Legal Contract Clause Classification

Team Members : Sowmya Polagoni, Venkat Saketh Kommi, Khaja Moinuddin Mohammed

1. Introduction

This project builds and compares two supervised models for multi-class classification of legal contract clauses using the CUAD dataset. The first model is a feedforward neural network trained on TF-IDF features, representing each clause as a sparse bag-of-words vector. The second model is an LSTM-based classifier trained on tokenized and padded word sequences to incorporate word order information. To simplify the task and handle class imbalance, the most frequent clause categories are modeled explicitly and less frequent categories are grouped into an "Other" class. The goal is to evaluate whether sequence modeling improves clause classification compared to a strong TF-IDF baseline, supporting practical applications such as contract review and compliance analysis.

2. Problem Statement

Legal contracts are lengthy documents that contain many clause types such as governing law, termination, liability, and insurance. Accurately identifying and categorizing these clauses is a critical step in contract review, compliance checks, and risk assessment. However, manual clause classification is slow, labor-intensive, and prone to human error, especially when reviewing large volumes of contracts written in complex legal language.

This project addresses the problem of **automated multi-class classification of legal contract clauses**. Given the text of an individual clause extracted from a contract, the goal is to predict its clause category accurately. The task is challenging due to class imbalance, variation in clause length, and the specialized vocabulary used in legal documents. Solving this problem can significantly reduce manual effort in legal workflows and help ensure that important clauses are not overlooked during contract analysis.

3. Data Source and Description

This project uses the Contract Understanding Atticus Dataset (CUAD**),** a publicly available, expert-annotated dataset created for legal contract analysis and clause classification. CUAD is released by The Atticus Project and is available on GitHub.

The dataset consists of real commercial contracts annotated with labeled clause spans. Each annotation specifies the start and end positions of a clause within the contract text along with its corresponding legal category. The data is provided in JSON format and includes:

- **CUADv1.json**: Primary training dataset containing 510 contracts with annotated clause spans
- **test.json:** Separate holdout dataset used for final, unbiased evaluation.

3.1 Dataset Size

After extracting clause text using annotation boundaries, the dataset initially contained 13,459 clause instances. Following data cleaning steps like removing empty clauses, clauses shorter than two words, and duplicate clause texts and the final dataset consisted of:

- **Training set**: 11,047 clauses from *CUADv1.json*
- **Test set**: 2,172 clauses from *test.json*

3.2 Clause Categories

The original CUAD dataset includes **41 distinct** clause categories. To reduce complexity and address class imbalance, the nine most frequent categories were retained, while the remaining categories were grouped into a single "Other" class. This resulted in a **10-class multi-class classification** problem. The final class distribution is imbalanced, reflecting real-world contract data, with categories such as *Parties* and *License Grant* appearing more frequently than others.

3.3 Text Characteristics

Clause lengths vary widely, ranging from very short phrases to clauses exceeding several hundred words. The median clause length is approximately **17 words**, with the **95th** percentile around **75 words**, requiring models to handle variable-length inputs.

3.4 Preprocessing Steps

The dataset was preprocessed using a consistent pipeline for both training and test data. Clause text was first extracted using the annotated start and end positions provided in the dataset. All text was lowercased and normalized to remove extra whitespace, and empty, duplicate, or very short clauses were removed to improve data quality. To manage class imbalance and reduce task complexity, infrequent clause categories were grouped into a single "Other" class, and all clause labels were integer-encoded for multi-class classification. For the feedforward model, clauses were transformed into TF-IDF feature vectors, while for the sequence-based model, text was tokenized and padded to a fixed length to support LSTM training. This preprocessing pipeline resulted in clean, labeled clause text suitable for training and evaluating both classification models.

4. Model Development

Two neural network models were implemented and compare

1. a Feedforward Neural Network (MLP) using TF-IDF features
2. Long Short-Term Memory (LSTM)–based sequence model.

4.1 Feedforward Neural Network (MLP) with TF-IDF Features

Input Representation

In the first approach, each clause is represented using **Term Frequency–Inverse Document Frequency (TF-IDF)** features. Clause text is vectorized using unigrams and bigrams, producing a **fixed-length 5,000-dimensional feature vector**. TF-IDF emphasizes terms that are important within a clause but less frequent across the corpus, making it effective for capturing legally meaningful keywords and phrases.

This representation is well-suited for legal documents, where clause categories are often defined by **distinctive terminology** such as "governing law," "assignment," or "limitation *of* liability."

Model Architecture

The feedforward neural network consists of the following layers:

- **Input Layer:**
  5,000-dimensional TF-IDF feature vector
- **Hidden Layer 1:**
  Dense layer with 512 units and ReLU activation
  Dropout layer with rate 0.3
- **Hidden Layer 2:**
  Dense layer with 256 units and ReLU activation
  Dropout layer with rate 0.3
- **Output Layer:**
  Dense layer with 10 units and Softmax activation for multi-class classification

The model contains approximately **2.7 million trainable parameters**. It is trained using the **Adam optimizer** with **sparse categorical cross-entropy loss**, making it suitable for integer-encoded labels.

This model is computationally efficient and effective at learning nonlinear relationships between TF-IDF features and clause categories. Because many legal clauses rely heavily on standardized wording rather than complex syntactic structure, the feedforward model serves as a **strong baseline** and a practical solution for legal clause classification.

4.2 LSTM-Based Text Classification Model

Input Representation

The second approach models clauses as **sequences of words** to preserve word order and contextual relationships. Clause text is tokenized into sequences of word indices using a vocabulary size of **10,000**. To handle variable-length clauses, sequences are **padded or truncated to a maximum length of 150 tokens**, covering the majority of clauses based on text length analysis.

This representation enables the model to capture contextual patterns that may be important for interpreting legal language.

Model Architecture

The LSTM-based classifier consists of the following layers:

- **Input Layer:**
  Sequence of 150 word indices

- **Embedding Layer:**
  Vocabulary size: 10,000
  Embedding dimension: 128
  Embeddings are randomly initialized and learned during training

- **LSTM Layer:**
  128 LSTM units
  Dropout rate: 0.3
  Recurrent dropout rate: 0.2

- **Dropout Layer:**
  Dropout rate: 0.3

- **Output Layer:**
  Dense layer with 10 units and Softmax activation.

The model contains approximately **1.4 million trainable parameters** and is trained using the **Adam optimizer** with **sparse categorical cross-entropy loss**.

Legal clauses often convey meaning through **sequential structure**, particularly in conditional or obligation-based statements (e.g., *"subject to," "shall remain in effect unless terminated"*). The LSTM architecture is designed to capture these dependencies, allowing the model to learn contextual relationships that are not available in bag-of-words representations.

4.3 Training Strategy and Model Comparison

To ensure a fair comparison, both models were trained using consistent configurations:

- **Optimizer:** Adam

- **Loss Function:** Sparse categorical cross-entropy

- **Output Activation:** Softmax (10 classes)

- **Validation Strategy:** 10% internal validation split
- **Early Stopping:** Patience of 5 epochs based on validation loss
- **Class Imbalance Handling:** Balanced class weights applied during training

The feedforward model was trained with a batch size of **64**, while the LSTM model used a batch size of **32** due to higher memory requirements. No pretrained embeddings, transformer-based models, or extensive hyperparameter tuning were used.

5. Evaluation and Results

5.1 Evaluation Metrics

The models were evaluated using accuracy, precision, recall, and F1-score. Because the CUAD dataset exhibits significant class imbalance across clause categories, macro-averaged precision, recall, and F1-score were used as primary evaluation metrics. Macro F1 assigns equal weight to each class, ensuring that performance on rare clause categories is not overshadowed by dominant classes such as *Parties* or *Document Name*. This makes macro F1 a more appropriate measure of overall model effectiveness for this task.

5.2 Overall Model Performance

This section evaluates the overall performance of the two implemented models on a held-out test set. Both models were trained on the same data and evaluated using identical metrics to ensure a fair comparison. Performance is reported using accuracy and macro-averaged precision, recall, and F1-score, which are especially important given the class imbalance in legal clause categories.

Model	Accuracy	Precision (Macro)	Recall (Macro)	F1-score (Macro)
TF-IDF + Feedforward NN	0.9319	0.9484	0.9319	0.9296
LSTM-based Model	0.9217	0.9338	0.9217	0.9205

Table 1.Overall Model Performance

The results in Table 1 show that the TF-IDF-based feedforward model outperformed the LSTM model across all reported metrics. The difference is most evident in the macro

F1-score, where the feedforward model achieved a score of **0.9296**, compared to **0.9205** for the LSTM. This indicates more balanced performance across both frequent and less frequent clause categories.

Figure. 1 illustrate the training and validation loss and accuracy for the **TF-IDF + Feedforward Neural Network** across epochs. The training loss decreases rapidly and stabilizes early, while validation loss remains relatively stable with a small gap between training and validation curves. This behavior indicates fast convergence and limited overfitting. The validation accuracy plateaus around 87–88%, suggesting that the model generalizes well to unseen data.

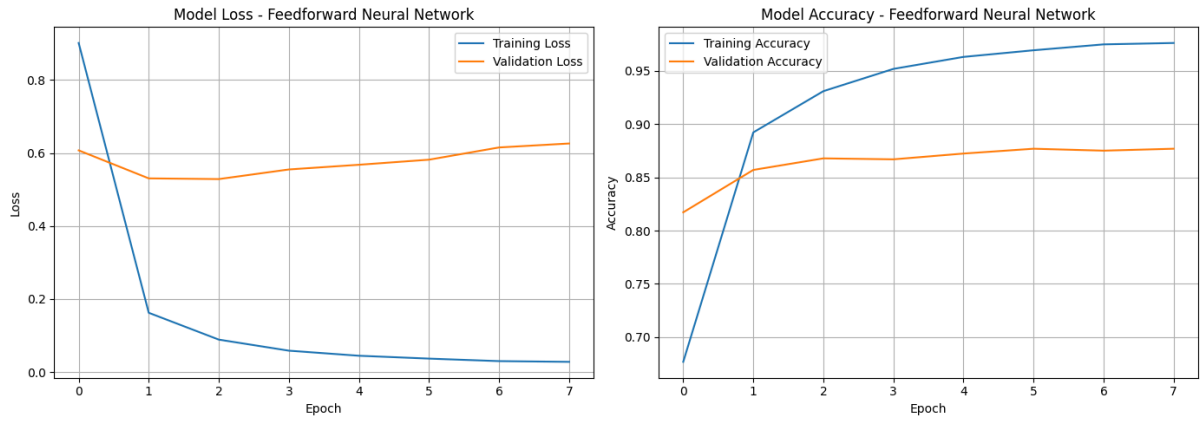

Figure. 1 Training and validation performance of TF-IDF model

Figure.2 shows the corresponding learning curves for the **LSTM-based text classifier**. Compared to the feedforward model, the LSTM requires more epochs to converge, reflecting the increased complexity of sequence modeling. While training accuracy continues to improve steadily, validation accuracy plateaus earlier and exhibits greater fluctuation. This suggests that the LSTM model is more prone to overfitting and requires stronger regularization or additional data to achieve further gains.

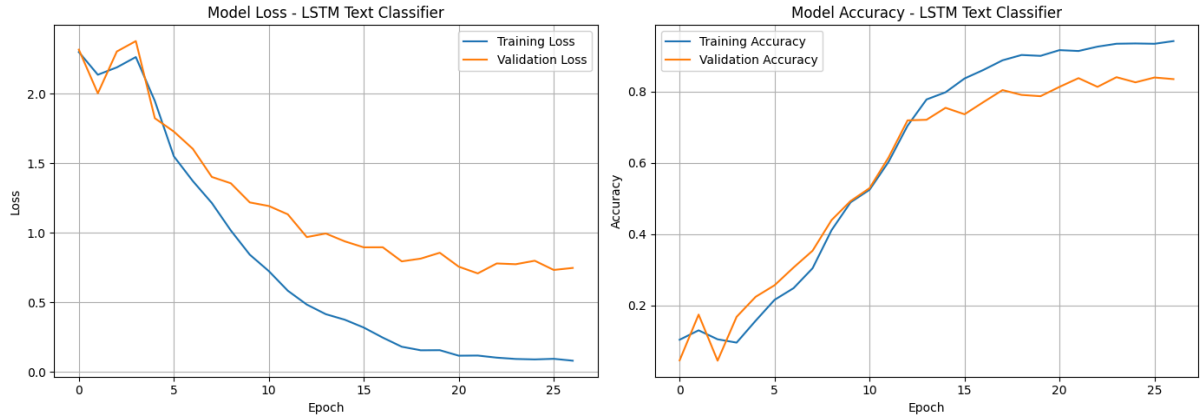

Figure.2 Training and validation loss and accuracy for the LSTM-based text classification model

5.3 Per-Class Performance Analysis

Overall performance scores give a general comparison between the models, but they do not show how well each model works for individual clause types. Looking at results for each clause category helps us understand where the models perform well and where they struggle. This is important because legal contracts often have uneven class distributions and similar language across different clause types.

Clause categories with clear and fixed wording such as **Agreement Date**, **Parties**, and **Governing Law** show strong performance for both models. These clauses usually follow standard formats and use specific keywords, which makes them easier to detect. On the other hand, clauses with broader meanings or overlapping content, especially those labeled as **Other**, are harder for both models to classify accurately.

The below table presents the per-class precision, recall, and F1-scores for **the feedforward**

Clause Category	Test Samples	Precision	Recall	F1-Score
Agreement Date	**87**	**0.88**	**1.00**	**0.94**
Anti-Assignment	**128**	**0.88**	**0.99**	**0.93**
Audit Rights	**102**	**0.99**	**1.00**	**1.00**
Cap On Liability	**106**	**0.94**	**0.96**	**0.95**
Expiration Date	**73**	**0.77**	**0.99**	**0.87**
Governing Law	**90**	**0.99**	**0.99**	**0.99**

Insurance	**116**	**0.97**	**0.99**	**0.98**
License Grant	**107**	**0.59**	**0.97**	**0.73**
Other	**998**	**0.99**	**0.86**	**0.92**
Parties	**365**	**0.99**	**0.99**	**0.99**
Macro Average	**—**	**0.90**	**0.98**	**0.93**
Weighted Average	**—**	**0.95**	**0.93**	**0.94**

Table 2: Per-Class Performance of Feedforward Neural Network (TF-IDF)

The feedforward model achieved very high recall across nearly all clause categories, indicating strong coverage of true clause instances. However, some categories such as *License Grant* showed lower precision despite high recall, suggesting that clauses from related intellectual property categories were occasionally misclassified as licensing clauses. The Other category also exhibited reduced recall due to its heterogeneous composition.

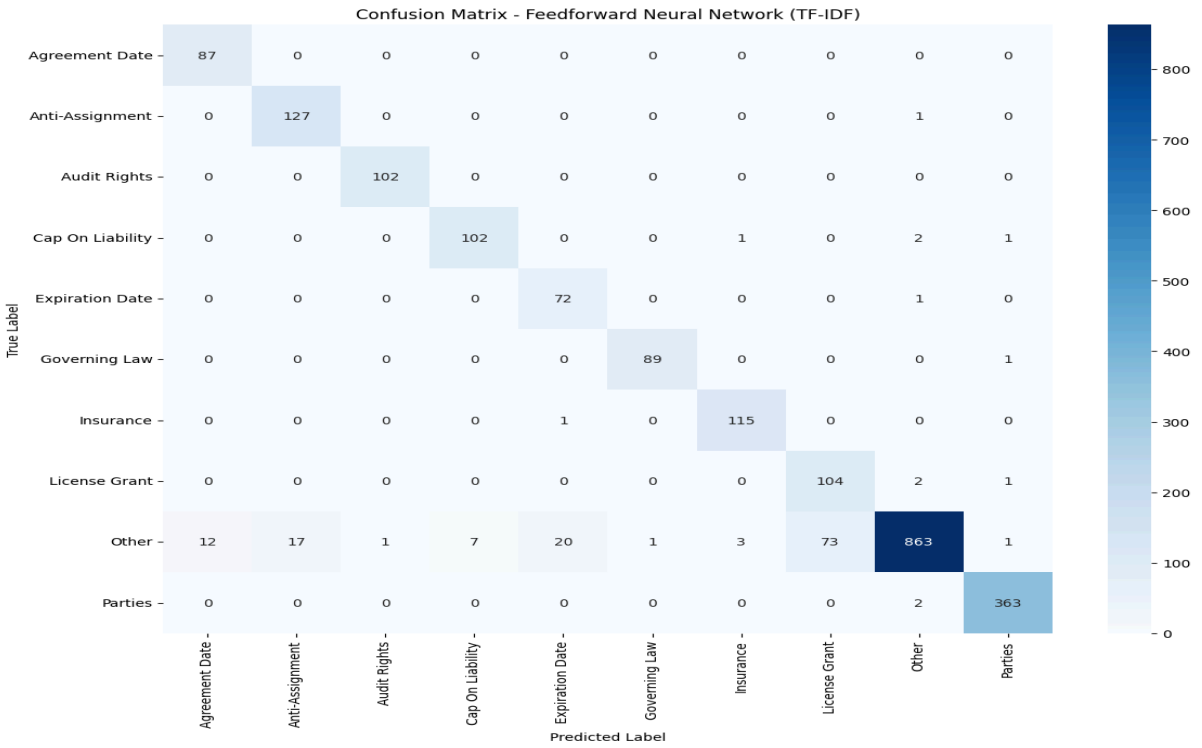

Figure.1 Confusion Matrix of TF-IDF Model

The above figure shows the **confusion matrix for the Feedforward Neural Network using TF-IDF features**. It compares the model's predicted clause categories with the true labels. Most values are concentrated along the diagonal, which means the model correctly classifies the majority of clauses across categories. Clause types such as **Agreement Date**, **Audit Rights**, **Governing Law**, **Insurance**, **License Grant**, and **Parties** are predicted very accurately, with almost no confusion. Similar to the LSTM model, most misclassifications occur in the **Other** category, where some clauses are confused with more specific clause types.

Now let us summarise the per-class performance of the LSTM-based text classification model.

Clause Category	**Test Samples**	**Precision**	**Recall**	**F1-Score**
Agreement Date	87	0.89	1.00	0.94
Anti-Assignment	128	0.81	0.98	0.89

Audit Rights	102	0.88	0.99	0.93
Cap On Liability	106	0.90	0.95	0.93
Expiration Date	73	0.85	0.97	0.90
Governing Law	90	0.98	1.00	0.99
Insurance	116	0.93	0.97	0.95
License Grant	107	0.65	0.97	0.78
Other	998	0.98	0.86	0.91
Parties	365	0.98	0.97	0.98

Table 3: Per-Class Performance of LSTM-Based Model

The LSTM model achieved strong recall for many categories, particularly for longer and more complex clauses. However, precision varied more across categories compared to the TF-IDF model. This suggests that while the LSTM effectively captured sequential patterns, it occasionally struggled to distinguish between clauses with similar contextual phrasing, especially in legally dense language.

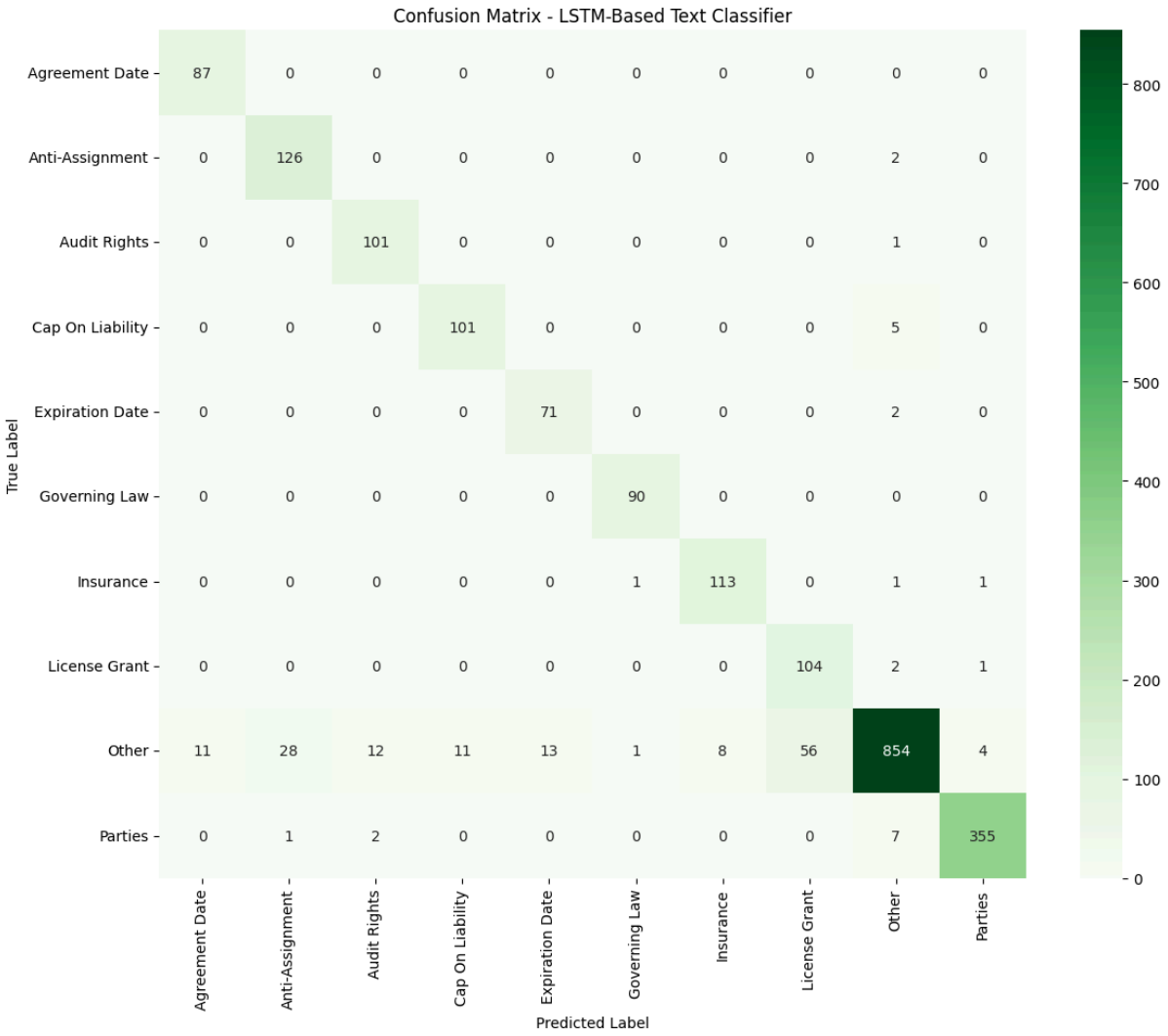

Figure 2 Confusion Matrix of LSTM Model

The confusion matrix shows how the model's predictions compare to the true clause labels. Most values are concentrated along the diagonal, indicating that the model correctly classifies many clause categories. However, some off-diagonal values are visible, especially involving the **Other** category, showing that the model sometimes confuses related or overlapping clause types. This suggests that while the LSTM performs well on clearly defined clauses with consistent language, it has more difficulty distinguishing between clauses that share similar wording or broader meanings.

6. Discussion

The results show that both the TF-IDF–based feedforward neural network and the LSTM-based model can classify legal contract clauses accurately, but the feedforward model performs better overall. This suggests that legal contract clauses are mainly distinguished by repeated keywords and standardized language rather than complex sentence structure. TF-IDF features capture these keyword patterns well, which allows the feedforward model to separate clause types with high accuracy and balanced performance.

The per-class results support this finding. Clause types such as **Agreement Date**, **Parties**, and **Governing Law** achieve very high precision and recall in both models because their language is clear and consistent. Some categories, like **License Grant**, show high recall but lower precision, meaning the model identifies most of these clauses but sometimes confuses them with similar clauses related to usage or intellectual property. The **Other** category is the hardest to classify because it contains many different and less common clause types grouped together.

The LSTM model is able to capture word order and sequence information and performs reasonably well on longer or more complex clauses. However, this added complexity does not lead to better overall results. The lack of pretrained embeddings and the limited size of the dataset likely reduce the LSTM's ability to fully use contextual information, leading to more variation in performance across classes and slightly lower macro F1-scores.

Overall, these results show that more complex models do not always perform better than simpler ones, especially in domains like legal text where language is highly standardized. From a practical point of view, the TF-IDF–based feedforward model is easier to interpret, faster to train, and cheaper to deploy, making it well suited for real-world legal contract review systems.

7. Conclusion and Future Work

This project explored automatic legal contract clause classification using two neural network approaches: a feedforward model with TF-IDF features and an LSTM-based sequence model. Both models achieved strong performance on the CUAD dataset, but the TF-IDF–based feedforward model consistently outperformed the LSTM across accuracy and macro-averaged evaluation metrics. This shows that keyword patterns and lexical features are highly effective for identifying legal clause types in structured contract text.

The results demonstrate that simpler, feature-based models can be both accurate and efficient for legal NLP tasks, especially when the language is standardized. In contrast, the LSTM model did training data. not gain a significant advantage without pretrained embeddings or larger domain-specific

Future Improvements:

- Data augmentation: Oversample rare categories or use synthetic clauses to reduce imbalance.
- Model enhancements: Add pretrained embeddings (e.g., GloVe, BERT) for LSTM; hyperparameter tuning (e.g., grid search). Experiment with BiLSTM or attention mechanisms.
- Preprocessing: Add lemmatization, entity recognition, or domain-specific tokenization. Increase vocab_size or use subword embeddings.
- Evaluation: Implement cross-validation, ROC curves, and statistical tests. Add qualitative error analysis (e.g., misclassified examples).
- Code quality: Modularize into scripts/functions; add logging, error handling, and configuration files. Optimize memory (e.g., sparse TF-IDF).
- Extensions: Multi-label classification (clauses can have multiple categories), ensemble models, or deployment as a web API. Test on larger datasets or real contracts.
- Experimentation: Compare with traditional models (e.g., SVM on TF-IDF); add regularization or data normalization.

References

1. Hendrycks, D., Burns, C., Chen, A., & Ball, S. (2021). *CUAD: An expert-annotated NLP dataset for legal contract review.* arXiv:2103.06268.https://arxiv.org/abs/2103.06268

2. Hassan, F., Le, Le, T., & Lv, X. (2021). Addressing legal and contractual matters in construction using natural language processing: A critical review. *Journal of Construction Engineering and Management, 147*(6), 04021049.https://www.researchgate.net/publication/354284117_Addressing_Legal_and_Contractual_Matters_in_Construction_Using_Natural_Language_Processing_A_Critical_Review

3. Aejas, B., Belhi, A., & Bouras, A. (2024). Contract clause extraction using question-answering task. In *International Conference on Web Information Systems and Technologies* (pp. 345–365). .https://link.springer.com/chapter/10.1007/978-981-96-0579-8_23

4. Mohite, A., Sheik, R., & Nirmala, S. J. (2025). Improving legal text classification through data augmentation using deep learning models. In *Recent Advances in Computing* (pp. 345–358). CRC Press.https://www.taylorfrancis.com/chapters/edit/10.1201/9781003570349-26/improving-legal-text-classification-data-augmentation-using-deep-learning-models-akshay-mohite-reshma-sheik-jaya-nirmala

5. Aejas, B., Belhi, A., & Bouras, A. (2025). Using AI to ensure reliable supply chains: Legal relation extraction for sustainable and transparent contract automation. *Sustainability, 17*(9), 4215.https://www.mdpi.com/2071-1050/17/9/4215