Methodologies for Cross-Domain Data Fusion: An Overview

Yu Zheng, *Senior Member*

Abstract— Traditional data mining usually deals with data from a single domain. In the big data era, we face a diversity of datasets from different sources in different domains. These datasets consist of multiple modalities, each of which has a different representation, distribution, scale, and density. How to unlock the power of knowledge from multiple disparate (but potentially connected) datasets is paramount in big data research, essentially distinguishing big data from traditional data mining tasks. This calls for advanced techniques that can fuse the knowledge from various datasets organically in a machine learning and data mining task. This paper summarizes the data fusion methodologies, classifying them into three categories: stage-based, feature level-based, and semantic meaning-based data fusion methods. The last category of data fusion methods is further divided into four groups: multi-view learning-based, similarity-based, probabilistic dependency-based, and transfer learning-based methods. These methods focus on knowledge fusion rather than schema mapping and data merging, significantly distinguishing between cross-domain data fusion and traditional data fusion studied in the database community. This paper does not only introduce high-level principles of each category of methods, but also give examples in which these techniques are used to handle real big data problems. In addition, this paper positions existing works in a framework, exploring the relationship and difference between different data fusion methods. This paper will help a wide range of communities find a solution for data fusion in big data projects.

Index Terms— Big Data, cross-domain data mining, data fusion, multi-modality data representation, deep neural networks, multi-view learning, matrix factorization, probabilistic graphical models, transfer learning, urban computing.

1 INTRODUCTION

In the big data era, a wide array of data have been generated in different domains, from social media to transportation, from health care to wireless communication networks. When addressing a problem, we usually need to harness multiple disparate datasets [84]. For example, to improve urban planning, we need to consider the structure of a road network, traffic volume, points of interests (POIs) and populations in a city. To tackle air pollution, we need to explore air quality data together with meteorological data, emissions from vehicles and factories, as well as the dispersion condition of a place. To generate a more accurate travel recommendation for users, we shall consider the user's behavior on the Internet and in the physical world. To better understand an image's semantic meanings, we can use its surrounding text and the features derived from its pixels. So, how to unlock the power of knowledge from multiple datasets across different domains is paramount in big data research, essentially distinguishing big data from tradition data mining tasks.

However, the data from different domains consists of multiple modalities, each of which has a different representation, distribution, scale and density. For example, text is usually represented as discrete sparse word count vectors, whereas an image is represented by pixel intensities or outputs of feature extractors which are real-valued and dense. POIs are represented by spatial points associated with a static category, whereas air quality is represented using a geo-tagged time series. Human mobility data is represented by trajectories [82], whereas a road network is denoted as a spatial graph. Treating different datasets equally or simply concatenating the features from disparate datasets cannot achieve a good performance in data mining tasks [8][46][56]. As a result, fusing data across modalities becomes a new challenge in big data research, calling for advanced data fusion technology.

This paper summarizes three categories of methods that can fuse multiple datasets. The *first* category of data fusion methods use different datasets at different stages of a data mining task. We call them stage-based fusion methods. For example, Zheng et al. [86] first partition a city into disjoint regions by road network data, and then detect the pairs of regions that are not well connected based on human mobility data. These region pairs could denote the design that is out of date in a city's transportation network. The *second* category of methods learns a new representation of the original features extracted from different datasets by using deep neural networks (DNN). The new feature representation will then be fed into a model for classification or prediction. The *third* category blends data based on their semantic meanings, which can be further classified into four groups:

- *Multi-view-based methods*: This group of methods treats different datasets (or features from different datasets) as different views on an object or an event. Different features are fed into different models, describing an object from different perspectives. The results are later merged together or mutually reinforce each other. Co-Training is an example of this category.
- *Similarity-based methods*: This group of methods leverages the underlying correlation (or similarity) between different objects to fuse different datasets. A typical method is coupled collaborative filtering (CF), a.k.a.

- *Yu Zheng is with Microsoft Research, Beijing 100080, China. E-mail: yuzheng@ microsoft.com.*

Building 2, No. 5 Danling Street, Haidian District, Beijing 100080, China.

context-aware CF, where different datasets are modeled by different matrices with common dimensions. Through decomposing these matrices (or tensors) together, we can achieve a better result than solely factorizing a single matrix (or a tensor). Manifold alignment also belongs to this group.

- *Probabilistic dependency-based methods*: This group models the probabilistic causality (or dependency) between different datasets using a graphic representation. Bayesian Network and Markov Random Field are representative models, denoting features extracted from different datasets as graph nodes and the dependency between two features with an edge.
- *Transfer learning-based methods*: This group of methods transfers the knowledge from a source domain to another target domain, dealing with the data sparsity problems (including the feature structure missing or observation missing) in the target domain. Transfer learning can even transfer knowledge between different learning tasks, e.g. from book recommendation to travel recommendation.

The rest of this paper goes deeper each category of methods, introducing the high-level principle and representative examples for each category. With this paper, researchers and professionals are more capable of choosing proper approaches to solve real-world data fusion problems with big data. This paper also shares a collection of public datasets that can facilitate research on big data.

2 Related Work

2.1 Relation to Traditional Data Integration

Conventional data fusion [10], which is regarded as a part of data integration, is a process of integration of multiple data representing the same real-world object into a consistent, accurate, and useful representation. Fig. 1 A) presents the paradigm of conventional data fusion. For example, there are three POI datasets for Beijing generated by three different data providers. Conventional data fusion aims to merge the three datasets into a database with a consistent data schema, through a process of schema mapping and duplicate detection. The records (from different datasets) describing the same POI, e.g. a restaurant, are generated in the same domain, i.e. POI.

As illustrated in Fig. 1 B), however, in the era of big data, there are multiple datasets generated in different domains, which are implicitly connected by a latent object. For instance, traffic conditions, POIs and demography of a region describe the region's latent function collectively, while they are from three different domains. Literally, records from the three datasets describe different objects, i.e. a road segment, a POI, and a neighborhood, respectively. Thus, we cannot merge them straightforwardly by a schema mapping and duplication detection. Instead, we need to extract knowledge from each dataset by different methods, fusing the knowledge from them organically to understand a region's function collectively. This is more about knowledge fusion rather than schema mapping, which significantly differentiates between traditional data fusion (studied in the database community) and cross-domain data fusion.

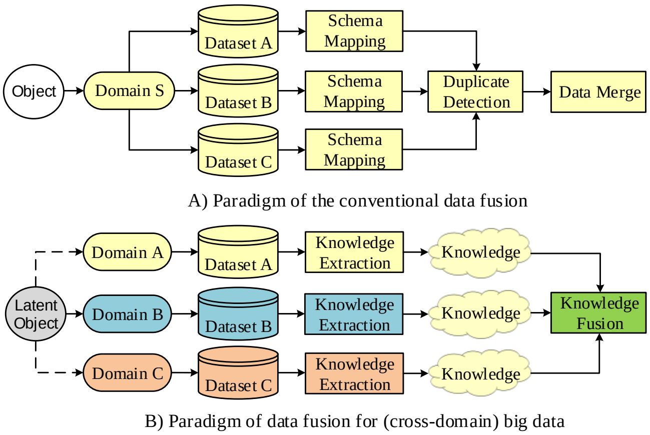


Fig. 1 Paradigms of different data fusion

2.2 Relation to Heterogeneous Information Network

An information network represents an abstraction of the real world, focusing on objects and interactions between objects. It turns out that this level of abstraction has great power in not only representing and storing essential information about the real-world, but also providing a useful tool to mine knowledge from it, by exploring the power of links [57]. Departing from many existing network models that view interconnected data as homogeneous graphs or networks, a heterogeneous information network consists of nodes and relations of different types. For example, a bibliographic information network consists of authors, conferences and papers as different types of nodes. Edges between different nodes in this network can denote different semantic meanings, e.g. an author publishes a paper, a paper is presented at a conference, and an author attends a conference. Quite a few algorithms have been proposed to mine a heterogeneous network, e.g. ranking and clustering [58][59].

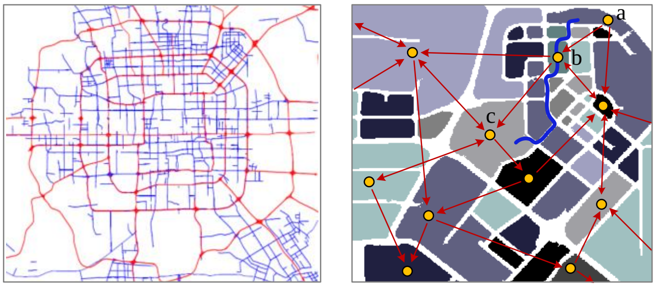
Heterogeneous information networks can be constructed in almost any domain, such as social networks, e-commerce, and online movie databases. However, it only links the object in a single domain rather than data across different domains. For instance, in a bibliographic information network, people, papers, and conferences are all from a bibliographic domain. In a Flickr information network, users, images, tags, and comments are all from a social media domain. If we want to fuse data across totally different domains, e.g. traffic data, social media, and air quality, such a heterogeneous network may not be able to find explicit links with semantic meanings between objects of different domains. Consequently, algorithms proposed for mining heterogeneous information networks cannot be applied to cross-domain data fusion directly.

3 Stage-Based Data Fusion Methods

This category of methods uses different datasets at the different stages of a data mining task. So, different datasets are loosely coupled, without any requirements on the consistency of their modalities.

Example 1: As illustrated in Fig. 2 A), Zheng et al. [86]

first partition a city into regions by major roads using a map segmentation method [75]. The GPS trajectories of taxicabs are then mapped onto the regions to formulate a region graph, as depicted in Fig. 2 B), where a node is a region and an edge denotes the aggregation of commutes (by taxis in this case) between two regions. The region graph actually blends knowledge from the road network and taxi trajectories. By analyzing the region graph, a body of research has been carried out to identify the improper design of a road network [86], detect and diagnose traffic anomalies [15][43] as well as find urban functional regions [74][76].



A) Map segmentation B) Region graph

Fig. 2. Map partition and graph building

Example 2: In friend recommendation, as illustrated in Fig. 3, Xiao et al. [67][68] first detect the stay points from an individual's location history (recorded in a form of spatial trajectories). As different users' location histories may not have any overlaps in the physical world, each stay point is then converted into a feature vector based on its surrounding POIs. For example, there are five restaurants, 1 shopping mall and 1 gas station around a stay point. In other words, the distance between these feature vectors denotes the similarity between the places people have visited.

Later, these stay points are hierarchically clustered into groups according to their feature vectors of POIs, formulating a tree structure, where a node is a cluster of stay points; a parent node is comprised of the stay points from its children nodes. By selecting the nodes (from the tree) that a user has at least one stay point in, we can represent the user's location history with a partial tree. A user's partial tree is further converted into a hierarchical graph, by connecting two nodes (on the same layer) with an edge, if the user has two consecutive stay points occurring in the two nodes. So, the hierarchical graph contains the information of a user's trajectories and the POIs of the places the user has visited. Because different users' hierarchical graphs are built based on the same tree structure, their location histories become comparable. Finally, the similarity between two users can be measured by the similarity between their hierarchical graphs.

Example 3: In the third example, Pan et al. [49] first detect a traffic anomaly based on GPS trajectories of vehicles and road network data. An anomaly is represented by a subgraph of a road network where drivers' routing behaviors significantly differ from their original patterns. Using the time span of the detected anomaly and the names of locations fallen in the anomaly's geographical scope as conditions, they retrieve the relevant social media (like tweets) that people have posted at the locations when the anomaly was happening. From the retrieved social media, they then try to describe the detected anomaly by mining representative terms, e.g. "parades" and "disasters", which barely occur in normal days but become frequent when the anomaly incurs. The first step scales down the scope of social media to be checked, while the second step enriches the semantic meaning of the results detected by the 1st step.

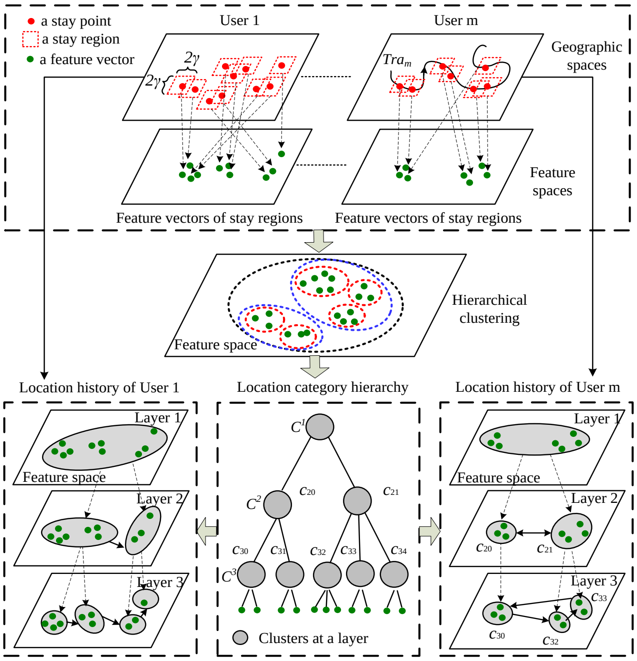


Fig. 3. Estimate user similarity using trajectories and POIs

Stage-based data fusion methods can be a meta-approach used together with other data fusion methods. For example, Yuan et al. [76] first use road network data and taxi trajectories to build a region graph, and then propose a graphical model to fuse the information of POIs and the knowledge of the region graph. In the second stage, a probabilistic-graphical-model-based method is employed in the framework of the stage-based method.

4. Feature-Level-Based Data Fusion

4.1 Direct Concatenation

Straightforward methods [66][70] in this category treat features extracted from different datasets equally, concatenating them sequentially into a feature vector. The feature vector is then used in clustering and classification tasks. As the representation, distribution and scale of different datasets may be very different, quite a few studies have suggested limitations to this kind of fusion [5] [46][56]. First, this concatenation causes over-fitting in the case of a small size training sample, and the specific statistical property of each view is ignored [69]. Second, it is difficult to discover highly non-linear relationships that exist between low-level features across different modalities [56]. Third, there are redundancies and dependencies between features extracted from different datasets which may be correlated.

Advanced learning methods [4][61][62] in this sub-category suggest adding a sparsity regularization in an objective function to handle the feature redundancy problem.

As a result, a machine learning model is likely to assign a weight close to zero to redundant features.

Example 4: Fu et al. [20] feed m features extracted from disparate datasets, such as taxi trajectories, POIs, road networks, and online social media, into a learn-to-rank model to predict the ranking (in terms of its potential investment value) of a residential real estate. Equation 1 is added to the learning-to-rank objective function to enforce sparse representations during learning.

$$P(\Psi|\Omega) = P(\boldsymbol{\omega}|0, \boldsymbol{\beta}^2)P(\boldsymbol{\beta}^2|a, b)$$
$$= \prod_m N(\omega_m|0, \beta_m^2) \prod_m Inverse - Gamma(\beta_m^2|a, b); \quad (1)$$

where $\boldsymbol{\omega} = (\omega_1, \omega_2, \dots, \omega_m)$ is a parameter vector of features, m is the number of features involved in a learning model, $\boldsymbol{\beta}^2 = (\beta_1^2, \beta_2^2, \dots, \beta_m^2)$ is the variance vector of the corresponding parameters. More specifically, the value of a parameter ω_m is assumed following a Guassian distribution with a zero mean and variance β_m^2. Setting a zero mean for the distribution reduces the probability of assigning ω_m a big value. A prior distribution, e.g. an inverse gamma, is further placed to regularize the value of β_m^2. To strengthen the sparsity, the constants a and b are usually set close to zero. Thus, β_m^2 tends to be small. In other words, feature weight ω_m has a very high probability of varying around the Gaussian expectation, i.e. zero. Through such a dual regularization (i.e., zero-mean Gaussian plus inverse-gamma), we can simultaneously regularize most feature weights to be zero or close to zero via a Bayesian sparse prior, while allowing for the possibility of a model learning large weights for significant features. In addition, the Bayesian sparse prior is a smooth function, and thus its gradient is easy to compute. Given that many objective functions are solved by gradient descent, the sparse regularization can be applied to many data mining tasks. However, the sparsity regularization of a Bayesian sparse prior is not as strong as L1 regularization.

4.2 DNN-Based Data Fusion

Recently, more advanced methods have been proposed to learn a unified feature representation from disparate datasets based on DNN. DNN is actually not fundamentally new in artificial intelligence. As depicted in Fig. 4 A), it is basically a multiple-layer neural network containing a huge number of parameters. Previously, a neural network is trained based on a back-propagation algorithm, which does not work well when the neural network has many hidden layers. Recently, new learning algorithms (a.k.a. deep learning), such as autoencoder and Restricted Boltzmann Machines (RBM), have been proposed to learn the parameters of a DNN layer by layer. Using supervised, unsupervised and semi-supervised approaches, Deep Learning learns multiple levels of representation and abstraction that help make sense of data, such as images, sound, and text. Besides being a predictor, DNN is also used to learn new feature representations [8], which can be fed into other classifiers or predictors. The new feature representations have proven more useful than hand-crafted features in image recognition [37] and speech translations [12]. A tutorial on DNN can be found in [40], and a survey on feature representation using DNN can be found in [8].

The majority of DNN is applied to handle data with a single modality. More recently, a series of research [46][52][56][55] starts using DNN to learn feature presentations from data with different modalities. This representation was found to be useful for classification and information retrieval tasks.

Example 5: Ngiam et al. [46] propose a deep autoencoder architecture to capture the "middle-level" feature representation between two modalities (e.g., audio and video). As shown in Table 1, three learning settings (consisting of cross-modality learning, shared representation learning, and multi-modal fusion) are studied. Fig. 4 B) presents the structure of the deep autoencoder for the cross-modality learning, where a single modality (e.g. video or audio) is used as the input to reconstruct a better feature representation for video and audio respectively. W.r.t. the shared representation learning and multi-modal fusion, which involve different modalities during training and testing, the paper adopts the architecture shown in Fig. 4. C). Extensive evaluations on these proposed deep learning models demonstrate that deep learning effectively learns 1) a better single modality representation with the help of other modalities; and 2) the shared representations capturing the correlations across multiple modalities.

Deep learning using Boltzmann Machines is another piece of work on multi-modality data fusion. Paper [56] first defines three criteria for a good multi-modality learning model: 1) the learned shared feature representation preserves the similarity of "concepts"; 2) the joint feature representation is easy to obtain in the absence of some modalities, and thus fills in missing modalities; 3) the new feature representation facilitates retrieval of one modality when querying from the other. A deep learning model, called multimodal Deep Boltzmann Machine (DBM), is proposed, to fuse images and texts for classification and retrieval problems. The proposed DBM model also satisfies the three criteria.

Table 1. Multi-Modal Feature Representation Learning [46]

	Feature learning	Supervised training	Testing
Classic deep learning	Audio	Audio	Audio
	Video	Video	Video
Cross modality learning	A + V	A	A
	A + V	V	V
Shared representation learning	A + V	A	V
	A + V	V	A
Multi-modal fusion	A + V	A + V	A + V

Example 6: As shown in Fig. 4 D), the multimodal DBM utilizes Gaussian-Bernoulli RBM (Restricted Boltzmann Machine) to model dense real-valued image features vectors, while employing replicated softmax to model sparse word count vectors. The multimodal DBM constructs a separate two-layer DBM for each modality and then combines them by adding a layer on top of them. Moreover, the multimodal DBM is a generative and undirected graphic model with bipartite connections between adjacent layers. This graphic model enables a bi-directional (bottom-up and top-down) search (denoted by the two red arrows).

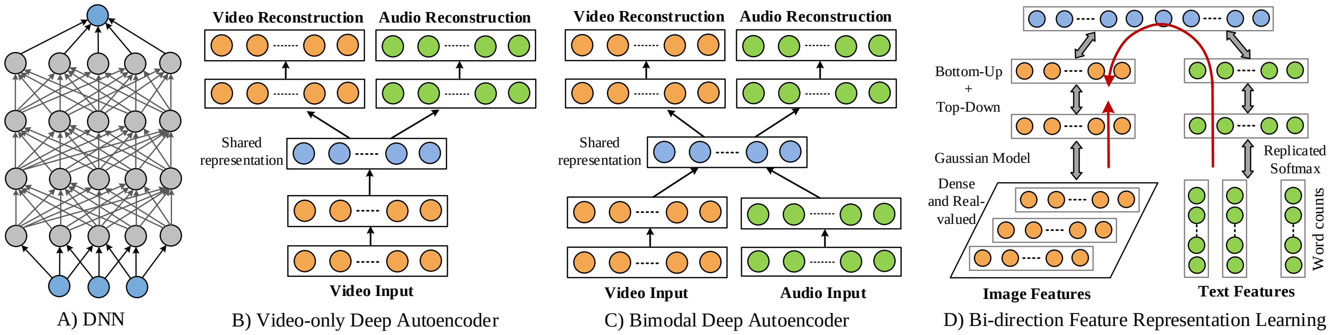


Fig. 4. Fusing multi-modality data using DNNs

Armed with a well-designed architecture, the key idea of multimodal DBM is to learn a joint density distribution over texts and images, i.e. $P(\mathbf{v}_{img}, \mathbf{v}_{text}; \theta)$ where θ include the parameters, from a large number of user-tagged images. The paper performs extensive experiments on classification as well as retrieval tasks. Both multimodal and unimodal inputs are tested, validating the effectiveness of the model in fusing multimodality data.

In practice, the performance of a DNN-based fusion model usually depends on how well we can tune parameters for the DNN. Finding a set of proper parameters can lead to a much better performance than others. Given a large number of parameters and a non-convex optimization setting, however, finding optimal parameters is still a labor-intensive and time-consuming process that heavily relies on human experiences. In addition, it is hard to explain what the middle-level feature representation stands for. We do not really understand the way a DNN makes raw features a better representation either.

5. Semantic Meaning-Based Data Fusion

Feature-based data fusion methods (introduced in Section 4) do not care about the meaning of each feature, regarding a feature solely as a real-valued number or a categorical value. Unlike feature-based fusion, semantic meaning-based methods understand the insight of each dataset and relations between features across different datasets. We know what each dataset stands for, why different datasets can be fused, and how they reinforce between one another. The process of data fusion carries a semantic meaning (and insights) derived from the ways that people think of a problem with the help of multiple datasets. Thus, they are interpretable and meaningful. This section introduces four groups of semantic meaning-based data fusion methods: multi-view-based, similarity-based, probabilistic dependency-based, and transfer-learning-based methods.

5.1 Multi-View Based Data Fusion

Different datasets or different feature subsets about an object can be regarded as different views on the object. For example, a person can be identified by the information obtained from multiple sources, such as face, fingerprint, or signature. An image can be represented by different feature sets like color or texture features. As these datasets describe the same object, there is a latent consensus among them. On the other hand, these datasets are complementary to each other, containing knowledge that other views do not have. As a result, combining multiple views can describe an object comprehensively and accurately.

According to [69], the multi-view learning algorithms can be classified into three groups: 1) co-training, 2) multiple kernel learning, and 3) subspace learning. Notably, co-training style algorithms [11] train alternately to maximize the mutual agreement on two distinct views of the data. Multiple kernel learning algorithms [23] exploit kernels that naturally correspond to different views and combine kernels either linearly or non-linearly to improve learning. Subspace learning algorithms [16] aim to obtain a latent subspace shared by multiple views, assuming that the input views are generated from this latent subspace.

5.1.1. Co-Training

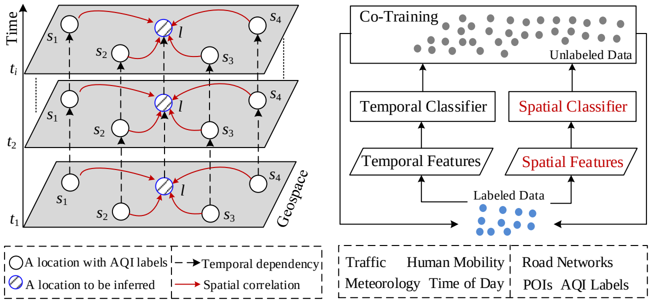
Co-training [11] was one of the earliest schemes for multi-view learning. Co-training considers a setting in which each example can be partitioned into two distinct views, making three main assumptions: 1) Sufficiency: each view is sufficient for classification on its own, 2) Compatibility: the target functions in both views predict the same labels for co-occurring features with high probability, and 3) Conditional independence: the views are conditionally independent given the class label. The conditional independence assumption is usually too strong to be satisfied in practice. Consequently, several weaker alternatives [6] have thus been considered.

In the original co-training algorithm [11], given a set L of labeled examples and a set U of unlabeled examples, the algorithm first creates a smaller pool U' containing u unlabeled examples. It then iterates the following procedures. First, use L to train two classifiers f_1 and f_2 on the view v_1 and v_2 respectively. Second, allow each of these two classifiers to examine the unlabeled set U' and add the p examples it most confidently labels as positive, and n examples it most confidently labels as negative to L, along with the labels assigned by the corresponding classifier. Finally, the pool U' is replenished by drawing $2p + 2n$ examples from U at random. The intuition behind the co-training algorithm is that classifier f_1 adds examples to the labeled set that classifier f_2 will then be able to use for learning. If the independence assumption is violated, on average the added examples will be less informative. Thus, Co-training may not be that successful. Since then, many variants have been developed.

Instead of assigning labels to the unlabeled examples, Nigam et al. [47] give unlabeled examples probabilistic labels that may change from one iteration to another by running EM (Expectation and Maximization) in each view. This algorithm, called Co-EM, outperforms co-training for many problems, but requires the classifier of each view to generate class probabilities. By reformulating the SVM (supported vector machine) in a probabilistic way, Brefeld et al. [12] develop a co-EM version of SVM to close this gap. Zhou et al. [92] expand the co-training style algorithms from classification to regression problems. They propose an algorithm, called CoREG, which employs two k-nearest neighbor (kNN) regressors. Each regressor labels the unlabeled data for the other during the learning process. For the sake of choosing the appropriate unlabeled examples to label, CoREG estimates the labeling confidence by consulting the influence of the labeling of unlabeled examples on the labeled examples. The final prediction is made by averaging the regression estimates generated by both regressors.

Example 7: Zheng et al. [83][85] propose a co-training-based model to infer the fine-grained air quality throughout a city based on five datasets: air quality, meteorological data, traffic, POIs and road networks. Fig. 5 A) illustrates the philosophy of the model from multi-view learning's perspective. Naturally, air quality has temporal dependency in an individual location (represented by the broken black arrows) and the spatial correlation among different locations (denoted by the red solid arrows). For example, the current air quality of a location depends on past hours. In addition, the air quality of a place could be bad if the air quality of its surrounding locations is bad. So, the temporal dependency and spatial correlation formulate two distinct views (a temporal view and a spatial view) on the air quality of a location.

As presented in Fig. 5 B), a co-training-based framework is proposed, consisting of two classifiers. One is a spatial classifier based on an artificial neural network (ANN), which takes spatially-related features (e.g., the density of POIs and length of highways) as input to model the spatial correlation between air qualities of different locations. The other is a temporal classifier based on a linear-chain conditional random field (CRF), involving temporally-related features (e.g., traffic and meteorology) to model the temporal dependency of air quality in a location. The two classifiers are first trained based on limited labeled data using non-overlapped features, and then infer unlabeled instances respectively. The instances that are confidently inferred by a classifier in each round are brought to the training set, which will be used to re-train the two classifiers in the next round. The iteration can be stopped until the unlabeled data has been consumed out or the inference accuracy does not increase any more. When inferring the label of an instance, we send different features to different classifiers, generating two sets of probabilities across different labels. The label that maximizes the production of the corresponding probabilities from the two classifiers is selected as the result.



A) Philosophy of the inference model B) Procedure of co-training

Fig. 5 Co-training-based air quality inference model

The proposed method was evaluated based on data from four cities, showing its advantages beyond four categories of baselines: interpolation-based methods, classical dispersion models, well-known classification models like decision tree and CRF, and ANNs. In the later two categories of baselines, all the features are fed into a single model without differentiating between their semantic meanings and views.

5.1.2. Multi-Kernel Learning

Multiple Kernel Learning (MKL) refers to a set of machine learning methods that uses a predefined set of kernels and learns an optimal linear or non-linear combination of kernels as part of the algorithm. A kernel is a hypothesis on the data, which could be a similarity notion, or a classifier, or a regressor. According to [23], there are two uses of MKL (as shown in Fig. 6):

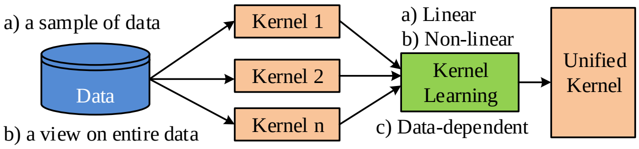


Fig. 6. Procedure of Multi-Kernel Learning

a) Different kernels correspond to different notions of similarity. A learning method picks the best kernel, or uses a combination of these kernels. A sample of data is retrieved from the entire set to train a kernel based on all features. As a specific kernel may be a source of bias, allowing a learner to choose among a set of kernels can result in a better solution. For example, there are several kernel functions, such as the linear, polynomial and Gaussian kernels, successfully used in SVM. This kind of MKL was not originally designed for multi-view learning, as the entire feature set is used for training each kernel.

b) A variation of the first use of MKL is to train different kernels using inputs coming from different representations possibly from different sources or modalities. Since these are different representations, they have different measures of similarity corresponding to different kernels. In such a case, combining kernels is one possible way to combine multiple information sources. The reasoning is similar to combining different classifiers. Noble [48] calls this method of combining kernels *intermediate combination*, in contrast with *early combination* (where features from different sources are concatenated and fed to a single learner) and *late combination* (where different features are fed to different classifiers whose decisions are then combined by a fixed or trained combiner).

There are three ways to combine the results of kernels: linear, non-linear, and data-dependent combinations. The linear combination consists of unweighted (i.e. mean) and weighted sum. Nonlinear combination methods [63] use nonlinear functions of kernels, namely, multiplication, power, and exponentiation. Data-dependent combination methods assign specific kernel weights for each data instance. By doing this, they can identify local distributions in the data and learn proper kernel combination rules for each region [23].

Existing MKL algorithms have two main groups of training methodology: 1) One-step methods calculate the parameters of the combination function and base learners in a single pass, using a sequential approach or a simultaneous approach. In the sequential approach, the combination function parameters are determined first, and then a kernel-based learner is trained using the combined kernel. In the simultaneous approach, both set of parameters are learned together. 2) Two-step methods use an iterative approach. In each iteration, we first update the parameters of the combination function while fixing that of the base learner. We then update the parameters of base learners while fixing the parameters of the combination function. These two steps are repeated until convergence.

Example 8: Ensemble and boosting methods [1], such as Random Forest [13], are inspired by MKL. Random Forest combines the idea of Bootstrap Aggregating (also called Bagging) and the random selection of features [27][28], in order to construct a collection of decision trees with a controlled variance. More specifically, it trains multiple Decision Trees by selecting a portion of training data each time based on Bagging and a portion of features according to the principle introduced by [27][28]. When a test case comes, different selections of the case's features are sent to corresponding Decision Trees (i.e. kernels) simultaneously. Each kernel generates a prediction, which is then aggregated linearly.

Example 9: Zheng et al. [89] forecast air quality for the next 48 hours of a location based on five datasets. Fig. 7 presents the architecture of the predictive model, which contains two kernels (spatial predictor and temporal predictor) and a kernel learning module (i.e. the prediction aggregator). The Temporal Predictor predicts the air quality of a station in terms of the data about the station, such as the local meteorology, AQIs of the past few hours and the weather forecast of the place. Instead, the Spatial Predictor considers the spatial neighbor data, such as the AQIs and the wind speed at the other stations, to predict a station's future air quality. The two predictors generate their own predictions independently for a station, which are combined by the Prediction Aggregator dynamically according to the current weather conditions of the station. Sometimes, local prediction is more important, while spatial prediction should be given a higher weight on other occasions (e.g. when wind blows strongly). The Prediction Aggregator is based on a Regression Tree, learning the dynamic combination between the two kernels from the data.

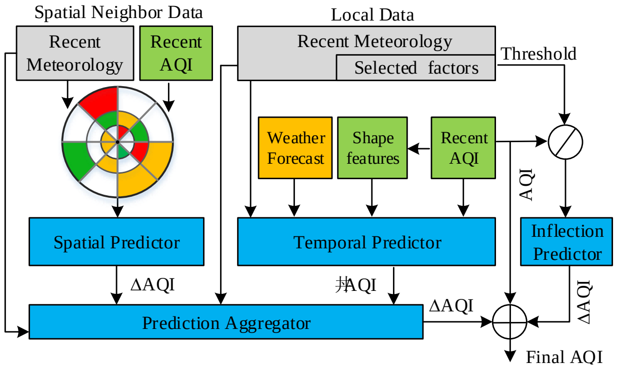


Fig. 7. MKL-based framework for forecasting air quality

The MKL-based framework outperforms a single kernel-based model in the air quality forecast example, for the following three reasons: 1) *From the feature space's perspective*: The features used by the spatial and temporal predictors do not have any overlaps, providing different views on a station's air quality. 2) *From the model's perspective*: The spatial and temporal predictors model the local factors and global factors respectively, which have significantly different properties. For example, the local is more about a regression problem, while the global is more about a nonlinear interpolation. Thus, they should be handled by different techniques. 3) *From the parameter learning's perspective*: Feeding all the features into a single model results in a big model with many parameters to learn. However, the training data is limited. For instance, we only have one and half year AQI data of a city. Decomposing a big model into three organically coupled small models scales down the parameter spaces tremendously, leading to more accurate learning and therefore the prediction.

5.1.3. Subspace Learning

Subspace learning-based approaches aim to obtain a latent subspace shared by multiple views by assuming that input views are generated from this latent subspace, as illustrated in Fig. 8. With the subspace, we can perform subsequent tasks, such as classification and clustering. Additionally, as the constructed subspace usually has a lower dimensionality than that of any input view, the "curse of dimensionality" problem can be solved to some extent.

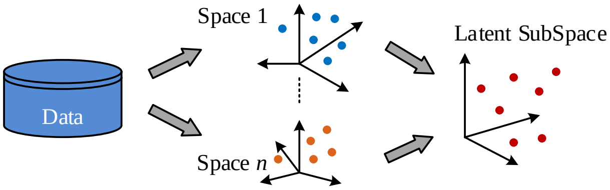


Fig.8. Concept of subspace learning

In the literature on single-view learning, principal component analysis (PCA) is a widely-used technique to exploit the subspace for single-view data. Canonical correlation analysis (CCA) [25] can be regarded as the multi-view version of PCA. Through maximizing the correlation between two views in the subspace, CCA outputs one optimal projection on each view. The subspace constructed by CCA is linear, and thus cannot be straightforwardly applied to non-linearly embedded datasets. To address this issue, the kernel variant of CCA, namely KCCA [38], was proposed to map each (non-linear) data point to a higher space in which linear CCA operates. Both CCA and KCCA exploit the subspace in an unsupervised way. Motivated by the generation of CCA from PCA, multi-view Fisher discriminant analysis [33] is developed to find informative projections with label information. Lawrence [39] casts the Gaussian process as a tool to construct a latent variable model which could accomplish the task of non-linear dimensional reduction. Chen et al. [16] develop a statistical framework that learns a predictive subspace shared by multiple views based on a generic multi-view latent space Markov network.

5.2 Similarity-Based Data Fusion

Similarity lies between different objects. If we know two objects (X, Y) are similar in terms of some metric, the information of X can be leveraged by Y when Y is lack of data. When X and Y have multiple datasets respectively, we can learn multiple similarities between the two objects, each of which is calculated based on a pair of corresponding datasets. These similarities can mutually reinforce each other, consolidating the correlation between two objects collectively. The latter enhances each individual similarity in turn. For example, the similarity learned from a dense dataset can reinforce those derived from other sparse datasets, thus helping fill in the missing values of the latter. From another perspective, we can say we are more likely to accurately estimate the similarity between two objects by combining multiple datasets of them. As a result, different datasets can be blended together based on similarities. Coupled matrix factorization and manifold alignment are two types of representative methods in this category.

5.2.1. Coupled Matrix Factorization

Before elaborating on the coupled matrix factorization, we need to introduce two concepts. One is collaborative filtering (CF); the other is matrix factorization. The latter can be an efficient approach to the implementation of CF models.

5.2.1.1 Collaborative Filtering

CF is a well-known model widely used in recommender systems. The general idea behind collaborative filtering is that similar users make ratings in a similar manner for similar items [21]. Thus, if similarity is determined between users and items, a potential prediction can be made as to the rating of a user with regards to future items. Users and items are generally organized by a matrix, where an entry denotes a user's rating on an item. The rating can be explicit rankings or implicit indications, such as the number of visits to a place or the times that a user has browsed an item. Once formulating a matrix, the distance between two rows in the matrix denotes the similarity between two users, while the distance between two columns stands for the similarity between two items.

Memory-based CF is the most widely-used algorithm that computes the value of the unknown rating for a user and an item as an aggregate of the ratings of some other (usually, the N most similar) users for the same item. There are two classes of memory-based CF models: user-based [45] and item-based [42] techniques. For example, user p's interest (r_{pi}) in a location i can be predicted according to Equation 1, which is an implementation of user-based collaborative filtering [45][91]:

$$r_{pi} = \overline{R_p} + d \sum_{u_q \in U'} sim(u_p, u_q) \times (r_{qi} - \overline{R_q}); \tag{2}$$

$$d = \frac{1}{|U'|} \sum_{u_q \in U'} sim(u_p, u_q); \tag{3}$$

$$\overline{R_p} = \frac{1}{|S(R_p)|} \sum_{i \in S(R_p)} r_{pi}, \ ; \tag{4}$$

where $sim(u_p, u_q)$ denotes the similarity between user u_p and u_q; $\overline{R_q}$ and $\overline{R_p}$ mean the average rating of u_p and u_q respectively, denoting their rating scale; $S(R_p)$ represents the collection of items rated by u_p; U' is the collection of users who are the most similar to u_q. $r_{qi} - \overline{R_q}$ is to avoid rating biases of different users. When the number of users becomes big, computing the similarity between each pair of users is impractical for a real system. Given that the number of items could be smaller than that of users, item-based CF, e.g. the Slop One algorithm [42], was proposed to address this issue. When the number of users and number of items are both huge, matrix factorization-based method is employed to implement a CF model.

5.2.1.2 Matrix Factorization

Matrix factorization decomposes a (sparse) matrix X into the production of two (low-rank) matrices, which denote the latent variables of users and items respectively. The production of the two matrices can approximate matrix X, therefore helping fill the missing values in X. There are two widely used matrix factorization methods: Singular Value Decomposition (SVD) [22][35] and non-negative matrix factorization (NMF) [30][41].

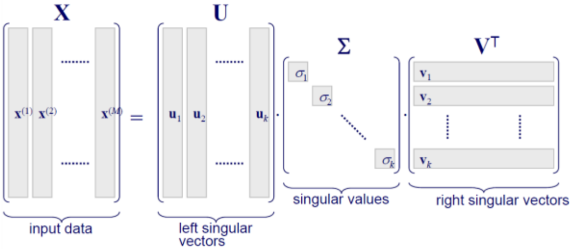


Fig. 9. SVD matrix factorization

1) SVD factorizes an $m\times n$ matrix X into the production of three matrices $X = U\Sigma V^T$, where U is a $m\times m$ real unitary matrix (a.k.a. left singular vectors), Σ is an $m\times n$ rectangular diagonal matrix with non-negative real numbers on the diagonal (a.k.a. singular values); V^T is a $n\times n$ real unitary matrix (a.k.a. left singular vectors). In practice, as shown in Fig. 9, when trying to approximate matrix X by $U\Sigma V^T$, we only need to keep top k biggest singular values in Σ and the corresponding singular vectors in U and V. SVD has some good properties. First, U and V are orthogonal matrices; i.e. $U \cdot U^T = I$ and $V \cdot V^T = I$. Second, the value of k can be determined by Σ. For example, select the first k diagonal entries (in Σ) whose sum is larger than 90% of the entire diagonal entries' sum. However, SVD is more computationally expensive and harder to parallelize, as compared to NFM.

2) *NFM* factorizes an $m\times n$ matrix R (with m users and n items) into a production of an $m\times K$ matrix P and $K\times n$

matrix Q , $R = P \times Q$, with the property that all three matrices have no negative elements. This non-negativity makes the resulting matrices easier to inspect [30]. Additionally, non-negativity is inherent to the data being considered in many applications, such as location recommendation [2][88], traffic estimation [53], and processing of audio spectrums. Each row of matric P denotes the latent feature of a user; each column of matrix Q stands for the latent feature of an item. K can be significantly smaller than m and n, denoting the number of latent features for a user and an item. To predict a rating of an item d_j by u_i, we can calculate the dot product of the two vectors corresponding to u_i and d_j as Equation 5.

$$\hat{r}_{ij} = p_i^T q_j = \sum_{k=1}^{k} p_{ik} q_{kj}; \qquad (5)$$

To find a proper P and Q, we can first initialize the two matrices with some values and calculate the difference between their product and R, as shown in Equation 6. We can then try to minimize e_{ij}^2 iteratively using gradient descent, which finds a local minimum of the difference.

$$e_{ij}^2 = (r_{ij} - \hat{r}_{ij})^2 = (r_{ij} - \sum_{k=1}^{K} p_{ik} q_{kj})^2; \qquad (6)$$

Specifically, to know in which direction we have to modify the values, we differentiate Equation 6 with respect to p_{ik} and q_{kj} separately:

$$\frac{\partial e_{ij}^2}{\partial p_{ik}} = -2(r_{ij} - \hat{r}_{ij})(q_{kj}) = -2e_{ij} q_{kj}; \qquad (7)$$

$$\frac{\partial e_{ij}^2}{\partial q_{kj}} = -2(r_{ij} - \hat{r}_{ij})(p_{ik}) = -2e_{ij} p_{ik}; \qquad (8)$$

Having obtained the gradient, we can now formulate the update rules for p_{ik} and q_{kj} as follows:

$$p'_{ik} = p_{ik} + \alpha \frac{\partial e_{ij}^2}{\partial p_{ik}} = p_{ik} + 2e_{ij} q_{kj}; \qquad (9)$$

$$q'_{kj} = q_{kj} + \alpha \frac{\partial e_{ij}^2}{\partial q_{kj}} = q_{kj} + 2e_{ij} p_{ik}; \qquad (10)$$

where α is a small value that determines the rate of approaching the minimum. When optimizing p_{ik}, NFM fixes q_{kj}, vice versa; the gradient descent is performed iteratively until the total error $\sum e_{ij}^2$ converges to its minimum. To avoid over fitting, a regularization is introduced to the error function.

$$e_{ij}^2 = (r_{ij} - \sum_{k=1}^{K} p_{ik} q_{kj})^2 + \frac{\beta}{2} \sum_{k=1}^{K} (\|P\|^2 + \|Q\|^2); \qquad (11)$$

As compare to SVD, NFM is flexible and can be parallelized, but it is less precise.

5.2.1.3. Coupled Matrix Factorization

Depending on applications, an item can also be a location [2][88][91], a website, or a company, while users can be drivers, or passengers, or subscribers of a service. We can even generalize a user to an object and an item to a property of the object. When there are multiple datasets concerning an object, we cannot simply deposit different properties from different sources into a single matrix. As different datasets have different distributions and meanings, factorizing them in a single matrix would lead to an inaccurate complementation of missing values in the matrix. Advanced methods [80][53] use coupled matrix factorization (or called context-aware matrix factorization) [54] to accommodate different datasets with different matrices, which share a common dimension between one another. By decomposing these matrices collaboratively, we can transfer the similarity between different objects learned from a dataset to another one, therefore complementing the missing values more accurately.

Example 10: Zheng et al. [80] propose a coupled matrix factorization method to enable location-activity recommendation. As illustrated in Fig. 10, a location-activity matrix X is built based on many users' location histories. A row of X stands for a venue and a column represents an activity (like shopping and dinning). An entry in matrix X denotes the frequency that a particular activity has been performed in a particular location. If this location-activity matrix is completely filled, we can recommend a set of locations for a particular activity by retrieving the top k locations with a relatively high frequency from the column that corresponds to that activity. Likewise, when performing activity recommendation for a location, the top k activities can be retrieved from the row corresponding to the location. However, the location-activity matrix is incomplete and very sparse, as we only have a portion of users' data (and an individual can visit very few locations). Accordingly, a traditional CF model does not work very well in generating quality recommendations. Solely factorizing X does not help much either as the data are over sparse.

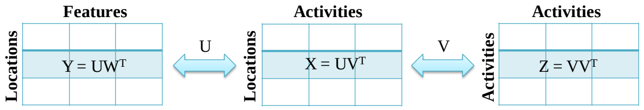


Fig. 10 Coupled matrix factorization for recommendation

To address this issue, the information from another two matrices (Y and Z), respectively shown in the left and right part of Fig. 10, are incorporated into the matrix factorization. One is a location-feature matrix; the other is an activity-activity matrix. Such kind of additional matrices are usually called contexts, which can be learned from other datasets. In this example, matrix Y, where a row stands for a location and a column denotes a category of POIs (such as restaurants and hotels) that fall in the location, is built based on a POI database. The distance between two rows of matrix Y denotes the similarity between two locations in terms of their geographical properties. The insight is that two locations with a similar geographical property could have similar user behaviors. Matrix Z models the correlation between two different activities, which can be learned from the search results by sending the titles of two activities into a search engine. The main idea is to propagate the information among X, Y and Z by requiring them to share low-rank matrices U and V in a collective matrix factorization model. As matrix Y and Z are built based on dense data, we can obtain an accurate decomposition of them, i.e. matrices U and V. Thus, Matrix X can be complemented more accurately by $X = UV^T$. More specifically, an objective function was formulated as Equation 12:

$$L(U,V,W) = \frac{1}{2} \| I \circ (X - UV^T) \|_F^2 + \frac{\lambda_1}{2} \| Y - UW^T \|_F^2 +$$

$$\frac{\lambda_2}{2}\parallel Z-VV^T\parallel_F^2+\frac{\lambda_3}{2}(\parallel U\parallel_F^2+\parallel V\parallel_F^2+\parallel W\parallel_F^2), \qquad (12)$$

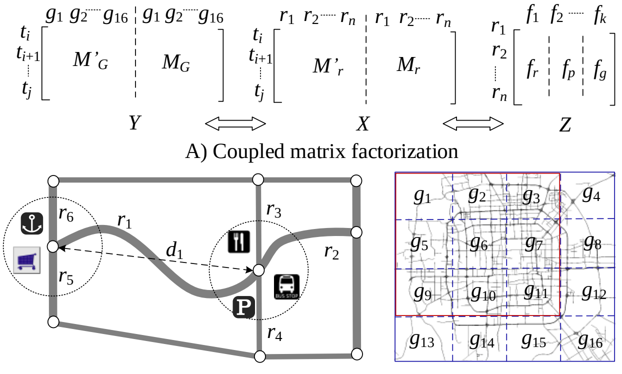
where $\|\cdot\|_F$ denotes the Frobenius norm. I is an indicator matrix with its entry $I_{ij}=0$ if X_{ij} is missing, $I_{ij}=1$ otherwise. The operator "$\circ$" denotes the entry-wise product. The first three terms in the objective function control the loss in matrix factorization, and the last term controls the regularization over the factorized matrices so as to prevent overfitting. In general, this objective function is not jointly convex to all the variables U, V and W. Consequently, some numerical method, such as gradient descent, was used to get local optimal solutions.

Example 11: Shang and Zheng et al. [53] propose a coupled-matrix factorization method to instantly estimate the travel speed on each road segment throughout an entire city, based on the GPS trajectory of a sample of vehicles (such as taxicabs). As shown in Fig. 11 A), after map matching the GPS trajectories onto a road network, they formulate a matrix M'_r with a row denoting a time slot (e.g., 2pm-2:10pm) and a column standing for a road segment. Each entry in M'_r contains the travel speed on a particular road segment and in a particular time slot, calculated based on the recently received GPS trajectories. The goal is to fill the missing values in row t_j, which corresponds to the current time slot. Though we can achieve the goal by solely applying matrix factorization to M'_r, the accurarcy of the inference is not very high as the majority of road segments are not covered by trajectories.

To address this issue, four context matrices (M_r, M_G, M'_G and Z) are built. Specifically, M_r stands for the historical traffic patterns on road segments. While the rows and columns of M_r have the same meaning as M'_r, an entry of M_r denotes the average travel speed derived from historical data over a long period. The difference between the two corresponding entries from M'_r and M_r indicates the deviation of current traffic situation (on a road segment) from its average patterns. As depicted in Fig.11 B), Z contains the physical features of a road segment, such as the shape of a road, number of lanes, speed constraint, and the distribution of surrounding POIs. The general assumption is that two road segments with similar geographical properties could have similar traffic conditions at the same time of day. To capture high-level traffic conditions, as demonstrated in Fig. 11 C), a city is divided into uniform grids. By projecting the recently received GPS trajectories into these grids, a matrix M'_G is built, with a column standing for a grid and a row denoting a time slot; an entry of M'_G means the number of vehicles traveling in a particular grid and at a particular time slot. Likewise, by projecting historical trajectories over a long period into the grids, a similar M_G is built, with each entry being the average number of vehicles traveling in a particular grid and at a particular time slot. So, M'_G denotes the real-time high-level traffic conditions in a city and M_G indicates the historical high-level traffic patterns. The difference between the same entries of the two matrices suggests the deviation of current high-level traffic conditions from their historical averages. By combining these matrices, i.e. $X=M'_r||M_r$ and $Y=M'_G||M_G$, a coupled matrix factorization is applied to X, Y, and Z, with the objective function as Equation 13.

$$L(T,R,G,F)=\frac{1}{2}||Y-T(G;G)^T||^2+\frac{\lambda_1}{2}||X-T(R;R)^T||^2$$
$$+\frac{\lambda_2}{2}||Z-RF^T||^2+\frac{\lambda_3}{2}(||T||^2+\left||R|\right|^2+||G||^2+||F||^2) \qquad (13)$$

where $\|\cdot\|$ denotes the Frobenius norm. The first three terms in the objective function control the loss in matrix factorization, and the last term is a regularization of penalty to prevent over-fitting.



A) Coupled matrix factorization

B) Extract features for a road segment

C) Divide a city into grids

Fig. 11 Estimate traffic conditions based on trajectories

5.2.3 Manifold Alignment

Manifold alignment utilizes the relationships of instances within each dataset to strengthen the knowledge of the relationships between the datasets, thereby ultimately mapping initially disparate datasets to a joint latent space [64]. Manifold alignment is closely related to other manifold learning techniques for dimensionality reduction, such as Isomap [60], locally linear embedding [51], and Laplacian Eigenmaps [7]. Given a dataset, these algorithms attempt to identify the low dimensional manifold structure of that dataset and preserve that structure in a low dimensional embedding of the dataset. Manifold alignment follows the same paradigm but embeds multiple datasets. There are two key ideas in manifold alignment:

1) Manifold alignment preserves the correspondences across datasets; it also preserves the individual structures within each dataset by mapping similar instances in each dataset to similar locations in the Euclidean space. As illustrated in Fig. 12, manifold alignment maps two datasets (X, Y) to a new joint latent space ($f(X), g(Y)$), where locally similar instances within each dataset and corresponding instances across datasets are close or identical in that space. The two similarities are modeled by a lossy function with two parts: one for preserving the local similarity within a dataset, and the other for the correspondences across different datasets.

Formally, with c datasets $X^1, X^2, \dots, X^c$, the local similarity within each data set is modeled by Equation 14:

$$C_\lambda(F^a)=\sum_{i,j}||F^a(i,.)-F^a(j,.)||^2\cdot W^a(i,j), \qquad (14)$$

where X^a is the a-th dataset, which is a $n_a\times p_a$ data matrix with n_a observations and p_a features. F^a is the embedding of X^a; W^a is an $n_a\times n_a$ matrix, where $W^a(i,j)$ is the similarity between instance $X^a(i,.)$ and $X^a(j,.)$. The sum is taken over all pairs of instances in that dataset. $C_\lambda(F^a)$ is the cost of preserving the local similarities within X^a. If two data instances, $X^a(i,.)$ and $X^a(j,.)$, from X^a are similar, which happens when $W^a(i,j)$ is larger, their locations in the latent space, $F^a(i,.)$ and $F^a(i,.)$ should be closer, i.e. $||F^a(i,.) - F^a(j,.)||^2$ is small. D^a is an $n_a \times n_a$ diagonal matrix with $D^a(i,i) = \sum_j W^a(i,j)$. $L^a = D^a - W^a$ is the Laplacian associated with X^a.

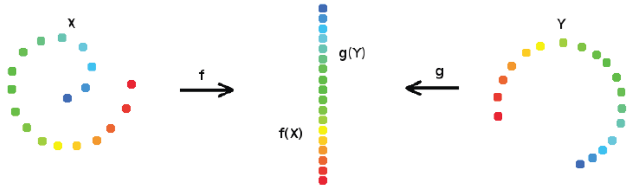


Fig. 12. Manifold alignment of two data sets [64]

To preserve the correspondence information about instances between two datasets, X^a and X^b, the cost of each pair of correspondence is $C_k(F^a, F^b)$:

$$C_k(F^a, F^b) = \sum_{i,j} ||F^a(i,.) - F^b(j,.)||^2 \cdot W^{a,b}(i,j), \quad (15)$$

where $W^{a,b}(i,j)$ is the similarity, or the strength of correspondence, of two instances, $X^a(i,.)$ and $X^b(j,.)$. If the two data points are in a stronger correspondence, which happens when $W^{a,b}(i,j)$ is larger, their locations in the latent space, $F^a(i,.)$ and $F^b(j,.)$, should be closer together. Typically, $W^{a,b}(i,j)$ =1 if $X^a(i,.)$ and $X^b(j,.)$ are in correspondence. So, the complete lossy function is:

$$C_1(F^1, F^2, \dots, F^k) = u \cdot \sum_a C_\lambda(F^a) + v \cdot \sum_{a \neq b} C_k(F^a, F^b); \quad (16)$$

Typically, $u = v$ =1.

2) At the algorithmic level, manifold alignment assumes the disparate datasets to be aligned have the same underlying manifold structure. The second loss function is simply the loss function for Laplacian eigenmaps using the joint adjacency matrix:

$$C_2(\mathbf{F}) = \sum_{i,j} ||\mathbf{F}(i,.) - \mathbf{F}(j,.)||^2 \cdot \mathbf{W}^{a,b}(i,j); \quad (17)$$

where the sum is taken over all pairs of instances from all datasets; $\mathbf{F}$ is the unified representation of all the datasets and $\mathbf{W}$ is the $(\sum_a n_a \times \sum_a n_a)$ joint adjacency matrix of all the datasets.

$$\mathbf{W} = \begin{pmatrix} vW^1 & uW^{1,2} & \cdots & uW^{1,c} \\ \vdots & & \ddots & \vdots \\ uW^{c,1} & uW^{c,2} & \cdots & vW^c \end{pmatrix}. \quad (18)$$

Equation 19 denotes that if two data instances, $X^a(i,.)$ and $X^a(j,.)$, are similar, regardless of whether they are in the same dataset $(a = b)$ or from different datasets $(a \neq b)$, which happens when $\mathbf{W}(i,j)$ is larger in either case, their locations in the latent space, $\mathbf{F}(i,.)$ and $\mathbf{F}(i,.)$, should be closer together. Making use of the fact that $||\mathrm{M}(i,.)||^2 = \sum_k M(i,k)^2$ and that the Laplacian is a quadratic difference operator,

$$\begin{aligned} C_2(\mathbf{F}) &= \sum_{i,j} \sum_k ||\mathbf{F}(i,k) - \mathbf{F}(j,k)||^2 \cdot \mathbf{W}^{a,b}(i,j) \\ &= \sum_k \sum_{i,j} ||\mathbf{F}(i,k) - \mathbf{F}(j,k)||^2 \cdot \mathbf{W}^{a,b}(i,j) \\ &= \sum_k tr(\mathbf{F}(.,k)'\mathbf{L}\mathbf{F}(.,k)) = tr(\mathbf{F}'\mathbf{L}\mathbf{F}) \end{aligned} \quad (19)$$

Where $tr(\cdot)$ denotes the matrix trace; $\mathbf{L} = \mathbf{D} - \mathbf{W}$ is the joint Laplacian matrix of all the datasets. $\mathbf{D}$ is an $(\sum_a n_a \times \sum_a n_a)$ diagonal matrix with $\mathbf{D}^a(i,i) = \sum_j \mathbf{W}(i,j)$. Standard manifold learning algorithms are then invoked on $\mathbf{L}$ to obtain a joint latent representation of the original datasets. Manifold alignment can therefore be viewed as a form of constrained joint dimensionality reduction that finds a low-dimensional embedding of multiple datasets that preserves any known correspondences across them [64].

Example 12: Zheng et al. [87] infer the fine-grained noise situation by using 311 complaint data together with social media, road network data, and POIs. As shown in Fig. 13, they model the noise situation of NYC with a three dimension tensor, where the three dimensions stand for regions, noise categories, and time slots, respectively. An entry $\mathcal{A}(i,j,k)$ stores the total number of 311 complaints of category c_j in region r_i and time slot t_k over the given period of time. This is a very sparse tensor, as there may not be people reporting noise situation anytime and anywhere. If the tensor can be filled completely, we are able to know the noise situation throughout the city.

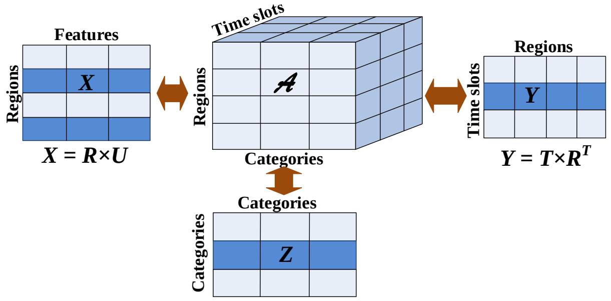


Fig. 13. Context-aware tensor decomposition with manifold

To deal with the data sparsity problem, they extract three categories of features, geographical features, human mobility features and noise category correlation features (denoted by matrices X, Y, and Z), from POI/road network data, user check-ins, and 311 data, respectively. For example, a row of matrix X denotes a region, and each column stands for a road network feature, such as the number of intersections and the total length of road segments in the region. Matrix X incorporates the similarity between two regions in terms of their geographic features. Intuitively, regions with similar geographic features could have a similar noise situation. $Z \in \mathbb{R}^{M \times M}$ is the correlation matrix between different categories of noise. $Z(i,j)$ denotes how often a category of noise c_i co-occurs with another category c_j.

These features are used as contexts in a context-aware tensor decomposition approach to supplement the missing entries of the tensor. More specifically, $\mathcal{A}$ is decomposed into the multiplication of a few (low-rank) matrices and a core tensor (or just a few vectors), based on $\mathcal{A}$'s non-zero entries. Matrix X can be factorized into the multiplication of two matrices, $X = R \times U$, where $R \in \mathbb{R}^{N \times d_R}$ and $U \in \mathbb{R}^{d_R \times P}$ are low rank latent factors for regions and geographical features, respectively. Likewise, matrix Y can be factorized into the multiplication of two matrices, $Y = T \times R^T$, where $T \in \mathbb{R}^{L \times d_T}$ is a low rank latent factor matrix for time slots. d_T and d_R are usually very small. The objective function is defined as Equation 20:

$$\mathcal{L}(S,R,C,T,U) = \frac{1}{2}||\mathcal{A} - S \times_R R \times_C C \times_T T||^2 + \frac{\lambda_1}{2}||X - RU||^2 + \frac{\lambda_2}{2}\mathrm{tr}(C^T L_Z C) + \frac{\lambda_3}{2}||Y - TR^T||^2 + \frac{\lambda_4}{2}(||S||^2 + ||R||^2 + ||C||^2 + ||T||^2 + ||U||^2); \quad (20)$$

where $\|\mathcal{A} - S \times_R R \times_C C \times_T T\|^2$ is to control the error of decomposing $\mathcal{A}$; $\|X - RU\|^2$ is to control the error of factorization of X; $\|Y - TR^T\|^2$ is to control the error of factorization of Y; $\|S\|^2 + \|R\|^2 + \|C\|^2 + \|T\|^2 + \|U\|^2$ is a regularization penalty to avoid over-fitting; λ_1, λ_2, λ_3, and λ_4 are parameters controlling the contribution of each part during the collaborative decomposition. Here, matrix X and Y share the same dimension of region with tensor $\mathcal{A}$. Tensor $\mathcal{A}$ has a common dimension of time with Y and a shared dimension of category with Z. Thus, they share latent spaces for region, time and category. This idea has been introduced in the coupled-matrix factorization. $\mathrm{tr}(C^T L_Z C)$ is derived from Equation 19 of the manifold alignment:

$$\sum_{i,j}\|C(i,.) - C(j,.)\|^2 Z_{ij} = \sum_k \sum_{i,j}\|C(i,k) - C(j,k)\|^2 Z_{ij}$$
$$= tr(C^T(D - Z)C) = tr(C^T L_Z C), \quad (21)$$

where $C \in \mathbb{R}^{M \times d_C}$ is the latent space of category; $D_{ii} = \sum_i Z_{ij}$ is a diagonal matrix, and $L_Z = D - Z$ is the Laplacian matrix of the category correlation graph. $tr(C^T L_Z C)$, which guarantees two (e.g. the ith and jth) noise categories with a higher similarity (i.e., Z_{ij} is bigger) should also have a closer distance in the new latent space C. In this case, only one data set, i.e. 311 data, is involved in the manifold alignment. So, $\mathbf{D} = D$. As there is no closed-form solution for finding the global optimal result of the objective function (shown in Equation 20), a numeric method, gradient descent, is employed to find a local optimization.

5.3. Probabilistic Dependency-Based Fusion

A probabilistic graphical model is a probabilistic model for which a graph expresses the conditional dependence structure between random variables. Generally, it uses a graph-based representation as the foundation for encoding a complete distribution over a multi-dimensional space. The graph can be regarded as a compact or factorized representation of a set of independences that hold in the specific distribution. Two branches of graphical representations of distributions are commonly used, namely, Bayesian Networks and Markov Networks (also called Markov Random Field [34]). Both families encompass the properties of factorization and independences, but they differ in the set of independences they can encode and the factorization of the distribution that they induce [9]. For instance, a Bayesian Network is a directed acyclic graph that factorizes the joint probability of n variables $X_1, X_2, \ldots, X_n$ as $P[X_1, X_2, \ldots, X_n] = \prod_{i=1}^{n} P[X_i|PA(X_i)]$. Markov Network is a set of random variables having a Markov property described by an undirected graph, which may be cyclic. Thus, a Markov network can represent certain dependencies that a Bayesian network cannot (such as cyclic dependencies). On the other hand, it cannot represent certain dependencies that a Bayesian network can (such as induced dependencies).

This category of approaches bridges the gap between different datasets by the probabilistic dependency, which emphasize more about the interaction rather than the similarity between two objects. This is different from the similarity-based methods introduced in Section 5. For example, variables (i.e. features extracted from different datasets) are represented by nodes, and the probabilistic dependency (or causality) between two different variables is denoted by the edge connecting them. The structure of a graphical model can be learned from data automatically or pre-defined by human knowledge. Graphical models usually contain hidden variables to be inferred. The learning process of graphical models is to estimate the probabilistic dependency between different variables given the observed data. Expectation and Maximization (EM) algorithms are commonly used methods. The inference process is to predict the status of hidden variables, given the values of observed variables and learned parameters. The inference algorithms include deterministic algorithms, such as variational methods, and stochastic algorithms like Gibbs Sampling. More details about graphical models can be referred to [9][36].

Example 13: Shang et al. [53] propose inferring traffic volume on a road based on POIs, road networks, travel speed and weather. Fig. 14 presents the graphical structure of the traffic volume inference (TVI) model, where a gray node denotes a hidden variable and white nodes are observations. One TVI model is trained for each level of road segments.

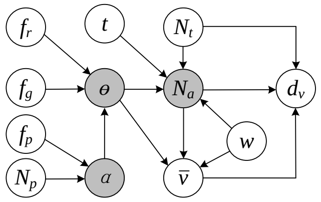


Fig. 14. The graphical structure of TVI model

Specifically, the traffic volume on each road lane N_a (i.e., the number of vehicles per minute per lane) of a road segment is influenced by four major factors, consisting of the weather conditions w, time of day t, the type of road θ, and the volume of observed sample vehicles N_t. Furthermore, a road's θ is co-determined by its road network features f_r (such as $r.len$), global position feature f_g, and surrounding POIs α which is influenced by f_p and the total number of POIs N_p. $\bar{v}$ and dv are the average travel speed and speed variance, respectively, inferred by TSE model. $\bar{v}$ is determined by θ, N_a, and w. dv is co-determined by N_t, N_a, and $\bar{v}$. Due to the hidden nodes, the conditional probability of N_a cannot be drawn simply by counting the occurrence of each condition. An EM algorithm was proposed to learn the parameters in an unsupervised manner.

Example 14: Yuan et al. [74][76] infer the functional regions in a city using road network data, points of interests, and human mobility learned from a large number of taxi trips. As depicted in Fig. 15, a LDA (Latent Dirichlet Allocation)-variant-based inference model was proposed, regarding a region as a document, a function as a topic, categories of POIs (e.g., restaurants and shopping malls) as metadata (like authors, affiliations, and key words), and human mobility patterns as words. The mobility pattern is defined as the commuting patterns between regions. That is when people leave a region and where they are heading to, and when people arrive at a region and where did there come from. Each commuting pattern stands for one word describing a region, while the frequency of the pattern denotes the number of occurrence of a word in a document.

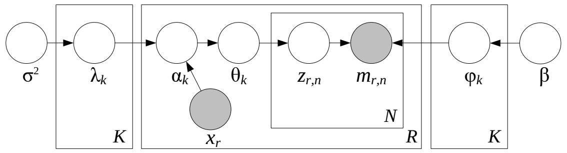
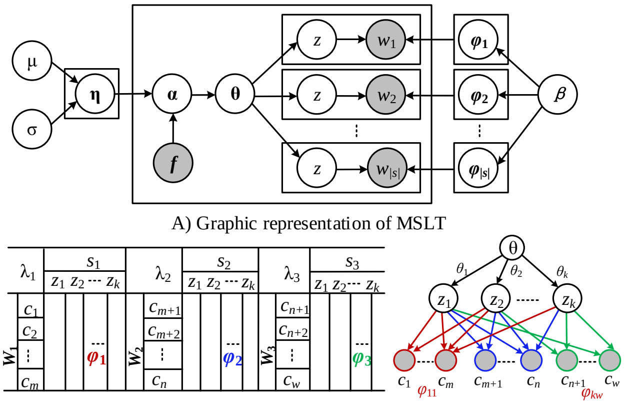


Fig. 15. Learning functional regions based on multiple datasets: A graphical model approach

By feeding POIs (denoted as x_r) and human mobility patterns (denoted as $m_{r,n}$) into different parts of this model, a region is represented by a distribution of functions, each of which is further denoted by a distribution of mobility patterns. N stands for the number of words (i.e., mobility patterns in a region); R denotes the number of documents (regions); K is the number of topics, which should be predefined. Before running the model, a city was partitioned into disjointed regions using major roads like high ways and ring roads. So, this example also uses stage-based data fusion techniques. This model can be estimated using EM and inferred using Gibbs sampling. Different from the basic LDA model [5], the Dirichlet prior is now specified for individual regions based on the observed POI features of each region.

Example 15: Zheng et al. [90] combine multiple datasets generated in a region to better estimate the distribution of a sparse dataset in the region and the latent function of the region, based on the following two insights. First, different datasets in a region can mutually reinforce each other. Second, a dataset can reference across different regions. Fig. 16 depicts the graphical presentation of the model, entitled MSLT (Multi-Source Latent Topic Model).



A) Graphic representation of MSLT

B) Topic-words distribution across different data sources

Fig.16. The graphic presentation of the MSLT model

$\boldsymbol{f}$ is a vector storing the features extracted from the road network and POIs located in a region. $\boldsymbol{\eta} \in \mathbb{R}^{k\times|\boldsymbol{f}|}$ is a matrix with each row $\boldsymbol{\eta}_t$ corresponding to a latent topic; k denotes the number of topics and $|\boldsymbol{f}|$ means the number of elements in $\boldsymbol{f}$. The value of each entry in $\boldsymbol{\eta}$ follows a Gaussian distribution with a mean μ and a standard deviation σ. $\boldsymbol{\alpha} \in \mathbb{R}^k$ is a parameter of the Dirichlet prior on the per-region topic distributions. $\boldsymbol{\theta} \in \mathbb{R}^k$ is the topic distribution for region d. $\boldsymbol{\mathcal{W}} = \{\boldsymbol{W}_1, \boldsymbol{W}_2, \dots, \boldsymbol{W}_{|\boldsymbol{S}|}\}$ is a collection of word sets, where $\boldsymbol{W}_i$ is a word set corresponding to dataset s_i and $|\boldsymbol{S}|$ denotes the number of datasets involved in the *MSLT*. $\boldsymbol{\beta} \in \mathbb{R}^{|\boldsymbol{W}_i|}$ is the parameter of the Dirichlet prior on the per-topic word distributions of $\boldsymbol{W_i}$. A word w in $\boldsymbol{W}_i$ is one of the categories s_i's instances pertain to, e.g. $\boldsymbol{W}_1 = \{c_1, c_2, \dots, c_m\}$.

As illustrated in Fig. 16 B), different datasets share the same distribution of topics controlled by $\boldsymbol{\theta}_d$, but having its own topic-word distributions $\boldsymbol{\varphi}_i$, $1 \le i \le |\boldsymbol{S}|$, indicated by arrows with different colors. $\boldsymbol{\varphi}_{iz}$ is a vector denoting the word distribution of topic z in word set $\boldsymbol{W}_i$. This is different from LDA and its variant *DMR* [44], which have a single word set and topic-word distribution. The topic distribution $\boldsymbol{\theta}_d$ of a region and the topic-word distribution $\boldsymbol{\varphi}_i$ of a dataset s_i are used to calculate the underlying distribution of each category in s_i, if s_i is very sparse, e.g. $prop(w_i) = \sum_t \theta_{dt}\varphi_{tw_i}$.

5.4. Transfer Learning-Based Data Fusion

A major assumption in many machine learning and data mining algorithms is that the training and future data must be in the same feature space and have the same distribution. However, in many real-world applications, this assumption may not hold. For example, we sometimes have a classification task in one domain of interest, but we only have sufficient training data in another domain of interest, where the latter data may be in a different feature space or follow a different data distribution. Different from semi-supervised learning, which assumes that the distributions of the labeled and unlabeled data are the same, transfer learning, in contrast, allows the domains, tasks, and distributions used in training and testing to be different.

In the real world, we observe many examples of transfer learning. For instance, learning to recognize tables may help recognize chairs. Learning riding a bike may help riding a moto-cycle. Such examples are also widely witnessed in the digital world. For instance, by analyzing a user's transaction records in Amazon, we can diagnose their interests, which may be transferred into another application of travel recommendation. The knowledge learned from one city's traffic data may be transferred to another city.

5.4.1. Transfer between the Same Type of Datasets

Pan and Yang et al. [50] present a good survey that classifies transfer learning into three categories, based on different tasks and situations between the source and target domains, as shown in Table 2. Fig. 17 presents another taxonomy of transfer learning according to whether label data are available in source and target domains.

Transductive learning is proposed to handle the cases where the task is the same but the source and target domain are different. Furthermore, there are two sub-categories of the difference between source and target domains. In the *first* category, the feature spaces between domains are the same, but the marginal probability distributions are different. Most of the existing works on transfer learning fall into this category. For example, in a traffic prediction task, we can transfer the traffic data from a city to another one where training data is limited. In the *second* category, the feature spaces between domains are different. For example, one domain has Chinese web pages; the other have English web pages. But, the task is the same, i.e. clustering web pages according to the similarity of their semantic meanings. Yang et.al [72] initiate the setting called "heterogeneous transfer learning" to handle this category of situation. There are two directions in this stream of work: 1) translation from the source to the target [18] or 2) projection both domains into a common latent space [72][93]. Although the source and the target domains are from different feature spaces in heterogeneous transfer learning, each domain itself is homogeneous with a single data source.

Table 2. Taxonomy of Transfer Learning (TL) [50]

Learning settings		Source and target domains	Source and target tasks
Traditional ML		the same	the same
TL	Inductive learning / unsupervised TL	the same	different but related
		different but related	different but related
	Transductive learning	different but related	the same

Different from Transductive Learning, inductive learning handles learning cases where the tasks are different in source and target domains. It focuses on storing knowledge gained while solving one problem and applying it to a different but related problem. Multi-task learning (MTL) [3] is a representative approach to inductive transfer learning. MTL learns a problem together with other related problems at the same time, using a shared representation. This often leads to a better model for the main task, because it allows the learner to use the commonality among the tasks [14]. MTL works well if these tasks have some commonality and are slightly under sampled. Fig. 17 presents two examples of MTL.

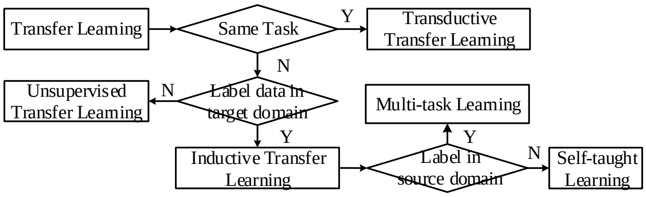

Transfer Learning
Same Task
Y
Transductive Transfer Learning
N
Unsupervised Transfer Learning
N
Label data in target domain
Y
Multi-task Learning
Y
Inductive Transfer Learning
Label in source domain
N
Self-taught Learning

Fig. 17. A different setting of transfer learning

Example 16: Fig. 18 A) illustrates the transferring learning between two classification tasks. One task is to infer an individual's interests in different travel packages in terms of her location history in the physical world (e.g. check-ins from a social networking service). The other task is to estimate a user's interests in different book styles based on the books the user has browsed on the Internet. If we happen to have the two datasets from a same user, we can associate the two tasks in a MTL framework, which learns a shared representation of a user's general interests. As the books a user has browsed may imply her general interests and characters, which can be transferred into the travel package recommendation. Likewise, the knowledge from a user's physical location can also help estimate a user's interests in different book styles. MTL is particularly helpful when the dataset we have is sparse; e.g. we only have a small amount of check-in data of a user.

Example 17: Fig. 18 B) presents another example of MTL, which co-predicts the air quality and traffic conditions at near future simultaneously. The general insight is that different traffic conditions will generate different volumes of air pollutants, therefore impacting the air quality differently. Likewise, people tend to go hiking or picnic in a day with good air quality, while preferring to minimize travel in a day with hazardous air quality. As a result, the traffic conditions are also affected by air quality. The shared feature representation of the two datasets can be regarded as the latent space of a location in a time slot.

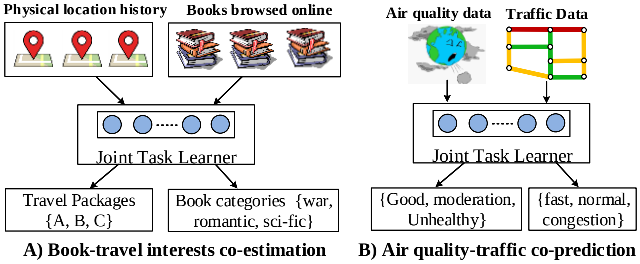

Physical location history
Books browsed online
Joint Task Learner
Travel Packages {A, B, C}
Book categories {war, romantic, sci-fic}
A) Book-travel interests co-estimation
Air quality data
Traffic Data
Joint Task Learner
{Good, moderation, Unhealthy}
{fast, normal, congestion}
B) Air quality-traffic co-prediction

Fig. 18. Examples of multi-task transfer learning

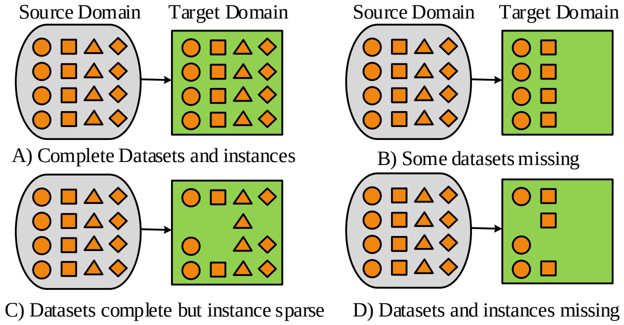
5.4.2 Transfer Learning among Multiple Datasets

In the big data era, many machine learning tasks have to harness a diversity of data in a domain so as to achieve a better performance. This calls for new techniques that can transfer the knowledge of multiple datasets from a source to a target domain. For example, a major city like Beijing may have sufficient datasets (such as traffic, meteorological, and human mobility etc.) to infer its fine-grained air quality. But, when applying the model to another city, we may not have some kind of datasets (e.g. traffic) at all or not enough observations in some datasets (e.g. human mobility). Can we transfer the knowledge learned from multiple datasets of Beijing to another city?

Fig. 19 presents the four situations of transfer learning when dealing with multiple datasets, where different shapes denote different datasets (a.k.a. views). As depicted in Fig. 19 A), a target domain has all kinds of datasets (that the source domain has), each of which has sufficient observations (as the source domain). That is, the target domain has the same (and sufficient) feature spaces as the source domain. This kind of situation can be handled by multi-view transfer learning [19][71][77]. For example, Zhang et al. [77] propose a multi-view transfer learning with a large margin approach (MVTL-LM), which leverages both labeled data from the source domain and features from different views. DISMUTE [19] performs feature selection for multi-view cross-domain learning. Multi-view discriminant transfer (MDT) [71] learns discriminant weight vectors for each view to minimize the domain discrepancy and the view disagreement simultaneously.

As shown in Fig. 19 B), some datasets do not exist in the target domain, while other datasets are as sufficient as the source domain. To deal with such kind of dataset (a.k.a. view structural) missing problems, a line of research on multi-view multi-task learning [26][32] has been proposed. However, these algorithms cannot handle the situations shown in Fig. 19. C), where a target domain has all kinds of datasets but some datasets may have very few observations (or say very sparse), and the situation presented in

Fig. 19 D), where some datasets do not exist (view missing) and some datasets are very sparse (observation missing). The problem is still open for research.



A) Complete Datasets and instances

B) Some datasets missing

C) Datasets complete but instance sparse

D) Datasets and instances missing

Fig. 19. Transferring knowledge from multiple datasets

6. Discussion

It is hard to judge which data fusion method is the best, as different methods behave differently in different applications. Table 3 presents a comparison among these data fusion methods (list in the first column), where the second column (Meta) indicates if a method can incorporate other approaches as a meta method. For instance, a semantic meaning-based data fusion methods can be employed in a stage-based fusion method. To select a proper data fusion method, we need to consider the following factors:

Table 3. Comparison of different data fusion methods

Methods		Meta	Labels		Goals	Train	Scale
			Vol	Pos			
Stage-based		Y	NA	NA	NA	NA	NA
F	Direct	N	L	Flex	F,P,C,O	S	Y
	DNN	N	L	Flex	F,P,A,O	U/S	Y
Semantic	Multiview	Y	S	Fix	F,P,O	S, SS	Y
	Probabil.	N	S	Fix	F,P,C,O,A	S/U	N
	Similarity	N	S	Flex	F,A,O	U	Y
	Transfer	Y	S	Fix	F,P,A	S/U	Y

1) The volume, properties and insight of datasets that we have in an application. *First*, as shown in the third column (Vol) of Table 3, the feature-based data fusion methods need a *large* (L) amount of labeled instances as training data, while the semantic meaning-based methods can be applied to a dataset with a *small* (S) number of labeled instances. *Second*, when studying a type of objects, e.g. geographical regions, we need to consider whether there are some object instances that can constantly generate labeled data (titled “Fixed” or “Flexible” in the 4th column “Position”). For example, we can have *fixed* monitoring stations constantly generating air quality data in some regions in Example 7. On the contrary, we cannot ensure that there are 311 complaints (mentioned in Example 12) constantly reported by people in a region. Sometimes region *A* and B have 311 complaints, while at other time intervals Region *C*, *D* and *E* have. In some extreme cases, there are no regions with 311 data. That is, regions with 311 data occur *flexibly*, which cannot formulate stable view-class label pairs for a region. As a consequence, it is not appropriate to use a multi-view-based fusion method to handle the 311 example. On the other hand, the former problem cannot be solved by a similarity-based fusion method either. As the location of a station is fixed, regions with and without labels are both fixed. We cannot calculate the similarity between a region with labeled data and another always without data. *Third*, some datasets do not change over time while others are temporally dynamic. Directly combining features extracted from static data and those from dynamic data would cause static features ignored by a machine learning model. For example, road networks and POIs around a building do not change over time, no matter what level of air pollution is over the building. Thus, the feature-based fusion methods do not work well for this example.

2) The goal, learning approach, and requirement of a machine learning and data mining task. *First*, goals of fusing multiple datasets includes Filling Missing Values (of a sparse dataset) [53][85][87] [88], Predict Future [89], Causality Inference, Object Profiling[73][76][74], and Anomaly Detection [15][90][49], etc. As presented in the fifth column, probabilistic dependency-based data fusion methods can achieve all these goals (F, P, C, O, A). Particularly, Bayesian Networks and the straightforward feature-based fusion methods (e.g. when using a linear regression model [24]) are usually good at dealing with causality inference problems (C). The directed edges of a Bayesian Network reveal the causality between different factors (i.e. nodes), and the weight of a feature in a linear regression model denotes the importance of a factor to a problem. As raw features have been converted into a middle-level feature representation by DNN, the semantic meaning of each feature is not clear any more. *Second*, the learning methods consist of supervised (S), unsupervised (U) and semi-supervised (SS) learning, as denoted in the sixth column. For example, supervised and semi-supervised learning approaches can be applied to multi-view-based data fusion methods. *Third*, there are some requirements for a data mining task, such as efficiency and scalability (shown in the most right column). Generally speaking, it is not easy for probabilistic dependency-based approaches to scale up (N). A graphical model with a complex structure, e.g. many (hidden) nodes and layers, may become intractable. With respect to the similarity-based data fusion methods, when a matrix becomes very large, NMF, which can be operated in parallel, can be employed to expedite decomposition (Y).

Generally, given the same amount of training data, the straightforward feature-based methods are not as good as semantic meaning-based approaches, as there are dependencies and correlations between features. Adding a sparsity regularization can alleviate the problem to some extents, but cannot solve it fundamentally. In some cases with a large amount of labeled data, particularly for images and speech data, feature-based fusion using DNNs can perform well. However, the performance of the model heavily relies on tuning parameters. Given a huge model with many parameters to learn, this is usually a time-consuming process that needs the involvement of human experiences. Additionally, there are a few overlaps between the multi-view-based approach and transfer learning. For instance, there are multi-view multi-task learning approaches [29].

7. Public Cross-Domain Big Data

7.1 Urban Big Data

Advances in sensing technology and large-scale computing infrastructure have been generating a diversity of data in cities. Quite a few cities, like New York City and Chicago, have opened their datasets to the public. Here are some links to the open datasets:

- NYC Open Data: https://data.cityofnewyork.us/.
- Chicago Open Data: https://data.cityofchicago.org/
- Urban Computing in Microsoft Research: http://research.microsoft.com/en-us/projects/urbancomputing/default.aspx [84].
- Urban Noise: 311 complaint data with social media, POIs and road networks in New York City. http://research.microsoft.com/pubs/217236/Context%20matrices%20and%20tensor%20data.zip [87].
- Urban Air: air quality data with meteorological data and weather forecasts in 5 Chinese cities [31][89][85]. http://research.microsoft.com/apps/pubs/?id=246398.
- Traffic speed+POIs+Road network: Features extracted from three datasets in Beijing have been accommodated in three matrices, used in *Example 11* [53]. https://onedrive.live.com/?cid=cf159105855090c5&id=CF159105855090C5%212774&ithint=file,.zip&authkey=!AFBIXgrChcesYC4. By adding a user dimension into the data, a tensor is built to describe the travel time of a particular user on a particular road at a specific time slot. The data was used in [65] and can be download from the following URL: http://research.microsoft.com/apps/pubs/?id=217493

7.2 Images/Videos and Texts

- Image + Text: (2,866 image-text pairs from Wikipedia): http://www.svcl.ucsd.edu/projects/crossmodal/
- Image + Text (1 million images with captions): http://vision.cs.stonybrook.edu/~vicente/sbucaptions/
- Video + Text (about 2,000 video snippets; about 40 sentences per snippet) http://research.microsoft.com/en-us/downloads/38cf15fd-b8df-477e-a4e4-a4680caa75af/
- Microsoft COCO dataset: More than 300,000 images, each of which has 5 captions. http://mscoco.org/.
- TACoS multilevel dataset: 44,762 video-sentence pairs. https://www.mpi-inf.mpg.de/departments/computer-vision-and-multimodal-computing/research/human-activity-recognition/mpii-cooking-activities-dataset/.
- Flickr30K dataset: 31,783 images, 5 captions per image. https://illinois.edu/fb/sec/229675.
- MMDB dataset: 160 sessions of 5-minute interaction from 121 children, including all multi-modal signals: images by camera, audio by microphones, activity/accelerometry sensors. http://cbi.gatech.edu/mmdb/overview.php

8. Conclusion

The proliferation of big data calls for advanced data fusion methods that can discover knowledge from multiple disparate datasets with underlying connections. This paper summarizes existing data fusion methods, classifying them into three major categories and giving real examples in each sub-category of methods. This paper explores the relationship and differences between different methods, helping people find proper data fusion methods to solve big data problems. Some public multi-modality datasets have been shared to facilitate further research into data fusion problems.

Acknowledgement

Thanks Ying Wei who helped survey a part of the DNN-based and the transfer learning-based data fusion methods. Thanks the co-authors (such as, Jing Yuan, Vincent W. Zheng, Tong Liu, Yanjie Fu, Yanchi Liu, Yilun Wang, Jingbo Shang, Wenzhu Tong, and Wei Liu) of my publications that have been referenced in the survey.

References

[1] S.Abney, “Bootstrapping,” Proc. of the 40th Annual Meeting of the Association for Computational Linguistics (ACL), pages 360–367, 2002.

[2] J. Bao, Y. Zheng, and M. F. Mokbel, “Location-based and Preference-Aware Recommendation Using Sparse Geo-Social Networking Data,” *Proc. ACM SIGSPATIAL Conf. Advances in Geographic Information Systems (GIS'12)*, pp. 199-208, 2012.

[3] J. Baxter, "A model of inductive bias learning," Journal of Artificial Intelligence Research, 12, pp.149–198, 2000

[4] C. Bishop. Pattern Recognition and Machine Learning, Springer 2006.

[5] D. Blei, A. Ng, and M. Jordan. Latent dirichlet allocation. The Journal of Machine Learning Research, 3:993–1022, 2003.

[6] M.F. Balcan, A. Blum, and Y. Ke, “Co-training and expansion: Towards bridging theory and practice,” *In Advances in neural information processing systems*, pp. 89-96. 2004.

[7] M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. Neural Computation, 15, 2003.

[8] Y. Bengio, A. Courville, and P. Vincent, “Representation learning: A review and new perspectives,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, No. 8, pp. 1798-1828, 2013.

[9] Christopher M. Bishop, "Chapter 8. Graphical Models," Pattern Recognition and Machine Learning. Springer. pp. 359–422, 2006.

[10] J. Bleiholder and F. Naumann. Data fusion. *ACM Computing Surveys*, vol. 41, No. 1, pp. 1–41, 2008.

[11] A. Blum and T. Mitchell, “Combining labeled and unlabeled data with co-training,” *Proc. of the eleventh annual conference on Computational learning theory*, pp. 92–100, 1998.

[12] U. Brefeld and T. Scheffer, “Co-em support vector learning,” *Proc. of the twenty-first international conference on Machine learning*, pp. 16, 2004.

[13] L. Breiman, "Random Forests," Machine Learning 45 (1): 5–32. 2001.

[14] R. Caruana, "Multitask learning: A knowledge-based source of inductive bias," Machine Learning, 28, pp.41–75, 1997.

[15] S. Chawla, Y. Zheng, and J. Hu, “Inferring the root cause in road traffic anomalies,” *Proc. of International Conference on Data Mining (ICDM'12)*, pp. 141-150, 2012.

[16] N. Chen, J. Zhu, and E.P. Xing, “Predictive subspace learning for multiview data: A large margin approach,” Advances in Neural Information Processing Systems, 24, 2010.

[17] G. E. Dahl, D. Yu, L. Deng, and A. Acero, “Context-Dependent Pre-Trained Deep Neural Networks for Large Vocabulary Speech Recognition,” IEEE Trans. Audio, Speech, and Language Processing, vol. 20, no. 1, pp. 33-42, Jan. 2012.

[18] W. Dai, Y. Chen, G.-R. Xue, Q. Yang, and Y. Yu. Translated learning: Transfer learning across different feature spaces. In NIPS, pages 353–360, 2008.

[19] Z. Fang and Z. M. Zhang, "Discriminative feature selection for multi-view cross-domain learning," *Proc. of International Conference on Information and Knowledge Management (CIKM 2013*), pp. 1321–1330, 2013.

[20] Y. Fu, Y. Ge, Y. Zheng, Z. Yao, Y. Liu, H. Xiong, N. Jing Yuan. Sparse Real Estate Ranking with Online User Reviews and Offline Moving Behaviors. *Proc. of IEEE International Conference on Data Mining (ICDM'14)*, pp. 120-129, 2014.

[21] D. Goldberg, N. David, M. O. Brain, T. Douglas, "Using collaborative filtering to weave an information tapestry," *Communications of the ACM*, vol. 35, no. 12, pp. 61–70, 1992.

[22] G. H. Golub and C. Reinsch, "Singular value decomposition and least squares solutions," *Numerische Mathematik*, vol. 14, no. 5, pp.403-420. 1970.

[23] M. Gonen and E. Alpaydn, "Multiple kernel learning algorithms," *Journal of Machine Learning Research*, 12 pp.2211–2268, 2011.

[24] C. W. Granger, "Investigating causal relations by econometric models and cross-spectral methods," *Econometrica: Journal of the Econometric Society*, pp. 424-438, 1969.

[25] D. Hardoon, S. Szedmak and J. Shawe-Taylor, "Canonical correlation analysis: An overview with application to learning methods," Neural computation, 16, 12, 2639-2664. 2004.

[26] J. He and R. Lawrence, "A graph-based framework for multi-task multi-view learning," *Proc. of International Conference on Machine Learning*, pp. 25–32, 2011.

[27] T.K. Ho, "Random Decision Forest," Proc. of the 3rd International Conference on Document Analysis and Recognition, pp. 278–282, 1995.

[28] T.K. Ho, "The random subspace method for constructing decision forests," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8), 832-844, 1998.

[29] Z. Hong, X. Mei, D. Prokhorov, D. Tao, Tracking via robust multi-task multi-view joint sparse representation. *Proc. IEEE International Conference on Computer Vision* (ICCV'13), pp. 649-656. 2013.

[30] P. O. Hoyer, "Non-negative matrix factorization with sparseness constraints," *The Journal of Machine Learning Research*, 5, pp. 1457-1469, 2004.

[31] H. P. Hsieh, S. D. Lin, Y. Zheng. Inferring Air Quality for Station Location Recommendation Based on Big Data. *Proc.the 21th SIGKDD conference on Knowledge Discovery and Data Mining*, pp.437-446, 2015.

[32] X. Jin, F. Zhuang, H. Xiong, C. Du, P. Luo, and Q. He. Multi-task multi-view learning for heterogeneous tasks. In CIKM, pages 441–450, 2014.

[33] M. Kan, S. Shan, H. Zhang, S. Lao and X. Chen, "Multi-view discriminant analysis," *Proc. of the 12th European Conference on Computer Vision*, pp. 808-821, 2012.

[34] R. Kindermann and J. L. Snell, "Markov random fields and their applications," Vol. 1. Providence, RI: American Mathematical Society, 1980.

[35] V. Klema and A. J. Laub, "The singular value decomposition: Its computation and some applications," Automatic Control, IEEE Transactions on, vol. 25, no. 2, pp.164-176, 1980.

[36] D. Koller, N. Friedman. Probabilistic Graphical Models. Massachusetts: MIT Press. p. 1208, 2009.

[37] A. Krizhevsky, I. Sutskever, and G. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," *Proc. Neural Information and Processing Systems*, 2012.

[38] P. L. Lai and C. Fyfe, "Kernel and nonlinear canonical correlation analysis," *International Journal of Neural Systems*, vol. 10, no. 5, 365-377, 2000.

[39] N. D. Lawrence, "Gaussian process latent variable models for visualisation of high dimensional data," Advances in neural information processing systems, 16, pp.329–336, 2004.

[40] Y. LeCun and M. Ranzato, "Deep learning tutorial," *In Tutorials in International Conference on Machine Learning*, 2013.

[41] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," *Proc. of Advances in neural information processing systems*, pp. 556-562, 2011.

[42] D. Lemire and A. Maclachlan, "Slope One: Predictors for Online Rating-Based Collaborative Filtering," *Proc. of SIAM Data Mining*. vol. 5, pp. 1-5. 2005.

[43] W. Liu, Y. Zheng, S. Chawla, J. Yuan and X. Xie, "Discovering Spatio-Temporal Causal Interactions in Traffic Data Streams," *Proc. ACM SIGKDD Conf. Knowledge Discovery and Data Mining (KDD'11)*, pp. 1010-1018, 2011.

[44] D. Mimno and A. McCallum. Topic models conditioned on arbitrary features with dirichlet-multinomial regression. *Uncertainty in Artificial Intelligence*, pages 411–418, 2008.

[45] A. Nakamura and N. Abe, "Collaborative Filtering Using Weighted Majority Prediction Algorithms," *Proc. of the 15th International Conference on Machine Learning*, pp. 395-403, 1998.

[46] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng, "Multimodal deep learning, " *Proc. the 28th International Conference on Machine Learning*, pp. 689-696, 2011.

[47] K. Nigam and R. Ghani, "Analyzing the effectiveness and applicability of co-training," Proc. of the ninth international conference on Information and knowledge management, pp. 86–93, 2000.

[48] W. S. Noble, "Support vector machine applications in computational biology," In Kernel Methods in Computational Biology, Bernhard Sch¨olkopf, Koji Tsuda and Jean-Philippe Vert, editors, chapter 3. The MIT Press, 2004.

[49] B. Pan, Y. Zheng, D. Wilkie and C. Shahabi, "Crowd Sensing of Traffic Anomalies based on Human Mobility and Social Media," *Proc. ACM SIGSPATIAL Conf. Advances in Geographic Information Systems (GIS'13),* pp. 334-343, 2013.

[50] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transaction on Knowledge Discovery and Data Engineering*, 22, 10, pp. 1345 – 1359, 2010.

[51] S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. Science, 290, 2000.

[52] K. Ryan, R. Salakhutdinov, and R. Zemel, "Multimodal neural language models," Proc. of the 31st International Conference on Machine Learning, pp. 595-603, 2014.

[53] J. Shang, Y. Zheng, W. Tong, E. Chang, and Y. Yu, "Inferring Gas Consumption and Pollution Emission of Vehicles throughout a City," *Proc. ACM SIGKDD Conf. Knowledge Discovery and Data Mining (KDD'14),* pp. 1027-1036, 2014.

[54] A. P. Singh and G. J. Gordon, "Relational learning via collective matrix factorization," In Proc. of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 650-658, 2008.

[55] R. Socher, A. Karpathy, Q. V. Le, C. D. Manning, and A. Y. Ng, "Grounded compositional semantics for finding and describing images with sentences," Transactions of the Association for Computational Linguistics, 2, 207-218, 2014.

[56] N. Srivastava, R. Salakhutdinov, "Multimodal Learning with Deep Boltzmann Machines," *Proc. Neural Information and Processing Systems*, 2012.

[57] Y. Sun and J. Han, "Mining heterogeneous information networks: principles and methodologies," *Synthesis Lectures on Data Mining and Knowledge Discovery*, vol. 3, no. 2, pp.1-159. 2012.

[58] Y. Sun, J. Han, P. Zhao, Z. Yin, H. Cheng, and T. Wu, "Rankclus: integrating clustering with ranking for heterogeneous information network analysis," *Proc the 12th International Conference on Extending Database Technology: Advances in Database Technology*, pp. 565-576, 2009.

[59] Y. Sun, Y. Yu, and J. Han, "Ranking-based clustering of heterogeneous information networks with star network schema," *Proc the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp.797-806, 2009.

[60] J. Tenenbaum, Vin de Silva, and J. Langford. A global geometric framework for non-linear dimensionality reduction. Science, 290, 2000.

[61] M. E. Tipping, Sparse Bayesian learning and the relevance vector machine, *The journal of machine learning research* 1 (2001): 211-244.

[62] D. G. Tzikas, L. Wei, A. Likas, Y. Yang, and N. P. Galatsanos, "A tutorial on relevance vector machines for regression and classification with applications," *University of Ioannina, Ioanni, GREECE, Illinois Institute of Technology, Chicago*, USA, 2006.

[63] M. Varma and B.R. Babu, "More generality in efficient multiple kernel learning," *Proc. of the 26th Annual International Conference on Machine Learning*, pp. 1065–1072, 2009.

[64] C. Wang, P. Krafft and S. Mahadevan, "Manifold alignment," In Manifold Learning: Theory and Applications, Eds. Yunqian Ma and Yun Fu, CRC press, 2011.

[65] Y. Wang, Y. Zheng, and Y. Xue, "Travel Time Estimation of a Path using Sparse Trajectories," *Proc. ACM SIGKDD Conf. Knowledge Discovery and Data Mining (KDD'14)*, pp. 25-34, 2014.

[66] Z. Wang, D. Zhang, X. Zhou, D. Yang, Z.Yu, and Z. Yu, Discovering and profiling overlapping communities in location-based social networks. Systems, *IEEE Transactions on Man, and Cybernetics: Systems*, vol. 44, no. 4, pp. 499-509, 2014.

[67] X. Xiao, Y. Zheng, Q. Luo, and X. Xie, "Finding Similar Users Using Category-Based Location History," *Proc. ACM SIGSPATIAL Conf. Advances in Geographic Information Systems (GIS'10)*, pp. 442-445, 2010.

[68] X. Xiao, Y. Zheng, Q. Luo, and X. Xie, "Inferring Social Ties between Users with Human Location History," *J. of Ambient Intelligence and Humanized Computing*, vol. 5, no. 1, pp. 3-19, 2014.

[69] C. Xu, T. Dacheng, and X. Chao, "A survey on multi-view learning," arXiv preprint arXiv: 1304.5634 (2013).

[70] D. Yang, D. Zhang, Z. Yu, and Z. Yu, Fine-grained preference-aware location search leveraging crowdsourced digital footprints from LBSNs. *Proc. ACM international joint conference on Pervasive and ubiquitous computing* (UbiComp'13), pp. 479-488, 2013.

[71] P. Yang and W. Gao, "Multi-view discriminant transfer learning," Proc. of International Joint Conference on Artificial Intelligence, pp. 1848–1854, 2013.

[72] Q. Yang, Y. Chen, G.-R. Xue, W. Dai, and Y. Yu, "Heterogeneous transfer learning for image clustering via the social web," Proc. of ACL, pp. 1–9, 2009.

[73] N. J. Yuan, F. Zhang, D. Lian, K. Zheng, S. Yu, and X. Xie, "We know how you live: exploring the spectrum of urban lifestyles," *Proc. the first ACM Conference on Online* Social Networks, pp. 3-14, 2013.

[74] J.Yuan, Y. Zheng, X. Xie, "Discovering regions of different functions in a city using human mobility and POIs," *Proc. ACM SIGKDD Conf. Knowledge Discovery and Data Mining (KDD'12)*, pp. 186-194, 2012.

[75] N. J. Yuan, Y. Zheng, and X. Xie, "Segmentation of Urban Areas Using Road Networks," Technical Report MSR-TR-2012-65, 2012.

[76] N. J. Yuan, Y. Zheng, X. Xie, Y. Wang, K. Zheng and H. Xiong, "Discovering Urban Functional Zones Using Latent Activity Trajectories," *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 3, pp. 1041-4347, 2015

[77] D. Zhang, J. He, Y. Liu, L. Si, and R. Lawrence. Multi-view transfer learning with a large margin approach. *Proc. of the 17th SIGKDD conference on Knowledge Discovery and Data Mining*, pp. 1208–1216, 2011.

[78] F. Zhang, N. J. Yuan, D. Wilkie, Y. Zheng, X. Xie, "Sensing the Pulse of Urban Refueling Behavior: A Perspective from Taxi Mobility", *ACM Trans. Intelligent Systems and Technology*, submitted for publication.

[79] V. W. Zheng, B. Cao, Y. Zheng, X. Xie, Q. Yang, "Collaborative Filtering Meets Mobile Recommendation: A User-centered Approach," *Proc. AAAI Conf. Artificial Intelligence (AAAI'10)*, pp. 236-241, 2010.

[80] V. W. Zheng, Y. Zheng, X. Xie, Q. Yang, "Collaborative Location and Activity Recommendations with GPS History Data," *Proc. International Conf. World Wide Web (WWW'10)*, pp. 1029-1038, 2010.

[81] V. W. Zheng, Y. Zheng, X. Xie, Q. Yang, "Towards Mobile Intelligence: Learning from GPS History Data for Collaborative Recommendation," *Artificial Intelligence Journal*, pp.17-37, 2012.

[82] Y. Zheng. Trajectory Data Mining: An Overview. ACM Transactions on Intelligent Systems and Technology, vol. 6, issue 3, pp. 1-29, 2015.

[83] Y. Zheng, X. Chen, Q. Jin, Y. Chen, X. Qu, X. Liu, E. Chang, W.-Y. Ma, Y. Rui, W. Sun, "A Cloud-Based Knowledge Discovery System for Monitoring Fine-Grained Air Quality," MSR-TR-2014-40, 2013.

[84] Y. Zheng, L. Capra, O. Wolfson, H. Yang. "Urban Computing: Concepts, Methodologies, and Applications," *ACM Trans. Intelligent Systems and Technology*, vol. 5, no. 3, pp. 38-55, 2014.

[85] Y. Zheng, F. Liu, H.P. Hsieh, "U-Air: When Urban Air Quality Inference Meets Big Data," *Proc. of the 19th SIGKDD Conference on Knowledge Discovery and Data Mining*. pp. 1436-1444, 2013.

[86] Y. Zheng, Y. Liu, J. Yuan, X. Xie, "Urban Computing with Taxicabs," *Proc. ACM Conf. Ubiquitous Computing (UbiComp'11)*, pp.89-98, 2011.

[87] Y. Zheng, T. Liu, Y. Wang, Y. Zhu, Y. Liu, E. Chang, "Diagnosing New York City's Noises with Ubiquitous Data," *Proc. ACM Conf. Ubiquitous Computing (UbiComp'14)*, pp. 715-725, 2014.

[88] Y. Zheng, X. Xie. Learning travel recommendations from user-generated GPS traces. *ACM Transaction on Intelligent Systems and Technology*, vol. 2, no. 1, 2-19. 2011.

[89] Y. Zheng, X. Yi, M. Li, R. Li, Z. Shan, E. Chang, T. Li, Forecasting Fine-Grained Air Quality Based on Big Data. *Proc. ACM SIGKDD Conf. Knowledge Discovery and Data Mining (KDD'15)*, 2015.

[90] Y. Zheng, H. Zhang, Y. Yu, Detecting Collective Anomalies from Multiple Spatio-Temporal Datasets across Different Domains. Submitted to ACM SIGSPATIAL 2015.

[91] Y. Zheng, L. Zhang, Z. Ma, X. Xie, W.-Y. Ma, "Recommending friends and locations based on individual location history," *ACM Trans. the Web*, vol 5, no. 1, pp.5-44, 2011.

[92] Z. H. Zhou and M. Li, "Semi-supervised regression with co-training," *Pro. Of the International Joint Conference on Artificial Intelligence* (IJCAI), 2005.

[93] Y. Zhu, Y. Chen, Z. Lu, S. J. Pan, G.-R. Xue, Y. Yu, and Q. Yang. Heterogeneous transfer learning for image classification. In AAAI, 2011.

Yu Zheng is a lead researcher from Microsoft Research and a Chair Professor at Shanghai Jiao Tong University. His research interests include big data analytics, spatio-temporal data mining, and ubiquitous computing. He leads the research on urban computing in Microsoft Research, passionate about using big data to tackle urban challenges. He frequently publishes referred papers as a leading author at prestigious conferences and journals, such as *KDD*, *VLDB*, *UbiComp*, and *IEEE TKDE*. He received five best paper awards from *ICDE'13* and *ACM SIGSPATIAL'10*, etc. Zheng currently serves as an Editor-in-Chief of *ACM Transactions on Intelligent Systems and Technology* and as a member of Editorial Advisory Board of *IEEE Spectrum*. He is also an Editorial Board Member of *GeoInformatica* and *IEEE Transactions on Big Data*. He has served as chair on over 10 well-known international conferences, e.g. the program co-chair of ICDE 2014 (Industrial Track) and UIC 2014. He has been invited to give over 10 keynote speeches at international conferences and forums (e.g. IE 2014 and APEC 2014 Smart City Forum). He is an IEEE senior member and ACM senior member. In 2013, he was named one of the Top Innovators under 35 by MIT Technology Review (TR35) and featured by Time Magazine for his research on urban computing. In 2014, he was named one of the top 40 Business Elites under 40 in China by Fortune Magazine, because of the business impact of urban computing he has been advocating. Zheng is also an Adjunct Professor at Hong Kong Polytechnic University, a visiting Chair Professor at XiDian University and an Affiliate Professor at Southwest Jiaotong University.