EFFICIENT SENTENCE SEGMENTATION USING SYNTACTIC FEATURES

Benoit Favre, Dilek Hakkani-Tür

International Computer Science Institute
1947 Center St, Berkeley, USA
{favre,dilek}@icsi.berkeley.edu

Slav Petrov, Dan Klein

Computer Science Division
University of California Berkeley, USA
{petrov,klein}@cs.berkeley.edu

ABSTRACT

To enable downstream language processing, automatic speech recognition output must be segmented into its individual sentences. Previous sentence segmentation systems have typically been very local, using low-level prosodic and lexical features to independently decide whether or not to segment at each word boundary position. In this work, we leverage global syntactic information from a syntactic parser, which is better able to capture long distance dependencies. While some previous work has included syntactic features, ours is the first to do so in a tractable, lattice-based way, which is crucial for scaling up to long-sentence contexts. Specifically, an initial hypothesis lattice is constructed using local features. Candidate sentences are then assigned syntactic language model scores. These global syntactic scores are combined with local low-level scores in a log-linear model. The resulting system significantly outperforms the most popular long-span model for sentence segmentation (the hidden event language model) on both reference text and automatic speech recognizer output from news broadcasts.

Index Terms— Speech processing

1. INTRODUCTION

Sentence segmentation takes word sequences transcribed from an audio document and annotates them with sentence boundaries. Because sentences are the main units for many natural language processing tasks, accurate segmentation is a prerequisite for many kinds of subsequent processing. For example, current machine translation and question answering systems operate at the sentence level, being trained from sentence segmented text, and assuming the presence of standard punctuation.

Sentence segmentation is typically cast as a local classification problem at the word boundary level [1, 2, 3], with pause duration being the most informative feature. However, it is very difficult for local, word-level models to cope with pauses in the middle of a sentence, and oversegmentation is a common result (see the example in Figure 1). To rule out such spurious segmentations, a model needs a broader view of the hypotheses, one which can assess the context more globally.

Two ways of implementing longer-span information have been proposed in previous work. Hidden event language models (HELM) [4] use spans of up to 5-grams, which on one hand are still far below the average sentence length, yet on the other hand require training from a very large corpus to overcome data sparsity. In contrast, [5] uses a rich set of non-local features including parser scores, but their feature choice and system design require them to rerank full segmentations. Because the number of full segmentations is exponential in the number of words in the lattice, it is only feasible for the short-sentence contexts they consider, and even then it requires long discourses to be broken into short segments before being analyzed.

In this paper, we use an estimate of the grammaticality of sentence hypotheses to improve sentence segmentation quality, while operating in a model which permits efficient search over entire discourses. On top of standard local features, which are described below, we score potential sentences using the generative syntactic language model induced by a probabilistic context-free grammar (PCFG) from a state-of-the-art syntactic parser. Of course, these scores are more expensive to compute than the local scores, and, for added efficiency, it is advantageous to skip the scoring of clearly suboptimal candidate sentence spans. We therefore use a two-pass approach. First, a word boundary level model based on prosodic and n-gram features is used to construct a lattice of sentence candidates. Low-scoring spans are pruned and the remaining spans are scored with the parser. The most probable path in the resulting lattice is then extracted as the final full segmentation. To be able to better handle noisy automatic speech recognizer (ASR) utterances, which are out-of-domain for standard treebank parsers, we use unsupervised domain adaptation and self-train the parsing grammar on speech transcripts. Sentence level information from the parser results in significantly better sentence segmentation performance both on forced alignments and speech recognizer output, and the system remains linear in the length of the input discourse.

2. MODEL

In sentence segmentation, we are given a sequence of words $w = w_1 \dots w_n$ and we are asked to output a segmentation list $b = b_0 \dots b_k$, where each b_i is the index where the i-th sentence ends ($b_0 = 0$ and $b_k = n$). Note that this is different from the usual boundary versus non-boundary notation. Assume we have a vector of local features at each point i, $f(w, i)$, which represents the local information at that point. As detailed below, these features are prosodic (such as pause duration) and lexical (n-gram identity) for our model. Then a local log-linear model of sentence segmentation takes the form

$$P(b|w) \propto \exp\left(\sum_{i=1}^{|b|-1} \theta^\top f(w, b_i)\right) \quad (1)$$

This work is supported by the Defense Advanced Research Projects Agency (DARPA) GALE project, under Contract No. HR0011-06-C-0023. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of DARPA.

Authorized licensed use limited to: International Computer Science Inst (ICSI). Downloaded on May 07,2010 at 17:34:55 UTC from IEEE Xplore. Restrictions apply.

where θ is a weight vector scoring the relative contributions of the various features.[1] With just local features, the optimal segmentation of an entire sequence breaks at points where $\theta^\top f(w, i) > 0$. At this point, the model is simply a collection of independent logistic regressions at each position.

We would like to additionally assign scores to *spans* (i, j) based on factors such as their grammaticality as sentences. Formally, we add features $g(w, i, j)$ to each span $w_i \dots w_j$. In the present work, the only span feature will be the log probability from the parser, as described below, but in principle arbitrary features are possible. The model with pair features takes the form

$$P(b|w) \propto \exp\left(\sum_{i=1}^{|b|-1} \theta^\top f(w, b_i) + \theta^\top g(w, b_{i-1}, b_i)\right) \quad (2)$$

Here, b_{i-1} is the index of the word before the first word of sentence i. Essentially, the (log) score of a hypothesis is the weighted combination of the local features and the span features for the segmentation points used in that hypothesis. The model is now formally a semi conditional random field (CRF) where the segment features are the span scores and the node features are the local scores. Finding the optimal b can be done using a minor variant of the standard Viterbi algorithm and computing expectations over both segmentation points and spans can be done using a corresponding forward-backward algorithm [6].

Of course, features can be associated with higher-order contexts, such as adjacent sentence spans. The model of [5] includes such arbitrary features and as a result loses the efficient inference we maintain. We therefore see the present model as an effective point where central global information can be incorporated yet where decoding is practical without making both short sentence and hard discourse break assumptions.

2.1. Local boundary features

Our local features include both lexical and prosodic features around the boundary considered. Lexical features are made of word n-grams and prosodic features include pause and phoneme duration, speaker normalized pitch and energy on both sides of the boundary, discontinuity of those features across the boundary and estimated speaker changes (essentially the same features as used in [1]).

The weights for these local features alone can be learned using a variety of methods; we used a boosting approach in which features are incrementally added one at a time.[2] Note that while boosting minimizes not the log loss but rather the exponential loss, we found it convenient for its incremental properties (we return to this point below). This classifier is referred to as *local* in the results.

2.2. Syntactic span features

Our only span feature is the log-probability of that span as a sentence according to a (syntactic) language model. In principle, any language model could be used as a grammaticality model. In fact, if we used an n-gram model over words, this formulation would be rather similar to the popular HELM [4], though it would not exploit the efficiencies that local language models allow. However, we use a syntactic language model based on a probabilistic context free grammar (PCFG). For a PCFG, the probability of a sentence hypothesis s can be computed by summing the probabilities of all valid parse trees t covering that sentence:

$$P(s) = \sum_{t:s} P(t) = \sum_{t:s} \prod_{r \in t} P(r) \quad (3)$$

where the probability of a parse tree is just the product of the probabilities of the productions r used to construct it [7]. This summation can be computed in time cubic in the length of the sentence, and can be vastly accelerated using many pruning techniques established in the parsing literature [8].

The ability of the underlying grammar to distinguish grammatically well formed sentences from ungrammatical sentence chunks will be crucial for the final performance of the system. In our experiments we use the grammar of the Berkeley parser [8].[3] This grammar gives state-of-the-art parsing performance on a variety of languages, and is because of its generative nature very well suited as syntactic language model. The grammar is automatically learned from a treebank following a latent variable approach by iteratively splitting non-terminals to better represent the data (for example, contextual variants of "noun phrase" are automatically inferred).

To improve the adequacy of the grammars to speech data, we retrained it for our purposes. Since ASR output has no case information and sentence internal punctuation, we stripped off all sentence internal punctuation and lowercased all words before training the parser on the Wall Street Journal (WSJ) section of the Penn Treebank. To further adapt the parser to the speech domain, we decided to self-train the parser on speech data: we parsed the speech training data using a grammar learned on correct parse trees from the treebank, and trained a new grammar using this unsupervised ground truth. While this process will likely introduce wrong parses, it serves as a form of domain adaptation and improves the parser's ability to handle disfluencies and other speech specific phenomena which do not occur in textual corpora.

Note that once we have span features, training, in principle, becomes more complex. However, in practice, we had only a single span feature. Therefore, we found it effective to first train on the local features only, freeze their weights, and perform a one-dimensional search for the optimum weight for the single span feature. This approach parallels the incremental boosting approach used for the local features and gave good results (see below) with rapid training. This training regime is also reminiscent of the piecewise training for CRFs presented in [9].

2.3. HELM span features

To contrast our parser-informed language model, we also combine the local features with an HELM feature (described in [1]) on the spans in place of the parser score. A HELM is a generative model of the sequence of words interleaved with boundary and non-boundary events $P(w_1 e_1 \dots w_n e_n)$ where e_i is simply mapped from sentence indexes in equation 2. The HELM can be used alone or in conjunction with a local classifier by converting its output to pseudo-likelihoods. Additionally, word-level posterior probabilities can be computed using forward-backward decoding, allowing to effectively use the HELM output as a local feature in our model (and combine it with the grammar).

[1] Note that in this formulation, only the positions where b does segment w have their features explicitly included in the scoring. However, it is equivalent to a model in which non-segmented positions are also scored via negated feature vectors.

[2] Implementation available at http://code.google.com/p/icsiboost

[3] Implementation available at http://nlp.cs.berkeley.edu

Authorized licensed use limited to: International Computer Science Inst (ICSI). Downloaded on May 07,2010 at 17:34:55 UTC from IEEE Xplore. Restrictions apply.

3. HYPOTHESIS LATTICE PRUNING

Given a sequence of n words, we want to find the most likely sequence of sentences according to our model as stated in Eq. 2. However, evaluating all possible sentence hypotheses is prohibitive since there are up to n^2 possible sentence spans and the parser runs in n^3 time. To get a reasonable processing time, we follow a two pass approach where a local word boundary model using prosodic and n-gram features is used to construct a lattice of sentence hypotheses. We use the local model only to rule out unlikely sentence boundaries, but not to make hard decisions about the existence of a sentence boundary.

This is in contrast to [5], where the local model is not only used for constructing a lattice but also for making hard decisions. Those hard decisions are necessary because they use a reranking approach which scores entire segmentation paths. The number of paths however is exponential in the number of potential boundaries, requiring them to frequently make hard decisions and allowing only a small number of rerankable boundaries to remain in between. While their approach is justified in the telephone speech domain, where speaker turns are rarely long, one needs a more tractable solution to handle general sentence segmentation tasks.

We construct our lattice in a way which attempts to generate the best lattice in terms of oracle performance. We do not need to restrict the size of sub-lattices (unlike [5]) because the model processes features at the sentence level and can therefore be efficiently globally decoded. In addition, we exploit two practical pruning strategies. First, we restrict the maximum sentence length to 50 words. Second, rule out sentence boundaries that get assigned a very low score by the local model. Both greatly decrease the number of spans the parser must score with little cost in search optimality. Figure 1 shows the sentence hypothesis lattice for a sequence of words, where candidate sentence spans are shown by arcs. For example, the score of a sentence boundary (that is, the posterior probability of a sentence boundary) in the local model after the word *new* is very low, hence no sentence boundary is allowed at that location for the full model. An (incorrect) sentence boundary after the word *president* is very likely according to the local classifier, and so that point is left as a candidate boundary. While the local model would incorrectly split here, the addition of the global syntactic feature removes this error.

4. EXPERIMENTS

The data set used for our experiments is a subset of the TDT4 English data.[4] It consists of 200 hours of close-captioned broadcast news. We use one million words for training, 83k words as a development set and 84k words as the test set. The average sentence length is 15 words. All the data is recognized using SRI's English broadcast news ASR system, and the word error rate is estimated to be around 18%.

For grammaticality assessment, we rely on grammars from the Berkeley parser [8] as described in section 2. To evaluate the importance of parsing accuracy, we present experiments with several grammars. The first is a refined grammar which was trained on the WSJ using the algorithm described in [8]. The second is the WSJ grammar after self-training on speech data. We refer to these grammars as WSJ and TDT4, respectively. A third grammar, TDT4*, is a baseline grammar that was obtained by the same domain adaptation procedure but without splitting the grammar symbols. We also compare the grammar approach to a hidden event language model (HELM) [4] trained on the same amount of data, as HELM represents the most popular alternative source of a global feature.

[4]LDC publication LDC2005S11.

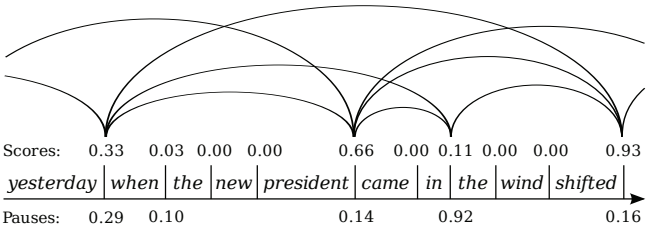

Fig. 1. Illustration of the scores output by a local classifier and the resulting lattice.

System	Dev.	Test
HELM alone	56.54	56.65
HELM w/ lattice	74.73	73.42
WSJ	77.32	76.00
TDT4	77.98	76.78
TDT4*	71.06	69.86

Table 1. Our models with global syntactic information outperform a HELM on the same lattice when a refined grammar is used (local features are excluded here).

Figure 2 shows the maximum recall and precision that can be obtained on the development set for a given size of the hypothesis lattice. Even for a number of arcs equal to only the number of words (i.e. a size factor of 1), both values are much higher than the local model alone, leaving lots of space for improvement. These graphs are plotted to show the upper-bounds that can be achieved for this task, if the grammar only makes correct decisions. At a size factor of 1, it is also noticeable that the operating point is close to the knee of the curve, meaning that a larger lattice will only slightly improve the oracle. Therefore, all the following experiments are performed on lattices of this size factor. The figure also shows the oracle for lattices constructed from the model that is the combination of the local model with a HELM. The intuition for running such an experiment is that even though word n-grams and syntactic parses are not orthogonal sources of information, one might benefit from the other.

Table 1 presents the performance of the HELM alone and its comparison to the grammar, as a language model, under fair conditions (that is, when it is given the same pruned lattice and local features are ignored, effectively running the model on span features only). These experiments try to factor out the effect of prosodic information and compare the linguistic modeling properties of the parser and the HELM. While the HELM on its own performs poorly, it is more competitive on a lattice constructed by the local model, which indirectly imports the value of those features. When given the same lattice, the WSJ and TDT4 grammars significantly outperform the HELM, showing the potential of grammatical features even by themselves. The low accuracy of the baseline grammar TDT4* indicates that high treebank parsing performance is crucial for precise syntactic modeling.

Table 2 presents F-measure results for different feature combinations. Unlike in the previous table, prosodic information not only affects the lattice but also the global model through the local classifier features (local features alone or HELM combined with local features). The combined versions of the WSJ and the TDT4 grammar

Authorized licensed use limited to: International Computer Science Inst (ICSI). Downloaded on May 07,2010 at 17:34:55 UTC from IEEE Xplore. Restrictions apply.

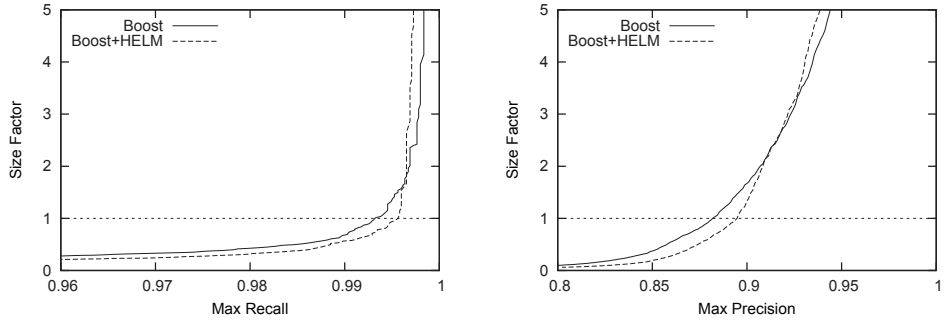

Fig. 2. Oracle results for the lattice made from local decisions. Maximum fmeasure and recall are much higher than the local model performance ($F = 76.33$ $R = 81.71$ $P = 71.61$). The operating point at a size factor of one is denoted by an horizontal line.

Grammar	Features	Ref.	ASR
-	local	76.14	62.87
WSJ		78.26	64.65
TDT4		79.55	**66.04**
TDT4*		74.51	62.08
-	local + HELM	78.51	64.62
WSJ		80.00	66.52
TDT4		80.48	**67.32**
TDT4*		77.90	63.98

Table 2. F-measure results for local features alone, local combined with HELM, and their combinations with grammaticality features estimated from different grammars: the original grammar (WSJ), an adapted grammar (TDT4), and an adapted but unsplit grammar (TDT4*). Results are presented on the test set on both reference text and ASR output.

outperform the local model significantly.[5] The unsupervised setting allows even greater performance and this grammar, when combined with Boosting, is significantly better than the HELM combined with local features, showing the benefits of the approach over the classical language model. The best result is achieved by combining local features, HELM features, and the adapted grammar scores.

The baseline grammar, TDT4*, trained using non-lexicalized, non-split production rules, seems to be less effective as a language model for sentence segmentation. The resulting performance is below the local models alone and would probably be even lower in the absence of local lattice pruning. Improved parsing performance seems to substantially help sentence segmentation.

Measuring the grammaticality of the sentence hypotheses is an effective feature for sentence segmentation. However, there are cases where the grammar alone cannot disambiguate (hence the necessity of punctuation in written language). Consider the following example: "John left at two. Jane will arrive". The sentence boundary could very well be placed differently, associating the temporal mark to Jane's arrival: "John left. At two, Jane will arrive." Both sentence hypotheses are grammatically correct, while the information conveyed is different. When spoken, the former example will likely contain a pause between "two" and "Jane" while in the latter, a pause can be placed between "left" and "At" and also in place of the comma. This illustrates why local acoustic features add value over lexically derived grammaticality.

[5]Two-tailed test at 95% using the sigf package from http://www.coli.uni-saarland.de/∼pado/sigf.html

5. CONCLUSION

We introduced a sentence segmentation model which combines global syntactic scores with local prosodic and lexical scores in a log-linear framework. Local predictions are used to generate a pruned sentence hypotheses lattice. In contrast to previous work, we do not require the local model to break up a given word sequence into small lattices by making hard decisions at regular intervals, but rather prune only where highly confident. Each sentence hypothesis is then scored with a state-of-the-art syntactic parser and the globally most likely segmentation is efficiently extracted via dynamic programming. Experimental results show that this model outperforms the most popular long-range sequence model for sentence segmentation (HELM [4]) on reference text and ASR output. Furthermore we show that the combination of HELM and syntactic scores yields additional improvements, indicating that they are capturing different phenomena.

6. REFERENCES

[1] M. Zimmerman, D. Hakkani-Tür, J. Fung, N. Mirghafori, L. Gottlieb, E. Shriberg, and Y. Liu, "The ICSI+ Multilingual Sentence Segmentation System," in *Proc. ICSLP*, 2006.

[2] J. Huang and G. Zweig, "Maximum Entropy Model for Punctuation Annotation from Speech," in *Proc. ICSLP*, 2002.

[3] A. Srivastava and F. Kubala, "Sentence Boundary Detection in Arabic Speech," in *Proc. Eurospeech*, 2003.

[4] A. Stolcke and E. Shriberg, "Automatic linguistic segmentation of conversational speech," *Proc. ICSLP*, vol. 2, 1996.

[5] B. Roark, Y. Liu, M. Harper, R. Stewart, M. Lease, M. Snover, I. Shafran, B. Dorr, J. Hale, A. Krasnyanskaya, and L. Yung, "Reranking for sentence boundary detection in conversational speech," in *Proc. ICASSP*, 2006.

[6] S. Sarawagi and W.W. Cohen, "Semi-markov conditional random fields for information extraction," *Advances in Neural Information Processing Systems*, vol. 17, pp. 1185–1192, 2005.

[7] M. Johnson, "PCFG models of linguistic tree representations," *Computational Linguistics*, vol. 24, no. 4, pp. 613–632, 1998.

[8] S. Petrov and D. Klein, "Learning and inference for hierarchically split PCFGs," in *Proc. AAAI*, 2007.

[9] C. Sutton and A. McCallum, "Piecewise pseudolikelihood for efficient crf training," in *Proc. ICML*, 2007.

Authorized licensed use limited to: International Computer Science Inst (ICSI). Downloaded on May 07,2010 at 17:34:55 UTC from IEEE Xplore. Restrictions apply.