Improved Models for Automatic Punctuation Prediction for Spoken and Written Text

Nicola Ueffing, Maximilian Bisani[1]*, Paul Vozila*
Nuance Communications, Inc., Burlington, MA, United States
[1] now with Amazon.de GmbH
nicola.ueffing@nuance.com, bisani@amazon.de, paul.vozila@nuance.com

Abstract

This paper presents improved models for the automatic prediction of punctuation marks in written or spoken text. Various kinds of textual features are combined using Conditional Random Fields. These features include language model scores, token *n*-grams, sentence length, and syntactic information extracted from parse trees. The resulting models are evaluated on several different tasks, ranging from formal newspaper text to informal, dictated messages and documents, and from written text to spoken text. The newly developed models outperform a hidden-event language model by up to 26% relative in F-score.

Evaluation of punctuation prediction on erroneous ASR output as well as evaluation against multiple references is not straightforward. We propose modifications of existing evaluation methods to handle these cases.

Index Terms: auto-punctuation, punctuation prediction, punctuation insertion, punctuation restoration

1. Introduction

The output of automatic speech recognition (ASR) systems typically does not contain punctuation marks. However, the presence of punctuation marks can largely increase readability of the generated text and help downstream language processing applications such as natural language understanding or machine translation. In dictation systems, the user experience can be improved if punctuation marks get automatically inserted and do not need to be dictated.

In the work presented here, we focus on the prediction of three different punctuation marks: comma, period, and question mark. These are by far the most frequent punctuation marks, so predicting these correctly will have a significant impact on the readability of the generated text. We developed auto-punctuation models which combine various kind of textual features using Conditional Random Fields (CRFs). While many existing approaches use both lexical and prosodic features, we focus on textual features in this work. Our main interest is in dictation applications, and previous in-house studies have shown that prosodic features are not clearly helpful in this setting. One possible explanation is that users tend to make pauses in 'unnatural' places when dictating text.

The rest of the paper is organized as follows: Related work is summarized in Section 2. The auto-punctuation models will be described in detail in Section 3. Section 4 gives an overview of the tasks and corpora, evaluation metrics, and experimental results. Finally, Section 5 concludes the paper.

2. Related work

[1] proposed a hidden-event modeling approach to boundary detection in speech recognition. Boundaries and punctuation marks are treated as hidden inter-word events. An *n*-gram language model (LM) is trained on punctuated texts and then used to predict punctuation marks between words. [2] applied a similar approach to the joint restoration of punctuation and capitalization. Recently, Conditional Random Fields (CRFs) have been shown to be well-suited for auto-punctuation. Punctuation prediction is modeled as a labeling task, with labels like "comma" or "none" assigned to the words, denoting the punctuation mark inserted after the word. [3] investigates CRFs using only token features. Factorial CRFs, modeling sentence type and punctuation marks simultaneously, are found to outperform linear CRFs on data from the travel domain. In [4], the authors also use factorial CRFs, combining token features with LM scores and prosodic features. They analyze the impact of the different feature types on a broadcast news corpus. [5] and [6] develop auto-punctuation models for broadcast news and broadcast conversations in different languages. Both prosodic and textual features are combined and tested under challenging ASR conditions. [7] proposes the integration of dynamic features, i.e. features depending on previous labeling decisions, for prosodic break prediction in text-to-speech systems. The approach is easily applicable to punctuation prediction as well. Some research has focused on predicting commas in a scenario where sentence boundaries are given [8][9]. While this is not the case in a real-world scenario, we found the proposed syntactic features interesting and investigated similar ones. Finally, [10] treats auto-punctuation as a machine translation task within a speech translation system. The authors publish results on the TED data that we also used for testing.

3. Punctuation prediction models

3.1. Baseline

The baseline model which we employed in our experiments uses a hidden-event LM as described in Section 2. At each position in the text, all different punctuation marks as well as "none" are hypothesized. Then, the most likely sequence according to the LM trained on punctuated texts is chosen. The LMs are 4-gram models including a class-based component.

The LMs for the different tasks described in Section 4.1 have been trained on matching training data. On the TED task, we added in-house training data to the TED training data to improve performance of the baseline model. On WSJ and TED, we experimented with higher-order word-*n*-grams too, but did not see a significant gain in performance. This might be due to sparsity of the long *n*-grams in the training data.

3.2. Features

3.2.1. Language model

This feature computes the difference in LM score with and without the insertion of punctuation, taking into account the scores of all affected *n*-grams. For each position, the LM

probability of the inserted punctuation mark given the surrounding words is divided by the LM probability of these context words without inserted punctuation. The log of this ratio yields a separate feature for each of the punctuation marks being predicted. The LM is the same model as used for the hidden-event baseline. Note that this LM score is computed offline. Surrounding punctuation decisions are therefore not reflected in the score.

3.2.2. Tokens

In order to capture contextual information, we use a binary feature over tokens surrounding the hypothesized punctuation mark. We considered unigrams in a 5-word window, and bigrams and trigrams in a 3-word window. Using a larger window for trigrams strongly increases model size, while the change in performance is negligible.

We experimented with vocabulary pruning as described in [4], but did not observe a significant gain. While this is an effective method to keep the size of the resulting model manageable, model performance decreased in some settings. We investigated another modification to these features that proved helpful when working on ASR output: replacing the full-form token by suffixes of length 4. This also reduces model size and increases robustness w.r.t. ASR errors.

3.2.3. Dynamic sentence length features

The features which are typically used in auto-punctuation do not take previous labeling decisions into account. To overcome this, we introduced one feature determining the current sentence length, i.e. the distance from the last predicted sentence-end punctuation mark. This means that the decoding has to keep track of decisions about previous sentence ends, enlarging the search space significantly. See [7] for a detailed description. As additional sentence-based feature we implemented a sentence count or sentence penalty, i.e. a constant value for each predicted sentence-end punctuation mark which is automatically learned in training. We introduced this because we found that the model often predicts too few sentence boundaries. The sentence count features helped overcome this problem.

3.2.4. Syntax-based

Syntactic information can be useful for predicting punctuation marks, especially when dealing with intrasentential punctuation, such as commas. We therefore extracted features from the constituents of a parse tree. This was inspired by the work on comma restoration reported in [8][9]. In contrast to the setup there, we are interested in ASR applications such as dictation where the sentence boundaries are unknown. Thus, we follow a 2-stage approach. In the first step, sentence boundaries are predicted based on the CRF model using LM scores and token features. The text is then segmented according to the predicted sentence boundaries, and these segments are parsed using the Berkeley parser [11]. The parser has been trained on data without punctuation. In order to avoid mismatch between training and testing data, we used automatically predicted sentence boundaries on the training data as well.

From the parse tree, we extract information about the constituents surrounding the hypothesized punctuation mark. We use the identities of the highest starting and ending constituents directly as features. The resulting feature values are pairs of constituents, e.g. “NP/PP” or “DT/JJ”.

We also investigated other syntax-based features, some of which have been proposed in [8][9], such as the number of constituents starting or ending in a position, POS, POS-based LM scores, and the identity of surrounding chunks. However, none of these yielded improvements in our experiments.

3.2.5. The classifier

We trained the CRFs using the Wapiti toolkit [12]. We chose this toolkit because it can handle large numbers of features efficiently, which was necessary when working on a large-vocabulary ASR task. Since Wapiti can handle only binary features, we binned and discretized the LM scores and sentence-length features.

3.2.6. 1-stage vs. 2-stage approach

We compared two different approaches proposed in auto-punctuation literature. The 2-stage approach determines the position of punctuation mark and the type of punctuation mark separately, whereas the 1-stage approach determines both at the same time. None of the approaches proved to be clearly superior. While the 1-stage approach performs best on the WSJ and TED data, the 2-stage approach performs better on the dictation task.

4. Experimental evaluation

4.1. Corpora

We evaluated the punctuation prediction models on a variety of different tasks, including both written and spoken text. The language is English in all cases. However, the models can be easily applied to text in other languages. The only features making use of additional resources are the parse-based features described in Section 3.2.4. As we will see in Section 4.4, the largest gain in performance is achieved using the features that do not require a parser, making the models applicable to languages other than English as well.

In the experiments presented here, comma, period, and question mark are automatically inserted. We preprocessed the data and mapped some punctuation marks to these three, e.g. exclamation marks are mapped onto periods. Table 1 presents corpus statistics for the test sets of the three different tasks.

4.1.1. WSJ

The Wall Street Journal (WSJ) corpus consists of newspaper text. The style is formal, and punctuation is used in a consistent manner. We used the standard ACL split: sections 2-21 for training, sections 22 and 24 as development corpora, section 23 for testing. The texts are already segmented into sentences. If this segmentation is used, punctuation prediction is very easy, especially for periods. Almost every segment boundary corresponds to a period. In order to make the task more challenging and to simulate a dictation setting, we completely ignore the given sentence segmentation. The text is passed to the auto-punctuation module as a long series of tokens.

The motivation behind working on WSJ data was our interest in syntactic features. There are good automatic parsers as well as manual parse trees available for these data.

4.1.2. TED

The second task on which we tested our models are transcriptions of presentations from the TED series (http://www.ted.com). Both the audio files and manual transcriptions containing punctuation are available from the website. The speech translation evaluation campaign of IWSLT2011 [13] used these data and additionally provided ASR output for the development and test sets. WER on the development set is 18.0%. Some participants of the evaluation published auto-punctuation results [10] that we compared against. Again, we did not use the sentence segmentation provided with the data.

4.1.3. In-house dictation corpus

We carried out experiments on an in-house dictation corpus comprising around 26k tokens. These dictations cover a wide range of domains and documents of differing levels of formality, comprising e.g., reports, private and business email. To create a test set, we selected dictated utterances that had been transcribed before. The transcriptions were manually annotated with punctuation marks by three different annotators. All of them are qualified native speakers. The audio files were not presented to the annotators; they had only the text as input.

The annotators inserted six different punctuation marks: Period, comma, question mark, exclamation mark, colon, and semicolon. However, we found that a) the use of exclamation mark, colon and semicolon is largely a matter of personal style and very inconsistent among different annotators, and b) these three punctuation marks occur quite infrequently. We therefore boiled the set of punctuation marks to be predicted down to a basic set: comma, period, and question mark. The remaining three punctuation marks inserted by the annotators were mapped onto the basic ones (exclamation mark to period, the others to comma). The inter-annotator agreement, measured using Cohen's kappa, for this basic set is κ=0.69.

The last row in Table 1 shows how largely the numbers of punctuation marks inserted by the annotators differ; the number of commas, for example, ranges from 1,393 to 1,975.

Table 1: *Corpus statistics of the evaluation sets*

Corpus	Tokens	Comma	Period	Question
WSJ	51,023	3,056	2,345	25
TED, manual	17,207	1,096	925	84
TED, ASR	17,344	1,084	904	82
Dictation	25,855	1,393-1,975	1,147-1,345	100 – 121

4.2. Evaluation metrics

We measured the performance of the automatic punctuation prediction models using precision *P* (=percentage of inserted punctuation marks which are correct), recall *R* (=percentage of reference punctuation marks which have been inserted by the model), and F1-score *F1* (=harmonic mean of *P* and *R*). These measures have been used in many other publications on punctuation prediction [3][4][8]. Additionally, we evaluate how often the model determines the correct *position* of a punctuation mark, but not necessarily the correct *type*, and denote this by *F1-pos.*

When evaluating automatic punctuation methods on ASR output, we determine the "reference" punctuation by mapping the punctuation marks from the correct transcript into the ASR output based on Levenshtein alignment. This approach has also been used in [5]. It allows for the generation of "reference" punctuation even for erroneous ASR output. However, this does not lead to satisfactory results for utterances containing many ASR errors. Consider the following example from the TED data:

Manual transcription	leave that up to us . clean coal , you've heard a lot about it . so let's take a tour […]
ASR output	the their ones think all heard a lot about it I think tour […]

Since it does not make much sense to evaluate punctuation prediction on this kind of output, we propose a modification of the evaluation method for ASR output. We restrict the evaluation to those punctuation marks whose context words to the left and the right have been recognized correctly. This covers about 2/3 of all punctuation marks in the TED test data and filters output regions with many ASR errors quite reliably.

On the in-house dictation corpus, we have three different punctuation references available. For each segment, we use the reference with minimal distance for all punctuation decisions. There are two possible ways of defining these segments for evaluation: documents and utterances. The documents are relatively long segments, with 162 words on average; whereas the utterances are very short (4 words on average) and often comprise only partial sentences. That is, evaluation at the document-level provides us with a lower bound for model performance, and utterance-level evaluation with an upper bound. The truth lies somewhere in the middle.

4.3. Overview of features

Table 2 shows which of the features described above proved useful on the different tasks in our experimental evaluation. There are two groups of features which improved the model consistently across all tasks: the LM scores and the token *n*-grams. The usefulness of the other types of features strongly depends on the task. The results will be discussed in detail in the following subsections.

Table 2: *Features in the best auto-punctuation models*

Feature	Dictation Task	TED		WSJ
		manual	ASR	
LM score	x	x	x	x
Full tokens	x			x
… suffix		x	x	
Sent. length		x	x	x
Sent. penalty			x	
Constituents				x

4.4. Results

Table 3 shows experimental results for the automatic punctuation models on the different tasks described above. The features used in the "best model" are detailed in Table 2. On the WSJ data, the CRF models outperform the hidden-event baseline significantly. The model combining language model scores and token *n*-grams already achieves a 22% relative improvement in F-score, with a significant increase for

both precision and recall. This can be further improved by adding more complex features into the model. The combined gain from the syntactic and the sentence-length features is 2.1 points in F1-score. Most of this gain, 1.8 points, comes from the syntactic features (This result is not shown in the table). On the clean and well-structured WSJ data, the automatic parses are of high quality and thus provide valuable information for punctuation prediction. However, when working with texts that are less formal, and especially when working with ASR output, the constituent features do not improve auto-punctuation. On TED ASR data, the constituent features cause a drop of 1.0 points F-score (This result is not shown in the table). This is probably due to mismatch between the parser's training data and the application data.

Table 3: *Experimental results for various punctuation prediction models on the different tasks*

Task / Model	P	R	F1	F1-pos
WSJ test set				
Hidden-event LM	59.9	45.8	51.9	68.5
CRF, token + LM	68.4	59.1	63.4 (+22%)	78.8
CRF, best model	69.5	61.9	65.5 (+26%)	80.2
TED, manual transcriptions, dev set				
Hidden-event LM	58.1	41.6	48.5	68.6
CRF, token + LM	49.3	57.1	52.9 (+9%)	75.5
CRF, best model	49.8	58.0	53.5 (+10%)	75.8
TED, ASR output, dev set				
Hidden-event LM	53.8	32.7	40.7	57.4
CRF, token + LM	46.1	54.5	49.9 (+23%)	63.2
CRF, best model	47.8	54.8	51.0 (+25%)	64.0
Dictation corpus, evaluation on utterance level				
Hidden-event LM	73.5	57.3	64.4	78.8
CRF, token + LM	59.1	63.1	61.0 (-5%)	81.5
CRF, best model	65.0	68.8	66.8 (+4%)	83.0
Best human	84.1	77.5	80.7	86.0
Dictation corpus, evaluation on document level				
Hidden-event LM	56.8	43.1	49.0	65.3
CRF, token + LM	47.5	50.9	49.1 (+0.3%)	70.5
CRF, best model	52.2	55.3	53.7 (+10%)	73.2
Best human	71.9	65.4	68.5	78.0

On the TED data, we evaluated the punctuation prediction models both on correct manual transcriptions and on ASR output. The CRF-based models outperform the baseline by 10% relative on the manual transcriptions. Most of the gain (4.4 points F-score) results from the LM scores and token *n*-gram features, and an additional 0.6 points absolute from the sentence length feature. On ASR output, the tendency is even stronger: The CRF-based models improve by up to 25% relative over the hidden-event baseline. The reason is that recall increases by over 50%. So obviously, the performance of the baseline model suffers from recognition errors, whereas the CRF-based models are more robust and still correctly identify the positions where punctuation marks have to be inserted. We observed a small gain from using suffix features instead of full-form tokens; this seems to make the model more robust w.r.t. ASR errors.

On the in-house dictation corpus, the gain from using CRFs is smaller. The CRFs using LM scores and token features do not outperform the baseline. However, using the CRFs in a 2-stage approach yields a gain between 4% and 10% relative over the baseline. We scored each human annotator against the two others and included the highest score among the three as "best human" in Table 3.

Note that all investigated models achieve high "F1-pos" scores for determining the position of the punctuation marks. On the dictation task, performance of the best model is close to human performance (F-score 83.0 vs. 86.0 on utterance level).

Table 4 presents a more detailed evaluation for some of the experiments described above. The performance of the punctuation prediction models is broken down for the different punctuation marks. We see that in most cases, the CRF-based models outperform the hidden-event model consistently for all three punctuation marks. The only exception is the dictation corpus, where the simpler CRF model performs worse for commas.

All models perform best at predicting periods. The baseline model achieves an F-score of 50-60 for periods, which is consistently improved upon by the CRF-based models.

The largest gain in auto-punctuation quality is obtained for question marks which the baseline model often fails to insert. Note that even the best models achieve relatively low F-scores for question marks. This is probably due to the fact that they occur very infrequently in the training (as well as the test) data, so that the model does not have much data to learn from.

Table 4: *Punctuation prediction performance for different punctuation marks (P / R / F1).*

Model	Comma	Period	Question
WSJ test set			
Hidden-event LM	63 / 47 / 54	57 / 45 / 50	0 / 0
CRF, token + LM	68 / 49 / 57	62 / 50 / 55	0 / 0
CRF, best model	72 / 59 / 65	67 / 67 / 67	100 / 12 / 21
TED, manual transcriptions, dev set			
Hidden-event LM	53 / 30 / 38	**62** / 59 / 60	**70** / 8 / 15
CRF, token + LM	44 / 46 / 45	55 / 72 / 62	53 / 32 / 40
CRF, best model	45 / 47 / 46	54 / 72 / 62	53 / 33 / 41
Dictation corpus, eval on document level			
Hidden-event LM	54 / 37 / 44	58 / 52 / 55	37 / 2 / 5
CRF, token + LM	49 / 31 / 38	47 / 76 / 58	45 / 9 / 15
CRF, best model	48 / 46 / 47	54 / 68 / 61	61 / 12 / 20

5. Conclusions

We presented models for automatic punctuation prediction which combine a variety of different textual features. To the best of our knowledge, this is the first work which extends a state-of-the-art model with syntactic features and with dynamic features based on the last predicted sentence boundary. The proposed models have been extensively evaluated on three tasks which differ largely in domain and style. This showed that the choice of features in the model depends on the kind of text. The token *n*-grams and LM scores, which have become state of the art recently [4][5], perform well across all types of text. The syntactic features convey useful information if the input text is formal and well-structured. Under noisier conditions, however, they are not helpful anymore. On ASR output, replacing the token *n*-gram features with suffix features increases the model's robustness against recognition errors.

Additionally, we presented modifications of the widely used evaluation metrics precision, recall and F-score to account for ASR errors and for the availability of multiple references.

6. References

[1] Stolcke, A., Shriberg, E., Bater, R., Ostendorf, M., Hakkani, D., Plauche, M., Lu, Y. "Automatic detection of sentence boundaries and disfluencies based on recognized words", ICSLP 1998.

[2] Gravano, A., Jansche, M., Bacchiani, M., "Restoring punctuation and capitalization in transcribed speech," ICASSP 2009.

[3] Lu, W., Ng, H.T., "Better Punctuation Prediction with Dynamic Conditional Random Fields", EMNLP 2010.

[4] Wang, X., Ng, H.T., Sim, K.C., "Dynamic Conditional Random Fields for Joint Sentence Boundary and Punctuation Prediction", Interspeech 2012.

[5] Kolář, J., Lamel, L. "Development and Evaluation of Automatic Punctuation for French and English Speech-to-Text", Interspeech 2012.

[6] Batista, F., Moniz, H., Trancoso, I., Mamede, N. J., "Bilingual experiments on automatic recovery of capitalization and punctuation of automatic speech transcripts," IEEE Transactions on Audio, Speech and Language Processing, Special Issue on New Frontiers in Rich Transcription, 2012.

[7] Luan, J., He, B., Xia, H., Wang, L. Daniele, B., Zhao, S., "Expand CRF to Model Long Distance Dependencies in Prosodic Break Prediction", Interspeech 2012.

[8] Shieber, S.M., Tao, X., "Comma Restoration Using Constituency Information", HLT-NAACL 2003.

[9] Favre, B., Hakkani-Tur, D., Shriberg, E., "Syntactically-informed Models for Comma Prediction," ICASSP 2009.

[10] Peitz, S., Freitag, M., Mauser, A., Ney, H., "Modeling Punctuation Prediction as Machine Translation", IWSLT 2011.

[11] Slav Petrov, Dan Klein, "Improved Inference for Unlexicalized Parsing", HLT-NAACL 2007.

[12] Lavergne, T., Cappé, O., Yvon, F. “Practical very large scale CRFs”, Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, 2010.

[13] Marcello Federico, Luisa Bentivogli, Michael Paul, Sebastian Stüker, “Overview of the IWSLT 2011 Evaluation Campaign”, Proceedings of the International Workshop on Spoken Language Translation, 2011.