AIG Investments.AI at the FinSBD Task: Sentence Boundary Detection through Sequence Labelling and BERT Fine-tuning

Jinhua Du , **Yan Huang** and **Karo Moilanen**
Investments AI, AIG (American International Group, Inc.)
{ Jinhua.Du | Yan.Huang | Karo.Moilanen }@aig.com

Abstract

This paper describes the method that Investments AI at AIG (American International Group, Inc.) submitted to the FinSBD-2019 shared task (*"Sentence Boundary Detection (SBD) in PDF Noisy Text of the Financial Domain"*) to extract meaningful, well-formed sentences from noisy unstructured financial text. We approach sentence boundary detection as a sequence labelling task to recognise the start and end token boundaries of sentence(-like) constructs. We evaluated two neural architectures, namely 1) Bidirectional Long Short-Term Memory (BiLSTM) and 2) Bidirectional Encoder Representations from Transformers (BERT). Our extensive experiments on the official FinSBD-2019 datasets demonstrate that a fine-tuned BERT model with customised hyper-parameters (BERT-SBD) outperforms BiLSTM models in several evaluation metrics. Our BERT-SBD submission ranked first on the English test set in terms of *MEAN F1* score in the joint sentence-begin-and-end test condition.

1 Introduction

The sentence is one of the most prominent building blocks in practical NLP and formal linguistics alike. Many, ultimately leaky, definitions for what a sentence is (not) can be found in both communities. At the level of informal common sense, a sentence is taken to represent a "complete thought"[1]. In Halliday's functional-thematic interpretation, the sentence is a basic unit of information composed of a topic (theme) and a comment; and the highest graphological unit of punctuation which conventionally begins with an upper-case letter and ends with a full stop [Halliday, 2004]. The sentence is conventionally the structurally highest construct in formal syntax (lexicogrammar), typically a clause complex or minimally at least one main clause (a predicator with an internal subject complement) [Huddleston and Pullum, 2002]. Many NLP applications and data sets[2] view sentences as arbitrarily truncated text snippets which are simply 'useful' in practical terms.

Regardless of how the sentence is defined formally, sentence boundary detection (SBD) (cf. sentence boundary disambiguation, sentence segmentation, sentence breaking, sentence chunking) is a foundational, critically important upstream step in many NLP applications and (sub)tasks, such as part-of-speech tagging, named entity recognition, dependency parsing, and semantic role labelling, to name a few. Sentence boundary detection attempts to determine the spans (bounds, begin/from-end/to token indices) of sentences and sentence-like constructs below paragraphs, sections, or other suprasentential structures. Because incorrect sentence spans can propagate and generate noise (and undesirable complications) for downstream tasks, SBD plays a critical role in practical NLP applications.

Despite its importance, SBD has received much less attention in the last few decades than some of the more popular subtasks and topics in NLP. On the one hand, (superficially) high baseline performance levels can be achieved by naïve lookup methods that capture obvious, frequent sentence-final punctuation characters such as *[.!?"]* in conjunction with elementary space and case heuristics [Reynar and Ratnaparkhi, 1997]. Such baselines leave little room for further optimisation on traditional test sets derived from formal news(wire) sources. On the other hand, the long tail of exceptions in SBD makes the task non-trivial and challenging: a good majority of potential sentence boundary markers exhibit graphemic (and deeper semantic) ambiguity, particularly the full stop (period) which occurs in abbreviations, initials, honorifics, ordinal numbers, email addresses, ellipses, and the like [Stamatatos and Fakotakis, 1999; Kiss and Strunk, 2006; Gillick, 2009].

Beyond traditional, well-formed, -edited, and -curated news data, the snowballing of noisy web and social media data since the late 1990s has made SBD much harder: when faced with unstructured user-generated content involving tweets, extremely complex graphemic devices (e.g. new emoji, abbreviations, and acronyms), mark-up, and (up to a point) machine-readable data, traditional (and most off-the-shelf) sentence breakers that were trained on "bare" ASCII data in the Penn Treebank (PTB) simply run out of steam [Gimpel *et al.*, 2011; Read *et al.*, 2012]. Canonical SBD approaches optimised for the canonical news genre en-

[1] For example, academic writing guides (https://www.uts.edu.au/sites/default/files/article/downloads/sentence.pdf)

[2] For example, the *"Brief one-sentence movie summary"* field in https://www.kaggle.com/PromptCloudHQ/imdb-data

Proceedings of the First Workshop on Financial Technology and Natural Language Processing (FinNLP@IJCAI 2019), pages 81-87, Macao, China, August 12, 2019.

counter many complications even in other formal domains such as the biomedical [Griffis *et al.*, 2016] or legal [Savelka *et al.*, 2017] ones.

Financial documents which are replete with extremely complex sentences provide one of the most unforgiving but also rewarding application domains for any SBD method. Addressing the lack of SBD research in financial NLP, the FinBSD-2019 shared task [Ait Azzi *et al.*, 2019] focused on the specific challenges that come with noisy financial texts, including impure data extracted and converted automatically from machine-readable formats (such as PDFs). The main task was to detect the spans (begin(ning)/from vs. end(ing)/to token boundaries) of "well-formed sentences" in financial prospectuses - official PDF documents[3] published by investment funds to describe their products to clients.

Datasets for the shared task were released in two languages, English and French. We participated only in the English track. Our system, which relies on state-of-the-art neural models and fine-tuning techniques, approached the FinBSD-2019 challenge as a generic sequence labelling task.

According to the organisers' automatic evaluation metrics, we reached the highest *MEAN F1* score in the English subtask with a 1-point margin over the second-best submission.

2 Task Definition

The majority of past approaches to sentence boundary detection fall into three broad classes: (a) rule-based methods which typically exploit hand-crafted character and spacing heuristics, lookup patterns (e.g. Stanford CoreNLP[4]), or syntactic dependencies (SpaCy[5]); (b) supervised machine learning trained on sentence boundary annotations; and (c) unsupervised machine learning with raw, unlabelled corpora ([Read *et al.*, 2012]).

In practical NLP work, rule-based methods are still popular as they offer the quickest and cheapest way to achieve reasonable performance levels for many NLP tasks. However, if labelled boundary annotations are available, supervised machine learning methods tend to offer greater recall. Previous supervised methods use various strategies to define SBD as a form of classification, for example (i) binary classification to classify each occurrence of *[.!?]* as a valid vs. invalid sentence boundary marker [Reynar and Ratnaparkhi, 1997], or (ii) sequence labelling over multiple classes to tag each token (commonly using a BIO (IOB) tagging scheme [Evang *et al.*, 2013]).

As FinSBD-2019 provided training data with boundary labels (`beginning` vs. `ending`) for each token in text, we opted for classification and evaluated state-of-the-art supervised neural models to classify each token in the text to a given class in conjunction with sequence labelling.

We observed the following in the training and development sets of FinSBD-2019 data:

- We found **953** distinct `beginning` tokens, where determiners, prepositions, conjunctions, and particles such as *A, The, In, For, And* cover more than 50% compared to nouns, pronouns, digits, and miscellaneous single-character constructs (e.g. *Investments, LUXEMBOURG, a, b, 1, 2*).
- We found **207** distinct `ending` tokens, where the full stop, semicolon, and colon cover more than 90% compared to ordinary nouns, numerical tokens (e.g. (year) *"2014"*), and the like. Note that most traditional sentence boundary gold standards do not use such implicit, structurally opaque tokens as sentence boundary markers.

Regarding well-formedness, we observe that the majority of FinSBD-2019 annotations appear to capture sentence or sentence-like constructs which fall under conventional definitions (cf. Section 1). However, many exceptions can be found in the data set, for example constructs devoid of a main verb or a sentence-final period, and other largely arbitrary fragments. Some example sentence(-like) annotations from the training data are shown below.

All Shares will be issued in registered form .
(b) the legal requirements and
Any member state of the EU .
– bonds and other forms of securitised debt ,

We are not aware of any inter-annotator agreement scores that would estimate human performance in financial sentence boundary detection, and shed light on the quality and reliability of FinSBD-2019 sentence annotations.

Owing to the fact that 1) both sentence `beginning` and `ending` tokens need to be recognised, and 2) punctuation characters do not alone suffice as `ending` tokens, we define the following three (3) classes for each token to be used in sequence labelling

- `S`: Start (sentence-initial token)
- `E`: End (sentence-final token)
- `O`: Other (sentence-internal token, neither the start nor the end)

which we use to annotate tokens in sentences such as

The/S company/O made/O £10k/O during/O 2015/O ./E

3 SBD Systems

Deep neural networks (DNN) have pushed the state of the art in many areas of NLP. A DNN model learns a hierarchy of nonlinear feature detectors that can capture more and more complex syntactic and semantic representations. Two DNN architectures are particularly popular, namely recurrent neural networks (RNN) with long short-term memory (LSTM) [Hochreiter and Schmidhuber, 1997] cells or gated recurrent units (GRU) [Cho *et al.*, 2014], and Transformer [Vaswani *et al.*, 2017] which exploits feedforward neural networks and multi-head self-attention mechanisms. We chose two open source systems that are variants of these two architectures for our submission, namely BiLSTM-CRF [Ma and Hovy, 2016] and BERT [Devlin *et al.*, 2018].

[3]Sample prospectus: https://bit.ly/2QztxR0 (via Google).

[4]https://stanfordnlp.github.io/CoreNLP/ssplit.html [Manning *et al.*, 2014]

[5]https://spacy.io/api/annotation#sentence-boundary

3.1 BiLSTM-CRF

An RNN or LSTM [Hochreiter and Schmidhuber, 1997] maintains a memory based on a history which enables the model to predict the current output on the basis of past information and outputs. Bidirectional LSTM [Schuster and Paliwal, 1997] is a variant of unidirectional LSTM which connects two hidden layers of opposite directions to the same output so it can capture information from past and future states simultaneously. In a sequence labelling task, we can efficiently access both past (via forward states) and future (via backward states) input representations for a specific time step t, as shown in Fig. 1.

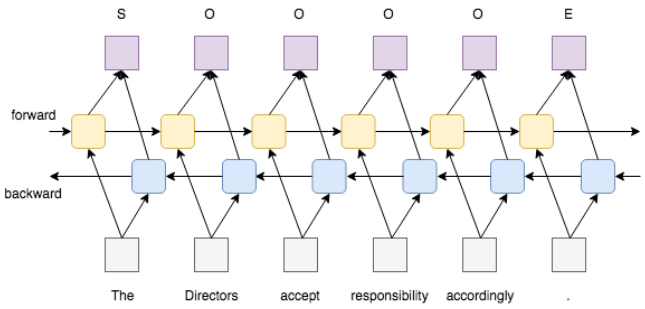

Figure 1: A BiLSTM architecture for sequence labelling for SBD.

We can see that each input token is encoded to a hidden state by the forward and backward LSTM network, respectively, through the integration of previous context information. In the output layer, two hidden states from the forward and backward networks are typically concatenated and then fed into a softmax function to generate a probability distribution for a given label set. The label with the highest probability is conventionally chosen as the final prediction.

Although the current hidden state in an LSTM network does exploit a limited history, the previous neighbour tag is not used when the current tag in the final output layer is predicted. However, the linear order of tags does matter in many sequence labelling tasks. For example, in our SBD task, the sentence-initial start tag (S) has to precede the sentence-final end tag (E). To account for such constraints, the linear-chain Conditional Random Fields (CRF) model is often connected to the output layer of an LSTM network. Fig. 2 shows a hybrid BiLSTM-CRF architecture of this kind.

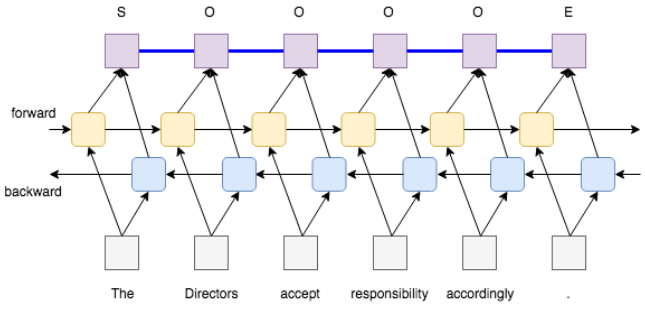

Figure 2: A BiLSTM-CRF architecture for sequence labelling for SBD.

In Fig. 2, the CRF network is represented by the blue lines which connect consecutive BiLSTM outputs. The CRF layer is parameterised by a state transition matrix which indicates the transition probability from one state to another. With such a layer, we can use past and future tags to predict the current tag, correspondingly.

The basic input unit for the BiLSTM-CRF network is conventionally a word (token) which is converted to a vector representation with a fixed dimension. Word vectors are generally pre-trained using neural networks on large-scale datasets (e.g. word2vec [Mikolov *et al.*, 2013], GloVe [Pennington *et al.*, 2014]). Pre-trained word embeddings can be used as initial values for input words or fine-tuned further during training. Pre-trained word embeddings, which can provide a boost for many NLP tasks, are convenient because task-specific training data sets tend to be relatively small. However, word-level inputs are not without their own complications the most prominent of which are 1) out-of-vocabulary (OOV) items, and 2) necessarily limited representative power regarding deeper semantics. Therefore, character-level embeddings are typically used in conjunction with word-level embeddings to represent words. Fig. 3 illustrates the use of a BiLSTM network to learn character-level embeddings for words.

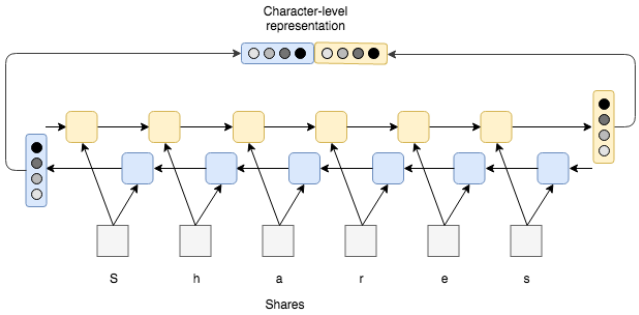

Figure 3: A BiLSTM network for character-level word representation.

The final input representation in our BiLSTM system is a concatenation of word-level embeddings derived from GloVe and character-level embeddings trained using the BiLSTM network. The complete architecture of our BiLSTM-CRF system for SBD is shown in Fig. 4.

The character representation in Fig. 4 is the output from the BiLSTM network for character-level word representation (see Fig. 3).

3.2 BERT

Conventional DNN models, which tend to require large datasets and which can take days to converge, are typically trained from scratch for a given task. Attention has recently moved towards more efficient transfer learning paradigms which first pre-train a DNN model on large datasets, and then fine-tune them towards a specific domain or task. Recent approaches have opted for pre-trained neural language models instead of pre-trained embeddings. BERT, which has achieved state-of-the-art performance in many NLP tasks, is the most representative pre-trained model in this regard.

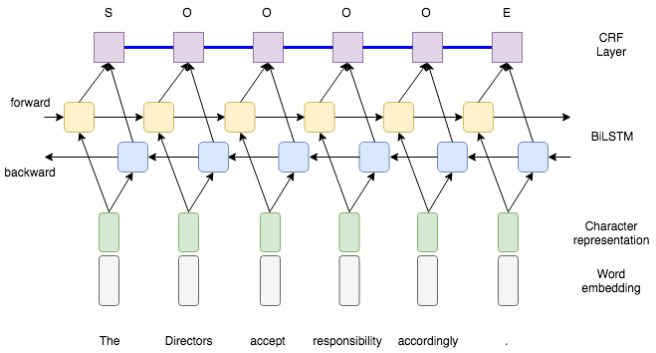

Figure 4: The BiLSTM-CRF architecture of our FinSBD-2019 submission.

BERT uses a multi-layer Transformer encoder [Vaswani *et al.*, 2017] to pre-train deep bidirectional representations by jointly conditioning on both left and right context across all layers [Devlin *et al.*, 2018]. As a result, pre-trained BERT representations can be fine-tuned conveniently using only one additional output layer. Fig. 5 illustrates the Transformer and BERT architectures.

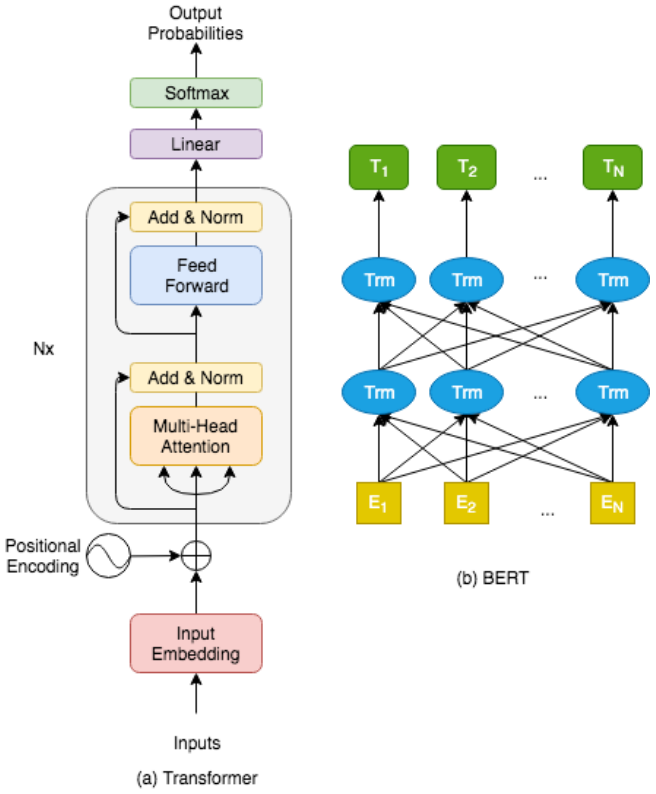

Figure 5: The Transformer (a) and BERT (b) architectures. NB. `Trm` refers to (a).

Transformer makes use of self-attention (instead of RNNs or CNNs) as its basic computational block. Transformer uses a combination of self-attention and feed-forward layers in the encoder. In the standard Transformer model, the encoder is composed of a stack of $N_x = 6$ identical layers, with each layer having two sublayers, namely a multi-head self-attention mechanism and a position-wise fully connected feed-forward network. A residual connection is utilised around each sublayer, followed by layer normalisation.

An attention function can be described as a way to map a query and a set of key-value pairs to an output, where the query, keys, values, and output are all vectors. The output is computed as a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key [Vaswani *et al.*, 2017].

For a given token, BERT's input representation is constructed by summing the corresponding token, segment, and position embeddings. BERT is trained using two unsupervised prediction tasks, Masked Language Model and Next Sentence Prediction. To fine-tune BERT towards a sequence labelling task, the final hidden representation T_i for each token i is fed into a classification layer over the label set. The predictions are not conditioned on the surrounding predictions.

Since we view our SBD task as a sequence labelling problem, we configure BERT to instantiate the token tagging architecture shown in Fig. 6, where C is the hidden state for the first token in the input which corresponds to the special `CLS` word embedding.

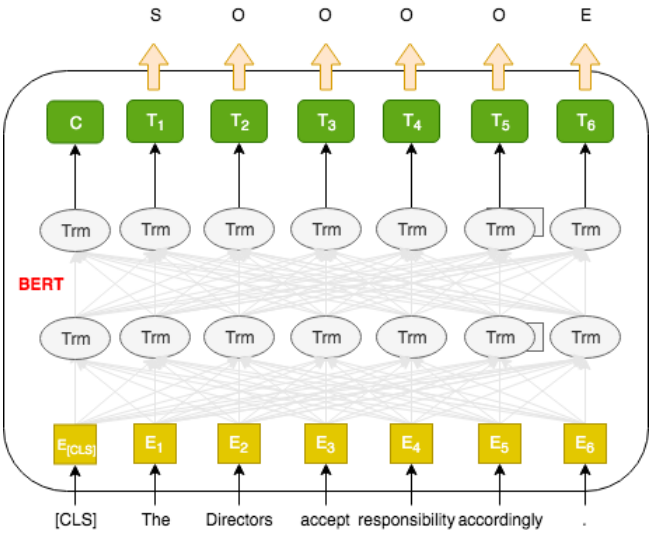

Figure 6: BERT fine-tuned towards sequence labelling for SBD.

4 Experiments

We built two neural systems using the official training data, and tuned their parameters on the validation set. This section summarises the data we used and the steps we took to build, fine-tune, test, and evaluate our systems.

4.1 Data

The FinSBD-2019 data set contains financial documents which had been pre-segmented automatically. The data

	Training	Validation	Testing
#Segment	57,497	2,036	2,505
#Token	904,057	49,859	56,952
#Vocabulary	12,047	2,843	3,539
#Sentence	22,342	1,384	1,265
Max Length	581	303	553
Min Length	1	1	1
Avg Length	15.7	24.5	22.7
Coverage (%)	–	89.69	86.89

Table 1: FinSBD-2019: summary statistics.

was provided as a JSON file which contains raw `text` (without any sentence boundaries) with accompanying `begin_sentence` and `end_sentence` token indices for sentence boundaries, respectively. The raw text had been pre-tokenised using NLTK [Loper and Bird, 2002]. The first token in the text is indexed 0. Table 1 shows summary statistics for the official FinSBD-2019 data set.

In Table 1, *#Segment* indicates the total number of segments (sequences of words on separate lines); *#Vocabulary* is the total number of unique tokens in the text; and *#Sentence* represents the total number of well-formed sentences in the text.

Note that the labels for the test set were released after submission. We can observe that the test set is somewhat different from the validation one, with more segments, tokens, and unique tokens, and fewer well-formed sentences in the former. This difference implies that the test set may be noisier or somehow more complicated, or simply of a poorer quality.

Max Length is the maximum length of the segment in the text, *Min Length* is the minimum segment length, and *Avg Length* is the average length of the segment over the text. It can be seen that the distribution of segment lengths is markedly unbalanced.

The *Coverage*, which indicates how many unique tokens from the validation or test set appear in the training set, can be used as a proxy to quantify the presence of out-of-vocabulary (OOV) or unknown tokens. We can see that the validation set and the test set are comparable in this regard.

4.2 Text Pre- and Post-Processing

We observe that the segments provided are not always correct syntactically, for example in cases where a (syntactic) sentence had been split across multiple segments. In such cases, we cannot use the provided segments as direct inputs to our SBD systems. We followed a simple text pre-processing strategy as follows:

- We remove all newline symbols in the text, and convert it to a single continuous token sequence.
- We split the resultant sequences into short(er) sequences through a sequence length parameter (L). We use $L = 60$ for our BiLSTM-CRF system and $L = 250$ for our BERT-SBD system in the submissions.
- We label each sequence using our pre-defined { `S`, `E`, `O` } label set following the CoNLL2003 BIO (IOB) tagging scheme [Tjong Kim Sang and De Meulder, 2003].

Parameter	BiLSTM-CRF	BERT-SBD
Pre-trained model	–	bert-base-cased
Max seq length	60	256
Lower case	False	False
Batch size	20	32
Learning rate	0.001	5e-5
Learning decay	0.9	–
Train epochs	15	5
Dropout	0.5	–
Optimiser	Adam	–
Hidden size char	100	–
Hidden size SBD	300	–
Char embedding dim	100	–
Word embedding dim	300	–

Table 2: Parameter configurations for our submission.

- Tokens which are in an unsupported encoding or which cannot be recognised by BERT are tagged as *UNK*.

We also rely on an additional, simple post-processing strategy to process the outputs from the two SBD systems: for predictions with only one `S` or `E`, we look for simple punctuations and upper-case characters in limited context windows to reconstruct the missing `E` or `S`, correspondingly.

4.3 System Settings

The hyper-parameters are shown in Table 2. *Hidden size char* denotes the hidden size of BiLSTM for character-level embedding training while *Hidden size SBD* indicates the hidden size of BiLSTM-CRF for the SBD task. *bert-base-cased* has 12 layers with a hidden size of 768 and 12 multi-head attentions, with 110M parameters in total. BERT-SBD uses default configurations for other parameters.

We use WordPiece [Wu *et al.*, 2016] embeddings with a 30k-token vocabulary, and denote split word pieces with `##`. In terms of pre-trained word embeddings, we use *glove.6B*[6] which is trained with 6B tokens and a 400k vocabulary from the *Wikipedia 2014 + Gigaword 5* corpora. We used public domain implementations [7,8] throughout our experiments which were run on four (4) Tesla M60 GPUs. It takes about 10 minutes to fine-tune the BERT-SBD model, and about 40 minutes to train the BiLSTM-CRF model.

4.4 Evaluation Metrics

Because the `beginning` (`BS`) and `ending` (`ES`) tokens[9] of sentences are evaluated separately, the official FinSBD-2019 evaluation metrics include 1) *F1* scores for predicting `BS` and `ES` tokens separately as well as 2) the mean of two separate *F1* scores. We refer to the latter as a soft (lenient) evaluation metric. During training and validation, we used standard evaluation metrics – *Precision* (*P*), *Recall* (*R*), and *F1* score – to evaluate `BS` and `ES`.

[6] http://nlp.stanford.edu/data/glove.6B.zip

[7] https://github.com/guillaumegenthial/sequence_tagging

[8] https://github.com/kamalkraj/BERT-NER

[9] Official FinSBD-2019 annotations use `BS` for sentence `beginning` and `ES` for sentence `ending`.

To extract well-formed sentences, both `beginning` and `ending` tokens need to be predicted accurately. We accordingly propose an additional harsh evaluation metric – *PairSE* – based on the use of *P*, *R*, and *F1* in information retrieval. *PairSE* considers the predicted boundary to be correct only when both `BS` and `ES` are correct, calculated as:

$$P = \frac{\{\text{Correct pairs of S and E}\}}{\{\text{All predicted pairs of S and E}\}}$$

$$R = \frac{\{\text{Correct pairs of S and E}\}}{\{\text{All ground truth pairs of S and E}\}}$$

$$F1 = \frac{2 \times P \times R}{P + R}$$

Consider the example sentence *"The company made £10k during 2015 ."* in Section 2: When *"The"* is predicted as `S` and *"."* as `E`, the pair is counted as a correct prediction for *PairSE* ; if either prediction is incorrect or missing, the pair is counted as an incorrect prediction.

4.5 Results and Analysis

Table 3, which includes both the official and our harsh *PairSE* evaluation metric, shows our performance on the validation set with different parameter settings.

System	Class	Official			*PairSE*		
		P	*R*	*F1*	*P*	*R*	*F1*
BiLSTM1	S	84.3	91.6	87.8			
	E	88.9	96.7	92.6	78.7	85.6	82.0
	Avg	86.6	94.1	90.2			
BiLSTM2	S	82.8	91.8	87.0			
	E	88.6	98.3	93.2	78.4	87.1	82.5
	Avg	85.7	95.1	90.1			
BERT1	S	89.5	94.4	91.8			
	E	91.6	97.4	94.4	86.1	91.6	88.8
	Avg	90.5	95.9	93.1			
BERT2	S	89.4	94.9	92.1			
	E	92.5	98.3	95.3	86.9	92.4	**89.6**
	Avg	91.0	96.6	**93.7**			

Table 3: Experimental results on the validation set.

In Table 3, we refer to our systems with different settings regarding the maximum input length. `BiLSTM1` stands for the BiLSTM-CRF system in which the input length is limited to 100. The input sequence length is set to 60 for `BiLSTM2`, which is also denoted as `AIG2` in our submissions. `BERT1` is the BERT-SBD system where the input length is constrained to 128 while `BERT2` is set to 256 in terms of the input length (denoted as `AIG1` in our submissions). In addition, *Avg* represents the MEAN of scores for the corresponding `S` and `E`.

On the basis of these results, we conclude that

- Two BERT systems significantly outperformed two BiLSTM-CRF systems across all evaluation metrics.
- `BERT2` dominated in terms of *F1* scores.
- Although `BiLSTM1` and `BiLSTM2` achieved similar scores regarding the official *F1* measure, `BiLSTM2` performed better than `BiLSTM1` in terms of our harsh *PairSE* metric. We therefore chose `BiLSTM2` as one of our submission systems.
- `BiLSTM1` obtained higher *P* and lower *R* levels compared to `BiLSTM2`, which demonstrates that LSTMs can learn long-distance dependencies for predicting sentence boundaries correctly given a long (enough) input sequence.
- For all systems, *Recall* was higher than *Precision* - we suspect our systems are prone to committing to a sentence boundary in ambiguous cases.
- All systems obtained higher scores on `E` than `S` which indicates that the `ending` of a sentence is easier to predict than the `beginning`, presumably due to the greater frequency of sentence-final punctuation characters (such as the period) and greater diversity of sentence-initial characters.

Table 4 shows our results on the test set for both the official evaluation metrics and our harsh *PairSE* one.

System	Official			*PairSE*		
	BS	ES	*MEAN*	*P*	*R*	*F1*
AIG2	0.83	0.88	0.855	71.9	86.6	78.6
AIG1	0.88	0.89	**0.885**	78.5	90.2	**84.0**

Table 4: Experimental results on the test set.

`AIG1` is our BERT-SBD system with input length 256, and `AIG2` is our BiLSTM-CRF system with input length 60. Our `AIG1` submission, which is significantly better than `AIG2`, ranks first amongst all submitted systems in terms of *MEAN F1* score.

5 Conclusions and Future Work

We have described the entry by Investments AI at AIG (American International Group, Inc.) to the FinSBD-2019 shared task (English track). We experimented with two neural systems - BiLSTM-CRF and BERT. We approached sentence boundary detection as a sequence labelling problem, and applied a BIO (IOB) tagging scheme to sentence-initial, -final, and -internal tokens to enrich FinSBD-2019 training data and to train our systems. We fine-tuned our systems with different hyper-parameter settings, and chose BERT-SBD with input length 256 and BiLSTM-CRF with input length 60 for our final submission to the shared task.

Our experimental results on the validation set to date show that our BERT-SBD system performs significantly better than the BiLSTM-CRF variant regarding both the official and our harsher *PairSE* metric. AIG Investments AI BERT-SBD system achieved the highest *MEAN F1* score in the shared task. Our approach and results motivate further research into the use of pre-trained language models for sentence boundary detection. In the future, we will explore more detailed error analyses, evaluate the performance of our SBD systems on even noisier financial documents.

Acknowledgments

The authors would like to thank the reviewers for their insightful comments, Guruprasad Sethurathinam for setting up Amazon Web Services (AWS) for our work, and AIG internal reviewers for providing useful feedback.

References

[Ait Azzi *et al.*, 2019] Abderrahim Ait Azzi, Houda Bouamor, and Sira Ferradans. The finsbd-2019 shared task: Sentence boundary detection in pdf noisy text in the financial domain. In *The First Workshop on Financial Technology and Natural Language Processing (FinNLP 2019)*, Macao, China, 2019.

[Cho *et al.*, 2014] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 EMNLP*, pages 1724–1734, October 2014.

[Devlin *et al.*, 2018] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[Evang *et al.*, 2013] Kilian Evang, Valerio Basile, Grzegorz Chrupała, and Johan Bos. Elephant: Sequence labeling for word and sentence segmentation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1422–1426, October 2013.

[Gillick, 2009] Dan Gillick. Sentence boundary detection and the problem with the U.S. In *Proceedings of NAACL*, pages 241–244, June 2009.

[Gimpel *et al.*, 2011] Kevin Gimpel, Nathan Schneider, Brendan O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A. Smith. Part-of-speech tagging for twitter: Annotation, features, and experiments. In *Proceedings of the 49th ACL*, pages 42–47, June 2011.

[Griffis *et al.*, 2016] Denis Griffis, Chaitanya Shivade, Eric Fosler-Lussier, and Albert M. Lai. A quantitative and qualitative evaluation of sentence boundary detection for the clinical domain. In *In Proceedings of AMIA Joint Summits on Translational Science*, pages 88–97, 2016.

[Halliday, 2004] M.A.K. Halliday. *An Introduction to Functional Grammar*. Arnold, London, United Kingdom, 2004.

[Hochreiter and Schmidhuber, 1997] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997.

[Huddleston and Pullum, 2002] Rodney Huddleston and Geoffrey K. Pullum. *The Cambridge Grammar of the English Language*. Cambridge University Press, Cambridge, United Kingdom, 2002.

[Kiss and Strunk, 2006] Tibor Kiss and Jan Strunk. Unsupervised multilingual sentence boundary detection. *American Journal of Computational Linguistics*, 32(4):485–525, 2006.

[Loper and Bird, 2002] Edward Loper and Steven Bird. Nltk: The natural language toolkit. In *Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics - Volume 1*, pages 63–70, 2002.

[Ma and Hovy, 2016] Xuezhe Ma and Eduard Hovy. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074, August 2016.

[Manning *et al.*, 2014] Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of ACL*, pages 55–60, 2014.

[Mikolov *et al.*, 2013] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26*, pages 3111–3119. 2013.

[Pennington *et al.*, 2014] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Proceedings of EMNLP*, pages 1532–1543, 2014.

[Read *et al.*, 2012] Jonathon Read, Rebecca Dridan, Stephan Oepen, and Lars Jørgen Solberg. Sentence boundary detection: A long solved problem? In *Proceedings of COLING 2012: Posters*, pages 985–994, December 2012.

[Reynar and Ratnaparkhi, 1997] Jeffrey C. Reynar and Adwait Ratnaparkhi. A maximum entropy approach to identifying sentence boundaries. In *Proceedings of the Fifth Conference on Applied Natural Language Processing*, pages 16–19, March 1997.

[Savelka *et al.*, 2017] Jaromir Savelka, Matthias Grabmair, and Kevin D. Ashley. Sentence boundary detection in adjudicatory decisions in the united states. *Traitement Automatique des Langues*, 58(2):21–45, 2017.

[Schuster and Paliwal, 1997] M. Schuster and K.K. Paliwal. Bidirectional recurrent neural networks. *Trans. Sig. Proc.*, 45(11):2673–2681, November 1997.

[Stamatatos and Fakotakis, 1999] Efstathios Stamatatos and N Fakotakis. Automatic extraction of rules for sentence boundary disambiguation. In *Proceedings of the Workshop on Machine Learning in Human Language Technology, Advance Course in Artificial Intelligence (ACAI99)*, pages 88–92, 1999.

[Tjong Kim Sang and De Meulder, 2003] Erik F. Tjong Kim Sang and Fien De Meulder. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147, 2003.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems 30*, pages 5998–6008. 2017.

[Wu *et al.*, 2016] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, and et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144, 2016.