Automatic Construction of Unique Signatures and Confusable Sets for Natural Language Directory Assistance Applications

E.E. Jan, Benoît Maison, Lidia Mangu and Geoffrey Zweig

IBM T. J. Watson Research Center
Yorktown Heights, NY, 10598
{ejan,bmaison,mangu,gzweig}@us.ibm.com

Abstract

This paper addresses the problem of building natural language based grammars and language models for directory assistance applications that use automatic speech recognition. As input, one is given an electronic version of a standard phone book, and the output is a grammar or language model that will accept all the ways in which one might ask for a particular listing. We focus primarily on the problem of processing listings for businesses and government offices, but our techniques can be used to speech-enable other kinds of large listings (like book titles, catalog entries, etc.). We have applied these techniques to the business listings of a state in the Midwestern United States, and we present highly encouraging recognition results.

1. Introduction

Over the past few years, automated directory assistance has emerged as an important application of speech recognition technology [2, 5, 3] - In 1995, telephone companies spent over $1.5B providing directory assistance service. It typically takes an operator about 25 seconds to complete a directory assistance call, and thus a reduction of just one second in this average work time represents a savings of over 60M a year [2].

A typical call flow for a directory assistance application is as follows: 1. first, the service type is identified, e.g. local directory assistance, nation-wide directory assistance, or reverse directory assistance. 2. second, the locality is identified. This involves city-name recognition for local directory assistance, and a combination of city & state name recognition for national DA. 3. third, the particular listing requested must be identified, in the context of the stated locality. Listing recognition includes business listing, government listing and residential name listing. In the case of reverse directory assistance, a digit recognizer is required to recognize the telephone number and return the associated name and address.

One of the most important characteristics of automated DA service is that it is necessary to maintain an extremely low false acceptance rate (approximately 1%). If customers are given incorrect information, service quality and customer satisfaction are degraded. However, since each call that is handled automatically frees up an operator to handle something else, even if just 20 or 30% of the calls can be handled automatically, overall costs can be reduced by an equal amount.

The task of automatically recognizing what a customer wants is not an easy one, especially for business and government listings, where the full listings tend to be long and complicated. This is the case because there are many ways of asking for a given listing, even for relatively simple entries. Suppose, for example, that a town has a beauty salon called “3-L Hair World” (located on North 3rd Street). A person might ask for this in many ways, e.g. “3-L,” “Hair World,” or “Hair World on 3rd Street,” and it is not clear exactly what to expect. Moreover, the database might also include entries for other beauty salons, such as “Susie’s Hair World” (located on Main Street). In this case, if a person asks for “Hair World” there would be an ambiguity – no listing would be uniquely identified.

This is a simple example, and in reality, for large businesses and government departments, the problem can be much worse. In these cases, specific numbers are often specified by a long chain of specializations, for example the “Westchester County Department of Social Services Field Operations Home Care Services (at 270 North Avenue in New Rochelle).” Numerous parts of this will clearly be omitted in any real request, but at the same time enough information must be given to distinguish it from the “Westchester County Department of Social Services Field Operations Medical Transportation” number.

A rule based approach has been proposed to generate business listing grammars in [4]. In this approach, the business listing are separated into different categories, and rules are made for each category. In contrast, in this work, we would like to to be able to take an electronic representation of a phone book, and without spending too much effort on writing rules, produce a system for automatically associating speech utterances with phone numbers. When the customer uniquely specifies a listing, we would of course like to return that number, but further, if an utterance is ambiguous and there are a reasonably small number of potential matches, we would also like to identify this set. In order to do this, one of two basic strategies can be used. The first strategy is to use an N-gram-based language model with a large-vocabulary recognizer to transcribe customer statements, and then to post-process the output to determine if there is enough information to return a phone number. In the second strategy, one can build a grammar such that the associated annotations carry the information to return to the customer. This second approach has the advantage that all the processing required to determine under what conditions a number can be returned is done at “compile time,s” before the utterances are spoken. Thus, the amount of computation at runtime is minimized - the utterance is run through a grammar, and either a number is returned, or the set of possible matching numbers is identified.

The contribution of this paper is to suggest a way of solving these two basic problems that arise in constructing directory assistance grammars:

1) Precomputing all the possible ways that a person might uniquely specify a listing, and

2) Precomputing the sets of listings that match non-unique partial specifications.

To continue with the earlier example, the first part of the problem is to identify utterances that uniquely identify a listing - e.g. “3-L,” “3-L Hair World,” “Susie’s,” “Susie’s Hair World,” “Hair World on 3rd Street,” or “Hair World on Main Street” Each of these unambiguously identifies a single listing.

The second part of the problem is to identify phrases that might refer to several listings. For example, the phrase “Hair Care” could refer to either 3-L or Susie’s. If a user says “Hair Care”, the automated system can say “There are two listings that match, 3-L and Susie’s. Which would you like?”

2. Grammar Construction Algorithms

In the following, we use the term “signature” to refer to a subsequence of the words in a listing that uniquely identifies it. In the previous example, “3-L”, “Hair 3rd”, and “Hair Main” are signatures. These subsequences of words are signatures because they occur in exactly one listing. Note that the words in a signature need not be consecutive in the listing.

The problem of identifying all the signatures that uniquely identify a listing is solved with the following steps:

a) Enumerate all possible n-word subsequences of words in a listing. Typically, n is 3.

b) For each subsequence, create a record containing the subsequence and the index of the listing it came from.

c) Sort the records into lexicographical order based on the subsequences.

d) Scan linearly over the records for cases where the associated subsequence does not occur in either the preceding or following record. Such cases indicate unique signatures.

In order to create a grammar that will identify uniquely specified listings, perform the following additional step: For each unique signature, add an entry to the grammar. This entry duplicates the original listing, except that all words not in the signature are made optional.

These steps are illustrated in Table 1 for the example of “3-L Hair World on North 3rd Street,” (database record 1) and “Suzie’s Hair World on Main Street” (database record 2). For clarity, two-word subsequences have been used in this example, and not all of them have been enumerated. A full enumeration would require more space than is available. Note that after Step c), the records are lexicographically sorted, and all those with “Hair World” in the first fields have been brought together. By scanning down the records at this point, it becomes apparent that “Hair World” occurs in multiple listings and therefore is not unique. However, “3-L Hair” occurs in just one listing and is unique.

In Step a) the subsequences of words were constrained to occur in the same order as in the original listing. (For example, “3-L Hair” is a subsequence, but “Hair 3-L” is not). It is anticipated that in some applications, it will be beneficial to allow the subsequences of words taken in any order, but in the examples presented here, it is not the case.

Table. 2 illustrates the grammar that results from performing the final step of our procedure. In this representation, question marks are used to indicate optional words. The figure indicates that a sentence (denoted by <S>::=) can be realized in one of the eight ways indicated on the lines below. Each of these lines specifies a phrase in which some of the words are optional, and after the colon, each specifies the database record that should be identified. The lines are separated by vertical bars (|), and the grammar is terminated by a period.

Table 1: *Flow Diagram for construction unique signature.*

<3-L Hair, 1> <3-L World, 1> <3-L North, 1> <Hair World, 1> <Hair North, 1> <Suzie’s Hair, 2> <Suzie’s World, 2> <Hair World, 2> <Hair Main, 2> <World Main, 2> ...	Data structure after Steps a) and b)
⇓	
<3-L Hair, 1> <3-L North, 1> <3-L World, 1> <Hair Main, 2> <Hair North, 1> <Hair World, 1> <Hair World, 2> <Suzie’s Hair, 2> <Suzie’s World, 2> <World Main, 2> ...	Data structure after sorting in Step c)
⇓	
<3-L Hair, 1> <3-L North, 1> <3-L World, 1> <Hair Main, 2> <Hair North, 1> <Suzie’s Hair, 2> <Suzie’s World, 2> <World Main, 2> ...	Unique Signatures identified in Step d), and stored in sorted order.

2.1. Unique Signatures in a language model approach

In fact, the basic strategy we describe can be used in conjunction with an N-gram language model as well. In this case, the database of Step c) is used to identify records with the following steps:

a) Enumerate the possible n-word subsequences of the recognized text.

b) For each of these subsequences, do a binary search on the sorted records to see if the subsequence if listed as a unique signature.

c) If all the entries found in Step b) are associated with a unique listing in the database, then an unambiguous match has been made.

From the example of Table 1 after the data structure of Step d), when the given text is “Give me 3-L Hair”, the subsequence “ 3-L Hair” is the only match, and record 1 is identified. However, if the input spoken token is “Give me Suzie’s 3-L Hair World”, the subsequence of “3-L Hair” matches record 1, but subsequence “Suzie’s Hair” matches record 2. There is ambiguity and unique match is impossible.

2.2. Generation of confusable sets

The problem of precomputing the sets of records that match non-unique partial specifications is solved with the following

Table 2: *Sample grammar constructed from unique signatures.*

```
<S> ::=
3-L Hair World? On? North? 3rd ? Street? : 1    |
3-L Hair? World? On? North 3rd ? Street? : 1    |
3-L Hair? World on? North? 3rd ? Street? : 1    |
3-L? Hair World? On? North 3rd ? Street? : 1    |
Suzie's? Hair World? On? Main Street? : 2       |
Suzie's Hair World? On? Main? Street? :2        |
Suzie's Hair? World on? Main? Street? :2        |
Suzie's? Hair? World on? Main Street? :2        .
```

steps:

a) Enumerate all possible n-word subsequences in a listing. If desired, sub-strings (i.e. *consecutive* words) can be used instead of subsequences to reduce the number of possibilities. The sub-string approach was used in our implementation. Typically n is 3.

b) For each substring, create a record containing the substring and the index of the listing it came from.

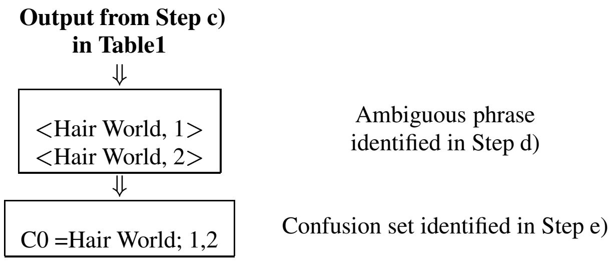
c) Sort the records into lexicographical order on the substring.

d) Scan linearly over the records for cases where a single substring occurs multiple times.

e) For each such case, group the associated listings into a confusion set.

Steps a) through c) are the same as Steps a) through c) in the process for finding unique signatures. In Step d), however, multiple records with the same lexical key are identified. In Step e), a confusion set is created for each lexical key identified in Step d). This process is illustrated in Table3.

Table 3: *Flow Diagram for construction confusion sets.*


Output from Step c) in Table1
<Hair World, 1>
<Hair World, 2>
Ambiguous phrase identified in Step d)
C0 =Hair World; 1,2
Confusion set identified in Step e)

2.3. Grammar generation with confusion sets

If a grammar is being produced, the following additional step is performed: For each confusion set, add an entry to the grammar that recognizes the associated lexical key, and which returns the value of the confusion set when it is recognized.

This step is illustrated in Table 4.

In the case that the text being matched comes from a Language Model based speech recognizer, if that text is an exact match to the lexical key of a confusion set, then the confusion set is identified.

2.4. Confidence-Based Extension

Some speech-recognition based applications, like directory assistance, can only tolerate a very low error rate. The traditional way to ensure that the user is very rarely given incorrect information is to set rejection thresholds very high. However, the overall automation rate suffers because a large number of correctly recognized requests are rejected along with the incorrectly recognized ones. Often, the confidence score of the entire utterance is not high enough because just a few words score low. Then the entire utterance is discarded. To improve this, we capitalize on the observation that even a small high confidence portion of the user utterance can often uniquely identify a specific request. This approach applies to both the language model-based recognition (section 2.1), and to the grammar-based approach (section 2.3), as a post-recognition step.

Table 4: *Sample grammar of Table 1 with confusion set added.*

```
<S> ::=
3-L Hair World? On? North? 3rd ? Street? : 1    |
3-L Hair? World? On? North 3rd ? Street? : 1    |
3-L Hair? World on? North? 3rd ? Street? : 1    |
3-L? Hair World? On? North 3rd ? Street? : 1    |
Suzie's? Hair World? On? Main Street? : 2       |
Suzie's Hair World? On? Main? Street? :2        |
Suzie's Hair? World on? Main? Street? :2        |
Suzie's? Hair? World on? Main Street? :2        |
Hair World :c0                                  .
```

The basic idea of the confidence based extension is to remove low-confidence words from recognized text, and to use the remaining text to match against the database. A detailed description of the process follows.

The recognized utterance consists of a sequence of words (w_i) with confidence scores (s_i). The entire sentence has its own confidence score (s). In the case of grammar-based decoding, s may not meet the minimum value for the recognition result to be accepted. The following procedure, illustrated with an example, will make the best possible use of the recognized text, either to initiate a disambiguation dialog or to identify the user request. Let us assume that the recognized words (with scores between 0 and 100) are: "Patrick's (20) Furniture (90) Restoration (80) on (50) twenty-third (40) street (95)" The procedure works by setting a minimum word-level confidence threshold t, and discarding all words i with score $s_i < t$. For example, if t θ= 3 , we obtain the word sequence: "Furniture (90) Restoration (80) on (50) twenty-third (40) street (95)". The matching algorithm described in Table 1 is used to find business names that contain those five words in the same order. If a unique match is found, the procedure stops, the listing has been identified (only one furniture restoration business on 23rd street). Otherwise, if the number of matches is more than one, the procedure stops, and a dialog is started to further disambiguate between the matches. If the number of matches is zero, the confidence threshold is set to a lower value and the procedure is repeated. If t reaches a preset minimum value $t = t_{min}$ with still no matches, a conventional utterance rejection procedure is used, e.g. the call is transferred to an operator or the user is prompted to repeat his query.

3. Experiment design and Results

3.1. IBM LVCSR System

The latest IBM telephony product, Websphere Voice Server, which is a HMM-based stack decoder, is used to conduct our studies[1]. The system uses MFCC-based front-end signal processing for a 39 dimensional feature space. Words are represented as sequences of phones. Each phone is modeled by a three-state HMM. For each sub-phonetic unit, a decision tree is constructed from the training data and the terminal nodes of the tree represent collections of instances of these classes grouped according to context. These context-dependent leaves are modeled by a mixture of Gaussian pdf's with diagonal covariance. Detailed match decoding with the context-dependent units follows a fast match decoding using a single state HMM to model each phone.

3.2. Grammar generation

A directory assistance database from a Midwestern state of the US was used to test this algorithm. This database contained approximately 300,000 business and government listings on which we performed recognition experiments. We used the grammar-based implementation of the unique signature technique throughout our experiments.

In order to support a query like “3-L Hair in North Street” or “3-L Hair”, data from both name and address fields are combined to make the input string, where the address is treated as optional, and used to construct the grammars. We discovered some problems and inconsistencies in the database. For example, different abbreviations may be used for the same business listings. Some name fields contain address data. Sometimes, the street name does not have street type, and so on.

In order to achieve good results, we have found it necessary to perform some text normalization before the grammars were created. This text normalization applied numerous rules, for example,

- Compound words are formed, e.g. Red Lobster is treated as the single word Red-Lobster
- Street addresses are normalized, e.g. 17 N. State St. is changed to 17 North State Street
- Minor word re-orderings are made, e.g. Health, Dept. of is changed to Health Department
- Synonym sets are introduced, e.g. “International Business Machines” can be recognized as “IBM”, “Big Blue”, or “International Business Machines”.
- Abbreviations and typos are cleaned up.

3.3. Experimental results

The test data was collected from several cities across the US. Two data collection efforts was made through our studies. Approximately 300 subjects participated. A tree-structured representation was used for business and government listing in the directory assistance database. The first data collection was focused on simple business listings: only the first level of the tree was extracted. Listings from approximately 200 cities were extracted in numbers proportional to the population. Approximately 5000 utterances were collected. This city-dependent data was decoded using our automatically generated city-based business grammars. The word error rate and sentence error rate were 9.22% and 10.3%, respectively. Since our grammars support a variety of ways to represent a given business listing, in this type of application, it is important to measure the transaction error. After translating the recognized business name into a phone number, we observed that the 10.3% sentence error rate translates to 7.9% in transaction error rate.

Those promising results led to our second data collection effort. From the literature, a majority DA inquiries are focused on a small portion of the listings. These types of listings are called frequently requested listing. The frequently requested business and government listings from 4 major and 2 small cities were extracted from the database. Several possible natural-sounding queries, with optional street names, for these listings were generated manually for this data collection. Approximately 8000 utterances and 4000 utterances were collected for business and government listings, respectively. The word error rate and sentence error rate were 13.29% and 18.97% for frequent request business listings and 8.85% and 13.85% for frequent request government listing, respectively (from Table 5). Due the more complicated nature of the queries in this test set, the error rate increased. Additional efforts are required to analyze the transaction for this test set. We believe that it should be significantly lower than the sentence error rate, as in the first test set.

Test Set	Word Error Rate (%)	Sentence Error Rate (%)
B	9.22	10.30
FB	13.29	18.97
FG	8.85	13.85

Table 5: Error rates across different test sets. B is the simple Business listing. FB and FG are natural language query of frequently requested business and government listings.

4. Conclusions

The unique signature algorithm has been proposed to generate natural language grammars or to be used with a language model for directory assistance applications. This approach calculates the unique signatures and the confusion sets for the input listings. It automatically generates grammars to cover all different varieties of inquiries for business and government listings, without the need for handcrafted rules. Our experiments demonstrate very encouraging results. In addition, since there are no rules to be developed manually, this approach supports a daily update of the grammars to reflect a daily update of the database.

5. References

[1] K. Davies, et al, “The IBM Conversational Telephony System for financial applications”, Proc. Eurospeech, pp. 275–278, 1999.

[2] M. Lenning, G. Bielby and J. Massicotte, “Directory assistance automation in Bell Canada: Trial results”, Speech Communication, 17: pp. 227–234, Nov. 1995.

[3] L. Boves and E. den Os, “Applications of Speech Technology: Designing for Usability”, Proc. IEEE Workshop on Automatic Speech Recognition and Understanding, Keystone, Co., Dec. 1999

[4] O. Scharenborg, J. Sturm and L. Boves, “Business Listings in Automatic Directory Assistance”, Proc. Eurospeech, pp 2381–2384, 2001.

[5] H. Schramm, B. Rueber, and A. Kellner, “Strategies for name recognition in automatic directory assistance systems”, Speech Communication, 31: pp. 329–338, 2000.

[6] M. Meyer and H. Hild, “Recognition of spoken and spelled proper names.” Proc. Eurospeech, pp. 1579–1582, Rhodes, Greece, Sep. 1997.