Using Integer Linear Programming for Detecting Speech Disfluencies

Kallirroi Georgila
Institute for Creative Technologies, University of Southern California
13274 Fiji Way, Marina del Rey, CA 90292, USA
kgeorgila@ict.usc.edu

Abstract

We present a novel two-stage technique for detecting speech disfluencies based on Integer Linear Programming (ILP). In the first stage we use state-of-the-art models for speech disfluency detection, in particular, hidden-event language models, maximum entropy models and conditional random fields. During testing each model proposes possible disfluency labels which are then assessed in the presence of local and global constraints using ILP. Our experimental results show that by using ILP we can improve the performance of our models with negligible cost in processing time. The less training data is available the larger the improvement due to ILP.

1 Introduction

Speech disfluencies (also known as speech repairs) occur frequently in spontaneous speech and can pose difficulties to natural language processing (NLP) since most NLP tools (e.g. parsers, part-of-speech taggers, information extraction modules) are traditionally trained on written language. Speech disfluencies can be divided into three intervals, the *reparandum*, the *editing term* and the *correction* (Heeman and Allen, 1999; Liu et al., 2006).

```
(it was) * (you know) it was set
```

In the above example, "it was" is the reparandum, "you know" is the editing term and the remaining sentence is the correction. The asterisk marks the interruption point at which the speaker halts the original utterance in order to start the repair. The editing term is optional and consists of one or more filled pauses (e.g. uh, uh-huh) or discourse markers (e.g. you know, so). Some researchers include editing terms in the definition of disfluencies. Here we focus only on detecting repetitions (the speaker repeats some part of the utterance), revisions (the speaker modifies the original utterance) or restarts (the speaker abandons an utterance and starts over). We also deal with complex disfluencies, i.e. a series of disfluencies in succession ("I think I think uh I believe that...").

In previous work many different approaches to detecting speech disfluencies have been proposed. Different types of features have been used, e.g. lexical features only, acoustic and prosodic features only or a combination of both (Liu et al., 2006). Furthermore, a number of studies have been conducted on human transcriptions while other efforts have focused on detecting disfluencies from the speech recognition output.

In this paper we propose a novel framework for speech disfluency detection based on Integer Linear Programming (ILP). With Linear Programming (LP) problems the goal is to optimize a linear objective function subject to linear equality and linear inequality constraints. When some or all the variables of the objective function and the constraints are non-negative integers, LP becomes ILP. ILP has recently attracted much attention in NLP. It has been applied to several problems including sentence compression (Clarke and Lapata, 2008) and relation extraction (Roth and Yih, 2004). Some of these methods (e.g. (Roth and Yih, 2004)) follow the two-stage approach of first hypothesizing a list of possible answers using a classifier and then selecting the best answer by applying ILP. We have adopted this two-stage approach and applied it to speech disfluency detection.

In the first stage we use state-of-the-art tech-

Proceedings of NAACL HLT 2009: Short Papers, pages 109–112,
Boulder, Colorado, June 2009.

niques for speech disfluency detection, in particular, Hidden-Event Language Models (HELMs) (Stolcke and Shriberg, 1996), Maximum Entropy (ME) models (Ratnaparkhi, 1998) and Conditional Random Fields (CRFs) (Lafferty et al., 2001). Nevertheless, any other classification method could be used instead. During testing each classifier proposes possible labels which are then assessed in the presence of local and global constraints using ILP. ILP makes the final decision taking into account both the constraints and the output of the classifier.

In the following we use the Switchboard corpus and only lexical features for training our 3 classifiers. Then we apply ILP to the output of each classifier. Our goal is not to investigate the best set of features or achieve the best possible results. In that case we could also use prosodic features as they have been shown to improve performance. Our target is to show that by using ILP we can improve with negligible cost in processing time the performance of state-of-the-art techniques, especially when not much training data is available.

The novelty of our work lies in the two following areas: First, we propose a novel approach for detecting disfluencies with improvements over state-of-the-art models (HELMs, ME models and CRFs) that use similar lexical features. Although the two-stage approach is not unique, as discussed above, the formulation of the ILP objective function and constraints for disfluency detection is entirely novel. Second, we compare our models using the tasks of both detecting the interruption point and finding the beginning of the reparandum. In previous work (Liu et al., 2006) Hidden Markov Models (combination of decision trees and HELMs) and ME models were trained to detect the interruption points and then heuristic rules were applied to find the correct onset of the reparandum in contrast to CRFs that were trained to detect both points at the same time.

The structure of the paper is as follows: In section 2 we describe our data set. In section 3 we describe our approach in detail. Then in section 4 we present our experiments and provide results. Finally in section 5 we present our conclusion and propose future work.

2 Data Set

We use Switchboard (LDC catalog LDC99T42), which is traditionally used for speech disfluency experiments. We transformed the Switchboard annotations into the following format:

```
I BE was IE one IP I was right
```

BE (beginning of edit) is the point where the reparandum starts and IP is the interruption point (the point before the repair starts). In the above example the beginning of the reparandum is the first occurrence of "I", the interruption point appears after "one" and every word between BE and IP is tagged as IE (inside edit). Sometimes BE and IP occur at the same point, e.g. "I BE-IP I think".

The number of occurrences of BE and IP in our training set are 34387 and 39031 respectively, in our development set 3146 and 3499, and in our test set 6394 and 7413.

3 Methodology

In the first stage we train our classifier. Any classifier can be used as long as it provides more than one possible answer (i.e. tag) for each word in the utterance. Valid tags are BE, BE-IP, IP, IE or O. The O tag indicates that the word is outside the disfluent part of the utterance. ILP will be applied to the output of the classifier during testing.

Let N be the number of words of each utterance and i the location of the word in the utterance (i=1,...,N). Also, let $C_{BE}(i)$ be a binary variable (1 or 0) for the BE tag. Its value will be determined by ILP. If it is 1 then the word will be tagged as BE. In the same way, we use $C_{BE-IP}(i)$, $C_{IP}(i)$, $C_{IE}(i)$, $C_O(i)$ for tags BE-IP, IP, IE and O respectively. Let $P_{BE}(i)$ be the probability given by the classifier that the word is tagged as BE. In the same way, let $P_{BE-IP}(i)$, $P_{IP}(i)$, $P_{IE}(i)$, $P_O(i)$ be the probabilities for tags BE-IP, IP, IE and O respectively. Given the above definitions, the ILP problem formulation can be as follows:

$$max[\sum_{i=1}^{N}[P_{BE}(i)C_{BE}(i) + P_{BE-IP}(i)C_{BE-IP}(i) + P_{IP}(i)C_{IP}(i) + P_{IE}(i)C_{IE}(i) + P_O(i)C_O(i)]] \quad (1)$$

subject to:

$$C_{BE}(i) + C_{BE-IP}(i) + C_{IP}(i) + C_{IE}(i) + C_O(i) = 1 \qquad \forall i \in (1, ..., N) \quad (2)$$

$$C_{BE}(1) + C_{BE-IP}(1) + C_O(1) = 1 \quad (3)$$

$$C_{BE-IP}(N) + C_{IP}(N) + C_O(N) = 1 \quad (4)$$

$$C_{BE}(i) - C_{BE-IP}(i-1) - C_{IP}(i-1) - C_O(i-1) \leq 0 \qquad \forall i \in (2, ..., N) \quad (5)$$

$$1 - C_{BE}(i) - C_{BE}(i-1) \geq 0 \quad \forall i \in (2, ..., N) \quad (6)$$

Equation 1 is the linear objective function that we want to maximize, i.e. the overall probability of the utterance. Equation 2 says that each word can have one tag only. Equation 3 denotes that the first word is either BE, BE-IP or O. Equation 4 says that the last word is either BE-IP, IP or O. For example the last word cannot be BE because then we would expect to see an IP. Equation 5 defines the transitions that are allowed between tags as described in Table 1 (first row). Equation 5 says that if we have a word tagged as BE it means that the previous word was tagged as BE-IP or IP or O. It could not have been tagged as IE because IE must be followed by an IP before a new disfluency starts. Also, it could not have been BE because then we would expect to see an IP. From Table 1 we can easily define 4 more equations for the rest of the tags. Finally, equation 6 denotes that we cannot transition from BE to BE (we need an IP in between).

We also formulate some additional rules that describe common disfluency patterns. First, let us have an example of a long-context rule. If we have the sequence of words "he was the one um you know she was the one", we expect this to be tagged as "he BE was IE the IE one IP um O you O know O she O was O the O one O", if we do not take into account the context in which this pattern occurs. We incorporate this rule into our ILP problem formulation as follows: Let (w_1,...,w_N) be a sequence of N words where both w_2 and w_{N-7} are personal pronouns, the word sequence w_3,w_4,w_5 is the same as the sequence w_{N-6},w_{N-5},w_{N-4} and all the words in between (w_6,...,w_{N-8}) are filled pauses or discourse markers. Then the probabilities given by the classifier are modified as follows: $P_{BE}(2)$=$P_{BE}(2)$+$th1$, $P_{IE}(3)$=$P_{IE}(3)$+$th2$, $P_{IE}(4)$=$P_{IE}(4)$+$th3$ and $P_{IP}(5)$=$P_{IP}(5)$+$th4$, where $th1$, $th2$, $th3$ and $th4$ are empirically set thresholds (between 0.5 and 1, using the development set of the corpus).

Now, here is an example of a short-context rule. If we have the same word appear 3 times in a row ("do do do") we expect this to be tagged as "do BE-IP do IP do O". To incorporate this rule into our ILP problem formulation we can modify the probabilities given by the classifier accordingly.

In total we have used 7 rules that deal with short-context and 5 rules that deal with long-context dependencies. From now on we will refer to the model that uses all rules (general ILP formulation and all pattern-based rules) as ILP and to the model that uses only the general ILP constraints and the short-context pattern-based rules as ILP-. In all rules, we can skip editing terms (see example above).

From Tag	To Tag
BE-IP or IP or O	BE
BE-IP or IP or O	BE-IP
BE or BE-IP or IP or IE	IP
BE or BE-IP or IP or IE	IE
BE-IP or IP or O	O

Table 1: Possible transitions between tags.

4 Experiments

For HELMs we use the SRI Statistical Language Modeling Toolkit. Each utterance is a sequence of word and Part-of-Speech (POS) pairs fed into the toolkit: `i/prp BE was/vbd IE one/cd IP i/prp was/vbd right/jj`. We report results with 4-grams. For ME we use the OpenNLP MaxEnt toolkit and for CRFs the toolkit CRF++ (both available from `sourceforge`). We experimented with different sets of features and we achieved the best results with the following setup (i is the location of the word or POS in the sentence): Our word features are $\langle w_i \rangle$, $\langle w_{i+1} \rangle$, $\langle w_{i-1}, w_i \rangle$, $\langle w_i, w_{i+1} \rangle$, $\langle w_{i-2}, w_{i-1}, w_i \rangle$, $\langle w_i, w_{i+1}, w_{i+2} \rangle$. Our POS features have the same structure as the word features. For ILP we use the `lp_solve` software also available from `sourceforge`.

For evaluating the performance of our models we use standard metrics proposed in the literature, i.e. F-score and NIST Error Rate. We report results for BE and IP. F-score is the harmonic mean of precision and recall (we equally weight precision and recall). Precision is computed as the ratio of the correctly identified tags X to all the tags X detected by the model (where X is BE or IP). Recall is the ratio of the correctly identified tags X to all the tags X that appear in the reference utterance. The NIST Error Rate measures the average number of incorrectly identified tags per reference tag, i.e. the sum of insertions, deletions and substitutions divided by the total number of reference tags (Liu et al., 2006). To calculate the level of statistical significance we always use the Wilcoxon signed-rank test.

Table 2 presents comparative results between our models. The ILP and ILP- models lead to significant improvements compared to the plain models for HELMs and ME ($p<10^{-8}$, plain models vs. ILP and ILP-). With CRFs the improvement is smaller,

	BE		IP	
	F-score	**Error**	**F-score**	**Error**
4gram	60.3	54.8	67.0	50.7
4gram ILP	**76.0**	**38.1**	**79.0**	**38.0**
4gram ILP-	73.9	39.5	77.9	38.3
ME	63.8	52.6	72.8	44.3
ME ILP	**77.9**	**36.3**	80.8	35.4
ME ILP-	75.6	37.2	**81.0**	**33.7**
CRF	78.6	34.3	82.0	31.7
CRF ILP	**80.1**	34.5	82.5	33.3
CRF ILP-	79.8	**33.5**	**83.4**	**30.5**

Table 2: Comparative results between our models.

	25%	**50%**	**75%**	**100%**
4gram	59.8	56.6	56.2	54.8
4gram ILP	40.2	38.9	38.2	38.0
4gram ILP-	42.1	40.7	39.8	39.5
ME	61.6	56.9	54.7	52.6
ME ILP	38.5	37.7	36.5	36.3
ME ILP-	39.7	38.7	37.6	37.2
CRF	40.3	37.1	35.5	34.3
CRF ILP	37.1	36.2	35.2	34.5
CRF ILP-	36.6	35.5	34.4	33.5

Table 3: Error rate variation for BE depending on the training set size.

$p<0.03$ (CRF vs. CRF with ILP), not significant (CRF vs. CRF with ILP-), $p<0.0008$ (CRF with ILP vs. CRF with ILP-). HELMs and ME models benefit more from the ILP model than the ILP- model (ME only for the BE tag) whereas ILP- appears to perform better than ILP for CRFs.

Table 3 shows the effect of the training set size on the error rates only for BE due to space restrictions. The trend is similar for IP. The test set is always the same. Both ILP and ILP- perform better than the plain models. This is true even when the ILP and ILP- models are trained with less data (HELMs and ME models only). Note that HELM (or ME) with ILP or ILP- trained on 25% of the data performs better than plain HELM (or ME) trained on 100% of the data ($p<10^{-8}$). This is very important because collecting and annotating data is expensive and time-consuming. Furthermore, for CRFs in particular the training process takes long especially for large data sets. In our experiments CRFs took about 400 iterations to converge (approx. 136 min for the whole training set) whereas ME models took approx. 48 min for the same number of iterations and training set size. Also, ME models trained with 100 iterations (approx. 11 min) performed better than ME models trained with 400 iterations. The cost of applying ILP is negligible since the process is fast and applied during testing.

5 Conclusion

We presented a novel two-stage technique for detecting speech disfluencies based on ILP. In the first stage we trained HELMs, ME models and CRFs. During testing each classifier proposed possible labels which were then assessed in the presence of local and global constraints using ILP. We showed that ILP can improve the performance of state-of-the-art classifiers with negligible cost in processing time, especially when not much training data is available. The improvement is significant for HELMs and ME models. In future work we will experiment with acoustic and prosodic features and detect disfluencies from the speech recognition output.

Acknowledgments

This work was sponsored by the U.S. Army Research, Development, and Engineering Command (RDECOM). The content does not necessarily reflect the position or the policy of the Government, and no official endorsement should be inferred.

References

J. Clarke and M. Lapata. 2008. Global inference for sentence compression: An integer linear programming approach. *Journal of Artificial Intelligence Research*, 31:399–429.

P. Heeman and J. Allen. 1999. Speech repairs, intonational phrases and discourse markers: Modeling speakers' utterances in spoken dialogue. *Computational Linguistics*, 25:527–571.

J. Lafferty, A. McCallum, and F. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. of ICML*.

Y. Liu, E. Shriberg, A. Stolcke, D. Hillard, M. Ostendorf, and M. Harper. 2006. Enriching speech recognition with automatic detection of sentence boundaries and disfluencies. *IEEE Trans. Audio, Speech and Language Processing*, 14(5):1526–1540.

A. Ratnaparkhi. 1998. *Maximum Entropy Models for natural language ambiguity resolution*. Ph.D. thesis, University of Pennsylvania.

D. Roth and W. Yih. 2004. A linear programming formulation for global inference in natural language tasks. In *Proc. of CoNNL*.

A. Stolcke and E. Shriberg. 1996. Statistical language modeling for speech disfluencies. In *Proc. of ICASSP*.