AUTOMATIC IDENTIFICATION OF COMMAND BOUNDARIES IN A CONVERSATIONAL NATURAL LANGUAGE USER INTERFACE

Ganesh N. Ramaswamy *Jan Kleindienst*

IBM Thomas J. Watson Research Center
Yorktown Heights, New York

ABSTRACT

In this paper, we propose a trainable system that can automatically identify the command boundaries in a conversational natural language user interface. The proposed solution makes the conversational interface much more user friendly, and allows the user to speak naturally and continously in a hands-free manner. The main ingredient of the system is the maximum entropy identification model, which is trained using data that has all the correct command boundaries marked. During training, a set of features and their weights are selected iteratively using the training data. The features consists of words and phrases, as well as their relative position to the potential command boundaries. Decoding is done by examining the product of the weights for the features that are present. We also propose several enhancements to the approach, such as combining it with a more effective language model at the speech recognition stage to generate additional tokens for the identification model. We conducted several experiments to evaluate the proposed approach, and the results are described.

1. INTRODUCTION

State-of-the-art conversational natural language user interface systems typically require the user to indicate the end of a command, or the command boundary, through some form of manual input, such as pausing between commands or clicking a microphone control button on the display. Such a requirement makes the user interface quite cumbersome to use.

There appears to be no published prior work that attempts to solve the problem of automatically identifying the command boundary in a conversational natural language user interface. The only related published prior work is that of determining if a punctuation mark in raw text corresponds to a sentence boundary [4], [7]. In a conversational system, the user usually does not dictate punctuation marks, and hence these solutions are not particularly useful.

In this paper, we propose a trainable system that can automatically identify the command boundaries in a conversational natural language user interface, employing statistical techniques commonly used in speech recognition and natural language understanding. The main ingredient of the system is the maximum entropy identification model. The training data is first marked with the command boundaries. For each command boundary, all the surrounding words within a window (including words which are both to the left and to the right of the boundary) are marked to indicate their relative position with respect to the boundary. The training data which is thus processed is then subjected to maximum entropy style feature extraction, with the features consisting of words and phrases, as well as their relative position to the boundary. The corresponding weights for the features are estimated using an iterative algorithm. During decoding, the test sentences are processed similarly to mark the relative position of each of the words in the current string, with respect to a hypothesized location of the command boundary. When possible, words occurring after the hypothesized location of the boundary are also marked. Then the decision of whether or not to declare the hypothesized location as a command boundary is made by examining the product of the weights for the features that are present.

We also propose several ways to strengthen the maximum entropy identification model. One such enhancement is using a more effective language model at the speech recognition stage. All the command boundaries in the language model training data are marked with a new token, and an additional set of baseforms for the boundary (most of them corresponding to various forms of silence) are included in the model. With this addition, the speech recognition engine produces a string of text with additional tokens to suggest potential command boundaries. Other enhancements to the identification model, such as taking advantage of extended periods of silence, are also discussed.

Besides identifying the command boundary, the proposed solution can also be used to recognize multiple commands in the same sentence. This will alleviate the need to construct and support compound commands, since now sentences containing multiple commands can be automatically decomposed using the same command boundary identification scheme.

The remainder of this paper is divided into 3 sections. In Section 2, we describe the proposed solution in greater detail, along with some enhancements and other implementation related issues. The results of the experiments conducted to evaluate the performance of the proposed solution are described in Section 3, and Section 4 concludes the paper.

2. IDENTIFICATION OF COMMAND BOUNDARIES

In this section, we describe in detail the proposed solution for identifying the command boundaries in a conversational system. First, we need to distinguish utterances that correspond to *complete* commands, from those that do not. For example, consider the problem of building a conversational natural language interface for an email application. For this domain, utterances such as "check new mail" and "show me the first message" are complete commands, but utterances such as "check" , "check new" , and "check new mail show" are not complete commands.

The identification problem can be stated quite simply as follows: given a *source* utterance S consisting of one more more words, we need to determine the output T in the *target* space , such that $T = 1$ when the utterance S is a complete command, and $T = 0$ otherwise. In other words, we need to evaluate the conditional probability $P(T|S)$ for both of the values T in the target space, and select as the output that T which maximizes $P(T|S)$.

Our objective here is to build a model from training data that can generate the values $P(T|S)$. The solution we propose to generate the values $P(T|S)$ uses the *maximum entropy* principle. The maximum entropy approach has been used successfully in many natural language processing problems [1], [2], [5], [6].

In order to construct a maximum entropy model, we first need to collect a large number of training utterances relevant to the domain, corresponding to complete commands. From these utterances, we can then generate a set of utterances that do not correspond to complete commands. For every entry in this augmented set, we also indicate the desired output ($T = 0$ or $T = 1$). For the example discussed earlier in this section, where the utterance "check new mail" was followed by "show me the first message", the following entries are made in the training set:

check // $T = 0$
check new // $T = 0$
check new mail // $T = 1$
check new mail show // $T = 0$
check new mail show me // $T = 0$

In the last two entries, we have added words from the subsequent utterance. Such entries are sometimes necessary to resolve certain ambiguities that may arise. For example, utterances such as "delete", "delete this" and "delete this one" are all complete commands. In these cases, although "delete" by itself may be a complete command, it is not so when followed by "this", and similarly, "delete this" is not a complete command when followed by "one". In some other cases, the entire meaning of the utterance may change if the command boundary is placed prematurely. For example, if the input utterance is "delete the second message", then terminating the command after "delete" would result in an incorrect command being executed. Hence the above mentioned *look ahead* step is necessary, and the number of words to look ahead is one of the parameters of the algorithm that needs to be chosen carefully. In the experiments that we describe in Section 3, we use a look ahead window size of two words.

It is obvious that the presence of a word or a phrase is not sufficient to make the decision, and we also need the *relative* position of the word or the phrase with respect to a *hypothesized location* of the command boundary. If we augment each word in the training set with $-n$ if the word is n positions to the left of the hypothesized command boundary, and with $+n$ if the word is n positions to the right of the hypothesized command boundary, then the entries in the processed training set will look like:

check-1 // $T = 0$
check-2 new-1 // $T = 0$
check-3 new-2 mail-1 // $T = 1$
check-4 new-3 mail-2 show-1 // $T = 0$
check-5 new-4 mail-3 show-2 me-1 // $T = 0$
check-3 new-2 mail-1 show+1 // $T = 1$
check-3 new-2 mail-1 show+1 me+2 // $T = 1$
check-4 new-3 mail-2 show-1 me+1 // $T = 0$

In the above example, we have added additional entries to accommodate the look ahead process.

In order to build a maximum entropy model, we first have to select a set of *features*. We use features of the form

$$f_{t,s}(T, S) = \begin{cases} 1 & \text{if } t = T, s \in S \\ 0 & \text{otherwise} \end{cases}$$

Although we have restricted ourselves to using only binary valued feature functions, the method we describe is applicable for all real-valued feature functions. The features contain one or more words from the preprocessed training set (with the relative positions augmented), along with the target output. For example, consider the feature

$$f_{(T = 1),\ \text{(new-2 mail-1)}}$$

This feature fires if the utterance S contains the word new and mail at first and seconds positions, respectively, to the left of the hypothesized command boundary, and if the target output is $T = 1$.

With a slight abuse of notation, we shall denote the features by f_i, where $i = 1, \ldots, n$. The total number of features, n, is an important parameter of the algorithm, which we will revisit in Section 3. Each feature consists of one or more augmented words, including long-distance relationships, along with the corresponding output in the target space. For selecting the features, we first construct a large pool of candidate features using the data from the processed training set, and the features can then be selected from this pool using the iterative procedure specified in [5].

Having selected a set of n features, the maximum entropy model for the joint distribution of $P(T, S)$ is of the form

$$P(T, S) = \mu \prod_{i=1}^{n} \alpha_i^{f_i(S,T)}$$

See [6] for the derivation. The α_i's are the *weights* corresponding to the feature f_i, and μ is a normalization constant. The weights α_i are chosen to maximize the likelihood of the training data, and we use the *Improved Iterative Scaling* algorithm [2] to calculate these weights. From a maximum entropy point of view, we can see that the model tends not to commit towards a particular output ($T = 0$ or $T = 1$) unless it has seen sufficient evidence for that outcome in the training data; it is maximally uncertain beyond meeting the evidence.

The decision rule to classify each hypothesized command boundary is straightforward. Given an utterance S, a hypothesized command boundary is classified as an actual command boundary if and only if $P(T = 1|S) > \frac{1}{2}$, where

$$P(T = 1|S) = \frac{P(T = 1, S)}{P(T = 1, S) + P(T = 0, S)}$$

Alternatively, we decide in favor of the command boundary if and only if $P(T = 1|S) > P(T = 0|S)$.

2.1. Enhancements to the Algorithm

There are several enhancements to the algorithm that can improve the accuracy of the classification. For example, we can introduce a new token to indicate the beginning of the utterance. Using the token "SB" to indicate the beginning of the utterance, the entries in the processed training set may look like:

SB-4 check-3 new-2 mail-1 // $T = 1$
SB-5 check-4 new-3 mail-2 show-1 // $T = 0$
SB-4 check-3 new-2 mail-1 show+1 me+2 // $T = 1$
SB-5 check-4 new-3 mail-2 show-1 me+1 // $T = 0$

Marking the beginning of each utterance in the training set is straightforward. During classification, the sentence beginning token is inserted before the first utterance, and for subsequent utterances, it is inserted after every declared command boundary. Although this procedure may occasionally cause classification errors to propagate to subsequent utterances, overall it improves accuracy significantly, since it adds valuable information regarding the length of the utterance.

Another important enhancement involves using the language model (at the speech recognition stage) to produce additional tokens that can further strengthen the maximum entropy model. Specifically, we introduce a new token "SE", corresponding to the command boundary which is inserted at the end of each sentence in the language model training data, and the language model is built using this data. Acoustic baseforms for this token, corresponding to various forms of silences, are added to the model. With this enhancement, the speech recognition stage, which precedes the command boundary identification stage, will produce utterances such as "check new mail SE show me the first message SE ...". All of the training data for the maximum entropy model construction should be first subjected to speech recognition using this enhanced language model, and now the entries in the processed training set may look like:

SB-5 check-4 new-3 mail-2 SE-1 // $T = 1$
SB-6 check-5 new-4 mail-3 SE-2 show-1 // $T = 0$
SB-5 check-4 new-3 mail-2 SE-1 show+1 me+2 // $T = 1$
SB-6 check-5 new-4 mail-3 show-2 SE-1 me+1 // $T = 0$

Although errors may be made by the speech recognition stage (the SE tokens may be missing or misplaced), this enhancement also provides a significant amount of additional information that can be used by the command boundary identification stage.

In practice, we can also take advantage of any extended period of silence present in the utterances. Suppose the utterance in question is significantly different from those seen in training. Then the maximum entropy identification stage may not decide in favor of the command boundary, if there is insufficient evidence for that outcome in the training data. In this case, if the utterance is followed by an extended period of silence, then we can force a command boundary. This enhancement is also useful to reset the system if the errors in classification starts propagating to subsequent commands. The minimum duration of the silence period, after which the boundary is forced, is a system parameter that depends on each individual user, and should be made adjustable by the user via the interface to the system.

3. PERFORMANCE EVALUATION

In this section we describe some of the experiments that were done to evaluate the proposed approach. The experiments were done within the context of a task involving a spoken natural language user interface to an email application, which is the same domain that we used in Section 2 to draw examples from.

We collected approximately 3000 sentences from this domain, each corresponding to a complete command. From these

Table 1: **Experiment 1.** The table below shows the error rates for several command boundary identification experiments, with varying number of features in the maximum entropy model. The test set consisted of approximately 1900 utterances (independent of the training set), of which about 17% utterances were complete commands. The error rates for the complete commands are shown in the "Miss" column, and the error rates for the remaining utterances are shown in the "False Alarm" column. The last column shows the overall error rates, which is the weighted average of the two individual error rates.

Features	Miss	False Alarm	Overall
100	3.66%	32.91%	28.00 %
150	4.15%	26.47%	23.14 %
200	4.63%	24.02%	20.77 %
250	5.61%	20.63%	18.11 %
300	6.83%	18.61%	16.64 %
350	7.32%	17.53%	15.82 %
400	8.29%	17.14%	15.66 %

Table 2: **Experiment 2.** The table below shows the error rates for the experiments where we introduced the "SB" token in the training utterances to indicate the beginning of each utterance.

Features	Miss	False Alarm	Overall
100	8.78%	5.34%	5.94 %
150	6.10%	5.03%	5.22 %
200	5.85%	5.09%	5.22 %
250	5.37%	5.09%	5.13 %
300	6.34%	5.50%	5.64 %
350	6.59%	5.29%	5.52 %
400	6.83%	4.98%	5.30 %

Table 3: **Experiment 3.** The table below shows the error rates for the experiments where used the enhanced language model of Section 2.1, in addition to the "SB" tokens introduced in Table 2. The row labeled "LM only" is for the experiment where we used just the language model for predicting the command boundaries (the test was done only on complete commands). The remaining rows are for the cases where we used both the enhanced language model and the maximum entropy model for the identification.

Features	Miss	False Alarm	Overall
LM only	23.9 %	6.5%	-
100	3.72%	4.22%	4.13 %
150	3.72%	4.09%	4.03 %
200	4.02%	2.40%	2.67 %
250	4.64%	2.27%	2.67 %
300	5.57%	2.01%	2.62 %
350	5.57%	2.01%	2.62 %
400	5.57%	2.14%	2.72 %

sentences, we constructed additional utterances (about 12000) which do not correspond to complete commands, similar to those discussed in Section 2. These 15000 utterances were processed to indicate the relative positions of the words with respect to a hypothesized command boundary, and the processed training data was used to construct the maximum entropy classification model.

Similarly, for the purpose of testing, we collected another set of about 320 sentences corresponding to complete commands and generated additional utterances that do not correspond to complete commands. In the final test set, which consisted of about 1900 utterances, approximately 17% of the utterances were for complete commands.

Table 1 shows the results of the first experiment. In this experiment, we did not use any of the enhancements proposed in Section 2.1. We constructed several different classification models with varying number of features, and the overall results improved with increased number of features. The column labeled "Miss" contains the error rates for the portion of the test data corresponding to complete commands, where the identification algorithm "missed" to recognize the utterances as being complete commands. Similarly the column labeled "False Alarm" contains the error rates for the portion of the test data corresponding to utterances that are not complete commands, where the identification algorithm incorrectly declared a command boundary. The overall error rates are the weighted averages of the two error rates.

Table 2 shows the results of the second experiment, where we introduced the "SB" token to indicate the beginning of the utterance, as we discussed in Section 2.3. The error rates dropped significantly compared to Table 1, suggesting that the statistics corresponding to the utterance length is very important in determining the sentence boundary. Also note that the best overall performance was obtained with 250 features, and the performance starts to deteriorate after that.

The language model containing the "SE" tokens, corresponding to the command boundaries as predicted by the language model, was introduced in the third experiment and the results are shown in Table 3. The language model constructed was a simple trigram language model, with bigram and unigram models mixed in for smoothing, as prescribed in [3]. The first row contains the error rates for the experiment where we used just the language model to perform the identification, without the maximum entropy classification stage. The utterances from the test set corresponding to complete commands were subjected our speech recognition engine, and we found that the anticipated "SE" tokens at the end of the utterances were missing in 23.9% of the utterances, and 6.5% of the utterances contained incorrect "SE" tokens. When we introduced the maximum entropy classification model, trained on data that included the "SE" tokens, into the system, the error rates dropped significantly to 2.62%, with 300 features. Such a low error rate clearly makes the proposed solution an excellent practical choice for identifying command boundaries in a conversational natural language user interface.

Tables 1-3 show the performance of the off-line testing. We also implemented the solution in a real-time system, along with the enhancement due to extended silence periods discussed in Section 2.1. The performance was comparable to that of Table 3, and since the command boundary identification is a relatively light-weight component compared to the speech recognition and natural language processing components, there was no noticeable degradation in the speed of execution of the commands.

As noted in the introduction, the command boundary identification scheme can also be used to recognize multiple commands present in the same sentence. For example, if the user were to say "forward this message to John and delete it,", the two commands can be decomposed automatically using the proposed approach for command boundary identification, as long as we have also included additional data in the training set corresponding to such compound commands.

4. CONCLUSIONS

We have described a new approach to automatically identify the command boundaries in a conversational natural language user interface. The proposed solution makes the conversational interface much more user friendly, and the user can speak naturally and continously in a hands-free manner. The experimental results show that the proposed solution can identify the command boundaries with a high degree of accuracy, and since it does not require any heavy-duty computation during identification, it is very suitable for practical implementation.

ACKNOWLEDGMENTS

The authors are grateful to Ponani S. Gopalakrishnan for posing the command boundary identification problem, and to Kishore Papineni for the introduction to maximum entropy based techniques.

REFERENCES

[1] Berger, A., Della Pietra, S., and Della Pietra, V., " A Maximum Entropy Approach to Natural Language Processing," *Computational Linguistics*, Vol. 22, No. 1, pp. 39-71, March 1996.

[2] Della Pietra, S., Della Pietra, V., and Lafferty, J., "Inducing Features of Random Fields," Technical Report CMU-CS95-144, School of Computer Science, Carnegie Mellon University, 1995.

[3] Jelinek, F., *Statistical Methods for Speech Recognition*, The MIT Press, 1997.

[4] Palmer, D. D., and Hearst, M. A., "Adaptive Multilingual Sentence Boundary Disambiguation," *Computational Linguistics*, Vol. 23, No. 2, pp. 241-267, June 1997.

[5] Papineni, K., Roukos, S., and Ward, T., "Feature-Based Language Understanding," *EUROSPEECH*, Rhodes, Greece, 1997.

[6] Ratnaparkhi, A., "A Simple Introduction to Maximum Entropy Models for Natural Language Processing," Institute for Research in Cognitive Science, Report 97-08, University of Pennsylvania, May 1997.

[7] Reynar, J. C., and Ratnaparkhi, A., "A Maximum Entropy Approach to Identifying Sentence Boundaries," *Fifth Conference on Applied Natural Language Processing*, Washington, D. C., pp. 16-19, April 1997.