Party Extraction from Legal Contract Using Contextualized Span Representations of Parties

Sanjeepan Sivapiran, Charangan Vasantharajan, Uthayasanker Thayasivam
Department of Computer Science and Engineering, University of Moratuwa
Colombo, Sri Lanka
{sanjeepan.18, charangan.18, rtuthaya}@cse.mrt.ac.lk

Abstract

Extracting legal entities from legal documents, particularly legal parties in contract documents, poses a significant challenge for legal assistive software. Many existing party extraction systems tend to generate numerous false positives due to the complex structure of the legal text. In this study, we present a novel and accurate method for extracting parties from legal contract documents by leveraging contextual span representations. To facilitate our approach, we have curated a large-scale dataset comprising 1000 contract documents with party annotations. Our method incorporates several enhancements to the SQuAD 2.0 question-answering system, specifically tailored to handle the intricate nature of the legal text. These enhancements include modifications to the activation function, an increased number of encoder layers, and the addition of normalization and dropout layers stacked on top of the output encoder layer. Baseline experiments reveal that our model, fine-tuned on our dataset, outperforms the current state-of-the-art model. Furthermore, we explore various combinations of the aforementioned techniques to further enhance the accuracy of our method. By employing a hybrid approach that combines 24 encoder layers with normalization and dropout layers, we achieve the best results, exhibiting an exact match score of **0.942** (**+6.2%** improvement).

1 Introduction

Extracting legal entities from legal documents is an essential challenge for legal assistive software (Leivaditi et al., 2020). Its goal is to extract structured information — "what are the legal attributes (agreement date, party, license, etc) that are involved" — from unstructured text. A contract document is a legally binding agreement between two or more parties. It outlines the terms and conditions of the relationship and sets forth the rights and obligations of each party (Chalkidis et al., 2017). Here, the party is a person or entity who takes part in a legal transaction, for example, a person with an immediate interest in an agreement or deed, or a plaintiff or a defendant in a lawsuit. A "third party" is a person who is a stranger to a transaction, contract, or proceeding.

Extracting parties' information from legal contracts can provide numerous benefits to legal assistant software such as Concord[1], ContractWorks[2], and HelloSign[3]. First and foremost, this can aid legal professionals in identifying parties involved in similar cases, trends, and patterns in legal disputes, and assist in more efficient legal research. Secondly, the extraction of parties allows reviewing multiple documents efficiently with the quick document search feature, and it leads to focusing on relevant information and documents through assistant software. Moreover, the manual extraction of parties is prone to human errors, which can lead to inaccuracies and inconsistencies (Hendrycks et al., 2021). By automating the process of party extraction, legal professionals can save time and improve the accuracy of their work.

Extracting parties can be a challenging task despite its usefulness (Leivaditi et al., 2020). One of the most challenging parts of the contract is it may contain numerous names of persons and organizations throughout its pages. These names can refer to various entities other than the parties, such as third-party beneficiaries, agents, assignees, guarantors, witnesses, and experts (Bommarito et al., 2018). As a result, it can be difficult to distinguish the parties from all the other types of individuals and entities mentioned in the contract. Secondly, legal contracts can be lengthy and complex, with various technical terms and clauses that require in-depth legal knowledge to understand fully. Ad-

[1] https://www.concordnow.com
[2] https://www.contractworks.com
[3] https://www.hellosign.com

ditionally, contracts can be written in a convoluted manner, leading to ambiguity in determining the parties' intent, which can create difficulties in extracting the relevant information accurately. Moreover, reviewing a large volume of contracts within a limited time frame can be a labor-intensive task for lawyers, often leading to errors and inconsistencies. Finally, the manual review may also be prone to errors due to human oversight or misinterpretation, which can result in inaccuracies in the extracted information (Hendrycks et al., 2021). Therefore, automated legal assistant software can be beneficial in addressing this challenge.

There has been extensive research on extracting various information from legal documents (legislation, court cases, contracts) (Almeida et al., 2020; Hendrycks et al., 2021), but there were only three studies found on extracting parties from legal contracts. The first system used a rule-based system to extract parties (Chalkidis et al., 2017), but it cannot scale out since law firms generate a plethora of contract documents, making it impossible to add processing rules for each new contract type. The second attempt focused on only one type of contract (leases) and was not applicable to other types of contracts (Leivaditi et al., 2020). The third and final system solved this problem using a question-answering method (Hendrycks et al., 2021). However, the extraction system results in a large number of false positives and less reliable outcomes.

In addition, there exist only two datasets that facilitate the extraction of parties' information from legal contract documents (Chalkidis et al., 2017; Hendrycks et al., 2021), out of which only one is publicly accessible. The initial dataset is annotated with extraction zones that are appropriate for rule-based extraction (Chalkidis et al., 2017). On the other hand, the second dataset (CUAD) is annotated for a question-answering system but is unsuitable for precise party detection (Hendrycks et al., 2021).

In this research, we try to develop a relatively accurate dataset with a precise match of the parties and doubled the size of the CUAD dataset (**Contribution 1**). We attempt to develop a scalable and accurate party extraction system with minimal overheads. To address this, we use RoBERTa (a pre-trained transformer-based language model (Vaswani et al., 2017)) (Liu et al., 2019) as the baseline model and empirically optimized this architecture to best learning capability and improve contextualized representations of the contracts in the parties extraction task. This optimized architecture includes transformer encoder layers, layer normalization, and dropout layers (**Contribution 2**).

The remaining sections of the paper are organized as follows: Firstly, in Section 2, we conduct a comprehensive literature review focusing on party extraction systems and party annotated datasets. Subsequently, in Section 3, we outline our proposed dataflow and the techniques applied to RoBERTa. We then present the experiments in Section 4, which encompasses the dataset description, environmental setup, and evaluation methodology. Moving forward, in Section 5, we present the experimental findings and compare the accuracy of our approach with different models derived from various techniques. Following that, in Section 6, we discuss the limitations of our approach and provide insights for future works. Finally, we conclude the paper with remarks in Section 7.

2 Related Work

Party extraction is currently being explored through various efforts. The primary methods used for text extraction are rule-based, machine learning, and transformers. However, to the best of our knowledge, only three researchers have attempted to extract parties from legal contracts, and only two have developed datasets for parties. This literature first examines previous research on dataset creation and highlights its advantages and limitations. Then, it delves into the details of the three party extraction systems.

2.1 Party Annotated Datasets

All of the available party extraction datasets for legal contract documents are annotated for both the parties involved in the contract and additional elements from the contract structure. (Wang, 2022).

Chalkidis et al. (2017) introduced a labeled dataset with gold contract element annotations. The contract elements they aimed to extract are Contract Title, Contracting Parties, Start Date, Effective Date, Termination Date, Contract Period, Contract Value, Government Law, Jurisdiction, Legislation Refs, and Clause Headings. They looked for the parties on the cover page and preamble (hereafter extraction zone) during the annotation to reduce the time needed to process the contracts. In practice, it would increase the false positives (tokens wrongly identified as contracting parties) during testing.

In the following year, there was another dataset released for contract review with more contract elements (Hendrycks et al., 2021). They chose a list of 41 label categories that lawyers pay particular attention to when reviewing a contract. The labels are broadly divided into the following three categories: General information, Restrictive covenants, and Revenue risks. They used the cover page, preamble, contract's first page, and signature part for the annotation of parties. Additionally, they also annotated the roles of the extracted parties. Since, their annotations include irrelevant terms, abbreviations, and sentences as parties, the quality of the dataset is quite low for the parties extraction.

2.2 Party Extraction System

In the view of party extraction systems, there were only two contracting parties extraction systems available.

Firstly, Chalkidis et al. (2017) experimentally compared several contract element extraction methods that used manually written rules and linear classifier (Logistic Regression, Support Vector Machine (SVM)) with hand-crafted features, word embeddings, and part-of-speech tag embeddings. Among their experiments, the linear classifier (SVM) performed best when both the hand-crafted features and the word and POS tag embeddings were used. In another experiment, manually written post-processing rules significantly improved the performance of the linear classifiers, leading to the same overall results for both LR and SVM, outperforming the rule-based system and the generic Named Entity Recognition. The F1 score of the two best systems was 0.89 in the extraction of parties. As we described in the section 2.1, identifying parties from extraction zones increase the false positive during testing. Therefore, the authors first identify the extraction zone using regular expressions and classify the tokens as contracting parties. This approach is unsuitable due to the generation of new types of contract documents by law firms, making it impossible to add expression rules for each new contract type.

In the following year, Hendrycks et al. (2021) conducted an experiment on a legal question-answering dataset using various generic models such as BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), ALBERT (Lan et al., 2020), and DeBERTa (He et al., 2021b). Their findings revealed that DeBERTa outperformed the other models, with almost 95% AUPR measure. However, most of the professional systems do not intend to use DeBERTa, a large language model which requires a high-computing resource to train and make predictions (Hendrycks et al., 2021). As we mentioned the section 2.1, even though their dataset quality is low for the parties extraction, the system achieved nearly 95% due to their customized evaluation algorithm. This algorithm used 0.5 as a threshold to find the overlap between extracted and actual strings to be counted as a valid match. Therefore, most of the sub-strings of the actual parties will be counted as valid matches. This leads to an increase in the evaluation metrics.

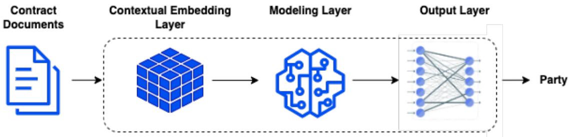

Figure 1: Overall System.

3 Methodology

The aim of this research is to create a question-answering model (Wang, 2022) to identify parties involved in legal contracts as shown in Figure 1. To accomplish this, we have opted to use RoBERTa (Liu et al., 2019) as our foundational model. There were several reasons behind our selection of RoBERTa (Liu et al., 2019) for this research. Firstly, it is a comparatively smaller model compared to other large language models (Devlin et al., 2019; He et al., 2021a). Secondly, RoBERTa (Liu et al., 2019) is equipped with a **Fast Tokenizer**, enabling us to process large volumes of text data quickly and efficiently. Lastly, this model has been shown to have performance improvements compared to BERT (Devlin et al., 2019) due to its dynamic masking technique.

3.1 Dataflow

In this section, we will look at how raw text inputs are processed by the model during fine-tuning. First, newly annotated documents are fed into the model. These documents are often large and lengthy, so a sliding window approach is used to divide them into smaller chunks of 512 tokens each as illustrated in Figure 2. Each token can be one or more characters long. (examples of tokens: ”as”, ”law”, and ”agree”).

Next, the inputs were tokenized using a *fast to-*

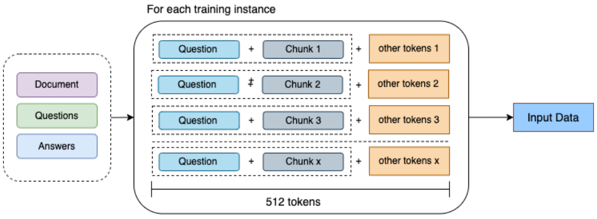

Figure 2: Chunks Creation. The document is divided into 512-token chunks, and each chunk is combined with a question to create a question-context pair. The pair also contains a flag that indicates whether the answer to the question is present in the chunk.

kenizer, which replaced each token with a unique integer identifier, or token ID. The vocabulary used for this step consisted of approximately 50,000 tokens, which were derived from *WordPiece* tokenization. *WordPiece* tokenization breaks down words into subwords or pieces, and the vocabulary includes these subwords, as well as some unique tokens used for specific purposes, such as *[CLS]* (classification), *[SEP]* (separator), and *[MASK]*.

In addition to the token IDs, positional embeddings were also created to indicate the relative positioning of the tokens between each other. This information is important for understanding the context of the text. Segment embeddings were also created to differentiate the question from the context. This information is important for tasks such as question answering, where the model needs to know which tokens belong to the question and which tokens belong to the context.

In the next step, the three embeddings were combined to create the input embeddings for the first encoder block. The encoder then uses multi-head attention to learn contextual information from the inputs. This is an important aspect, unlike RNN (Sherstinsky, 2020) or LSTM (Hochreiter and Schmidhuber, 1997), BERT-based (Devlin et al., 2019) models like RoBERTa (Liu et al., 2019), as it allows them to learn and memorize from neighboring sentences with accurate contextual information.

Each encoder block in a BERT-based model learns from the inputs and transfers the learned information to the next encoder block. Once the final encoder block is completed, the model's output is transformed into answer token logits using multiple layers. These layers are used during prediction to generate accurate answers to questions (van Aken et al., 2019).

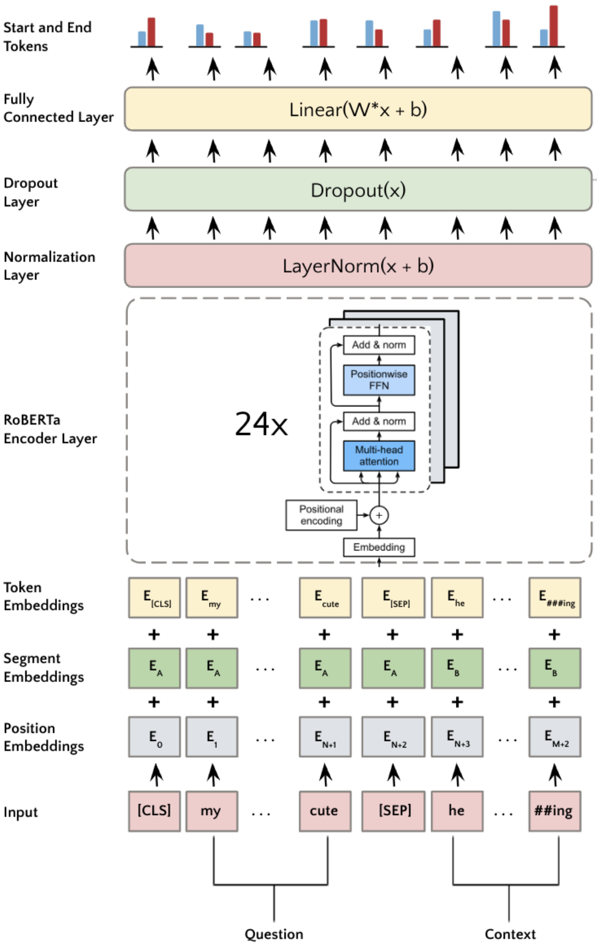

Figure 3: Best Performing Architecture. This architecture consists of 4 layers: (1) RoBERTa encoder layer (24), (2) Normalization layer, (3) Dropout layer, and (4) Fully connected layer.

3.2 Salient Factors

We have implemented several techniques to increase the learning capability of the model to learn complex structures within the legal space. These techniques involve altering the activation function and increasing the encoder layers. Additionally, we have stacked normalization and dropout layers as additional components on top of the output encoder layer. Our best-performing architecture is illustrated in Figure 3.

- **Encoder Layers**: A model's learning ability and performance are greatly influenced by the number of encoder layers it possesses (van Aken et al., 2019). To improve the model's performance, we conducted experiments where we adjusted the number of encoder layers to achieve the desired level of complexity.

Traditionally, when refining language models for extraction tasks, it is common practice to maintain the original quantity of transformer encoder layers in the architecture. However, our findings indicate that incorporating additional encoder layers enhances the transformer's capacity to capture intricate patterns and relationships within the input data. This expanded capacity enables the model to learn more detailed representations, improving performance.

Also, increasing the number of layers enables the model to capture information at various levels of granularity. Lower layers tend to focus on capturing local and syntactic legal information, such as word order and sentence structure, while subsequent layers have the ability to learn more abstract and semantic concepts. This hierarchical representation allows the model to understand both fine-grained details and the broader context of legal contracts, contributing to its overall effectiveness in capturing meaningful information from the input documents.

- **Activation Function**: The choice of activation function can impact the learning capacity, convergence speed, and generalization ability of the RoBERTa (Liu et al., 2019) model. We have experimented with different activation functions (Agarap, 2018; Hendrycks and Gimpel, 2020) during the encoder layer computation to identify the most suitable one for our model. By replacing the GELU (Dan and Kevin, 2016) activation function with the New GELU function, we have observed improvements in the model's performance.

- **Layer Normalization**: Layer normalization is a method that normalizes the inputs within each layer (Ba et al., 2016), making the training process faster. It reduces reliance on the activation scale and focuses on relative differences between activations, making the model more resilient to changes in input scale and improving generalization. We incorporated layer normalization into the output layer of the model, resulting in better overall performance.

- **Dropout**: Overfitting is a common issue in deep learning models, and dropout (Srivastava et al., 2014) is a regularization technique that can help address this problem. We have introduced a dropout of 0.2 in our model to mitigate overfitting issues that may arise due to increased model encoder layers from the above modifications. Our experiments have shown that dropout has effectively reduced overfitting.

4 Experiments

This section outlines our proposed solution for identifying parties mentioned in contract documents, which we approached as a question-answering task. Our methodology for training and evaluating the model is described, along with the results from experiments using various configurations.

4.1 Dataset

There is only one dataset available for party extraction which is **Contract Understanding Atticus Dataset** (CUAD) (Hendrycks et al., 2021). It contains 510 agreements of 25 different types, collected from the Electronic Data Gathering, Analysis, and Retrieval (EDGAR) system. Even though the dataset has been useful in extracting parties and clauses from legal documents, it has several shortcomings that limit its usability for our study.

One of the main shortcomings of CUAD is that the annotations include irrelevant terms such as *We*, *You*, and *Us*. These terms are not parties to the contract and are, therefore, not relevant to the task of extracting the exact match of the parties. Another issue with the annotations is that they include the parties' abbreviations, such as *PCQ* and *ABW*. While these abbreviations may be used in the contract, they are not always immediately recognizable and can be confusing for automated systems attempting to extract the parties. Annotations also include sentences as party names, such as *This agreement shall apply to said ABW and all of its subsidiaries and related companies*. These annotations are problematic because they do not accurately capture the parties to the contract and can lead to incorrect extractions. Furthermore, some parties are captured from the headings, signature part, and other places rather than from the actually mentioned sentences (the contract's first page other than the cover page). This can lead to inconsistencies as parties may have different names in different parts of the contract.

From the above study, we introduce a newly annotated dataset that comprises 1000 legal contract documents collected from CUAD (510 documents) and the EDGAR database (490 documents). This dataset is specifically annotated for accurate party detection along with the solution for the above shortcomings. This collection of contract documents falls into 25 different types (purchase agreements, employment contracts, lease agreements, etc) and have varying lengths that span from just a few pages to over one hundred pages. Figure 5 illustrated the distribution of contract documents' lengths. We then divide the dataset into a training set (2500 annotations across 900 documents) and a test set (253 annotations across 100 documents). We split the dataset randomly with a test set size of 0.1 to ensure both datasets were representative of the overall data distribution.

4.2 Experimental Set Up

We conducted our experiment step by step to identify the correct approach to achieve accurate party extraction with a high exact match. Finally, our experiments are mainly divided into four phases as follows:

1. Evaluate the CUAD's best model (DeBERTa)
2. Evaluate re-annotated and newly annotated dataset
3. Evaluate with different activation functions and layers stacks such as normalization and dropout
4. Evaluate with different numbers of encoder layers

In the first phase, we simply evaluate the trained DeBERTa model (CUAD's best model released by the authors) on our test dataset. This provides us with the initial baseline for our future experiments. In the second phase, we conducted experiments on re-annotated and newly annotated datasets separately to ensure their contribution towards the improvement of the accurate party extraction. After ensuring their positive contributions, then we combined both datasets and execute a new experiment to define a new baseline to do further studies.

We applied several techniques in the third phase to rescale the features (output from the encoder layers), normalize them, and randomly drop some units in the features to prevent overfitting in the training. From this phase, we identified an improvement over our new baseline (experiment D).

Algorithm 1 Evaluation Algorithm.
JS: Jaccard Similarity

```
procedure EXACTMATCH($instances, preds$)
    $ems \leftarrow array()$
    for $inst$ in $instances$: do
        $best \leftarrow 0$
        for $pred$ in $preds$: do
            $score \leftarrow JS(inst.answer, pred)$
            $best \leftarrow \max(score, best)$
        end for
        $ems \leftarrow append(best)$
    end for
    $em \leftarrow average(ems)$
    return $em$
end procedure
```

In the last phase, we considered the experiment with a high exact match from the previous phase for further studies. As we mentioned in the section 3, the ability to learn the complex structure of the legal text matters. Therefore, we explored into increasing the learning of such input space to the model. Finally, we found varying the number of encoder layers significantly improves the learning capability of the model. Then, we conducted additional experiments by altering the number of encoder layers from 12 (in the original RoBERTa) to 8, 16, 24, and 32.

To run all experiments, we used Amazon EC2's *g5.xlarge* instance and optimize the model's performance by tuning hyperparameters including batch size (32), learning rate (1e-04), and number of epochs (10).

4.3 Evaluation Criteria

Our aim is to identify the exact match between parties and to achieve this, we have chosen two key metrics. The first one is Jaccard similarity (Niwattanakul et al., 2013), which measures the similarity between sets and will help us determine how closely the predicted parties align with the actual parties in the test set. The second metric is the exact match, which will simply tell us if the predicted and actual parties are an exact match. By utilizing these two metrics, we can effectively evaluate the performance of our model on the test set and ensure that we are finding the precise match of the parties we are interested in. Algorithm 1 depicts our evaluation method.

Name	Experiment	Exact Match
A	Test on CUAD best model (DeBERTa)	0.887
B	Re-annotated CUAD (510 documents)	0.929
C	Newly annotated documents (490 documents)	0.921
D	**Baseline (B + C)**	**0.934**
E	Baseline + New GELU	0.933
F	Baseline + LayerNorm + Dropout + l=12	0.913
G	Baseline + LayerNorm + Dropout + l=8	0.905
H	Baseline + LayerNorm + Dropout + l=16	0.938
I	**Baseline + LayerNorm + Dropout + l=24**	**0.942**
J	Baseline + LayerNorm + Dropout + l=32	0.905

Table 1: Experimental Results. There are 5 stages of experiments: (1) Evaluation of CUAD's best model (A), (2) Evaluation of re-annotated, newly annotated, and whole dataset (B, C, and D), (3) Different architectural techniques (E, and F), (4) Change in the number of encoder layers (G, H, I, and J).

$$J(U, V) = \frac{|U \cap V|}{|U \cup V|}$$

where J is Jaccard similarity, U is annotated party answer and V is the predicted party from the fine-tuned model.

The results were compared to the ground truth using the Jaccard similarity coefficient, which measures the overlap between the predicted parties and the annotated parties. The coefficient is calculated as the ratio of the intersection of the two sets to the union of the two sets, ranging from 0 to 1, where 0 indicates no overlap and 1 indicates a perfect match. Qualitative experiments determined that a threshold of 0.5 is reasonable for determining the validity of the match.

5 Results and Analysis

Table 1 presents the results of experiments conducted to evaluate the performance of RoBERTa model including several variations of a baseline model, each with different modifications or additions, on the newly annotated dataset. In terms of datasets, the re-annotated CUAD's dataset significantly improved the performance of the system compared to the state-of-the-art DeBERTa model presented by CUAD (from the exact match (EM) of 0.887 to 0.929). This implies that most of the errors identified from the existing dataset have been resolved by our re-annotation process (experiment B). On the other hand, our newly annotated dataset achieved a higher score (0.921) than the current state-of-the-art model (0.887) and slightly less than that of experiment B (0.929), indicating that this data contributed positively to the experiment's performance. This motivates us to combine both datasets from experiments B and C to define a new experiment D as our new baseline.

The baseline experiment (D) achieved an EM of 0.934, which indicates the quality of the combined dataset compared to the existing dataset (CUAD). Even though our baseline model achieved a comparatively higher score, we found some mistakenly identified parties with non-formal forms and partial forms during our error analysis. By Further studies, we concluded that this is due to the inability of the models to learn the complex structure of the legal text. Therefore, we explored additional experiments to increase the learning capability of our model to learn legal text's complex structure.

Finally, we found that passing the output features from the 12th encoder layer of the original RoBERTa through additional layers will increase the further learning of the model on the training dataset. This will help to learn the complex structure of the input space and the association between the different sub-tokens of a party. For example,

- **Complex Structure:** The model (from experiment F) often failed while predicting the parties with some identified complex structures as depicted in Table 2.

- **Association:** There are four sub-tokens in the following party **SQUARE TWO GOLF INC.** such as **SQUARE**, **TWO**, **GOLF** and **INC.**. All of these sub-tokens together need to be identified as a party according to our goal (accurate party extraction).

From the above analysis and studies, we conducted several experiments by varying the number of encoder layers (l) from 8 to 32. But, as you can see in Figure 4, the average exact match is increased from 0.905 (experiment G) to 0.942 along with the number of layers until l=24 (experiment I). During the experiments, we also kept the normalization layer and dropout (0.2) layers on top of the final encoder layer to prevent overfitting. Even after, our model reached a lower exact match (0.905) than that of l=24 while using l=32. This shows our model is getting overfitted even using normalization and dropout layers. Finally, we concluded our experiments and fix our best model from experiment I (l=24). Through our analysis presented in Table 2, we have determined that our model has made substantial progress in comprehending the complex formatting of legal text and the relationships between its sub-tokens, leading to superior predictive capabilities.

6 Discussions

The results of our experiments demonstrate that increasing the number of additional encoder layers indeed leads to improved outcomes. Previous research models often struggled to identify and learn these complex structures when the number of encoder layers was less. By increasing the number of encoder layers in our approach, we were able to address this limitation and achieve significant improvements in exact matches.

The augmentation of encoder layers allowed our model to better capture and represent the nuanced relationships and patterns present within legal contracts. This, in turn, facilitated the identification and extraction of relevant information pertaining to the legal parties involved. The increased depth of the model architecture enabled it to learn and comprehend intricate complexities, which were previously challenging to capture effectively.

Furthermore, We introduced some modifications to the activation function and implemented additional normalization techniques on top of the final encoder layer. These adjustments were designed to complement the increased depth of the model and further enhance its ability to learn complex structures within legal contracts. The combined impact of increasing the encoder layers, replacing the activation function, and incorporating additional normalization techniques proved to be highly effective in our research as indicated in Table 1.

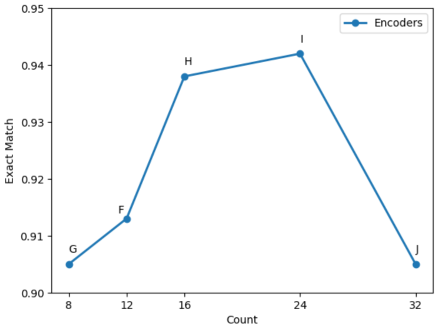

Figure 4: System Performance against the number of encoder layers

One of the main limitations of our model is the potential loss of context during the chunking process of input documents. Legal documents often contain intricate language and nuanced details that are crucial for accurately identifying parties. Chunking the input documents may lead to the loss of this contextual information, which can adversely affect the model's legal understanding. To mitigate this limitation, future research could explore the use of Longformer (Beltagy et al., 2020) architectures. Longformer models are specifically designed to handle long-range dependencies in a text.

7 Conclusion

We propose a novel method to accurately predict the parties from a legal contract document. We mainly divided the approach into two phases according to the literature review as follows: (1) Dataset Creation: We introduced a large-scale high-quality dataset that includes 1000 contract documents annotated for parties by legal experts; (2) Modeling: Our dataset underwent evaluation using various techniques to assess the performance of RoBERTa. Ultimately, our most successful model exhibited a noteworthy improvement of 6.2% in Exact Match performance compared to CUAD's best model (DeBERTa).

Our research revealed that the availability of data is a critical bottleneck, as nonessential annotations and a lower amount of data will significantly drop the performance, highlighting the importance of our dataset's extensive annotations. Moreover, we demonstrated that the performance of the models is greatly affected by their architecture, indicating that advancements in algorithms by the NLP community could aid in tackling this issue. In conclusion, our dataset not only acts as a benchmark for evaluating NLP models in the Legal domain but also accelerates research toward resolving a significant real-world problem in the legal firm.

Data Availability

Our newly annotated dataset is available under an open-source license at RTUthaya.lk[4]. This dataset is intended for research purposes only.

References

Abien Fred Agarap. 2018. Deep learning using rectified linear units (relu). *arXiv preprint arXiv:1803.08375*.

Betty van Aken, Benjamin Winter, Alexander Löser, and Felix A. Gers. 2019. How Does BERT Answer Questions? In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. ACM.

Melonie Almeida, Chamodi Samarawickrama, Nisansa de Silva, Gathika Ratnayaka, and Amal Perera. 2020. Legal Party Extraction from Legal Opinion Text with Sequence to Sequence Learning. In *2020 20th International Conference on Advances in ICT for Emerging Regions (ICTer)*, pages 143–148.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The Long-Document Transformer. *arXiv:2004.05150*.

Michael James Bommarito, Daniel Martin Katz, and Eric M. Detterman. 2018. LexNLP: Natural Language Processing and Information Extraction For Legal and Regulatory Texts. *InfoSciRN: Legal Informatics (Topic)*.

Ilias Chalkidis, Ion Androutsopoulos, and Achilleas Michos. 2017. Extracting Contract Elements. In *Proceedings of the 16th Edition of the International Conference on Articial Intelligence and Law*, ICAIL '17, page 19–28, New York, NY, USA. Association for Computing Machinery.

Hendrycks Dan and Gimpel Kevin. 2016. Bridging Nonlinearities and Stochastic Regularizers with Gaussian Error Linear Units. *ArXiv*, abs/1606.08415.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021a. DeBERTaV3: Improving DeBERTa using ELECTRA-Style Pre-Training with Gradient-Disentangled Embedding Sharing.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021b. DeBERTa: Decoding-enhanced BERT with Disentangled Attention. In *International Conference on Learning Representations*.

Dan Hendrycks, Collin Burns, Anya Chen, and Spencer Ball. 2021. CUAD: An Expert-Annotated NLP Dataset for Legal Contract Review. *ArXiv*, abs/2103.06268.

Dan Hendrycks and Kevin Gimpel. 2020. Gaussian error linear units (gelus).

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. In *International Conference on Learning Representations*.

Spyretta Leivaditi, Julien Rossi, and E. Kanoulas. 2020. A Benchmark for Lease Contract Review. *ArXiv*, abs/2010.10386.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *ArXiv*, abs/1907.11692.

Suphakit Niwattanakul, Jatsada Singthongchai, Ekkachai Naenudorn, and Supachanun Wanapu. 2013. Using of Jaccard coefficient for keywords similarity. In *Proceedings of the international multiconference of engineers and computer scientists*, volume 1, pages 380–384.

Alex Sherstinsky. 2020. Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. *Physica D: Nonlinear Phenomena*, 404:132306.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Zhen Wang. 2022. Modern question answering datasets and benchmarks: A survey. *arXiv preprint arXiv:2206.15030*.

[4] https://rtuthaya.lk/contact-for-resources

Actual Party		Prediction
Columbia Laboratories, (Bermuda) Ltd.	**Model 1**	Columbia Laboratories
	Model 2	Columbia Laboratories, (Bermuda)
	Our Model	Columbia Laboratories, (Bermuda) Ltd.
DR. GAETANO MORELLO N.D. INC.	**Model 1**	DR. GAETANO
	Model 2	GAETANO MORELLO
	Our Model	DR. GAETANO MORELLO N.D. INC.
Scientific Products Pharmaceutical Co. LTD	**Model 1**	Scientific Products Pharmaceutical
	Model 2	Scientific Products Pharmaceutical Co.
	Our Model	Scientific Products Pharmaceutical Co. LTD
Shenzhen LOHAS Supply Chain Management Co., Ltd.	**Model 1**	Shenzhen LOHAS Supply Chain Management
	Model 2	Shenzhen LOHAS Supply Chain Management Co.
	Our Model	Shenzhen LOHAS Supply Chain Management Co., Ltd.

Table 2: Example predictions of different models: In this table, we have two models, referred to as **Model 1** and **Model 2**, originating from experiment D and experiment F, respectively. Additionally, we have our best model obtained from experiment I, which is denoted as **Our Model**.

Appendix

A Example predictions of different models

We infer different models from various configurations of the experiment and compare their outputs for getting better accuracy. The intermediate outputs are shown in Table 2.

B Number of Pages vs Documents Count

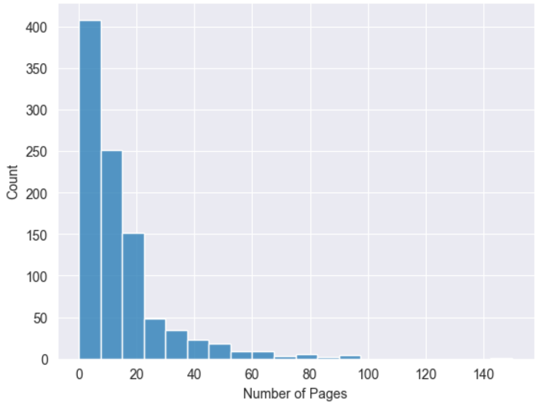

Figure 5: Number of Pages vs Documents Count. These contracts show a significant variation in length, spanning from just a few pages to well over one hundred pages. Additionally, a considerable proportion of the documents fall within the 0-20 page range.

C Number of Annotations vs Characters Bin in which Parties found

According to the Figure 6,

- 22% of the documents don't have the parties in their first two pages.

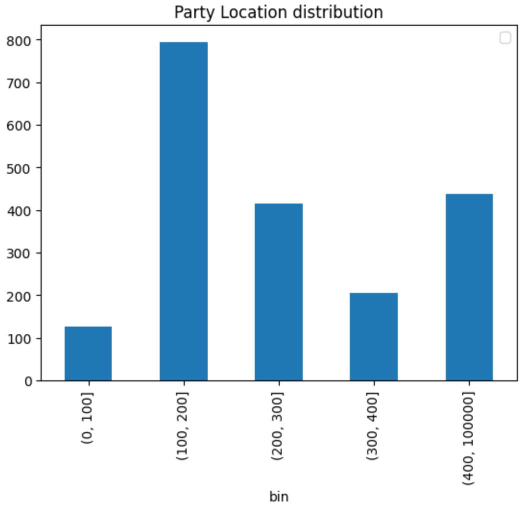

Figure 6: Number of annotations vs characters bin in which parties found

- 46% of the documents have the parties on their first page
- 31% of the documents have the parties on their second page.