Robust Location Understanding in Spoken Dialog Systems Using Intersections

Michael L. Seltzer, Yun-Cheng Ju, Ivan Tashev, and Alex Acero

Speech Technology Group, Microsoft Research, Redmond, WA USA
{mseltzer, yuncj, ivantash, alexac}@microsoft.com

Abstract

The availability of digital maps and mapping software has led to significant growth in location-based software and services. To safely use these applications in mobile and automotive scenarios, users must be able to input precise locations using speech. In this paper, we propose a novel method for location understanding based on spoken intersections. The proposed approach utilizes a rich, automatically-generated grammar for street names that maps all street name variations into a single canonical semantic representation. This representation is then transformed to a sequence of position-dependent subword units. This sequence is used by a classifier based on the vector space model to reliably recognize spoken intersections in the presence of recognition errors and incomplete street names. The efficacy of the proposed approach is demonstrated using data collected from users of a deployed spoken dialog system.

Index Terms: location-based spoken dialog systems, spoken address recognition, vector space model

1. Introduction

The widespread availability of digital maps and mapping software has generated significant growth in location-based software and services such as route planning, navigation, and locating nearby businesses, such as restaurants, e.g. [1]. Recently, such offerings have started moving from desktop computers to mobile devices and embedded computers. In these applications, the ability for the user to easily input locations is critical [2]. Because these devices have small screens and are often used in hands-busy/eyes-busy environments, e.g. while driving, speech is a safe and natural way to do so.

Locations can be conveyed by the user in several ways. For example, a business or point of interest can indicate a location if the corresponding address is known. However, this only works for unique entities, and not for chain businesses or residences. For example, in Seattle, WA asking for directions to "Starbucks" is highly ambiguous, as there are many Starbucks cafés in the city, and often more than one on a single street.

The full street address, on the other hand, always corresponds to a unique location. Unfortunately, using the street address is difficult in practice because of recognition errors, especially with a single recognition pass This was confirmed in [3], where an iterative multi-pass approach using a class-based language model was proposed for spoken addresses. The difficulty is even more apparent when one considers that state-of-the-art recognition accuracy for a five digit number in noise conditions that are realistic for mobile scenarios is about 90% [4]. This means that one out of ten house numbers or zip codes will be misrecognized. In such cases, the disambiguation strategies to correct these errors must often resort to tedious digit-by-digit confirmation.

In this paper, we propose the use of intersections as a way for users to convey their exact location to a spoken dialog system. Intersections have many advantages over conventional methods of location entry. They are a highly natural and are often used to convey location in human-human communication. In addition, it is easier to determine the nearest intersection than the nearest valid address while driving. Because an intersection consists of two streets, there are a limited number of ways an intersection can be misrecognized: either one of the two streets is wrong, or both are wrong. This makes disambiguation potentially much simpler compared to a street address. Finally, if intersections can be recognized reliably, users can uniquely identity an otherwise ambiguous point of interest with a reference intersection, such as "Starbucks on the corner of Pine Street and 3rd".

Recognizing intersections reliably is a challenging problem. In major cities, there can be thousands of street names and many more intersections. For example, in the city of Seattle, there over 3500 unique street names and over 20,000 intersections. In addition, streets and intersections are often spoken informally, with incomplete specifications and a variety of pronunciations.

In this work, we propose a novel method for understanding spoken intersections. In this approach, the user's speech is recognized using a rich, automatically generated probabilistic context free grammar (PCFG) for street names that maps all pronunciation variations of a street name to a single canonical representation during recognition. This representation is then expanded using a positition-dependent phonetic tokenization. This tokenization enables the intersection classifier, which is based on the vector space model, to accurately classify an intersection despite the presence of recognition errors and incomplete street names. The proposed approach only requires a single recognition pass and works with any recognition engine. We verify the validity of our approach and compare it to other methods using data collected from users of a spoken dialog application that recognizes intersections in Washingon state.

2. Generating a grammar for street names

There is a high degree of variability in the way people refer to street names. For example, users may use a partial street name, either out of familiarity or convenience, e.g. "the gas station on 148th," or because they do not know or remember the complete name or order of the terms, e.g. "5th Ave S" vs. "S 5th Ave".

In order to recognize these entities robustly, we first create a rich PCFG that captures the variety of ways people refer to streets. We start with a geographic database that contains all street names and intersections in a particular city. The street names follow conventional address abbreviations, such as standard street suffixes, e.g. St, Ave, Ln, and compass directions, e.g. N, S, NE, SW. Each full street name in the database is first parsed into a sequence of entities using regular expressions

Table 1: Street name entity labels

Street Entity	Examples
<InterstatePrefix>	I, US
<HighwayPrefix>	Hwy, SR, Route
<CardinalAlpha>	9A
<AlphaCardinal>	A101,
<Cardinal>	101, 5, 90
<Ordinal>	148th, 3rd
<Direction>	N, S, NE, SW
<StreetType>	Ave, Ln, St, Ct
<TextName>	Main, Ashbury

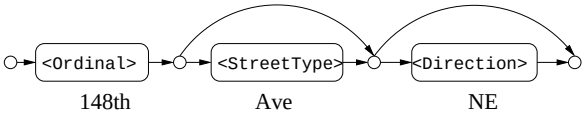

Figure 1: Grammar generating graph for "148th Ave NE"

based on hand-crafted rules. A list of the entities and examples are shown in Table 1.

Once this labeling is performed, all possible variations of the full street name are enumerated using a graph-based representation. The graph is constructed such that all paths in the graph constitute a valid pronunciation of that street name. A valid street name must include a Cardinal, Ordinal, AlphaCardinal, CardinalAlpha, or TextName, while all other labels such as Direction, StreetSuffix, and HighwayPrefix can be skipped. An example graph for "148th Ave NE" is shown in Figure 1.

Once all paths of the graphs are enumerated, the utterance corresponding to each path through the graph is extracted, text normalized, and added to the grammar. For each valid path through the grammar, additional utterances are also generated for alternate pronunciations and common prefix substitutions. This frequently occurs for ordinal street names, e.g. "140th" can be pronounced "one hundred and fortieth", "one fortieth", "a hundred fortieth", etc. and for highways, where a variety of prefixes, such as "I, interstate, or highway" are common. Each variation is mapped to a common semantic identity that represents the path taken through the word graph. This semantic identity is used as the semantic tag in a W3C standardized SRGS grammar [5]. The benefit of the proposed PCFG is that a large variety of possible street name pronunciations are collasped down to a concise canonical representation.

Because the grammar is quite large, adding prior probabilities to the entries in the grammar can significantly improve recognition accuracy over a simple uniform distribution. In this work, we set the priors for streets based on the number of intersections of which it is a member. This idea has intuitive appeal, as long, busy streets will have many intersections, while smaller neighborhood streets will have significantly fewer. Once this grammar is generated, duplicate entries are counted and aggregated and their weights are combined. For example, both "148th Ave NE" and "NE 148th Pl" will generate entries in the grammar as "148th", so this entry's weight is proportion to the sum of the weights of all streets with the root ordinal "148th". This increases the prior probability of a root ordinal which is shared across several street names.

In order to recognize naturally spoken intersections, the street PCFG is embedded in a context free grammar that captures typical ways of specifying an intersection, such as "I'm on the corner of <Street> and <Street>". This intersection CFG is then combined with a domain-independent n-gram filler model to generate a hybrid CFG/n-gram grammar which is much more robust than a standalone CFG to variations in the grammar [6].

3. Robust Recognition of Intersections

Once the user's utterance is recognized and parsed, we have a recognition hypothesis (and corresponding semantic tag) for each of the streets in the intersection. However, there is a high likelihood that user uttered only a fragment of the complete street name. In addition, because of the high acoustic confusability among many street names, especially numeric streets, there is a good chance that recognition errors will occur. In this section, we address both of these issues.

3.1. Vector space model using TF-IDF

The street name classification is performed using a vector space model (VSM). The VSM was originally proposed for information retrieval of text documents [7]. In this model, each document d_i in a collection $\mathcal{D}$ is represented as a vector $\mathbf{v}_i$, whose components represent the importance of particular terms in the document. The most commonly used values for the components in these vectors are based on Term Frequency-Inverse Document Frequency (TF-IDF).

The TF-IDF score is composed to two terms, the term frequency (TF), which is the ratio of the number of occurances a word to the total number of words in the document, and inverse document frequency (IDF), which is the ratio of the total number of documents in a collection to the number of documents containing that word. Thus, for word w_i in document d_j, we compute TF-IDF as

$$v_{ij} = TF_{ij} \cdot \log(IDF_i) = \frac{N_{ij}}{N_j} \cdot \log\left(\frac{M}{M_i}\right) \qquad (1)$$

where N_{ij} is the number occurances of w_i in d_j, N_j is the total number of words in d_j, M is the number of documents in the collection $\mathcal{D}$ and M_i is the number of documents that contain w_i.[1]

The TF-IDF scores can be computed for each document for all terms (words) that appear in the document collection. Thus, each document d_j is represenented by a vector $\mathbf{v}_j = [v_{0j}, v_{1j}, \ldots, v_{Kj}]$ where K is the number of unique words present in $\mathcal{D}$. The vector space model then computes the similarity of two documents as the cosine of the angle between their corresponding vectors, as

$$S(d_i, d_j) = \cos\theta = \frac{\mathbf{v}_i \cdot \mathbf{v}_j}{\| \mathbf{v}_i \| \| \mathbf{v}_j \|} \qquad (2)$$

If the two documents are identical, then $S(d_i, d_j) = 1$ and if the documents share no terms, then $S(d_i, d_j) = 0$. When a query q is made, it is treated as a document, and the appropriate vector $\mathbf{v}_q$ is created. Documents similar to the query can then be identified.

In our system, streets are considered documents with words corresponding to the entities that comprise a particular street's full name. For example, if the street names were used direclty in the VSM, "148th Ave NE" would be represented by a vector with non-zero TF-IDF scores for the terms "148th", "Ave", and

[1] Note that this is one of many variants of TF-IDF that have been proposed [8].

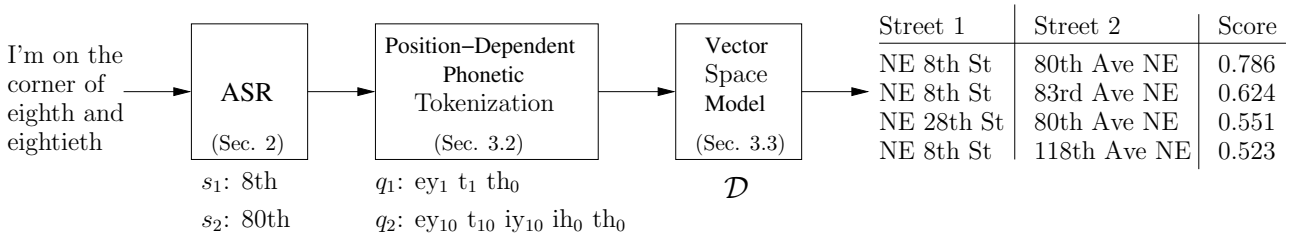

Figure 2: The components of the intersection understanding system. The speech is recognized by the street and intersection grammar, which maps the user's spoken words to canonical semantic tags. The tags are then tokenized into a sequence of position-dependent phonemes which are then used to query the intersection database $\mathcal{D}$ using the vector space model.

Table 2: Three different representation of ordinal numbers.

Ordinal	Word	Position Dependent Phonemes
30th	thirtieth	th$_{10}$ er$_{10}$ t$_{10}$ iy$_{10}$ ih$_0$ th$_0$
38th	thirty eighth	th$_{10}$ er$_{10}$ t$_{10}$ iy$_{10}$ ey$_1$ t$_1$ th$_0$
13th	thirteenth	th$_{10}$ er$_{10}$ t$_{10}$ iy$_{10}$ n$_1$ th$_0$

"NE". This model can find candidate street names when only a partial street name is spoken by the user.

3.2. Position-dependent phonetic tokenization of numeric streets

The high acoustic confusability among street names will cause recognition errors. To enable the system to recover from some misrecognitions we propose a novel tokenization scheme for numeric street names. We have focused on these streets in particular as we have observed empirically, that these are often the streets with the highest acoustic confusability.

In the proposed tokenization scheme, each numeric street name in the database is represented by a sequence of phonemes based on a standard speech recognition dictionary. In addition, each phoneme in the sequence is labeled with the position of its parent digit, where "100" marks the hundreds place, "10" marks the tens place, and "1" marks the ones place. The phonemes of ordinal suffixes (st, nd, rd, th) are labeled with "0". For example, the "one" in "one hundred" is represented as "w$_{100}$ ah$_{100}$ n$_{100}$", while "eighth" is transformed to "ey$_1$ t$_1$ th$_0$".

There are several advantages to this tokenization scheme. It enables the hypothesized numeric strings to be decomposed into a series of elements such that recognition errors remain local to the incorrect subword units. Also, by augmenting the phonemes with position of the corresponding digit, sequence information is preserved, which allows the downstream classification to separate digits that are acoustically identical but semantically different, such as the three in "300" and the three in "23". Table 2 compares the proposed position-dependent phonetic tokenization to two other common representations for three acoustically confusable entities. Note how the phonetic similarity of the three entities is captured in the proposed scheme but is missing from the ordinal and word-based representations. The sequence information is also retained, as the leading "th" is represented differently than the trailing "th".

3.3. Intersection recognition using the vector space model

We now describe how the streets and intersections grammars described in Section 2, and the VSM and the position dependent phonetic tokenization scheme described in this section are combined to understand the user's spoken intersection. A block diagram of the complete system is shown in Figure 3.

First, the user's utterance is recognized and parsed, and the semantic identities of the two hypothesized streets or street fragments are generated. The semantic tag for the first street entity s_1 is then tokenized as described in Section 3.2 to create a street query q_1. The vector of TF-IDF scores is created for this query and the VSM is then used to compare the similarity of q_1 to the first street of all intersections in the database. This results in a ranked list of candidate intersections, based on the hypothesized first street s_1.

The semantic tag of the second street s_2 is then transformed to a query q_2 and the vector space model is then used to compute the similarity between s_2 and all second streets in the just-generated list of candidate intersections. The overall score for an intersection is then computed as the product of the VSM scores of the two streets in the intersection. Thus, the score for intersection I is

$$I(s_1 = d_i, s_2 = d_j) = S(q_1, d_i) \cdot S(q_2, d_j) \qquad (3)$$

where $S(q_1, d_i)$ and $S(q_2, d_j)$ are computed according to (2)

In order to reduce redundency in our database, all intersections are represented once with an arbitrary street ordering. If if a user refers to an intersection with the opposite ordering, it will not match in our database. As a result, the above procedure for intersection search is performed both with the original ordering and with the queries q_1 and q_2 swapped. Thus, the final hypothesized intersection is

$$I(s_1 = d_i, s_2 = d_j) = \underset{i,j}{\text{argmax}} \ \{S(q_1, d_i) \cdot S(q_2, d_j), S(q_2, d_i) \cdot S(q_1, d_j)\} \qquad (4)$$

The top scoring intersection or a ranked list of candidate intersections is then returned to the user. We have found empirically, that in densely populated areas, many users believe a location to be in particular city while it is actually in a neighboring city. To account for this, the intersections for all bordering cities are also included in the search, with a fixed penalty applied to their VSM scores to reflect the fact that they are outside the user's specified search area. The value of this penalty is tuned using development data.

Table 3: Street name documents for "NE 138th St" for word-level, semantic, and the proposed phonetic representations

Terms	Street name document
Words	north east one hundred and thirty eighth street
Semantics	NE 138th St
Pos-dep Phones	NE w_{100} ah_{100} n_{100} th_{10} er_{10} t_{10} iy_{10} ey_1 t_1 th_0 St

4. Experiments

To evaluate the proposed method for intersection recognition, we developed a telephone-based spoken dialog system that tries to identify any intersection in Washington state. The system works as follows. The caller is first asked for the city they are interested in. Once the city is identified, the streets grammar described in Section 2 is loaded. This grammar contains the streets in the user's desired city plus the streets in all neighboringing cities. The user is then asked to tell the system what intersection they are on. The user's response is recognized, parsed, and the semantic identities of the two hypothesized streets or street fragments are obtained. These semantic tags are then tokenized and processed by the vector space model. The top scoring intersection is then returned to the user for confirmation.

We collected a total of 104 intersection dialogs from 22 distinct users. Once the data was collected, we compared the proposed approach to the two other methods of representation. In the the first method, the streets were represented in the database by their text transcriptions. In this case, the semantic tags are ignored, and the VSM then computes the similarity between word-level TF-IDF vectors and query vectors that are derived from the recognized text for each street. In the second method, the original full street names themselves are used in the database and the queries are generated from the semantic tags associated with the recognized text. Table 3 shows example street name documents for "NE 138th St" corresponding to each of the three representations evaluated in these experiments.

The results of the experiment are shown in Figure 3, with the performance obtained using the highest scoring intersection in black and the performance using the top four intersections shown in white. As the figure shows, the proposed approach results in intersection error rate of 8.6% for the single best hypothesis. This is a substantial performance improvement over the other more conventional methods based on a VSM applied to the recognized text or the semantic tags. The figure also shows that using semantic tags reduces the variability in the queries of the street names and results in improved performance over a word-based approach. Expanding the semantic tags to position dependent phonemes enables the classification to recover from some misrecognitions caused by acoustically confusable street names. The error-recovery benefits are further evident from the results obtained when the top four intersections are returned. The error rate for the proposed system is reduced 22% relative when the top four intersections are returned. The resulting error rate of 6.7% is a 30% relative improvement over the top four intersections hypothesized by the other two approaches.

5. Conclusion

In this paper we have proposed a novel method for understanding spoken intersections as a means of conveying location. Recognition is performed using an automatically generated PCFG for street name pronunciations that maps all pronunciations of a street name to a single semantic identity. This semantic tag is then expanded to a string of position-dependent phonemes that preserves the sequence information of the phonetic units. The intersection classification is done using a vector space model with TF-IDF features derived from the position-dependent phonetic tokens, as well as the other terms such as street type and direction. The proposed approach requires only a single recognition pass and results in an intersection error rate of 8.6%. When the vector space model returns the top four intersections, the error rate drops to 6.7%, highlighting the system's ability to recover from recognition errors.

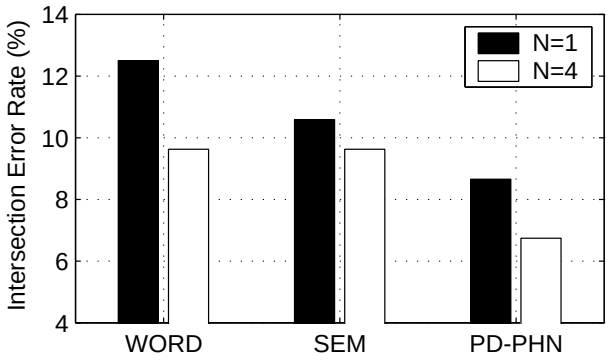

Figure 3: The intersection error rate using word level features (WORD), semantic features (SEM), and position-dependent phonetic features (PD-PHN). The last column shows the performance of PD-PHN using the top 4 candidates.

6. References

[1] A. Gruenstein, S. Seneff, and C. Wang, "Scalable and portable web-based multimodal dialog interaction with geographical databases," in *Proc. of Interspeech*, Pittsburgh, PA, Sep 2006.

[2] O. Tsimhoni, D. Smith, and P. Green, "Address entry while driving: speech recognition vs. a touch-screen keyboard," *Human Factors*, vol. 46, no. 4, pp. 600–610, Winter 2004.

[3] A. Venkataraman, H. Franco, and G. Myers, "An architecture for rapid retrieval of structured information using speech with application to spoken address recognition," in *Proc. of ASRU*, St. Thomas, USVI, Dec 2003.

[4] D. Macho, L. Mauuary, B. Noe, Y. M. Cheng, D. Ealey, D. Jouvet, H. Kelleher, D. Pearce, and F. Saadoun, "Evaluation of a noise-robust DSR front-end on Aurora databases," in *Proc. ICSLP*, Denver, USA, Sep 2002, pp. 17–20.

[5] A. Hunt and S. McGlashan, *Speech Recognition Grammar Specification Version 1.0*, 2002.

[6] D. Yu, Y. C. Ju, Y. Wang, and A. Acero, "N-gram based filler model for robust grammar authoring," in *Proc. of ICASSP*, Toulouse, France, May 2006.

[7] G. Salton, A. Wong, and C. S. Yang, "A vector space model for automatic indexing," *Communications of the ACM*, vol. 18, 1975.

[8] M. Lan, C.-L. Tan, H.-B. Low, and S.-Y. Sung, "A comprehensive comparative study on term weighting schemes for text categorization with support vector machines," in *Proc. WWW*, Chiba, Japan, 2005, pp. 1032–1033.