Interspeech 2018
2-6 September 2018, Hyderabad

Punctuation Prediction Model for Conversational Speech

Piotr Żelasko[12], *Piotr Szymański*[13], *Jan Mizgajski*[1], *Adrian Szymczak*[1], *Yishay Carmiel*[1], *Najim Dehak*[4]

[1] Intelligent Wire, USA
[2] Department of Computer Science, Electronics and Telecommunications, AGH University of Science and Technology, al. Mickiewicza 30, Kraków, Poland
[3] Department of Computational Intelligence, Wrocław University of Technology, Wybrzeże Stanisława Wyspiańskiego 27, 50-370 Wrocław, Poland
[4] Center for Language and Speech Processing, The Johns Hopkins University, Baltimore, MD, USA

pzelasko@intelligentwire.com, piotr.szymanski@spoken.com, jan.mizgajski@spoken.com, adrian.szymczak@spoken.com, ycarmiel@intelligentwire.com, ndehak3@jhu.edu

Abstract

An ASR system usually does not predict any punctuation or capitalization. Lack of punctuation causes problems in result presentation and confuses both the human reader and off-the-shelf natural language processing algorithms. To overcome these limitations, we train two variants of Deep Neural Network (DNN) sequence labelling models - a Bidirectional Long Short-Term Memory (BLSTM) and a Convolutional Neural Network (CNN), to predict the punctuation. The models are trained on the Fisher corpus which includes punctuation annotation. In our experiments, we combine time-aligned and punctuated Fisher corpus transcripts using a sequence alignment algorithm. The neural networks are trained on Common Web Crawl GloVe embedding of the words in Fisher transcripts aligned with conversation side indicators and word time infomation. The CNNs yield a better precision and BLSTMs tend to have better recall. While BLSTMs make fewer mistakes overall, the punctuation predicted by the CNN is more accurate - especially in the case of question marks. Our results constitute significant evidence that the distribution of words in time, as well as pre-trained embeddings, can be useful in the punctuation prediction task.

Index Terms: punctuation prediction, speech recognition

1. Introduction

Automatic Speech Recognition (ASR) systems are becoming widely adopted in various applications, such as voice commands, voice assistants, dictation tools or conversation transcribers. In many ASRs, a serious limitation is the lack of any punctuation or capitalization (with exception of some recent end-to-end models). This can be problematic both in the case of visual presentation of the outputs, where the non-punctuated transcripts are confusing and difficult to read, and when these transcripts are used as inputs for downstream tasks such as those in the domain of Natural Language Processing (NLP). Off-the-shelf NLP systems are usually trained on punctuated text, thus lack punctuation can cause a significant deterioration of their performance.

We are especially interested in addressing this issue in the domain of telephone conversational speech. Our application transcribes telephone calls between customers and agents, and performs their semantic annotations to find particular and specific events, as well as an intents and moods of the interlocutors. Providing punctuation became crucial for us to provide a high quality service.

Unlike many other machine learning tasks, punctuation prediction does not abound reference datasets that would enable supervised learning. In principle any punctuated text source such as blogs, news articles or Wikipedia, could be used for training a punctuation prediction model, but most of them are hardly representative of the conversational language. On the other hand, speech transcripts with proper punctuation are rather difficult to find or time-consuming to annotate. In this work, we show that the English Fisher corpus [1], which contains about 11000 distinct conversations, can be successfully used to provide data for punctuation prediction.

To leverage the fact that we are working with conversational speech, we propose to use the recognition from both sides of the conversation to predict punctuation, as well as relative timing and duration of each word, which, to the best of our knowledge, has not been used before for punctuation prediction task. Two variants of Deep Neural Network (DNN) sequence labelling models - a Bidirectional Long Short-Term Memory (BLSTM) and a Convolutional Neural Network (CNN) were trained to predict the punctuation outputs for each word in the dialogue sequence. Pre-trained GloVe [2] word embeddings were used with the intent of making the model more robust to different conversation topics than those that can be found in English Fisher corpus [1]. Both models achieve results that are on par with other work performed in this task for comparable domains.

The related research is presented in section 2. Section 3 describes our approach to data preparation as well as model architectures and the details of their training. We present and discuss the results in section 4. Finally, we conclude our work in section 5.

2. Related work

Early attempts focused on finding sentence boundaries ("dot prediction"), and for that purpose, several linguistic features were used: an n-gram language model, turn markers and parts of speech (POS) information [3]. Subsequent research employed a maximum entropy model, which predicted dots, commas and question marks based on lexical features (words, n-grams and previous predictions) and prosodic features, represented as pause tokens of a specific length [4]. It has been shown that the presence of pauses in speech can serve as an

10.21437/Interspeech.2018-1096

indicator of punctuation marks, but there is a significant variation in how different speakers use pauses [5]. Conditional Random Fields (CRF) based models were also proposed for this task [6, 7].

Recently, an LSTM model with several variants has been proposed for this task, which similarly uses words and pauses tokens as inputs [8, 9]. The authors decided not to use additional prosodic features such as F0 or phone durations due to their subpar performance in [10]. We wish to emphasize that relative word timing and duration have not been investigated by any of these works, and in principle, their fidelity should be higher than artificial, discretized pause tokens.

3. Methods

3.1. Data preparation

Unlike other telephone speech corpora the Fisher corpus [1] has properly punctuated transcripts. While the most widely used version of the Fisher transcripts (available in LDC catalogue numbers LDC2004T19 and LDC2005T19) are the *.txt* files containing time alignment, the majority of conversations also has a second transcript version in a *.txo* file, which does not have time alignment, but has rich punctuation and proper capitalization. The availability of this data provides an opportunity to utilize the information from both sides of the conversation to predict punctuation.

We represent a dialogue $\mathcal{W}$ as an ordered set $\mathcal{W} = \{w_i\}$ of words w, where each w has several properties:

- t_i is the textual representation of word w_i;
- c_i is a binary feature, representing which conversation side uttered word w_i;
- s_i is a real number, describing time offset (in seconds) at which the word w_i started;
- d_i is a real number, describing the duration (in seconds) of the word w_i;
- p_i is the punctuation symbol, which appears after word w_i.

The set is ordered on the s property of each word, i.e. the starting time. This formulation allows to elegantly represent interjections, interruptions and simultaneous speech, which are often encountered in dialogues. The p properties are only known at the training time and are being predicted during inference. With this representation in mind, we treat the punctuation prediction problem as a sequence labelling task.

To fit the Fisher data into our model definition, we need to combine information from time-annotated and punctuated transcripts. The first step is computing the forced alignment of the time-annotated transcripts to obtain word-level information about starting times and durations (s and d properties). For that purpose, we used the Kaldi ASR toolkit [11] with a LSTM-TDNN acoustic model trained with lattice-free Maximum Mutual Information (MMI) criterion [12]. In order to minimize the differences between two transcript versions we edited the Fisher data preparation script not to exclude single-word utterances and the text in parentheses, .

The next step is extraction of punctuation properties p and conversation side properties c from the punctuated transcripts. We retain blanks (no punctuation), dots, commas and question marks. Other punctuation classes were rejected (converted to blanks) due to their low frequency (e.g. exclamation marks or triple dots) or the fact that it is modeled by other properties of the representation (double dash - that marks an interruptions).

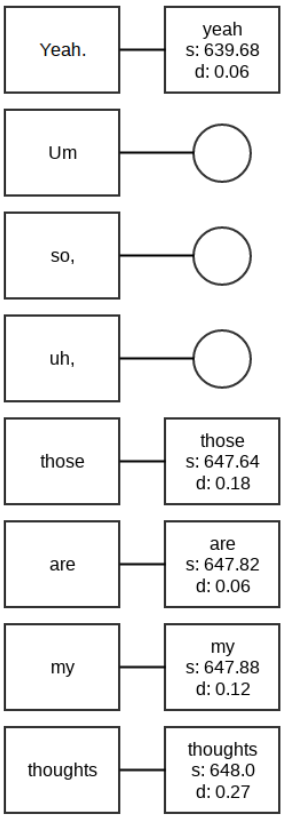

Figure 1: *An example of alignment between two word sequences in Fisher: the time-annotated and the punctuation annotated. The s stands for start time and d stands for duration, both in seconds. The circles represent a blank symbol, i.e. no match for a given word in the second sequence.*

Finally, we combine the information obtained from both sources. This task is not trivial, since both transcript versions may have slight differences. We observed that this problem could be viewed as global alignment between two symbol sequences, which can be obtained by the application of the Needleman-Wunsch algorithm [13]. The algorithm, originating in bioinformatics for DNA sequence alignment, is based on dynamic programming and is available in open-source Biopython library [14]. We compute the alignment between two transcript versions separately for each channel in each recording and remove the words which appeared in only one of the transcripts. Then, we concatenate the words from both channels into one sequence and sort it by the starting time s, which yields our dialogue representation.

Table 1: *The total count of labels for each of the punctuation classes available in our training data set.*

Class	Count	Percentage
blank	1429905	79.1%
comma	208289	11.5%
dot	148624	8.2%
question mark	22182	1.2%

Since this is a sequence labelling task, we're predicting punctuation class for each word. This results in a heavy class imbalance, as shown in table 1. We attempted to mitigate this issue by introducing sample weighting based on predicted class frequency, however, it resulted in the model being skewed towards high recall, but much lower precision for the under-represented classes, manifesting as frequent false positives.

3.2. Punctuation model

3.2.1. Features

There are several input features which we explored for our experiments. The features which we used in every experiment are word embeddings and a conversation side indicator. The word embeddings are 300-dimensional pre-trained GloVe [2] embeddings[1], trained on Common Web Crawl data. Those weights are fixed during training. We selected the embeddings for 50000 most frequent words. Then we expanded this representation by added zeroes values to embed all out of vocabulary words to save GPU memory. Increasing the vocabulary size to 100000 words did not provide any significant performance gains. Additionally, we trained our own GloVe embeddings on conversational-like data (around 525M words) gathered by the University of Washington[2] to investigate if these embeddings trained on conversational data would perform better, however, in some experiments, they resulted in either the F1 score being 0.2-0.3% lower or a lack of model convergence. We suspect this might be caused by a much smaller data quantity compared to the official GloVe embeddings.

The conversation side feature is a one-dimensional binary feature.

We used the word time information described by the interval between the start of the current word and start of the previous word, and duration of the current word, as features to the model. We provided the interval instead of absolute offset time to obtain a more normal-like distribution for this feature. Both of these features are speaker-adapted, i.e. they are standardized with regard to other words uttered by the same speaker in the same dialogue. This also means that the pauses are not modelled explicitly as word tokens - they must be inferred by the model based on the subsequent word timings.

In some experiments, we used part of speech (POS) tags predicted by SpaCy [3], although we didn't notice any significant improvement. We hypothesize that either the POS tags did not introduce any predictive information, or that the performance of the tagger was poor in the absence of punctuation (and thus sentence segmentation).

3.2.2. Architecture

We evaluated the performance of two types of models - one based on Convolutional Neural Nets (CNN), and the other based on Bidirectional Long Short-Term Memory (BLSTM) networks. The input layer is a concatenation of the features described in 3.2.1. Both models were implemented using Keras [15] with Tensorflow [16] backend.

The BLSTM model consists of four BLSTM layers, with each direction having 128 weights. This model has the advantage of seeing a large context of words during training, and possibly the whole conversation during inference.

The CNN model uses several layers of 1D convolutions, which can be interpreted as fully-connected layers processing the input in small windows. We additionally use dilated convolutions to broaden the context seen by each consecutive CNN layer. Each layer is followed by a SELU activation [17], which yielded a small improvement over batch normalization [18] with ReLU [19]. The setup which worked best for us is six 1D CNN layers, each with the filter size of 128 and padding which doesn't modify the word sequence length (i.e. *same*). The context width is equal to 3 for first five layers and equal to 20 for the last layer. The middle four layers have a dilation rate of 2.

The final layer in both CNN and BLSTM model is fully-connected and followed by a softmax activation - this layer is applied separately at each time step to retrieve punctuation prediction for a given word.

To regularize the model we apply several measures:

- a dropout layer with probability 0.5 before the softmax layer;
- 0.001 weight decay for the softmax layer weights and also for the BLSTM recurrent layers;
- we add Gaussian noise with standard deviation 0.1 to the time feature and embedding inputs, before the last softmax activation, and before SELU activations in the CNN model;
- SELU activations in the CNN model, which constrain the weights to a zero mean and unit variance distribution (which was verified by inspecting in TensorBoard).

3.3. Training

To train the models, we use a standard, categorical cross-entropy loss function and the Adam optimizer [20] with default settings proposed by the authors. The number of epochs is determined by early stopping, with two epochs patience. We divide the Fisher conversations into training, validation and test sets with proportions 8:1:1. To best utilize the GPU, we use a batch size of 256 and each sample in the batch is created by traversing the conversation in windows of 200 words.

4. Results

We present the results achieved by the CNN and BLSTM models with and without time features in table 2. Each model is evaluated with precision, recall and F1 scores for each punctuation class separately. We see that CNN models yield slightly higher precision for the punctuation classes, and BLSTM tends to have the better recall (and the inverse is true for the blank symbol). Although the BLSTM model makes fewer mistakes overall, the punctuation predicted by the CNN model is more accurate - especially in the case of question marks. The word-level time features yield minor improvement in both models, which suggests that the prosodic information carried by the relative word timing and their duration is useful in the punctuation prediction task.

For the BLSTM+T model we show the confusion matrix in figure 2. This matrix is row-normalized to better illustrate per-class mistakes, but the reader should note that due to the class imbalance (shown in table 1), this confusion matrix is almost symmetric regarding absolute numbers.

We observe several interesting types of mistakes. First of all, the blanks and commas are most frequently confounded

[1]The *glove.42B.300d.zip* embeddings, which are available at https://nlp.stanford.edu/projects/glove.

[2]The *525M_fisher_conv_web-filt+periods.gz* data set, which is available at https://ssli.ee.washington.edu/data.

[3]https://spacy.io/

Table 2: *The per-class precision, recall and F1-score (in %) achieved by the CNN and BLSTM models with pre-trained GloVe embeddings. All models used 300-dimensional word embeddings and 1-dimensional boolean conversation side features, and the +T models additionaly used two 1-dimensional time features. The ϵ symbol denotes a blank prediction.*

Model	Class	Precision	Recall	F1
CNN	ϵ	91.7	**95.5**	93.5
	.	67.7	58.6	62.8
	?	70.8	45.1	55.1
	,	68.3	58.1	62.8
CNN+T	ϵ	92.3	95.2	93.8
	.	**68.6**	63.3	65.9
	?	**72.9**	46.7	57.0
	,	**68.7**	60.3	64.2
BLSTM	ϵ	92.7	94.9	93.8
	.	66.9	63.1	64.9
	?	70.2	47.3	56.5
	,	67.9	61.8	64.7
BLSTM+T	ϵ	**93.5**	94.7	**94.1**
	.	67.9	**66.7**	**67.3**
	?	64.7	**54.6**	**59.2**
	,	68.2	**64.1**	**66.1**

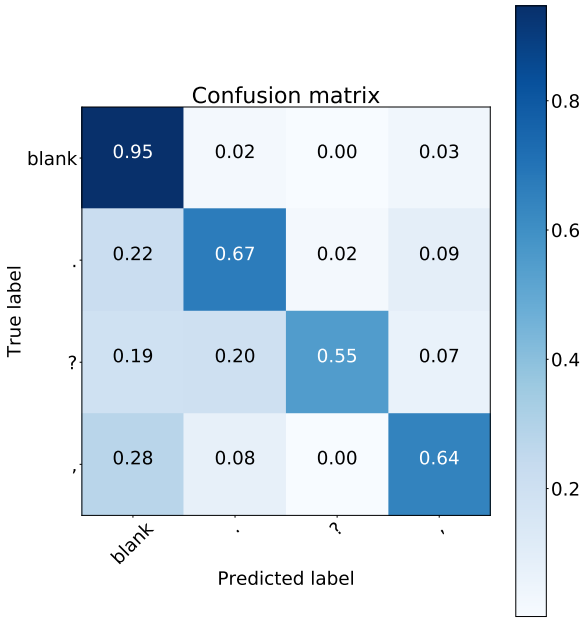

Figure 2: *Confusion matrix for the BLSTM+T model, normalized with regard to true labels (i.e. rows).*

types (around 55k false positives and 44k false negatives), which in our opinion is the least harmful type of mistake, given that the placement of commas in transcribed speech can often be arbitrary. All of the punctuation classes labels are missed about 20% of the time (i.e. blank is predicted) relatively to their occurrence count. The question mark is the most difficult class to predict and is often mistaken with the dot (about 20% of question marks), relatively rarely inserted in place of any other class. This can most likely be explained by the scarcity of labels for this class.

Below is an example part of a Fisher dialogue showcasing the predictions of the punctuation model. Note: words start with a capital letter only after a dot appears.

L: Oh, and that's west paterson. I don't know
R: Oh,
L: if
R: okay.
L: that counts.
R: Okay. Okay. Yeah, west peterson is nice. [laughter] So, i didn't even understand the ah, the topic of the day did you hear it?
L: I [noise] i heard first i heard censorship. And then i heard, ah, today's topic is something about public schools. It was i think, ah, should public schools
R: Do something about books
L: be allowed
R: kids
L: to censor
R: read?
L: certain books.

Besides the quantitative evaluation, we also performed a qualitative investigation of the predictions of both models on the ASR transcripts of calls from a different domain than Fisher. Since we do not have the golden labels for this data, this evaluation is highly subjective. We observed that the CNN model tends to yield less confusing mistakes and outputs transcripts with higher, subjective readability, which is supported by the higher precision scores obtained by this model. We suspect that this effect is amplified by the fact that the BLSTM model is more vulnerable to ASR mistakes due to the larger context size during inference.

5. Conclusions

We presented two kinds of punctuation predictions DNN models - BLSTM and CNN based - which operate on a conversation, represented as a sequence of words, and utilize word embeddings, conversation side and per-word timing information as features. We used two versions of the Fisher corpus transcripts - time-aligned and punctuated - along with sequence alignment procedure to procure the training and evaluation data. Our results constitute significant evidence that the distribution of words in time, as well as pre-trained word embeddings, can be useful in the punctuation prediction task in the domain of conversational speech. We've shown that the CNN architecture tends to achieve better precision scores, while the BLSTM variant is characterized by overall better recall and F1 measure. These models can be easily applied in a production environment to provide punctuation annotations for speech recognition system transcripts, where all of the model input features are available. For the future work, we'd like to investigate how much improvement can be gained by using prosodic features, as well as more sophisticated neural network architectures, such as models with attention [21].

6. References

[1] C. Cieri, D. Miller, and K. Walker, "The Fisher corpus: a resource for the next generations of speech-to-text." in *LREC*, vol. 4, 2004, pp. 69–71.

[2] 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)*, 2014, pp. 1532–1543.

[3] A. Stolcke and E. Shriberg, "Automatic linguistic segmentation of conversational speech," in *Spoken Language, 1996. ICSLP 96. Proceedings., Fourth International Conference on*, vol. 2. IEEE, 1996, pp. 1005–1008.

[4] J. Huang and G. Zweig, "Maximum entropy model for punctuation annotation from speech," in *Seventh International Conference on Spoken Language Processing*, 2002.

[5] M. Igras-Cybulska, B. Ziółko, P. Żelasko, and M. Witkowski, "Structure of pauses in speech in the context of speaker verification and classification of speech type," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2016, no. 1, p. 18, 2016.

[6] 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.

[7] N. Ueffing, M. Bisani, and P. Vozila, "Improved models for automatic punctuation prediction for spoken and written text." in *Interspeech*, 2013.

[8] O. Tilk and T. Alumäe, "Lstm for punctuation restoration in speech transcripts," in *Sixteenth annual conference of the international speech communication association*, 2015.

[9] O. Tilk and T. Alumae, "Bidirectional recurrent neural network with attention mechanism for punctuation restoration." in *Interspeech*, 2016, pp. 3047–3051.

[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] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz *et al.*, "The Kaldi speech recognition toolkit," in *IEEE 2011 workshop on automatic speech recognition and understanding*, no. EPFL-CONF-192584. IEEE Signal Processing Society, 2011.

[12] D. Povey, V. Peddinti, D. Galvez, P. Ghahremani, V. Manohar, X. Na, Y. Wang, and S. Khudanpur, "Purely sequence-trained neural networks for asr based on lattice-free mmi." in *Interspeech*, 2016, pp. 2751–2755.

[13] S. B. Needleman and C. D. Wunsch, "A general method applicable to the search for similarities in the amino acid sequence of two proteins," *Journal of molecular biology*, vol. 48, no. 3, pp. 443–453, 1970.

[14] P. J. A. Cock, T. Antao, J. T. Chang, B. A. Chapman, C. J. Cox, A. Dalke, I. Friedberg, T. Hamelryck, F. Kauff, B. Wilczynski, and M. J. L. de Hoon, "Biopython: freely available python tools for computational molecular biology and bioinformatics," *Bioinformatics*, vol. 25, no. 11, pp. 1422–1423, 2009. [Online]. Available: + http://dx.doi.org/10.1093/bioinformatics/btp163

[15] F. Chollet *et al.*, "Keras," https://github.com/fchollet/keras, 2015.

[16] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "TensorFlow: A system for large-scale machine learning." in *OSDI*, vol. 16, 2016, pp. 265–283.

[17] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, "Self-normalizing neural networks," in *Advances in Neural Information Processing Systems*, 2017, pp. 972–981.

[18] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*, 2015, pp. 448–456.

[19] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 807–814.

[20] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[21] W. Chan, N. Jaitly, Q. Le, and O. Vinyals, "Listen, attend and spell: A neural network for large vocabulary conversational speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. IEEE, 2016, pp. 4960–4964.