Implementation of three seq2seq models for segmentation, punctuation and capitalization prediction.

Isabel Olmos Cánovas
University of Wolverhampton
Murcia, Spain
I.Olmos@wlv.ac.uk

Tatjana Kuznetsova
University of Wolverhampton
Stafford, United Kingdom
t.kuznetsova@wlv.ac.uk

Zane Makejeva
University of Wolverhampton
Wolverhampton,United Kingdom
z.makejeva@wlv.ac.uk

***Abstract*—Automatic Speech Recognition (ASR) systems often provide a stream of unsegmented transcripts whose punctuation and formatting have been lost. Most Natural Language Processing (NLP) applications require segmented and well-formatted text as input, which ASR output cannot provide. This paper provides three sequence-to-sequence methods for word segmentation, sentence segmentation, and punctuation and capitalization restoration. Two similar BLSTM (Bidirectional Long Short-Term Memory) models will be implemented for the segmentation tasks. On the other hand, punctuation and capitalization will be restored using pre-trained BART (Bidirectional and Autoregressive Transformer).**

Keywords— Automatic Speech Recognition (ASR), Natural Language Processing (NLP), word segmentation, sequence-to-sequence, Bidirectional Long Short-TermMemory (BLSTM), Bidirectional and Autoregressive Transformer.

I. Introduction and background

Word segmentation is a Natural Language Processing (NLP) task that can be defined as the process of inserting proper word boundaries between the characters in a sentence. The task of word boundary detection is fundamental for Eastern Languages such as Chinese, Japanese or Thai, which do not use spaces between words. In contrast, word segmentation is not necessary for most NLP tasks in Western languages due to the natural presence of word spacing. Consequently, word segmentation in such languages is less extensively investigated than in Chinese, and considerably fewer resources are available.

However, since the Internet and personal computers became widely distributed in the Western world, automatic word segmentation grew in demand. According to [14] some contexts in which word segmentation is crucial are: hashtag splitting -to improve the performance of sentiment analysis [9] and event detection [11]-, scenarios where word spacing has been lost or distorted for some reason, and optical character recognition (OCR) systems that fail to recognize spaces between words in handwritten or low-resolution scans [6].

Apart from hashtags, word segmentation is also useful for examples of contemporary text formats that do not have spaces between phrases such as source code identifiers, URLs and email addresses [14]. On the other hand, current word segmentation methods may also be helpful when dealing with documents with no spaces, dating historical centuries back, such as Latin manuscripts [13]. In all these contexts, word segmentation becomes an essential prerequisite before natural language tasks are conducted.

As stated by [14] the main question lies on whether word segmentation should be viewed as a task of language modelling, where candidates for word segmentation are to be identified through a heuristic search algorithm, ranked according to their expected likelihood [3]; or as a character-based sequence labelling problem, where labels are assigned to characters according to their place in a word [7]. The latter approach was the one adopted in this research.

Similar to word segmentation, detecting sentence boundaries is widely assumed to be a fundamental NLP task. Sentence boundary detection (SBD) is a linguistic problem, whose main purpose is to find appropriate breaks in sentences. Most languages, such as English, have markers that serve as sentence boundaries, while other languages, including Thai, Lao, and Myanmar, do not have these markers. As a result, there is not much research on sentence segmentation in raw text.

Hence, the majority of present SBD research is found in the field of speech recognition. Automatic Speech Recognition systems have been recently used in a variety of applications to produce automatic indexing, searching, or even for online subtitling, obtaining excellent results. Nonetheless, such ASR output exhibits certain limitations, since the text produced by a standard ASR system consists of raw single-case words without punctuation or capitalization. When two sentences are pronounced back-to-back, a recognition engine will recognize them as one sentence by default. Moreover, these constraints pose an obstacle for readers to grasp context and conduct NLP tasks effectively.

As a result, sentence boundary detection is also regarded as a punctuation restoration task in speech recognition since when the model attempts to restore the period in the text, the position of the sentence boundary is also determined. It is also noteworthy to mention that capitalization is one of the most important aspects in improving human readability, parsing, and Named Entity Recognition [18], and because capitalization and punctuation are two correlated tasks, it was decided that a joint model will be built in order to restore both problems.

II. related works

A. Word segmentation

Many proposed models for word segmentation are based on Chinese language or other logographic languages where a character represents a word or a minimal unit of meaning. Currently, there are many approaches to solve Chinese Word Segmentation (CWS) tasks, including character or word-based models using traditional statistical or neural network settings. Nonetheless, CWS based on neural networks has received a lot of attention in recent years, since they can learn word or character-based features automatically [16].

Therefore, word segmentation can be modelled as a token tagging task or a character-based sequence labelling task [8]. However, these methods present two problems: effective representation of characters and the transition between characters to use contextual information [21]. In Chinese, it is especially important to address word segmentation as a sequence labelling task, because individual characters are morphemes in and of themselves. Labelling schemes define word segmentation as a sequence-to-sequence translation task, where characters are "translated" into their label position. BMES is a common labelling system, where a word's first character is marked with B (begin), the intermediate characters with M (middle), the final character with E (end), and single-word characters with S (single). Another common labelling system consists of converting the first character of each word to '1' and the insider characters to '0'. For sequence labelling tasks, linear models such as Maximum Entropy (ME) and Conditional Random Fields (CRF) are commonly used [20]. Nevertheless, the performance of these methods highly depends on the creation of handcrafted features [16].

On the other hand, as stated by [14], characters have no individual meaning for Western languages, and different approaches to word segmentation can be carried instead of or in combination with sequence labelling [22]. According to [14], the popular methods for word segmentation are Recurrent neural networks (RRN) and transformer architectures. One of the significant differences between Transformer architectures and RNN, such LSTM, is that, for LSTM, computation of states cannot happen in parallel with its decoding speed dramatically reduced. For word segmentation, which often requires vast amounts of raw data before more pre-processing steps take place, such scalability issues are especially relevant.

Because general neural networks still face the problem of learning long-distance sentence information, past work has been successful with LSTM models [24]. Although long-distance RNNs have shown better performance, this approach only considers left-to-right sentence information.

On the other hand, Bidirectional LSTM networks (BLSTM) do not require previous knowledge or pre-design and are competent in establishing hierarchical feature representation in both directions. Many successful word segmentation models are based on BRNNs, such as a proposed sequence to sequence model based on that is adaptable to character-based languages as well space-delimited languages [15]. Similarly, [16] performed sequence-to-sequence CWS using a BLSTM encoder-decoder framework, where the encoder captures the whole bidirectional input information without context window limitations, and the attention-based decoder directly outputs the segmented sequence by simultaneously considering the global input context information and the dependencies of previous outputs.

B. *Sentence segmentation*

Due to the ability of automated learning features, deep learning algorithms have recently achieved outstanding results on sentence classification problems, and these methods can also reduce the need for handcrafted feature engineering. Convolutional neural networks (CNNs) and recurrent neural networks (RNNs) have become popular architectures, and they are frequently coupled with sequence-based or tree-structured models.

Consequently, most research on sentence segmentation also implements LSTM. [23], for example, proposed a BLSTM model to mine the sentence level representation using sequential information about all words. At the same time, these authors employed features derived from the lexical resources such as WordNet or NLP systems including dependency parser and named entity recognizers (NER) Their findings revealed that employing merely word embedding as an input feature is sufficient to produce state-of-the-art results.

According to [23], for sentence modeling, CNNs excel in collecting n-gram features at various positions in sentences using convolutional filters and extracting important relations via pooling operations. In their research, sequence-based and CNN-based models were effectively integrated with both sequence-based model and tree-structured model for sentence classification.

Similarly, [4] proposed a hybrid L-MCNN model to describe the semantics of phrases in order to classify them more efficiently. This model comprises two parts: BLSTM and multichannel convolutional neural networks (CNNs). The results demonstrated that the L-MCNN model can capture both long-distance dependencies and local information inside sentences to improve classification performance by integrating the advantages of both architectures.

[5] also carried a different approach combining BLSTM with a CRF layer. Because of the CRF layer, this model had sentence-level tag information. On POS ((Part-of-Speech), chunking, and NER datasets, the BI-LSTM-CRF model proved to deliver state-of-the-art accuracy. In this experiment, the data was annotated using BIO annotation, where the first word of each sentence is labelled as "B-sent" and all other words are assigned the label "O". Compared to earlier observations, this approach demonstrated to be more resilient and less reliant on word embeddings.

C. Punctuation and true-casing restoration

A vast number of prior experiments have used acoustic and lexical features simultaneously to train models to automatically restore punctuation. Although such acoustic-lexical models can attain a high degree of accuracy, they require high-quality raw speech data, which is expensive. Consequently, more researchers have begun to focus on approaches that restore punctuation only through lexical data.

Some technical directions of lexical-based punctuation restoration include traditional n-gram, Conditional Random Field (CRF), and other statistical sequence labelling methods, where punctuation prediction is modelled as a labelling task, with labels such as "comma", "period" or "none" assigned to the words, denoting the punctuation mark inserted after the word.

Nonetheless, because these methods cannot represent word semantics properly, the implementation of neural networks (LSTM, Bidirectional RNN, CNN) and transformers has gained more attention. Machine translation-based models that convert non-punctuated text into punctuated text have proved to achieve efficient results. Nonetheless, according to [19], despite the fact that phrase-based machine translation (PBMT) can be utilized for this purpose, neural machine translation (NMT) approaches such as those based on encoder-decoder architectures produce the best results. An example of a transformer-based implementation that has achieved excellent results can be seen in [10], who employed a pre-trained BERT model with bidirectional LSTM and a CRF layer, obtaining state-of-the-art results on reference transcriptions.

In recent studies, capitalization and punctuation recovery are treated as correlated multiple sequence labelling tasks. [12] restored punctuation and capitalization using transformer architecture. Other related works that have followed the trend of using pre-trained BERT for English punctuation and true-casing restoration are [2],[1],[17]. [17] fine tuned pre-trained BERT to improve punctuation and capitalization on ASR output. On the other hand, [2] also fine-tuned BERT, for punctuation restoration, achieving a new state of the art on benchmark data. [1] employed pre-trained BERT for automatic punctuation in English and Bangla.

III. METHODOLOGY

The methodology applied consisted of the implementation of three different Seq2Seq models with an encoder-decoder architecture. The approaches for word and sentence segmentation were performed with neural networks, meanwhile, the second approach, for punctuation and capitalization restoration, was based on transformers.

A. *Word segmentation*

In this research, word segmentation has been modelled as a character level sequence labelling task, in which ‘1’ represents characters at the begging of the word before the space, and ‘0’ the characters at the middle and end of the word. Therefore, to follow this approach, a Seq2seq model composed of the encoder and decoder was implemented.

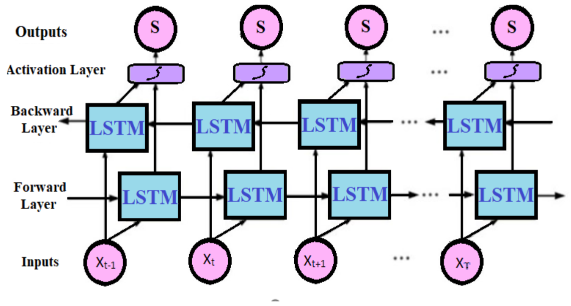


Image I. Encoder Layer - BLSTM.

An encoder reads in the input- in this case a sequence of characters and the labels associated with the word boundaries- and produces a vector containing information about this data relevant for the task. The idea is that the representation produced by the encoder can be used by the decoder to generate correct output to solve word segmentation problems. Image I is an illustration of the encoder layer (i.e. LSTM), which after applying a softmax activation function produces output states which are passed onto the decoder. The decoder then takes the states alongside the decoder input, which is a previous prediction from a step in the past to produce a final prediction of a character before which space should be placed, as shown in Image II.

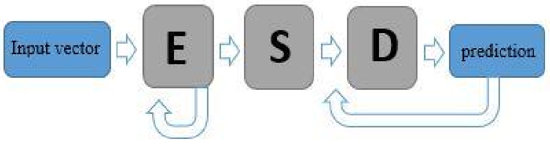


Image II. Encoder-Decoder model.

Following previous state-of-the-art research, BLSTM encoder-decoder architecture was chosen, and one hot character encoding was implemented as a character embedding system. These character embeddings were fed into the BLSTM units. One of the reasons for utilizing BLSTM model for this task is that it can capture long-distance dependencies of the sequence input without having a vanishing gradient problem.

Additionally, a BLSTM also considers information after the current state in the sequence, as well as the information before it which also improves the accuracy of the prediction. This entails replicating the first recurrent layer in the network such that there are two layers side by side, then feeding the input sequence to the first layer and a reversed version of the input sequence to the second layer. Finally, classification is performed using a softmax function and categorical cross-entropy loss. The obtained output is stored as a separate document to be used in the next stage of the pipeline - sentence segmentation.

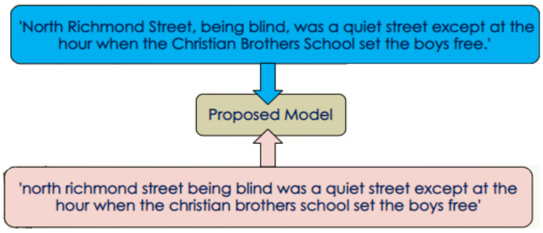
B. *Sentence segmentation*

The BLSTM word segmentation model was also adapted to predict sentence boundaries. The changes were applied to the original dataset, labels and representation of the textual data (see Section IV). Different from the previous model, words were converted to word embeddings using Word2Vec to maintain word context through meaningful numerical representations. Word2Vec is a neural network model that learns word associations. Hence, to convert words to word embeddings, Google news Word2Vec pre-trained model was used. This pre-trained model includes word vectors for a vocabulary of 3 million words and phrases that have been trained on nearly 100 billion words from a Google News dataset. The vector length includes 300 features, which means that each word will be represented by 300 dimensions. To download word2vec-google-news-300, ‘gensim.downloader’ was used.

The first step was to tokenize the words (vocabulary size = 38,726), and transform each sentence into a sequence of integers or indexes. Subsequently, the sequences were padded to ensure that all of them had the same length. This process consists of adding ‘0s’ at the end of each sequence until each sequence has the same length as the longest sequence, being the maximum length for inputs 360, and the maximum length for outputs 385. Following the same process as before, sentence classification was performed using a Softmax function together with categorical cross-entropy loss.

C. *Punctuation and capitalization*

The last model focuses on recovering the two most common and crucial forms of punctuation — commas and periods. Therefore, all other punctuation symbols were eliminated from the corpus, including question marks, exclamation marks, semicolons, and colons. Because capitalization and punctuation restoration are related tasks, they were treated as a Machine Translation or correlated sequence problem using Transformers, treating the non-punctuated and uncapitalized version of our text as the source language, and the punctuated and capitalized version as target language. See Image III.


'North Richmond Street, being blind, was a quiet street except at the hour when the Christian Brothers School set the boys free.'

'north richmond street being blind was a quiet street except at the hour when the christian brothers school set the boys free'

Image III. Proposed model for punctuation and capitalization restoration.

Specifically, the method for true-casing and capitalization restoration consisted of a joint approach using Neural Transfer Learning. Because most recent approaches are based on pre-trained transformers, this same method was followed by employing pre-trained BART (Bidirectional and Auto-Regressive Transformer). BART is a denoising autoencoder for pretraining sequence-to-sequence models, that is trained by corrupting text with an arbitrary noising function, and learning a model to reconstruct the original text. This pre-trained model uses a standard transformer-based neural machine translation architecture that, despite its simplicity, could be generalized as BERT (due to the Bidirectional Encoder) and GPT (for the left-to-right decoder). See Image IV.

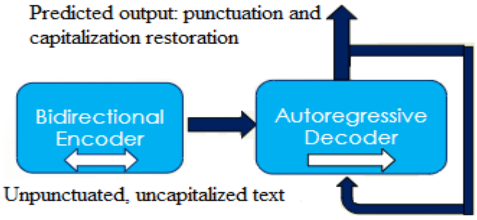

Predicted output: punctuation and capitalization restoration
Bidirectional Encoder
Autoregressive Decoder
Unpunctuated, uncapitalized text

Image IV. Bidirectional Encoder Autoregressive Decoder.

Therefore, the overall architecture of the model follows that of the encoder-decoder architecture, where the encoder and decoder both consist of several Transformer specific self-attention and position-wise feed-forward layers.

BART-large was applied from SimpleTransformers library, which is based on the Transformers library by HuggingFace. BART-large uses 12 encoder and 12 decoder segments. This pre-trained model applies the GeLu activation function for the feedforward segments together with 0.1 dropout to avoid overfitting. Moreover, the optimizer of choice is Adam.

It is also noteworthy to mention that there was an unsuccessful attempt to freeze the layers in order to apply fine-tuning. Nonetheless, due to the shortage of information, this was not possible. Instead, some parameters were adjusted to re-train the model on the new data with a very low learning rate. Among the parameters that were readjusted, it could be listed some task-specific parameters related to summarization, such as maximum length or length penalty.

IV. DATASET

Brown Corpus was used as the dataset to solve the four tasks implemented in this experiment. The major reasons to choose this corpus were its versatility and ease of implementation. For the word segmentation task, data was processed by removing punctuation and spaces in the train x dataset and converting characters to ‘0’ and ‘1’ in the train y dataset. After processing, the data set contained 49,024 sentences and 900,405 words which were split into 80% for training and 20% for testing.

The data employed in the second task consisted of 9,767 rows with five sentences each. Train x contained sentences with only lowercase words (punctuation, capitalization and non-alphabetic characters were removed). On the other hand, train y integrated already capitalized and punctuated sentences whose words were labelled according to the following tags: ‘1’ for the initial word of each sentence, ‘2’ for comma, ‘3’ for period and ‘0’ for the rest of words. By labelling the data in such a manner, there was an attempt to also predict commas and periods, although this was not possible, as it is explained in the next section. Similar to the previous model, the dataset was split into 20% for testing and 80% for training.

Similarly, the training data implemented in the BART model consisted of 20,000 samples of uncapitalized and unpunctuated sentences, with their respective capitalized and punctuated sentences. Although there were several attempts to train this model with the whole dataset (nearly 50,000 samples), this was not possible due to memory issues. Because BART is an autoencoder, it was not necessary to assign any labels to the data, as it automatically converts textual information to numerical data to create the word embeddings. Following the same procedure, 20% of the data was used for evaluation.

V. EVALUATION

A. Word segmentation

BLSTM model was trained with Adam optimizer, learning rate of 0.003 and batch size of 62 sentences. The evaluation is based on the prediction of 6 short sequences where the optimal accuracy is achieved. Table I shows the loss and percentage accuracy, recall, precision and F-score of the word segmentation model and its last epoch metrics, which was trained 5 times for 40 epochs with the same hyperparameters achieving over 96% accuracy overall.

The accuracy levels tend to be either 96% or 98% and rarely anything in between, as well as have little fluctuation between different metrics in the last epoch. The final model is not an optimal approach in mind for this task, the idea was to implement a model that would predict longer sequences, however with time constraints it was difficult to try different approaches that could have improved the existing model, such as splitting the dataset into longer sequences and trying different labelling methods.

TABLE I.

BLSTM Word Segmentation Model Evaluation of Test Data				
Model	***Accuracy/Loss***	***Recall***	***Precision***	***F-Score***
word_seg1	0.0361/0.9888	0.9888	0.9888	0.9888
word_seg2	0.0895/0.9622	0.9622	0.9622	0.9621
word_seg3	0.0438/0.9862	0.9861	0.9862	0.9861
word_seg4	0.0420/0.9859	0.9859	0.9859	0.9859
word_seg5	0.0887/0.9686	0.9686	0.9687	0.9686

Table I. BLSTM Word segmentation evaluation.

B. Sentence Segmentation

The second model for sentence segmentation was trained for 20 epochs at a learning rate of 0.002 and a batch size of 64. Because sentence segmentation was treated as a multi-class classification problem, softmax was implemented along with cross-entropy loss. One limitation exhibited by this model is that it was trained on a maximum sequence length for outputs of 385 words. Hence, because the output of the previous model contained 2,231 words, they had to be split into six sequences of 385 words.

In its twentieth epoch, the model depicted the following results: training loss - 0.0411, training accuracy - 0.9921, recall - 0.9260, precision - 0.9938, validation loss: 0.0413, validation accuracy-0.9918, validation recall - 0.8580, validation precision - 0.9946. Although the results were satisfactory, the validation accuracy demonstrates that the model might have been overfitted, as the accuracy value was too high and hardly changed throughout the epochs. When test data is decoded, the number of tokens in the output sentence matches the number of tokens in the input sequence, without predicting any sentence boundaries before getting split when it reaches the final index position. Hence, the outputs from the model were not plausible sentences in this case, and other options for sentence segmentation were explored.

B. *Punctuation and capitalization*

Because the results from the previous model were not acceptable enough, NNSplit library was used to obtain sentence segmentation to be able to restore true-casing and punctuation. Hence, after having sentences segmented, punctuation and capitalization were treated similarly to any other type of token that needs to be recovered and included in the output. Therefore, BART model was trained with Adam optimizer, learning rate of 4e-5 and batch size of 8 during 5 epochs. This model was measured according to the training and evaluation loss, which were 0.1391 and 0.0421, respectively.

VI. RESULTS

A. *Word segmentation*

A small sample of the result predicting the test file – story.txt can be found in Table II. The extracted sequences were originally a one long sequence output, but for the purpose of demonstration, were split into short sequences. As already mentioned, this model has been trained on long sequences but can only predict short sequences, thus the output is decoded word by word based on a six-word prediction to form a long sequence text to use in the next model for sentence prediction. The results shown in Table II, illustrate the first 22 words in the story.txt which were predicted correctly except for the last word.

TABLE II.

BLSTM Word Segmentation Model Result	
Input	`northrichmondstreetbeingblindwas`
Output	`north richmond street being blind was`
Input	`aquietstreetexceptatthehourwhen`
Output	`a quiet street except at the hour when`
Input	`thechristianbrothersschoolsettheboysfreean`
Output	`the christian brothers school set the boys freean`

Table II. BLSTM Word Segmentation Model results

Similar to the example of 'freean', several words were predicted as one, while others were split into two or three words. Furthermore, the chosen dataset for training included a general American English corpus, this may have impacted the quality of the prediction due to the language style of the task text to be predicted associated with fiction type of texts. Despite this, the overall quality of the prediction is not perfect but most of the words are predicted correctly.

B. *Sentence segmentation*

The results obtained in the BLSTM model were not accurate enough, which might be a consequence of not having implemented POS in the model. Furthermore, this model might have been able to make predictions if pre-tagged ngrams data or even a CRF-layer had been used. On the other hand, regarding the validation accuracy obtained, it could be concluded that the model was overfitted and therefore could not predict the text correctly (validation accuracy = 0.99).

It was complicated to understand the root cause of the problem or what other solutions could be applied to improve the existing model. Therefore, an additional approach was taken by using a pre-trained NNSplit model. By using this library, more accurate sentences, compared to the BLSTM results, were used as an input for the punctuation and capitalization task.

However, issues also arose using this method, as NNSplit's generated sentences were not faultless and some mistakes could be seen in the prediction. The output of the model consisted of each line being represented as a sentence, where several inaccuracies occurred such as sentences being too long or sentences consisting of two words, as shown in Table III. Although two-word sentences are unusual, they can still be plausible sentences if they consist of a noun and a verb, therefore this method performed better than the previous approach, and was adopted to obtain punctuation and true-casing for the next task in the pipeline.

TABLE III.

Examples of inconsistencies using NNSplit Library	
Long sentence	`he turned a silver brace let round and round her wrists he could not goshesaid because there would be a retreat that week in her conventher brother and two other boys were fighting for their caps and i was alone at the railings`
Short sentence	`many times`

Table III. Examples of inconsistencies using NNSplit Library.

C. *Punctuation and true-casing*

The results for the third model were very accurate at predicting capitalization and punctuation. As can be noticed in Table IV, this model does capitalize proper nouns and the beginning of the sentence. It also adds commas to non-defining relative clauses, to separate independent clauses when they are joined by coordinating conjunctions, and to separate two or more coordinate adjectives that describe the same noun, among other uses of the comma.

TABLE VI.

BART-Model Results for true-casing and punctuation restoration	
Input	`north richmond street being blind was a quiet street except at the hour when the christian brothers school set the boys freean`
Output	`North Richmond Street, being blind, was a quiet street except at the hour when the Christian Brothers School set the boys .`
Input	`freean uninhabited house of two storeys stood at the blindend detached from it sneigh bours in a square ground`
Output	`Freean uninhabited house of two storeys stood at the blindend, detached from it, sneigh bours in a square ground .`
Input	`the other houses of the street conscious of decent lives within the mgazed at one another with brown imperturbable faces`
Output	`The other houses of the street, conscious of decent lives within the mgazed at one another with brown imperturbable faces .`

Table VI. BART-Model results for true-casing and punctuation restoration.

VII. CONCLUSION

Due to certain factors, such as time constraints and availability, a single model that would perform all the tasks such as word segmentation, sentence segmentation, capitalization and punctuation could not be achieved. Tasks were divided between team members to find the best solutions for each problem, although punctuation and capitalization were grouped into the same task. As more ideas were explored, the BART approach was discovered and considered as the next approach in the pipeline due to getting errors using other approaches.

Despite difficulties, using this model made it possible to implement a pre-trained model which utilises SimpleTransformers library and HuggingFace framework, suitable for sequence classification problems which presented to be a good solution to complete the task whilst other approaches failed. The idea was to use the output from each model in the pipeline of tasks to address each problem.

High accuracy of 98% was reached with word segmentation tasks using BLSTM approach, however, some limitations applied, in particular with the sequence prediction being limited to 6 words only. Nonetheless, better performance was achieved with BART punctuation and capitalization restoration utilizing sequence to sequence transfer learning method, with an evaluation loss of 0.0421 for corpus test data. However, due to inaccuracies in the word segmentation output, the tasked text classification produced more inconsistencies compared to test data. Although high accuracy of 99% was also identified in the test results of the sentence segmentation problem, in reality, the model was struggling to learn and output plausible sentences based on human evaluation, and a lot of effort was made to find the issue in the source code.

Overall, word segmentation BLSTM and BART models performed to a good standard, except sentence segmentation where efforts to adopt a BLSTM approach did not achieve the same results as word segmentation, however, if more time were allocated for the project, the solution could have been improved. Certain issues arose during the process of finding the right solution as due to many approaches available it was difficult to focus on the specific idea and knowing how to implement which delayed the progress of this project.

Contributions

Table III summarises the contributions of each member of the group to the written report and coding development.

TABLE IV.

CONTRIBUTIONS	
Tatjana	Coding: Word segmentation (BLSTM). Report: methodology, evaluation, results for word segmentation. Datasets and Conclusion.
Isabel	Coding: Sentence segmentation (BLSTM), punctuation and capitalization (BART). Report: Introduction and literature review. Methodology, evaluation, and results for punctuation and capitalization. Methodology, evaluation and results for sentence segmentation.
Zane	Coding: NN-Split library. Report: literature review on sentence segmentation.

Table IV. Contributions to the final project

References

[1] Alam, T., Khan, A. and Alam, F., (2020) Punctuation Restoration using Transformer Models for Resource-Rich and-Poor Languages. In *Proceedings of the Sixth Workshop on Noisy User-generated Text (W-NUT 2020)*, pp. 132-142.

[2] Courtland, M., Faulkner, A. and McElvain, G., (2020). Efficient Automatic Punctuation Restoration Using Bidirectional Transformers with Robust Inference. In *Proceedings of the 17th International Conference on Spoken Language Translation*, pp. 272-279.

[3] Doval, Y. and Gómez‐Rodríguez, C., (2019) Comparing neural‐and N‐gram‐based language models for word segmentation. *Journal of the Association for Information Science and Technology*, *70*(2), pp.187-197.

[4] Guo, Y., Li, W., Jin, C., Duan, Y. and Wu, S., (2018) An integrated neural model for sentence classification. In *2018 Chinese Control And Decision Conference (CCDC)*, pp. 6268-6273. IEEE.

[5] Huang, Z., Xu, W. and Yu, K., (2015) Bidirectional LSTM-CRF models for sequence tagging.

[6] Inuzuka, M.A., Rocha, A.S. and Nascimento, H.A., (2020) Segmentation of words written in the Latin alphabet: a systematic review. In *International Conference on Computational Processing of the Portuguese Language*, pp. 291-302. Springer, Cham.

[7] Li, J., Du, Q., Shi, K., He, Y., Wang, X. and Xu, J., (2018) Helpful or Not? An investigation on the feasibility of identifier splitting via CNN-BiLSTM-CRF. In *SEKE*, pp. 175-174.

[8] Ma, J., Ganchev, K. and Weiss, D. (2020) State-of-the-art Chinese word segmentation with bi-LSTMS. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018*, pp. 4902–4908. doi: 10.18653/v1/d18-1529.

[9] Maddela, M., Xu, W. and Preoţiuc-Pietro, D., (2019) Multi-task pairwise neural ranking for hashtag segmentation. *arXiv preprint arXiv:1906.00790.*

[10] Makhija, K., Ho, T.N. and Chng, E.S., (2019) Transfer Learning for Punctuation Prediction. In *2019 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pp. 268-273. IEEE.

[11] Morabia, K., Murthy, N.L.B., Malapati, A. and Samant, S., (2019) SEDTWik: segmentation-based event detection from tweets using Wikipedia. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop*, pp. 77-85.

[12] Nguyen, B., Nguyen, V.B.H., Nguyen, H., Phuong, P.N., Nguyen, T.L., Do, Q.T. and Mai, L.C., (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)*, pp. 1-5. IEEE.

[13] Rhodes, D., (2013) *Conditional random field Latin word segmenter*. Tech. rep., University of Stanford.

[14] Rodrigues, R.C., Rocha, A.S., Inuzuka, M.A. and do Nascimento, H.A.D., (2020) Domain Adaptation of Transformers for English Word Segmentation. In *Brazilian Conference on Intelligent Systems*, pp. 483-496. Springer, Cham.

[15] Shao, Y., Hardmeier, C. and Nivre, J., (2018) Universal word segmentation: Implementation and interpretation. *Transactions of the Association for Computational Linguistics, 6,* pp.421-435.

[16] Shi, X., Huang, H., Jian, P., Guo, Y., Wei, X. and Tang, Y.K., (2017) Neural Chinese word segmentation as sequence to sequence translation. In *Chinese National Conference on Social Media Processing*, pp. 91-103. Springer, Singapore.

[17] Sunkara, M., Ronanki, S., Dixit, K., Bodapati, S. and Kirchhoff, K., (2020) Robust prediction of punctuation and truecasing for medical asr. In *Proceedings of the First Workshop on Natural Language Processing for Medical Conversations*, pp. 53-62.

[18] Thu, H.N.T., Thai, B.N., Bao, H.N.V., Do Quoc, T., Chi, M.L. and Minh, H.N.T., (2019) Recovering capitalization for automatic speech recognition of vietnamese using transformer and chunk merging. In *2019 11th International Conference on Knowledge and Systems Engineering (KSE)*, pp. 1-5. IEEE.

[19] Vāravs, A. and Salimbajevs, A., (2018) Restoring punctuation and capitalization using transformer models. In *International Conference on Statistical Language and Speech Processing*, pp. 91-102. Springer, Cham.

[20] Wang, C. and Xu, B., (2017) Convolutional neural network with word embeddings for Chinese word segmentation. *arXiv preprint arXiv:1711.04411*.

[21] Yang, H., (2019) BERT Meets Chinese Word Segmentation. *arXiv preprint arXiv:1909.09292*.

[22] Zhang, Y. and Clark, S., (2011) Syntactic processing using the generalized perceptron and beam search. *Computational linguistics*, *37*(1), pp.105-151.

[23] Zhang, S., Zheng, D., Hu, X. and Yang, M., (2015) Bidirectional long short-term memory networks for relation classification. In *Proceedings of the 29th Pacific Asia conference on language, information and computation*, pp. 73-78.

[24] Zhou, H., Yu, Z., Zhang, Y., Huang, S., Dai, X. and Chen, J., (2017) Word-context character embeddings for chinese word segmentation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 760-766.