# SALIENCY PARSING FOR AUTOMATED DIRECTORY ASSISTANCE

*Issac Alphonso and Shuangyu Chang*

Tellme Networks, Inc
1310 Villa Street, Mountain View, CA, 94041
{alphonso, shawn}@tellme.com

10.21437/Interspeech.2006-112

## ABSTRACT

In a statistical language model based automated directory assistance system, extracting the salient information from the recognition output can significantly increase the accuracy of the backend listing database search. In this paper, we describe a Hidden Markov model (HMM) based saliency parser that was developed to accurately and efficiently identify salient words from the recognition output by modeling both the syntactic structure as well as the lexical distribution. The parser can be trained using a relatively small data set with coarse syntactic class labels, without the need for detailed syntactic knowledge or a treebank-like corpus. Experimental results on a research corpus of directory assistance utterances betoken the parser's importance within the automated system. The results demonstrate that the proposed saliency parser can significantly improve the overall automation rate without increasing the error rate.

**Index Terms**: saliency parsing, speech tagging, named entity extraction

## 1. INTRODUCTION

Directory Assistance (DA) over the telephone is ideally suited for an automated speech recognition enabled application. In a typical DA call the caller requests a business listing in a specific city. The automated system makes an attempt to recognize the caller's utterance. If the recognition is successful, the system matches the requested listing with a database of business numbers and releases the information. If recognition fails or the requested business listing cannot be found the call is transferred to an operator.

In this research, we focus on a case where the speech recognizer employs a large-vocabulary statistical language model (SLM). The diagram in Figure 1 shows the various stages involved in such an SLM-based DA system. An SLM can potentially accept a complex caller utterance, containing not only the proper business listing name, but also additional information, such as the street address and business type, in a natural sentence. For example, instead of containing just the business name, for example, "dominos", the recognized utterance may have "i'd like dominos on jamaica avenue in brooklyn."

In many cases, such unparsed raw text strings are likely to cause significant problem for the backend search, which normally expects only the salient information, such as the business listing name "dominos." Hence, the non-salient information can significantly reduce the precision of the search, which is paramount for DA applications. Unlike web searches, phone based search applications have restrictions on the number of matches that can be presented as well as the number of interactions between a caller and the system.


*Figure 1*: An overview of the directory-assistance system showing the role of the saliency parser.

Therefore, it can be very helpful to first identify components within the recognition output that are the most salient to the listing database search. The saliency parser serves precisely this function of identifying the most relevant words within the recognition output. The success of the parser is measured by its ability to both increase the automation rate and to reduce the number of listing release errors.

In this paper, we first review a number of existing approaches that can potentially be used to parse the recognition output but have certain shortcomings for DA. Next we propose an HMM based parser that has several advantages over the existing approaches. The saliency parser description is followed by experimental results on a research DA corpus. Finally, we conclude with a general discussion and future work.

## 2. PREVIOUS WORK

There are several statistical parsers described in the literature that can potentially be used for saliency parsing [1]. Charniak described a probabilistic context-free grammar (PCFG) parser in [2]. The grammar rules are trained on counts from a treebank corpus using maximum-likelihood estimates. The PCFG's extensive dependence on formal syntactic structure appears to be unsuitable for parsing SLM recognition output of relatively short and often ungrammatical DA listing utterances. Charniak also assessed the utility of unsupervised training by incorporating the results of Viterbi parses on a held out set. However, the overall effect was marginal and further discussion was thereby omitted.

The PCFG framework was extended by Collins in [3]. Collins proposed using lexicalized grammars where the parse tree is represented by a sequence of decisions corresponding to a head-centered derivation. The independence assumption on the grammar rules is reduced to some extent by the incorporation of a distance measure in the derivation of the parse tree. Further, various other linguistically motivated model refinements are suggested. In the end, many of these refinements tend to have a strong dependency on the availability of a training corpus with high-quality detailed syntactic tagging, such as a treebank.

Jelinek describes a decision tree parser that does not require a grammar to construct a probabilistic model in [4]. A parse derivation is constructed by either labeling or extending active nodes based on an *n*-node window around the node in question. The active nodes are extended by making a binary decision at each step in the derivation of the parse tree. Each internal node in the parse tree is represented as an *n*-tuple feature vector. The decision tree parser however requires a set of deterministic rules to determine which nodes are to be extended, tagged or labeled at each step in the derivation of the parse tree.

The decision tree approach was extended by Magerman in [5]. The decision tree parser is said to automatically discover disambiguation criteria for decisions made during the parsing process. The candidate disambiguates are the words in the sentence, relationship among the words, and relationships among constituents already constructed. The parser combines a stack decoder with a breadth-first algorithm to identify the highest probability parse for any given sentence.

Finally, Kubala et al. describes an HMM-based named entity extraction approach in [6]. The approach considers each word to be an ordered two-element vector (word and word-feature). The top-level model is conditioned on the named entity class and the previous word. Further, there is special handling involved for first and last words in the sentence. These above mentioned constraints add additional overhead when training the class based n-gram language models. Kubala approach is closest in relationship to our parser.

For saliency parsing we would like a data-driven approach that could readily adapt to new applications without the need for detailed syntactic knowledge of the domain or a large amount of training data with high-quality syntactic tagging. The parser needs to be able to model both the coarse syntactic structure as well as the lexical distribution in the recognition result. Due to the stringent latency limit and scalability requirement we need a parser with the ability to trade-off accuracy for speed. These requirements lead to the consideration of an HMM-based saliency parser.

## 3. PARSER FRAMEWORK

The HMM-based parsing algorithm interprets parsing as a statistical pattern recognition process. The hidden states of the HMM represent the various syntactic classes we would like to model. For example, a syntactic class could correspond to the listing name, street address, location information, or filler words in a DA utterance. It is straightforward to rank the various permutations of the syntactic classes within an HMM framework by using the likelihood evaluation of different paths. Furthermore, the parser can impose statistical constraints on the permutations using general structures normally found in listing request utterances.

In our HMM-based parsing framework, each syntactic class can transition to any other class, i.e., the HMM is fully ergodic. The transition matrix associated with each HMM state has each corresponding member element set to one. The transition between the various syntactic classes is however restricted by a state-level N-gram statistical language model. More formally, we define the transition between syntactic classes as follows.

Let $\overline{c} = (c_1, c_2, ..., c_N)$ be the set of all defined syntactic classes. The probability of transitioning from one syntactic class to another is given by the tri-gram probability distribution:

$$P(c_{i+1} \mid c_i) = P(c_{i+1} \mid c_{i-1}, c_i)$$

The classes $c_i$ and $c_{i-1}$ represent the current and previous syntactic classes respectively.

The emission probabilities at each hidden state are represented by a class-based N-gram statistical language model. Thus, the likelihood of observing a word is conditioned on both the word history and the syntactic class. We define the emission probability at any given syntactic class as follows.

Let $\overline{w} = (w_1, w_2, ..., w_N)$ be an ordered set of all words in the recognition result. The probability of observing a word in a syntactic class is then given by the tri-gram distribution:

$$P(w_j \mid c_i) = P(w_j \mid w_{j-1}, w_{j-2}, c_i)$$

Here, $w_{j-1}$ and $w_{j-2}$ represent the previous words visited during the Viterbi search.

The dependence on the word history introduces a unique problem during the Viterbi search. The one-step Markov independence assumptions are no longer. The additional N-gram dependency means that we can no longer prune sub-optimal paths even though they have the same word history, unless the shared word history goes beyond the N-1 words. Search paths that are sub-optimal given the current observation have the capacity to become optimal when the next observation is available. Hence, the Viterbi search was modified to retain N-1 word history on all paths to ensure the optimality of the search.

The parser framework, as shown in Figure 2, can be better understood by using the example introduced earlier. If we take the Viterbi alignment of the business listing "i'd like dominos on jamaica avenue in brooklyn" to be as follows: "(i'd like)/*filler* (dominos)/*business* (on Jamaica avenue)/*street* (in Brooklyn)/*location*". Then, as an example the state transition probability of the syntactic class *location* is given by $P_t(location|business,street)$. In practice the state transition probability uses a tri-gram distribution trained on a hand labeled corpus of these coarse syntactic classes.

Furthermore, the emission probability of observing the word "avenue" in the syntactic class *street* is given by $P_e(avenue|on,jamaica)$. However the emission probability of observing the word "Jamaica" in the syntactic class *street*

would be given by $P_e(jamaica|on)$ where the tri-gram language model had to back-off to the bi-gram instead.

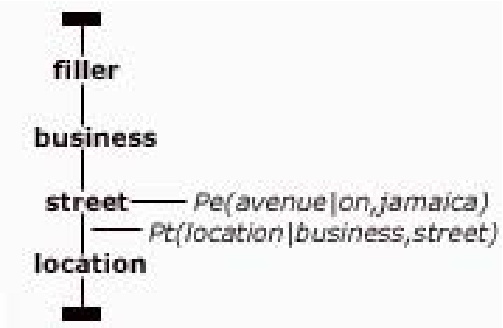


*Figure 2*: The HMM representation of the Viterbi alignment for a sample utterance (see text). The diagram shows examples of both the transition ($P_t$) as well as emission ($P_e$) probabilities.

## 4. EXPERIMENTS

To evaluate the efficacy of the proposed saliency parser, we experimented on a research corpus of 4,565 utterances collected from an automated DA application. For each utterance the caller was prompted for a listing in a major US city. The system was designed to respond with the telephone number of the requested business listing, or a *no-match* status if either the recognition or backend database search failed. The speech recognition used a large vocabulary tri-gram SLM trained on manually transcribed listing requests from the same city. The training and test sets are completely disjoint. The same backend search database and algorithms were used in all quoted experimental results. For simplicity only the business name information is used in the backend database search.

We trained the state-level language model in the parser using an automatically labeled and hand validated corpus of about 35,000 recognition results. The coarse syntactic classes that we defined for this task include *business*, *street*, *filler* and *location*. The *filler* class can be regarded as a garbage model, intended to absorb any non-salient information in the recognition result. The language model for each of the *street* and *location* classes were trained on a set of over 400,000 street address and location utterances, mostly artificially generated from a large list of popular street location names. The training data for the filler class contained about 14,000 automatically labeled and hand validated utterances. Due to the excessively large vocabulary of the *business* class, we adopted a vacuous language model for *business* that always returned the same probability for any word string. Intuitively, accurate modeling of the remaining syntactic classes would automatically lead to a reasonable result for the relatively more open *business* class. In all cases, the training data set is completely disjoint from the test set.

A performance comparison between systems that use the saliency parser with systems that do not can be seen in Table 1. The two evaluation metrics used here consist of the percentage of correctly accepted utterances and the percentage of falsely accepted utterances, measured against manually labeled references. A correct-accept (*CA*) occurs when the system correctly recognizes the requested listing and the CA rate is closely correlated with automation rate. A false-accept (*FA*) occurs when we mistakenly recognize one business listing as another. Note that a requested listing may not exist in our database, and in many cases, the requested listing may not even have a releasable telephone number. In such cases, the listing is also treated as a false-accept if the system classifies it as one of the many business listings in the database. In general, higher *FA* rates usually lead to a bad caller experience in DA.

In this test data set about half of the user queries do not have corresponding listings in the backend database according to manually labeled references. The very low in-database rate creates significant risks of FA errors and makes the recognition task very difficult.

The first row, in Table 1, shows the performance without using the saliency parser. In this case, we take the entire recognition result as being salient. The next row shows the performance using the HMM-based saliency parser. Finally, for reference, the last row in the table shows the performance where the salient information is manually labeled. In this case, human transcribers manually selected the salient information from each recognition output. This third condition is included to provide a rough upper limit on the performance of a saliency parser. The results show that the saliency parser improves the correct-accept rate by 2.2% absolute (or 8.3% relative) over the baseline, at the same false-accept rate.

*Table 1:* A comparison of three systems on the correct-accepts (*CA*) and false-accepts (*FA*) as a percentage of all utterances in the test-set. The operating point was chosen to have *FA/all* rate close to 9.5%.

| System | CA/all | FA/all |
|---|---|---|
| No Parsing | 26.6% | 9.4% |
| Auto Parse | 28.8% | 9.6% |
| Hand Parse | 29.8% | 9.6% |

A more complete comparison between the three test systems is presented in Figure 3, as receiver operating characteristics (ROC) curves. For each system, in Table 1, the ROC curves plot the overall *FA* rate on the horizontal axis and the overall CA rate on the vertical axis at different confidence threshold levels. The top-most curve represents the performance of the DA system using the manually labeled salient results. The middle curve represents the performance using the proposed saliency parsing approach. Finally, the bottom curve represents the performance of the baseline system with no saliency parsing.

The ROC curves in Figure 3 clearly show that the hand parsed system performs the best. The next best system is the one that uses the saliency parser (auto parse). Both the hand parsed system and the system that uses the saliency parser outperform the baseline in all regions of interest. In Table 1 we presented the results at an *FA* rate of around 9.5%. The FA rate was selected to provide an acceptable trade-off between automation and caller experience. In practice, the *FA* error

rate for DA applications is generally bounded by service level agreements (SLA's) with vendors.

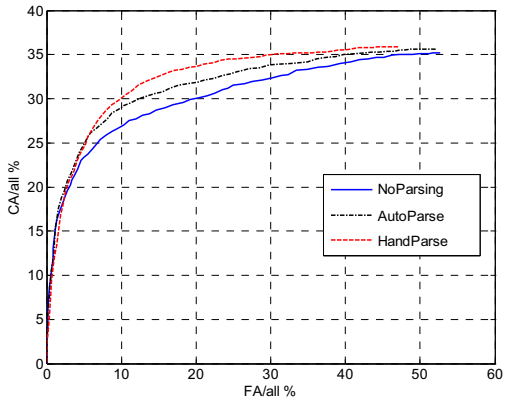


*Figure 3*: The ROC curves for systems in Table 1. The horizontal axis corresponds to the false-accept (*FA*) rate and the vertical axis the correct-accept (*CA*) rate. The top curve corresponds to the hand parser results. The middle curve corresponds to results obtained using the saliency parser. The bottom curve corresponds to a system without saliency parsing.

## 5. DISCUSSION AND FUTURE WORK

There are several advantages to using an HMM-based saliency parser in an SLM-based DA system. One advantage is that it allows us to control the tradeoff between accuracy and speed using common pruning techniques found in many speech recognition systems. The proposed saliency parser implementation uses two pruning techniques: beam pruning and instance pruning. In beam pruning, only those search paths with a log probability that falls within a delta of the best likelihood score are retained. Instance pruning on the other hand limits the number of active search paths allowed at any given state at any time interval. The beam and instance pruning techniques allow the system to achieve a good trade-off between speed and accuracy, a critical feature for production applications.

A similar framework as outlined in [2] was implemented to iteratively improve the performance of the saliency parser in an unsupervised manner. The Viterbi alignments were combined with the original corpus for an augmented training set. Experiments using the unsupervised framework however did not yield any significant improvements. That does not necessarily suggest that an unsupervised framework for training the language models is not worth pursuing. The limited or lack of performance gains is likely due to the difficulties in parsing the imperfect SLM recognition output and warrants further investigation.

In this paper, we demonstrated the applicability of the HMM-based saliency parser for a task with a limited number of syntactic classes and relatively little constraint on permutation or ordering of the classes. It would be interesting to explore the possibility of applying the HMM-based parser to more general tasks that require deep syntactic parsing [1]. Compared to traditional syntactic parsing approaches, the proposed saliency parser requires very little knowledge of the syntactic structure of the language. The training data only requires coarse syntactic class labels rather than an expensive treebank-style tagging. Thus the HMM-based parser may prove especially helpful in tasks where the syntactic structure is not well understood, such as text from a linguistically less studied language.

Furthermore, current experiments have focused on retrieving primarily the business name from the listing request. In many instances street addresses and city names can help the disambiguation process. The proposed parser is capable of delineating words in each syntactic class from the best alignment path. It will be interesting to investigate whether the parsed result is sufficiently accurate to extract useful information from the syntactic classes other than business name as is currently done.

## 6. CONCLUSION

We have demonstrated on real DA listing data that the proposed saliency parser can accurately extract potentially salient information from an SLM recognition output and help achieve an automation rate that significantly exceeds the baseline system. The parser uses a data driven HMM-based framework that lends itself to rapid prototyping. Training the parser does not require detailed syntactic knowledge or a treebank corpus, and it can be bootstrapped using a relatively small hand labeled corpus of coarse syntactic classes. Furthermore, the parser framework can leverage widely used speech recognition decoding and pruning techniques in order to trade-off speed and accuracy. Finally, the proposed saliency parser may be further extended to applications in other domains.

## 7. REFERENCES

[1] D. Jurafsky and J. H. Martin, *Speech and Language Processing*, Prentice Hall, Upper Saddle River, New Jersey, USA, 2000.

[2] E. Charniak, "Statistical Parsing with a Context-free Grammar and Word Statistics," in *Proceedings of the Fourteenth National Conference on Artificial Intelligence*, AAAI Press/MIT Press, Menlo Park, CA, USA, 1997.

[3] M. J. Collins, *Head-driven Statistical Models for Natural Language Parsing*, Ph.D. Thesis, Department of Computer and Information Science, University of Pennsylvania, Philadelphia, 1999.

[4] F. Jelinek, J. Lafferty, D. Magerman, R. Mercer, A. Ratnaparkhi and S. Roukos, "Decision Tree Parsing using a Hidden Derivation Model," in *Proceedings of the 1994 Human Language Technology Workshop*, Princeton, New Jersey, 1994.

[5] D. Magerman, "Statistical Decision-Tree Models for Parsing," in *Proceedings for the 33rd Annual Meeting of the Association for Computer Linguistics*, Boston, MA, USA, 1995.

[6] F. Kubala, R. Schwartz, R. Stone, R. Weischedel, "Named Entity Extraction from Speech," in *DARPA Broadcast News Transcription and Understanding Workshop*, Lansdowne, VA, USA, 1998.