interspeech 2008 incorporating SST 08

Decoding-Time Prediction of Non-Verbalized Punctuation

Anoop Deoras[1] *and Jürgen Fritsch*[2]

[1]Center for Language and Speech Processing, Johns Hopkins University
[2]M*Modal, Pittsburgh USA

adeoras@jhu.edu, juergen.fritsch@mmodal.com

Abstract

This paper presents novel methods that integrate lexical prediction of non-verbalized punctuations with Viterbi decoding for Large Vocabulary Conversational Speech Recognition (LVCSR) in a single pass. We describe two different approaches - one based on a modified finite state machine representation of language models and one based on an extension of an LVCSR decoder. We discuss advantages over traditional punctuation prediction approaches based on post-processing of recognition hypotheses, including experimental evaluation of the proposed approach using a state-of-the-art LVCSR decoder. Experiments were performed on a medical documentation corpus and results demonstrate that the proposed methods yield improved punctuation prediction accuracy while at the same time reducing system complexity and memory requirements.

Index Terms: Punctuation Prediction, Sentence Boundary Detection, Speech Recognition.

1. Introduction

While cooperative speakers using a dictation system would typically learn to verbalize punctuation symbols such as periods and commas as part of their dictations in order to improve the segmentation and completeness of automatically transcribed text, most punctuation symbols are not normally verbalized in natural - spontaneous and/or conversational - speech. Human listeners typically have no problem identifying and inserting non verbalized punctuation (at least periods), but standard speech recognition systems attempt to transcribe speech literally and thus do not generally identify non-verbalized punctuations.

Consistent and accurate prediction and insertion of all punctuation (verbalized as well as non-verbalized) in transcripts of conversational speech is critical to many tasks involving automatic speech recognition. In particular, accurate phrase and sentence segmentation is needed for speech-to-speech translation, parsing and rendering of transcribed speech into written language.

Traditionally, most approaches to predicting non-verbalized punctuation are applied in a post-processing step, *after* the completion of speech decoding, either using the generated best-scoring hypothesis or the word lattice as input, sometimes including acoustic and/or prosodic features. Stolcke et al. [1] have tried to make use of prosodic cues extracted from the spoken data, for instance, by extracting pause durations which may indicate sentence boundaries, thus providing evidence for non-verbalized periods. Hirschberg and Nakatani [2] also made use of various acoustic/prosodic features in order to carry out topic and phrase boundary identification. As against this, Gotoh and Renals [3] have tried to identify sentence boundaries in broadcast speech using statistical finite state models derived from news transcripts and speech recognizer outputs. They claim that their work is a step towards the production of structured speech transcriptions which may include punctuation or content annotation. Ramabhadran et al. [4] rely exclusively on prosodic cues for predicting non-verbalized punctuation as part of a transcription system for parliamentary speeches. Common to all of these approaches is the need for separate punctuation prediction models that are applied in a second pass after the initial decoding of speech recordings in a first pass over the data.

In the area of speech-based medical documentation, physicians are used to conversationally document patient encounters and performed medical procedures, assuming that a human medical transcriptionist will listen to their dictations, correcting non-grammatical and incomplete phrases, as well as inserting non-verbalized punctuation symbols. Due to the repetitive nature of their dictations, physicians often times speak comparatively fast and without discernible pauses or other prosodic cues in place of non-verbalized punctuation, making traditional approaches rather ineffective.

In the absence of acoustic/prosodic cues for predicting non-verbalized punctuation, we focus on lexical cues in this paper and outline an approach for incorporating prediction of such non-verbalized punctuation during LVCSR decoding, rather than in a separate post-processing step. To that end, we use a punctuation-aware statistical language model and modify M*Modal's LVCSR decoder [5] such that it will allow us to predict non-verbalized punctuation without necessarily requiring acoustic evidence for each predicted punctuation. Punctuation-aware statistical language models are trained from written-language (including all relevant punctuation) rather than spoken-language representations (including only punctuation that was actually spoken) of medical reports. Our approach has two main benefits: First, rather than using a spoken-language model for LVCSR decoding and a separate written-language model for punctuation prediction, it only requires to train and maintain a single model for both of these tasks. Second, combining the tasks of decoding speech into a sequence of words and at the same time inserting non-verbalized punctuation should result in higher overall accuracy compared to traditional approaches - both in terms of raw word accuracy as well as punctuation accuracy. This is because traditional lexical approaches mostly work off of first-pass recognition hypothesis that contain a certain number of recognition errors, while the punctuation prediction models are typically trained on error-free written-language texts.

The rest of the paper will proceed as follows. After reviewing traditional punctuation prediction, which will be our baseline model, as a post-processor to an automatic speech recognition system in section 2, we will briefly describe the acoustic modeling of non-verbalized punctuations using single-state HMMs in section 3. In section 4, we will discuss a finite state machine model that allows to use one and the same language

Accepted after peer review of full paper

model for joint speech decoding and punctuation prediction. In section 5, we continue to develop the same idea, but from a different perspective, concentrating on extensions to lexical prefix tree expansion in the decoder. In section 6, we provide experimental results for the different methods on conversational medical data, and finally conclude our work with section 7.

2. Punctuation Prediction as a Post-Processor

In this setting, a written-form statistical language model was built in addition to a spoken-form statistical language model. The former was trained on fully punctuated data and served as the language model for punctuation prediction with a post-processor. The latter was trained on partially punctuated data (excluding non-verbalized punctuation) and served as the language model for automatic speech recognition. The transcribed output generated by the LVCSR system, which lacks most of the non-verbalized punctuation, is then fed into the punctuation post-processor. The post-processor implements a variant of Beeferman et.al.'s light weight method [6] for prediction and insertion of intra-sentence punctuations. The input to the post-processor is a sequence $W = w_1 w_2 \dots w_n$. This sequence misses non-verbalized punctuations. We hypothesize optional non-verbalized punctuations between any two consecutive tokens w_i and w_{i+1} in this sequence by turning it into a directed acyclic FSM, as shown in Fig. 1

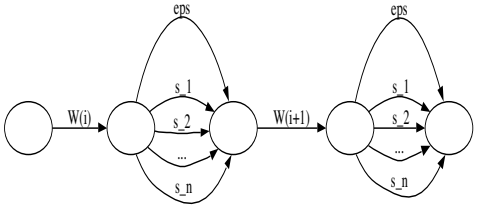

Figure 1: Figure shows a small part of a typical FSM constructed from the decoder hypothesis by inserting optional punctuation arcs between each successive hypothesis word.

with $s_1 \dots s_n$ denoting the different punctuation symbols to be predicted, and eps denoting a non-emitting arc. Each arc may have a certain weight attached to it for modeling the prior probability of each punctuation symbol and the epsilon transition.

We then compute the maximum likelihood path through this graph with respect to the written-form language model. The output of the post-processor is the modified sequence $Y = y_1 \dots y_m$, with $m >= n$ in general since it includes predicted non-verbalized punctuations. This simple model works remarkably well and Beeferman et.al. claim that it works better than approaches based on detecting prosodic cues from the spoken data [6].

However, the model suffers from 2 disadvantages: 1. A separate language model needs to be built and loaded to carry out punctuation post-processing. 2. The post-processor is applied to (potentially erroneous) speech recognition hypotheses using a written-form language model that was trained on reference transcriptions. The accuracy of punctuation prediction suffers from this mismatch and the presence of recognition errors in the input sequence.

3. Acoustic Modeling of Non-Verbalized Punctuation

In a first attempt at unifying the punctuation prediction and decoding models, we removed the punctuation prediction post-processor entirely and used the written-form instead of the spoken-form language model for LVCSR decoding. To account for non-verbalized punctuations predicted by the written-form language model, we allowed each of the punctuation symbols to be modeled acoustically as a short pause, in addition to their standard pronunciations.

We wanted to determine if modeling non-verbalized punctuations via short pauses would allow the LVCSR decoder to insert these punctuations, even in the absence of clear pauses in the acoustic signal. To allow for such cases, we had to use a model that allows for the shortest possible pause, in our case a single frame of speech of length $8ms$, by using a single state Hidden Markov Model.

Not surprisingly, initial experiments with this setup resulted in many false insertions of punctuations. Conclusions were apparent that in the absence of acoustic/prosodic cues for non-verbalized punctuations, one has to allow for prediction of punctuation symbols without any acoustic evidence, relying primarily on language modeling information.

4. Auto-Punctuation FSM

In this approach, the spoken-form language models were replaced with fully punctuation-aware written-form language models for decoding and the punctuation post-processor was removed. Our LVCSR decoder uses a finite state machine abstraction layer as an interface to the language modeling component [5], thus allowing to decode with arbitrary language models, including (weighted) finite state grammars and statistical n-gram models, as long as these models can be represented as finite state graphs. In order to simultaneously detect the most likely insertions and deletions of punctuations (verbalized and non-verbalized) while decoding the input audio, we have modified the finite state machine abstraction layer, initially for n-gram models.

Let us use the example of a trigram language model (the approach can be generalized to arbitrary order n-gram models, but for the sake of argument, we will use this simple example). During Viterbi beam search, the LVCSR decoder expands hypothesized partial sequences of words $w_1 \dots w_{i-1}$ by likely following words w_i. To do that, it is querying the language model for the probability $p(w_i|w_{i-1}, w_{i-2})$ for word w_i preceeded by the 2-word history w_{i-1}, w_{i-2}. For every such hypothesized word, the decoder also queries the acoustic model for the likelihood of the word w_i given the acoustic speech signal, and combines it with the language model probability for the total word score. In order to predict non-verbalized punctuations in the absence of acoustic evidence, the decoder needs the ability to hypothesize tokens without consuming input frames.

To this end, we expand the finite state machine for the inclusion of non-verbalized punctuations. Fig. 2 shows the FSM abstraction for one state for a trigram language model. Even if we would attempt to predict only the two most common non-verbalized punctuations in western languages (periods and commas), we face a prohibitive increase in the number of arcs in the FSM. Hence it becomes essential to prune the finite state machine transitions. Simply pruning the locally improbable paths was found to work well in practice. In this greedy approach, we retain only the single most probable arc for any given state

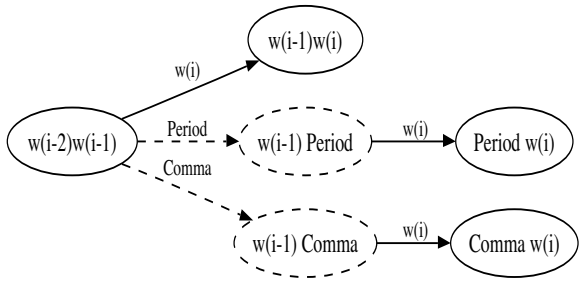

Figure 2: Figure shows the FSM abstraction for one state for a trigram language model. The state to the left denotes the history state representing $w_{i-2}w_{i-1}$. Three arcs emanate from this state for each vocabulary word w_i (only the arcs corresponding to a single word w_i are shown). The middle arc passes through the non-verbalized punctuation 'Period'. The lower most arc passes through 'Comma'. The top most arc directly passes through the word w_i without any (non-verbalized) punctuation.

and vocabulary token. To illustrate this idea in detail, consider some history state $H = W_{i-2}W_{i-1}$, out of which 3 arcs emanate. We compute the likelihood of each arc by computing the contributing language model scores and select the path which yields the maximum (local) likelihood and discard the others:

$$\begin{aligned} P = \max(&P(w_i|w_{i-2}w_{i-1}) \\ &, P('period'|w_{i-2}w_{i-1}).P(w_i|w_{i-1}'period') \\ &, P('comma'|w_{i-2}w_{i-1}).P(w_i|w_{i-1}'comma')) \end{aligned}$$

While the disadvantage to this method is that, because we do not propagate all the paths (three in the case of two punctuation) emanating from a state, we may end up pruning the path which locally seems improbable, but globally could be a part of a more probable path. However, in the case of trigram models, we did not observe a deterioration of punctuation prediction performance with this type of pruning.

If our greedy pruning accepts a local path containing one of the non-verbalized punctuations, P, then the next state becomes (P, W_i), after consuming a spoken-word form W_i. In the setting of this FSM abstraction, if we treat arcs passing through non-verbalized punctuations as intermediate arcs, then the arc label visible to the decoder will be W_i instead of P.

The information whether a non-verbalized punctuation was hypothesized can be obtained by inspecting the history states (see Fig. 3). Whenever the word history encoded by an FSM state does not match the words on the preceding arcs, the presence of a corresponding non-verbalized punctuation is indicated. Once this is detected, a non-verbalized punctuation needs to be inserted between the labels of the preceding two arcs of the relevant FSM state.

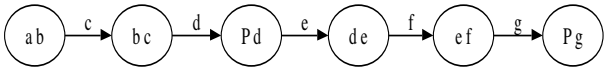

Figure 3: Figure shows a small segment of a hypothesized word network, where non-verbalized punctuations have been recognized. The alphabets $a, \ldots, g$ represent the words $\in \mathcal{V}$. The alphabet P represents the non-verbalized punctuation.

In the example illustrated in Fig. 3, the final hypothesized word sequence would be: $c, \mathbf{P}, d, e, f, \mathbf{P}, g$.

5. Auto-Punctuation Decoder

While representing non-verbalized punctuations explicitly via alternative paths in the FSM for decoding is effective, it creates a certain overhead due to the need to precompute and compare probabilities of competing transitions for each FSM state, even though many of these will never be considered by the decoder. We therefore investigated an alternative approach with significantly reduced complexity and computational overhead.

The basic idea is to integrate prediction of non-verbalized punctuations directly into the word expansion step of our single-prefix-tree LVCSR decoder [5]. This decoder uses a priority heap to represent alternative linguistic theories in each node of the prefix tree. In contrast to approaches based on tree copies, the heap approach with a single prefix tree is more dynamic, more scalable and allows us to employ different search and pruning techniques by effectively controlling hypothesis merging.

When decoding input speech using the single-prefix-tree and a Viterbi beam search, we expand a result graph with new word transitions at each frame by inspecting the heaps of all leaf and stub nodes in the prefix tree. The most probable entries across all the heaps of all these nodes are expanded by adding the corresponding word to the result graph and re-entering the prefix tree at the appropriate roots and stubs using the FSM states resulting from the word transition. Our extension for auto-punctuation is simple: each entry selected for expansion in each frame is expanded not just by taking the corresponding word transition, but also by taking additional, non-verbalized punctuation transitions, before re-entering the prefix tree (see Fig. 4).

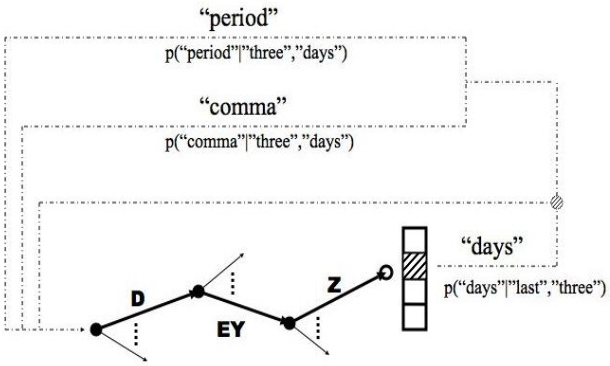

Figure 4: Auto-punctuation using modified single-prefix-tree decoder: Shown is the normal plus two newly added non-verbalized punctuation transitions for recognized word "days" in trigram context "next three days".

All transitions for a given word share a common acoustic score and phonetic context since non-verbalized punctuations only contribute to the language model score. The transitions with additional non-verbalized punctuations compete with all other transitions in a given frame. Thus, we can rely on the various existing pruning strategies for weeding out unlikely paths, including those involving non-verbalized punctuations. In practice, an auto-punctuating decoder can be implemented with almost no computational overhead by adding a threshold pruning

step such that an auto-punctuation transition is only followed if the probability of the punctuation given the current word history is above an empirically determined minimum probability.

6. Experiments and Results

In this section, we compare the two approaches presented in sections 4 and 5, with respect to the baseline model of an auto-punctuation post-processor, in terms of speed and recognition accuracy. The experiments were performed using data from a medical transcription domain serviced by M*Modal.

The training data consists of documents from an Internal Medicine corpus with a total of 2.9 billion word tokens. The test data consists of 120 held-out documents from the same corpus consisting of $109k$ word tokens. The test data comprises speech from 25 different speakers.

We have used 39 dimensional MFCC acoustic feature vectors and trigram language models. We typically interpolate multiple language models such as speaker-independent, speaker-dependent and domain-dependent language models to form the language model used for decoding. The size of the recognition vocabulary is $57k$ words.

The baseline consists of performing auto-punctuation (AP) as a post-processor to the decoder. The language models used in the decoder are derived from spoken-form text, while the language models used in the AP post-processor are derived from written-form text. While we interpolate language models for decoding, only speaker-independent language models were used for the AP post-processor.

For the experiments with AP FSM and AP Decoding, we have removed the AP post-processor and replaced the language model hierarchy in the decoder with an equivalent hierarchy of language models derived from written-form text.

Table 1 compares the performance of the AP FSM and AP Decoder approaches with respect to the baseline model (i.e. AP Post-Processor).

	Baseline	AP FSM	AP Decoder
# Periods	3571	3571	3571
% Period Errs	33.2	28.7	25.3
# Commas	1641	1641	1641
% Comma Errs	76.8	55.9	54.1
Proc Time [xRT]	1.81	2.35	1.88

Table 1: *Results for joint decoding and prediction of non-verbalized (NV) punctuations (AP FSM - section 4 and AP Decoder - section 5) and comparison with baseline, i.e. punctuation prediction as a post-processor to decoding (Section 2). Results are given in terms of total punctuation prediction errors (false positives and false negatives).*

From the results, one can see that the punctuation prediction error rates decrease significantly for the AP FSM and AP Decoder approaches, compared to the baseline. The AP FSM approach leads to a slight increase in the overall processing time by roughly 30%, while the AP Decoder approach avoids this overhead almost completely (increasing processing time by only 4% compared to the baseline) while at the same time yielding the lowest punctuation prediction error rates. Finally, changes in overall (word) recognition accuracy were minor [1] and almost exclusively due to changes in punctuations between the baseline and the two proposed approaches, showing that merging the tasks of LVCSR decoding and punctuation prediction does not cause any measurable adverse effects on the pruning behavior of the LVCSR decoder.

[1] Punctuations make up only 4.8% of all tokens in the testset. 73% of all punctuations in the testset are non-verbalized.

7. Conclusion

We have presented a novel approach to the problem of predicting non-verbalized punctuations in the context of large vocabulary conversational speech recognition. We have detailed two techniques for using written-form language models directly in a single-pass Viterbi decoder for producing maximum likelihood hypotheses that include non-verbalized punctuations. This greatly simplifies the complexity and memory requirements of speech recognition systems that are designed to produce correctly punctuated transcriptions from sloppy, conversational speech. We also demonstrated how the proposed approach improves punctuation prediction accuracy over traditional post-processing approaches for lexical cues in a medical transcription domain. Finally, we would like to note that the approach can be generalized to include non-lexical cues, for instance syntactic (e.g. part-of-speech), acoustic or prosodic cues, potentially further improving prediction accuracy.

8. Acknowledgements

The work presented in this paper was performed primarily during a summer internship of the first author at M*Modal. The authors wish to thank Girija Yegnanarayanan, Detlef Koll and the R&D team at M*Modal for their support.

9. References

[1] A. Stolcke, E. Shriberg, D. Hakkani-Tur, G. Tur, Z. Rivlin, and K. Sonmez, "Combining Words and Speech Prosody for Automatic Topic Segmentation," in *In Proceedings of DARPA Broadcast News Transcription and Understanding Workshop*, 1999.

[2] J. Hirschberg and C. H. Nakatani, "Acoustic Indicators of Topic Segmentation," in *Proc. ICSLP*, 1998.

[3] Y. Gotoh and S. Renals, "Sentence Boundary Detection in Broadcast Speech Transcripts," in *Proc. International Speech Communication Association (ISCA) Workshop: Authomatic Speech Recognition: Challanges for the New Millenium (ASR-2000), Paris, September*, 2000.

[4] B. Ramabhadran, O. Siohan, L. Mangu, G. Zweig, M. Westphal, H. Schulz, and A. Soneiro, "The IBM 2006 Speech Transcription System for European Parliamentary Speeches," in *Proc. International Conference on Spoken Language Processing (ICSLP)*, 2006.

[5] M. Finke, J. Fritsch, D. Koll, and A. Waibel, "Modeling and Efficient Decoding of Large Vocabulary Conversational Speech," in *Proc. International Conference on Spoken Language Processing (Eurospeech)*, 1999.

[6] D. Beeferman, A. Berger, and J. Lafferty, "Cyberpunc: A Lightweight Punctuation Annotation System for Speech," in *Proc. IEEE International Conference on Acoustics Speech and Signal Processing pages 689–692 Seattle, WA*, 1998.