INTERSPEECH 2019
September 15–19, 2019, Graz, Austria

Noisy BiLSTM-based Models for Disfluency Detection

Nguyen Bach, Fei Huang

Machine Intelligence Technology Lab, Alibaba Group
(nguyen.bach, f.huang)@alibaba-inc.com

Abstract

This paper describes BiLSTM-based models to disfluency detection in speech transcripts using residual BiLSTM blocks, self-attention, and noisy training approach. Our best model not only surpasses BERT in 4 non-Switchboard test sets, but also is 20 times smaller than the BERT-based model [1]. Thus, we demonstrate that strong performance can be achieved without extensively use of very large training data. In addition, we show that it is possible to be robust across data sets with noisy training approach in which we found insertion is the most useful noise for augmenting training data.

Index Terms: disfluency detection, LSTM, noise, robust, BERT, speech recognition.

1. Introduction

Disfluencies are interruptions in the regular flow of speech, such as using uh and um, pausing silently, repeating words, or interrupting oneself to correct something said previously. Disfluencies typically include filler pauses, explicit editing terms, discourse markers, coordinating conjunctions.

State-of-the-art speech translation systems are making progress to generate more usable translation outputs. Disfluencies are important to model in speech translation because they cause problems not only for translation but also for higher level natural language processing, such as information extraction, summarization and translation. Disfluencies also degrade transcript readabililty for humans and make the speech translation output less intelligible.

Disfluency detection is the task of distinguishing fluent from disfluent segments. We categorize disfluency detection methods into four main approaches: speech-based, parsing-based, tagging-based, and translation-based.

Speech-based approach Speech is passed through an energy based voice activity detector to identify silent and low energy regions which are useful to detect filled pauses [2]. Another method is to locate evidence of a general disfluency which are called interruption points (IPs). Generally, it looks in the nearby context of the IP to find the disfluent words. The most successful approaches so far combine the detection of IPs using prosodic features and language modeling techniques and speech signals [3, 4].

Parsing-based approach The tree adjoining grammar (TAG) noisy channel model was proposed to identify the possibility of a word being disfluent in [5]. Following along the line of TAG models is the extension of using a language model and MaxEnt reranker [6, 7]. [8, 9] proposed syntax-based models such as transition-based dependency parsing to jointly perform dependency parsing and disfluency detection.

Tagging-based approach The disfluency detection problem is treated as tagging problem in which words are tagged by disfluency types or simply fluent/disfluent tags. This line of work includes sequence modeling techniques such as hidden Markov models (HMM), conditional random field (CRF) [10], semi-Markov models [11], bi-directional long short-term memory (BiLSTM), auto-correlational neural network [12, 13], and convolutional neural network (CNN) [14].

Translation-based approach This approach considers disfluent text as the source language and the clean text as target language. Phrase-based translation model [15] and seq2seq-based model [16] were proposed to demonstrate this idea. [17] archived the state-of-the-art performance on the Switchboard data set with a Transformer-based model [18].

In this paper, we explore different BiLSTM-based architectures to detect disfluency words. We try to answer the following questions:

- Can we push the current state-of-the-art performance on commonly used speech corpus, Switchboard, further?
- How embeddings and self-attention will affect prediction performances?
- Can we replicate the success of residual architecture in this task?
- How robust are the networks across different test sets?

In Section 2, we describe the network architectures and training method. We describe the experiment setting in Section 3. Experimental results and analysis are reported in Section 4.

2. Methods

Throughout this paper we consider the disfluency detection problem as a sequence labeling problem which falls into the tagging-based approach. The input is sequence of transcription words $x = (x_1, x_2, ..., x_T)$ and the output is a sequence of labels $y = (y_1, y_2, ..., y_T)$ where x_i is in the ASR vocabulary and y_i is in $\{@dis, O\}$ tag set[1].

There are several ways to tackle the sequence label problem and we choose to focus on the state-of-the-art architectures described in [19] and [20]. Figure 1 shows the core building blocks of these models which include

- Sequence representation at character and word level given the input x
- Bidirectional LSTM captures both past and future semantic information in sequence from left and right context. Eq. 1 shows formulas to update LSTM unit at time t [21]

$$\begin{aligned} i_t &= \sigma(W_i h_{t-1} + U_i x_t + b_i) \\ f_t &= \sigma(W_f h_{t-1} + U_f x_t + b_f) \\ \tilde{C}_t &= tanh(W_c h_{t-1} + U_c x_t + b_C) \\ C_t &= f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \\ o_t &= \sigma(W_o h_{t-1} + U_o x_t + b_o) \\ h_t &= o_t \odot tanh(c_t) \end{aligned} \quad (1)$$

[1] $@dis$ and O are referred to as disfluency and ordinary.

http://dx.doi.org/10.21437/Interspeech.2019-1336

where σ and $\odot$ are element-wise sigmoid function and product, respectively. x_t is the input vector, e.g embeddings, at time t. h_t is the output vector, e.g hidden state, at time t. Us are weight matrices for input x_t, and Ws are weight matrices for hidden state h_t. bs denotes bias vectors.

- CRF allows us to model the label sequence y jointly as showed in Eq. 2 in which a softmax over all possible tag sequences yields a probability for the sequence y

$$p(y|x) = \frac{e^{s(x,y)}}{\sum_{\tilde{y} \in Y_x} e^{s(x,\tilde{y})}}, \quad s(x,y) = \sum_{j=1}^{m} \sum_{t=1}^{T} \lambda_j f_j(x, t, y_t, y_{t-1}) \tag{2}$$

where f_j and λ_j are the j^{th} feature function and weight.

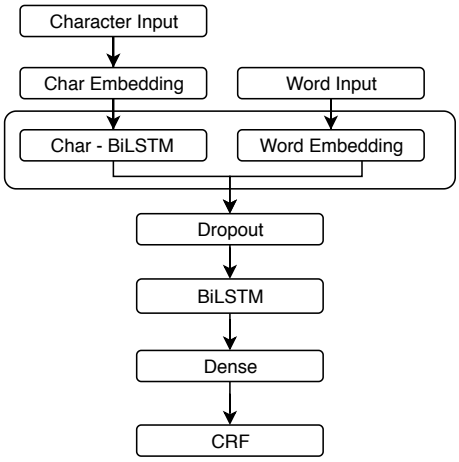

Figure 1: *A BiLSTM CRF (BC) architecture for sequence labeling [19] and [20]*

2.1. Word representation

A modification for the BiLSTM CRF architecture is to have a richer word embedding layer such as GLoVe [22] and ELMo [23] embeddings. [23] shows the state-of-the-art performance in the CoNLL 2003 NER task when using the ELMo embedding. In the disfluency detection context we create two variants

- GloVe BiLSTM CRF (GBC): replace the word embedding layer with GloVe
- ELMo GloVe BiLSTM CRF (EGBC): combine ELMo and GloVe word embeddings.

We decided not using the BERT embeddings since it obtains lower performance than the fine-tune approach as shown in [1].

2.2. Self-Attention

Self-attention, also known as intra-attention, is an attention mechanism relating different positions of a single sequence in order to compute a representation of the same sequence. It has been shown to be very useful in machine translation [18, 24], machine reading [25], or image description generation [26]. Similarly, self-attention mechanism has been showing beneficial for the sequence labeling task as in [27] and [28]. We follow [28] and use multiplicative attention in our implementations.

2.3. Residual BiLSTM block

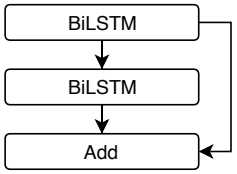

Figure 2: *A BiLSTM residual block*

A traditional way of adding more representational power to a neural network is layer stacking which has been successfully used in a lot of works [29, 30]. The core idea is introducing an identity shortcut connection that skips one or more BiLSTM layers. [29] shows that stacking layers do not degrade the network performance, because we could simply stack identity mappings[2] upon the current network, and the resulting architecture would perform the same. Figure 2 presents a ResNet-style BiLSTM residual block in which we skip one BiLSTM layer.

2.4. Noisy training

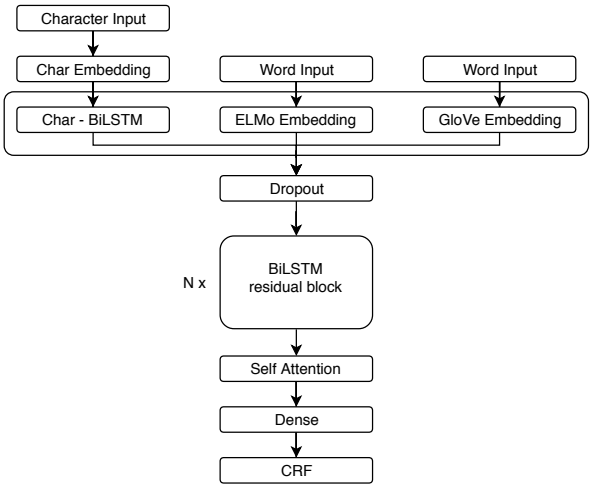

Figure 3: *ELMo Glove residual BiLSTM Attention CRF architectures in which the residual block is repeated N times*

Figure 3 illustrates the model architecture which includes richer word representation, residual BiLSTM, and self-attention blocks. One of the key challenges in disfluency detection is dealing with noise. Particularly, the noise comes from

- ASR errors: insertion, repetition, and deletion of words
- multi-turn dialog: interruptions, topic interweave, and chitchat content
- non-native speaker: grammatical errors

In this paper we decide to focus on dealing with ASR error types. Following the idea in [31] we use the corrupted inputs to train a robust disfluency detection model. Essentially, in the beginning of each training epoch, we randomly select a n percent of the training data, clean it and augment the clean transcription with a noise type.

[2] An identity mapping is layer that does not do anything

- **Insertion (iNoise)**: we first pick how many words to insert with a probability constraint. The next step is to randomly select insert position and words. All insertion words will be labeled as disfluency.
- **Deletion (dNoise)**: we randomly delete a word in a given segment and after deletion all remaining words are labeled as ordinary, non-disfluent words.
- **Repetition (rNoise)**: we randomly pick a position to start a repetition, and randomly pick repetition length in between 1 and 4 words. All repeated words are labeled as disfluency.

3. Experiment settings

Benchmark data The English Switchboard Corpus[3] is probably the most widely used data set for benchmarking the English disfluency performance. The Switchboard corpus has three disfluency annotation levels which are

- DFF: disfluency annotation only
- MGD: disfluency annotation and part-of-speech tags
- DPS: disfluency annotation, part-of-speech tags and turns joined

We use the DPS annotation. We follow the same train and test splitting by [5] in order to keep a direct comparison with previous works. Specifically, we use sw4[0-1]* files as the test set and the rest is for training and development sets. All data is preprocessed with lower case, no punctuation, and treebank tokenizer.
We apply a similar labeling scheme as in [17] with 2 labels O and $@dis$. An example of the Switchboard data is

> and/*O* i/*O* do/*O* n't/*O* have/*O* to/*@dis* to/*O* be/*O* writing/*O* checks/*O*

Other data sets To further evaluate our methods, we use additional corpora

- CallHome: telephone conversations between family members and close friends
- SCOTUS: oral arguments between justices and advocates
- FCIC: two hearings from Financial Crisis Inquiry Commission
- Interview: two TV interviews of a non-native English speaker

CallHome, SCOTUS, FCIC corpora are obtained via [32]. The Interview is our in-house corpus which followed the Switchboard's disfluency annotation guidelines.
Metrics We use token-based precision (P), recall (R), and F-score (F1) as the evaluation metrics which is also used in other works [11, 12, 33, 17].
Settings Our implementations are based on Tensorflow, Keras, and Anago[4]. We use Nvidia P100 GPUs to conduct training and decoding experiments. All hyper parameters are tuned in the development set. The BERT [1] experiment is based on the PyTorch pretrained BERT[5] with uncased base model. The BERT fine-tuned model is trained with 4 epochs as suggested by the BERT team with max sequence length 75 and batch size 32. For our models, we fix the batch size and training epochs as 128 and 10 respectively. BiLSTM output space for word and character are 100 and 25 respectively. The residual BiLSTM contains 6 residual blocks. We fix the percentage amount data which will be used during noisy training to 1 percent.

[3]https://catalog.ldc.upenn.edu/LDC99T42
[4]https://github.com/Hironsan/anago
[5]https://github.com/huggingface/pytorch-pretrained-BERT

4. Results

Table 1: *Results of disfluency detection on the English Switchboard data set. The first section shows pervious work. The second section describes our baseline with the richer word embedding method. The third section presents the contribution of residual, self-attention, and noisy training over our baseline. The forth section shows the BERT fine-tuned model performance.*

Method	P	R	F1
Weight sharing [17]	92.1	90.2	**91.1**
Transition-based [33]	91.1	84.1	87.5
BiLSTM [12]	91.6	80.3	85.9
Semi-CRF [11]	90.0	81.2	85.4
EGBC	95.9	86.3	90.9
GBC	93.1	80.9	86.6
BiLSTM CRF (BC) [20]	91.6	79.6	85.2
EGBC + residual + iNoise	95.7	88.3	**91.8**
EGBC + residual + self-attention	94.5	88.6	91.5
EGBC + residual	96.1	86.9	91.2
BERT fine-tune [1]	94.7	89.8	**92.2**

4.1. Switchboard results

Table 1 shows our main result on the Switchboard data set. We first observe that using richer word representations improves disfluency labelling significantly. For example the BiLSTM CRF (BC) in [19] gains 1.4 F1 score when replacing its word embedding with GloVe embedding as showed in GBC model. The gain is even bigger with 5.7 F1 improvement when both GloVe (G) and ELMo (E) embeddings are used as showed in EGBC model. The EGBC model performs very close the state-of-the-art weight-sharing model in [17].

We found that the EGBC model can be further improved with residual BiLSTM, self-attention, and noisy training. Residual BiLSTM and insertion noise help the EGBC model outperform the state-of-the-art system [17] by 0.7 F1 score. We further use the pretrained uncased base BERT model for fine-tuning on the Switchboard train set and archive 92.2 F1 score. This is our best model on the Switchboard corpus.

4.2. Non-Switchboard results

The previous section shows experimental results when train and test are both on the Switchboard. The BERT fine-tuned model is the best model on Switchboard, however, our proposed model is only behind by 0.4 F1 score. An interesting question is how would these models trained on Switchboard perform differently on other data sets.

Table 2 shows the model performances on CallHome, FCIC, SCOTUS, and Interview data sets. We also compute the average F1 score across test set to get a performance overview.

Table 2: *Results on 4 non-Switchboard test sets. The last column shows an average F1 score across 4 test sets. The first is our best model on Switchboard, and the second is the baseline model. The third section describes different noisy training scheme over the baseline including insertion, deletion, and repetition noise. The last section presents the model with residual BiLSTM block, self-attention, and insertion noise.*

Method	**CallHome**			**FCIC**			**SCOTUS**			**Interview**			**Average F1**
	P	R	F1	P	R	F1	P	R	F1	P	R	F1	
BERT fine-tune [1]	23.8	58.0	33.7	45.5	57.2	50.7	66.4	71.1	68.7	47.1	48	47.6	**50.18**
EGBC	24.8	55.9	34.3	46.3	54.8	50.2	67.1	71.4	69.2	53.5	43.5	48	50.43
EGBC + iNoise	23.9	60.8	34.3	48	57	52.3	69.4	70.6	**70**	47.9	44.9	46.4	**50.75**
EGBC + dNoise	22.6	59.5	32.8	43.1	58.6	49.7	63.4	70.7	66.8	45	43.9	44.4	48.43
EGBC + rNoise	21.4	66.5	32.4	40.2	63	49.1	62	72.6	66.9	41.7	47.5	44.4	48.2
EGBC + residual + self-attention + iNoise	24.7	60.1	**35.0**	50.7	53.8	**52.2**	71.7	67.7	69.6	57.5	42.3	**48.7**	**51.38**

We found that by adding insertion noise during training helps the EGBC to be more robust. We see that the EGBC with residual blocks, self-attention and insertion noise outperforms the BERT fine-tuned model by 1.2 F1 score on average across 4 test sets. Our models surpassed BERT in all 4 non-Switchboard test sets.

Furthermore, it is noted that the BERT model is a huge model in comparison with our models. The number of parameters in BERT is nearly 20 times larger than our biggest model as showed in Table 3. As a result, our models require less resource for model development and product deployment than the BERT model

Table 3: *The number of model parameters*

Method	Parameters
BERT fine-tune (base uncased)	110 million
EGBC	2.9 million
EGBC + residual + self-attention + iNoise	5.6 million

Table 4 presents the disfluency detection output of the EGBC with residual blocks, self-attention and insertion noise.

Table 4: *Examples of disfluency detection. The red cross-out is our model disfluency detection. The blue underline is the reference annotation.*

Switchboard	but they 've been in office since the ~~the~~ nineteen forties
CallHome	and then at night claudia gives ~~the~~ me a
FCIC	~~i~~ i not ~~referring to a specific~~ i referring to the fact that we look at our risks and we look at our positions
SCOTUS	~~what in~~ ~~in~~ what respect do you claim he is not properly deportable
Interview	life is a full of failures if people say ~~you know~~ nobody helped me when we do business you always want people to help you but if there 's no people to help you

5. Conclusions

In this paper we proposed methods to detect disfluency based on residual BiLSTM blocks, self-attention, and noisy training. Our major contributions are model architectures, noisy training approach for disfluency detection task. We shows that on the Switchboard data set the new state-of-the-art results is 92.2 F1 score with the BERT fine-tuned model. Experimental results show that by combining residual BiLSTM blocks, self-attention, and insertion noise on top of a strong baseline, our model outperforms BERT by 1.2 F1 score on average across 4 non-Switchboard test sets. We show that our models are not only nearly 20 times smaller than BERT-based model but also surpasses BERT in 4 non-Switchboard test sets.

This work can be expanded in several directions. First, we plan to dive deeper into the noisy training approach, and particularly with speech recognition transcription. We envision there are still some room for improvement in the noisy training approach. Moreover, we intend to explore how disfluency detection can be used for speech translation application the way that benefit users most.

6. Acknowledgements

We would like to thank the Alibaba machine translation team for their supports. Also, we would like to thank Luo Wei, Li Bo, and Boxing Chen for their helpful comments.

7. References

[1] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," *CoRR*, vol. abs/1810.04805, 2018. [Online]. Available: http://arxiv.org/abs/1810.04805

[2] E. E. Shriberg, "Phonetic consequences of speech disfluency," in *ICPhS 1999. Proceedings of the 14th International Congress of Phonetic Sciences*, J. J. Ohala, Y. Hasegawa, M. Ohala, D. Granville, and A. C. Bailey, Eds., 1999, pp. 619–622.

[3] A. Stolcke, E. Shriberg, R. A. Bates, M. Ostendorf, D. Hakkani, M. Plauché, G. Tür, and Y. Lu, "Automatic detection of sentence boundaries and disfluencies based on recognized words," 1998.

[4] Y. Liu, E. Shriberg, and A. Stolcke, "Automatic disfluency identification in conversational speech using multiple knowledge sources," in *INTERSPEECH*. ISCA, 2003.

[5] M. Johnson and E. Charniak, "A tag-based noisy-channel model of speech repairs," in *Proceedings of the 42nd Annual Meeting of*

the Association for Computational Linguistics (ACL-04), 2004. [Online]. Available: http://www.aclweb.org/anthology/P04-1005

[6] M. Johnson, E. Charniak, and M. Lease, "An improved model for recognizing disfluencies in conversational speech," in *In Proceedings of Rich Transcription Workshop*, 2004.

[7] S. Zwarts and M. Johnson, "The impact of language models and loss functions on repair disfluency detection," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 2011, pp. 703–711. [Online]. Available: http://aclweb.org/anthology/P11-1071

[8] M. Honnibal and M. Johnson, "Joint incremental disfluency detection and dependency parsing," *Transactions of the Association for Computational Linguistics*, vol. 2, pp. 131–142, 2014. [Online]. Available: http://aclweb.org/anthology/Q14-1011

[9] S. Wu, D. Zhang, M. Zhou, and T. Zhao, "Efficient disfluency detection with transition-based parsing," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Association for Computational Linguistics, 2015, pp. 495–503. [Online]. Available: http://aclweb.org/anthology/P15-1048

[10] Y. Liu, E. Shriberg, A. Stolcke, and M. P. Harper, "Comparing hmm, maximum entropy, and conditional random fields for disfluency detection," in *INTERSPEECH*. ISCA, 2005, pp. 3313–3316.

[11] J. Ferguson, G. Durrett, and D. Klein, "Disfluency detection with a semi-markov model and prosodic features," in *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 2015, pp. 257–262. [Online]. Available: http://aclweb.org/anthology/N15-1029

[12] V. Zayats, M. Ostendorf, and H. Hajishirzi, "Disfluency detection using a bidirectional LSTM." INTERSPEECH16.

[13] J. Hough and D. Schlangen, "Joint, incremental disfluency detection and utterance segmentation from speech," in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*. Association for Computational Linguistics, 2017, pp. 326–336. [Online]. Available: http://aclweb.org/anthology/E17-1031

[14] P. Jamshid Lou, P. Anderson, and M. Johnson, "Disfluency detection using auto-correlational neural networks," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2018, pp. 4610–4619. [Online]. Available: http://aclweb.org/anthology/D18-1490

[15] S. Maskey, B. Zhou, and Y. Gao, "A phrase-level machine translation approach for disfluency detection using weighted finite state transducers," in *INTERSPEECH*. ISCA, 2006.

[16] E. Cho, J. Niehues, T.-L. Ha, and A. Waibel, "Multilingual disfluency removal using nmt," in *Proceedings of the 13th International Workshop on Spoken Language Translation (IWSLT)*, Seattle, USA, December 8-9 2016.

[17] F. Wang, W. Chen, Z. Yang, Q. Dong, S. Xu, and B. Xu, "Semi-supervised disfluency detection," in *Proceedings of the 27th International Conference on Computational Linguistics*. Association for Computational Linguistics, 2018, pp. 3529–3538. [Online]. Available: http://aclweb.org/anthology/C18-1299

[18] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *NIPS*, 2017, pp. 6000–6010.

[19] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer, "Neural architectures for named entity recognition," in *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, 2016, pp. 260–270. [Online]. Available: http://aclweb.org/anthology/N16-1030

[20] X. Ma and E. Hovy, "End-to-end sequence labeling via bi-directional lstm-cnns-crf," in *Proceedings of the 54th ACL*. Association for Computational Linguistics, 2016, pp. 1064–1074. [Online]. Available: http://aclweb.org/anthology/P16-1101

[21] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997. [Online]. Available: http://dx.doi.org/10.1162/neco.1997.9.8.1735

[22] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 2014, pp. 1532–1543. [Online]. Available: http://aclweb.org/anthology/D14-1162

[23] M. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. Association for Computational Linguistics, 2018, pp. 2227–2237. [Online]. Available: http://aclweb.org/anthology/N18-1202

[24] G. Tang, M. Müller, A. Rios, and R. Sennrich, "Why self-attention? a targeted evaluation of neural machine translation architectures," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2018, pp. 4263–4272. [Online]. Available: http://aclweb.org/anthology/D18-1458

[25] J. Cheng, L. Dong, and M. Lapata, "Long short-term memory-networks for machine reading," in *Proceedings of the 2016 Conference on EMNLP*. Austin, Texas: Association for Computational Linguistics, Nov. 2016, pp. 551–561. [Online]. Available: http://www.aclweb.org/anthology/D16-1053

[26] K. Xu, J. L. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhutdinov, R. S. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," in *Proceedings of the 32Nd ICML - Volume 37*, ser. ICML'15. JMLR.org, 2015, pp. 2048–2057. [Online]. Available: http://dl.acm.org/citation.cfm?id=3045118.3045336

[27] M. Rei, G. Crichton, and S. Pyysalo, "Attending to characters in neural sequence labeling models," in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. Osaka, Japan: The COLING 2016 Organizing Committee, Dec. 2016, pp. 309–318. [Online]. Available: http://www.aclweb.org/anthology/C16-1030

[28] G. Zheng, S. Mukherjee, X. L. Dong, and F. Li, "Opentag: Open attribute value extraction from product profiles," in *Proceedings of the 24th ACM SIGKDD*, ser. KDD '18. New York, NY, USA: ACM, 2018, pp. 1049–1058. [Online]. Available: http://doi.acm.org/10.1145/3219819.3219839

[29] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

[30] Q. Tran, A. MacKinlay, and A. Jimeno Yepes, "Named entity recognition with stack residual lstm and trainable bias decoding," in *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Asian Federation of Natural Language Processing, 2017, pp. 566–575. [Online]. Available: http://aclweb.org/anthology/I17-1057

[31] N. Bach, H. Chen, K. Fan, C.-C. Leung, B. Li, C. Ni, R. Tong, P. Zhang, B. Chen, B. Ma, and F. Huang, "Alibaba speech translation systems for IWSLT 2018," in *Proc. of the International Workshop on Spoken Language Translation*, Bruges, Belgium, 2018.

[32] V. Zayats, M. Ostendorf, and H. Hajishirzi, "Multi-domain disfluency and repair detection." INTERSPEECH14.

[33] S. Wang, W. Che, Y. Zhang, M. Zhang, and T. Liu, "Transition-based disfluency detection using lstms," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2017, pp. 2785–2794. [Online]. Available: http://aclweb.org/anthology/D17-1296