trajectories is another important problem. It is more challenging than identifying locations, since activities inherently include the high level of uncertainty. The basic idea to handle the problem is that people's daily activities can be inferred from contextual information such as location, time, and duration. Generally, approaches to the problem can be classified into two: statistics-based and POI (Point-Of-Interest)-based approach. As a statistics-based approach, Liao et al. [14] proposed a relational Markov network (RMN) which is an extension of conditional random fields (CRF) for inferring daily activities from GPS trajectories. However, statistics-based approaches require a large set of data enough to build a stochastic model. Unlike the statistics-based approach, POI-based approaches use the predefined POIs to identify activities. Xie et al. [17] proposed semantic join that identifies a sequence of activities by associating the closest POIs to a trajectory.

Continuous Query Indexing. The query indexing technique is essential for scalable continuous query processing [15][10][11]. Prabhakar et al. [15] proposed a query indexing technique for scalable evaluation of multiple continuous static range queries on moving objects. As opposed to traditional indexing of moving objects, such query indexing approach reduces the cost of maintaining the index by minimizing the number of index updates, as queries are not added/removed as frequently as object move. Lee et al. [10][11] proposed a BMQ-index for efficiently processing border-crossing monitoring queries. The BMQ-index consists of two linked lists for each dimension, and each list is a set of projected borders of monitoring regions to each dimension. By aggregating search results for each dimension, it can efficiently detect border crossing events of moving objects.

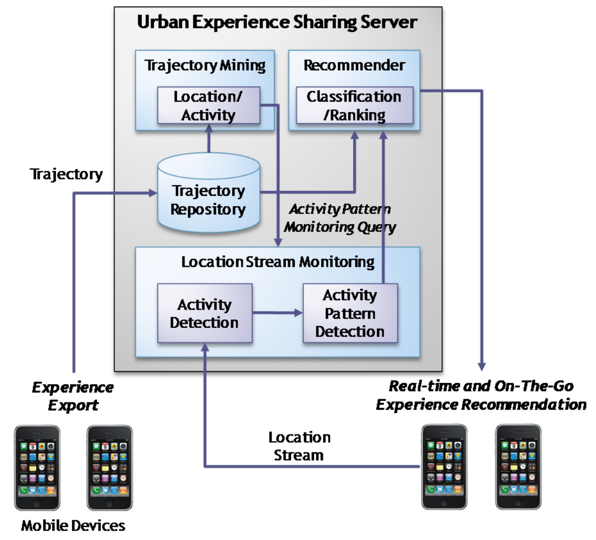

Urban Experience Sharing Server
Trajectory Mining
Location/ Activity
Recommender
Classification /Ranking
Trajectory
Trajectory Repository
Activity Pattern Monitoring Query
Location Stream Monitoring
Activity Detection
Activity Pattern Detection
Experience Export
Real-time and On-The-Go Experience Recommendation
Location Stream
Mobile Devices

Figure 1. MetroScope system architecture.

Complex Event Processing. Complex event processing (CEP) systems [7][16][12] have been proposed for monitoring complex event patterns in diverse application domains such as RFID management, sensor networks, environment monitoring, and stock monitoring. The CEP systems provide event operators such as sequence, conjunction (AND), and disjunction (OR) that detect composite events by connecting primitive events. Wu et al. [16] proposed a NFA (Non-deterministic Finite Automata)-based algorithms named SASE for efficiently matching an event sequence over a large number of event sources. Unlike NFA-based approaches, recently, Lee et al. [12] proposed a network-based algorithm, i.e., Event-centric Composition Queue (ECQ). Using a shared processing technique that shares common event queues, the ECQ can efficiently process a large number of event monitoring queries.

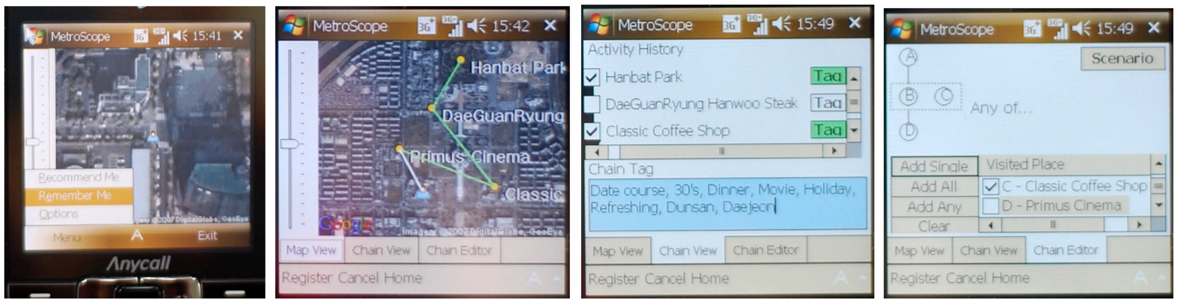
3. SYSTEM OVERVIEW

3.1 Key Concepts

MetroScope is a *trajectory-based experience sharing system*. Unlike the existing location sharing applications such as Foursquare [3], it aims to allow people to share their daily life experiences through trajectories, and take reference information on other people's diverse and interesting experiences in a city. Trajectory-based experience sharing is composed of two main operations: *experience export* and *experience recommendation*. There would be diverse requirements for such a trajectory-based experience sharing system. Among them, we emphasize that *real-time*, *on-the-go*, and *personalized* experience sharing is important.

Experience Export. Experience sharing begins with experience export. Experience export is a set of procedures that make a recorded personal trajectory public. MetroScope allows people to upload their trajectories recorded by mobile devices to a central sharing server. However, uploading just raw GPS trajectories are insufficient. For intelligent experience sharing, semantic information such as significant locations and activities should be automatically extracted from the uploaded trajectories. Such extracted semantic information is used for finding people who will be highly likely interested in the trajectory.

Experience Recommendation. To help people find other's relevant and interesting other people's experiences, MetroScope supports experience recommendation. We argue that *real-time*,



Activity History

✓	Hanbat Park	Tag
	DaeGuanRyung Hanwoo Steak	Tag
✓	Classic Coffee Shop	Tag

Chain Tag

Date course, 30's, Dinner, Movie, Holiday, Refreshing, Dunsan, Daejeon

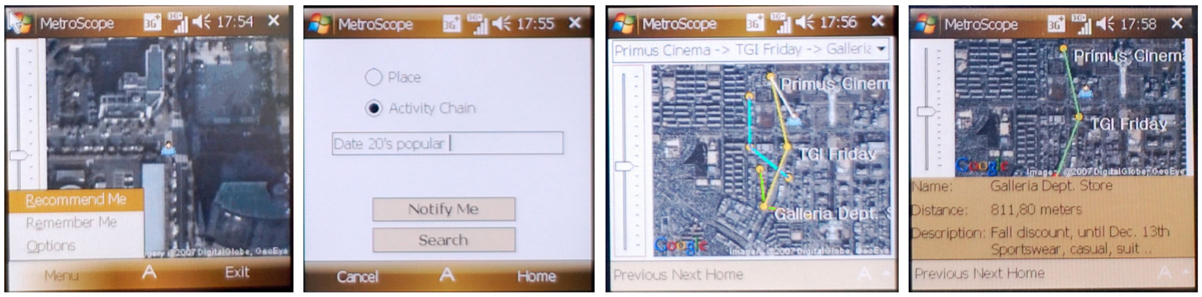
Visited Place

✓ C - Classic Coffee Shop

D - Primus Cinema

(a) Home view (b) Location history (c) Experience description (d) Recommendation rule

Figure 2. Screenshots of MetroScope's experience export.



Date 20's popular

Name: Galleria Dept. Store

Distance: 811,80 meters

Description: Fall discount, until Dec. 13th Sportswear, casual, suit ..

(a) Home view (b) Service mode (c) Recommended experiences (d) Experience details

Figure 3. Screenshots of MetroScope's experience recommendation.

on-the-go, and *personalized* recommendation is important in upcoming mobile computing era. Unlike PC era, in many cases, people using mobile devices want to take reference information on the surrounding area in a real-time manner, while they are on-the-go. For example, young dating couples often want to know some nice and new places to have dinner or to have fun, while hanging around a downtown. Also, many people often prefer to take personalized information such as gourmets' Thai restaurant recommendation than general information such as the most popular restaurant in a downtown. To achieve real-time, on-the-go, and personalized recommendation, we propose to monitor activity patterns over location streams reported from people who want to get recommended.

3.2 System Architecture

The system architecture of MetroScope is shown in Figure 1. As shown in the figure, MetroScope consists of a number of Mobile Clients and a Sharing Server. In the following subsections, we further provide the design and implementation of MetroScope.

3.2.1 MetroScope Mobile Client

The MetroScope Mobile Client is a location-aware mobile device such as smartphones, and plays as an agent of a user for the experience export and recommendation. For the experience export, the MetroScop Mobile Client is responsible for recording a user's trajectories and uploading them to a MetroScope Sharing Server when the user request. For the experience recommendation, the MetroScope Mobile Client is responsible for sending a user's location stream to the MetroScope Sharing Server and receiving relevant and interesting experiences for the user. To clearly deliver how people export their experience and get recommended, we provide prototype implementation as follows. For the prototype implementation, we use a smart phone, SAMSUNG blackjack 2 (SCH-M480) running Windows Mobile 6. It is equipped with a Marvell PXA310 (624MHz) processor, 3G/WiFi network interfaces, and a GPS receiver. The prototype is implemented in C# and runs on Microsoft .Net Compact Framework 2.0.

Figure 2 illustrates how the MetroScope Mobile Client helps a user to export her experience. First, the Mobile Client shows the current location of a user on a map in the home view (Figure 2-(a)). In the home view, a user starts to export her experience by selecting "Remember Me" in the menu (Figure 2-(a)). Then, the user's location history is automatically extracted and visualized on a map (Figure 2-(b)). Meanwhile, in the "Chain View" tab, the user can edit her experience by selecting which locations to export and making a note of additional information on each location. Also, the user can make a note that describes the overall experience to export (Figure 2-(c)).

Figure 3 describes how the MetroScope Mobile Client helps a user to get recommended with other people's experiences. A user can start experience recommendation by selecting "Recommend Me" in the menu of the home view (Figure 3-(a)). Then, the user can select one among two recommendation modes, i.e., "Notify Me" for proactive recommendation continuously updated by the server and "Search" for one time recommendation (Figure 3-(b)). In case of "Search", the user can input some keywords. In the "Notify Me" mode, the MetroScope Mobile Client continuously sends location data to the MetroScope Sharing Server. Then, the server can observe activity patterns of the user and continuously provide appropriate recommendations based on the patterns. Recommended experiences provided by the server are visualized on a map along with the city dynamics information (Figure 3-(c)). The sequence of circles presents a recommended experience, and each circle points a place and an activity. The size of a circle represents the crowdedness of a place. The bigger a circle is, the more crowded a place is. Also, the popularity of the experience is presented by the thickness and color of lines. If a user selects one of the experiences in the drop-down list, the details of the experience are shown on a map along with corresponding description (Figure 3-(d)).

3.2.2 MetroScope Sharing Server

The MetroScope Sharing Server plays as an orchestrator that enables the real-time and on-the-go experience sharing among a large number of people. As shown in Figure 1, the MetroScope Sharing Server is responsible for (1) mining the metadata such as significant locations and activities from the trajectories uploaded from users, (2) monitoring activity patterns over many people's location streams, and (3) recommending interesting experiences to relevant people according to their activity patterns detected by the monitoring unit. Among these functions, monitoring activity patterns over people's location streams is the key technique.

The MetroScope Sharing Server recommends relevant and interesting experiences as follows. First, the Sharing Server extracts semantic information such as significant locations and activities from trajectories uploaded from the Mobile Clients. The extracted information is maintained in a XML file named *Trajectory-based Experience Description Language (TEDL)*. The TEDL effectively describes a series of activities extracted from a trajectory. Here, an activity includes location coordinates, place name, place category, star time, and end time. For example, Bob's dating experience can be represented as follows (for convenience, we provide the example in a natural language form, rather than in a XML form): A1) drinking a cup of coffee at Starbucks from 2:00 p.m. to 2:30 p.m., A2) watching a movie at a multiplex from 3:00 p.m. to 5:00 p.m., A3) having dinner at T.G.I Friday from 6:00 p.m. to 8:00 p.m. The extracted semantic information is used for generating a monitoring query in the next step. Second, the Sharing Server associates the exported experiences with an activity monitoring query. And then, it starts to monitor activity patterns over people's location streams to find people who may be relevant and interested in the associated experience. If the Sharing Server finds people who match the associated monitoring query, it sends the experience to those people.

Query 1:

```
RECOMMEND Experience.title = "Bob's Dating" TO People
FROM Experience, People, PeopleProfile
PATTERN People == SEQ(ACT("Starbucks", 30min, 60min),
                      ACT("Cinema", 60min, 180min))
WHERE People.id = PeopleProfile.id
    AND PeopleProfile.age >= 20
    AND PeopleProfile.age < 30
```

Data schema:

- Experience<id, title, path(trajectory), path(TEDL), path(AQL)>
- People<id, location_x, location_y, timestamp>
- PeopleProfile<id, age, gender>

Activity Pattern Monitoring. For activity pattern monitoring, we propose a SQL-like query language, i.e., *Activity Pattern Monitoring Query Language (AQL)* [13]. Query 1 is an example monitoring query. It can be used to recommend the Bob's dating experience to people who have been drinking a cup of coffee at Starbucks and then watching movie at a multiplex, and is in their 20s. As shown in Query 1, the PATTERN clause specifies an activity pattern to monitor.

MetroScope first detects primitive activities from continuously incoming location streams, and then detects activity patterns by composing the primitive activities. In the AQL, we develop a new operator, **ACT**. The **ACT** operator is specified as **ACT**(*place_id*, *min_time*, *max_time*), where *place_id* is the identifier of a monitoring place, *min_time* and *max_time* are the minimum staying time and maximum staying time at the place specified by *place_id*, respectively. The operator detects whether a person stays for a specified amount of time (i.e., more than the *min_time* and less than the *max_time*) at a given place, while monitoring the person's location stream. The basic idea of using **ACT** operator is that people's daily activities can be effectively detected by location, time, and duration. The details can be found in our previous work [13]. It is similar to the POI-based activity mining from trajectories discussed in Section 2, since it exploits predefined POIs to detect activities. However, our approach is significantly different from those, since we apply the basic idea to detect activities over continuously incoming location streams. We adopt *event operators* (i.e., sequence (SEQ), conjunction (AND), disjunction (OR), etc) from the existing complex event processing (CEP) systems (i.e., Sentinel (a.k.a. Snoop) [7] and SASE [16]). Event operators connect primitive or composite events together to form new composite events. We developed *scalable activity pattern monitoring techniques* based on our previous work [8][9][10][11][12][13].

4. CONCLUSION AND FUTURE WORK

In this paper, we proposed MetroScope, a *trajectory-based real-time and on-the-go experience sharing system* in a metropolitan city. We plan to further study other important issues of experience sharing systems. The issues may include activity detection accuracy, activity monitoring scalability, recommendation performance, location privacy, etc.

5. REFERENCES

[1] EveryTrail. http://www.everytrail.com.

[2] Facebook. http://www.facebook.com.

[3] Foursquare. http://foursquare.com.

[4] ShareMyRoutes. http://www.sharemyroutes.com.

[5] Twitter. http://www.twitter.com.

[6] D. Ashbrook, and T. Starner. Using GPS to learn significant locations and predict movement across multiple users. *Personal and Ubiquitous Computing, vol. 7, no. 5*, 2003.

[7] S. Chakravarthy, V. Krishnaprasad, E. Anwar, and S.-K. Kim. Composite events for active databases: Semantics, Contexts and Detection. In *Proc. of VLDB*, 1994.

[8] K. Cho, I. Hwang, S. Kang, B. Kim, J. Lee, S. Lee, S. Park, Y. Rhee, and J. Song. HiCon: a hierarchical context monitoring and composition framework for next generation context-aware services. *IEEE Network, Vol. 22, No. 4*, 2008.

[9] B. Kim, S. Lee, Y. Lee, I. Hwang, Y. Rhee, and J. Song. Mobiiscape: Middleware support for scalable mobility pattern monitoring of moving objects in a large-scale city. *Journal of Systems and Software, Volume 84, Issue 11*, 2011.

[10] J. Lee, Y. Lee, S. Kang, S. Lee, H. Jin, B. Kim, and J. Song. BMQ-Index: shared and incremental processing of border monitoring queries over data streams. In *Proc. of MDM*, 2006.

[11] J. Lee, S. Kang, Y. Lee, S. Lee, and J. Song. BMQ-Processor: a high-performance border crossing event detection framework for large-scale monitoring applications. *IEEE Transactions on Knowledge and Data Engineering (TKDE), Vol. 22, No. 2*, 2009.

[12] S. Lee, Y. Lee, B. Kim, K. S. Candan, Y. Rhee, and J. Song. High-performance composite event monitoring system supporting large number of queries and sources. In *Proc. of ACM DEBS*, 2011.

[13] Y. Lee, S. Lee, B. Kim, J. Kim, Y. Rhee, and J. Song. Scalable activity-travel pattern monitoring framework for large-scale city environment. *To appear in IEEE Transactions on Mobile Computing (TMC)*.

[14] L. Liao, D. Fox, and H. Kautz. Location-based activity recognition using relational Markov networks. In *Proc. of IJCAI*, 2005.

[15] S. Prabhakar, Y. Xia, D. V. Kalashnikov, W. G. Aref, S. E. Hambrusch. Query indexing and velocity constrained indexing: scalable techniques for continuous queries on moving objects. *IEEE Transactions on Computer, vol. 51, no. 10*, 2002.

[16] E. Wu, Y. Diao, and S. Rizvi. High-performance complex event processing over streams. In Proc. of SIGMOD, 2006.

[17] K. Xie, K. Deng, and X. Zhou. From trajectories to activities: a spatio-temporal join approach. In *Proc. of ACM LBSN*, 2009.

[18] Y. Zheng, L. Zhang, X. Xie, and W. Y. Ma. Mining interesting locations and travel sequences from GPS trajectories. In *Proc. of WWW*, 2009.

[19] Y. Zheng, X. Xie, and W. Ma. GeoLife: A collaborative social networking service among user, location and trajectory. *IEEE Data Eng. Bull., 33(2):32-40*, 2010.

Geo-social Recommendations based on Incremental Tensor Reduction and Local Path Traversal

Panagiotis Symeonidis
Aristotle University
Thessaloniki, Greece
symeon@csd.auth.gr

Alexis Papadimitriou
Aristotle University
Thessaloniki, Greece
apapadi@csd.auth.gr

Yannis Manolopoulos
Aristotle University
Thessaloniki, Greece
manolopo@csd.auth.gr

Pinar Senkul
Middle East Technical University
Ankara, Turkey
senkul@ceng.metu.edu.tr

Ismail Toroslu
Middle East Technical University
Ankara, Turkey
toroslu@ceng.metu.edu.tr

ABSTRACT

Social networks have evolved with the combination of geographical data, into Geo-social networks (GSNs). GSNs give users the opportunity, not only to communicate with each other, but also to share images, videos, locations, and activities. The latest developments in GSNs incorporate the usage of location tracking services, such as GPS to allow users to "check in" at various locations and record their experience. In particular, users submit ratings or personal comments for their location/activity. The vast amount of data that is being generated by users with GPS devices, such as mobile phones, needs efficient methods for its effective management. In this paper, we have implemented an online prototype system, called Geo-social recommender system, where users can get recommendations on friends, locations and activities. For the friend recommendation task, we apply the FriendLink algorithm, which performs a local path traversal on the friendship network. In order to provide location/activity recommendations, we represent data by a 3-order tensor, on which latent semantic analysis and dimensionality reduction is performed using the Higher Order Singular Value Decomposition (HOSVD) technique. As more data is accumulated to the system, we use incremental solutions to update our tensor. We perform an experimental evaluation of our method with two real data sets and measure its effectiveness through recall/precision.

Keywords

tensor, geographical, social, geo-social, recommendations [1]

[1]This work has been partially funded by the Greek GSRT (project number 10TUR/4-3-3) and the Turkish TUBITAK (project number 109E282) national agencies as part of Greek-Turkey 2011-2012 bilateral scientific cooperation.

1. INTRODUCTION

Over the past few years, social networks have attracted a huge attention after the widespread adoption of Web 2.0 technology. Social networks combined with geographical data, have evolved into Geo-social networks (GSNs). GSNs such as Facebook Places, Google Places, Foursquare.com, etc., which allow users with mobile phones to contribute valuable information, have increased both in popularity and size. These systems are considered to be the next big thing on the web [4]. An interesting statistic is that more than 250 million users are daily accessing Facebook through their mobile devices and they are twice as active than non-mobile users.

GSNs allow users to use their GPS-enabled device, to "check in" at various locations and record their experience. In particular, users submit ratings or personal comments for the location/activity they visited/performed. That is, they "check in" at various places, to publish their location online, and see where their friends are. Moreover, they can either comment on a friend's location or comment on their own. These GSN systems, based on a user's "check in" profile, can also provide activity and location recommendations. For an activity recommendation, if a user plans to visit some place, the GSN system can recommend an activity (i.e. dance, eat, etc.). For a location recommendation, if a user wants to do something, the GSN system can recommend a place to go. Recently, Zheng et al. [12] proposed a User Collaborative Location and Activity Filtering (UCLAF) system, which is based on Tensor decomposition. However, as the authors claim, they do not update their system online as more users accumulate data continuously over time. Moreover, even though their system provides location and activity recommendations to users, it does not consider the case of providing also friend recommendations.

Our prototype system GeoSocial is an online recommender system that relies on user check-ins to provide friend, location and activity recommendations. Every registered user is presented with the option of checking in. The procedure involves selecting the location he is currently at, the activity he is performing there, and finally rating that activity. Based on the users' "check in" history and friendship network, GeoSocial provides friend, location and activity recommendations. Friends are recommended based on

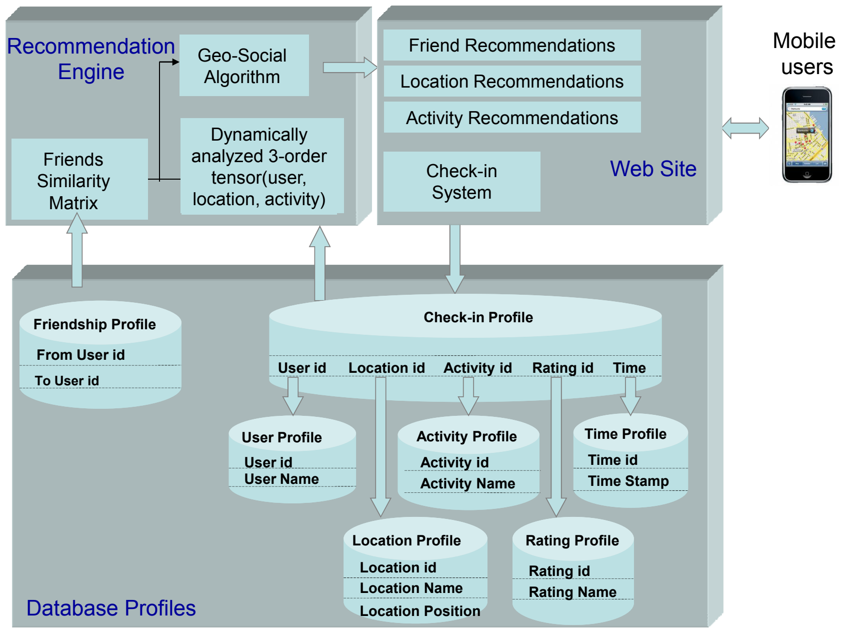


Figure 1: Components of the Geo-social recommender system.

the FriendLink algorithm [6] and the average geographical distances between users' "check-ins", which are used as link weights. Users, locations and activities are also inserted into a 3-order tensor, which is then used to provide location and activity recommendations.

The remainder of this paper is organized as follows. Section 2 summarizes the related work, whereas Section 3 describes the GeoSocial recommender system and its components. Section 4 describes the FriendLink algorithm, which performs a local path traversal on the friendship network to provide friend recommendations. Section 5 explains the main steps that are followed when performing the tensor reduction to detect latent associations between the user, location and activity dimensions and also the way we update the tensor data by implementing the Incremental Tensor Reduction (ITR) algorithm. In Section 6 we study the performances of ITR and FriendLink in terms of friend, location and activity recommendations. Finally, Section 7 concludes the paper and proposes possible future work.

2. RELATED WORK

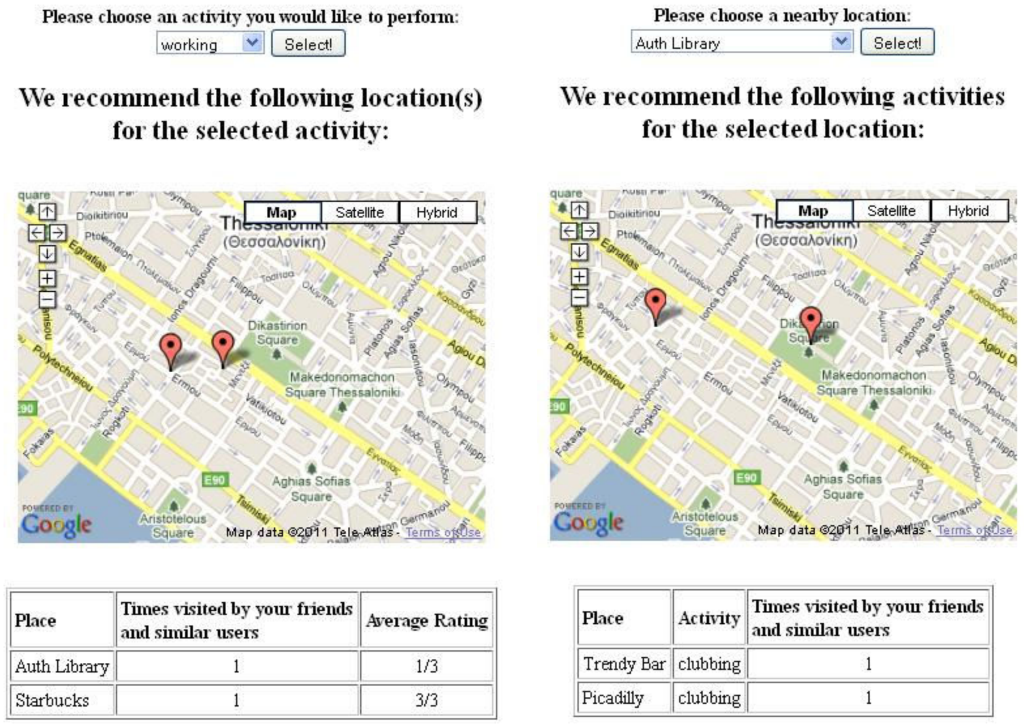
Recently emerged GSNs (i.e. Gowalla.com, Foursquare.com, Facebook Places etc.) provide to users activity or location recommendation. For example, in Gowalla.com a target user can provide to the system the activity he wants to do and the place he is (e.g. coffee in New York). Then, the system provides a map with coffee places which are nearby the user's location and were visited many times from people he knows. Moreover, Facebook Places allows users to see where their friends are and share their location in the real world.

There is a little research on the scientific field of GSNs. Backstrom et al. [1] use user-supplied address data and the network of associations between members of the Facebook social network to measure the relationship between geography and friendship. Using these measurements, they can predict the location of an individual. Scellato et al. [10] proposed a graph analysis based approach to study social networks with geographic information. They also applied new geo-social metrics to four large-scale Online Social Network data sets (i.e. Liveljournal, Twitter, FourSquare, BrightKite). Quercia et al. [7] address the mobile cold-start problem when recommending social events to users without any location history.

Leung et al. [5] propose the Collaborative Location Recommendation (CLR) framework for location recommendation. The framework considers activities and different user classes to generate more precise and refined recommendations. The authors also incorporate a dynamic clustering algorithm, namely Community-based Agglomerative-Divisive Clustering (CADC), into the framework in order to cluster the trajectory data into groups of similar users, similar activities and similar locations. The algorithm can also be updated incrementally when new GPS trajectory data is available.

Ye et al. [11] believe that user preferences, social influence and geographical influence should be considered when providing Point of Interest recommendations. They study the geographical clustering phenomenon and propose a power-law probabilistic model to capture the geographical influence among Points of Interest. Finally, the authors evaluate their proposed method over the Foursquare and Whrrl datasets and discover, among others, that geographical influence is more important than social influence and that item similarity is not as accurate as user similarity due to a lack of user check-ins.

Moreover, there are tensor-based approaches. For example, Biancalana et al. [2] implemented a social recommender system based on a tensor that is able to identify user preferences and information needs and suggests personalized recommendations for possible points of interest (POI). Furthermore, Zheng et al. [13] proposed a method, where geographical data is combined with social data to provide location



Please choose an activity you would like to perform:
working

We recommend the following location(s) for the selected activity:

Place	Times visited by your friends and similar users	Average Rating
Auth Library	1	1/3
Starbucks	1	3/3

(a)

Please choose a nearby location:
Auth Library

We recommend the following activities for the selected location:

Place	Activity	Times visited by your friends and similar users
Trendy Bar	clubbing	1
Picadilly	clubbing	1

(b)

Figure 2: Location and activity recommendations made by the Geo-social recommender system.

and activity recommendations. The authors used GPS location data, user ratings and user activities to propose location and activity recommendations to interested users and explain them accordingly. Moreover, Zheng et al. [12] proposed a User Collaborative Location and Activity Filtering (UCLAF) system, which is based on Tensor decomposition.

In contrast to the aforementioned tensor-based methods, our Geosocial recommender system provides (i) location and activity recommendations (ii) friend recommendations by combining FriendLink algorithm [6] with the geographical distance between users. Moreover, our tensor method includes an incremental stage, where newly created data is inserted into the tensor by incremental solutions. [9, 3].

3. GEOSOCIAL SYSTEM DESCRIPTION

Our GeoSocial system consists of several components. The system's architecture is illustrated in Figure 1, where three main sub-systems are described: (i) the Web Site, (ii) the Database Profiles and (iii) the Recommendation Engine. In the following sections, we describe each sub-system of GeoSocial in detail.

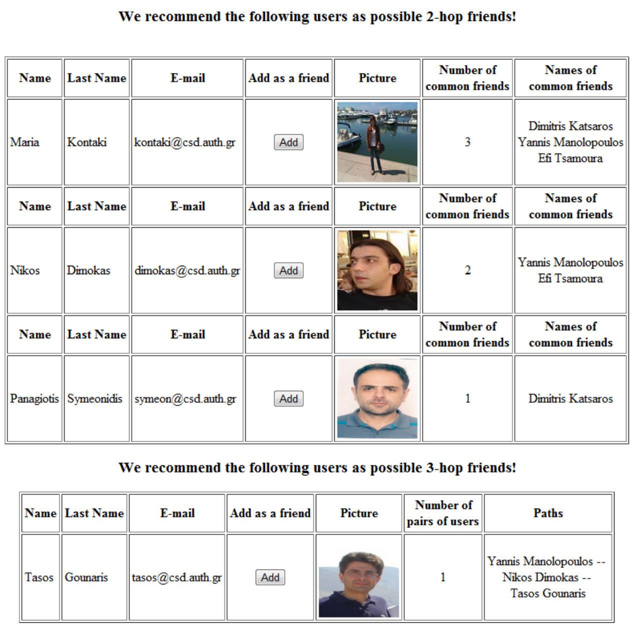
3.1 GeoSocial Web Site

The GeoSocial system uses a web site [2] to interact with the users. The web site consists of four sub-systems: (i) the friend recommendation, (ii) the location recommendation, (iii) the activity recommendation and (iv) the check-in system. The friend recommendation sub-system is responsible for evaluating incoming data from the Recommendation Engine of GeoSocial and providing updated friend recommendations. In order to provide such recommendations, the web site sub-system implements the FriendLink algorithm [6] and also considers the geographical distance between users and "check in" points. The same applies to the location and activity recommendation sub-systems where new and updated location and activity recommendations are presented to the user as new "check-ins" are stored in the Database profiles. Finally, the check in system is responsible for passing the data inserted by the users to the respective Database profiles.

Figure 2a shows a location recommendation while Figure 2b depicts an activity recommendation. As shown in Figure 2a, the user selects an activity that he would like to perform, in this case working, and the system provides location recommendations where he could perform his selected activity, in this case either Starbucks or the Auth Library. As shown in Figure 2b, the user selects a nearby location, i.e. Auth Library and the system provides activities that he could perform. In this case the user's location is near the Auth Library and the system proposes clubbing at the "Trendy bar" or the "Picadily" as possible activities.

Figure 3a presents a scenario where the GeoSocial system recommends 4 possible friends to a target user. As shown, the first table includes 3 users, namely Maria Kontaki, Nikos Dimokas and Panagiotis Symeonidis, who are connected to the target user via 2-hop paths. The results are ordered based on the second to last column of the table, which indicates the number of common friends that the target user shares with each possible friend. As shown in Figure 3a, Maria Kontaki is the top recommendation because she shares 3 common friends with the target user. The common friends are then presented in the last column

[2]http://delab.csd.auth.gr/geosocial



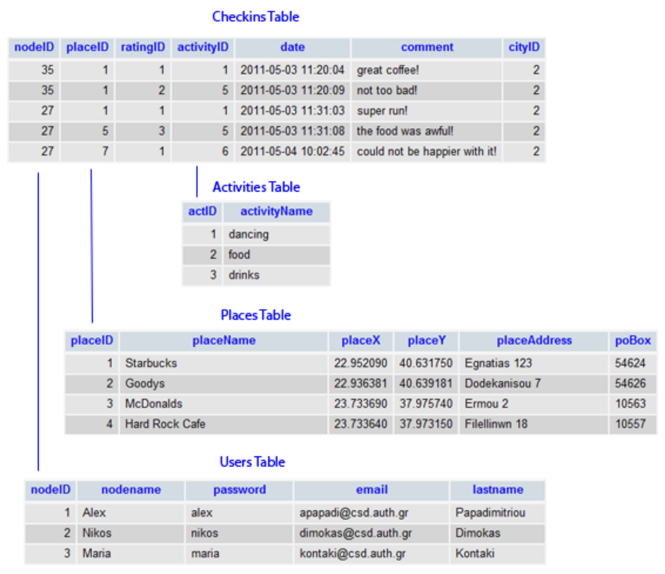
We recommend the following users as possible 2-hop friends!

Name	Last Name	E-mail	Add as a friend	Picture	Number of common friends	Names of common friends
Maria	Kontaki	kontaki@csd.auth.gr	Add		3	Dimitris Katsaros Yannis Manolopoulos Efi Tsamoura
Name	**Last Name**	**E-mail**	**Add as a friend**	**Picture**	**Number of common friends**	**Names of common friends**
Nikos	Dimokas	dimokas@csd.auth.gr	Add		2	Yannis Manolopoulos Efi Tsamoura
Name	**Last Name**	**E-mail**	**Add as a friend**	**Picture**	**Number of common friends**	**Names of common friends**
Panagiotis	Symeonidis	symeon@csd.auth.gr	Add		1	Dimitris Katsaros

We recommend the following users as possible 3-hop friends!

Name	Last Name	E-mail	Add as a friend	Picture	Number of pairs of users	Paths
Tasos	Gounaris	tasos@csd.auth.gr	Add		1	Yannis Manolopoulos -- Nikos Dimokas -- Tasos Gounaris

(a)



Checkins Table

nodeID	placeID	ratingID	activityID	date	comment	cityID
35	1	1	1	2011-05-03 11:20:04	great coffee!	2
35	1	2	5	2011-05-03 11:20:09	not too bad!	2
27	1	1	1	2011-05-03 11:31:03	super run!	2
27	5	3	5	2011-05-03 11:31:08	the food was awful!	2
27	7	1	6	2011-05-04 10:02:45	could not be happier with it!	2

Activities Table

actID	activityName
1	dancing
2	food
3	drinks

Places Table

placeID	placeName	placeX	placeY	placeAddress	poBox
1	Starbucks	22.952090	40.631750	Egnatias 123	54624
2	Goodys	22.936381	40.639181	Dodekanisou 7	54626
3	McDonalds	23.733690	37.975740	Ermou 2	10563
4	Hard Rock Cafe	23.733640	37.973150	Filellinwn 18	10557

Users Table

nodeID	nodename	password	email	lastname
1	Alex	alex	apapadi@csd.auth.gr	Papadimitriou
2	Nikos	nikos	dimokas@csd.auth.gr	Dimokas
3	Maria	maria	kontaki@csd.auth.gr	Kontaki

(b)

Figure 3: (a) Friend recommendations provided by the GeoSocial system. (b) Database check-in profile.

of the table. The second table contains one user, namely Tasos Gounaris, who is connected to the target user via a 3-hop path. The last column of the table indicates the number of pairs that connect the target user with the possible friend. As shown in Figure 3a, Tasos Gounaris is connected to the target user via 1 3-hop path. This path is presented in the last column of the table.

3.2 GeoSocial Database Profiles

The database that supports the GeoSocial system is a MySQL(v.5.5.8) [3] database. MySQL is an established Database Management System (DBMS), which is widely used in online, dynamic, database driven websites.

The database profile sub-system contains five profiles where data about the users, locations, activities and their corresponding ratings is stored. As shown in Figure 3b, this data is received by the Check-In profile and along with the Friendship profile, they provide the input for the Recommendation Engine sub-system.

The collected data is stored in a database table, called "checkins", which is shown in Figure 3b. Each table field represents the respective data that is collected by the Check-In profile. NodeID, placeID and activityID refer to specific IDs given to users, locations and activities respectively. We also store information about the time of the check-in inside the date field and an optional user comment inside the comment field.

3.3 GeoSocial Recommendation Engine

The recommendation engine is responsible for collecting the data from the database and producing the recommendations which will then be displayed on the web site. As shown in Figure 1, the recommendation engine constructs a friends similarity matrix by implementing the FriendLink algorithm proposed in [6]. The average geographical distances between users' "check-ins" are used as link weights. To obtain the weights, we calculate the average distance between all pairs of POIs that two users have checked-in. The recommendation engine also produces a dynamically analyzed 3-order tensor, which is firstly constructed by the HOSVD algorithm and is then updated using incremental methods [9, 3], both of which are explained in later sections.

[3]http://www.mysql.com

4. THE FRIENDLINK ALGORITHM

In this section, we describe our FriendLink [6] algorithm, which can be used for the task of friend recommendations. Here, we describe how FriendLink is applied on GSNs and how the recommendation of friends is performed according to the detected associations.

When using an GSN, users explicitly declare their friends so that they are able to share information location with them (i.e. photos etc.) After some time, the geo-social network accumulates a set of connection data (graph of friendships), which can be represented by an undirected graph.

Our method assumes that persons in an GSN can use all the pathways connecting them, proportionally to the pathway lengths. Thus, two persons who are connected with many unique pathways have a high possibility to know each, proportionally to the length of the pathways they are connected with.

Definition 1. The similarity $sim(v_x, v_y)$ between two graph nodes v_x and v_y is defined as the counts of paths of varying length ℓ from v_x to v_y:

$$sim(v_x, v_y) = \sum_{i=2}^{\ell} \frac{1}{i-1} \cdot \frac{|paths^{i}_{v_x,v_y}|}{\prod_{j=2}^{i}(n-j)} \quad (1)$$

where

- n is the number of vertices in a graph G,
- ℓ is the maximum length of a path between the graph nodes v_x and v_y (excluding paths with cycles). By the term "paths with cycles" we mean that a path can not be closed (cyclic). Thus, a node can exist only one time in a path (e.g. path $v_1 \rightarrow v_2 \rightarrow v_3 \rightarrow v_1 \rightarrow v_5$ is not acceptable because v_1 is traversed twice),
- $\frac{1}{i-1}$ is an "attenuation" factor that weights paths according to their length ℓ. Thus, a 2-step path measures the non-attenuation of a link with value equals to 1 ($\frac{1}{2-1} = 1$). A 3-step path measures the attenuation of a link with value equals to $\frac{1}{2}$ ($\frac{1}{3-1} = \frac{1}{2}$) etc. In this sense, we use appropriate weights to allow the lower effectiveness of longer path chains.
- $\left|paths^{\ell}_{v_x,v_y}\right|$ is the count of all length-ℓ paths from v_x to v_y,
- $\prod_{j=2}^{i}(n-j)$ is the count of all possible length-ℓ paths from v_x to v_y, if each vertex in graph G was linked with all other vertices. By using the fraction $\frac{\left|paths^{\ell}_{v_x,v_y}\right|}{\prod_{j=2}^{i}(n-j)}$, our similarity measure is normalized and takes values in [0,1]. If two nodes are similar we expect the value $sim(v_x, v_y)$ to be close to 1. On the other hand, if the two nodes are dissimilar, we expect the value $sim(v_i, v_j)$ to be close to 0.

Our FriendLink approach finds similarities between nodes in an undirected graph constructed from these connection data. The FriendLink algorithm uses as input the connections of a graph $\mathcal{G}$ and outputs a similarity matrix between any two nodes in $\mathcal{G}$. Therefore, friends can be recommended to a target user u according to their weights in the similarity matrix.

Friendlink computes node similarity between any two nodes in a graph $\mathcal{G}$. The initial input of Friendlink is the number n of nodes of $\mathcal{G}$, the adjacency matrix A, and the length ℓ of paths that will be explored in $\mathcal{G}$. To enumerate all simple paths in $\mathcal{G}$, Rubin's algorithm [8] can be employed. However, Rubin's algorithm uses O(n^3) matrix operations to find all paths of different length between any pair of nodes, where n is the number of nodes in $\mathcal{G}$. In the following, we customize Rubin's algorithm to create only paths of length up to ℓ for our purpose.

As shown in Figure 4, FriendLink consists of a main program and two functions. In the main program, we modify the adjacency matrix so instead of holding 0/1 values, the (i, j) entry of the matrix A is a list of paths from i to j. Then, in the function Combine Paths(), we perform the matrix multiplication algorithm. However, instead of multiplying and adding entries, we concatenate pairs of paths together. Finally, in the function Compute Similarity(), we update the similarity between nodes i and j, for each length-ℓ path we find, where i is the start node and j is the destination node (i.e all paths of length [2..ℓ]). For the update of the similarity value between nodes i and j we use Equation 1. Notice that, we do not take into account cyclic paths in our similarity measure.

```
Algorithm FriendLink ($\mathcal{G}$, $A$, $n$, $\ell$ )
Input
    $\mathcal{G}$: an undirected graph
    $A$: adjacency matrix of graph $\mathcal{G}$,
    $n$: number of nodes of graph $\mathcal{G}$,
    $\ell$: maximum length of paths explored in $\mathcal{G}$,
    $m$: the length of a path
Output
    $sim(i, j)$: similarity between node $i$ and node $j$ in $\mathcal{G}$

1.  Main Program
2.      for $i$ = 1 to $n$
3.          for $j$ = 1 to $n$
4.              if $A(i, j) = 1$ then
5.                  $A(i, j) = j$
6.              else
7.                  $A(i, j) = 0$
8.              end if
9.          end for $j$
10.     end for $i$
11. for $m$ = 2 to $\ell$
12.     Combine Paths()
13.     Compute Similarity($m$)
14. end for $m$
15. End Main Program

16. Function Combine Paths()
17. for $i$ = 1 to $n$
18.     for $j$ = 1 to $n$
19.         for $k$ = 1 to $n$
20.             if $A(i, k) <> 0$ and $A(k, j) <> 0$ then
21.                 $A(i, j) = concatenate(A(i, k), A(k, j))$
22.             end if
23.         end for $k$
24.     end for $j$
25. end for $i$
26. return $A(i, j)$
27. End Function

28. Function Compute Similarity($m$)
29. for $i$ = 1 to $n$
30.     for $j$ = 1 to $n$
31.         $denominator = 1$
32.             for $k$ = 2 to $m$
33.                 $denominator = denominator * (n - k)$
34.             end for $k$
35.     $sim(i, j) = sim(i, j) + \frac{1}{m-1} \cdot \frac{\left|paths^{m}_{i,j}\right|}{denominator}$
36.     end for $j$
37. end for $i$
38. return $sim(i, j)$
39. End Function
```

Figure 4: The FriendLink algorithm.

5. OUR INCREMENTAL TENSOR REDUCTION APPROACH

In this section we provide details on how HOSVD is applied on tensors and how location/activity recommendation is performed based on the detected latent associations.

Our Tensor Reduction approach initially constructs a tensor, based on usage data triplets $\{u, l, a\}$ of users, location and activity. The motivation is to use all three objects that interact inside a location-based social network. Consequently, we proceed to the unfolding of $\mathcal{A}$, where we build three new matrices. Then, we apply SVD in each new matrix. Finally, we build the core tensor $\mathcal{S}$ and the resulting tensor $\hat{\mathcal{A}}$. The 6 steps of the proposed approach are summarized as follows:

- *Step 1*: The initial tensor $\mathcal{A}$ construction, which is based on usage data triplets (user, location, activity).

- *Step 2*: The matrix unfoldings of tensor $\mathcal{A}$, where we matricize the tensor in all three modes, creating three new matrices (one for each mode).
- *Step 3*: The application of SVD in all three new matrices, where we keep the c-most important singular values for each matrix.
- *Step 4*: The construction of the core tensor $\mathcal{S}$, that reduces the dimensionality.
- *Step 5*: The construction of the $\hat{\mathcal{A}}$ tensor, that is an approximation of tensor $\mathcal{A}$.
- *Step 6*: Based on the weights of the elements of the reconstructed tensor $\hat{\mathcal{A}}$, we recommend location/activity to the target user u.

Steps $1-5$ build a model and can be performed off-line. The recommendation in Step 5 is performed on-line, i.e., each time we have to recommend a location/activity to a user, based on the built model. In the following, we provide more details on each step.

5.1 The initial construction of tensor $\mathcal{A}$

From the usage data triplets (user, location, activity), we construct an initial 3-order tensor $\mathcal{A} \in R^{I_u \times I_l \times I_a}$, where I_u, I_l, I_a are the numbers of users, locations and activities, respectively. Each tensor element measures the number of times that a user u checked in a location l and made an activity a.

5.2 Matrix unfolding of tensor $\mathcal{A}$

As described, a tensor $\mathcal{A}$ can be unfolded (matricized), i.e., we build matrix representations of tensor $\mathcal{A}$ in which all the column (row) vectors are stacked one after the other. The initial tensor $\mathcal{A}$ is matricized in all three modes. Thus, after the unfolding of tensor $\mathcal{A}$ for all three modes, we create 3 new matrices A_1, A_2, A_3, as follows:

$$A_1 \in R^{I_u \times I_l I_a},$$

$$A_2 \in R^{I_l \times I_u I_a},$$

$$A_3 \in R^{I_u I_l \times I_a}$$

5.3 Application of SVD on each matrix

We apply SVD on the three matrix unfoldings A_1, A_2, A_3. We result, in total, to 9 new matrices.

$$A_1 = U^{(1)} \cdot S_1 \cdot V_1^T \tag{2}$$

$$A_2 = U^{(2)} \cdot S_2 \cdot V_2^T \tag{3}$$

$$A_3 = U^{(3)} \cdot S_3 \cdot V_3^T \tag{4}$$

For Tensor Dimensionality Reduction, there are three dimensional parameters to be determined. The numbers c_1, c_2 and c_3 of left singular vectors of matrices $U^{(1)}$, $U^{(2)}$, $U^{(3)}$, respectively, that are preserved. They will determine the final dimensionality of the core tensor S. Since each of the three diagonal singular matrices S_1, S_2 and S_3 are calculated by applying SVD on matrices A_1, A_2 and A_3, respectively, we use different c_1, c_2 and c_3 numbers of principal components for each matrix $U^{(1)}$, $U^{(2)}$, $U^{(3)}$. The numbers c_1, c_2 and c_3 of singular vectors are chosen by preserving a percentage of information of the original S_1, S_2, S_3 matrices after appropriate tuning (the default percentage is set to 50% of the original matrix).

5.4 The construction of the core tensor $\mathcal{S}$

The core tensor $\mathcal{S}$ governs the interactions among users, locations and activities. Since we have selected the dimensions of $U^{(1)}$, $U^{(2)}$ and $U^{(3)}$ matrices, we proceed to the construction of $\mathcal{S}$, as follows:

$$\mathcal{S} = \mathcal{A} \times_1 {U_{c_1}}^{(1)^T} \times_2 {U_{c_2}}^{(2)^T} \times_3 {U_{c_3}}^{(3)^T}, \tag{5}$$

where $\mathcal{A}$ is the initial tensor, ${U_{c_1}}^{(1)^T}$ is the tranpose of the c_1-dimensionally reduced $U^{(1)}$ matrix, ${U_{c_2}}^{(2)^T}$ is the tranpose of the c_2-dimensionally reduced $U^{(2)}$ matrix, ${U_{c_3}}^{(3)^T}$ is the tranpose of the c_3-dimensionally reduced $U^{(3)}$ matrix.

5.5 The construction of tensor $\hat{\mathcal{A}}$

Finally, tensor $\hat{\mathcal{A}}$ is build as the product of the core tensor $\mathcal{S}$ and the mode products of the three matrices $U^{(1)}$, $U^{(2)}$ and $U^{(3)}$ as follows:

$$\hat{\mathcal{A}} = \mathcal{S} \times_1 {U_{c_1}}^{(1)} \times_2 {U_{c_2}}^{(2)} \times_3 {U_{c_3}}^{(3)}, \tag{6}$$

$\mathcal{S}$ is the reduced core tensor, ${U_{c_1}}^{(1)}$ is the c_1-dimensionally reduced $U^{(1)}$ matrix, ${U_{c_2}}^{(2)}$ is the c_2-dimensionally reduced $U^{(2)}$ matrix, ${U_{c_3}}^{(3)}$ is the c_3-dimensionally reduced $U^{(3)}$ matrix.

5.6 The generation of the location/activity Recommendation list

Tensor $\hat{\mathcal{A}}$ measures the associations among the users, locations and activities and acts as a model that is used during the recommendation.

Each element of $\hat{\mathcal{A}}$ represents a quadruplet $\{u, l, a, p\}$, where p is the likeliness that user u will visit location l and perform activity a. Therefore, locations/activities can be recommended to u according to their weights associated with $\{u,\ a\}$ and $\{u,\ l\}$ pairs, respectively. If we want to recommend to u N activities for location l, then we select the N corresponding activities with the highest weights.

5.7 Inserting new users, locations, or activities over time

As new users, locations, or activities are being introduced to the system, the $\hat{\mathcal{A}}$ tensor, which provides the recommendations, has to be updated. The most demanding operation for this task is the updating of the SVD of the corresponding unfoldings. We can avoid the costly batch recomputation of the corresponding SVD, by considering incremental solutions [9, 3]. Depending on the size of the update (i.e., number of new users, locations, or activities), different techniques have been followed in related research. For small update sizes we can consider the *folding-in* technique [9], whereas for larger update sizes we can consider Incremental SVD techniques [3].

5.7.1 Update by Incremental SVD

Folding-in incrementally updates SVD, but the resulting model is not a perfect SVD model, because the space is not orthogonal [9]. When the update size is not big, loss of orthogonality may not be a severe problem in practice. Nevertheless, for larger update sizes the loss of orthogonality may result to an inaccurate SVD model. In this case we need to incrementally update SVD so as to ensure orthogonality. This can be attained in several ways. Next we describe how to use the approach proposed by Brand [3].

Let $M_{p\times q}$ be a matrix, upon we which apply SVD and maintain the first r singular values, i.e.,

$$M_{p\times q} = U_{p\times r} S_{r\times r} V_{r\times q}^T$$

Assume that each column of matrix $C_{p\times c}$ contains the additional elements. Let $L = U\backslash C = U^T C$ be the projection of C onto the orthogonal basis of U. Let also $H = (I - UU^T)C = C - UL$ be the component of C orthogonal to the subspace spanned by U (I is the identity matrix). Finally, let J be an orthogonal basis of H and let $K = J\backslash H = J^T H$ be the projection of C onto the subspace orthogonal to U. Consider the following identity:

$$[U\ J] \begin{bmatrix} S & L \\ 0 & K \end{bmatrix} \begin{bmatrix} V & 0 \\ 0 & I \end{bmatrix}^T =$$

$$\Big[U (I - UU^T)C/K\Big] \begin{bmatrix} S & U^T C \\ 0 & K \end{bmatrix} \begin{bmatrix} V & 0 \\ 0 & I \end{bmatrix}^T =$$

$$\Big[USV^T\ C\Big] = [M\ C]$$

Like an SVD, the left and right matrixes in the product are unitary and orthogonal. The middle matrix, denoted as Q, is diagonal. To incrementally update the SVD, Q must be diagonalized. If we apply SVD on Q we get:

$$Q = U' S' (V')^T$$

Additionally, define U'', S'', V'' as follows:

$$U'' = [U\ J]U',\ S'' = S',\ V'' = \begin{bmatrix} V & 0 \\ 0 & I \end{bmatrix} V'$$

Then, the updated SVD of matrix $[M\ C]$ is:

$$[M\ C] = [USV^T\ C] = U'' S'' (V'')^T$$

This incremental update procedure takes $O((p+q)r^2 + pc^2)$ time [3].

Returning to the application of incremental update for new users, locations, or activities, in each case we result with a number of new rows that are appended in the end of the unfolded matrix of the corresponding mode. Therefore, we need an incremental SVD procedure in the case where we add new rows, whereas the aforementioned method works in the case where we add new columns. In this case we simply swap U for V and U'' for V''.

6. EXPERIMENTAL CONFIGURATION

In this section, we study the performance of our approach in terms of friend, location and activity recommendations. To evaluate the aforementioned recommendations we have chosen two real data sets. The first one, denoted as geosocial data set is extracted from our newly developed site. There are 1,173 triplets in the form user–location–activity. To these triplets correspond 102 users, 46 locations and 18 activities. The second data set, denoted as UCLAF [4] [12] data set consists of 164 users, 168 locations and 5 different types of activities, including “Food and Drink”, “Shopping”, “Movies and Shows”, “Sports and Exercise”, and “Tourism and Amusement”.

The numbers c_1, c_2, and c_3 of left singular vectors of matrices $U^{(1)}$, $U^{(2)}$, $U^{(3)}$ for our approach, after appropriate tuning, are set to 25, 12 and 8 for the geosocial dataset, and to 40, 35, 5 for the UCLAF data set. Due to lack of space we do not present experiments for the tuning of c_1, c_2, and c_3 parameters. The core tensor dimensions are fixed, based on the aforementioned c_1, c_2, and c_3 values.

6.1 Evaluation Metrics

We perform 4-fold cross validation and the default size of the training set is 75% – we pick, for each user, 75% of his check-ins and friends randomly. The task of all three recommendation types (i.e. friend, location, activity) is to predict the friends/locations/activities of the user’s 25% remaining check-ins and friends, respectively. As performance measures we use precision and recall, which are standard in such scenarios. For a test user that receives a list of N recommended friends/locations/activities (top-N list), the following are defined:

– ***Precision***: is the ratio of the number of relevant friends/locations/activities in the top-N list relative to N.

– ***Recall***: is the ratio of the number of relevant friends/locations/activities in the top-N list relative to the total number of relevant friends/locations/activities, respectively.

6.2 Comparison Results

In this section, we study the accuracy performance of our method in terms of precision and recall. This reveals the robustness of our method in attaining high recall with minimal losses in terms of precision. We examine the top-N ranked list, which is recommended to a test user, starting from the top friend/location/activity. In this situation, the recall and precision vary as we proceed with the examination of the top-N list. In Figure 5, we plot a precision versus recall curve. As it can be seen, our ITR approach presents high accuracy. The reason is that we exploit altogether the information that concerns the three entities (friends, locations, and activities) and thus, we are able to provide accurate location/activity recommendations. Notice that activity recommendations are more accurate than location recommendations. A possible explanation could be the fact that the number of locations is bigger than the number of activities. That is, it is easier to predict accurately an activity than a location. Notice that for the task of friend recommendation, the performance of Friendlink is not so high. The main reason is data sparsity. In particular, the friendship network has average nodes’ degree equal to 2.7 and average shortest

[4]http://www.cse.ust.hk/ vincentz/aaai10.uclaf.data.mat

distance between nodes 4.7, which means that the friendship network can not be consider as a "small world" network and friend recommendations can not be so accurate.

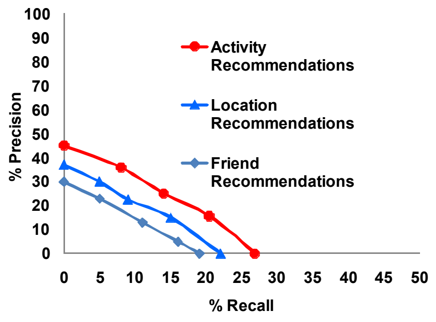


Figure 5: Precision Recall diagram of ITR and FriendLink for activity, location and friend recommendations on the Geosocial data set

For the UCLAF data set, as shown in Figure 6, our ITR algorithm attains analogous results. Notice that the recall for the activity recommendations, reaches 100% because the total number of activities is 5. Moreover, notice that in this diagram, we do not present results for the friend recommendation task, since there is no friendship network in the corresponding UCLAF data set.

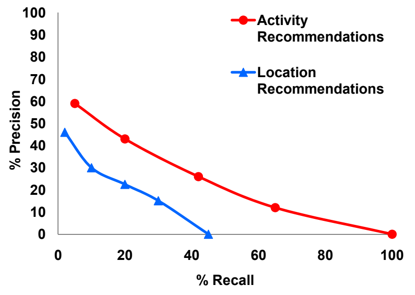


Figure 6: Precision Recall diagram of ITR for activity and location recommendations on the UCLAF data set

7. CONCLUSION AND FUTURE WORK

In this paper we have proposed a Geo-social recommender system, which is capable of recommending friends, locations and activities. For the location/activity recommendation task, we used a tensor, which is updated by incremental tensor approaches, as new users, locations, or activities are being inserted into the system. For the friend recommendation task, we apply the FriendLink algorithm, which performs a local path traversal on the friendship network.

As future work, we plan on conducting a user study concerning the recommendations in our Geo-social web site to measure user satisfaction. We are also planning on comparing our method to other state-of-the-art methods in terms of effectiveness and efficiency. Moreover, we want to extend our Geo-social recommender system by taking also into account the geographical influence and semantics. That is, the geographical proximities of locations and their semantics could play a determinant role on users' check-in behavior.

References

[1] L. Backstrom, E. Sun, and C. Marlow. Find me if you can: improving geographical prediction with social and spatial proximity. In *WWW '10: Proceedings of the 19th international conference on World Wide Web*, pages 61–70, New York, NY, USA, 2010. ACM.

[2] C. Biancalana, F. Gasparetti, A. Micarelli, and G. Sansonetti. Social tagging for personalized location-based services. 2011.

[3] M. Brand. Incremental singular value decomposition of uncertain data with missing values. In *European Conference on Computer Vision (ECCV2002)*, 2002.

[4] Economist. A world of connections: A special report on networking. *The Economist: Editorial Team*, 2010.

[5] K. W.-T. Leung, D. L. Lee, and W.-C. Lee. Clr: a collaborative location recommendation framework based on co-clustering. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information*, SIGIR '11, pages 305–314, New York, NY, USA, 2011. ACM.

[6] A. Papadimitriou, P. Symeonidis, and Y. Manolopoulos. Friendlink: Predicting links in social networks via bounded local path traversal. In *Proceedings of the 3rd Conference on Computational Aspects of Social Networks (CASON'2011)*, Salamanca, Spain (to appear), 2011.

[7] D. Quercia, N. Lathia, F. Calabrese, G. Di Lorenzo, and J. Crowcroft. Recommending Social Events from Mobile Phone Location Data. In *Proceedings of IEEE ICDM 2010*, Dec. 2010.

[8] F. Rubin. Enumerating all simple paths in a graph. *IEEE Transactions on Circuits and Systems*, 25(8):641–642, 1978.

[9] B. Sarwar, J. Konstan, and J. Riedl. Incremental singular value decomposition algorithms for highly scalable recommender systems. In *International Conference on Computer and Information Science*, 2002.

[10] S. Scellato, C. Mascolo, M. Musolesi, and V. Latora. Distance Matters: Geo-social Metrics for Online Social Networks. In *Proceedings of the 3rd Workshop on Online Social Networks (WOSN 2010)*, June 2010.

[11] M. Ye, P. Yin, W.-C. Lee, and D.-L. Lee. Exploiting geographical influence for collaborative point-of-interest recommendation. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information*, SIGIR '11, pages 325–334, New York, NY, USA, 2011. ACM.

[12] V. W. Zheng, B. Cao, Y. Zheng, X. Xie, and Q. Yang. Collaborative filtering meets mobile recommendation: A user-centered approach. *AAAI'10*, pages 236–241, 2010.

[13] W. Zheng, Y. Zheng, X. Xie, and Q. Yang. Collaborative location and activity recommendations with gps history data. In *WWW '10: Proceedings of the 19th international conference on World Wide Web*, pages 1029–1038, New York, NY, USA, 2010. ACM.