Appropriately Handled Prosodic Breaks Help PCFG Parsing

Zhongqiang Huang[1], **Mary Harper**[1,2]
[1]Laboratory for Computational Linguistics and Information Processing
Institute for Advanced Computer Studies
University of Maryland, College Park, MD USA
[2]Human Language Technology Center of Excellence
Johns Hopkins University, Baltimore, MD USA
{zqhuang,mharper}@umiacs.umd.edu

Abstract

This paper investigates using prosodic information in the form of ToBI break indexes for parsing spontaneous speech. We revisit two previously studied approaches, one that hurt parsing performance and one that achieved minor improvements, and propose a new method that aims to better integrate prosodic breaks into parsing. Although these approaches can improve the performance of basic probabilistic context free grammar (PCFG) parsers, they all fail to produce fine-grained PCFG models with latent annotations (PCFG-LA) (Matsuzaki et al., 2005; Petrov and Klein, 2007) that perform significantly better than the baseline PCFG-LA model that does not use break indexes, partially due to mis-alignments between automatic prosodic breaks and true phrase boundaries. We propose two alternative ways to restrict the search space of the prosodically enriched parser models to the n-best parses from the baseline PCFG-LA parser to avoid egregious parses caused by incorrect breaks. Our experiments show that all of the prosodically enriched parser models can then achieve significant improvement over the baseline PCFG-LA parser.

1 Introduction

Speech conveys more than a sequence of words to a listener. An important additional type of information that phoneticians investigate is called prosody, which includes phenomena such as pauses, pitch, energy, duration, grouping, and emphasis. For a review of the role of prosody in processing spoken language, see (Cutler et al., 1997). Prosody can help with the disambiguation of lexical meaning (via accents and tones) and sentence type (e.g., yes-no question versus statement), provide discourse-level information like focus, prominence, and discourse segment, and help a listener to discern a speaker's emotion or hesitancy, etc. Prosody often draws a listener's attention to important information through contrastive pitch or duration patterns associated words or phrases. In addition, prosodic cues can help one to segment speech into chunks that are hypothesized to have a hierarchical structure, although not necessarily identical to that of syntax. This suggests that prosodic cues may help in the parsing of speech inputs, the topic of this paper.

Prosodic information such as pause length, duration of words and phones, pitch contours, energy contours, and their normalized values have been used in speech processing tasks like sentence boundary detection (Liu et al., 2005). In contrast, other researchers use linguistic encoding schemes like ToBI (Silverman et al., 1992), which encodes tones, the degree of juncture between words, and prominence symbolically. For example, a simplified ToBI encoding scheme uses the symbol 4 for major intonational breaks, p for hesitation, and 1 for all other breaks (Dreyer and Shafran, 2007). In the literature, there have been several attempts to integrate prosodic information to improve parse accuracy of speech transcripts. These studies have used either quantized acoustic measurements of prosody or automatically detected break indexes.

Gregory et al. (2004) attempted to integrate quantized prosodic features as additional tokens in the same manner that punctuation marks are added into text. Although punctuation marks can significantly improve parse accuracy of newswire text, the quantized prosodic tokens were found harm-

Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the ACL, pages 37–45,
Los Angeles, California, June 2010. ©2010 Association for Computational Linguistics

ful to parse accuracy when inserted into human-generated speech transcripts of the Switchboard corpus. The authors hypothesized that the inserted pseudo-punctuation break n-gram dependencies in the parser model, leading to lower accuracies. However, another possible cause is that the prosody has not been effectively utilized due to the fact that it is overloaded; it not only provides information about phrases, but also about the state of the speaker and his/her sentence planning process. Hence, the prosodic information may at times be more harmful than helpful to parsing performance.

In a follow-on experiment, Kahn et al. (2005), instead of using raw quantized prosodic features, used three classes of automatically detected ToBI break indexes (1, 4, or p) and their posteriors. Rather than directly incorporating the breaks into the parse trees, they used the breaks to generate additional features for re-ranking the n-best parse trees from a generative parsing model trained without prosody. They were able to obtain a significant 0.6% improvement on Switchboard over the generative parser, and a more modest 0.1% to 0.2% improvement over the reranking model that also utilizes syntactic features.

Dreyer and Shafran (2007) added prosodic breaks into a generative parsing model with latent variables. They utilized three classes of ToBI break indexes (1, 4, and p), automatically predicted by the approach described in (Dreyer and Shafran, 2007; Harper et al., 2005). Breaks were modeled as a sequence of observations parallel to the sentence and each break was generated by the preterminal of the preceding word, assuming that the observation of a break, b, was conditionally independent of its preceding word, w, given preterminal X:

$$P(w, b|X) = P(w|X)P(b|X) \tag{1}$$

Their approach has advantages over (Gregory et al., 2004) in that it does not break n-gram dependencies in parse modeling. It also has disadvantages in that the breaks are modeled by preterminals rather than higher level nonterminals, and thus cannot directly affect phrasing in a basic PCFG grammar. However, they addressed this independence drawback by splitting each nonterminal into latent tags so that the impact of prosodic breaks could be percolated into the phrasing process through the interaction of latent tags. They achieved a minor 0.2% improvement over their baseline model without prosodic cues and also found that prosodic breaks can be used to build more compact grammars.

In this paper, we re-investigate the models of (Gregory et al., 2004) and (Dreyer and Shafran, 2007), and propose a new way of modeling that can potentially address the shortcomings of the two previous approaches. We also attribute part of the failure or ineffectiveness of the previously investigated approaches to errors in the quantized prosodic tokens or automatic break indexes, which are predicted based only on acoustic cues and could misalign with phrase boundaries. We illustrate that these prosodically enriched models are in fact highly effective if we systematically eliminate bad phrase and hesitation breaks given their projection onto the reference parse trees. Inspired by this, we propose two alternative rescoring methods to restrict the search space of the prosodically enriched parser models to the n-best parses from the baseline PCFG-LA parser to avoid egregious parse trees. The effectiveness of our rescoring method suggests that the reranking approach of (Kahn et al., 2005) was successful not only because of their prosodic feature design, but also because they restrict the search space for reranking to n-best lists generated by a syntactic model alone.

2 Experimental Setup

Due to our goal of investigating the effect of prosodic information on the accuracy of state of the art parsing of conversational speech, we utilize both Penn Switchboard (Godfrey et al., 1992) and Fisher treebanks (Harper et al., 2005; Bies et al., 2006), for which we also had automatically generated break indexes from (Dreyer and Shafran, 2007; Harper et al., 2005)[1]. The Fisher treebank is a higher quality parsing resource than Switchboard due to its greater use of audio and refined specifications for sentence segmentation and disfluency markups, and so we utilize its eval set for our parser evaluation; the first 1,020 trees (7,184 words) were used for development and the remaining 3,917 trees (29,173 words) for evaluation. We utilized the Fisher dev1 and dev2 sets containing 16,519 trees (112,717 words) as the main training data source and used the Penn Switchboard

[1] A small fraction of words in the Switchboard treebank could not be aligned with the break indexes that were produced based on a later refinement of the transcription. We chose not to alter the Switchboard treebank, so in cases of missing break values, we heuristically added break *1* to words in the middle of a sentence and *4* to words that end a sentence.

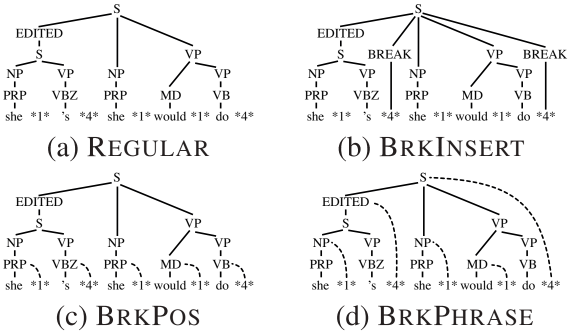
treebank containing 110,504 trees (837,863 words) as an additional training source to evaluate the effect of training data size on parsing performance. The treebank trees are normalized by downcasing all terminal strings and deleting punctuation, empty nodes, and nonterminal-yield unary rules that are not related to edits.

We will compare[2] three prosodically enriched PCFG models described in the next section, with a baseline PCFG parser. We will also utilize a state of the art PCFG-LA parser (Petrov and Klein, 2007; Huang and Harper, 2009) to examine the effect of prosodic enrichment[3]. Unlike (Kahn et al., 2005), we do not remove EDITED regions prior to parsing because parsing of EDITED regions is likely to benefit from prosodic information. Also, parses from all models are compared with the gold standard parses in the Fisher evaluation set using SParseval bracket scoring (Harper et al., 2005; Roark et al., 2006) without flattening the EDITED constituents.

3 Methods of Integrating Breaks

Rather than using quantized raw acoustic features as in (Gregory et al., 2004), we use automatically generated ToBI break indexes as in (Dreyer and Shafran, 2007; Kahn et al., 2005) as the prosodic cues, and investigate three alternative methods of modeling prosodic breaks. Figure 1 shows parse trees for the four models for processing the spontaneous speech transcription *she's she would do*, where the speaker hesitated after saying *she's* and then resumed with another utterance *she would do*. Each word input into the parser has an associated break index represented by the symbol 1, 4, or p enclosed in asterisks indicating the break after the word. The automatically detected break *4* after the contraction is a strong indicator of an intonational phrase boundary that might provide helpful information for parsing if modeled appropriately. Figure 1 (a) shows the reference parse tree (thus the name REGULAR) where the break indexes are not utilized.

The first method to incorporate break indexes, BRKINSERT, shown in Figure 1 (b), treats the *p* and *4* breaks as tokens, placing them under the highest nonterminal nodes so that the order of words and breaks remain unchanged in the terminals. This is similar to (Gregory et al., 2004), except that automatically generated ToBI breaks are used rather than quantized raw prosodic tokens.



(a) REGULAR (b) BRKINSERT

(c) BRKPOS (d) BRKPHRASE

Figure 1: Modeling Methods

The second method, BRKPOS, shown in Figure 1 (c), treats breaks as a sequence of observations parallel to the words in the sentence as in (Dreyer and Shafran, 2007). The dotted edges in Figure 1 (c) represent the relation between preterminals and prosodic breaks, and we call them *prosodic rewrites*, with analogy to grammar rewrites and lexical rewrites. The generation of words and prosodic breaks is assumed to be conditionally independent given the preterminal, as in Equation 1.

The third new method, BRKPHRASE, shown in Figure 1 (d), also treats breaks as a sequence of observations parallel to the sentence; however, rather than associating the prosodic breaks with the preterminals, each is generated by the highest nonterminal (including preterminal) in the parse tree that covers the preceding word as the right-most terminal. The observation of break, b, is assumed to be conditionally independent of grammar or lexical rewrite, r, given the nonterminal X:

$$P(r, b|X) = P(r|X)P(b|X) \tag{2}$$

The relation is indicated by the dotted edges in Figure 1 (d), and it is also called a *prosodic rewrite*. The potential advantage of BRKPHRASE is that it does not break or fragment n-gram dependencies of the grammar rewrites, as in the BRKINSERT method, and it directly models the dependency between breaks and phrases, which the BRKPOS method explicitly lacks.

4 Model Training

Since automatically generated prosodic breaks are incorporated into the parse trees deterministi-

[2] We use Bikel's randomized parsing evaluation comparator to determine the significance ($p < 0.005$) of the difference between two parsers' outputs.

[3] Due to the randomness of parameter initialization in the learning of PCFG-LA models with increasing numbers of latent tags, we train each latent variable grammar with 10 different seeds and report the average F score on the evaluation set.

cally for all of the three enrichment methods (BRKINSERT, BRKPOS, and BRKPHRASE), training a basic PCFG is straightforward; we simply pull the counts of grammar rules, lexical rewrites, or prosodic rewrites from the treebank and normalize them to obtain their probabilities.

As is well known in the parsing community, the basic PCFG does not provide state-of-the-art performance due to its strong independence assumptions. We can relax these assumptions by explicitly incorporating more information into the conditional history, as in Charniak's parser (Charniak, 2000); however, this would require sophisticated engineering efforts to decide what to include in the history and how to smooth probabilities appropriately due to data sparsity. In this paper, we utilize PCFG-LA models (Matsuzaki et al., 2005; Petrov and Klein, 2007) that split each nonterminal into a set of latent tags and learn complex dependencies among the latent tags automatically during training. The resulting model is still a PCFG, but it is probabilistically context free on the latent tags, and the interaction among the latent tags is able to implicitly capture higher order dependencies among the original nonterminals and observations. We follow the approach in (Huang and Harper, 2009) to train the PCFG-LA models.

5 Parsing

In a basic PCFG without latent variables, the goal of maximum probability parsing is to find the most likely parse tree given a sentence based on the grammar. Suppose our grammar is binarized (so it contains only unary and binary grammar rules). Given an input sentence $w_1^n = w_1, w_2, \cdots, w_n$, the inside probability, $P(i, j, X)$, of the most likely sub-tree that is rooted at nonterminal X and generates subsequence w_i^j can be computed recursively by:

$$P(i,j,X) = \max(\max_Y P(i,j,Y)P(X \to Y),$$
$$\max_{i<k<j,Y,Z} P(i,k,Y)P(k+1,j,Z)P(X \to Y\,Z)) \quad (3)$$

Backtracing the search process then returns the most likely parse tree for the REGULAR grammar.

The same parsing algorithm can be directly applied to the BRKINSERT grammar given that the break indexes are inserted appropriately into the input sentence as additional tokens. Minor modification is needed to extend the same parsing algorithm to the BRKPOS grammar. The only difference is that the inside probability of a preterminal is set according to Equation 1. The rest of the algorithm proceeds as in Equation 3.

However, parsing with the BRKPHRASE grammar is more complicated because whether a nonterminal generates a break or not is determined by whether it is the highest nonterminal that covers the preceding word as its right-most terminal. In this case, the input observation also contains a sequence of break indexes $b_1^n = b_1, b_2, \cdots, b_n$ that is parallel to the input sentence $w_1^n = w_1, w_2, \cdots, w_n$. Let $P(i, j, X, 0)$ be the probability of the most likely sub-tree rooted at nonterminal X over span (i, j) that generates word sequence w_i^j, as well as break index sequence b_i^{j-1}, excluding b_j. According to the independence assumption in Equation 2, with the addition of prosodic edge $X \to b_j$, the same sub-tree also has the highest probability, denoted by $P(i, j, X, 1)$, of generating word sequence w_i^j together with the break index sequence b_i^j. Thus we have:

$$P(i,j,X,1) = P(i,j,X,0)P(b_j|X) \quad (4)$$

The structural constraint that a break index is only generated by the highest nonterminal that covers the preceding word as the right-most terminal enables a dynamic programming algorithm to compute $P(i, j, X, 0)$ and thus $P(i, j, X, 1)$ efficiently. If the sub-tree (without the prosodic edge that generates b_j) over span (i, j) is constructed from a unary rule rewrite $X \to Y$, then the root nonterminal Y of some best sub-tree over the same span (i, j) can not generate break b_j because it has a higher nonterminal X that also covers word w_j as its right-most terminal. If the sub-tree is constructed from a binary rule rewrite $X \to Y Z$, then the root nonterminal Y of some best sub-tree over some span (i, k) will generate break b_k because Y is the highest nonterminal that covers word w_k as the right-most terminal[4]. In contrast, the root nonterminal Z of some best subtree over some span $(k+1, j)$ can not generate break b_j because Z has a higher nonterminal X that also covers word w_j as its right-most terminal. Hence,

[4] Use of left-branching is required for the BRKPHRASE method to ensure that the prosodic breaks are associated with the original nonterminals, not intermediate nonterminals introduced by binarization. Binarization is needed for efficient parametrization of PCFG-LA models and left- versus right-branching binarization does not significantly affect model performance; hence, we use left-branching for all models.

$P(i,j,X,1)$ and $P(i,j,X,0)$ can be computed recursively by Equation 4 above and Equation 5 below:

$$P(i,j,X,0) = \max(\max_{Y} P(i,j,Y,0)P(X \rightarrow Y),$$
$$\max_{i<k<j,Y,Z} P(i,k,Y,1)P(k+1,j,Z,0)P(X \rightarrow Y\ Z)) \qquad (5)$$

Although dynamic programming algorithms exist for maximum probability decoding of basic PCFGs without latent annotations for all four methods, it is an NP hard problem to find the most likely parse tree using PCFG-LA models. Several alternative decoding algorithms have been proposed in the literature for parsing with latent variable grammars. We use the best performing max-rule-product decoding algorithm, which searches for the best parse tree that maximizes the product of the posterior rule (either grammar, lexical, or prosodic) probabilities, as described in (Petrov and Klein, 2007) for our models with latent annotations and extend the dynamic parsing algorithm described in Equation 5 for the BRKPHRASE grammar with latent annotations.

6 Results on the Fisher Corpus

6.1 Prosodically Enriched Models

Table 1 reports the parsing accuracy of the four basic PCFGs without latent annotations when trained on the Fisher training data. All of the grammars have a low F score of around 65% due to the overly strong and incorrect independence assumptions. We observe that the BRKPHRASE grammar benefits most from breaks, significantly improving the baseline accuracy from 64.9% to 67.2%, followed by the BRKINSERT grammar, which at 66.2% achieves a smaller improvement. The BRKPOS grammar benefits the least among the three because breaks are attached to the preterminals and thus have less impact on phrasing due to the independence assumptions in the basic PCFG. In contrast, both the BRKPHRASE and BRKINSERT methods directly model the relationship between breaks and phrase boundaries through governing nonterminals; however, the BRKPHRASE method does not directly change any of the grammar rules in contrast to the BRKINSERT method that more or less breaks n-gram dependencies and fragments rule probabilities.

The bars labeled DIRECT in Figure 2 report the parsing performance of the four PCFG-LA models trained on Fisher. The introduction of latent annotations significantly boosts parsing accuracies, providing relative improvements ranging from 16.8% up to 19.0% when trained on Fisher training data due to the fact that the PCFG-LA models are able to automatically learn more complex dependencies not captured by basic PCFGs.

REGULAR	BRKINSERT	BRKPOS	BRKPHRASE
64.9	66.2	65.2	**67.2**

Table 1: Fisher evaluation parsing results for the basic PCFGs without latent annotations trained on the Fisher training set.

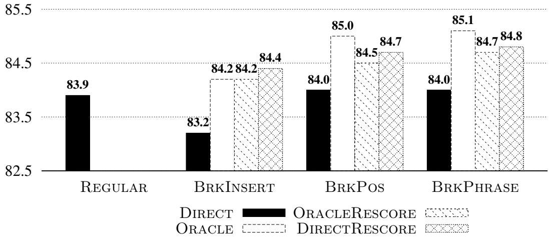


Figure 2: Parsing results on the Fisher evaluation set of the PCFG-LA models trained on the Fisher training data. The DIRECT bars represent direct parsing results for models trained and evaluated on the original data, ORACLE bars for models trained and evaluated on the modified oracle data (see Subsection 6.2), and the ORACLERESCORE and DIRECTRESCORE bars for results of the two rescoring approaches (described in Subsection 6.3) on the original evaluation data.

However, the prosodically enriched methods do not significantly improve upon the REGULAR baseline after the introduction of latent annotations. The BRKPHRASE method only achieves a minor insignificant 0.1% improvement over the REGULAR baseline; whereas, the BRKINSERT method is a significant 0.7% worse than the baseline. Similar results for BRKINSERT were reported in (Gregory et al., 2004), where they attributed the degradation to the fact that the insertion of the prosodic "punctuation" breaks the n-gram dependencies. Another possible cause is that the insertion of "bad" breaks that do not align with true phrase boundaries hurts performance more than the benefits gained from "good" breaks due to the tightly integrated relationship between phrases and breaks. For the BRKPOS method, the impact of break indexes is implicitly percolated to the nonterminals through the interaction among latent tags, as discussed in (Dreyer and Shafran, 2007), and its performance may thus be less affected by the "bad" breaks. With latent annotations (in contrast to the basic PCFG), the model is now significantly better than BRKINSERT and is on par with BRKPHRASE.

6.2 Models with Oracle Breaks

In order to determine whether "bad" breaks limit the improvements in parsing performance from prosodic enrichment, we conducted a simple oracle experiment where all *p* and *4* breaks that did not align with phrase boundaries in the treebank were systematically converted to *1* breaks[5]. When trained and evaluated on this modified oracle data, all three prosodically enriched latent variable models improve by about 1% and were then able to achieve significant improvements over the REGULAR PCFG-LA baseline, as shown by the bars labeled ORACLE in Figure 2. It should be noted, however, that the BRKINSERT method is much less effective than the other two methods in the oracle experiment, suggesting that broken n-gram dependencies affect the model in addition to the erroneous breaks.

6.3 N-Best Re-Scoring

As mentioned previously, prosody does not only provide information about phrases, but also about the state of the speaker and his/her sentence planning process. Given that our break detector utilizes only acoustic knowledge to predict breaks, the recognized *p* and *4* breaks may not correctly reflect hesitations and phrase boundaries. Incorrectly recognized breaks could hurt parsing more than the benefit brought from the correctly recognized breaks, as demonstrated by superior performance of the prosodically enhanced models in the oracle experiment. We next describe two alternative methods to make better use of automatic breaks.

In the first approach, which is called ORACLERESCORE, we train the prosodically enhanced grammars on cleaned-up break-annotated training data, where misclassified *p* and *4* breaks are converted to *1* breaks (as in the oracle experiment). If these grammars were used to directly parse the test sentences with automatically detected (unmodified) breaks, the results would be quite poor due to mismatch between the training and testing conditions. However, we can automatically bias against potentially misclassified *p* and *4* breaks if we utilize information provided by n-best parses from the baseline REGULAR PCFG-LA grammar. For each hypothesized parse tree in the n-best list, the *p* and *4* breaks that do not align with the phrase boundaries of the hypothesized parse tree are converted to *1* breaks, and then a new score is computed using the product of posterior rule probabilities[6], as in the max-rule-product criterion, for the hypothesized parse tree using the grammars trained on the cleaned-up training data. In this approach, we convert the posterior probability, $P(T|W,B)$, of parse tree T given words W and breaks B to $P(B'|W,B)P(T|W,B')$, where B' is the new break sequence constrained by T, and simplify it to $P(T|W,B')$, assuming that conversions to a new sequence of breaks as constrained by a hypothesized parse tree are equally probable given the original sequence of breaks. We consider this to be a reasonable assumption for a small n-best ($n = 50$) list with reasonably good quality.

In the second approach, called DIRECTRESCORE, we train the prosodically enhanced PCFG-LA models using unmodified, automatic breaks, and then use them to rescore the n-best lists produced by the REGULAR PCFG-LA model to avoid the poorer parse trees caused by fully trusting automatic break indexes. The size of the n-best list should not be too small or too large, or the results would be like directly parsing with REGULAR when $n = 1$ or with the prosodically enriched model when $n \to \infty$.

The ORACLERESCORE and DIRECTRESCORE bars in Figure 2 report the performance of the prosodically enriched models with the corresponding rescoring method. Both methods use the same 50-best lists produced by the baseline REGULAR PCFG-LA model using the max-rule-product criterion. Both rescoring methods produce significant improvements in the performance of all three prosodically enriched PCFG-LA models. The previously ineffective (0.7% worse than REGULAR) BRKINSERT PCFG-LA model is now 0.3% and 0.5% better than the REGULAR baseline using the ORACLERESCORE and DIRECTRESCORE approaches, respectively. The best performing BRKPOS and BRKPHRASE rescoring models are 0.6-0.9% better than the REGULAR baseline. It is interesting to note that rescoring with models trained on cleaned up prosodic breaks is somewhat poorer

[5] Other sources of errors include misclassification of *p* breaks as *1* or *4* and misclassification of *4* breaks as *1* or *p*. Although these errors are not repaired in the oracle experiment, fixing them could potentially provide greater gains.

[6] The product of posterior rule probabilities of a parse tree is more suitable for rescoring than the joint probability of the parse tree and the observables (words and breaks) because the breaks are possibly different for different trees.

than models trained using all automatic breaks.

7 Models with Augmented Training Data

Figure 3 reports the evaluation results for models that are trained on the combination of Fisher and Switchboard training data. With the additional Switchboard training data, the nonterminals can be split into more fine-grained latent tags, enabling the learning of deeper dependencies without over-fitting the limited sized Fisher training data. This improved all models by at least 2.6% absolute. Note also that the patterns observed for models trained using the larger training set are quite similar to those from using the smaller training set in Figure 2. The prosodically enriched models all benefit significantly from the oracle breaks and from the rescoring methods. The BRKPOS and BRKPHRASE methods, with the additional training data, also achieve significant improvements over the REGULAR baseline without rescoring.

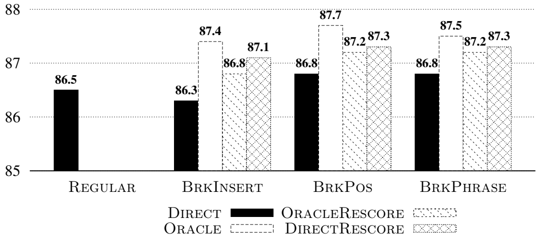


Figure 3: Parsing results on the Fisher evaluation set of the PCFG-LA models trained on the Fisher+Switchboard training data.

8 Error Analysis

In this section, we compare the errors of the BRKPHRASE PCFG-LA model and the DIRECTRESCORE approach for that model to each other and to the baseline PCFG-LA model without prosodic breaks. All models are trained and tested on Fisher as in Section 6. The results using other prosodically enhanced PCFG-LA models and their rescoring alternatives show similar patterns.

Figure 4 depicts the difference in F scores between BRKPHRASE and REGULAR and between BRKPHRASE+DIRECTRESCORE and REGULAR on a tree-by-tree basis in a 2D plot. Each quadrant also contains +/– signs roughly describing how much BRKPHRASE+DIRECTRESCORE is better (+) or worse (–) than BRKPHRASE and a pair of numbers (a, b), in which a represents the percentage of sentences in that quadrant containing *p* or *4* breaks that do not align with true phrase boundaries, and b represents the percentage of such *p* and *4* breaks among the total number of *p* and *4* breaks in that quadrant.

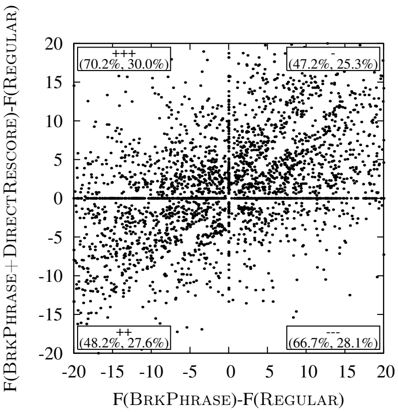


Figure 4: 2D plot of the difference in F scores between BRKPHRASE and REGULAR and between BRKPHRASE+DIRECTRESCORE and REGULAR, on a tree-by-tree basis, where each dot represents a test sentence. Each quadrant also contains +/– signs roughly describing how much BRKPHRASE+DIRECTRESCORE is better (+) or worse (–) than BRKPHRASE and a pair of numbers (a, b), in which a represents the percentage of sentences in that quadrant containing *p* or *4* breaks that do not align with true phrase boundaries, and b represents the percentage of such *p* and *4* breaks among the total number of *p* and *4* breaks in that quadrant.

Each dot in the top-right quadrant represents a test sentence for which both BRKPHRASE and BRKPHRASE+DIRECTRESCORE produce better trees than the baseline REGULAR PCFG-LA model. The BRKPHRASE+DIRECTRESCORE approach is on average slightly worse than the BRKPHRASE method (hence the single minus sign), although it also often produces better parses than BRKPHRASE alone. In contrast, the BRKPHRASE+DIRECTRESCORE approach on average makes many fewer errors than BRKPHRASE (hence + +) as can be observed in the bottom-left quadrant, where both approaches produce worse parse trees than the REGULAR baseline. The most interesting quadrant is on the top-left where the BRKPHRASE approach always produces worse parses than the REGULAR baseline while the BRKPHRASE+DIRECTRESCORE approach is able to avoid these errors while producing better parses than the baseline (hence + + +). Although the BRKPHRASE+DIRECTRESCORE approach can also produce worse parses than REGULAR, as in the bottom-right quadrant (hence – – –), altogether the quadrants suggest that, by restricting the search space to the n-best lists produced by the baseline REGULAR parser, the BRKPHRASE+DIRECTRESCORE approach is able to avoid many bad parses trees at the expense of somewhat poorer parses in cases when BRKPHRASE is able to benefit from the full search space.

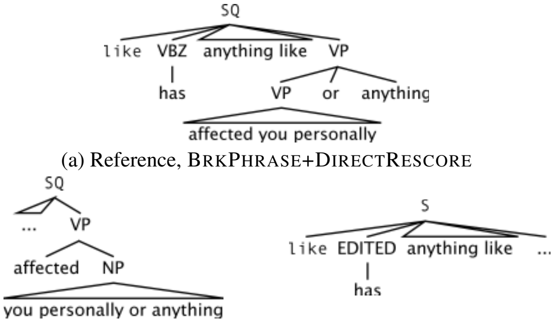
The reader should note that the top-left quadrant of Figure 4 has the highest percentage (70.2%) of sentences with "bad" *p* and *4* breaks and the highest percentage (30.0%) of such "bad" breaks among all breaks. This evidence supports our argument that "bad" breaks are harmful to parsing performance and some parse errors caused by misleading breaks can be resolved by limiting the search space of the prosodically enriched models to the n-best lists produced by the baseline REGULAR parser. However, the significant presence of "bad" breaks in the top-right quadrant also suggests that the prosodically enriched models are able to produce better parses than the baseline despite the presence of "bad" breaks, probably because the models are trained on the mixture of both "good" and "bad" breaks and are able to somehow learn to use "good" breaks while avoiding being misled by "bad" breaks.

	REGULAR	BRKPHRASE	BRKPHRASE +DIRECTRESCORE
NP	90.4	90.4	**90.9**
VP	84.7	84.7	**85.6**
S	84.4	84.3	**85.2**
INTJ	93.0	**93.4**	**93.4**
PP	76.5	76.7	**77.9**
EDITED	60.4	62.2	**63.3**
SBAR	67.2	67.0	**68.8**

Table 2: F scores of the seven most frequent non-terminals of the REGULAR, BRKPHRASE, and BRKPHRASE+DIRECTRESCORE models.

Table 2 reports the F scores of the seven most frequent phrases for the REGULAR, BRKPHRASE, and BRKPHRASE+DIRECTRESCORE methods trained on Fisher. When comparing the BRKPHRASE method to REGULAR, the break indexes help to improve the score for edits most, followed by interjections and prepositional phrases; however, they do not improve the accuracy of any of the other phrases. The BRKPHRASE+DIRECTRESCORE approach obtains improvements on all of the major phrases.

Figure 5 (a) shows a reference parse tree of a test sentence. The REGULAR approach correctly parses the first half (omitted) of the sentence but it fails to correctly interpret the second half (as shown). The BRKPHRASE approach, in contrast, is misguided by the incorrectly classified interruption point *p* after word "has", and so produces an incorrect parse early in the sentence. The BRKPHRASE+DIRECTRESCORE approach is able to provide the correct tree given the n-best list produced by the REGULAR approach, despite the break index errors.



(a) Reference, BRKPHRASE+DIRECTRESCORE

(b) REGULAR

(c) BRKPHRASE

Figure 5: Parses for *like*$_{*1*}$ *has*$_{*p*}$ *anything*$_{*1*}$ *like*$_{*1*}$ *affected*$_{*1*}$ *you*$_{*4*}$ *personally*$_{*4*}$ *or*$_{*1*}$ *anything*$_{*4*}$

9 Conclusions

We have investigated using prosodic information in the form of automatically detected ToBI break indexes for parsing spontaneous speech by comparing three prosodic enrichment methods. Although prosodic enrichment improves the basic PCFGs, that performance gain disappears when latent variables are used, partly due to the impact of misclassified ("bad") breaks that are assigned to words that do not occur at phrase boundaries. However, we find that by simply restricting the search space of the three prosodically enriched latent variable parser models to the n-best parses from the baseline PCFG-LA parser, all of them attain significant improvements. Our analysis more fully explains the positive results achieved by (Kahn et al., 2005) from reranking with prosodic features and suggests that the hypothesis that inserted prosodic punctuation breaks n-gram dependencies only partially explains the negative results of (Gregory et al., 2004). Our findings from the oracle experiment suggest that integrating ToBI classification with syntactic parsing should increase the accuracy of both tasks.

Acknowledgments

We would like to thank Izhak Shafran for providing break indexes for Fisher and Switchboard and for comments on an earlier draft of this paper. This research was supported in part by NSF IIS-0703859. Opinions, findings, and recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the funding agency or the institutions where the work was completed.

References

Ann Bies, Stephanie Strassel, Haejoong Lee, Kazuaki Maeda, Seth Kulick, Yang Liu, Mary Harper, and Matthew Lease. 2006. Linguistic resources for speech parsing. In *LREC*.

Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *ACL*.

Anne Cutler, Delphine Dahan, and Wilma v an Donselaar. 1997. Prosody in comprehension of spoken language: A literature review. *Language and Speech*.

Markus Dreyer and Izhak Shafran. 2007. Exploiting prosody for PCFGs with latent annotations. In *Interspeech*.

John J. Godfrey, Edward C. Holliman, and Jane McDaniel. 1992. SWITCHBOARD: Telephone speech corpus for research and development. In *ICASSP*.

Michelle L. Gregory, Mark Johnson, and Eugene Charniak. 2004. Sentence-internal prosody does not help parsing the way punctuation does. In *NAACL*.

Mary P. Harper, Bonnie J. Dorr, John Hale, Brian Roark, Izhak Shafran, Matthew Lease, Yang Liu, Matthew Snover, Lisa Yung, Anna Krasnyanskaya, and Robin Stewart. 2005. 2005 Johns Hopkins Summer Workshop Final Report on Parsing and Spoken Structural Event Detection. Technical report, Johns Hopkins University.

Zhongqiang Huang and Mary Harper. 2009. Self-training PCFG grammars with latent annotations across languages. In *EMNLP*.

Jeremy G. Kahn, Matthew Lease, Eugene Charniak, Mark Johnson, and Mari Ostendorf. 2005. Effective use of prosody in parsing conversational speech. In *EMNLP-HLT*.

Yang Liu, Andreas Stolcke, Elizabeth Shriberg, and Mary Harper. 2005. Using conditional random fields for sentence boundary detection in speech. In *ACL*.

Takuya Matsuzaki, Yusuke Miyao, and Jun'ichi Tsujii. 2005. Probabilistic CFG with latent annotations. In *ACL*.

Slav Petrov and Dan Klein. 2007. Improved inference for unlexicalized parsing. In *HLT-NAACL*.

Brian Roark, Mary Harper, Yang Liu, Robin Stewart, Matthew Lease, Matthew Snover, Izhak Shafran, Bonnie J. Dorr, John Hale, Anna Krasnyanskaya, and Lisa Yung. 2006. Sparseval: Evaluation metrics for parsing speech. In *LREC*.

Kim Silverman, Mary Beckman, John Pitrelli, Mari Ostendorf, Colin Wightman, Patti Price, Janet Pierrehumbert, and Julia Hirshberg. 1992. ToBI: A standard for labeling English prosody. In *ICSLP*.