AUTOMATIC DISFLUENCY REMOVAL FOR IMPROVING SPOKEN LANGUAGE TRANSLATION

Wen Wang, Gokhan Tur, Jing Zheng, Necip Fazil Ayan

Speech Technology and Research Laboratory, SRI International, Menlo Park, CA, USA
{wwang,gokhan,zj,nfa}@speech.sri.com

ABSTRACT

Statistical machine translation (SMT) systems for spoken languages suffer from conversational speech phenomena, in particular, the presence of speech disfluencies. We examine the impact of disfluencies from broadcast conversation data on our hierarchical phrase-based SMT system and implement automatic disfluency removal approaches for cleansing the MT input. We evaluate the efficacy of proposed approaches and investigate the impact of disfluency removal on SMT performance across different disfluency types. We show that for translating Mandarin broadcast conversational transcripts into English, our automatic disfluency removal approaches could produce significant improvement in BLEU and TER.

Index Terms— statistical machine translation, spoken language translation, automatic disfluency detection, broadcast conversation

1. INTRODUCTION

In recent years, spoken language translation has been gaining more attention in the statistical machine translation (SMT) community. Under the DARPA GALE program,[1] the audio document translation task includes the domains of broadcast news (BN) and broadcast conversations (BC) for the Chinese-to-English and Arabic-to-English language pairs. The BN genre consists of "talking head" style broadcasts, i.e., generally one person reading a news script. The BC genre, by contrast, is more interactive and spontaneous, referring to free speech in news-style TV and radio programs and consisting of talk shows, interviews, call-in programs, live reports, and round-tables. Speech disfluencies are common phenomena in spontaneous speech and they are those phrases/words in speech that when removed will yield the originally intended, more fluent utterances. Presence of disfluencies in the BC data can hurt SMT as disfluencies do not add semantic information, while making sentences longer and interrupting the flow of speech. In past research, removing disfluencies is able to make sentences less ill-formed and hence help downstream NLP tasks such as machine translation (MT). Rao et al. investigated the effect of disfluencies on SMT and introduced an automatic disfluency removal scheme based on the noisy-channel model [1]. In their work, disfluency removal is viewed as a translation task where the source language is the disfluent speech and the target language is the fluent speech. They showed that automatic disfluency removal prior to Arabic-to-English BC translation can produce up to 8% relative improvement in BLEU. Our disfluency detection systems are different from their work as we develop a combination of statistical models using Hidden Markov Model (HMM) and Conditional Random Field (CRF) and heuristic rule-based systems. Our translation task is Chinese-to-English BC translation.

[1]The goal of the GALE program is to develop computer software techniques to analyze, interpret, and distill information from speech and text in multiple languages.

2. IMPACT OF DISFLUENCIES ON SRI SMT

The SRI SMT system used in this work is a hierarchical phrase-based decoder (HPBT). For the hierarchical phrase-based translation model [2], a CKY-style search algorithm with LM intersection is implemented in our in-house SRInterp$^{\mathrm{TM}}$ decoder for this purpose, following Chiang's approach. The feature weights are optimized using minimum error rate training (MERT) to maximize BLEU [3] using the Amoeba simplex search on N-best lists of the development test set. For the GALE Year 3 MT evaluation, we trained the HPBT system on a parallel text corpus containing 2.4 million sentences, with about 59 million running words in the English side. The vast majority of the training data came from the text domain, especially newswire. GIZA++ was used to generate word alignments with grow-diag-final-and symmetrization heuristics [4]. A 4-gram word n-gram LM trained from about 5 billion words and estimated with modified Kneser-Ney smoothing was used in decoding.

In this work, we use two GALE test sets for the experiment: Dev08 and Test08. Characteristics of the two test sets are shown in Table 1. We use Dev08 to optimize the log-linear model scaling factors, and evaluate on Test08. We use case-insensitive IBM BLEU [3] with four references as our target metric for minimum error rate training and to report results. Since we want to measure the effect of syntax without factoring in automatic speech recognition (ASR) errors, we test the systems on the human transcripts of the test sets. For our speech translation evaluation system, we assigned a weight of 5.0 to the limited amount of BN and BC data made available by LDC to the GALE community and obtained 30.9% BLEU on the Test08 BN subset and a lower BLEU as 30.0% on the Test08 BC subset [5]. When closely examining the BC decoding output, we noticed that the translation performance on more conversational style sentences is worse than that on more structured sentences. The translation rules, trained mainly on more structured newswire text data, are not well adapted to the conversational style BC data that bear numerous disfluencies.

Table 1. Tuning and testing data size summary.

	#Sentences		#English Tokens	
	BN	**BC**	**BN**	**BC**
Dev08(tune)	529	1134	13k	17k
Test08(test)	483	937	12k	14k

3. SPEECH DISFLUENCIES

After examining the GALE Chinese-to-English BC data about the speech disfluency types defined in [6], we focus on the following two types of disfluencies.

Fillers: fillers include filled pauses (FP), discourse markers (DM), and explicit editing terms (ET). FPs are words used by the speakers as floor holders to maintain control of a conversation. They can also indicate hesitations of the speaker, for example, *uh*, *um*. DMs are words/phrases that are related to the structure of the discourse and help beginning or keeping a turn or serving as acknowledgment, for example, *I mean*, *you know*. An explicit ET is an editing term in an edit disfluency that is not an FP or a DM. For example, *we have two action items* $*$ *sorry three action items from the meeting*, where *sorry* is an explicit ET.

Edit disfluencies: Edit disfluencies involve syntactically relevant content that is either repeated, revised, or abandoned. The basic pattern for edit disfluencies appears like **(reparandum)** $*$ **<editing term> correction**. The reparandum is the portion of the utterance that is corrected or abandoned entirely (in the case of restarts). An interruption point (IP), marked with '$*$' in the pattern, is the point at which the speaker breaks off the original utterance and then repeats, revises, or restarts the utterance. The editing term is optional and consists of one or more filler words. The correction is the portion of the utterance that corrects the original reparandum. Markups of the reparandum and editing terms facilitate modeling edit disfluencies and cleaning them up. Note that repetitions impose a special case of edit disfluencies and are easier to model compared to the rest of the subtypes, such as revisions (content replacements) and restarts (fresh starts). Revisions denote the cases when a speaker modifies the original utterance with a similar syntactic structure, e.g., *we have two action items* $*$ *sorry three action items from the meeting*. Restarts denote the cases when a speaker abandons an utterance or a constituent and restarts all over again, e.g., *He* $*$ *I like this idea*.

4. AUTOMATIC DISFLUENCY REMOVAL

We followed the DARPA EARS Metadata Extraction (MDE) [7] task definitions and developed automatic disfluency removal by implementing models for detecting different structural events, namely, filler word detection (detecting words used as fillers), edit word detection (detecting words within the boundary of reparandum of an edit disfluency), and interruption point detection. Since we observed that in the MT parallel text training corpus, translations of disfluent source sentences also bear a certain degree of disfluencies (and less disfluent in many cases compared to the source side), the ultimate goal of our work is to detect disfluencies in both source and target side sentences for cleaning up the SMT training data and also apply disfluency detection on the source side of test sets to clean up the test input. This paper focused on an intermediate milestone, where the training data for SMT is the structured newswire text, and disfluency detection is only applied to the test set source side before decoding to examine its effect on the SMT performance. We measured the accuracy of detected events, but we focused on measuring the final MT performance by BLEU and TER before and after disfluency removal.

Disfluency removal requires explicitly detecting the boundary of the reparandum and filler words so that they can be removed. The disfluency removal system in this work was implemented as three systems. The combination of the first system (System I) and the second system (System II) was used for detecting the boundary of the reparandum and System III was used for detecting filler words. System I combines a word-based hidden-event LM and a POS-based hidden-event LM for IP detection and then applies knowledge-based rules to identify the onset of the disfluency. This cascade system can hence detect the boundary of the reparandum. System II uses a CRF model to jointly model IP detection and reparandum boundary detection. The union of the System I and System II is the final system output for reparandum detection. System III is a rule-based system for filler word detection. Details of the systems are described in the following subsections.

4.1. System I: Hidden-event LMs And Knowledge-based Rules

The IP detection task is a two-class classification problem, detecting "IP" or "non-IP" for each between-word location. We trained a hidden-event word-based LM and a hidden-event POS-based LM for detecting IPs, as described in [6]. The hidden-event word-based LM models the joint distribution of the word sequence W and its event sequence E, $P(W, E)$, where the event sequence is composed of a sequence of possible non-word tokens appearing between words, e.g., whether an IP appears between two certain words. The hidden-event word-based LM is directly trained from BC transcripts with the manually labeled IPs, using the event labels and word surface form.

In the hidden-event POS-based LM, the word surface forms are replaced by their part-of-speech (POS) tags so that we can model syntactically generalizable patterns. For example, pronouns and certain adverbs are frequently repeated in the GALE Chinese-to-English BC data. To create a high-quality Chinese POS tagger for the BC genre, we applied semi-supervised training approaches [8] to bootstrap Chinese conversational speech POS taggers from Chinese newswire treebank in the LDC Chinese treebank 6.0 and GALE Ontonotes broadcast news and conversation treebanks and a large amount of unlabeled GALE Mandarin BC manual transcripts. The POS tagging accuracy measured on the unseen heldout set of the Ontonotes BC treebank is 92.7%. We then used the POS tagger to tag our IP detection training and test data. Similar to [6], we maintained the identity of some cue words (i.e., filler words and discourse markers). Then we modeled the joint distribution of the POS sequence P and its event sequence E. In this work, we trained 5-gram word-based hidden-event LM and 5-gram POS-based hidden-event LM. During testing for both hidden-event word-based and POS-based LMs, we used a forward-backward approach to maximize the posterior probability $P(E|W)$ or $P(E|P)$. To combine the word-based and POS-based hidden-event LM for IP detection, we interpolated the posterior probabilities from both models.

After IPs are detected, we used knowledge-based rules to locate the onset of the reparandum. This rule-based component uses the IP hypotheses as input and employs heuristic rules to locate the onset of the reparandum in a disfluency, i.e., where the edit disfluency starts. For revisions, a repeated word across an IP is a helpful cue for determining the reparandum, with additional cues from repeated POS sequences. For example, if we correctly detect the IP between "items" and "sorry" in *we have two action items sorry three action items from the meeting*, the combined cues from the repeated phrase "action items" and the same POS tag *CD* for "two" and "three" will help delimiting the reparandum "two action items".

4.2. System II: CRF for Joint Detection

We used the CRF modeling approach to combine lexical features and shallow syntactic features for joint detection of edit words and IP words. System II is different from System I in that it directly detects the entire region of the reparandum, instead of from a postprocessing rule-based system using IP hypotheses. A CRF is an undirected graphical model that defines a global log-linear distribution of the state (or label) sequence E conditioned on an observation sequence, in our case including the word sequence W and the POS tag sequence P. The conditional probability is computed as follows:

$$P(E|W,P) = \frac{1}{Z_\lambda(W,P)} \exp(\sum_k \lambda_k G_k(E,W,P))$$

where the functions G are potential functions over the events and the observations, the index k represents different features, λ_k's are the feature weights, and Z_λ is the normalization term. Each of the potential functions can be either a state feature function of the event at position i and the observation sequence $s(E_i, W, P, i)$ or a transition feature function of the observation sequence and the labels at position $i-1$ and i, $t(E_{i-1}, E_i, W, P, i, i-1)$. These features are indicator functions that capture the co-occurrence of events and some observations, which is similar to maximum entropy (ME) modeling. The major difference between ME and CRF is that the feature functions G_k are over the sequence E in a CRF versus individual events E_i in the ME model. Hence, the CRF model is optimized globally over the entire sequence, whereas the ME model makes a decision at each point individually without considering the context event information. In this work, we used the Mallet package [9] to implement the CRF model and used a first-order model that includes only two sequential events in the feature set. The CRF model is trained to maximize the conditional log-likelihood of a given training set $P(E|W,P)$. During testing, the most likely sequence E is found using the Viterbi algorithm. To avoid over-fitting, a Gaussian prior [10] with a zero mean, which was optimized on the development test set, was employed to the parameters. Following [6], we included IP into the target classes and used 5 states, as *outside edit* (O), *begin edit with an IP* (B-E+IP), *begin edit* (B-E), *inside edit with an IP* (I-E+IP), and *inside edit* (I-E). Compared to the hidden-event LMs in System I, the CRF model does not make independence assumptions on features and can hence explore various correlated features.

4.3. System III: Filler Word Detection

System III is a rule-based system for filler word detection. By studying the GALE Ontonotes Chinese BC treebank and incorporating linguistic knowledge for Chinese, we defined a list of possible Chinese filler words, including filled pauses and discourse markers. Also, POS tags assigned by our Chinese BC POS tagger could provide important cues, as most of filler words have POS tags as *IJ* (interjection) and *SP* (possible sentence-final particle).

5. EXPERIMENTAL DATA AND RESULTS

We first evaluated the effect on SMT by removing manually marked up disfluencies. The data with manual disfluency annotation was extracted from the GALE Ontonotes Chinese BC parallel treebank. The GALE Ontonotes Chinese BC parallel treebank is composed of treebanks for both source and target side text for 3,230 sentences of GALE Y1 Mandarin BC manual transcriptions from 10 shows. Note that all disfluency studies in this work are based on audio manual transcripts so that we can measure the effect of disfluency removal on the SMT performance without factoring in ASR errors. Among all 3,230 sentences, 203 sentences were manually annotated with edit disfluencies and fillers. The manual markups only discriminate spans between edit disfluencies (with DFL as the non-terminal) and fillers (with FLR as the non-terminal). We created four test sets out of the 203 sentences based on different disfluency removal strategies, namely, the original set of sentences (denoted *withED-withFLR*), the alternatives with just removing edit disfluencies (denoted *woED-withFLR*), just removing fillers (denoted *withED-woFLR*), and removing both edit disfluencies and fillers (denoted *woED-woFLR*. Similarly, we also used the manual disfluency markups on the target side to create the four versions of translation references. We ran 1-best decoding on the four test sets using our HPBT system trained on only newswire and webtext portion of the GALE Y3 evaluation system training data. In order to obtain fair scoring for the four different test sets, we combined all four versions of the translation references into multiple references for computing case-insensitive IBM BLEU [3] with closest reference length for each translation hypothesis. We also computed TER [11]. The results are shown in Table 2. For the HPBT system, the best BLEU score was obtained from removing edit disfluencies but keeping fillers, followed by removing all disfluencies. Removing all disfluencies produced the best TER score. This is consistent with our intuition that edit disfluencies are detrimental to MT performance; whereas, fillers might serve similar functions as punctuation so that they might constrain reordering and hence could help improving the MT performance. The scores on the withED-woFLR test set are worse than those from the baseline withED-withFLR condition, which is also consistent with this hypothesis.

Table 2. Comparing results from decoding the four different versions of GALE ontonotes test sets. Results are in 4-reference case-insensitive BLEU (%) and TER (%).

Testset	BLEU	TER
Baseline: withED-withFLR	19.1	74.2
withED-woFLR	18.5	72.7
woED-withFLR	**20.2**	70.8
woED-woFLR	19.6	**69.5**

Next, we evaluated our automatic Mandarin disfluency detection models described in Section 4. Since there is little publicly available Mandarin structured MDE annotated corpus, we randomly split the 203 sentences with edit disfluencies and fillers into 102 sentences (3K words) as the seed corpus (denoted *DF-seed*) for bootstrapping our statistical disfluency detection models and the rest 101 sentences (3K words) for testing the model performance (denoted *DF-test*). There are 146 edit disfluencies and 150 fillers in the seed corpus, and 150 edit disfluencies and 150 fillers in the test set. We reformatted the seed corpus and the test set into the EARS LDC MDE annotation style, i.e., marking up filler words, edit words, and IP, and manually proof-read the two corpora. The seed corpus *DF-seed* is too small for training high-quality statistical models for edit word and IP detection. So we bootstrapped the statistical models from the seed corpus and the large amount of unlabeled GALE BC manual transcripts in a self-training strategy. For each iteration of self-training, we held out a small subset of the unlabeled BC manual transcripts. We then employed System I and System II for edit word detection, filtered out hypotheses with posteriors lower than a certain threshold, and combined the two systems as a union to generate the final annotation output for the small subset. This final annotation output was added to the training pool of System I and System II. The procedure was repeated until all BC manual transcripts were explored for enhancing the statistical models. For evaluation, the hypothesized structural events are mapped to the reference events in the *DF-test* test set. The error rate is the average number of misclassified words per reference event word, computed using the NIST tools [6]. Table 3 shows the results in NIST error rate (%) for edit word, IP, and filler word detection. In our study, System I, as a hybrid sequential model of HMM and heuristic rules, performs better than the CRF based System II, for both edit word and IP detection. This observation is different from the observation on the English CTS data reported in [6], where the CRF is better at finding edit words but poorer at IP detection, compared to the HMM model. This is probably due to our very limited manual disfluency annotated training data and the relatively noisy self-training procedure. Due to incorporation of heuristic rules, the hybrid model is more robust than the CRF model. As a preliminary effort on Mandarin BC disfluency detection and considering lack of the training data, these results are promising as they are only 3% to 6% higher than the error rate from the state-of-the-art English MDE detection models [6].

Table 3. NIST error rate (%) for edit word, IP, and filler word detection on the *DF-test* test set.

System	**NIST Error Rate (%)**		
	Edit Word	Edit IP	Filler Word
System I	58.5	39.2	–
System II	61.2	42.5	–
System I + System II	56.9	37.6	–
System III	–	–	30.9

Finally, we applied the three automatic disfluency detection systems on the two GALE test sets Dev08 and Test08 BC manual transcripts, for detecting edit words and filler words and removing them from the SMT input. The HPBT system was trained only on GALE newswire and web text parallel data. To focus on investigating the effect of disfluency removal on SMT, we manually pulled out all sentences with disfluencies in these two BC test sets and denoted them as **Dev08 BC-DF** and **Test08 BC-DF**, while the entire Dev08 and Test08 BC test sets are denoted **Dev08 BC** and **Test08 BC**. Dev08 BC-DF includes 339 sentences out of the 1,134 sentences Dev08 BC test set and Test08 BC-DF includes 242 sentences out of the 937 sentences Test08 BC test set. Similar to the studies we conducted on the Ontonotes data, we also compared decoding the original BC-DF input, as well as removing automatically detected edit words, filler words, and both. For scoring, we manually cleaned up disfluencies in the 4-reference translation references corresponding to each of the four SMT input conditions, and combined them for 16-reference scoring. The results are shown in the top half of Table 4. It is interesting to observe the same patterns of the effect from removing edit words and filler words on the SMT performance, between manual disfluency annotation and automatic disfluency detection. The best SMT performance was obtained by removing edit disfluencies but keeping the fillers, followed by removing both of them, and followed by the baseline of the original data. The worst SMT performance came from keeping edit disfluencies but removing fillers. These consistent observations suggest that edit disfluencies are detrimental to our HPBT system performance, whereas fillers might serve as barriers to constrain reordering and thus showed to be helpful. Overall, automatic disfluency removal could achieve 0.8% and 0.7% absolute improvement on BLEU, and 2.8% and 2% absolute improvement on TER, on the two BC-DF test sets. On the entire Dev08 BC and Test08 BC test sets, these improvements on the BC-DF subsets resulted in 0.4% and 0.3% absolute improvement on BLEU, and 1.7% and 1.5% absolute improvement on TER, respectively.

In summary, we have reviewed how disfluencies impacted SRI GALE Y3 SMT systems on BC data and investigated the impact of BLEU and TER from removing manually annotated disfluencies. We found the best SMT performance was obtained by removing edit disfluencies but keeping fillers. We developed three systems for automatic detection of edit words and filler words in Mandarin BC

Table 4. Comparing results from decoding Dev08 and Test08 BC-DF test sets under the four different conditions, as well as comparison on the entire Dev08 BC and Test08 BC test sets. Edit words and filler words were automatically detected. Results are in 16-reference case-insensitive BLEU (%) and TER (%).

Testset	**Dev08 BC-DF**		**Test08 BC-DF**	
	BLEU	TER	BLEU	TER
Baseline: withED-withFLR	28.6	63.6	26.5	64.4
withED-woFLR	27.8	67.2	26.0	68.4
woED-withFLR	**29.4**	**60.8**	**27.2**	**62.4**
woED-woFLR	29.1	62.2	26.9	63.8
Testset	**Dev08 BC**		**Test08 BC**	
	BLEU	TER	BLEU	TER
Baseline: withED-withFLR	27.6	63.9	26.2	64.4
woED-withFLR	28.0	62.2	26.5	62.9

data. We have examined the impact of BLEU and TER on GALE BC test sets after removing automatically detected disfluencies. We found that consistent with the observation on removing manually annotated disfluencies, removing automatically detected edit words while keeping the filler words improved performance for the lightly syntax-aware HPBT system.

Acknowledgments The authors thank Yang Liu, Andreas Stolcke, Dilek Hakkani-Tür, Elizabeth Shriberg, and Kristin Precoda for valuable discussions and suggestions. This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR0011-06-C-0023 (approved for public release, distribution unlimited). 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.

6. REFERENCES

[1] S. Rao, I. Lane, and T. Schultz, "Improving spoken language translation by automatic disfluency removal:Evidence from conversational speech transcripts", *in Proceedings of Machine Translation Summit*, Copenhagen, 2007.

[2] J. Zheng, "SRInterp: SRI's Scalable Multipurpose SMT Engine", Technical report, SRI International, 2008.

[3] K. Papineni, S. Roukos, T. Ward, and W. J. Zhu, "BLEU: A method for automatic evaluation of machine translation", *in Proceedings of ACL*, Philadelphia, PA, 2002.

[4] P. Koehn, F. Och, and D. Marcu, "Statistical phrase-based translation", *in Proceedings of HLT-NAACL*, pp. 127–133, 2003.

[5] J. Zheng, N. F. Ayan, W. Wang, and D. Burkett, "Using syntax in large-scale audio document translation", *in Proceedings of Interspeech*, Brighton, UK, 2009.

[6] Y. Liu, E. Shriberg, A. Stolcke, D. Hillard, M. Ostendorf, and M. Harper, "Enriching speech recognition with automatic detection of sentence boundaries and disfluencies", *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, pp. 1526–1540, Sep. 2006, Special Issue on Progress in Rich Transcription.

[7] DARPA Information Processing Technology Office, "Effective,affordable, reusable speech-to-text (EARS)", 2003, http://www.darpa.mil/ipto/programs/ears.

[8] W. Wang, Z. Huang, and M. P. Harper, "Semi-supervised learning for part-of-speech tagging of Mandarin transcribed speech", *in Proceedings of ICASSP*, 2007.

[9] A. McCallum, "Mallet: A machine learning for language toolkit", 2002, http://mallet.cs.umass.edu.

[10] A. McCallum and W. Li, "Early results for named entity recognition with conditional random fields", *in Proceedings of the CoNLL*, 2003.

[11] M. Snover, B. Dorr, R. Schwartz, L. Micciulla, and J. Makhoul, "A study of translation edit rate with targeted human annotation", *in Proceedings of AMTA*, 2006.