Human Language Technologies – The Baltic Perspective
I. Skadiņa and R. Rozis (Eds.)

doi:10.3233/978-1-61499-701-6-59

Bidirectional LSTM for Automatic Punctuation Restoration

Askars SALIMBAJEVS [a,1]
[a] *Tilde, Latvia*

Abstract. The output of generic automatic speech recognition systems consists of raw word sequences without any punctuation symbols. When sequences are longer, it is difficult for humans to read and understand them. Also, many natural language understanding and processing tools expect that input will contain punctuation. We present a bidirectional recurrent neural network for punctuation restoration in speech utterances. The proposed model showed promising results, F1-scores of 0.732 for commas and 0.708 for periods on raw output from a speech recognizer.

Keywords. Speech recognition, punctuation, natural language processing, bidirectional, LSTM, recurrent neural networks

1. Introduction

Since people usually do not pronounce punctuation when speaking, the output of generic automatic speech recognition (ASR) systems is a raw word sequence without any punctuation symbols. This is sufficient in many cases, such as search queries, voice commands, or simple short informal messages, where word sequences are typically short. However, when sequences are longer, it is difficult for humans to read and understand them. Also, many of the natural language processing (NLP) and understanding (NLU) tools are typically incompatible or perform badly with such raw input.

There have been many previous studies on automatic punctuation restoration in speech transcripts. One of the most widely used approaches for punctuation restoration is based on the so-called hidden event language model (LM), which uses a traditional N-gram statistical model trained on texts that include punctuation tokens [1]. During decoding, the hidden event LM is used to predict where punctuation symbols should be inserted, based on n-gram probability of such sequences of words and punctuation.

Punctuation restoration can also be solved using conditional random fields (CRFs) [2], [3] and recurrent neural networks [4].

Many approaches combine textual information with acoustic/prosodic features [4], [5], [6]. However, this requires a corpus that simultaneously contains both acoustic/prosodic and punctuation annotations, which is rare.

This paper describes a punctuation restoration model for a Latvian dictation system. The model is a bidirectional long short-term memory model (BLSTM). BLSTM is a bidirectional recurrent neural network (BRNN) [7] that contains long short-term memory cells (LSTM) [8] in its hidden layers. Recurrent networks and LSTM recurrent networks

[1] Corresponding Author: Askars Salimbajevs, SIA Tilde, Vienibas gatve 75A, Riga, Latvia, LV-1004; E-mail: askars.salimbajevs@tilde.lv

have been used in many sequence labeling tasks in speech recognition and spoken language understanding [9], [10]. For example, forward LSTM and bidirectional LSTM can be used in acoustic modeling for phoneme classification [11], [12] or in language modeling [13].

LSTM networks have already been used for punctuation recovery in punctuation-free tasks [4]. However, we are not aware of any prior work using bidirectional LSTM, and, to the best of our knowledge, there is very little or no research on punctuation restoration for Latvian.

Unfortunately, there is no Latvian corpus that contains punctuation and acoustic (e.g., pauses) annotations. Therefore, no acoustic features were used in this work, and the model is trained on purely textual information.

The following section describes training and evaluation data. Section 3 describes the BLSTM model for punctuation restoration. Section 4 presents experimental results. Section 5 concludes the paper.

2. Data

In this work, a 50 million sentence text corpus is used, which was collected by crawling Latvian web news portals. The corpus contains about 905M million tokens, i.e., words and punctuation symbols. Because text was automatically collected, it may contain a lot of noise, garbage, and spelling errors. Also, we need to replace punctuation and other special symbols with their respective full words (i.e., "comma", "period", "at sign", etc.), convert numbers from digits to written form, expand abbreviations (e.g., "km"), etc. To deal with these issues, the corpus is preprocessed as follows:

- The raw text is processed with natural language processing tools, which perform tokenizing, garbage filtering (for example, mixed case tokens, non-alphanumeric tokens), number conversion (from digits to words and tries to find correct inflection), some abbreviation expansion, and true casing.
- Optionally, the corpus can be stemmed to minimize vocabulary and model size.
- If the corpus is stemmed, a vocabulary is created from 100,000 of the most frequent stems. Otherwise, the vocabulary is created from 800,000 of the most frequent words, which are checked by a spell checker. Out-of-vocabulary (OOV) tokens are replaced with the "<UNK>" token.
- Next, punctuation symbols are replaced by the respective word forms and some more garbage processing is performed (incorrect punctuation sequences are filtered).
- Also, because the web news corpus is already segmented into sentences (this segmentation is not perfect, of course) during text collection, we randomly glue some sentences together to obtain utterances that contain 2 or more sentences. This is done in order to have training samples for cases where the model is expected to split an utterance into several sentences.

After processing, the text corpus contains 21 million utterances, 589 million words, 41 million punctuation commands, and 40 million punctuation symbols. From this processed corpus, two held-out sets are selected: 2000 utterances for validation during training and 3000 utterances for evaluation. All other data is used for model training.

During the development of the punctuation model, a small 1-hour long speech corpus of the debates in the Parliament of Latvia was collected. The corpus contains 439 segments, which were recorded by about 300 different speakers (estimated). The interesting feature of this corpus is that its annotation contains punctuation. This makes it possible to perform punctuation restoration on the ASR output and to compare it with the reference annotation.

3. Method

The goal of this work is to develop a model for punctuation restoration in speech transcripts. In this work, we focus on two of the most frequent and probably most important punctuation types – commas and periods.

3.1. Hidden-event Language Model

The given task can be solved using the hidden-event LM approach [1], which uses a traditional N-gram language model trained on a corpus that contains regular text and so-called hidden-events. Applying this model to a raw sequence of words produces a sequence of the most likely words and hidden-events between them.

In our case, hidden-events are commas. We annotate our 21 million utterance training corpus with these hidden-events and then train a 4-gram language model.
During decoding, the hidden-event LM produces a sequence of words and "punctuation-events" that can be unambiguously converted into a text with punctuation.

3.2. Bidirectional LSTM

We propose to solve the given problem by using a bidirectional recurrent neural network with long short-term memory (BLSTM). The model we train can be described as a classifier that will predict whether the current word in a word sequence corresponds to one of these 3 classes:

- A period must be inserted after this word.
- A comma must be inserted after this word.
- This word is a regular word and should be left as is.

3.2.1. Motivation

The BLSTM approach has several advantages over the classic hidden event N-gram LM, the LSTM recurrent model, and traditional feed-forward neural networks.

Probably, the most important problem of the N-gram model is weak generalization ability because of the data sparsity. For example, for a vocabulary of size 100,000, the number of all possible 3-grams is 10^{15}, but our training corpus has only about 50 million 3-grams. Many different smoothing methods were developed to deal with this issue. However, as it is known from language modeling, recurrent neural networks are much better at generalizing to unseen sequences [14], [15].

The generalization problem can also be solved by using simple feed-forward neural networks. However, they share a common weakness with the N-gram models, as their context size is limited to a fixed number of tokens.

On the other hand, RNNs can theoretically use an unlimited previous context for their decisions. In practice, however, the ability of pure RNNs to remember long-term dependencies is limited, so RNNs are combined with LSTM units to make remembering long contexts possible.

Bidirectional LSTM RNN further extends RNN LSTM models by allowing to use not only the previous context of the word but also the next context, which can be important for the correct punctuation.

3.2.2. Model architecture

The BLSTM neural network consists of two LSTM layers, an input layer, and an output layer. The architecture of our BLSTM model is shown in Figure 1.

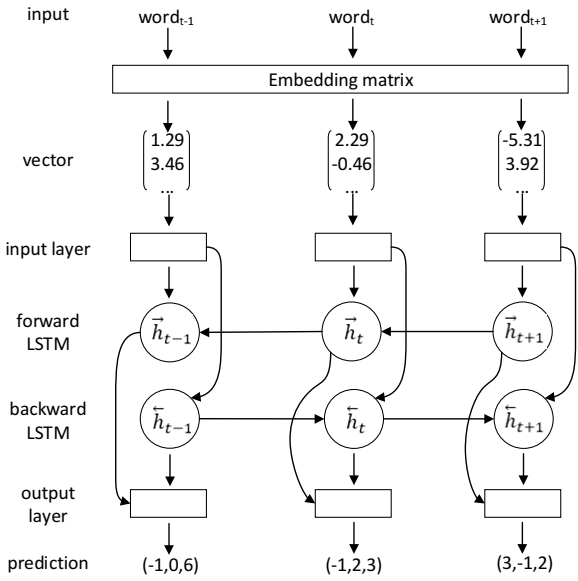

Figure 1. *BLSTM network for punctuation restoration.*

Since, in our case, the input consists of words, we use a special embedding matrix that translates each word to a 128-dimensional vector. This matrix consists of 100,000 (in case of stems) or 800,000 (when using full word forms) rows and 128 columns; each row vector represents a word from the vocabulary. The matrix is initialized with random values from uniform distribution in the range of (-1, 1) and then is trained together with the whole network.

Word vectors are then fed into the input layer of BLSTM that is common for both forward and backward 128-dimensional hidden LSTM layers. The backward LSTM layer $\overleftarrow{h}_t$ has a recurrent connection with the hidden layer $\overleftarrow{h}_{t-1}$ from the previous word, and the forward LSTM layer $\vec{h}_t$ is connected accordingly to the hidden layer activations $\vec{h}_{t+1}$ from the next word. Baseline LSTM units with forget gates and without peephole connections are used in this work. Hidden layer activations $\vec{h}_t$ and $\overleftarrow{h}_t$ are then passed to the softmax output layer, which calculates the probabilities of each class. To sum up, the

forward pass of punctuation BLSTM can be described with the following equations (1-6):

$$x_t = embedding(word) \tag{1}$$

$$inp_t = (W_I x_t + b_I) \tag{2}$$

$$\overrightarrow{h_t} = LSTM(inp_t, \overrightarrow{h_{t+1}}) \tag{3}$$

$$\overleftarrow{h_t} = LSTM(inp_t, \overleftarrow{h_{t-1}}) \tag{4}$$

$$\overleftrightarrow{h_t} = concat(\overrightarrow{h_t}, \overleftarrow{h_t}) \tag{5}$$

$$pred(t) = softmax\left(W_o \overleftrightarrow{h_t} + b_O\right) \tag{6}$$

where W_O and W_I are weight matrices, b_O and b_I are bias vectors for output and input layers respectively, inp_t is input layer activations, *embedding* is a lookup function in an embedding matrix, x_t is a word embedding vector, and *word* is the word we are trying to classify.

The model is implemented using TensorFlow [16] framework and trained by optimizing cross-entropy using the Adam method [17]. All weights are initialized with random values from normal distribution with mean 0 and standard deviation 1. During training, weights are updated in "mini-batches" of 128 sequences. The learning rate at the start of the training is 0.0005 and is halved each time when there is no improvement on the validation set (adopted from [4] and [18]). Training is stopped when the learning rate has been decreased 3 consecutive times and no improvement has been observed, or it is stopped when the maximum number of 20 epochs has been reached.

4. Results

We evaluate both hidden-event and BLSTM models on three different metrics:

- Precision for each class and overall precision.
- Recall for each class and overall recall.
- F1 for each class and overall F1.

First, the evaluation is performed on a held-out dataset of 3000 utterances; the results are presented in Table 1. The baseline hidden-event LM is outperformed by the proposed BLSTM model in all metrics by a large margin.

Table 1. *Results of the evaluation on a held-out test data*

Metric		Hidden-event	BLSTM
Precision			
	comma	0.726	**0.826**
	period	0.600	**0.798**
Recall			
	comma	0.444	**0.655**
	period	0.155	**0.713**
F1			
	comma	0.551	**0.731**
	period	0.246	**0.753**

The biggest improvement is gained in recall; the difference is especially large (0.713 compared to 0.155) for periods. Noticeable improvement is also seen in precision, from 0.726 to 0.826 (14% relative) for commas and from 0.600 to 0.798 (33% relative) for periods.

Next, we performed evaluation on a 1-hour long speech corpus of debates in the Parliament of Latvia. This time, raw ASR transcripts were used as input for both models. First, word error rate (WER) is measured, and commas and periods are calculated as word tokens. Then, the alignment from the WER calculation is used to evaluate precision, recall, and F1. The results are presented in Table 2.

Table 2. *Results of the evaluation on an ASR output*

Metric		Hidden-event	BLSTM
Precision			
	comma	0.709	**0.845**
	period	0.732	**0.789**
Recall			
	comma	0.497	**0.646**
	period	0.149	**0.642**
F1			
	comma	0.585	**0.732**
	period	0.247	**0.708**
WER		28.25%	**15.68%**

Again, in comparison with the hidden-event model, improvement is gained in all metrics, and the biggest difference is seen in recall for periods (0.642 compared to 0.149). Overall, the proposed model achieved F1-scores of 0.732 for commas and 0.708 for periods, which is very close to the numbers obtained on a held-out set.

5. Conclusion

In this paper, we presented a model for punctuation restoration in transcripts from an automatic speech recognition system. We concentrated on the restoration of two of the most frequent punctuation symbols: commas and periods. The model was trained on a large text corpus of written text that was automatically collected from the Web. The model evaluation showed promising results. BLSTM outperformed the hidden-event LM and achieved F1-scores of 0.732 for commas and 0.708 for periods on raw ASR transcripts of 1-hour long speech recordings of debates in the Parliament of Latvia.

Acknowledgements

The research has been supported by the ICT Competence Centre project (ID 1.2.1.1/16/A/007), within the project "2.3. Speech recognition and synthesis for professional application", of EU Structural funds.

References

[1] A. Stolcke, E. Shriberg, R. A. Bates, M. Ostendorf, D. Hakkani, M. Plauch, G. Tr, and Y. Lu, "Automatic detection of sentence boundaries and disfluencies based on recognized words," in ICSLP 1998, Sydney, Australia, 1998.

[2] W. Lu and H. T. Ng, "Better punctuation prediction with dynamic conditional random fields," in EMNLP 2010, Cambridge, MA, USA, 2010.

[3] J. Lafferty, A. McCallum, and F. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in proc. of ICML'01, 2001.

[4] Tilk, O., & Alumäe, T. (2015). LSTM for Punctuation Restoration in Speech Transcripts. In Interspeech 2015. Dresden, Germany.

[5] J. Kolář, J. Švec, and J. Psutka, "Automatic punctuation annotation in Czech broadcast news speech," in SPECOM 2004, Saint Petersburg, Russia, 2004.

[6] J. Huang and G. Zweig, "Maximum entropy model for punctuation annotation from speech," in ICSLP 2002, Denver, CO, USA, 2002

[7] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," IEEE Trans. Signal Process., vol. 45, no. 11, pp. 2673–2681, 1997.

[8] S. Hochreiter, S. Hochreiter, J. Schmidhuber, and J. Schmidhuber, "Long short-term memory.," Neural Comput., vol. 9, no. 8, pp. 1735–80, 1997.

[9] K. Yao, G. Zweig, M. Hwang, Y. Shi, and D. Yu, "Recurrent Neural Networks for Language Understanding," Proc. Interspeech, no. 1, pp. 104–108, 2013.

[10] K. Yao, B. Peng, Y. Zhang, D. Yu, G. Zweig, and Y. Shi, "Spoken language understanding using long short-term memory neural networks," 2014 IEEE Spoken Language Technology Workshop (SLT). pp. 189–194, 2014.

[11] A. Senior, H. Sak, and I. Shafran, "Context dependent phone models for LSTM RNN acoustic modelling," in ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings, 2015, vol. 2015-August, pp. 4585–4589.

[12] H. Sak, A. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for large scale acoustic modeling," Proc. Annu. Conf. Int. Speech Commun. Assoc., no. September, pp. 338–342, 2014.

[13] M. Sundermeyer, R. Schl, and H. Ney, "LSTM Neural Networks for Language Modeling," Proc. Interspeech, 2012.

[14] T. Mikolov, S. Kombrink, L. Burget, J. Cernocky, and S. Khudanpur, "Extensions of recurrent neural network language model," ICASSP, IEEE Int. Conf. Acoust. Speech Signal Process. - Proc., pp. 5528–5531, 2011.

[15] T. Mikolov, A. Deoras, S. Kombrink, L. Burget, and J. Cernocký, "Empirical Evaluation and Combination of Advanced Language Modeling Techniques," in INTERSPEECH, 2011, pp. 605–608.

[16] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, L. Kaiser, M. Kudlur, J. Levenberg, D. Man, R. Monga, S. Moore, D. Murray, J. Shlens, B. Steiner, I. Sutskever, P. Tucker, V. Vanhoucke, V. Vasudevan, O. Vinyals, P. Warden, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems," 2015, p. 19.

[17] D. P. Kingma and J. L. Ba, "Adam: a Method for Stochastic Optimization," *Int. Conf. Learn. Represent.*, pp. 1–13, 2015.

[18] T. Mikolov, S. Kombrink, A. Deoras, L. Burget, and J. Černocký, "RNNLM --- Recurrent Neural Network Language Modeling Toolkit," *Proc. ASRU 2011*, pp. 1–4, 2011.