A Maximum Entropy Shallow Functional Parser
for Spoken Language Understanding

Pierce G. Buckley, David Horowitz, Partha Lal

Vox Generation Ltd.
Manor House, 21 Soho Square, London W1D 3QP, UK
pbuckley@voxgeneration.com

Abstract

In this paper we investigate a maximum entropy approach to spoken language understanding. We compare this approach with a parser based on finite-state transducers. The parsers are evaluated on a corpus of utterances modelling human-computer interactions within a single domain. The corpus was annotated with task-oriented semantic categories to obtain a set of shallow functional parse trees. We focus our investigation on the quality of the structured concepts produced by each approach. A direct comparison shows that the maximum entropy parser achieves better performance than the finite-state parser, even with very limited training data.

1. Introduction

In the FASIL project, a virtual personal assistant is being developed to enable users to interact with their email, address book and calendar through both audio and graphical interfaces [1, 2]. We focus our attention in this paper on spoken language interactions concerning the sending of new emails.

One of our objectives is to enable more natural forms of human-computer interaction. In particular, we investigate approaches to enable a virtual personal assistant application to reliably recognise and interpret a greater variety of spoken language utterances. Here we concentrate on spoken language understanding; research in this project on language modelling and recognition is reported separately [3]. In this paper we compare two approaches to analyse the semantics of natural language inputs. The first approach uses the maximum entropy framework [4], while the second is based on finite-state transducers [5, 6]. We apply both approaches to the problem of how to map an input sentence to a shallow functional parse tree representing the semantic interpretation of the sentence.

2. Related work

The use of finite-state transducers for language processing is well represented in the literature [3, 5, 6, 7]. Several authors have proposed statistical approaches to natural language understanding. Approaches based on hidden Markov models have been reported in a number of papers [8, 9]. He and Young [10] introduced a method based on hidden vector state models.

Bender et al. [11] used the maximum entropy framework to map sentences to a flat string of concepts and compared this method to an approach based on statistical machine translation. The search procedure used for the maximum entropy approach investigated here is closest to that proposed by Ratnaparkhi [12] and is described in more detail in Section 4.

The remainder of the paper is organised as follows. The corpus and semantic representation used are described in Section 3, and then in Section 4 we describe the maximum entropy and finite-state approaches to parsing spoken language utterances. The results of experiments are reported in Section 5. The remainder of the paper discusses possible further work and some conclusions.

3. The corpus and semantic representation

For the purposes of the experiments reported here, we made use of a corpus of transcribed conversational speech data. The corpus statistics are summarised in Table 1. The speech data contains a mixture of longer natural interactions and shorter directed dialogue utterances.

<table>
<thead>
<tr>
<th></th>
<th>Train</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>Utterances</td>
<td>1134</td>
<td>305</td>
</tr>
<tr>
<td>Tokens</td>
<td>5744</td>
<td>8384</td>
</tr>
<tr>
<td>Vocab</td>
<td>207</td>
<td>37</td>
</tr>
</tbody>
</table>

Table 1: Summary of the corpus of transcribed and annotated conversational speech data. “lang” refers to natural language and “sem” refers to semantic annotations.

1This research is supported by the EU Grant FASiL IST 2001-38685, www.fasil.co.uk. FASiL stands for Flexible and Adaptive Spoken Language and Multi-Modal Interfaces.
Figure 1: An example of a simple shallow functional parse tree for the utterance “Send an email to John Smith”.

The corpus was annotated using a custom-designed, task-oriented semantic tree structure, which we refer to as a shallow functional parse tree. A simple example is shown in Figure 1. Sequences of words which have a particular syntactic and semantic function are assigned a functional category label. The functional categories were derived from the task frame semantics used by the dialogue manager [1]. In order to be able to pass an appropriate representation of the utterance to the dialogue manager, all attributes should be correctly assigned within the hierarchical structure of the parse tree. To illustrate how we evaluate the quality of parses, consider the correct example in Figure 1. Let’s say the parser correctly determines that the user wishes to add someone to the “To:” list (i.e. a TOCOMMAND is identified) but the scope of the TOCOMMAND does not cover the person list (PERL) containing the person (PER) “John Smith”, then we would consider this to be a concept error.

This type of shallow parse representation has the advantage of being close to the representation of semantics used by the application, but it suffers from having an insufficiently abstract representation of syntax and semantics. One concrete effect is that the formalism cannot deal with discontinuous constituents. We regard the shallow parsers investigated here as first steps towards commercial technologies that support more general grammar formalisms.

4. The maximum entropy and finite-state parsers

We wish to compare the performance and characteristics of two approaches to parsing: a statistical maximum entropy parser and a rule-based finite-state parser. Before looking at the results of experiments, we will briefly describe each approach.

4.1. Maximum entropy parser

The maximum entropy framework has been applied successfully to a variety of natural language processing tasks [4, 11, 12]. The maximum entropy parser investigated here uses a bottom-up parsing procedure similar to that reported in [12]. The maximum entropy parser makes use of a set of feature functions \( \{ f_i(x, y) \}_{i=1}^M \). For any context \( x \) and outcome \( y \) a feature function \( f_i \) returns 1 if the \( i^{th} \) feature is present in \( x \) and can predict the outcome \( y \), otherwise it takes the value 0. The feature functions capture information in \( x \) which is useful in predicting \( y \). For each feature function \( f_i \) there is a corresponding model parameter \( \lambda_i \). The probability of an outcome \( y \) given a context \( x \) is modelled as

\[
p_\lambda(y|x) = \frac{1}{Z_\lambda(x)} \exp \left( \sum_{i=1}^{M} \lambda_i f_i(x, y) \right),
\]

where \( Z_\lambda(x) \) is a normalisation factor given by

\[
Z_\lambda(x) = \sum_y \exp \left( \sum_{i=1}^{M} \lambda_i f_i(x, y) \right).
\]

The initial constituent trees are simply the words of the input sentence. At each stage, the parser decides which constituent trees to process and takes one of three actions: 1) construct a new constituent, 2) attach to an existing constituent, or 3) decide whether a constituent is complete. The parser applies these actions until a complete parse for the sentence has been derived. For example, given the partially constructed parse \( T \) in Figure 2, the parser determines that the 2nd tree should be used to construct a new constituent tree. If we give this tree the index 0 and refer to it as \( T(0) \), then features are extracted from a window of two trees back and two trees forward, say, \( T(-2), T(-1), T(0), T(1) \) and \( T(+2) \). Examples of feature descriptions for these trees are shown in Table 2. The features define the information available to the maximum entropy model to predict the appropriate outcomes and their probabilities using (1). In this case the correct outcome would be TOCOMMANDStart.

![Figure 2: An example of constituent trees of a partially constructed shallow functional parse, \( T \), for the utterance “Send an email to him”.](image)
The highest accumulated score, as determined by (4). In the best partial parses are those whose constructions have the notation is scored as follows:

$$\lambda^* = \arg \max_{\lambda} \left( \sum_{x,y} p(x,y) \log p_\lambda(y|x) \right)$$  \hspace{1cm} (3)

where $p(x,y)$ is the empirical distribution of the training data. To train the model parameters $\lambda$ we used the limited memory variable metric procedure presented in [13].

Parse trees are scored using the same algorithm as presented in [12]. A partial derivation $T$ is arrived at by deciding on a set of outcomes $\{y_1, ..., y_k\}$, each outcome $y_i$ being a decision taken in a context $x_i$. The partial derivation is scored as follows:

$$\text{score}(T) = \prod_{i=1}^{l} p_\lambda(y_i|x_i)$$  \hspace{1cm} (4)

where $p_\lambda(y|x)$ is the model conditional distribution.

We implemented a $K$-breadth-first search algorithm where only the best $K$ partial parses are explored. The best partial parses are those whose constructions have the highest accumulated score, as determined by (4). In the experiments reported here only the first best hypothesis was processed at each step.

### 4.2. Finite-state parser

The finite-state parser investigated here was encoded using the W3C speech recognition grammar specification (SRGS)\(^2\). The grammar is then converted to a finite-state transducer (FST) representation. Parses are not scored and no probabilities or weights are used, instead the search procedure uses FST operations to determine all matching parses. Typically the search results in a number of parse trees. The best result is then determined using two heuristic rules. The first heuristic chooses the parses with the maximum span, i.e. the parses covering the most words. The second heuristic chooses from the remaining parses the one with the least number of top level concepts. We then decide arbitrarily between any remaining parses. This simple procedure actually produces quite good results, but it has two major drawbacks. Since the initial search explores the entire set of rules, it does not scale to larger vocabularies, longer utterances or more complex semantic spaces. Secondly, it is very time-consuming to develop and difficult to tune.

### 5. Experiments and results

Since both parsers take sentences as input and produce shallow functional parse trees as output, their performance can be compared directly. All experiments were conducted on the same test set summarised earlier in Table 1. The results are shown in Tables 3 and 4. Two measures of performance were used to evaluate the parsing accuracy. The tree error rate (TER) is simply the percentage of incorrect trees. The second measure is the concept error rate (CER). In order to determine the concept error rate, we extracted the name of the top level concepts from each parse tree and concatenated each attribute to form a string of concept tokens. The CER is then the Levenshtein-alignment error for each pair of reference and hypothesised concept token strings. This measure is intended to capture the notion of parse quality discussed in Section 3.

<table>
<thead>
<tr>
<th>FST parser</th>
<th>ME parser</th>
</tr>
</thead>
<tbody>
<tr>
<td>dir utts</td>
<td>2.4</td>
</tr>
<tr>
<td>nat utts</td>
<td>36.7</td>
</tr>
<tr>
<td>all utts</td>
<td>14.4</td>
</tr>
</tbody>
</table>

Table 3: Comparison of performance of the FST and ME parsers. "dir" refers to utterances from directed dialogue interactions, while "nat" refers to more natural conversational interactions.

<table>
<thead>
<tr>
<th>DEV</th>
<th>TEST</th>
</tr>
</thead>
<tbody>
<tr>
<td>8.5</td>
<td>3.8</td>
</tr>
<tr>
<td>8.2</td>
<td>4.6</td>
</tr>
</tbody>
</table>

Table 4: Performance of the maximum entropy parser after tuning on development data. TEST refers to the same 305 utterances of heldout data used in all experiments, so the error rates can be compared directly with results for "all utts" in Table 3.

Note that, for directed utterances, while 1% of parse trees produced by the ME parser have some error, they happen to have no effect w.r.t. the more forgiving CER measure. The ME parser outperforms the FST parser for

---

\(^2\)http://www.w3.org/TR/2004/REC-speech-grammar-20040316/
the shorter directed dialogue utterances, but the difference is not significant for the longer utterances taken from more natural interactions. This is probably due to the fact that 70% of the training data is made up of directed dialogue utterances. In the course of the experiments we noted that using feature cutoffs greater than 1 actually caused both the TER and CER to increase. It turns out that many features in longer natural utterances occur only once in the training corpus, indicating that there is insufficient training data. Since this is often the case when developing speech applications, we wanted to investigate how easy it is to improve the performance of the ME parser as new data becomes available. To maintain the split between training and test data, we divided the original training data into a new smaller training set (1005 utterances) and a development set (129 utterances). We then annotated 58 more utterances to tune performance on the development set. As can be seen in Table 4, using a small amount of additional data can significantly improve performance on unseen data. In contrast, the performance of the FST parser could not be improved on this test set, despite having been continuously tuned in a test deployment of the application. As soon as more annotated data becomes available, we will be able to determine more precisely the effect of training set size on performance.

6. Future work

In the immediate future, we intend to develop further shallow functional parse treebanks in the three languages of the project: English, Portuguese and Swedish. In the longer term, we think it would be beneficial to investigate how the maximum entropy and other parsing methodologies scale to domains with larger vocabularies and more complex semantics. Considering the disadvantages of shallow functional parse trees, we are currently looking into grammar formalisms and appropriate parsing strategies to provide more extensible and reusable frameworks for representing the syntax and semantics of spoken language utterances. Finally, we are investigating the closer integration of syntactic and semantic parsing mechanisms with the speech decoding algorithm.

7. Conclusions

We have compared the performance of two approaches to parsing spoken language utterances. One was a rule-based finite-state transducer parser, while the other was based on the maximum entropy statistical framework. We used a task-oriented representation to encode the syntactic and semantic structure of utterances. The two parsing methods were compared on a task specific corpus in the domain of a virtual personal assistant. For this corpus, we have demonstrated that the maximum entropy approach achieves better accuracy without the need for the time-consuming development work required for the rule-based parser.

8. References


