RESTORING PUNCTUATION AND CAPITALIZATION IN TRANSCRIBED SPEECH

Agustín Gravano[*]

Department of Computer Science
Columbia University
New York, NY 10027
agus@cs.columbia.edu

Martin Jansche, Michiel Bacchiani

Google Inc.
New York, NY 10011
{mjansche,michiel}@google.com

ABSTRACT

Adding punctuation and capitalization greatly improves the readability of automatic speech transcripts. We discuss an approach for performing both tasks in a single pass using a purely text-based n-gram language model. We study the effect on performance of varying the n-gram order (from $n = 3$ to $n = 6$) and the amount of training data (from 58 million to 55 billion tokens). Our results show that using larger training data sets consistently improves performance, while increasing the n-gram order does not help nearly as much.

Index Terms— Speech recognition, punctuation, capitalization.

1. INTRODUCTION

State-of-the-art automatic speech recognition (ASR) systems still produce raw word streams that, even with no recognition errors, are often difficult to read—not only for humans, but also for natural language processing tools, which usually expect formatted text as input. Issues that need to be addressed include formatting numbers, dates and places, removing disfluencies, inserting punctuation symbols, and correctly capitalizing all words. To illustrate this problem, consider the following excerpt of a broadcast news transcript, along with its formatted version:

> *to simulate the terrain of mars scientists put the rover in hawaii's kilauea volcano the kids sit two thousand four hundred miles away at the nasa ames research center in mountain view california*
>
> *To simulate the terrain of Mars, scientists put the rover in Hawaii's Kilauea volcano. The kids sit 2400 miles away at the NASA Ames Research Center in Mountain View, CA.*

In this work, we limit the problem scope to restoring punctuation symbols and capitalization in ASR output in English.

[*]Work done during an internship at Google Inc.

Previous studies on punctuation restoration have experimented with data-driven techniques for annotating transcribed speech with sentence boundaries alone [1, 2, 3], and with sentence boundaries and other punctuation symbols, predominantly commas and question marks [4, 5, 6, 7, 8]. Capitalization recovery has been explored with various methods for statistical machine translation [9, 10], but has received much less attention for enriching transcribed speech [5].

From these works, it is clear that models that exploit textual and acoustic/prosodic information significantly outperform purely text-based models. However, the availability of massive amounts of written data, together with unceasing progress in computational power and storage capacity, pose the question of the extent to which text-based models may be improved when increasing both the training data size and the n-gram order. The textual models used in the mentioned studies are typically based on word bigrams or trigrams trained on corpora with not more than a few million tokens. This study is aimed at assessing the impact on performance of scaling those two dimensions: we trained and evaluated a series of n-gram language models, systematically varying the n-gram order (from $n = 3$ to $n = 6$) and the size of the training data set (from 58 million to 55 billion tokens of written English).

The rest of the paper is organized as follows: Section 2 describes the corpora used for training and evaluation; Section 3 outlines the method for training and testing the language models; Section 4 summarizes the results; and Section 5 presents some conclusions and possible directions of future work.

2. MATERIALS

The **training data** were taken from a collection of Internet news articles gathered over several years up to June 2007, whose English portion contains 55 billion tokens (including punctuation symbols). We used two corpora as **evaluation data**: one consisting of written news articles, to test our models in the same conditions in which they were trained, and one consisting of broadcast news transcripts, to evaluate how the models generalize to transcribed speech. The written news articles were taken from the Wall Street Journal (WSJ) por-

tion of the Penn Treebank 3 corpus,[1] with 1.3 million tokens from 1989 Wall Street Journal articles. The broadcast news (BN) data were taken from the 1996 CSR HUB4 corpus,[2] collected between January 1992 and April 1996, with 39 million tokens. There is neither textual nor temporal overlap between the three corpora.

All three corpora were conditioned in the same manner, replacing numeric strings (dates, dollar amounts, numeric quantities) with orthographic strings (e.g. *June 1996* becomes *June nineteen ninety six*, *$250* becomes *two hundred and fifty dollars*), replacing abbreviations with corresponding full-word forms (e.g. *Dr. Roberts* becomes *Doctor Roberts*, *Pivet Dr.* becomes *Pivet Drive*), and replacing punctuation characters with corresponding word tokens (e.g. ',' becomes ',COMMA'). The two evaluation corpora had unequal distributions of number of sentences per document, which could bias the results in favor of the corpus with shorter documents. To control for this factor, we subsampled the two corpora to create test sets in which document length was uniformly distributed between 5 and 20 sentences per document. Finally, we removed all punctuation symbols.

3. METHOD

Our objective is to insert both punctuation and capitalization at once, in one single pass. Consistent with what has been reported in previous studies, the most frequent punctuation symbols in all three corpora are, by far, commas and periods, accounting for 40-47% and 31-40% of all symbols, respectively. In consequence, the natural first approach was to consider just these two symbols, either ignoring other symbols, or collapsing them with these two. Additionally, we considered a second symbol set, with commas, periods, question marks and dashes, aimed at studying the performance of the language model on lower frequency symbols. Table 1 shows the rules we defined for converting punctuation characters into either two or four symbols; other symbols, such as quotes, slashes and hyphens, were removed from the data. These transformations were applied to both training and test data.

Original punctuation characters (a)	(b)	Replacement symbol
, ( ) --	,	,COMMA
: ; ! . ? ...	: ; ! .	.PERIOD
	?	?Q-MARK
	( ) ... --	--DASH

Table 1: Conversion tables from original punctuation characters into (a) two or (b) four punctuation symbols.

Let $w_1^L = (w_1, ..., w_L)$ denote a string of L tokens over a fixed vocabulary. An *n-gram language model* assigns a probability to w_1^L according to $\Pr(w_1^L) = \prod_{i=1}^{L} \Pr(w_i | w_1^{i-1}) \approx \prod_{i=1}^{L} \Pr(w_i | w_{i-n+1}^{i-1})$, where the approximation reflects a Markov assumption that only the most recent $n-1$ tokens are relevant when predicting the next token. We trained a series of large-scale *n*-gram models using the distributed infrastructure described in [11]. The models employ the smoothing technique known as *stupid backoff*, which helps avoid a very costly backoff weight computation and which has been shown to work just as well as correct backoff [11]. These language model scores are not probabilities anymore, but retain certain aspects of negative log probabilities.

After training such a language model, it may be used to restore capitalization and punctuation with the following procedure.[3] Given an uncapitalized, unpunctuated sequence of words $w_1, ..., w_L$, we define a *hyper-string FSA* (this construction can be generalized to take word lattices as input) as a finite state automaton (FSA) with:

- two states s_i and t_i for each word w_i $(1 \leq i \leq L)$;
- one extra state s_{L+1};
- one arc from s_i to t_i emitting word w_i capitalized, and one arc emitting w_i uncapitalized $(1 \leq i \leq L)$;
- one arc from t_i to s_{i+1} emitting each of the punctuation symbols being considered, and one arc emitting the empty string ε $(1 \leq i \leq L)$.

The only initial state is s_1, and the only final state is s_{L+1}. Thus, a hyper-string FSA accepts not only the original word sequence, but also all of its possible combinations of punctuations and capitalizations. Figure 1 shows the hyper-string FSA corresponding to the string *"mars scientists"*, taken from the example presented in the Introduction.

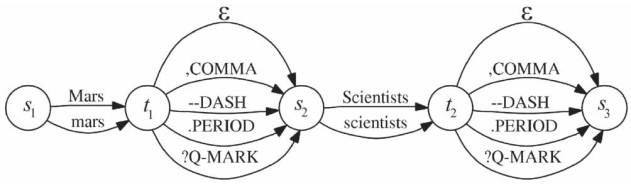


Fig. 1: Hyper-string FSA with all combinations of punctuation and capitalization for *"mars scientists"*.

The distributed infrastructure mentioned above provides an interface for treating the language model as a weighted FSA, where each arc emits a token (word or punctuation symbol) with a certain cost. We then compose the language model and the hyper-string FSA. The result of this composition is an FSA that compactly represents all strings accepted by the hyper-string FSA together with their language model scores. A final decoding step computes the least cost path along the composed FSA, which is equivalent to the maximum posterior probability sequence of punctuation symbols and capitalized/uncapitalized words according to the language model. In

[1] Treebank-3 (LDC99T42), Linguistic Data Consortium, 1999.

[2] 1996 CSR HUB4 Language Model (LDC98T31); Linguistic Data Consortium, 1998.

[3] In this work we address the problem of capitalizing only the initial letter, leaving other cases, such as *NASA* or *LaTeX*, for future research.

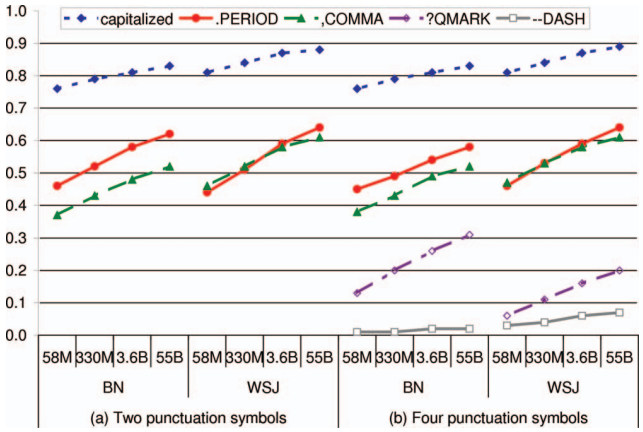
Test corpus: BN. Considering (a) two, (b) four punctuation symbols.

	Size	Capit.	,COMMA	.PERIOD	–DASH	?QMARK
(a)	58M	.80 .72	.45 .32	.55 .39		
	330M	.81 .77	.45 .41	.56 .48		
	3.6B	.82 .80	.47 .50	.59 .57		
	55B	.83 .83	.48 .56	.61 .64		
(b)	58M	.81 .71	.47 .32	.50 .40	.01 .02	.43 .08
	330M	.81 .76	.47 .41	.51 .47	.01 .03	.46 .13
	3.6B	.82 .80	.48 .50	.53 .55	.01 .04	.46 .18
	55B	.83 .83	.49 .55	.56 .60	.01 .06	.47 .24

Test corpus: WSJ. Considering (a) two, (b) four punctuation symbols.

	Size	Capit.	,COMMA	.PERIOD	–DASH	?QMARK
(a)	58M	.87 .75	.54 .40	.58 .36		
	330M	.88 .81	.55 .51	.60 .45		
	3.6B	.88 .86	.55 .62	.63 .55		
	55B	.89 .88	.55 .68	.65 .62		
(b)	58M	.87 .75	.57 .40	.59 .38	.02 .06	.20 .04
	330M	.88 .81	.56 .50	.60 .47	.03 .10	.25 .07
	3.6B	.89 .86	.56 .61	.63 .56	.04 .17	.33 .11
	55B	.89 .88	.57 .67	.65 .63	.04 .21	.29 .15

(i)



(ii)

Fig. 2: (i) Precision, recall and (ii) F-score of 5-grams trained on data sets of varying size (58 million to 55 billion tokens).

the example above, a satisfactory answer in its context would be *"Mars* ,COMMA *scientists* ε*"*.

4. RESULTS AND DISCUSSION

To assess the impact on performance of the amount of training data available, we prepared four data sets with varying numbers of tokens, each with a different order of magnitude: 58 million, 330 million, 3.6 billion and 55 billion. We trained a 5-gram model for each data set, considering either two or four punctuation symbols (as explained in the previous section), and evaluated the trained models on the BN and WSJ corpora. Figure 2 summarizes the results; the table on the left lists precision and recall of each capitalization and punctuation decision; the chart on the right shows a visual representation of the corresponding F-scores.[4] In all cases, increasing the size of the training data set consistently improved the performance of the language models; both precision and recall increased log-linearly with respect to the amount of training data available, without leveling off. Larger amounts of training data would presumably further increase performance.

Next, we trained a series of language models varying the maximum order of the n-grams, from $n = 3$ to $n = 6$, to investigate how this affected performance. We used a fixed training data set with 3.6 billion tokens, and again considered both two and four punctuation symbols. Figure 3 summarizes the results, with all precision and recall scores listed on the left, and F-scores plotted on the right. The BN corpus has a vocabulary of 38 million words (OOV rate 0.2%); WSJ has 1.3 million words (OOV rate 0.1%). For both corpora n-gram coverage for the highest-order n-grams is shown in the column labeled "Covrg." Whereas n-gram coverage remains substantial as n increases, the quality of capitalization and period restoration is flat, presumably because the important clues are highly local and adequately captured by trigrams.

When taking four punctuation symbols into account, periods and commas were inserted much more reliably than the other two. For question marks, the low performance confirms that n-gram models are too simple, and do not capture long range syntactic information needed to distinguish statements from questions: e.g. *"scientists put the rover in Hawaii's Kilauea volcano"* ends in a period, but simply prefixing it with *"why did"* would make a question mark more appropriate.

Looking at differences across corpora, all models performed better on written news articles (WSJ) than on broadcast news reference transcripts (BN). Question marks are an exception, for which the results were better on BN. Although further analysis of the two corpora revealed no significant differences in the distributions of question lengths, we did find that BN questions have a much higher frequency of common endings, such as *'is that correct?'*, *'do you think?'* or *'didn't you?'*, than WSJ questions. While n-grams may model unbounded questions poorly, they are still able to handle these cases correctly, which may explain why question marks are predicted more accurately on BN data.

All language models performed poorly on dashes—probably due to the imprecise definition of this symbol, typically used to indicate a break in the flow of a sentence. Ellipses and parentheses (which were combined with dashes into one symbol) also present high variability in their usage, further complicating the task.

Capitalization restoration was unaffected by the choice of two or four punctuation symbols, and achieved good results, with precision and recall approaching the 0.9 level. Since some capitalization decisions are tied to insertions of sentence boundaries, we further analyzed the types of capitalization er-

[4] Computed as $F = 2 \cdot \text{precision} \cdot \text{recall} \,/\, (\text{precision} + \text{recall})$.

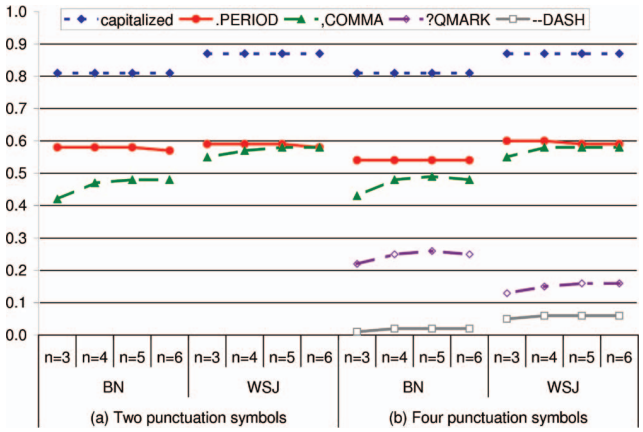
Test corpus: BN. Considering (a) two, (b) four punctuation symbols.

	Order	Capit.	,COMMA	.PERIOD	–DASH	?QMARK	Covrg.
(a)	$n=3$	.82 .81	.49 .37	.58 .58			83%
	$n=4$	.81 .81	.46 .49	.58 .58			53%
	$n=5$	.82 .80	.47 .50	.59 .57			25%
	$n=6$	.83 .80	.48 .47	.61 .54			9%
(b)	$n=3$	.82 .80	.50 .37	.52 .56	.01 .02	.41 .15	82%
	$n=4$	.82 .81	.47 .48	.52 .56	.01 .04	.43 .18	51%
	$n=5$	.82 .80	.48 .50	.53 .55	.01 .04	.46 .18	24%
	$n=6$	.83 .79	.49 .46	.55 .52	.01 .04	.49 .17	8%

Test corpus: WSJ. Considering (a) two, (b) four punctuation symbols.

	Order	Capit.	,COMMA	.PERIOD	–DASH	?QMARK	Covrg.
(a)	$n=3$	.88 .86	.57 .52	.61 .57			78%
	$n=4$	.88 .86	.53 .62	.61 .57			49%
	$n=5$	.88 .86	.55 .62	.63 .55			25%
	$n=6$	.89 .85	.58 .58	.65 .52			12%
(b)	$n=3$	.88 .86	.58 .52	.61 .58	.03 .12	.27 .08	77%
	$n=4$	.88 .86	.55 .61	.61 .58	.04 .16	.27 .11	48%
	$n=5$	.89 .86	.56 .61	.63 .56	.04 .17	.33 .11	25%
	$n=6$	.90 .85	.59 .57	.65 .53	.04 .16	.35 .11	12%

(i)



(ii)

Fig. 3: (i) Precision, recall and (ii) F-score of n-grams of varying order (3 to 6), trained on a 3.6-billion-token data set.

rors. Table 2 shows the distribution of such errors made by the 5-gram language model with two punctuation symbols trained on 55 billion tokens, when applied to the WSJ and BN corpora. The first column shows the expected combinations of punctuation and capitalization; the second column, the decisions made by the model. The two upper rows correspond to genuine capitalization errors, while the two lower rows combine punctuation and capitalization errors. Note that, for both corpora, the latter represented the majority of all capitalization errors; therefore, further improvements in

Expected	Predicted	WSJ	BN
foo	*Foo*	15%	10%
Foo	*foo*	19%	14%
foo	*.Foo*	33%	41%
.Foo	*foo*	32%	34%

Table 2: Distribution of capitalization error types.

punctuation insertion should lead to significant improvements in capitalization restoration.

5. CONCLUSIONS AND FUTURE WORK

We have presented an approach to punctuation and capitalization restoration using purely text-based n-gram language models. Our results suggest that using larger training data sets leads to consistent improvements in performance, while increasing the n-gram order does not help nearly as much. Furthermore, we show that low-frequency symbols such as question marks and dashes are much harder to model using simple n-grams than commas and periods.

A natural direction to continue this research is to combine the text-based models presented here with more complex models described in the literature, which include other textual features (e.g. part-of-speech and shallow syntactic information), as well as acoustic/prosodic features from the audio signal (e.g. pause duration and word final intonation).

6. REFERENCES

[1] E. Shriberg, A. Stolcke, D. Hakkani-Tür, and G. Tür, "Prosody-based automatic segmentation of speech into sentences and topics," *Speech Comm.*, vol. 32(1-2), pp. 127–154, 2000.

[2] Y. Liu, E. Shriberg, A. Stolcke, D. Hillard, M. Ostendorf, and M. Harper, "Enriching speech recognition with automatic detection of sentence boundaries and disfluencies," *IEEE Trans. Audio Speech Lang. Process.*, vol. 14(5), pp. 1526–1540, 2006.

[3] Y. Gotoh and S. Renals, "Sentence boundary detection in broadcast speech transcripts," in *ISCA Workshop on Automatic Speech Recognition*, 2000.

[4] D. Beeferman, A. Berger, and J. Lafferty, "Cyberpunc: A lightweight punctuation annotation system for speech," in *ICASSP*, 1998, vol. 2, pp. 689–692.

[5] E.W. Brown and A.R. Coden, "Capitalization recovery for text," *IR Techniques for Speech Applications*, 2002.

[6] H. Christensen, Y. Gotoh, and S. Renals, "Punctuation annotation using statistical prosody models," in *ISCA Workshop on Prosody in Speech Recognition and Understanding*, 2001.

[7] J. Huang and G. Zweig, "Maximum entropy model for punctuation annotation from speech," in *ICSLP*, 2002.

[8] B. Favre, R. Grishman, D. Hillard, H. Ji, D. Hakkani-Tur, and M. Ostendorf, "Punctuating Speech for Information Extraction," in *ICASSP*, 2008.

[9] C. Chelba and A. Acero, "Adaptation of maximum entropy capitalizer: Little data can help a lot," *Computer Speech and Language*, vol. 20(4), pp. 382–399, 2006.

[10] Wei Wang, Kevin Knight, and Daniel Marcu, "Capitalizing machine translation," in *HLT/ACL*, 2006.

[11] T. Brants, A.C. Popat, P. Xu, F.J. Och, and J. Dean, "Large language models in Machine Translation," in *EMNLP*, 2007.