Punctuation Prediction using a Bidirectional Recurrent Neural Network with Part-of-Speech Tagging

Chin Char Juin
Anglo-Chinese School (Independent),
Singapore, 139650
chincharjuin@gmail.com

Richard Xiong Jun Wei
Hwa Chong Institution,
661 Bukit Timah Road
Singapore, 269734
junweixiong@hotmail.com

Luis Fernando D'Haro
Institute for Infocomm Research,
A*STAR
Singapore, 138632
luisdhe@i2r.a-star.edu.sg

Rafael E. Banchs
Institute for Infocomm Research,
A*STAR
Singapore, 138632
rembanchs@i2r.a-star.edu.sg

Abstract— **Most automatic speech recognition (ASR) systems are incapable of generating punctuation, making it difficult to read the transcribed output and less appropriate for tasks such as dictation. This paper introduces a procedure to automatically insert punctuation into unpunctuated sentences by using a bi-directional recurrent neural network with attention mechanism and Part-of-Speech (POS) Tags. Using the WikiText Long Term Dependency Language Modelling Dataset and handling 11 different punctuation symbols, the model managed to achieve a punctuation error rate of 31.4% and an F1 score of 78.5%. When the system was trained on consecutive sentences and a smaller dataset using the Europarl v7 corpus, the model still managed to achieve a punctuation error rate of 48.1% and an F1 score of 64.7%. In both cases, our proposed system outperforms previous state-of-the-art systems trained on the same datasets, showing the advantage of using POS tags information and an encoder-decoder network.**

Keywords— deep neural networks, punctuation prediction, sequence-to-sequence approach

I. INTRODUCTION

Most current automatic speech recognition (ASR) systems are only able to generate a sequence of unpunctuated text. While this may be sufficient for simple commands such as search queries or quick message replies, longer sentences will undoubtedly be much more difficult to read and analyse, especially for applications such as speech transcription. Therefore, an accurate punctuation will allow ASR output to be much more readable, and more suitable for applications like machine transcription, translation or summarisation.

Previous attempts involved a variety of methods, such as the use of Dynamic Conditional Random Fields [1][2] and *n*-gram models [3][4]. There have also been approaches that rely on audio features, such as pause duration, and word intensity and pitch [5][10][11].

Recently, there has been an increased interest in recurrent neural networks for punctuation prediction as a machine translation task, where unpunctuated sentences are "translated" into punctuated sentences considering the punctuation symbols as any other word. In this kind of systems, Long Short Term Memory (LSTM) networks [22] have been reported in several papers [5][6] due to their enhanced ability to learn long-term dependencies. This enables the model to better recall past information allowing better prediction of the next word or punctuation symbol.

On the other hand, bi-directional LSTMs (bi-LSTMs) [23] can potentially provide better results in comparison with the conventional LSTMs [7][8] especially on long sentences or more complex grammatical structures. In these models, the network is split into two directions, encoding from both ends of the sentence concurrently. This allows "future" contextual information to be available to the network by looking ahead for contextual features that can improve the prediction accuracy.

Our system also utilise a bi-directional LSTM for predicting punctuation, where the novelty is that it implements the bi-directional LSTM in combination with an attention mechanism and POS tagging information, for predicting a higher number of punctuation symbols using a sequence-to-sequence topology.

The rest of the paper is organized as follows: section II describes the architecture of our proposed system, section III presents statistics of the databases used in our experiments and the best hyper-parameter values for our model, section IV discusses the obtained results, and section V presents the conclusions and future work.

II. METHOD

Figure 1 shows the architecture of our proposed system. It is based on an encoder-decoder architecture, which has successfully been used in different applications like neural machine translation [16][17], document summarization [20], or answer generation [21].

In this kind of architecture, the system is divided into two parts: a) an encoder that is responsible for taking a sequence of words in a source sentence (typically in a given language, or in our case a sentence without punctuation) and generates a low-dimensional fixed-length feature vector that encodes the full sequence of words in the sentence; and b) a decoder that realizes a generative model that is conditioned on the representation created by the encoder. Here, the decoder is used to generate a new sentence in another language (or in our case, to generate the same source sentence but including the punctuation symbols).

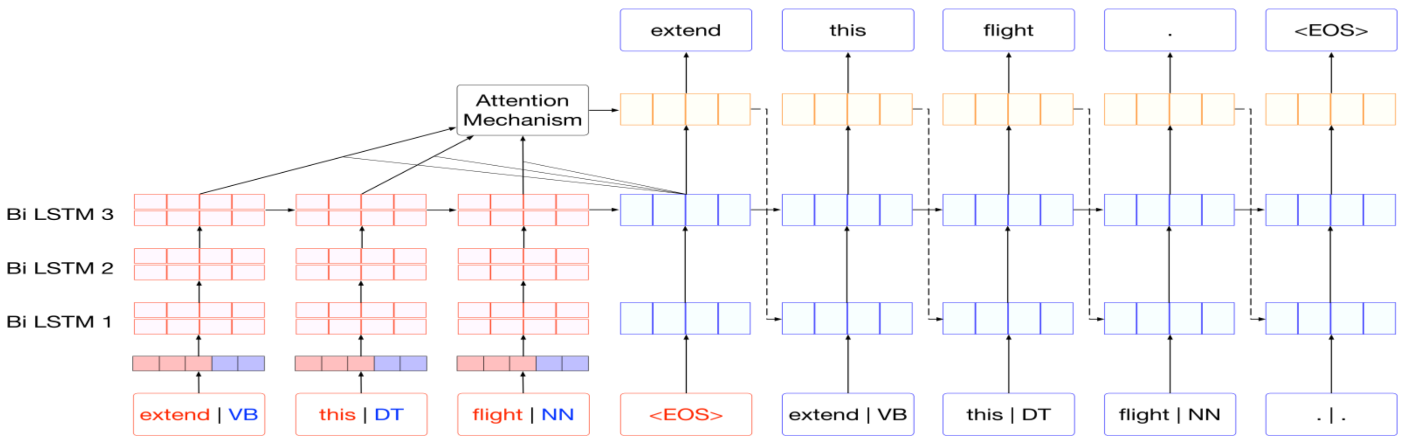

Figure 1. Architecture of our Encoder-Decoder machine translation system. Figure adapted from the OpenNMT website.

TABLE 1. STATISTICS OF THE DATABASES USED IN OUR EXPERIMENTS

Dataset	Split	No. Sent.	No. Words	Vocab Size	OOV	!	'	(	)	,	-	.	:	;	?	“
WikiText	Train	2673437	63939560	5002	1.895%	0.007%	0.442%	0.231%	0.232%	2.485%	0.428%	2.315%	0.057%	0.065%	0.004%	0.571%
	Dev	10000	256822	5002	1.717%	0.008%	0.391%	0.240%	0.240%	2.369%	0.412%	2.294%	0.051%	0.069%	0.002%	0.482%
	Test	3000	72234	5002	1.499%	0.003%	0.421%	0.269%	0.269%	2.389%	0.442%	2.389%	0.036%	0.061%	0.001%	0.442%
Europarl	Train	686355	44664758	5002	9.817%	0.017%	0.457%	0.000%	0.000%	4.936%	0.758%	3.255%	0.127%	0.085%	0.101%	0.000%
	Dev	10000	645373	5002	9.943%	0.014%	0.511%	0.000%	0.000%	4.956%	0.790%	3.267%	0.143%	0.088%	0.100%	0.000%
	Test	3000	199034	5002	9.717%	0.007%	0.528%	0.000%	0.000%	4.888%	0.787%	3.194%	0.103%	0.059%	0.068%	0.000%

In brief, the encoding process is broken down into several time steps depending on the number of words in the sentence, where at time step t, the current input word, x_t, is mapped into a low-dimensional vector embedding W_e using a lookup table, after applying a one-hot encoding scheme. Here, the vector embedding can be randomly initialized and trained together with the neural translation network or kept fixed by loading a pre-trained model. In both cases, the main purpose of the embedding is to encode the semantics of each word based on the co-occurrences of similar words in similar contexts [24].

In our system, we extend the semantic information encoded in the word embedding (red vectors in Figure 1) by appending syntactic information of the Part-Of-Speech (POS) tags of each word in the source sentence. Our hypothesis is that the POS tags provide the model with further information about the nature of each word in the sentence, allowing the system to differentiate how the same word is used in different contexts. For instance, many words in English could be used either as adjectives or as nouns, and their different functions in the sentence would affect where punctuation should be inserted.

In our implementation, the system generates a new vector embedding (purple colour in the figure) for the POS tags whose size is automatically calculated based on the number of different POS tags found in the training set (35 in our case). Then, both the word and POS embedding are merged by concatenating them.

After this, the sequence of input word embeddings, $We_1 \dots We_T$, is processed through a bi-directional LSTM layer which internally consists of two layers of LSTM units: a) the first one processes the sentence in the forward direction, while b) the second one processes the sentence in the backward direction. Here, by using the bi-directional recurrent neural network, the model is able to predict punctuation insertions through contextual evidence from before and after the current word.

Then, the system concatenates the internal hidden states of both LSTMs at each time t, generating a vector representation that encodes the semantics and syntax of the current word and POS information as well as the information from the previous context t-1. In addition, by stacking several Bi-LSTM layers (3 in our case), the system is able to model even longer term dependencies, which is relevant for longer or more complex sentences.

After the encoder finishes processing the source sentence, a semantically/syntactically encoded vector is generated, and is then fed into the decoder in order to generate a new sentence. Here, the decoder uses the encoded vector and the previously generated word (or the special symbol End-Of-Sentence, EOS, to indicate the beginning of the decoded sentence) to calculate the output probabilities for all possible words in the target vocabulary (in our case it is the same vocabulary as the source sentence extended with the punctuation symbols). The word with the highest probability is then selected as generated word t, which is then used as input to generate the next word. The process is repeated until the system outputs the EOS symbol.

In order to improve the performance of the encoder-decoder two modifications are typically used: a) an attention mechanism [18][19] that allows for identifying important contextual clues in the source sentence that highlight the inclusion of specific punctuation marks in the target sentence (for example, the model may call attention to specific words that point to the inclusion of quotation marks, and therefore adjust the possible outputs in the appropriate direction); and b) a beam-search algorithm [25] that allows the decoder for keeping several hypotheses and choosing the best one based on a scoring function, instead of just selecting the most probable word.

For a more detailed explanation of the encoder-decoder architecture and current research, we recommend reading [16].

III. Datasets and Model Parameters

A. Datasets

In this section we will describe the two datasets used in our experiments. The first one, WikiText, was selected mainly because of the high number of sentences, and wide coverage of the punctuation terms which is important to train statistical approaches like the encoder-decoder network and to test the system on written text; while the second dataset, Europarl v7, was mainly selected to test the robustness of the system given a lower number of sentences, punctuation coverage, and on conversational text which is common in applications where Automatic Speech Recognition (ASR) systems are used.

In both databases, the data was pre-processed by splitting them into individual sentences. Then, capitalization information was removed, as ASR systems usually do not provide such information. Then, the selected set of 11 different punctuation symbols (listed out in the headers of Table 1 and Table 3) was removed. This set represents, up to the best of our knowledge, the widest number of punctuation symbols targeted to be predicted in the literature; in comparison, most previous reported models were usually trained to predict only four symbols [5][6][7]. It is important to mention that in order to focus our system only on those cases where the punctuation symbols are used as such, we did not remove them in cases like numbers or abbreviations, where they are treated as part of the current word. For example, in names like "h. gammarus", the period involved is not removed, and is treated as attached to the previous word. Then, pre-processed sentences were split into two sets: a source set with unpunctuated sentences, and a target set with the punctuation intact.

Then, POS tags were added to the source sentences by using the Natural Language Toolkit (NLTK), appending them at the end of each word with the special Unicode character FFE8 (i.e. symbol |) as required by the toolkit we used in our experiments (see section III.B). As commented before, adding the tags as additional features and not as part of the main word, allows the extra information to influence training and prediction without affecting the original word embeddings.

1) WikiText

The WikiText Long Term Dependency Language Modelling Dataset (a.k.a. WikiText-103) is a collection of over 100 million tokens extracted from a set of high quality articles on Wikipedia [12]. From this dataset we randomly selected 2.6M sentences for the training data, while 10K sentences were selected for the validation data, and for the testing data, 3K out-of-domain sentences were used.

The idea of using this corpus is to train the model on a large amount of data which is typically required for the deep neural networks in order to model the different words in the vocabulary (including the punctuation symbols), but also to test the model to punctuate written text so as to assess the effectiveness of the system in applications such as document writing.

2) Europarl v7

The Europarl V7 [14][15] is a parallel corpus extracted from the European Parliament web site by Philipp Koehn and used mainly for research on statistical machine translation. The database is made of manually transcribed sentences mainly from the speeches given by the members of the parliament. The corpus consists of more than 2.2M English sentences, from where we extracted 680K short paragraphs, each containing 2-3 consecutive sentences for training purposes.

The idea of using this corpus was to test the model's ability to punctuate longer and consecutive sentences, as well as to train the system on sentences that could resemble more the way people talk rather than on written text. Similarly, 10K short paragraphs were chosen for validation, and 3K out-of-domain short paragraphs were chosen for testing. All other hyper-parameters and procedures were left intact.

B. Training Parameters

The model was trained using OpenNMT[1] system [13], which was initially implemented by Kim et al [9] and now maintained by the Harvard NLP and Systran group. The toolkit provides an open-source neural machine translation engine based on the Torch/PyTorch mathematical libraries.

Our models were trained in Ubuntu 14.04 with an NVIDIA GTX 1060 GPU. Table 2 summarises the final parameters chosen for our final model, which includes a global attention mechanism [19] with beam search, and trained for 10 epochs.

[1] http://www.opennmt.com

TABLE 2. SUMMARY OF TRAINING PARAMETERS

Parameter	Details
Word embedding size	256
Number of RNN layers	3
RNN hidden unit size	512
Bi-directional LTSM	Enabled
Beam Search	1

IV. RESULTS AND DISCUSSION

The models for both corpuses were evaluated in terms of both per punctuation and overall word-error-rates and F1-scores. An overall BLEU score was also computed. For both corpora, a baseline state-of-the-art system (T-BRNN) [7] was also trained in order to compare the performance of the systems. Here, the main difference with the T-BRNN is the usage of POS tags and the stacking of several Bi-LSTM layers.

A. Results for the WikiText Dataset

From the results summarised in Table 3, the model for the WikiText Dataset was able to predict apostrophes and periods with the lowest error rates and highest F1 scores. This is to be expected, as there are certain tell-tale features that allow the model to predict them highly accurately. For periods, the model learns quickly to insert them at the end of each sentence, since our corpus is split into individual sentences. As for the apostrophes, it can usually be determined by the "_POS" part-of-speech tag, which refers to possessives such as an " 's " or " 'd ".

From the results, it seems that the model was also quite successful at adding in commas, with a relatively low WER of 43.3% and a high F1 score of 75.9. Looking at the predictions, the system also shows that it is able to insert commas at suitable positions either for separating different clauses (see Example 1), or different items in a list (see Example 2).

Example 1:

Source: although it is frequently found in american lobsters the disease has only been seen in captive h. gammarus where prior occupation of the tanks by h. americanus could not be ruled out

Prediction: although it is frequently found in American lobsters, the disease has only been seen in captive h. gammarus, where prior occupation of the tanks by h. americanus could not be ruled out.

Reference: although it is frequently found in american lobsters, the disease has only been seen in captive h. gammarus, where prior occupation of the tanks by h. americanus could not be ruled out.

Example 2:

Source: it may fetch very high prices and may be sold fresh frozen canned or powdered

Prediction: it may fetch very high prices and may be sold fresh, frozen, canned, or powdered.

Reference: it may fetch very high prices and may be sold fresh, frozen, canned, or powdered.

The system is also able to learn complex formatting styles for different numbering systems (e.g. dates, money, coordinates, etc.), as shown in example 3.

Example 3:

Source: it came from 57 53 n 11 32 e near marstrad sweden 48 kilometres or 30 miles northwest of gothenburg but both it and the <unk> have since been lost.

Prediction: it came from 57° 53' n 11° 32' e near marstrad, sweden, 48 kilometres or 30 miles northwest of gothenburg, but both it and the <unk> have since been lost.

Reference: it came from 57° 53' n 11° 32' e near marstrad, sweden, 48 kilometres or 30 miles northwest of gothenburg, but both it and the <unk> have since been lost.

As for the other punctuation marks, the model had more difficulty in predicting them, as shown by their higher error rates. Exclamation and question marks were the most difficult to predict, mainly due to them being too rare in the dataset. Quotation marks were the next hardest to predict, mainly due to the ambiguity on whether quotation marks can be inserted. For instance, there are many sentences that can be grammatically sound with or without the quotation marks, and unless there are features such as "said", "told", etc. it is difficult to determine when to use the quotes.

Colons and semicolons were also slightly harder to predict compared to commas and dashes, at 50.9% and 52.8% WER respectively compared to 43.3% and 40.5% WER for commas and dashes respectively. This may be because they are significantly rarer than the rest and/or due to their similarity in use with commas.

Overall, as shown in Table 3, the error rate for punctuation only was quite low at 31.4% for the WikiText dataset. Comparing with T-BRNN [7], our system performed on par in terms of F1-score, scoring a 78.5 compared to the 80.0 of the T-BRNN, despite our system having to predict more punctuations. This shows the potential of our system in restoring punctuation for individual sentences, allowing it to be used in applications such as voice transcription for text messages or web searches.

B. Results for the English Europarl v7

To further test the performance of our model on longer sequences featuring consecutive sentences, we also trained and evaluated it with the Europarl v7 corpus. Since this dataset is composed of a repository of transcribed speeches, it also matches more closely the eventual output from an ASR compared to the WikiText dataset, which was derived from Wikipedia articles. The results for this corpus are also summarised in Table 3. The trends for the different punctuations were similar to the model trained with the WikiText dataset; however, colons, semicolons, dashes, and quotation marks had a much lower F1-score and higher WER as it is significantly rarer in spoken sentences. Also, brackets did not appear at all in the corpus therefore not tested.

Due to the fact that 2-3 sentences were trained and tested consecutively, the performance of the model was expectedly poorer when compared to the initial model trained with the WikiText dataset. Even so, the overall F1 metrics posed are still significantly higher than the past state-of-the-art trained with the same datasets (T-BRNN [7]), with an improvement in F1-score by more than 10%, despite having to deal with a wider variety of punctuation marks.

V. CONCLUSION

This paper presented a bi-directional LSTM with attention mechanism and POS tagging that improves the state-of-the-art in punctuation prediction for individual sentences. A set of 11 different punctuation marks were predicted, which is larger than most sets previously reported in the literature. The model with the WikiText dataset achieved a relatively good WER of 31.4% when counting only the punctuations, and an F1-score of 78.5, which is on par with past systems despite having to predict more types of punctuations. The model trained with English Europarl v7 produced poorer results compared to that trained with WikiText due to the consecutive sentences being punctuated at once, but even so, it achieved a comparable WER of 48.1% for punctuations only, and an overall F1-score of 64.7, which is significantly higher than what past systems can achieve. As such, the usage of POS tagging has shown to bring significant improvements over previous state-of-the-art. Moreover, the overall results in Table 4 shows that our advantage of using the POS tags, allowing our system to vastly outperform similar models that do not use POS tags.

However, one limitation of our system in general, for both models trained with WikiText and Europarl, is that the model may still sometimes substitute certain words of the original sentence, even when out-of-vocabulary words are replaced by UNK tokens. This can be fixed by applying a biased decoding feature, which will be worked on in the future.

Furthermore, this architecture will be able to correctly keep the same structure and words of the source sentence (i.e. avoiding the deletion of existing words or the insertion of new words) resolving a problem with the system, given the generative nature of the decoder.

Future work could include training with a larger dataset of sentences, especially with more speech-oriented corpora. We also attempted to implement biased decoding to further improve the predictions while reducing the possibility of inserting or deleting important words in the source sentence. However, this is not currently supported by the OpenNMT Torch system, and we faced significant difficulty in trying to implement it ourselves. As a result, we only managed to test it on a similar TensorFlow system, to determine if it indeed provides any improvement in performance. The preliminary testing results showed that this reduced the WER for punctuations by 31.3%. As such, this is a highly promising technique we hope to explore further.

TABLE 3: PER PUNCTUATION WER AND F1 SCORES

Model	Dataset	Punctuation	!	'	(	)	,	-	.	:	;	?	“	All (Punct)
BiLSTM-POS	**WikiText Dataset**	***Error Rate (%)***	100	15.4	39.2	40.9	43.3	40.5	2.72	50.9	52.8	100	67.2	**31.4**
		Precision (%)	0.00	94.8	75.4	81.1	79.5	83.0	98.1	72.5	66.2	0.00	79.5	**83.4**
		Recall (%)	0.00	85.2	66.2	66.1	72.6	67.2	99.1	54.7	50.6	0.00	40.6	**74.1**
		F1 Score (%)	0.00	89.7	70.5	72.8	75.9	74.2	98.6	62.4	57.3	0.00	53.7	**78.5**
	English Europarl v7	***Error Rate (%)***	100	25.0	—	—	57.8	88.9	30.1	85.9	99.2	60.0	97.9	**48.1**
		Precision (%)	0.00	94.6	—	—	76.4	28.9	81.8	26.8	2.30	59.4	12.0	**75.1**
		Recall (%)	0.00	76.3	—	—	48.3	13.8	84.4	18.5	1.71	58.5	2.08	**56.8**
		F1 Score (%)	0.00	84.5	—	—	59.2	18.7	83.1	21.8	1.96	59	3.55	**64.7**
T-BRNN [7]	**WikiText Dataset**	***Precision (%)***	0.00	—	—	—	77.8	86.3	85.1	73.5	65.4	0.00	—	**81.8**
		Recall (%)	0.00	—	—	—	73.5	75.4	86.6	47.2	19.1	0.00	—	**78.4**
		F1 Score (%)	0.00	—	—	—	75.6	80.5	85.8	57.5	29.6	0.00	—	**80**
	English Europarl v7	***Precision (%)***	0.00	—	—	—	63.8	14.3	77.2	30.0	0.00	76.5	—	**67.4**
		Recall (%)	0.00	—	—	—	53.9	0.2	39.5	4.4	0.00	9.9	—	**45.6**
		F1 Score (%)	0.00	—	—	—	58.4	0.4	52.3	7.6	0.00	17.6	—	**54.4**

TABLE 4: SUMMARY OF OVERALL WER, BLEU AND F1 SCORES

Model	Dataset	Overall WER (%)	BLEU Score	Precision (%)	Recall (%)	F1 Score (%)
BiLSTM-POS	WikiText Dataset	6.14	0.8696	83.4	74.1	78.5
	English Europarl v7	5.17	0.8898	75.1	56.8	64.7
T-BRNN [7]	WikiText Dataset	3.76	—	81.8	78.4	80.0
	English Europarl v7	5.99	—	67.4	45.6	54.4

REFERENCES

[1] Peitz, S., Freitag, M., Mauser, A. and Ney, H. (2011). Modelling Punctuation Prediction as Machine Translation. *The International Workshop on Spoken Language Translation (IWSLT)*, pp.238-245.

[2] Lu, W. and Ng, H. (2010). Better Punctuation Prediction with Dynamic Conditional Random Fields. *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pp.177–186.

[3] Matusov, E., Mauser, A. and Ney, H. (2006). Automatic Sentence Segmentation and Punctuation Prediction for Spoken Language Translation. *The International Workshop on Spoken Language Translation (IWSLT)*, pp.158-165.

[4] Cho, E., Niehues, J. and Waibel, A. (2011). Segmentation and Punctuation Prediction in Speech Language Translation Using a Monolingual Translation System. *The International Workshop on Spoken Language Translation (IWSLT)*.

[5] Ottokar Tilk, Tanel Alumäe. (2015). LSTM for Punctuation Restoration in Speech Transcripts. *Interspeech 2015*.

[6] Xu, K., Xie, L. and Yao, K. (2016). Investigating LSTM for Punctuation Prediction. *The International Symposium on Chinese Spoken Language Processing (ISCSLP)*.

[7] Ottokar Tilk, Tanel Alumäe. (2016). Bidirectional Recurrent Neural Network with Attention Mechanism for Punctuation Restoration. *Interspeech 2016,* pp.3047-3051.

[8] Salimbajevs, A. (2016). Bidirectional LSTM for Automatic Punctuation Restoration. Human Language Technologies – The Baltic Perspective: Proceedings of the Seventh International Conference Baltic HLT 2016, pp.59-65.

[9] Kim, Y. Sequence-to-Sequence Learning with Attentional Neural Networks. Available at: https://github.com/harvardnlp/seq2seq-attn [Accessed 14 Dec. 2016].

[10] 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.

[11] Kolář, Jáchym, Jan Švec, and Josef Psutka, "Automatic punctuation annotation in Czech broadcast news speech," in SPECOM 2004, Saint Petersburg, Russia, 2004.

[12] Merity, S., Xiong, C., Bradbury, J. and Socher, R., 2016. Pointer Sentinel Mixture Models. arXiv preprint arXiv:1609.07843.

[13] Klein, G., Kim, Y., Deng, Y., Senellart, J. and Rush, A.M., 2017. OpenNMT: Open-Source Toolkit for Neural Machine Translation. arXiv preprint arXiv:1701.02810.

[14] Koehn, P., 2005, September. Europarl: A parallel corpus for statistical machine translation. In MT summit (Vol. 5, pp. 79-86).

[15] Tiedemann, J., 2012, May. Parallel Data, Tools and Interfaces in OPUS. In LREC (Vol. 2012, pp. 2214-2218).

[16] Sutskever, I., Vinyals, O. and Le, Q.V., 2014. Sequence to sequence learning with neural networks. In Advances in neural information processing systems (pp. 3104-3112).

[17] Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H. and Bengio, Y., 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078.

[18] Bahdanau, D., Cho, K. and Bengio, Y., 2014. Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473.

[19] Luong, M.T., Pham, H. and Manning, C.D., 2015. Effective approaches to attention-based neural machine translation. arXiv preprint arXiv:1508.04025.

[20] Rush, A.M., Chopra, S. and Weston, J., 2015. A neural attention model for abstractive sentence summarization. arXiv preprint arXiv:1509.00685.

[21] Vinyals, O. and Le, Q., 2015. A neural conversational model. arXiv preprint arXiv:1506.05869.

[22] Sepp Hochreiter; Jürgen Schmidhuber (1997). "Long short-term memory". Neural Computation. 9 (8): 1735–1780. doi:10.1162/neco.1997.9.8.1735. PMID 9377276

[23] Schuster, Mike, and Kuldip K. Paliwal. "Bidirectional recurrent neural networks." Signal Processing, IEEE Transactions on 45.11 (1997): 2673-2681.2. Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan.

[24] Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S. and Dean, J., 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems* (pp. 3111-3119).

[25] Schwartz, R. and Chow, Y.L., 1990, April. The N-best algorithms: an efficient and exact procedure for finding the N most likely sentence hypotheses. In Acoustics, Speech, and Signal Processing, 1990. ICASSP-90., 1990 International Conference on (pp. 81-84). IEEE.