Efficient Deep Learning-based Sentence Boundary Detection in Legal Text

Reshma Sheik and **Gokul T. Adethya** and **Dr. S. Jaya Nirmala**
Department of Computer Science and Engineering
National Institute of Technology, Tiruchirappalli, Tamil Nadu, India
rezmasheik@gmail.com

Abstract

A key component of the Natural Language Processing (NLP) pipeline is Sentence Boundary Detection (SBD). Erroneous SBD could affect other processing steps and reduce performance. A few criteria based on punctuation and capitalization are necessary to identify sentence borders in well-defined corpora. However, due to several grammatical ambiguities, the complex structure of legal data poses difficulties for SBD. In this paper, we have trained a neural network framework for identifying the end of the sentence in legal text. We used several state-of-the-art deep learning models, analyzed their performance, and identified that Convolutional Neural Network(CNN) outperformed other deep learning frameworks. We compared the results with rule-based, statistical, and transformer-based frameworks. The best neural network model outscored the popular rule-based framework with an improvement of 8% in the F1 score. Although domain-specific statistical models have slightly improved performance, the trained CNN is 80 times faster in run-time and doesn't require much feature engineering. Furthermore, after extensive pretraining, the transformer models fall short in overall performance compared to the best deep learning model.

1 Introduction

From linguistic theory, a sentence is a textual segment or span of one or more grammatically correct words representing a complete thought. Declarative statements, questions, exclamations, requests, commands, and suggestions can all be expressed in sentences. A grammatical subject and grammatical predicate are typically present in an expressive sentence. The person, place, or object (including abstract concepts) that the sentence is about is the grammatical subject, which is typically a noun phrase, and a verb phrase serves as the grammatical predicate(Savelka et al., 2017).

It is quite simple for a person to separate a given text into sentences. However, one realizes how difficult this task is when attempting to condense this segmentation into rules that a machine could follow. Many Natural Language Processing (NLP) applications, including part-of-speech taggers, named entity recognition, document indexing, and question-answering, use sentence boundary detection as a crucial pre-processing step. The pre-processing requirements depend on both the nature of the corpus and the NLP application. Thus SBD faces challenges when working with a specialized corpus like legal text because it offers issues distinguishing between citations, abbreviations, and law-specific keywords.

Existing SBD systems work well for generic corpora but pose serious issues when dealing with Legal Domain. The language, structure, and content are very different and more challenging to understand. When working with SBD, the fundamental presumptions used with the generic corpora are not always applicable. Legal documents typically consist of smaller components like paragraphs, sentences, etc. Sentences can be lengthy and contain intricate structures like lists. There is no standard formatting structure or style for legal documents, even those of the same category (such as statutes or judgments). A sentence segmentation system must resolve these issues to create pure sentences from the mixture of such diverse textual elements.

One main difficulty in SBD is to pinpoint potential boundary spots(e.g. "."). The uncertainty of the delimiter period "." symbol, which has multiple purposes in the legal domain, makes it challenging. It could denote the end of a sentence, an acronym, an initialism, a numerical number (Grefenstette and Tapanainen, 1994), or part of a citation in legal text. To determine if a punctuation character is actually a sentence end marker, a sentence boundary detection system must to resolve the issue of using ambiguous punctuation characters.

Proceedings of the Natural Legal Language Processing Workshop 2022, pages 208 - 217

With this motivation of identifying the potential end-of-sentence marker, we build a deep-learning architectural framework using the context at the character level surrounding the period as input.

The main contribution of this paper is that we use a context window-based deep learning framework for sentence boundary detection in legal text. We compared various deep learning models, identified the best framework, and compared the models with the existing state-of-the-art rule-based and statistical models. We also compared the deep learning model with various transformer architectures like LEGAL-BERT(Chalkidis et al., 2020) and XL-Net(Yang et al., 2019).

The rest of the paper is organized as follows: Section 2 provides an existing literature summary of earlier studies about sentence boundary detection. Section 3 gives a brief description of the dataset and its associated pre-processing. We describe our methodologies and the suggested neural network architecture in Section 4. Section 5 covers results with evaluation techniques and performance comparisons. Finally, the paper concludes with some future steps in Section 6.

2 Related Work

Sentence Boundary Detection in a normal English text is regarded as an answered problem with robust methods. In the NLP literature, several methods for identifying sentence borders have been studied, encompassing algorithms and models ranging from rule-based, statistical, and machine learning-based models. Several techniques for recognizing sentence boundaries across various corpora are presented in the seminal paper(Read et al., 2012).

Decision tree algorithm, Support Vector Machines (SVM)(Gillick, 2009), Bayesian networks, and unsupervised methods like Punkt(Kiss and Strunk, 2006) are among the various algorithms for generic SBD. The identification of sentence boundaries in speech transcriptions is crucial for enhancing readability and supporting later language processing modules. In (Liu et al., 2005), prosodic and textual information sources are used to identify speech sentence boundaries, and (Donabauer et al., 2021) detect sentence boundaries and speaker changes in the unpunctuated text. Recently a well-known rule-based model called Pragmatic Sentence Boundary Disambiguation(pySBD)(Sadvilkar and Neumann, 2020) with more than 98 percent test coverage was developed as an open-source package. PySBD supports 22 languages and is robust in noisy text and domains. We have used pySBD as our baseline model for comparison. In general, these algorithms work well for processing text that adheres to the rules of normal English but operates poorly in other areas. To yield acceptable findings in fields like medical(Le et al., 2021), scientific(Miah et al., 2022), legal, and finance areas(Au et al., 2020), the algorithms used are heavily customized.

The main issues with current Legal SBD systems are a lack of compliance with known sentence patterns, sentence length, and the use of punctuation, particularly periods as non-sentence ending characters. Legal experts use "linguistic indications, structure, and semantic interpretations which interact with domain knowledge" (Wyner and Peters, 2011). (Savelka et al., 2017) examines the use of conditional random fields (CRF) models (Lafferty et al., 2001) for sentence boundary identification in legal documents since these models are frequently used for sequence modeling, or assigning labels for the items in the connected input sequence. Here they developed many CRF models using basic textual features. They employed an aggressive tokenization technique that divides the text into more tokens than normal. The tokens are represented using simple features like the length of the token, whether it is a digit or space, or is written in upper-case or lower-case, etc. A token's features are a combination of its characteristics and features obtained from nearby tokens. For the final model's training, these corresponding feature extractors were utilized, which led to an increase in the inference time of the models. Later (Sanchez, 2019) examined the same dataset with Punkt, CRF, and Bidirectional LSTM neural network architecture. They enhanced the performance of the Punkt model by training it with an updated abbreviation set based on the legal text domain. In neural network architecture, the sentence token is represented by a concatenation of word2vec(Mikolov et al., 2013) embeddings using a three-word window and eight features fed as input to the stacked BiLSTM network with a softmax layer as output. They concluded that this token classification architecture did not result in an SBD that was superior to the CRF model.

The issue of detecting sentence boundaries can be viewed as a classification problem. So our work is inspired by (Schweter and Ahmed, 2019) which

Decisions	#Docs	#Chars	#Tokens	#Sentences	#Average Tokens/Sentence
CC	20	984,756	367,740	8295	19.82
IP	20	932,133	343,831	7,262	21.42
BVA	20	474,478	170,166	3,727	20.73
SC	20	960,890	31,872	602	24.20
Total	80	3,352,257	1,237,414	26,052	21.54

Table 1: Statistics of the dataset

Delimiter	#Occurences	#Occurrence as EOS
.	45048	17835
:	1221	287
!	28	5
;	2453	18

Table 2: Delimiter and its occurences in the dataset

builds an end-to-end methodology independent of the effectiveness of any tokenization technique to make the classification. They developed a general-purpose framework for identifying the potential end-of-sentence markers that can be adapted to multi-lingual benchmarks for 12 distinct languages which work on zero-shot scenarios resulting in building a robust, language-independent SBD.

In addition to processing English legal texts, (Glaser et al., 2021) used CRF and neural network architectures to find the sentence boundaries in German legal documents. They produced and released a dataset of numerous German legal papers with annotations. However, none of the previous literature has applied transformer-based pre-trained language models for the SBD in the legal domain at the context level and used domain-specific transformer models like LEGAL-BERT.

3 Dataset

The algorithm was trained using a dataset (Savelka et al., 2017) of 80 court decisions in four different domains: Cyber Crime (CC), Intellectual Properties (IP), Board of Veterans (BVA), and the United States Supreme Court (SC). These decisions were put in four JSON files of 20 decisions each, along with the list of offsets that denotes the sentence boundaries. The complete dataset of 26052 annotated sentences is publicly available. [1] The summary of the statistics of the four decision sets is specified in Table 1.

The dataset composition of the delimiter occurrences is shown in Table 2. We focused on identifying the period as a potential end-of-sentence (EOS) marker even though the method of SBD can be used for various delimiters. This is because, compared to other delimiters in legal text, the period symbol frequently appears as the EOS markers (98.3%), and the dataset is insufficient to train deep learning models for other delimiters. Furthermore, only 40% of the period's occurrences in the dataset are identified as true boundary delimiters, making it challenging to classify.

3.1 Data Preprocessing

All models were trained using BVA, IP, and SC decisions and tested using CC decisions. The protocol outlined in (Sanchez, 2019) served as the foundation for the manual sentence demarcation for the test file. The sentence in the files was extracted using offset boundaries, and the start and end words were given the labels *BEGIN* and *END* based on the positions provided in the dataset. This annotation is required for comparison with the baseline frameworks. For the deep learning model architecture, the character level context window is taken and retained as input to the model after the period (delimiter) symbol has been located inside the files. For the transformer architecture, we extract the context at the sub-word level.

4 Model Architecture

The architecture of the deep learning framework is depicted in Fig. 1. This context window-based model architecture is used for training sequence classification neural network models. Once the file identifies the delimiter, the corresponding left and right contexts with efficient window size are

[1] https://github.com/jsavelka/sbd_adjudicatory_dec

Raw chunk	*Left Context*	*Delimiter*	*Right Context*	Input Chunk
12d 95. The r	12d 95	.	The r	12d 95 The r

Table 3: Example of the input

taken and fed as input to the embedding layer. The input embeddings from the embedding layer are provided to the deep learning model framework and bypassed to an optional attention layer. The last vector output from this framework is fed into the dense layer with sigmoid activation. Thus, this architecture serves as a binary classification model to determine if the period denotes the end of the sentence or not.

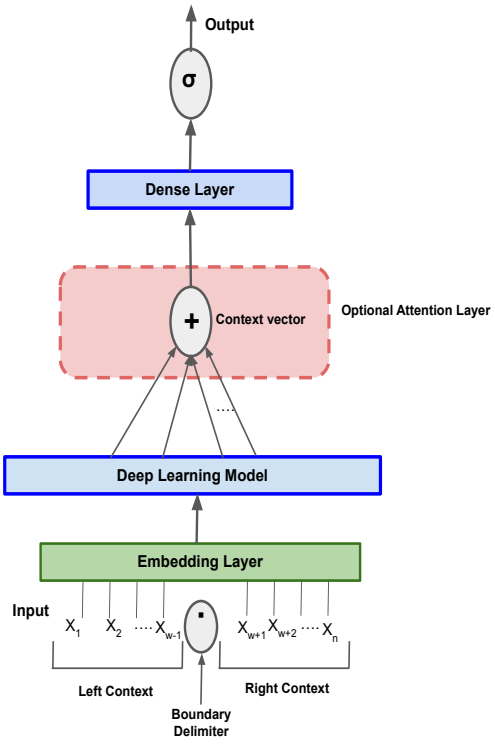

Figure 1: Model Architecture

4.1 Deep Learning Models

Here we used five different architectures of neural networks: Long Short Term Memory(LSTM)(Gers et al., 2000), Gated Recurrent Unit(GRU)(Chung et al., 2014), Bidirectional LSTM(BiLSTM), Bidirectional GRU(BiGRU), and Convolutional Neural Network(CNN). BiLSTM and BiGRU neural network models were also trained to incorporate the attention mechanisms(Bahdanau et al., 2014) to attend to and focus on key parts in input sentences. Each of these models captures information data at the character level. Given the context of surrounding characters, our models identify likely end-of-sentence markers. Our model takes the concatenation of this left and right context excluding the delimiter. This fixed-size context window is fed as input to the model. Table 3 shows an example of the input along with the retrieved left and right contexts.

4.1.1 LSTM

We employ a typical 128-embedding size LSTM network with a hidden size of 256. A dropout probability of 0.2 is used at the hidden layer. The final output vector from the LSTM is fed to the dense layer with a sigmoid activation to classify the output.

4.1.2 BiLSTM

To provide more effect to context, here we used a Bidirectional LSTM architecture that processes input text sequences in the forward and backward directions thus making input size to 512 when feeding to subsequent dense layer. Other factors used are comparable to the LSTM design.

4.1.3 BiLSTM with Attention

Attention weights are used to incorporate an attention mechanism into the BiLSTM architecture(Lin et al., 2017). In this model, soft alignment scores between each hidden state and the final hidden state of the LSTM will be computed using attention. Thus drawing out global dependencies between the inputs and output using the attention process.

4.1.4 GRU

GRU is a more condensed form of LSTM that uses fewer parameters as there is no explicit memory unit. The GRU uses 256 hidden states and an embedding size of 128. During the training process, We employ dropout with a probability of 0.2 after the hidden layer.

4.1.5 BiGRU

The design of this framework is identical to that of BiLSTM, using GRU in place of LSTM.

4.1.6 BiGRU with Attention

Adding attention to the BiGRU architecture enables the inputs to interact and determine who deserves more attention. These interactions and attention scores are combined to create the outputs.

4.1.7 CNN

We used a 1D convolution layer for the CNN architecture with six filters and a kernel size of five. The output of the convolution filter is concatenated to represent the context after being fed through a global max pooling layer. Before the prediction layer, we apply a 250-dimensional hidden layer with ReLU activation and a learning rate of 0.001. A dropout with a 0.2 probability is used during training.

4.2 Transformer Based Models

In contrast to the deep learning architectures discussed above, the transformer-based encoder model reads the input at the subword level as the models used here are pre-trained using sub-word level tokens as input. Six subwords on the left and right of the delimiter period are extracted and concatenated without the delimiter to provide input to the sequence classification model. In our experiments, we have used the pre-trained language models LEGAL-BERT and XLNet.

4.2.1 LEGAL-BERT

LEGAL-BERT(Chalkidis et al., 2020) is a family of BERT models designed to aid in legal NLP research. There are three options of LEGAL-BERT used in the paper for domain adaptation. They are (i) using the original BERT straight out of the box, (ii) adding extra pre-training on domain-specific corpora, and (iii) pre-train BERT from scratch on domain-specific corpora. Here in our experiments, we have used legal-bert-base-uncased, a model trained from scratch in the legal corpora with a number of output labels fixed as two.

4.2.2 XLNet

In the generalized autoregressive model known as XLNet(Yang et al., 2019), each subsequent token depends on every preceding token. XLNet is "generalized" because it uses a process known as "permutation language modeling" to capture bi-directional context. It overcomes the drawbacks of BERT while integrating the concepts of auto-regressive models and bi-directional context modeling. We have used xlnet-base-cased models for our experiments.

4.3 Experimental Setup

We have used the Torch version '1.12.1+cu113' for implementation and the Hugging Face library [2] for fine-tuning the pre-trained language models. The deep learning models were trained for a maximum of 25 epochs, whereas the transformer models got trained for ten epochs. The optimal number of epochs and the model's training time per epoch are shown in Table 4. When compared to other models, it has been found that the transformer models require around 40 times more training time than CNN models. The training/validation loss, accuracy, and F1-score concerning the number of epochs for the CNN architecture are shown in Fig. 2, 3, and 4, respectively. In our tests, we experimented with a one-side context size ranging from 3 to 10 and observed that the context size of six characters produces a better result in deep learning models. The inefficiencies in the fixed-size context input are padded with extra token embedding(s). Since our models are trained on period as the only delimiter, including them in the input did not show any performance improvement. The addition of an extra input delimiter can be used for extending the same architecture to handle multiple delimiters(Schweter and Ahmed, 2019).

With a learning rate of $1e-3$ and a mini-batch size of 32, all models are trained using averaged stochastic gradient descent algorithm. The optimizer used is the Adam optimizer(Kingma and Ba, 2015) with binary cross entropy as a loss function. For pre-trained models, we used a learning rate of $5e-5$. The code used for experimenting with the models is publicly available. [3]

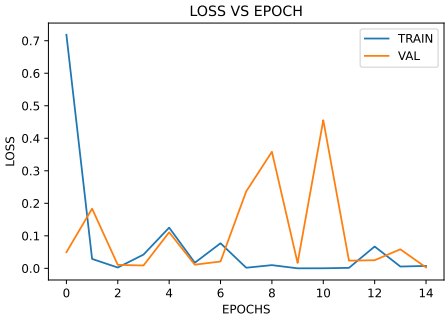

Figure 2: Loss vs Epoch

[2] https://huggingface.co/
[3] https://github.com/NLLP-ML/SBD

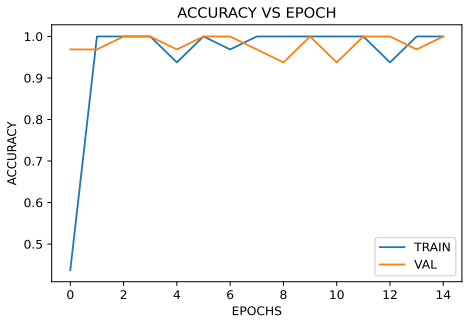

Figure 3: Accuracy vs Epoch

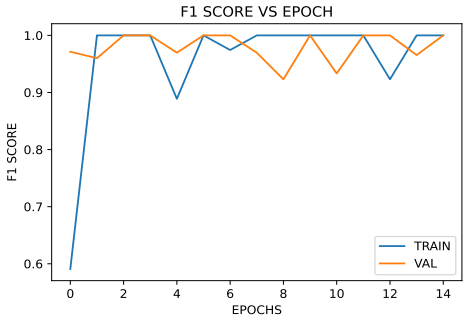

Figure 4: F1 Score vs Epoch

Model	#Epochs	Training time/ epoch (in sec)
LSTM	10	1.88
BiLSTM	10	2.87
BiLSTM + attn	10	2.98
GRU	15	1.86
BiGRU	10	2.86
BiGRU + attn	25	3.06
CNN	15	**1.78**
LEGAL-BERT	10	94.24
XLNet	4	106.61

Table 4: No: of epochs with average training time

Moreover, the number of trainable parameters is also higher for transformer-based models, as shown in Table 5. Thus it can be trained effectively with the help of GPU architectures. The CNN models are the best in model size and number of trainable parameters.

Model	Model size	# Parameters
LSTM	1.55 MB	4,08,449
BiLSTM	3.21 MB	8,03,969
BiLSTM + attn	3.21 MB	8,03,969
GRU	1.18 MB	3,09,633
BiGRU	2.31 MB	6,06,337
BiGRU + attn	2.31 MB	6,06,337
CNN	**116 kB**	**29,275**
LEGAL-BERT	418 MB	110M
XLNet	449 MB	110M

Table 5: Number of trainable parameters

5 Results and Discussion

We organize the results into three subsections. The first two sections focus on evaluation patterns used in our research to compare with the existing state-of-the-art models. In the first section, we compare the deep learning models against the baseline models based on offset boundaries. The second evaluation provides an inference time-based performance analysis of all the models. The final result section covers the performance assessment of the deep learning models to the architecture for the binary classification task.

5.1 Evaluation based on Offset Boundaries

Since the baseline models are evaluated based on offset boundaries, we post-process the results obtained from the deep learning architecture to label the words representing *BEGIN* and *END* tokens. Each document in the test set was sentence tokenized, and the model assigned the predicted labels for the test file.

Table 6 summarizes the results with other baseline models. We calculate the F1 score for each model at the *BEGIN* and *END* token levels. We used the state-of-the-art rule-based pySBD model and statistical Conditional Random Field(CRF) model as baselines. The result shows that the CNN model outperformed other neural network architectures and the pySBD framework. It is also observed that the statistical CRF has a slightly better F1 score than the CNN model. This might be due to

Comparison - Neural Network Models			
Model	***Begin***	***Last***	***Average F1-Score***
LSTM	0.809	0.862	0.8354
BiLSTM	0.805	0.853	0.8292
BiLSTM + attn	0.811	0.859	0.8347
GRU	0.809	0.859	0.8342
BiGRU	0.808	0.855	0.8316
BiGRU + attn	0.803	0.851	0.8267
CNN	0.822	0.871	**0.8464**
LEGAL-BERT	0.827	0.865	0.8462
XLNet	0.801	0.849	0.8247
Comparison - Other Models			
Model	***Begin***	***Last***	***Average F1-Score***
Rule-based pySBD	0.751	0.77	0.761
Statistical -CRF(Sanchez, 2019)	0.894	0.892	0.893

Table 6: Comparison at token level (F1- score)

the CRF models' ability to locate boundary delimiters other than periods in the legal text. However, its performance level depends on how inventively the features were created.

5.1.1 Error Analysis

The errors in SBD identified by the CNN and CRF models had many things in common. Both models found it challenging to identify the characters out of the sentence as in examples 1 and 2 in Table 7. As shown in example 3, the CRF models had difficulty in finding out the true boundary with the delimiter ":", but CNN doesn't have that ability making it weaker than CRF models. In example 4, the best-performed CNN model could not capture sentences with multiple periods, whereas the CRF models could correctly identify the boundaries. Most of the citations within sentences are not properly handled by the baseline CRF models and are considered as separate sentences (Sanchez, 2019) as shown in example 5.

5.2 Comparison based on Inference Time

The run-time of the models for the exact hardware specification is shown in Table 8. It is evident that the CNN model has the fastest inference time and that of CRF models, with inference time 84 times longer than CNN models. The transformer models LEGAL-BERT and XLNet have the highest inference times of 112 and 113 seconds, respectively.

5.3 Comparison based on the Model Architecture

The results of the context window-based deep learning models are shown in Table 9. Here the performance of nine deep learning frameworks to correctly identify the end of sentence boundary in the legal text was showcased based on accuracy, precision, recall, and F1 score. The table makes it clear that CNN performed the best among others. Employing an attention mechanism to the LSTM/GRU architecture doesn't help in improving performance. The outcomes of the transformer models were not improved even after intensive pretraining. We have also observed that domain-specific LEGAL-BERT performed better in the F1 score when compared to the generic XLNet model. In light of the model size and runtime, the CNN models performed well.

Overall, we found that CNN outperformed the pySBD model and produced the best results among the deep-learning models. The best neural network model outperformed the popular rule-based framework by 8% in terms of the F1 score. In contrast to statistical models, the deep learning model's inference time is 84 times shorter. It is also possible to parallelize the SBD task at runtime by using batch processing in the proposed neural network architecture. LEGAL-BERT performs very close to the CNN model in the F1 score. Despite having scores that are on par with the CNN model, the domain-specific LEGAL-BERT models might be difficult

Example 1	See, e.g., Cal. Family Code Ann. §760 (West 2004). **Sentence 1:** 4 See, e.g., Cal. **Sentence 2:** Family Code Ann. §760 (West 2004).
Example 2	In the first case, petitioner David Riley was stopped by a police officer for driving with expired registration tags. I A In the first case, petitioner David Riley was stopped by a police officer for driving with expired registration tags.
Example 3	Decided: November 26, 2002. **Sentence 1:** Decided: **Sentence 2:** November 26, 2002.
Example 4	Id., at 180. . . .” Id., at 180.
Example 5	Franklin also moved to dismiss eleven of the fourteen copyright infringement counts on the ground that Apple failed to comply with the procedural requirements for suit under 17 U. S. C. § § 410, 411. < 714 F. 2 d 1245 >. **Sentence 1:** Franklin also moved to dismiss eleven of the fourteen copyright infringement counts on the ground that Apple failed to comply with the procedural requirements for suit under 17 U. S. C. § § 410, 411. **Sentence 2:** < 714 F. 2 d 1245 >.

Table 7: Errors in SBD: The actual sentence in the text is marked in grey, while the predicted sentence is marked in red.

Model	Runtime(in sec)
LSTM	0.85
BiLSTM	1.75
BiLSTM + attn	1.71
GRU	0.62
BiGRU	1.19
BiGRU + attn	1.31
CNN	**0.16**
LEGAL-BERT	112.86
XLNet	113.07
pySBD	5.50
CRF	13.41

Table 8: Inference time of models

to implement because of their high memory requirements and slow speeds. The CNN models display impressive performance given the model size, training, and testing times. As a result, CNN with a small number of trainable parameters outperformed huge models.

6 Conclusion

This paper uses a context window-based deep learning model framework for efficient sentence boundary detection in legal text. We compared various deep learning models, including transformers, for analysis. We showed that CNN showed a better performance when compared to other deep learning models. This model also outperformed the popular rule-based pySBD framework. Even though the statistical model has a minor performance improvement, the trained CNN had a decent performance without the requirement of exhaustive feature engineering compared to domain-specific CRF models. Also, CNN is faster than the state-of-the-art CRF models by multiple folds compared to the running time. As a result, the Convolutional Neural Network is the model with the best performance.

In the future, we plan to broaden the scope of our architecture by including the different delimiters found in legal text. Also, we aim to chain multiple models together to improve the SBD performance. However, we could demonstrate that our model had an excellent performance and could thus be incorporated into NLP pipelines for various downstream legal tasks.

Limitations

The major limitation addressed in this paper is the choice of delimiter used in the deep learning architecture. Here we have only used the period “.” as the potential end of sentence marker in the legal text. We could explore more sentence ending punctuation’s like colons “:”, exclamation “!”, etc., to the architecture and thereby improve the results. In

Model	Accuracy	Precision	Recall	F1-Score
LSTM	0.968	0.974	0.953	0.963
BiLSTM	0.964	0.970	0.945	0.957
BiLSTM + attn	0.967	0.971	0.952	0.961
GRU	0.966	0.973	0.947	0.960
BiGRU	0.967	0.969	0.951	0.960
BiGRU + attn	0.964	0.955	0.954	0.955
CNN	**0.981**	0.976	0.978	**0.977**
Legal-BERT	0.979	0.975	0.977	0.976
XLNet	0.970	0.939	0.993	0.965

Table 9: Results of the classification performance of the models

contrast to the pre-trained transformers, which are trained using sub-word level embedding, we have analyzed the performance of deep learning models using character-level embedding. The requirement of considerable GPU resources is another limitation of transformer-based models. More training data is also necessary for deep learning models to produce better results. These limitations can be future opportunities to facilitate further research.

Ethics Statement

Our work contributes to implementing deep learning models for sentence boundary detection in legal text. The dataset used in this paper is publicly available for research, and we have appropriately cited it in the article. The code we implemented is made open to facilitate future research.

References

Willy Au, Bianca Chong, Abderrahim Ait Azzi, and Dialekti Valsamou-Stanislawski. 2020. Finsbd-2020: The 2nd shared task on sentence boundary detection in unstructured text in the financial domain. In *Proceedings of the Second Workshop on Financial Technology and Natural Language Processing*, pages 47–54.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. 2020. Legal-bert: The muppets straight out of law school. *arXiv preprint arXiv:2010.02559*.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Gregor Donabauer, Udo Kruschwitz, and David Corney. 2021. Making sense of subtitles: Sentence boundary detection and speaker change detection in unpunctuated texts. In *Companion Proceedings of the Web Conference 2021*, pages 357–362.

Felix A Gers, Jürgen Schmidhuber, and Fred Cummins. 2000. Learning to forget: Continual prediction with lstm. *Neural computation*, 12(10):2451–2471.

Dan Gillick. 2009. Sentence boundary detection and the problem with the us. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pages 241–244.

Ingo Glaser, Sebastian Moser, and Florian Matthes. 2021. Sentence boundary detection in german legal documents. In *ICAART (2)*, pages 812–821.

Gregory Grefenstette and Pasi Tapanainen. 1994. What is a word, what is a sentence? problems of tokenization.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Tibor Kiss and Jan Strunk. 2006. Unsupervised multilingual sentence boundary detection. *Computational linguistics*, 32(4):485–525.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Daniel X Le, James G Mork, and Sameer Antani. 2021. Hybrid ensemble-rule algorithm for improved medline® sentence boundary detection. In *AMIA Annual Symposium Proceedings*, volume 2021, page 677. American Medical Informatics Association.

Zhouhan Lin, Minwei Feng, Cicero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. *arXiv preprint arXiv:1703.03130*.

Yang Liu, Andreas Stolcke, Elizabeth Shriberg, and Mary Harper. 2005. Using conditional random fields for sentence boundary detection in speech. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 451–458, Ann Arbor, Michigan. Association for Computational Linguistics.

Md Miah, Saef Ullah, Junaida Sulaiman, Talha Bin Sarwar, Ateeqa Naseer, Fasiha Ashraf, Kamal Zuhairi Zamli, and Rajan Jose. 2022. Sentence boundary extraction from scientific literature of electric double layer capacitor domain: Tools and techniques. *Applied Sciences*, 12(3):1352.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

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

Nipun Sadvilkar and Mark Neumann. 2020. Pysbd: Pragmatic sentence boundary disambiguation. *arXiv preprint arXiv:2010.09657*.

George Sanchez. 2019. Sentence boundary detection in legal text. In *Proceedings of the natural legal language processing workshop 2019*, pages 31–38.

Jaromir Savelka, Vern R Walker, Matthias Grabmair, and Kevin D Ashley. 2017. Sentence boundary detection in adjudicatory decisions in the united states. *Traitement automatique des langues*, 58:21.

Stefan Schweter and Sajawel Ahmed. 2019. Deep-eos: General-purpose neural networks for sentence boundary detection. In *KONVENS*.

Adam Wyner and Wim Peters. 2011. On rule extraction from regulations. In *Legal Knowledge and Information Systems*, pages 113–122. IOS Press.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.