Chapter 4: Structured Language Model Parsing

Oren Schwartz, Univ. of Pennsylvania

Introduction

The Structured Language Model (SLM) was introduced by Ciprian Chelba and Frederick Jelinek as a language model that uses a statistical parser in a left to right manner in order to exploit syntactic information for use in a language model as part of a speech recognition system (Chelba, Jelinek, 1998; Chelba, 1997). In traversing a sentence, the Structured Language Model produces a series of partial lexical parses whose exposed headwords are used instead of the previous two words in a trigram-like prediction process. It is conjectured that this language model could be easily modified to produce good complete parses for Czech. One of the attractive features of the parsing method used in this model is that it can be easily modified to handle "crossing" or non-projective dependencies, a feature of the Prage Dependency Treebank that our other parsers currently ignore. Chelba, who implementated his Structured Language Model in C++, was gracious enough to allow us access to his code in order to modify it for our purposes. During the workshop, with time constraints and the usual range of difficulties encountered, we had time only to modify this Structured Language Model to work properly with the Czech data, to experiment with unknown word statistics, and to use part of speech tags generated by a version of Jan Hajic's exponential model statistical tagger. The results obtained during the workshop are encouraging as they were obtained with a version of the SLM which, though modified, is still not optimized for parsing.

The Structured Language Model

The Structured Language Model consists of two main parts: a parser and a predictor. The model proceeds along a sentence in a left to right manner, constructing partial parses for the sentence prefix available at each word. These partial parses consist of binary branching lexical parse trees where each node is associated with a lexical item and a nonterminal or part of speech (POS) label. Each nonterminal label and lexical item pair that covers a partial parse is referred to as a headword. The predictor uses the last two exposed headwords over a sentence prefix to predict the next word in the sentence. With parameter reestimation Chelba and Jelinek report results that this technique does achieve significantly lower perplexities for the UPenn Treebank of Wall Street Journal text (Chelba, Jelinek, 1998).

Partial parses are constructed in a binary manner by considering the last two headwords and their associated tag information. The parser can choose from three moves at any given point: ADJOIN-RIGHT, ADJOIN-LEFT, or NULL. An ADJOIN-RIGHT move creates a new nonterminal that covers the last two words, percolating the right word up the parse as the headword of the new phrase. ADJOIN-LEFT acts similarly, except that the left word is percolated up. After each adjoin operation, headwords are renumbered. The parser continues to build syntactic structure over a sentence prefix in this manner until the most probable move is the NULL move, which does not change the parse structure, but passes control to the predictor, which then predicts the next word and its POS tag from the previous

two headwords. The model proceeds down each sentence in this manner until the end of sentence marker is reached. Because of the large (exponential) number of possible parse trees associated with any given sentence, this model uses a multiple stack search with pruning through the space of sentence parse tree hypotheses (see Chelba, Jelinek 1998). In this manner, hypotheses with equal numbers of parser operations and predictions are compared against eachother.

Probability Model

The probability model used by the SLM was not changed during the course of this workshop. The probability of a sentence consists of two main parts, the predictor probabilities and the parser probabilities. For our purposes, a headword consists of a word and a nonterminal label, $h = \langle w, t \rangle$ where the nonterminal label may be a POS tag if the headword is a terminal item in the parse tree. The total probability of a sentence of length n and its associated parse tree, T_n, is the product of the predictor probabilities for each new word-tag pair, multiplied by the product of the probabilities of the sequence of moves taken by the parser at each position k along the sentence.

The predictor decomposes the probability of the current headword, h_0^k , given partial parse T_k of the k-word sentence prefix as follows. We first predict the current word given the previous two headwords for a particular partial parse of a k-word sentence prefix, and then the POS tag associated with it given those headwords and the word itself. The predictor probability is then

$$P(h_0^k \mid h_{-2}^k, h_{-1}^k, T_k) = P(w_0^k \mid h_{-2}^k, h_{-1}^k, T_k) \cdot P(t_0^k \mid h_{-2}^k, h_{-1}^k, T_k, w_0^k)$$

The parser probability for the partial parse of the k-word sentence prefix given the partial parse of the (k-1)-word sentence prefix is the product of the probabilities of the sequence of actions of length N_k, ending with a NULL move:

$$P(T_k \mid T_{k-1}) = \prod_{i=0}^{N_k} P(a_i^k \mid h_{-2}^i, h_{-1}^i, T_{k-1}, a_1^k \ldots a_{i-1}^k)$$

The total probability of a sentence is then

$$P(W, T) = \prod_{k=0}^{n} \left[P(h_0^k \mid h_{-2}^k, h_{-1}^k, T_k) P(T_k \mid T_{k-1}) \right]$$

Dependencies and Binary Trees

Clearly, the parses constructed by the Structued Language Model are binary lexical parse trees. Given a lexical parse tree of any form, a corresponding dependency structure for the same sentence can be uniquely determined. The SLM is a statistical model and thus requires training data in the form of binary lexical parse trees. The correspondence from dependency structures to lexical parse trees for a given sentence is one to many, as neither nonterminal labels nor the number of intermiediate nonterminal phrases and their internal structure are uniquely defined by a dependency structure. As the Collins parser also works with lexical context-free trees, efforts were made by our team to find an optimal way to construct these trees from dependencies. For the Structured Language Model parsing effort, the same techniques used with the Collins parser were used with a simple modification to restrict these trees to be binary branching. The modification converts any rule with more than two children into a binary branching sequence of rules by introducing new nonterminals to cover the headword of the phrase and its left neighbor if it exists or its right neigbor otherwise. This is done recursively through the tree until each tree is uniformly binary branching. New nonterminals are constructed from the nonterminal label of the parent of the phrase by adding a prime symbol. The existence of unary constructions is allowed, provided that the child is a terminal. By using the same methods for converting dependencies to trees, we benefit from all improvements to these algorithms made during the workshop.

Unknown Words

The unknown word problem in Czech is substantial, due in large part to the heavily inflectional quality of the language. Approximately 12% of all tokens and 35% of all words encountered in the test data set were never seen in the training set. Because the SLM was developed as a language model, it has little use for out of vocabulary words, as they will be predicted infrequently. The first attempt to use the SLM for parsing Czech did not change the manner in which unknown words were dealt with, leading to a fairly low baseline result. To alleviate some of the problems caused by unknown words, we add a threshold in the training data such that if a word is seen less than that threshold number of times, it is considered unknown. Currently, all unknown words are mapped to a unique string, "<unk>" so that the statistics gathered for unknown words in training hold for those encountered in the test.

Threshold	Unknown words in test	Unknown tokens in test
none	35%	12%
3	57%	19%
5	67%	23%

Part of Speech (POS) Tags

Jan Hajic (Hajic, Hladka, 1998) brought to the workshop an exponential model statistical parser that performs at ~92% accuracy which was run on our test data without having seen any of it in training. These POS tags for each word in the test data were available to our parsers. By using these tags for unknown words we notice a substantial improvement over our baseline performance for SLM parsing. When we use these tags for all words, we notice a very small (<.3%) improvement over the results with these tags for unknown words only, suggesting that the SLM is doing well in assigning POS tags to words that it does recognize.

It should be noted that even though the part of speech information in the Prague Dependency Treebank is quite extensive, all results shown here were achieved with a drastically reduced tag set. If the full set of tags were to be used, a drastic sparse data problem would arise. The reduced tag set used here is the same one used in the Collins parser for this workshop. Again, further work in determining an optimal reduced set of tags may significantly improve performance here, as well as in the Collins parser.

Preliminary Results

Training was done on a set of 19,126 sentences. Two test sets were used, a development set with 3,697 sentences and an evaluation set with 3,807 sentences. The composition of these data sets are explained in the Data section.

Baseline results:

No unknown word statistics. No use of external (MDt = Machine Disambiguated tags) POS tags.

devel	54.7%
eval	55.5%

Use MDt tags for:

unknown word threshold:3	none	unk	all
devel	57.91%	67.42%	67.54%
eval	57.70%	67.16%	67.45%

unknown word threshlod:5	none	unk	all
devel	--	68.04%	68.18%
eval	57.29%	68.12%	68.32%

Further Research

There are many possible modifications to the algorithm and details of its implementation that have not yet been explored, each of which with the potential to improve the performance of the SLM in terms of the parsing effort. Some are specific to the SLM, while others are potentially beneficial to many parsers. Some of the more promising avenues are mentioned here.

Crossing Dependencies Modification

One of the interesting features of the sentences in the Prague Dependency Bank is that they contain crossing or non-projective dependencies. Fortunately, these crossing dependencies make up only about 2% of all dependencies, so for the most part handling these dependencies correctly was not a top priority for most of our parsing efforts. The SLM can be easily modified to produce crossing dependencies directly. The modification is as follows. The parser, instead of considering only the last exposed headword and the current headword as possibilities for an adjoin operation, now considers the current headword and the entire list of previous exposed headwords as possibilities for adjoin operations. It must now predict an action together with an integer specifying which previous headword should participate in the adjoin operation. Of course, we must condition these probabilities on certain features to penalize adjoins of headwords that are not adjascent, as these non-projective dependencies are only a small fraction of all dependencies. The exact nature of the features to condition on, and the details of the decomposition of the probability model for smoothing purposes are open questions.

Closing Parses

When the end of sentence marker is reached in the Structured Language Model, all partial parses are forced to ADJOIN-RIGHT with probability 1 to the end of sentence marker until the only two headwords are the end of sentence and beginning of sentence markers. When the SLM is used as a language model, once the end of the sentence has been predicted, there is no further use for its syntactic structure, and closing parses in this manner is perfectly acceptable. For our ends, however, it would clearly be beneficial to allow the statistics to guide the closing of a parse tree.

Direct Optimization for Parse Structure and Parameter Re-Estimation

The SLM currently uses the perplexity measurement as criteria for optimization in the reestimation of parameters. We would like to perhaps introduce a measure for optimization of the parse structures

output directly. In so doing, we may be able to explore reestimation techniques with regard to the parses we create.

Predictor Probability Decomposition

Because the SLM was designed as a language model for use in continuous speech recognizers, the predictor first predicts the next word from the previous two headwords, and then the POS tag from that word and the previous two headwords (see Probability Model above). As we are now focusing on parsing, we might consider predicting the POS first, as this might be more consistent with recovering reliable parses for sentences with unknown words.

Data Annotation Problems

Certain aspects of the manner in which the Prague Dependency Treebank is annotated may be less than optimal for the purpose of producing a statistical parser for Czech sentences. For example, in the PDT coordinated phrases are dependent on the coordinator in the sentence. Certainly for predicting, one would assume that the headwords of the coordinated phrases are more likely to give a good estimate of what the next word would be than the coordinator. We might like to explore the effects of handling certain features (e.g. coordination, punctuation, etc.) in the Czech data in slightly different ways so that our parsers can make use of as much information as possible.

Nonterminal Labels and POS Tag Sets

It should be noted that the manner in which sentences are annotated with dependency structures and the manner in which those structures convert to trees are very important concerning the performance of our parsers. Nonterminal labels should ideally describe completely the terminals they cover to the rest of the sentence. Fully exploring the best choices for nonterminal labels and rule generation is a task which has yet to be completed, and one which may yield significant improvements in parser performance.

Likewise, finding an optimal set of POS Tags may also increase performance significantly. There was simply not enough time during the workshop to explore fully techniques for clustering these tags, but efforts were made and are explained elsewhere in this report.

References

Chelba, Ciprian and Frederick Jelinek, "Exploiting Syntactic Structure for Language Modeling." 1998.

Chelba, Ciprian, "A Structured Language Model," 1997.

Hajic, Jan and Barbora Hladka, "Tagging Inflective Languages: Prediction of Morphological Categories for a Rich, Structured Tagset," In: Proceedings of ACL/Coling’98, Montreal, Canada, Aug. 5-9, pp. 483-490, 1998.