Investigating LSTM for Punctuation Prediction

Kaituo Xu[1]*, Lei Xie*[1]*, Kaisheng Yao*[2]

[1]School of Computer Science, Northwestern Polytechnical University, Xi'an
[2]Microsoft Corporation, Redmond, 98052, WA

{ktxu,lxie}@nwpu-aslp.org, kaisheny@microsoft.com

Abstract

We present a neural network based punctuation prediction method using Long Short-Term Memory (LSTM) network. The proposed method uses bidirectional LSTM to encode both the past and future observation as its inputs. It models the dependency between input features and output labels through multiple layers. We also empirically study the impacts of modeling the dependency between output labels. Our results show that using a deep bi-directional LSTM is able to achieve state-of-the-art performance in punctuation prediction.

Index Terms: punctuation prediction, long short term memory, recurrent neural network, conditional random field

1. Introduction

A standard automatic speech recognizer (ASR) typically generates only a stream of words, without any punctuation symbols. The missing punctuations make the transcripts difficult to read and create barriers to many downstream language processing tasks, such as sentiment analysis, information extraction and machine translation (MT) [1, 2]. For example, various models in current MT systems are trained on punctuated texts, and thus for an acceptable level of quality in translation, those systems expect correctly punctuated texts. Moreover, the output of MT is expected to have correct punctuation in the output language as well. Automatic punctuation prediction improves the readability as well as facilitates subsequent tasks by inserting suitable punctuation marks in the text.

A substantial amount of work has been done in punctuation prediction and a related task named sentence boundary detection [3, 4], which only predicts sentence ends. Some previous research explores both lexical and prosodic features [3, 4, 5, 6, 7, 8, 9]. However, acoustic signal may not be readily available in real applications. In this paper, we address the punctuation prediction problem solely from lexical input, i.e., restoring major punctuation marks from an input stream of words.

Recently, neural networks have re-emerged as an powerful tool in many tasks. Specifically, with powerful sequential learning ability, recurrent neural network (RNN) and its variants have shown their superior performances in a variety of sequential labelling tasks, e.g., part-of-speech (POS) tagging, chunking and named entity recognition (NER) [10], prosodic boundary prediction [11] and language understanding [12]. Punctuation prediction can be regarded as a typical sequential labelling task, i.e., label each inter-word position using an appropriate punctuation mark (or non-punctuation). Recently, long short-term memory (LSTM) was suggested for punctuation prediction in [13]. LSTM-RNN uses specifically designed gates to control information flow and thus has exceptional long context modeling ability. In [13], LSTM was used to model only past contextual inputs. But we believe that punctuation labeling might be more accurate if both the past and future contexts are both considered. What follows affects the current decision on the punctuation. Meanwhile, Tilk *et al.* only use one LSTM layer in punctuation prediction [13]. Recent studies suggest that using multiple hidden layers can learn hierarchical features and boost the performance [14]. On the other hand, studies in various sequential labeling tasks [10, 11, 12] have also indicated that using a conditional random fields (CRF) layer on top of LSTM can catch the output context information, leading to further performance gain.

This paper performs an extensive study on the use of LSTM for punctuation prediction. Our contributions can be summarized as follows. 1) We propose to use bidirectional LSTM (BLSTM) and deep network architecture to consider the both past and future inputs as well as to model the complex relationships between input feature and output labels. 2) We investigate whether modeling the context of output punctuation labels, through a CRF layer, can achieve performance gain for punctuation prediction, as expected in other sequential labelling tasks. 3) Our study concludes that a 2-layer BLSTM model can produce state-of-the-art performance in punctuation prediction, while modeling output contexts does not lead to improved performances.

2. Related Work

Punctuation prediction and its related tasks have been studied in the speech and language processing field for many years. According to the features used, we can roughly divide previous studies into two types: one only uses lexical features (words and N-grams, etc.), such as [15, 16, 17]; the other integrates both lexical and prosodic features (pause duration and pitch, etc.), such as [3, 4, 5, 6, 7, 8, 9]. Based on the above features, a variety of models can be used, including language model (LM), maximum entropy model (Maxent), statistical finite state, CRF, decision tree (DT) and neural networks (NN).

As an early approach, Beeferman *et al.* [15] studied a trigram LM for punctuation annotation for speech transcripts. As another popular approach, hidden event LM treats sentence boundary and punctuation as target events [5], where prosodic and LM cues are modeled by DT and N-gram, respectively, and subsequently integrated in a hidden event LM. Also, punctuation prediction is jointed addressed with other tasks, e.g., in [16], an N-gram model simultaneously predicts punctuation and case information for English.

Punctuation prediction can be treated as a sequence labelling task and tackled by CRF [3, 18]. Liu *et al.* [3] have shown that a linear chain CRF yields a lower error rate than HMM and Maxent on the NIST sentence boundary detection task. They owe the performance gain to CRF's abilities to directly estimate the posterior boundary label probabilities, to support

simultaneous correlated features and to model sequence information. Wang *et al.* [9] have adopted a dynamic CRF [17], which connects variables in different layers by introducing a pairwise factor, to jointly perform sentence boundary detection and punctuation prediction using lexical and prosodic features.

Neural networks offer a flexible architecture to construct complex models. Recently, a deep neural network (DNN) approach was proposed in [4] for sentence boundary detection. Recurrent networks, such as LSTM models, are able to model sequential information. Tilk *et al.* [13] have recently proposed a two-stage LSTM model to predict punctuations. An LSTM learns lexical features on a large text corpus in the first stage. The second stage combines pause durations with lexical features and adapts the model to the target domain.

3. Methods

3.1. LSTM

Allowing cyclical connections in a feed-forward neural network, we obtain recurrent neural networks (RNNs). RNNs have recently produced outstanding performances on many tasks including sequential labelling [19] and language modeling [20]. In theory, RNN can learn from the entire historical inputs. But in practice, it can access only a limited range of context because of the vanishing gradient problem. Long short-term memory (LSTM) [21] uses purpose-built memory cells to store information, which is designed to overcome this problem. LSTM is composed of a set of recurrently connected memory blocks and each block consists of one or more self-connected memory cells and three multiplicative gates, i.e., input gate, forget gate and output gate. The three gates are designed to capture long-range contextual information by using nonlinear summation units. Specifically, in this study, we use the LSTM with forget gates [22] and peephole connections [23] to predict punctuation, which is theoretically implemented as follows.

$$
\begin{aligned}
\boldsymbol{i}_t &= \sigma(\boldsymbol{W}_{xi}\boldsymbol{x}_t + \boldsymbol{W}_{hi}\boldsymbol{h}_{t-1} + \boldsymbol{W}_{ci}\boldsymbol{c}_{t-1} + \boldsymbol{b}_i) \\
\boldsymbol{f}_t &= \sigma(\boldsymbol{W}_{xf}\boldsymbol{x}_t + \boldsymbol{W}_{hf}\boldsymbol{h}_{t-1} + \boldsymbol{W}_{cf}c_{t-1} + \boldsymbol{b}_f) \\
\boldsymbol{c}_t &= \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \tanh(\boldsymbol{W}_{xc}\boldsymbol{x}_t + \boldsymbol{W}_{hc}\boldsymbol{h}_{t-1} + \boldsymbol{b}_c) \\
\boldsymbol{o}_t &= \sigma(\boldsymbol{W}_{xo}\boldsymbol{x}_t + \boldsymbol{W}_{ho}\boldsymbol{h}_{t-1} + \boldsymbol{W}_{co}\boldsymbol{c}_t + \boldsymbol{b}_o) \\
\boldsymbol{h}_t &= \boldsymbol{o}_t \odot \tanh(\boldsymbol{c}_t)
\end{aligned}
\tag{1}
$$

where $\boldsymbol{x}_t$ is the input vector encoded in one-hot scheme; σ is the element-wise logistic sigmoid function; $\boldsymbol{i}$, $\boldsymbol{f}$, $\boldsymbol{o}$ and $\boldsymbol{c}$ denote the input gate, forget gate, output gate and memory cell, respectively, and all of them are the same size as the LSTM output vector $\boldsymbol{h}$; $\boldsymbol{W}_{xi}$ is the input-input gate matrix, $\boldsymbol{W}_{hc}$ is the hidden-cell matrix, and so on; $\odot$ is the element-wise product.

3.2. Deep bidirectional LSTM

Figure 1 shows the proposed bidirectional LSTM (BLSTM) architecture for punctuation prediction. BLSTM consists of a forward LSTM and a backward LSTM [24]. Given an input sequence $(\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n)$, the forward LSTM reads it from left to right, but the backward LSTM reads it in a reverse order. The two LSTMs have different parameters. Apparently, BLSTM can utilize both past inputs and future inputs for a specific time.

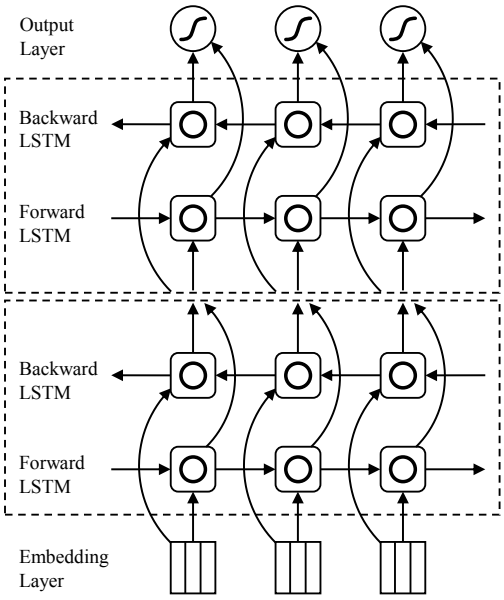

Figure 1: *Multi-layer bidirectional LSTM for punctuation prediction.*

3.3. LSTM-CRF

LSTM networks can use contextual inputs features, but for tasks that have strong dependencies across output labels, they lack capacity to model output label information. As we discussed in Section 2, CRF can use sentence-level label information and it is widely used in sequence labelling tasks [25]. Recent studies have shown that, a *hybrid* LSTM-CRF model yields exceptional performance in tasks like POS tagging, chunking and named entity recognition, where strong relationships exist between output labels [10, 26].

In an LSTM-CRF model, illustrated in Figure 2, a CRF layer is sitting on top of a LSTM layer and its parameters is a transition matrix. For a given input sequence $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n)$ and the corresponding prediction sequence $\boldsymbol{Y} = (y_1, y_2, \ldots, y_n)$, the score of the sequence is defined as

$$
s(\boldsymbol{X}, \boldsymbol{Y}) = \sum_{i=0}^{n} A_{y_i, y_{i+1}} + \sum_{i=1}^{n} P_{i, y_i}, \tag{2}
$$

where $\boldsymbol{P}$ is the matrix of score outputted by the LSTM layer and P_{i,y_i} denotes the score of the y_i-th label of the i-th word in a sequence; $\boldsymbol{A}$ is the transition matrix generated by the CRF layer and it is position independent, and $A_{i,j}$ denotes the score of a transition from the label i to label j; y_0 and y_n represent the start and end labels of a sequence, respectively.

Since LSTM-CRF only models bigram information between output labels, the dynamic programming algorithm can be effectively used to compute $\boldsymbol{A}$ and optimal label sequences for inference. Readers may refer to [25, 26] for more details.

As we mentioned above, BLSTM can adopt both past and future inputs, so we can replace the LSTM layer in LSTM-CRF model with a BLSTM layer to form a BLSTM-CRF model. Thus this model can use past inputs, future inputs and sentence level label information for punctuation prediction.

Table 1: *Punctuation prediction results of different LSTM models. C: comma; P: period.*

Model	*prec.*(C)	*rec.*(C)	F_1(C)	*prec.*(P)	*rec.*(P)	F_1(P)
CRF-unigram	72.69	62.12	66.99	77.62	63.22	69.68
LSTM	71.80	63.62	67.46	68.67	68.85	68.76
2layer-LSTM	71.71	68.89	70.27	76.22	65.70	70.57
BLSTM	74.70	74.81	74.76	80.97	69.75	74.94
2layer-BLSTM	77.99	72.30	**75.04**	76.77	75.71	**76.23**

Table 2: *Punctuation prediction results for models with and without output label context information. C: comma; P: period*

Model	*prec.*(C)	*rec.*(C)	F_1(C)	*prec.*(P)	*rec.*(P)	F_1(P)
LSTM	71.80	63.62	67.46	68.67	68.85	68.76
LSTM-CRF	61.98	51.72	56.39	69.08	58.15	63.15
BLSTM	74.70	74.81	74.76	80.97	69.75	74.94
BLSTM-CRF	68.62	63.09	65.74	72.42	67.48	69.86

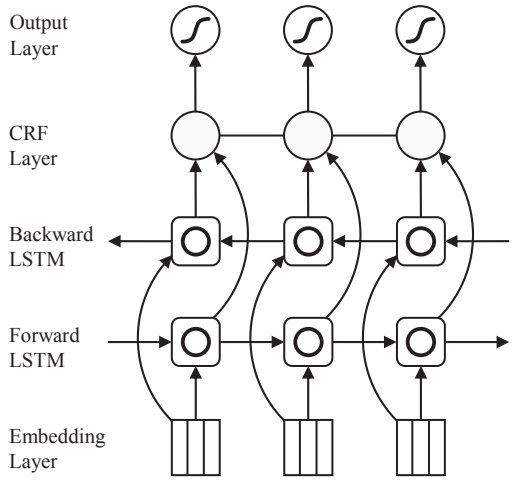

Figure 2: *Bidirectional LSTM with a CRF layer for punctuation prediction.*

3.4. Task and Input

In this paper, given an input sequence of words, we label each word based on the punctuation before this word. Specifically, we label each word with *comma*, *period* and *non-punctuation*. Our punctuation prediction only relies on lexical information, or simply word identities. Hence it works on pure texts without accessing to the speech waveforms (e.g., to extract prosodic features).

We encode each word in a one-hot scheme, and use an *embedding layer* as LSTM network input. The embedding layer connects the one-hot input and the subsequent LSTM layer, and it consists of a weight matrix $\boldsymbol{W}_e$ which will be updated in the whole network training procedure. The embedding layer works as follows: assume the size of vocabulary is n and the dimension of embedding layer is m. Then the weight matrix $\boldsymbol{W}_e$ of the embedding layer is of size $n \times m$ (W_{ij} is the weight of connection from unit i to unit j). When the input word's index is i, the embedding layer will pass the i-th row of $\boldsymbol{W}_e$ to the subsequent LSTM layer. That is, it acts like a lookup-table, making the training procedure more efficient. Furthermore, it efficiently represents each word with a number that is the index of this word in a prepared vocabulary. Because when the training data is large, directly representing each word with a high-dimensional one-hot vector is impractical and a waste of storage space.

4. Experiments

4.1. Data

We use Chinese texts from People's Daily to do the experiments. The training dataset contains 275,000 sentences and 17.5M words in total, while the test data set contains 34,600 sentences and around 0.2M words in total. The size of validation data set is similar to test data set. We focus on the prediction of two most common types of punctuation: comma and period. Thus we map question and exclamation marks to periods, replace colon and semicolon with commas, and remove all other punctuation symbols from the corpus. We use the Mecab toolkit [27] to do the word segmentation.

4.2. Metrics

We evaluate the punctuation prediction performance by precision (*prec.*), recall (*rec.*), and their harmonic mean—F1-score (F_1). We report these metrics for commas and periods, respectively, on the test set.

4.3. Experimental setup

We carry out two sessions of experimentation. The first session is designed to test LSTM and BLSTM with one or two hidden layers, which aims to see if considering the both past and future inputs and using a deeper model can improve the performance. The second session is to investigate whether modeling the context of output punctuation labels, through a CRF layer, can achieve performance gain. We train all the LSTM networks using the back-propagation algorithm. Network weights are updated for every training example using Adadelta with a decay rate of 0.95 and a constant of 1e-6. In all the networks, each LSTM layer consists of 100 single-cell LSTM blocks. Between the word input and the LSTM layer, we add an embedding layer whose dimension is set to 100 empirically. The input vocabulary consists of the 100K most frequent Chinese words and two special symbols—one for unknown words and the other for the end of input.

Table 3: *Punctuation prediction results for CRFs with and without output label context information. C: comma; P: period.*

Model	*prec.*(C)	*rec.*(C)	F_1(C)	*prec.*(P)	*rec.*(P)	F_1(P)
CRF-unigram	72.69	62.12	66.99	77.62	63.22	69.68
CRF-bigram	71.30	62.58	66.65	78.04	62.32	69.30

We use CRF++ toolkit [28] to build a CRF-based punctuation predictor as a baseline. All the CRF models used in this paper is a linear-chain CRF. The baseline CRF model, named CRF-unigram, considers an input context window size of 5 (previous two words, current word and future two words) without adding output label context information.

4.4. LSTM results

Table 1 presents the evaluation results for the LSTM models. We notice that BLSTM improves F_1 by 10.82% and 8.99%, in predicting commas and periods respectively, as compared with LSTM. This result supports our view that both past and future contexts are useful for punctuation prediction, and BLSTM is a superior architecture in this task.

When comparing two-layer models with single-layer models, we observe that using more hidden layers helps. For example, a 2layer-LSTM achieves about 4.17% relative F_1 gain for comma prediction, as compared with a single layer LSTM. We believe that the performance may be further improved when more layers are used, but training such models would be very time-consuming. Finally, the 2-layer BLSTM model achieves the best performance in punctuation prediction. It improves F_1 by 12.02% relative over CRF-unigram (baseline) and 11.24% relative over LSTM in predicting commas; it improves F_1 by 9.40% relative over CRF-unigram (baseline) and 10.86% relative over LSTM in predicting periods.

4.5. LSTM-CRF results

Table 2 summarizes the experimental results for LSTM models with and without output label context information. Surprisingly, we can clearly see that adding a CRF layer to model output context significantly decreases the performance, for both LSTM and BLSTM. This observation shows inconsistency with other sequential labeling tasks, e.g., NER [10, 26] and POS tagging [10], where the addition of a CRF layer clearly boosts the performance. We believe that the difference may come from the nature of the specific tasks. For POS tagging, which gives every word a POS tag (e.g., noun, verb, adjective, ...), the numbers of different labels are relatively balanced due to the language rules. For instance, two verbs can be neighbors. Restricted by the grammars, more importantly, the decision on the current tag is highly related to the previous and future tags. On the contrary, punctuation prediction is a highly *label imbalanced* task: most output labels are not punctuation. Specially in our corpus, the proportion of non-punctuation and punctuation is 85% versus 15%. Thus the CRF layer may learn more knowledge about transiting from non-punctuation to non-punctuation, and it will be more likely to predict the output label to be non-punctuation.

To validate the above hypothesis, we perform a sanity check on CRF. Specifically, we train a CRF-bigram model, which has the same input with CRF-unigram but also uses the previous output label and the current label as a bigram feature. Results in Table 3 show that apparent F_1 degradation is achieved by CRF-bigram. This sanity check confirms our hypothesis.

5. Conclusions

In this paper, we have compared the performance of a variety of LSTM based models for punctuation prediction. Our conclusions are as follows. 1) By modeling both past and future contexts of inputs, bidirectional LSTM significantly outperforms unidirectional LSTM. 2) A two-layer LSTM outperforms a single layer LSTM. 3) Modeling the output label dependences, by the addition of a CRF layer on top of an LSTM or BLSTM, does not improve performance on top of BLSTM. We have observed 23% and 24% relative reduction of errors, F_1 improved from 67.46 to 75.04 and from 68.76 to 76.23 for comma and period predictions, using 2-layer BLSTM models, in comparison to the baseline LSTM (the relative error rate reduction is computed as $\frac{(y-x)}{(100.0-x)} \times 100.0$).

6. Acknowledgements

This work was supported by the National Natural Science Foundation of China (Grant No. 61571363).

7. References

[1] S. Peitz, M. Freitag, A. Mauser, and H. Ney, "Modeling punctuation prediction as machine translation." in *IWSLT*, 2011, pp. 238–245.

[2] E. Cho, J. Niehues, and A. Waibel, "Segmentation and punctuation prediction in speech language translation using a monolingual translation system." in *IWSLT*. Citeseer, 2012, pp. 252–259.

[3] Y. Liu, A. Stolcke, E. Shriberg, and M. Harper, "Using conditional random fields for sentence boundary detection in speech," in *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 2005, pp. 451–458.

[4] C. Xu, L. Xie, G. Huang, X. Xiao, E. Chng, and H. Li, "A deep neural network approach for sentence boundary detection in broadcast news." in *INTERSPEECH*, 2014, pp. 2887–2891.

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

[6] J.-H. Kim and P. C. Woodland, "The use of prosody in a combined system for punctuation generation and speech recognition." in *INTERSPEECH*, 2001, pp. 2757–2760.

[7] H. Christensen, Y. Gotoh, and S. Renals, "Punctuation annotation using statistical prosody models," in *ISCA Tutorial and Research Workshop (ITRW) on Prosody in Speech Recognition and Understanding*, 2001.

[8] J. Huang and G. Zweig, "Maximum entropy model for punctuation annotation from speech." in *INTERSPEECH*, 2002.

[9] X. Wang, H. T. Ng, and K. C. Sim, "Dynamic conditional random fields for joint sentence boundary and punctuation prediction." in *INTERSPEECH*, 2012, pp. 1384–1387.

[10] Z. Huang, W. Xu, and K. Yu, "Bidirectional LSTM-CRF models for sequence tagging," *arXiv preprint arXiv:1508.01991*, 2015.

[11] C. Ding, L. Xie, J. Yan, W. Zhang, and Y. Liu, "Automatic prosody prediction for chinese speech synthesis using BLSTM-RNN and embedding features," in *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. IEEE, 2015, pp. 98–102.

[12] K. Yao, B. Peng, G. Zweig, D. Yu, X. Li, and F. Gao, "Recurrent conditional random field for language understanding," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 4077–4081.

[13] O. Tilk and T. Alumäe, "LSTM for punctuation restoration in speech transcripts," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[14] Y. Bengio, "Learning deep architectures for AI," *Foundations and trends® in Machine Learning*, vol. 2, no. 1, pp. 1–127, 2009.

[15] D. Beeferman, A. Berger, and J. Lafferty, "Cyberpunc: A lightweight punctuation annotation system for speech," in *Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on*, vol. 2. IEEE, 1998, pp. 689–692.

[16] A. Gravano, M. Jansche, and M. Bacchiani, "Restoring punctuation and capitalization in transcribed speech," in *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*. IEEE, 2009, pp. 4741–4744.

[17] W. Lu and H. T. Ng, "Better punctuation prediction with dynamic conditional random fields," in *Proceedings of the 2010 conference on empirical methods in natural language processing*. Association for Computational Linguistics, 2010, pp. 177–186.

[18] C. Xu, L. Xie, and Z. Fu, "Sentence boundary detection in chinese broadcast news using conditional random fields and prosodic features," in *Signal and Information Processing (ChinaSIP), 2014 IEEE China Summit & International Conference on*. IEEE, 2014, pp. 37–41.

[19] A. Graves, *Supervised sequence labelling*. Springer, 2012.

[20] T. Mikolov, M. Karafiát, L. Burget, J. Cernockỳ, and S. Khudanpur, "Recurrent neural network based language model." in *INTERSPEECH*, vol. 2, 2010, p. 3.

[21] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[22] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with LSTM," *Neural computation*, vol. 12, no. 10, pp. 2451–2471, 2000.

[23] F. A. Gers, N. N. Schraudolph, and J. Schmidhuber, "Learning precise timing with LSTM recurrent networks," *The Journal of Machine Learning Research*, vol. 3, pp. 115–143, 2003.

[24] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional LSTM and other neural network architectures," *Neural Networks*, vol. 18, no. 5, pp. 602–610, 2005.

[25] J. Lafferty, A. McCallum, and F. C. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," 2001.

[26] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer, "Neural architectures for named entity recognition," *arXiv preprint arXiv:1603.01360*, 2016.

[27] T. Kudo, "Mecab: Yet another part-of-speech and morphological analyzer," *Software available at http://mecab. sourceforge. net*, 2005.

[28] ——, "CRF++: Yet another CRF toolkit," *Software available at http://crfpp. sourceforge. net*, 2005.