Comparison of Token- and Character-Level Approaches to Restoration of Spaces, Punctuation, and Capitalization in Various Languages

Laurence Dyer[1], Anthony Hughes[1], Dhwani Shah, Burcu Can

Research Group in Computational Linguistics (RCGL),
University of Wolverhampton, Wolverhampton, UK
{l.j.dyer, a.j.hughes2, d.r.shah2, b.can}@wlv.ac.uk

Abstract

This study compares token- and character-level approaches to restoration of spaces, punctuation, and capitalization to unformatted sequences of input characters, which is a vital step in use cases such as formatting outputs of automatic speech recognition systems for automatic transcription, and formatting hashtags. We obtain an overall F-score of 0.95 for our main English dataset with a hybrid pipeline model composed of a Naive Bayes classifier for space restoration and a BiLSTM classifier with a pre-trained Transformer layer for restoration of punctuation and capitalization. We also propose a novel character-level end-to-end BiLSTM model (overall F-score 0.90) which has the advantages of being able to restore mid-token capitalization and punctuation and of not requiring space characters to be present in input strings. We demonstrate that this model is language agnostic through experiments on Japanese, a *scriptio continua* language, and Gujarati, a low-resource language. We also compare our models with the only existing work on this task of which we are aware by carrying out experiments on the same dataset, and find that all of our models outperform those in that work.

1 Introduction

The accuracy of automatic speech recognition (ASR) systems has improved dramatically in recent years, with increasingly sophisticated deep learning architectures bringing about year-on-year improvements in word error rates (WERs) on established benchmark datasets such as LibriSpeech (Synnaeve, 2022). It has been claimed that ASR systems have reached the level of human parity (Xiong et al., 2016) or even super-human performance (Nguyen et al., 2020). The result has been a resurgence of various speech technologies, not least automatic transcription. Automatic transcription is widely used in the medical field (Van der Straten, 2017) and is now a hot topic in the general business community after many businesses shifted to fully remote or hybrid working following the COVID-19 pandemic. Transcription services integrated into online conferencing applications like Microsoft Teams and Zoom are used on a daily basis by organizations around the world to facilitate information sharing and record keeping.

However, most modern end-to-end ASR systems output streams of lowercase words without punctuation or capitalization (Guan, 2020), so these features must be restored by dedicated models in the post-processing stage. The presence or absence of certain punctuation marks can dramatically alter the understood meaning of a sentence—as anyone who has read the book *Eats, Shoots & Leaves* (Truss, 2004) knows—and the same set of letters can carry completely different meanings depending on the capitalization ("The Who" vs. "the WHO"). Capitalization and punctuation are also known to improve the readability of transcriptions (Jones et al., 2003). Moreover, absence of capitalization and punctuation negatively impacts the performance of downstream natural language processing (NLP) applications such as neural machine translation (Peitz et al., 2011), named entity recognition, and part-of-speech tagging (Mayhew et al., 2019). Appropriate punctuation and capitalization styles may vary depending on domain, so the ability to easily train models to restore these features is likely to become increasingly important in the future.

Most work on punctuation and capitalization restoration to date has been targeted at the token level, assuming that inputs are already correctly segmented into words. However, there are some

[1]Laurence Dyer and Anthony Hughes contributed equally to this work as first authors.

Repositories containing the code used for all of the experiments in this paper, together with comprehensive documentation and links to interactive demo notebooks, are available online at: https://ljdyer.github.io/Space-Punct-Cap-Restoration.

drawbacks to this approach. Firstly, it necessarily restricts the possible positions of each feature within a single word, and therefore cannot learn or restore mid-token punctuation or capitalization such as that found in "iPhone" or "yahoo.com". Secondly, it cannot be applied directly to *scriptio continua* languages like Chinese, Japanese, and Thai. Space restoration may also be required for certain use cases in spaced languages, for example when dealing with hashtags (Abd-hood and Omar, 2021).

For these reasons, in this study we assume raw data that comprises streams of lowercase characters and digits only, and examine a range of approaches to restoring spaces, punctuation, and capitalization to generate readable output texts. We compare pipeline approaches that first restore spaces and then tackle capitalization and punctuation at the token level with end-to-end approaches that perform all tasks simultaneously at the character level. We implement a combination of statistical and deep learning models in order to ascertain whether using neural network architectures leads to better performance in each part of the task under consideration.

2 Related Work

2.1 Space Restoration/Word Segmentation

Space restoration for spaced languages is largely equivalent to the problem of word segmentation for *scriptio continua* languages such as Chinese, for which a wealth of literature exists due to its applications in higher-level NLP tasks including part-of-speech tagging and machine translation. The problem can be tackled using dictionary-based approaches and statistical machine learning methods including Naive Bayes, support vector machines (SVM), and conditional random fields (CRF) (Haruechaiyasak et al., 2008). However, most recent work in this area employs neural networks.

Shao et al. (2018) propose a bidirectional recurrent neural network (RNN) architecture with gated recurrent units (GRU) that works at character level, and demonstrate its effectiveness on over 75 languages from a diverse range of language families. Sun (2010) compares word- and character-based approaches to Chinese word segmentation and proposes a classifier ensemble method after observing that each method leads to different types of error. Zhang et al. (2018) and Liu et al. (2019a) have proposed methods for incorporating information from dictionaries to improve neural network models. Higashiyama et al. (2019) integrate both word character information and character attention into a character-based neural network for Japanese language word segmentation. Zheng and Zheng (2022) improve on existing results for Vietnamese word segmentation by using an improved LSTM model with a convolutional neural network (CNN) extraction portion.

2.2 Restoration of Capitalization and Punctuation

CRF (Lui and Wang, 2013) and finite state automata (FSA) (Gravano et al., 2009) have been used to restore capitalization and punctuation, but state-of-the-art approaches use neural network models. Che et al. (2016) demonstrate that both deep neural networks (DNNs) and CNNs can outperform CRF-based approaches for restoration of commas, periods, and question marks on a dataset of Ted Talks.

Recent work has demonstrated the advantages of using Transformer architectures for restoration tasks. Nguyen et al. (2019) restore capitalization and punctuation in a single model using a Transformer-enriched sequence-to-sequence LSTM model, while Wang et al. (2018) apply an Encoder-Decoder Transformer architecture with attention to restoration of punctuation only. Yi et al. (2020) and Courtland et al. (2020) achieve state-of-the-art results for punctuation restoration using deep bidirectional Transformer architectures with the pre-trained language models BERT (Devlin et al., 2018) and RoBERTa (Liu et al., 2019b) respectively. Guerreiro et al. (2021) improve on the work of Yi et al. (2020) by training and evaluating their models at chunk level rather than segment level.

Guerreiro et al. (2021) also successfully train a single model for punctuation restoration in multiple languages using the multilingual language model XLM-RoBERTa (Conneau et al., 2019). However, they observe that better results for English are obtained when using the monolingual RoBERTa. Gupta et al. (2022) use IndicBERT (Kakwani et al., 2020) to carry out punctuation restoration for 11 Indic languages, including Gujarati.

All of the studies cited above adopt purely lexical approaches, but others (Tilk and Alumäe, 2015; Klejch et al., 2017; Yi and Tao, 2019) have incorporated audio features from original ASR inputs.

Guan (2020) proposes an end-to-end ASR system with punctuation restoration included. Zhu et al. (2022) train a BiLSTM model with a hybrid representation to enable restoration of punctuation to unpunctuated texts either with or without accompanying audio.

Gale and Parthasarathy (2017) is a rare example of a character-level approach to punctuation restoration, in which results on-par with state-of-the-art CRF-based models are obtained using various architectures containing LSTM networks. The models in that paper differ from our own in that they restore punctuation only, and each output tag contains only a single feature.

2.3 Restoration of Spaces, Capitalization and Punctuation

The only study of which we are aware that deals with restoration of spaces, punctuation, and capitalization as a single task is Sivakumar et al. (2021). The authors use a single model type— bigram-level GRU—but adopt a pipeline approach where one feature is restored successively in each model component. Interestingly, they find that the best results are obtained by restoring spaces later in the restoration process, after periods and commas but before capitalization.

3 Models

In this section we describe the models used in this study, which encompass a range of both traditional and neural machine learning techniques for restoring spaces, punctuation, and capitalization, or a subset of those features. The model names in **bold** are unique names that we assigned to the models for the purpose of this paper.

NB is a Naive Bayes-based model that uses dynamic programming with memoization to recursively assign probabilities to possible word segmentations based on frequency lists of words learnt during training. Our implementation closely follows the description of the bigram model in Norvig (2009), but we treat the maximum possible word length L and smoothing parameter λ, which are assigned constant values in the original work, as tunable hyperparameters.

Since **NB** assigns probabilities to word segmentations recursively, inference can only be run on string of up to around 100 characters in length due to recursion limits, so longer inputs have to be broken up into shorter chunks. We use the method in Jenks (2018), where the last few words of each chunk is appended to the beginning of the following chunk to prevent words being incorrectly segmented due to being cut off at the end of a chunk. The same method is employed for **BiLSTMCharSpace** and **BiLSTMCharE2E** to allow for control of the model input length without the need for padding tokens.

BiLSTMCharSpace and **BiLSTMCharE2E** are character-level BiLSTM sequence-to-sequence classification models. They were implemented from scratch for this paper and are not based on existing literature. **BiLSTMCharSpace** is for restoration of spaces only, a binary classification task, so binary cross entropy is used for the loss function and sigmoid is used for the activation function. **BiLSTMCharE2E** uses multi-class classification to restore any possible combination of features following each character. The number of possible classes when restoring n features is 2^n, so for our standard set of features for English a maximum of 16 classes are possible. Categorical cross entropy is used for the loss function and softmax is used for the activation function.

Training examples for **BiLSTMCharSpace** and **BiLSTMCharE2E** were generated using a sliding window over each document with a step size of one word (or three characters in the case of the **OshieteQA** dataset). This approach ensures that less frequent words appear multiple times in the training data. The sequence length was set to 200 characters so that at least two or three sentences would be included in each training sample in order to provide sufficient context for restoration of end punctuation.

CRF uses CRF to restore punctuation and capitalization through multi-class token classification. Our implementation is based on Lui and Wang (2013), but where that study assigns only 4 classes to capture presence or absence of any punctuation and/or capitalization, we increase the number of classes to capture presence or absence of a period and/or comma directly after a token and whether a token is lowercase, title cased, or fully capitalized. Also, the previous study uses named entity recognition (NER) results on uncapitalized input text as a model feature, but we skip this step as we found in initial experiments that the low accuracy of NER without capitalization meant that the impact on model performance was minimal.

BERTBiLSTM is a token-level model which

uses a pre-trained Transformer layer, BERT (Devlin et al., 2018), with the final hidden layer attached to a BiLSTM with sigmoid activation. Classes capture combinations of features, for example a period following a token and capitalization of the first letter of the token. Unlike **CRF**, this model does not have a class for full capitalization of a token. It has a maximum sequence length of 256, with a padding token used to fill any remaining slots in each sequence. Padding tokens are masked to avoid the attention layer deriving importance from them. The model is based on the punctuation restoration model in Alam et al. (2020), but expands on that work by restoring capitalization in the same model.

Models for space restoration and models for restoring other features were combined to form pipeline models for restoration of all features. For example, **NB + BERTBiLSTM** refers to the result of inputting the outputs of **NB** into **BERTBiLSTM**. **BiLSTMCharE2E** is the only model that can restore all features in a single inference step.

4 Experiments

4.1 Languages

The languages studied in this paper are English, Japanese, and Gujarati. These languages were selected in order to ascertain the suitability and performance of the novel character-level model **BiLSTMCharE2E** on a diverse range of language types.

English is a spaced language with rich use of capitalization and punctuation. It is also the most widely studied language in the field and its inclusion was essential for comparison with existing work. Japanese is a *scriptio continua* language in which spaces are not inserted between word tokens, and is also in high demand for NLP applications. Gujarati is a low-resource language belonging to the group of Indic languages with special typographical and orthographic characteristics that need to be taken into account when developing character-based models (W3C Group, 2022).

Further, the authoring team contained at least one member who is either native or proficient in each of the languages under study, which made direct qualitative evaluation and analysis of model outputs possible.

4.2 Datasets

In this section we describe the datasets used in this study. The dataset names in **bold** are unique names that we assigned to the datasets for the purpose of this paper.

Our main dataset for English was a corpus of transcripts of TED Talks collected in June 2020 (Gupta, 2020) (hereafter referred to as **TedTalks**). This dataset was chosen because it consists of high-quality professional transcriptions of spoken English, and so closely reflects the intended use case and desired outputs of our models. Ted Talk transcripts have also been used in many prior studies on punctuation and capitalization restoration including Che et al. (2016), Wang et al. (2018), and Courtland et al. (2020). Our aim when preparing this dataset was to establish the feasibility of each of the models under investigation. We therefore carried out thorough data cleaning with the aim of restricting the input vocabulary while retaining as much of the context of the original as possible. For example, we replaced question marks and exclamation marks with periods, "$" signs with the acronym "USD", and "%" signs with the word "percent". We also removed speaker indicators (e.g. "Narrator:") and descriptions of audience activity (e.g. "(Applause)"), which appear frequently in the original corpus. The cleaned dataset consisted of only upper- and lower-case alphabetic characters, digits, spaces, commas, and periods.

The second dataset for English was the **Brown** corpus, which was used for comparison with Sivakumar et al., 2021. Data cleaning was carried out following the procedure described in Section *A.2* of that paper in order to mirror their experiments as closely as possible.

The Japanese and Gujarati datasets were used to ascertain the feasibility of **BiLSTMCharE2E** on languages with larger character vocabularies. Only minimal data cleaning was carried out for these datasets.

For Japanese, we collected our own corpus, **JapaneseQA**, of questions and answers written in 2021 from the popular Q&A site Oshiete! goo (2022) ("教えて！goo"/"Tell me! goo"). The dataset contains questions and answers from a wide range of categories (16 in total), with the most common being "life advice", "education/science/learning" and "health/beauty/fashion". Both full-width and half-width spaces (which do not generally belong in Japanese text) were re-

Dataset	Language	Num. docs	Num. words	Num. chars	Num. unique chars
TedTalks	English	3,997	7,149,001	39,490,824	65
Brown	English	500	1,023,563	5,921,920	65
JapaneseQA	Japanese	42,940	N/A	25,634,557	5,489
GujaratiNews	Gujarati	3,498	906,498	3,814,697	1,470

Table 1: Datasets used in the experiments

moved, as were line breaks after appending a period (“。”) to lines without end punctuation. Exclamation marks (”！”) were replaced with periods. The texts in this dataset are not transcribed from spoken Japanese, but are written in a conversational style that is similar to polite spoken Japanese, and were chosen for this reason. Since they are written by individual site users as opposed to professional writers or transcribers, the punctuation usage is less consistent than for our other datasets.

For Gujarati, we collected our own corpus, **GujaratiNews**, of news articles written in 2021 and 2022 from the news website Gujarat Samachar (2022). We used written data due to the difficulty of obtaining large collections of transcriptions of spoken Gujarati. We collected article headlines and body text from six of the site's category pages. A period and space were appended to lines without end punctuation before removing line breaks.

The numbers in Table 1 describe the size of each dataset before splitting into train and test sets. A randomly selected 20% of the documents in each dataset were saved for testing, with the remaining 80% used for training and validation.

4.3 Evaluation Metrics

A combination of character- and word-level metrics was used to evaluate the results of our experiments. All metrics were calculated by comparing reference documents in the test sets with document-level model outputs (after combining chunks, etc.).

We used character-level metrics—precision, recall, and F-score for each restored feature—as our primary metrics in order to ensure comparability across all experiments. These scores were calculated based on the features possessed by corresponding non-feature characters in the reference and hypothesis outputs. Since our models were not designed to restore multiple instances of the same feature character after a single input character (e.g. `...`), only presence or absence of each feature for each non-feature character was considered. The character-level metrics were highly informative for understanding the performance of models on each restored feature and identifying future areas for improvement. We do not report character-level accuracy because it takes into account true negatives and therefore can be misleadingly high for features that do not occur frequently.

We also provide word error rate (WER) for all experiments in which spaces were among the features restored. Apart from being a standard metric for ASR tasks, this metric has the advantage of summarizing the overall performance in each experiment in a single percentage, and we (like Sivakumar et al., 2021) found that it closely correlated with our subjective evaluation of model outputs based on observation. Reference and hypothesis texts were split at space characters and the Levenshtein distance between them——the sum of substitutions (S), deletions (D), and insertions (I) required to get from the hypothesis to the reference——obtained. Any number of false positives or negatives across the possible features for a single word prompted a single edit (because, for example, `However,` is not the same as either `however,` or `However.,`). WER is calculated as follows:

$$\text{WER } (\%) = \frac{(S + D + I)}{\text{len}_{\text{ref}}} \times 100 \qquad (1)$$

where len_{ref} is the number of words in the reference text.

Lower WERs indicate better model performance, with 0% indicating a perfect match between reference and hypothesis outputs. WER is affected by the relative frequency of restored features in each dataset, so should not be used to compare the results of experiments on different datasets or when restoring different features.

In addition to quantitative metrics, we also found qualitative observation of reconstructed texts to be highly informative, so we include selected extracts from the model outputs to illustrate some of our observations. In all of the sample outputs presented, a dark gray highlight indicates a false positive, while a light gray highlight indicates a false negative. Fea-

Model	Dataset	Units	Batch size	Dropout	Recurrent dropout
BiLSTMCharSpace	**TedTalks**	$\{128, \mathbf{256}\}$	$\{\mathbf{2048}, 4096, 8192\}$	$\{\mathbf{0}, 0.1, 0.2\}$	$\{0, \mathbf{0.1}, 0.2\}$
BiLSTMCharSpace	**Brown**	$\{128, \mathbf{256}\}$	$\{2048, \mathbf{4096}, 8192\}$	$\{0, 0.1, \mathbf{0.2}\}$	$\{\mathbf{0}, 0.1, 0.2\}$
BiLSTMCharE2E	**TedTalks**	$\{128, \mathbf{256}\}$	$\{\mathbf{2048}, 4096, 8192\}$	$\{\mathbf{0}, 0.1, 0.2\}$	$\{\mathbf{0}, 0.1, 0.2\}$
BiLSTMCharE2E	**Brown**	$\{\mathbf{128}, 256\}$	$\{\mathbf{2048}, 4096, 8192\}$	$\{0, 0.1, \mathbf{0.2}\}$	$\{0, \mathbf{0.1}, 0.2\}$
BiLSTMCharE2E	**OshieteQA**	$\{32, 64, \mathbf{128}\}$	$\{\mathbf{128}, 256\}$	$\{\mathbf{0}, 0.1, 0.2\}$	$\{0, 0.1, \mathbf{0.2}\}$
BiLSTMCharE2E	**GujaratiNews**	$\{64, 128, \mathbf{256}\}$	$\{\mathbf{512}, 1024, 2048\}$	$\{\mathbf{0}, 0.1, 0.2\}$	$\{0, \mathbf{0.1}, 0.2\}$

Table 2: Hyperparameters used for **BiLSTMCharSpace** and **BiLSTMCharE2E**

tures without any highlight are true positives—i.e. correctly restored features.

4.4 Hyperparameter Tuning

Tunable hyperparameters for **NB** are L, the maximum possible word length, and λ, the smoothing parameter. Hyperparameters are applied at inference time, so grid search was carried out by running inference on 5% of the test data with each candidate hyperparameter combination and selecting the one with the highest space F-score for the final model. Combinations with $L \in \{14, 16, 18, 20, 22\}$ and $\lambda \in \{6.0, 8.0, 10.0, 12.0, 14.0\}$ were tested. The hyperparameters selected for use in the final models were $L = 16, \lambda = 12.0$ for **TedTalks** and $L = 14, \lambda = 14.0$ for **Brown**.

Tunable hyperparameters for **BiLSTMCharSpace** and **BiLSTMCharE2E** are the number of BiLSTM units, batch size, dropout rate, and recurrent dropout rate. Grid search was carried out for each model/dataset pair by training on 10% of the training data for one epoch with each candidate hyperparameter combination and selecting the one with the highest validation accuracy for use in the final model. Some candidate hyperparameter combinations overloaded GPU memory, which suggests that GPU resources were optimized for successful combinations. Candidate units and/or batch sizes had to be reduced for the Gujarati and Japanese datasets to generate a sufficient number of viable hyperparameter combinations; this is thought to be due to the larger input vocabularies for these models. Candidate hyperparameters for all model/dataset pairs are presented in Table 2, with the hyperparameters used in the final models displayed in **bold**.

CRF has three tunable hyperparameters: C_1, C_2, and the $possible\_transitions$ parameter, which specifies whether to generates transitions between feature pairs that were not present in the training set. We used the values $C_1 = 1.0$, $C_2 = 1\text{e}{-3}$, and $possible\_transitions = True$, following the example notebook by Korobov (2016).

BERTBiLSTM also has three tunable hyperparameters: learning rate, decay and gradient clip. These were set to $5e-6$, 0, and 1 respectively, the same values as used in Alam et al. (2020).

5 Results

All five of our pipeline and end-to-end models were applied to restoration of capitalization, spaces, periods, and commas for the two English-language datasets. The results for **TedTalks** are presented in Table 3. The metrics for pipelines containing **BiLSTMCharSpace** were almost identical to those containing **NB**, with precision, recall, and F-score differing by no more than 0.01 for each feature and WER differing by no more than 0.28%. The results of pipelines containing **BiLSTMCharSpace** are therefore omitted for brevity. The results for the most performant pipeline model and for the end-to-end model for **Brown** are presented in Table 4.

5.1 Space Restoration (English)

Both **NB** and **BiLSTMCharSpace** have F-scores of 0.99 for the space restoration task on the **TedTalks** dataset. The numbers of space restoration errors in the results for each model are very close, but the errors occur in different places. Of the 22 errors in the results for **BiLSTMCharSpace** on the first document in the test set, only 7 were in words where **NB** also made an error. This is consistent with the observation made in Sun (2010), and suggests it may be possible to obtain higher performance by combining the outputs of the two models in some way.

NB assigns probabilities based on whole tokens learnt during training, so errors occur for unlearnt words that can be formed by concatenating two or more learnt words, such as in `un filled` ("unfilled" was not in the training data). **BiLSTMCharSpace** learns at the character level and appears to have implicitly learnt some of the common morphemes in the English language, as it was able to output the correct word in this and similar examples.

NB + CRF
WER: 19.85%

	Precision	Recall	F-score
Spaces (' ')	0.99	0.99	0.99
CAPS	0.74	0.37	0.49
Periods ('.')	0.59	0.34	0.43
Commas (',')	0.46	0.22	0.30
All	**0.95**	**0.86**	**0.90**

NB + BERTBiLSTM
WER: 8.49%

	Precision	Recall	F-score
Spaces (' ')	0.99	0.99	0.99
CAPS	0.88	0.81	0.84
Periods ('.')	0.82	0.82	0.82
Commas (',')	0.77	0.67	0.72
All	**0.96**	**0.95**	**0.95**

BiLSTMCharE2E
WER: 20.48%

	Precision	Recall	F-score
Spaces (' ')	0.98	0.98	0.98
CAPS	0.69	0.62	0.65
Periods ('.')	0.58	0.53	0.56
Commas (',')	0.50	0.38	0.43
All	**0.92**	**0.89**	**0.90**

Table 3: Results from a selection of models on the **TedTalks** dataset.

BiLSTMCharSpace + BERTBiLSTM
WER: 17.27%

	Precision	Recall	F-score
Spaces (' ')	0.97	0.97	0.97
CAPS	0.77	0.73	0.75
Periods ('.')	0.69	0.70	0.69
Commas (',')	0.60	0.40	0.48
All	**0.93**	**0.91**	**0.92**

BiLSTMCharE2E
WER: 24.46%

	Precision	Recall	F-score
Spaces (' ')	0.97	0.97	0.97
CAPS	0.74	0.42	0.54
Periods ('.')	0.55	0.38	0.45
Commas (',')	0.51	0.11	0.17
All	**0.94**	**0.84**	**0.89**

Table 4: Results from the best pipeline model and the end-to-end model on the **Brown** dataset.

Both models tend to commit errors in cases where part of an input string can be segmented into more than one pair of learnt words, such as the input sequence `chargeshe`. **NB** sometimes chooses the wrong segmentation for the context (in this case `charge she`), whereas **BiLSTMCharSpace** sometimes inserts a space in both positions (`charge s he`). The output of **BiLSTMCharSpace** may be easier to post-edit in such cases, as errors would be picked up by a spellchecker.

BiLSTMCharE2E performs slightly worse than **NB** and **BiLSTMCharSpace** in the space restoration task for **TedTalks**, which suggests that a larger number of possible output classes reduces performance on each individual feature. However, the F-score is still very high, at 0.98, which shows that **BiLSTMCharE2E** is able to accurately restore spaces at the same time as other features.

The results of each model for the space restoration task were also very similar for the **Brown** dataset, with pipelines containing **NB** and **BiLSTMCharSpace** both having F-scores of 0.97 for space restoration. However, for the **Brown** dataset, a pipeline containing **BiLSTMCharSpace** had the highest overall F-score, which suggests that **BiLSTMCharSpace** may perform better than **NB** when less training data is available.

5.2 Capitalization and Punctuation Restoration (English)

Pipeline models containing **BERTBiLSTM** outperformed those containing **CRF** on all features. This is thought to be because **BERTBiLSTM** can take into account more contextual information and also benefits from the pre-trained word Transformer layer. The outputs of **NB + BERTBiLSTM** for the **TedTalks** dataset are generally very readable and closely match the punctuation style of the training data. Figure 1 shows a typical extract. The one divergence from the reference document is also a valid punctuation. The outputs of the other models contain more severe errors that result in less readable text.

BiLSTMCharE2E outperformed pipelines containing **CRF** at capitalization and punctuation restoration (likely owing to its ability to take into account a greater amount of contextual information), with a lower overall F-score due solely to poorer performance at space restoration. Combined with its simplicity due to being able to restore all features in a single model, this makes **BiLSTM-**

```
Imagine a brilliant neuroscientist named Mary. Mary lives in a black
 and white room., She only reads black and white books, and her scre
ens only display black and white. But even though she has never seen
```

Figure 1: Output of **NB + BERTBiLSTM** for an extract from the test set for **TedTalks**

WER: N/A

	Precision	**Recall**	**F-score**
Periods (’。’)	0.54	0.53	0.54
Commas (’、’)	0.31	0.31	0.31
Q. marks (’？’)	0.56	0.48	0.52
All	**0.43**	**0.43**	**0.43**

Table 5:
Results from **BiLSTMCharE2E** on **JapaneseQA**

CharE2E a very viable model for the task under consideration. Further, observation of output texts confirms that **BiLSTMCharE2E** is able to correctly restore mid-token punctuation and capitalization. It was the only model able to correctly restore the middle periods in `B.C.` and `D.C.`, and the final-letter capital in `PhD`, to cite a few examples observed in the first 100 test documents.

The gap in performance between the best-performing pipeline model and **BiLSTMCharE2E** was smaller for **Brown** than for **TedTalks**, which suggests that the negative impact on performance from reducing the volume of training data is lower for **BiLSTMCharE2E**.

Based on reading of the paper and consultation with the authors, we believe the "Distance" metric in Sivakumar et al. (2021) to be equivalent to the metric referred to in this paper as WER. Since the lowest Distance reported in that paper at the testing stage is 29.7, we conclude that all of our models (WER 17.27-24.46%) outperform those in that paper. Inspection of the sample outputs presented in that paper compared with those from our models supports this claim.

5.3 Other Languages

Results for **BiLSTMCharE2E** on Japanese and Gujarati datasets demonstrate that this model can handle large input vocabularies, does not depend on extensive data cleaning, and is readily applicable to *scriptio continua* languages.

Results for the **OshieteQA** dataset are presented in Table 5. F-scores for punctuation marks are comparable to those for the same model on the **TedTalks** dataset, and scores of more than 0.5 for both periods and question marks indicate that the model is able to differentiate to a large extent between sentences and questions based on context. The overall F-score is lower because spaces were not among the features considered for Japanese.

Since **JapaneseQA** consists of texts written by non-professional writers, there is a lot of irregular punctuation in the dataset. Inspection of deviations from the reference examples in the model results reveals that in some cases the model actually renders better punctuations than the "gold standard". A similar normalizing effect is noted in Tilk and Alumäe, 2015. In the example in Figure 2, the author has overused commas, and the model left out some of the unnecessary ones. In order to have this verified by expert and impartial judges, 5 native Japanese speakers were given the characters stripped of punctuation and asked to insert commas and periods as they deemed appropriate. The first comma omitted by the model was included by only 3 out of the 5 participants, and the second was not included by any of them. The irregularity of the dataset may have led to the quantitative metrics underestimating the model’s performance.

Initial results for Gujarati revealed a flaw in the implementation of **BiLSTMCharE2E**. Our original implementation split input strings at the byte level for training and inference, but Gujarati differs from English and Japanese in that graphemes are not equivalent to bytes because combinations of vowel and consonant characters can form a single grapheme. While the model was able to learn to some extent from byte-level information, space restoration performance was surprisingly low (F-score 0.69), and incorrectly restored spaces sometimes caused characters intended to be displayed together with another character as a single grapheme to be displayed separately, leading to ugly and unreadable outputs. This issue was resolved by splitting strings into grapheme clusters rather than bytes. Grapheme clusters are equivalent to bytes for most languages, so the improved model implementation still works as intended for those languages.

Results for **GujaratiNews** when **BiLSTMCharE2E** is trained on grapheme clusters are presented in Table 6. The F-score for spaces is close to those for the English datasets, and F-scores for

ＢＤについては、[3]リージョンＡで、[3]北米も同じだから、[2]再生も可ってことになる。
ＤＶＤビデオの場合だと、[5]プレ イヤーによっては、[0]リージョンフリーのものもある。

For BD it's region A, the same as North America, so playback is possible.
In the case of DVD video, some players are region-free.

Figure 2: An extract from the results of **BiLSTMCharE2E** for the first document in the test set for **JapaneseQA**. The numbers in superscript indicate how many out of a sample of 5 native Japanese speakers included each comma.

WER: 13.85%

	Precision	**Recall**	**F-score**
Spaces (' ')	0.97	0.97	0.97
Periods ('.')	0.86	0.81	0.83
Commas (',')	0.57	0.45	0.50
All	**0.96**	**0.94**	**0.95**

Table 6:
Results from **BiLSTMCharE2E** on **GujaratiNews**

periods and commas are the highest of all of the datasets. This may be due partly to greater regularity of the punctuation usage in the dataset due to its genre, and regularity in the characters used to end clauses and sentences in Gujarati. The scores are nonetheless impressive given the relatively small size of the dataset.

6 Conclusion and Future Work

We assessed a variety of pipeline and end-to-end models for restoration of spaces, punctuation and capitalization on unformatted English, Japanese, and Gujarati texts. Of all of the models considered in this study, the pipeline model **NB + BERTBiLSTM** outperforms the others for restoration of all features under consideration. Both components of this pipeline model take into account token-level information, which suggests that there are advantages to using this information over purely character-based approaches. In particular, a neural network model with a pre-trained Transformer layer was highly performant in restoration of capitalization and punctuation, allowing the pipeline model to render readable outputs that could be used in the real world with very minimal post-editing. **BERTBiLSTM** could be improved further by including a class to restore full capitalization to a token to improve scores for capitalization restoration.

The end-to-end character-based model **BiLSTMCharE2E** outperformed pipeline models containing a CRF-based component for punctuation and capitalization restoration, and only slightly underperformed those models for space restoration. Furthermore, **BiLSTMCharE2E** was shown to be easily applicable to different languages and sets of features, and to be able to restore features to individual characters inside tokens.

Due to the observed advantages of using pre-trained word embeddings for token-level models, we are very interested in investigating the effect of using pre-trained byte level language models such as ByT5 (Xue et al., 2021) on both character- and token-level models. This is left for future work.

References

Samia Abd-hood and Nazlia Omar. 2021. Hashtag segmentation: A comparative study involving the Viterbi, triangular matrix and word breaker algorithms. *Journal of Advances in Information Technology*, 12:311–318.

Tanvirul Alam, Akib Khan, and Firoj Alam. 2020. Punctuation restoration using transformer models for high- and low-resource languages. In *Proceedings of the Sixth Workshop on Noisy User-generated Text (W-NUT 2020)*, pages 132–142, Online. Association for Computational Linguistics.

Xiaoyin Che, Cheng Wang, Haojin Yang, and Christoph Meinel. 2016. Punctuation prediction for unsegmented transcript based on word vector. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 654–658, Portorož, Slovenia. European Language Resources Association (ELRA).

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *Computing Research Repository*, arXiv:911.02116. Version 2.

Maury Courtland, Adam Faulkner, and Gayle McElvain. 2020. Efficient automatic punctuation restoration using bidirectional transformers with robust inference. In *Proceedings of the 17th International Conference on Spoken Language Translation*, pages 272–279, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of deep bidirectional transformers for language

understanding. *Computing Research Repository*, arXiv:1810.04805. Version 2.

William Gale and Sarangarajan Parthasarathy. 2017. Experiments in character-level neural network models for punctuation. In *Interspeech 2017*, pages 2794–2798.

Agustin Gravano, Martin Jansche, and Michiel Bacchiani. 2009. Restoring punctuation and capitalization in transcribed speech. In *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 4741–4744, Taipei, Taiwan. IEEE.

Yushi Guan. 2020. End to end ASR system with automatic punctuation insertion. *Computing Research Repository*, arXiv:2012.02012.

Nuno Miguel Guerreiro, Ricardo Rei, and Fernando Batista. 2021. Towards better subtitles: A multilingual approach for punctuation restoration of speech transcripts. *Expert Systems with Applications*, 186:115740.

Gujarat Samachar. Gujarat Samachar [online]. 2022. [Accessed 31 July 2022].

Anirudh Gupta, Neeraj Chhimwal, Ankur Dhuriya, Rishabh Gaur, Priyanshi Shah, Harveen Singh Chadha, and Vivek Raghavan. 2022. indic-punct: An automatic punctuation restoration and inverse text normalization framework for Indic languages. arXiv:2203.16825. [Submitted to InterSpeech 2022].

Vishal Gupta. TED Talk [online]. 2020. [Accessed 31 July 2022].

Choochart Haruechaiyasak, Sarawoot Kongyoung, and Matthew Dailey. 2008. A comparative study on Thai word segmentation approaches. In *5th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology*, volume 1, pages 125–128, Krabi, Thailand. IEEE.

Shohei Higashiyama, Masao Utiyama, Eiichiro Sumita, Masao Ideuchi, Yoshiaki Oida, Yohei Sakamoto, and Isaac Okada. 2019. Incorporating word attention into character-based word segmentation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2699–2709, Minneapolis, Minnesota. Association for Computational Linguistics.

Grant Jenks. Python word segmentation [online]. 2018. [Accessed 31 July 2022].

Douglas Jones, Florian Wolf, Edward Gibson, Elliott Williams, Evelina Fedorenko, Douglas Reynolds, and Marc Zissman. 2003. Measuring the readability of automatic speech-to-text transcripts. In *8th European Conference on Speech Communication and Technology (Eurospeech 2003)*, pages 1585–1588.

Divyanshu Kakwani, Anoop Kunchukuttan, Satish Golla, Gokul N.C., Avik Bhattacharyya, Mitesh M. Khapra, and Pratyush Kumar. 2020. IndicNLPSuite: Monolingual corpora, evaluation benchmarks and pre-trained multilingual language models for Indian languages. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4948–4961, Online. Association for Computational Linguistics.

Ondřej Klejch, Peter Bell, and Steve Renals. 2017. Sequence-to-sequence models for punctuated transcription combining lexical and acoustic features. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5700–5704.

Mikhail Korobov. python-crfsuite [online]. 2016. [Accessed 20 August 2022].

Junxin Liu, Fangzhao Wu, Chuhan Wu, Yongfeng Huang, and Xing Xie. 2019a. Neural Chinese word segmentation with dictionary. *Neurocomputing*, 338:46–54.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. RoBERTa: A robustly optimized BERT pretraining approach. *Computing Research Repository*, arXiv:1907.11692.

Marco Lui and Li Wang. 2013. Recovering casing and punctuation using conditional random fields. In *Proceedings of the Australasian Language Technology Association Workshop 2013 (ALTA 2013)*, pages 137–141, Brisbane, Australia.

Stephen Mayhew, Tatiana Tsygankova, and Dan Roth. 2019. ner and pos when nothing is capitalized. *Computing Research Repository*, arXiv:2012.02012. Version 2.

Binh Nguyen, Vu Bao Hung Nguyen, Hien Nguyen, Pham Ngoc Phuong, The-Loc Nguyen, Quoc Truong Do, and Luong Chi Mai. 2019. Fast and accurate capitalization and punctuation for automatic speech recognition using transformer and chunk merging. In *2019 22nd Conference of the Oriental COCOSDA International Committee for the Co-ordination and Standardisation of Speech Databases and Assessment Techniques (O-COCOSDA)*, pages 1–5, Cebu, Philippines. IEEE.

Thai-Son Nguyen, Sebastian Stüker, and Alex Waibel. 2020. Super-human performance in online low-latency recognition of conversational speech. *Computing Research Repository*, arXiv:2010.03449. Version 5.

Peter Norvig. 2009. Natural language corpus data. In Toby Seagaran and Toby Hammerbacher, editors, *Beautiful Data*, pages 219–242. O'Reilly, Sebastopol, Canada.

Oshiete! goo. Oshiete! goo [online]. 2022. [Accessed 31 July 2022].

Stephan Peitz, Markus Freitag, Arne Mauser, and Hermann Ney. 2011. Modeling punctuation prediction as machine translation. In *International Workshop on Spoken Language Translation*, pages 238–245, San Francisco, California.

Yan Shao, Christian Hardmeier, and Joakim Nivre. 2018. Universal word segmentation: Implementation and interpretation. *Transactions of the Association for Computational Linguistics*, 6:421–435.

Jasivan Sivakumar, Jake Muga, Flavio Spadavecchia, Daniel White, and Burcu Can. 2021. A GRU-based pipeline approach for word-sentence segmentation and punctuation restoration in English. In *2021 International Conference on Asian Language Processing (IALP)*, pages 268–273.

Savina van der Straten. Voice tech landscape: 150+ infrastructure, horizontal and vertical startups mapped and analysed [online]. 2017. [Accessed 31 July 2022].

Weiwei Sun. 2010. Word-based and character-based word segmentation models: Comparison and combination. In *Coling 2010: Posters*, pages 1211–1219, Beijing, China. Coling 2010 Organizing Committee.

Gabriel Synnaeve. wer_are_we [online]. 2022. [Accessed 31 July 2022].

Ottokar Tilk and Tanel Alumäe. 2015. LSTM for punctuation restoration in speech transcripts. In *Interspeech 2015*, pages 683–687.

Lynne Truss. 2004. *Eats, Shoots & Leaves: the Zero Tolerance Approach to Punctuation*. Gotham Books, New York, New York.

W3C Group. Gujarati Gap Analysis [online]. 2022. Accessed 3 Nov 2022].

Feng Wang, Wei Chen, Zhen Yang, and Bo Xu. 2018. Self-attention based network for punctuation restoration. In *24th International Conference on Pattern Recognition (ICPR)*, pages 2803–2808, Beijing, China. IEEE.

Wayne Xiong, Jasha Droppo, Xuedong Huang, Frank Seide, Mike Seltzer, Andreas Stolcke, Dong Yu, and Geoffrey Zweig. 2016. Achieving human parity in conversational speech recognition. *Computing Research Repository*, arXiv:1610.05256. Version 2.

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. 2021. ByT5: Towards a token-free future with pre-trained byte-to-byte models. *Computing Research Repository*, arXiv:2105.13626. Version 3.

Jiangyan Yi and Jianhua Tao. 2019. Self-attention based model for punctuation prediction using word and speech embeddings. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7270–7274.

Jiangyan Yi, Jianhua Tao, Ye Bai, Zhengkun Tian, and Cunhang Fan. 2020. Adversarial transfer learning for punctuation restoration. *Computing Research Repository*, arXiv:2004.00248.

Qi Zhang, Xiaoyu Liu, and Jinlan Fu. 2018. Neural networks incorporating dictionaries for Chinese word segmentation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1).

Kexiao Zheng and Wenkui Zheng. 2022. Deep neural networks algorithm for Vietnamese word segmentation. *Scientific Programming*, 2022.

Yaoming Zhu, Liwei Wu, Shanbo Cheng, and Mingxuan Wang. 2022. Unified multimodal punctuation restoration framework for mixed-modality corpus. *Computing Research Repository*, arXiv:2202.00468.