INTERSPEECH 2009 BRIGHTON

Language Modeling and Dialog Management for Address Recognition

Rajesh Balchandran, Leonid Rachevsky, Larry Sansone

IBM T J Watson Research Center, Yorktown Heights, NY 10598, USA

rajeshb,lrachevs,lsansone@us.ibm.com

Abstract

This paper describes a language modeling and dialog management system for efficient and robust recognition of several arbitrarily ordered and inter-related components from very large datasets - such as with a complete addresses specified in a single sentence with address components in their natural sequence. A new two-pass speech recognition technique based on using multiple language models with embedded grammars is presented. Tests with this technique on complete address recognition task yielded good results and memory and CPU requirements are sufficiently low to make this technique viable for embedded environments. Additionally, a goal oriented algorithm for dialog based error recovery and disambiguation, that does not require manual identification of all possible dialog situations, is also presented. The combined system yields very high task completion accuracy, for only a few additional turns of interaction.

Index Terms: full string address, one shot destination entry, language model for address recognition, dialog management for address recognition

10.21437/Interspeech.2009-97

1. Introduction

In recent years, speech recognition has been employed for address input by voice for GPS navigation and similar applications. Users are required to first speak the name of the state, then the city, followed by the street name and finally the house number. The next generation of such systems, aim to make address input by voice more natural, so users can speak multiple components of the address in the same sentence (for example city and state or street name and house number), and even the entire address (for example "Fifteen State Street Boston Massachusetts") in a single sentence - rather than requiring four or more turns. Optionally, the full string address could be specified along with prefix and suffix words - for example "I need directions to Fifteen State Street Boston Massachusetts please". This paper describes a technique to enable this type of spoken address entry - where users can specify the address in a natural format with some or all address components.

1.1. Background

The challenge in building grammars or language models for recognizing a complete address is the huge volume of names to be covered when considering a whole country. For example, there are approximately 19,000 cities and a few million streets across the USA, resulting in multiple-millions of addresses. Even a single state such as Massachusetts has approximately 500 cities and 125,000 streets, and several hundred house numbers in each street. Recognizing a complete address such as, "Fifteen State street, Boston Massachusetts" using conventional approaches based on grammars is very difficult as the size and complexity of grammars become impractical adversely affecting memory requirements, processing time, and recognition accuracy.

1.1.1. Problem description

The general problem that is being considered in this paper is the design of an Automatic Speech Recognition (ASR) system that can recognize multiple components from a large dataset, such as addresses in a country, yellow-pages listing etc., where the ordering of components is flexible. The ASR system also needs to take into account the relationship constraints between components, as implicitly specified in the dataset, and should be scalable to very very large datasets . Finally, given that applications such as address recognition are mainly deployed in automobile or hand-held GPS navigation devices, the whole system must be able to run on standard embedded platforms.

This problem has been studied by *Parthasarathy et al* [1], [2]. They describe the limitations of traditional grammar based one-pass ASR, and also present a two-pass recognition technique where, an index-based retrieval method is used in the first-pass to produce a shortlist of form-entries and these are re-scored in the second-pass to obtain the final result. This technique has been employed in server based telephony applications with good results. Other approaches to address this problem, include requiring users to specify addresses in a non-natural format, such as, "Massachusetts, Boston, State Street, Fifteen", excluding the street suffix from street names etc. – all requiring users to interact in an unnatural manner.

In this paper, an alternative two-step technique is presented, based on multiple language models where, the first pass is used to accurately recognize some components, and the values of these components are used to dynamically update the language model used for the second pass – which is used to recognize remaining components. This technique has been developed to be suitable for use in embedded platforms. Language models with embedded grammars form the core of this technique and are reviewed next.

1.1.2. Statistical language models with embedded grammars

Speech recognition is generally carried out using grammars or Statistical Language Models (SLM). A hybrid approach [3], [4] involves embedding grammars (known as *embedded grammars*) within an SLM, to represent lists of items (called *Named Entities*) such as city names, numbers etc. This provides flexibility for the user, while allowing for dynamic content to be updated when required, simply by swapping associated embedded grammars. For example, the grammar of street names could be updated based on the selected city. The IBM Embedded Via Voice (EVV) [5] ASR engine supports SLMs with embedded grammars and the ability to dynamically swap embedded grammars at runtime.

2. Two step approach with language models

The core idea of this technique is to use two specially designed SLMs – one for recognizing the city and state and the other for recognizing the street name and number. The second SLM is updated based on the city and state recognized using the first SLM. This is carried out transparent to the user - so the user perceives full address recognition in one step.

2.1. City and state model

This is the language model that is used to recognize the specified city and state, across the whole country. It is also used to recognize just the state or just the city. This language model includes embedded grammars representing each address component, and the training data is composed of various combinations of these address component embedded grammars, so as to be able to capture various ways in which users can specify the address. A sample is shown below:

$EG_{(Num)}\ EG_{(Streets \in Mumble)}\ EG_{(Cities \in State_i, State_i)}$

$EG_{(Cities \in State_i, State_i)}$

$EG_{(Cities \in State_i)}$

$EG_{(States)}$

$EG_{(Num)}\ EG_{(Streets \in CurrentCity)}$

where,

$i = 1...50$ (Number of states in the country)

$EG_{(Num)}$ is an embedded grammar of house numbers. It could be a list of valid numbers for the whole country or a subset of representative numbers - for example numbers from 1...10,000. It could also be alpha-numeric.

$EG_{(Streets \in Mumble)}$ represents a mumble model [6] for street names - to capture any street that may be specified by the user, even if it is not recognized accurately, as long as it is not confused with any other Embedded Grammar in the language model. For example, this could be an embedded grammar of a phonetically balanced subset of street names obtained from the set of all the streets in the country.

$EG_{(Cities \in State_i, State_i)}$ is an embedded grammars of all the cities in $state_i$ including the state name, for example, "Boston Massachusetts".

$EG_{(Cities \in State_i)}$ is an embedded grammars of all the cities in $state_i$ without the state name, for example, "Boston".

$EG_{(States)}$ is an embedded grammar of state names.

$EG_{(Streets \in CurrentCity)}$ is an embedded grammar of streets in the current city, for example, based on current location of the car.

Entries within each of these embedded grammars could be weighted based on some criteria, such as relative importance of cities (for example a larger or more populous city is weighted higher). Additionally, prefix phrases such as, "I want to go to", "I need directions to" and suffix phrases such as, "please" could also be included in the training. For countries with different address formats or sequencing, the components of this language model could be modified accordingly to reflect local address specifics.

To summarize, this language model is used during the first step of recognition to enable the recognizer to accurately recognize the city and state.

2.2. Street and number model

This model is used to recognize the specified House number and Street name components of the address after having recognized the city and state using the first SLM (section 2.1). This language model also uses embedded grammars representing each address component, and the training data is composed of various combinations of these address-component embedded grammars, for example as shown below:

$EG_{(Num)}\ EG_{(Streets \in City_x)}\ EG_{(City_x, State_y)}$

$EG_{(Streets \in City_x)}\ EG_{(City_x, State_y)}$

where,

$EG_{(Num)}$ is the embedded grammar of house numbers. If available, $EG_{(Num)}$ could be replaced, *at runtime* by a city (or state) specific grammar of numbers, based on the city (or state) recognized in the first step.

$EG_{(Streets \in City_x)}$ is the embedded grammar of streets in the city that was recognized in the first step (e.g. $City_x$). This language model is updated with this grammar at runtime in preparation for the second recognition step.

$EG_{(City_x, State_y)}$ is an embedded grammar containing the name of the city and state (and optionally just the city) recognized in the first step (e.g. $City_x$, $State_y$). This embedded grammar is generated at runtime and used to update the language model in preparation for the second recognition step.

2.3. Runtime processing

To recognize the compete address, two recognition steps are carried out at runtime - as described below:

2.3.1. Step 1

- In the first step, recognition is carried out using the *City and State* language model. The user speaks an address, for example, "Directions to Fifteen State Street Boston Massachusetts". The speech recognition engine, using the City and State language model decodes all the address components. However, only the city and state can be accurately recognized, as only the city and state embedded grammars are complete and valid for the whole country.
- The recognized house number could be valid or not depending on whether the house number embedded grammar is valid for the whole country or not.
- The street name is not likely to be accurate, as a *mumble model* was used for street names.
- If the user had only specified the city and state, without any street name, no further processing is needed and the system could prompt the user for the street name and number - through an additional turn of interaction.
- If the user specified just the street name and house number in the current city - for example "Fifteen State Street". The speech recognition engine can be expected to reliably recognize the specified street name and house number as the City and State language model includes embedded grammars for the street names and house numbers of the current city. In this case, no further processing is required.

In general, the output of Step 1 is the unambiguous identification of a city and state - say $City_x, State_y$.

2.3.2. Step 2

- In the second step, recognition is carried out using the *Number and Street* language model, after updating embedded grammars based on the results of the first step.
- Specifically, the $EG_{(Streets \in City_x)}$ embedded grammar is updated with the grammar of streets in recognized city, that is, $City_x$.
- Additionally, the $EG_{(City_x, State_y)}$ embedded grammar is also updated with the recognized city and state names - that is, $City_x$, $State_y$.
- Optionally, the $EG_{(Num)}$ embedded grammar of house numbers is also updated if $City_x$ or $State_y$ specific house number information is available.
- Following these embedded grammar updates, audio saved after Step 1 is run through the speech recognition engine using this Number and Street language model.
- The speech recognition engine recognizes the specified house number and street name (or just the street name), in addition to the city and state.

The main output of Step 2 is the unambiguous identification of street name and house number. Following this, the complete address is obtained by combining the results of Steps 1 and 2. If the house number was not specified, the dialog system would prompt the user for this in a subsequent turn.

3. Dialog system for flexible address recognition and error recovery

Being able to recognize a complete address, from a single utterance is only one step in developing a robust address recognition system. Equally significant, is being able to handle incompletely specified input, various types of ambiguities and error conditions, and having an intelligent dialog [7] with the user to correct these errors and obtain a valid address at the end. This is particularly the case with recognizing a complete address spoken in a single turn, as the inherent complexity of the task significantly increases the possibility of errors in one or more components of the address.

3.1. Complexities in dialog

For the complete address recognition task, several error conditions can arise, such as:

Missing information - some component of the address was not specified, for example, house number was left out.

Multiple values are present for the same component, for example due to a mis-recognition, two cities are recognized

Ambiguous values - some component is ambiguous, by itself or in combination with other components. For example, a partially specified street name could be ambiguous.

Designing the dialog flow and error handling for the address recognition task by explicitly identifying various such error conditions, across several turns of interaction, keeping track of previous inputs, is very tortuous and can easily result in scenarios that get missed out in the dialog logic. For example, for the first turn of input, with four possible components, sixteen situations need to be considered, and then for each of these several more situations arise for the second turn and so on.

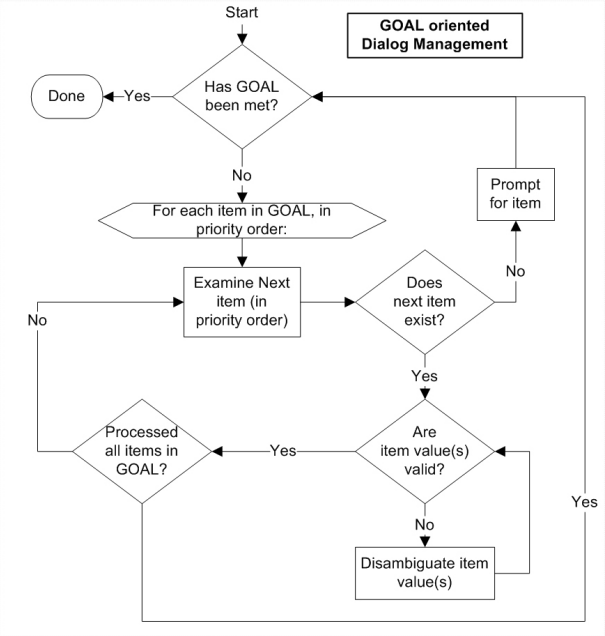

Figure 1: Dialog Flow

3.2. Goal oriented dialog flow

Considering the complexities described above, a goal oriented approach for managing the dialog that does not require manual identification of all possible scenarios was developed. The flow chart for this is shown in Figure 1. This technique is based on defining a *goal*, consisting of information needed to completing the task, and defining the *relative priority* between various components. Following this, the dialog application can follow the flow as specified, to handle various situations. For the address task, the goal and priority order can be represented as follows:

$$\begin{aligned} Goal &= \{state, city, street, number\} \\ Priority\ Order &= state \rightarrow city \rightarrow street \rightarrow number \end{aligned}$$

The algorithm iteratively tries to achieve the goal, validating available input components, and prompting for missing input components, in the specified priority order. This approach makes implementation of the dialog logic straightforward and ensures its completeness. This algorithm was implemented on a *state* based, programmable dialog management framework.

4. Experiments

Experiments were conducted to measure recognition accuracy after the first turn, and to measure task completion across turns when using the dialog system.

4.1. Off-line test results (first turn accuracy)

4.1.1. Setup

For this test, 470 complete street addresses specified in natural order (number → street → city → state) and 210 partial addresses consisting of just the city and state across eight states in the North Eastern U.S., were recorded from 8 subjects, using an AKG microphone from a distance of about 15-20 inches. IBM's Embedded Via Voice (EVV) product ASR engine was used for recognition with the two Language Models described in section 2. Results are shown in Table 1.

Task	Number of sentences	Word Error Rate	Task accuracy
City-State only	210	5.99%	90.00%
Complete address			
After step 1 (City-State)	470	5.67%	87.02%
After steps 1 & 2 (Complete address)	470	8.52%	77.00%

Table 1: Off-line test results - (first turn only)

4.1.2. Error analysis

Accuracy is higher for the city and state recognition task as compared to the complete address. This is as expected, given the considerably increased complexity of the full address task. One of the main source of ASR errors is the ambiguity in the spellings of street names in the address database, for example, *ME-135* could be pronounced, *M E one thirty five* or *Maine one thirty five*. Additionally, pronunciations for street names not presented in the base dictionary were generated automatically using the Text-To-Speech engine. This also affects accuracy, as automatically generated pronunciations are often different from the way people speak, especially for difficult names such as *manahawkin*, *penn rr avenue* etc. The dialog system helps close this gap in task accuracy, and steps to improve first turn accuracy are discussed in the future work section.

4.1.3. Memory requirements

One of the key requirements for running on an embedded platform is meeting stringent memory requirements. For the proposed technique, memory usage for the first pass - which requires the most memory - is relatively small, in the order of 15-20 MB for entire United States. This is a critical advantage of this technique as its memory requirement is linear with data size as opposed to exponential increase with conventional grammar based approaches. Memory requirements for the second pass are significantly lower.

4.2. Results on task completion with dialog system

4.2.1. Setup and measurement

For this test, 19 subjects were asked to interact with the dialog system and get it to correctly recognize 20 complete addresses. The system interacts with the user to resolve incomplete information, errors and ambiguous situations, using voice prompts and a graphical display. Figure 2 shows plots of task completion accuracy (percentage of correctly recognized address) and average number of turns vs. maximum number of turns. For example, for a maximum of 4 turns, task accuracy is 91.8% and average number of turns is 1.3.

4.2.2. Error analysis

First turn accuracy for subjects using the live system (74.74%) is comparable to that obtained with off-line testing (77%). The dialog system helps close this gap in task accuracy, mainly by prompting for missing information, resolving ambiguous and invalid input, and by allowing the user to *go back* and correct errors, all the while trying to reach the goal of getting a valid compete address. With few additional dialog turns the overall task completion accuracy is significantly improved - 96.84% for 1.54 turns on average.

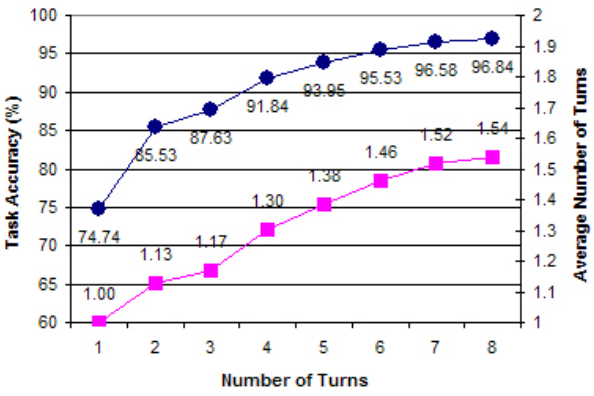

Figure 2: Task Accuracy with Dialog System

5. Conclusions and future work

In this paper, a new two-pass speech recognition technique based on using multiple language models with embedded grammars was presented. This technique is applicable to the general problem of recognizing several arbitrarily ordered and interrelated components from very large datasets - such as address recognition. Memory and CPU requirements were found to be sufficiently small to make this technique viable for embedded environments. Off-line tests on an address recognition task, where users could specify the address in natural order, yielded very good results. A goal oriented algorithm for dialog based error recovery and disambiguation, that does not require manual identification of all possible dialog situations, was also presented. The combined system resulted in very high task completion accuracy, for only a few additional turns of interaction.

While the main focus of paper is on address recognition, the proposed technique can easily be adapted to other application domains such as, recognizing directory listings, locating points of interest such as a restaurant in a specific city etc.

The main focus for future work would be to to improve the accuracy of the first turn of interaction. Likely approaches for this include, replacement of automatically generated pronunciations by manually created pronunciations for names, providing alternates for names, (for example, “J F K boulevard” could be expanded to, “John F Kennedy boulevard”) and other forms of text normalization. Additionally, the dialog system could be enhanced by including support for N-best disambiguation.

6. References

[1] Parthasarathy, S., Allauzen, Cyril and Munkong, R., “Robust access to large structured data using voice form-filling”, in INTERSPEECH-2005, 2493-2496.

[2] Parthasarathy, S. and Antonio Moreno-Daniel, ”Directory retrieval using voice form-filling”, in ICASSP-2007, 3491-3495.

[3] Picheny et al, “Speech Recognition lecture”, 2005, Online: www.ee.columbia.edu/ stanchen/e6884/slides/lecture11.pdf.

[4] Gillett, John Ward, Wayne, “A language model combining trigrams and stochastic context-free grammars”, in ICSLP-1998.

[5] Kenneth White, Harvey Ruback, Roberto Sicconi et al., “Honda Next Generation Speech User Interface”, in SAE World Congress, 2009, 2009-01-0518

[6] Gopalakrishnan, P. S., Nahamoo, David “Immediate recognition of embedded command words”, in EUROSPEECH-1991, 21-24.

[7] Balchandran et al, “Dialogue Management for Conversational Applications”, in SIGdial-2004.