Document-level Natural Language Inference in Contracts (ContractNLI)

Manav Shah, Chirag Jain, Madhav Tank

November 2024

1 Introduction

Contract NLI is crucial because contracts are foundational to legal and business agreements, containing complex language that must be interpreted accurately. Legal contracts often consist of intricate clauses that may require clear understanding and analysis, especially in cases where contradictions or ambiguities exist. Furthermore, it aids in reducing the time and cost of manual contract analysis, making it essential for industries that rely on large volumes of contracts, such as finance, law, and insurance.

Our task, given a contract and a set of hypotheses (each being a sentence), is to classify whether each hypothesis is *entailed by*, *contradicts*, or is *not mentioned by* (neutral to) the contract. Additionally, we must identify evidence for our decision as spans within the contract.

More formally, the task consists of:

- **NLI classification**: Whether the hypothesis is *entailed by*, *contradicted by*, or *not mentioned in* the contract.
- **Evidence identification**: Specific spans of text within the contract that justify the classification for entailment or contradiction.

2 Exploratory Data Analysis (EDA)

2.1 Dataset Structure

The dataset is in JSON format with the following fields:

- **Metadata**: ID, file name, text.
- **Spans**: Token boundaries for sections of the document.
- **Annotation Sets**: Specify NLI decisions and evidence spans.

2.2 Annotation Types

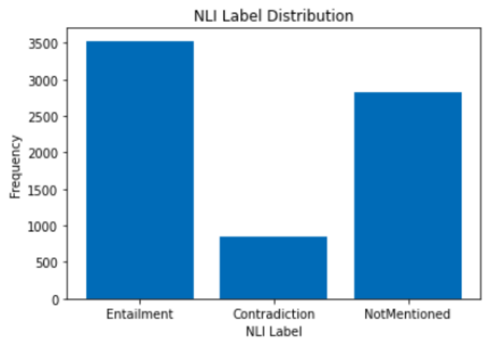
Each annotation corresponds to a hypothesis and includes the following fields:

- **choice**: The NLI classification, which can be one of the following:
  - **Entailment**
  - **Contradiction**
  - **Not Mentioned**
- **spans**: A list of span indices that mark the evidence in the document supporting the decision.

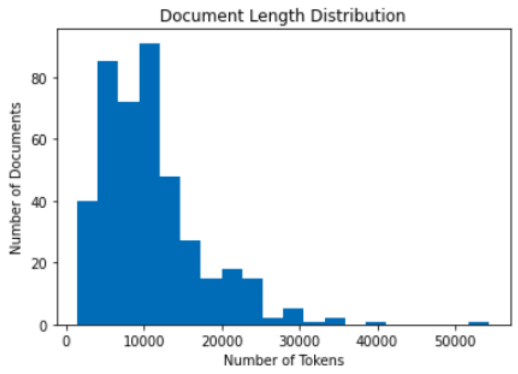
Example:

```
{
  "id": 34,
  "file_name": "Annex E_Non-Disclosure.pdf",
  "text": "...",
  "spans": [[0, 44], [8573, 8579], ...],
  "annotation_sets": [{
    "annotations": {
      "nda-11": {"choice": "NotMentioned", "spans": []},
      "nda-16": {"choice": "Entailment", "spans": [39, 40]}
    }
  }]
}
```

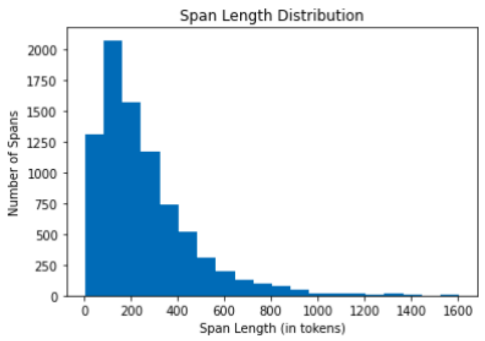
2.3 Key Findings



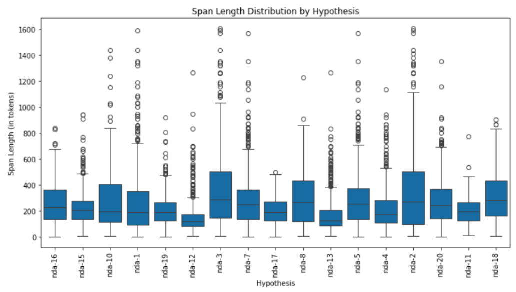
(a) NLI Label Distribution



(b) Document Length Description



(c) Span Length Description



(d) Span Length Description by Hypothesis

Figure 1: Data Analysis

2.4 Data Insights and Analysis

1. NLI Label Distribution:

- **Entailment**: 50%

- **Contradiction**: 15%
- **Not Mentioned**: 35%

Inference: Techniques like class weighting or oversampling **Contradiction** may balance model performance.

2. **Document Length**:

   - Average: $\sim$ 2,500 tokens

   Token limits require splitting long documents for transformer models.

3. **Evidence Span Length**:

   - Median: 30 tokens

   Models should handle both fine-grained and multi-sentence spans.

4. **Evidence Density**:

   - Average: $\sim$ 10 spans per document (varies from 1–30)

   Models should handle both sparse and dense evidence.

5. **Hypothesis-Specific Span Variability**:

   - Example: Hypothesis `nda-16` median span = 40 tokens; `nda-1` = 20 tokens.

   Complex spans may require tailored attention mechanisms.

6. **Token Overlap**:

   - 20% overlap between evidence and non-evidence spans.

   Challenge: Disambiguate relevant and irrelevant text.

7. **Label Co-occurrence**:

   - 40% of documents contain **Entailment** + **Not Mentioned** labels.
   - Only 10% have all three labels.

8. **Evidence Locations**:

   - 60% appear in the middle of documents.

3 Baseline Models

3.1 RoBERTa and DeBERTa-v3 for NLI Task

In this approach, each document-hypothesis pair is structured as:

```
hypothesis [SEP] document
```

This format emphasizes the hypothesis, enabling the model to effectively capture semantic relationships and context. By positioning the hypothesis at the forefront, the model is guided to focus on its alignment with the document.

Given the 512-token limitation of transformer models, documents exceeding this length are truncated. This strategy prioritizes the initial sections of documents, under the assumption that the most relevant content typically appears earlier.

The models **DeBERTa** and **RoBERTa** were selected due to their exceptional performance in natural language understanding tasks, making them well-suited for this application.

Results:

Evaluation Metrics for DeBERTa:

Metric	Value
Accuracy	0.66783
Precision	0.66450
Recall	0.66783
F1 Score	0.66140

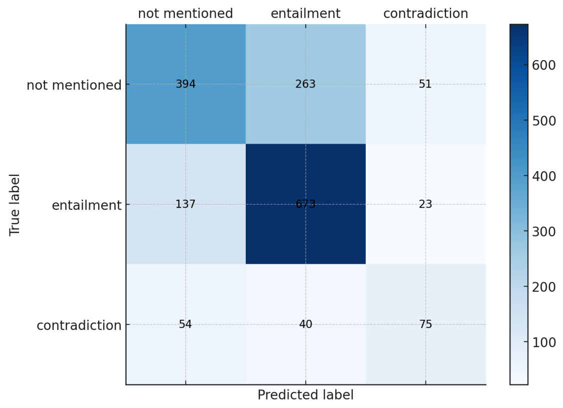


Figure 2: Confusion Matrix for DeBERTa

Evaluation Metrics for RoBERTa:

Metric	Value
Accuracy	0.6705
Precision	0.6753
Recall	0.6705
F1 Score	0.6686

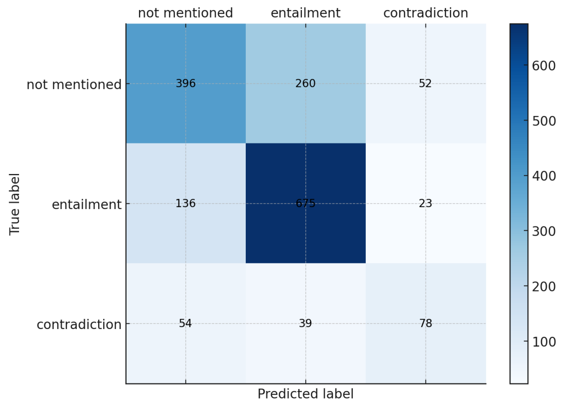


Figure 3: Confusion Matrix for RoBERTa

As we can see, **RoBERTa model outperforms the DeBERTa model.**

3.2 Span TF-IDF + SVM vs. Span TF-IDF + Cosine Similarity for EI

Two baseline models were implemented and compared for this task:

- **Span TF-IDF + SVM**
- **Span TF-IDF + Cosine Similarity**

Both models leverage **TF-IDF (Term Frequency-Inverse Document Frequency)** to transform spans and hypotheses into numerical vectors but differ in their approach to classifying spans. This report provides an overview of how each model works and compares their performance, particularly with respect to **Precision @ 80% Recall (P@R80) and mean average precision (mAP)**.

Span TF-IDF + SVM

The Span TF–IDF + SVM model is a **span-level binary classification model**. The process begins with converting both the document spans and hypotheses into vectorized representations using TF–IDF. The **TF–IDF vectors capture the relative importance** of words across the corpus.

Once the spans and hypotheses are transformed into numerical vectors, the **SVM (Support Vector Machine)** classifier is trained to distinguish between **relevant and non-relevant spans**. This is done by concatenating the TF–IDF vector of each span with that of the hypothesis and feeding the combined vector into a linear SVM. The SVM attempts to find a hyperplane that best separates the spans that are relevant from those that are not, based on the combined features of the span and hypothesis.

Span TF-IDF + Cosine Similarity

The **Span TF–IDF + Cosine Similarity** model takes a more straightforward, **non-learning-based** approach. In this model, each span from the document and the hypothesis is transformed into a TF–IDF vector. Instead of training a classifier, the cosine similarity between the **TF–IDF vector of each span and the hypothesis vector** is computed.

Cosine similarity is a metric that measures the cosine of the angle between two vectors. It ranges from 0 to 1, where 1 indicates that the vectors are identical (high similarity), and 0 indicates no similarity. For each span, the cosine similarity score is treated as a prediction score, indicating how similar the span is to the hypothesis.

Evaluation:

Model	**Precision @ 80% Recall**	**Mean Average Precision**
Span TF–IDF + Cosine Similarity	0.0546	0.3731
Span TF–IDF + SVM	0.3152	0.4157

Table 1: Performance Comparison of Models

Analysis:

The results show that the **Span TF–IDF + SVM** model outperforms the **Span TF–IDF + Cosine Similarity** model in both **Precision @ 80% recall** and **Mean Average Precision**.

1. **Precision @ 80% recall**:

   The **Span TF–IDF + SVM** model achieves a significantly higher precision of **0.3152** compared to **0.0546** for the **Span TF–IDF + Cosine Similarity** model. This suggests that the SVM-based model is better at distinguishing relevant spans when aiming for 80% recall, meaning it identifies fewer false positives at this level of recall.

2. **Mean Average Precision (mAP)**:

   The **Span TF–IDF + SVM** model also outperforms in mAP, with a score of **0.4157** compared to **0.3731** for the **Span TF–IDF + Cosine Similarity** model. This indicates that the SVM model has a better overall performance across all recall levels, providing more accurate and consistent predictions.

4 Methodology

4.1 RoBERTa and DeBERTa-v3 for NLI Task

In our updated approach, instead of passing the entire document (with truncation at 512 tokens), we train the models using **spans** and the **hypothesis**, paired with the target label. This strategy allows the models to focus on specific, relevant portions of the document, avoiding the loss of important context due to truncation.

By aligning the input to specific spans, the models can better understand the semantic relationship between the hypothesis and the corresponding section of the document, leading to a significant improvement in performance.

Input Format: `span [SEP] hypothesis`

This ensures that the relevant span is paired directly with the hypothesis, enabling the model to learn more precise relationships without being overwhelmed by unrelated content from the full document.

Results:

Evaluation Metrics for DeBERTa:

Metric	Value
Accuracy	0.9498
Precision	0.9502
Recall	0.9498
F1 Score	0.9500

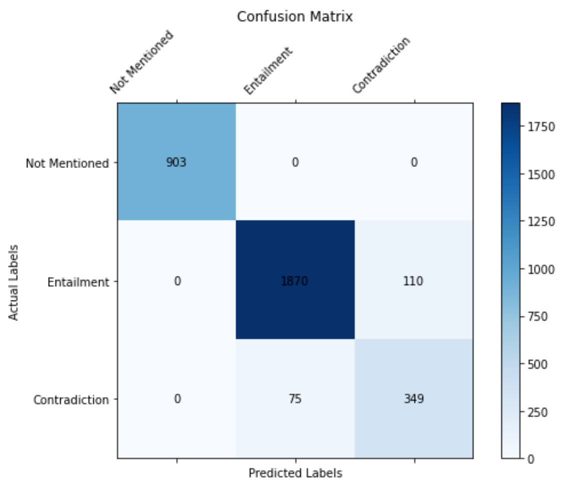


Figure 4: Confusion Matrix for DeBERTa

Evaluation Metrics for RoBERTa:

Metric	Value
Accuracy	0.944
Precision	0.946
Recall	0.944
F1 Score	0.945

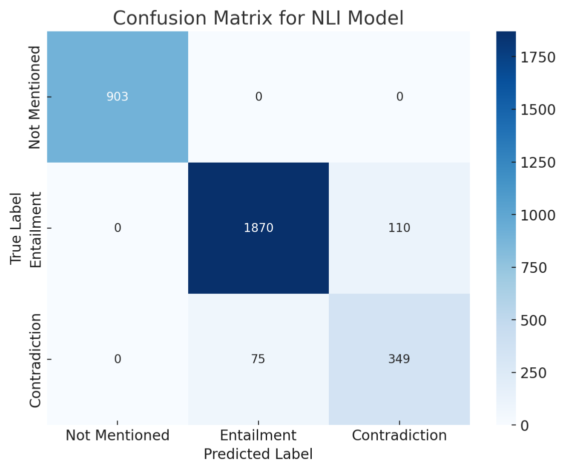


Figure 5: Confusion Matrix for RoBERTa

Comparison with Full-Document Approach:

By training on spans, DeBERTa achieved near-perfect results, significantly outperforming its performance in the full-document approach. Similarly, RoBERTa also showed substantial improvement, reinforcing the effectiveness of focusing on spans for the Natural Language Inference task.

4.2 Separate Models for Entailment and Contradiction Detection in Evidence Identification

This approach uses **two distinct BERT-based models** for evidence identification (EI), focusing separately on **entailment** and **contradiction** relationships. Each model performs **binary classification** to determine the relevance of evidence spans for a given hypothesis.

Key Details of the Approach

1. Models and Their Purpose

- **Entailment Model**:
  - Classifies whether the given input (hypothesis and evidence pair) supports **entailment** or not.
  - Output: Binary classification (1 = entailment, 0 = not entailment).
- **Contradiction Model**:
  - Classifies whether the given input (hypothesis and evidence pair) indicates **contradiction** or not.
  - Output: Binary classification (1 = contradiction, 0 = not contradiction).

2. Input to Each Model

- **Input Format**: For both models, the input is a concatenation of:
  - **Hypothesis**: A query or claim in natural language.
  - **Evidence**: A specific span (sentence) from the document.
  - Combined as: `hypothesis [SEP] evidence`.
- The models predict whether this input pair matches their respective class.

3. Training Process

- **Positive Examples**: Pairs of hypotheses and evidence spans labeled as either **entailment** or **contradiction** are used as positive examples (label = 1) for the respective model.
- **Negative Examples**:
  - Spans that are not entailment are randomly selected and labeled as **0** for the entailment model.
  - Similarly, spans that are not contradiction are labeled as **0** for the contradiction model.
  - This ensures the models learn to distinguish between relevant and irrelevant evidence for their specific class.

4. Inference Process

- At inference time, the goal is to identify the **most relevant evidence span** from the document for a given hypothesis:
- For the **entailment model**:
  - Iterate over all spans in the document.
  - Compute the probability of entailment for each span using the model.
  - Select the span with the **maximum probability of entailment**.
- Similarly, the **contradiction model** identifies spans with the maximum probability of contradiction.

Advantages of This Approach:

1. **Focused Training**: Each model specializes in identifying a specific relationship (entailment or contradiction), leading to better performance for these tasks.
2. **Dynamic Evidence Selection**: At inference, the models assess all potential spans, ensuring that the best evidence for the hypothesis is selected.
3. **Balanced Learning**: By including random negative examples, the models learn to avoid false positives, improving robustness.

Evaluation:

Model	**Precision @ 80% Recall**	**Mean Average Precision**
Entailment	0.514072	0.6203835
Contradiction	0.45712	0.52398

Table 2: Performance Comparison of Models

5 Span-NLI-BERT

The paper that we follow for this project will be ContractNLI (Koreeda & Manning, 2021), which introduces Span-NLI-BERT.

In tasks involving long documents (e.g., contracts), transformer models have a fixed input limit (like 512 tokens in BERT). The challenge is to divide the long document into smaller, overlapping segments (or contexts) that can fit into the model without losing the continuity of meaning across span boundaries.

5.1 Key Idea

The algorithm dynamically adjusts the context boundaries **to ensure that spans are not split** and that each span is given enough surrounding information (by allowing a small overlap between contexts). This way, even though the document is split into smaller pieces, the model can still access enough context to make correct predictions, both for natural language inference (entailment, contradiction, or neutral) and span identification (for evidence).

5.2 Algorithm: Dynamic Context Segmentation

Algorithm 1 Dynamic context segmentation

```
Require: Span boundary token indices $B = [b_0, b_1, \ldots]$, Tokens $T = [t_0, t_1, \ldots]$, min. # of surrounding tokens $n$, max. context length $l$
Ensure: List of overlapping contexts
 1: $contexts \leftarrow []$
 2: $start \leftarrow 0$
 3: while $\text{len}(B) > 0$ do
 4:     for $b_i \in B$ where $b_i - start \leq l$ do
 5:         $B.\text{remove}(b_{i-1})$
 6:         $end \leftarrow b_{i-1}$
 7:     end for
 8:     $contexts.\text{append}(T[start : (start + l)])$
 9:     $start \leftarrow end - n$
10: end while
11: return $contexts$
```

5.2.1 Input

- **Span boundary token indices**: These indicate the start and end of the spans in the document (like clause or sentence boundaries).
- **Tokens**: The sequence of tokens in the document.
- **Min. number of surrounding tokens (`n`)**: This ensures that each context has sufficient surrounding information, so segments don't lose the relevant context around spans.
- **Max. context length (`l`)**: The maximum number of tokens that can fit into one context (defined by the model's limitations, such as 512 for BERT).

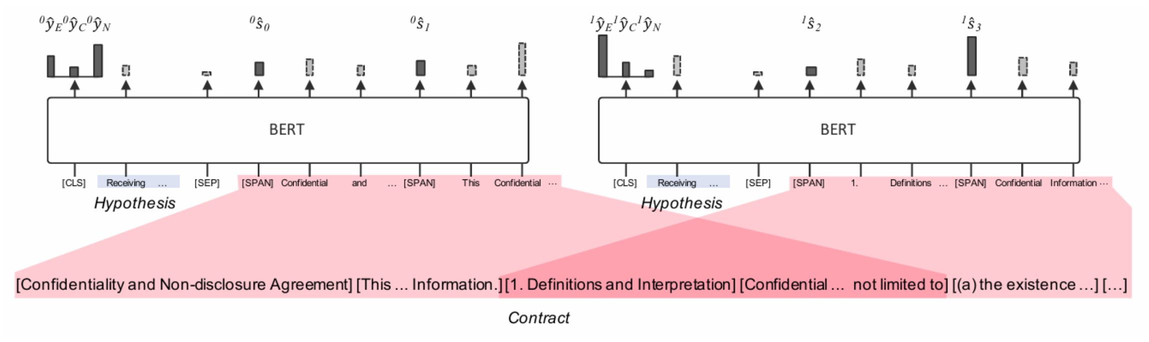
5.2.2 Output

- A list of overlapping contexts that can be fed into the transformer model.

5.3 [CLS] and [SEP] tokens

- In BERT-based models, the [CLS] token is a special token that is added at the **beginning of every input sequence** (i.e., every document or context fed into the model). The final hidden state corresponding to this token is typically used as the **aggregate representation** of the entire input sequence.
- After splitting the document into contexts, the tokens from the hypothesis and the contract are concatenated, with a [SEP] token separating them. The entire sequence starts with the [CLS] token.
- The hidden state corresponding to this [CLS] token (after being processed by the transformer model) is then passed through a **multi-layer perceptron (MLP)**, which classifies the relationship between the contract and hypothesis into **entailment, contradiction, or neutral**.
- **[SPAN] token**: The model inserts special markers (SPAN tokens) to indicate the boundaries of different spans.
- The input to the model might look like: **[CLS] Hypothesis [SEP] Contract Text [SEP]**

5.4 Model Architecture



BERT
BERT
[CLS]
Receiving ...
[SEP]
[SPAN]
Confidential
and
[SPAN]
This
Confidential ...
Hypothesis
1.
Definitions ...
Information ...
[Confidentiality and Non-disclosure Agreement] [This ... Information.] [1. Definitions and Interpretation] [Confidential ... not limited to] [(a) the existence ...] [...]
Contract

Figure 6: Model architecture of Span NLIBERT

The model handles both tasks simultaneously using two multi-layer perceptrons (MLPs) on top of the transformer model (e.g., BERT).

- One MLP predicts **evidence spans** (sections of text that provide support).
- Another MLP predicts the **NLI label** (entailment, contradiction, neutral) using the [CLS] token, which summarizes the whole input.

5.4.1 Evidence Identification (EI)

- On top of each [SPAN] token, the model uses a 3-layer MLP to predict whether that span is evidence for the hypothesis or not.
- **Sigmoid activation**: This gives a probability for each span, where:
  - A higher value means the model thinks it's more likely to be relevant evidence.
  - A lower value means the model thinks it's less likely to be relevant.
- The loss for span identification is computed using **cross-entropy loss**. It compares the predicted probabilities for each span ($\hat{s}_i$) with the actual ground truth labels (0 or 1, where 1 means the span is relevant, 0 means it's not).

$$\ell_{\text{span}} = \sum_i (-s_i \log \hat{s}_i - (1 - s_i) \log(1 - \hat{s}_i)) \tag{1}$$

Although there exists no evidence span when NLI label is NOTMENTIONED, we nevertheless incorporate such an example in the evidence identification loss with negative span labels $s_i = 0$.

5.4.2 Natural language inference (NLI)

- We use a setup with the context and hypothesis fed into a Transformer-based encoder followed by an MLP decoder with softmax activation to provide predicted probabilities of the "Entailed", "Contradicted" and "Not Mentioned" labels.
- Even though the "Not Mentioned" label does not require evidence, it has been incorporated into the model in the sense that all spans have ground truth negative likelihoods $s_i = 0$.
- The NLI loss is calculated using **cross-entropy loss:**

$$\ell_{\text{NLI}} = \begin{cases} -\sum_{L \in \{E,C,N\}} y_L \log \hat{y}_L & \text{if any span is evidence} \\ 0 & \text{if no span is evidence} \end{cases} \tag{2}$$

The multitask loss for a single context is then:

$$\ell = \ell_{\text{span}} + \lambda \ell_{\text{NLI}} \tag{3}$$

where λ is a hyperparameter that controls the balance between the two losses.
We mix contexts from different documents during training, thus contexts from a single document may appear in different mini batches.
Since each document is predicted as multiple contexts, results from these contexts have to be aggregated to obtain a single output for a document. For the evidence identification, we simply take the average of span probabilities over different model outputs:

$$\hat{s}_i^* = \frac{1}{M_i} \sum_m \hat{s}_i^m,$$

where M_i is the number of contexts that have the full i-th span in its context.
For NLI, we weighted the NLI probabilities by the sum of the span probabilities:

$$\hat{y}^* = \frac{1}{\sum_m \frac{1}{S_m} \sum_i \hat{s}_i^m} \sum_m \left(\hat{y}^m \cdot \frac{1}{S_m} \sum_i \hat{s}_i^m \right),$$

where S_m is the number of `[SPAN]` tokens in the m-th context. This is based on an intuition that contexts with evidence spans should contribute more to NLI.

Evaluation:

Table 3: Evaluation Metrics

Metric	Value
Accuracy	0.8271
Precision	0.7124
Recall	0.7524
F1 Score	0.7319
mAP	0.8701
P@R80	0.6539

6 Long short-term memory (LSTM)

- Contracts often contain long sequences of text, and transformer-based models like BERT have limitations on sequence length (typically 512 tokens). Using an LSTM allows processing the entire contract without truncation, ensuring no loss of important context.
- Created input pairs by combining the contract text and hypothesis with a separator [SEP].

6.1 Model Architecture

- The model is a bidirectional LSTM that captures context from both past and future words, improving the understanding of long-range dependencies in the text
- The final hidden states from both directions are concatenated and passed through a fully connected layer to classify the relationship between the contract and hypothesis into one of three classes: *Entailment*, *Contradiction*, or *Not Mentioned*.

Evaluation

Metric	Value
Accuracy	0.60
Precision	0.65
Recall	0.54
F1 Score	0.58

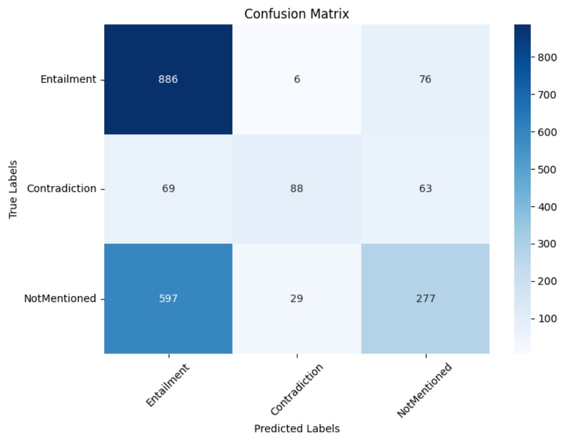


Figure 7: Confusion Matrix for LSTM

7 Retrieval-Augmented Generation

This approach addresses the ContractNLI problem using a **Retrieval-Augmented Generation (RAG)** methodology. It is designed to identify evidence for a given hypothesis within a contractual document and classify their relationship as entailment, contradiction, or neutral. The process is as follows:

1. **Semantic Embedding of Documents**: Each document is divided into sentences, and a Sentence Transformer model is used to generate embeddings for these sentences. This prepares the document for efficient semantic search.
2. **Query-Based Semantic Search**: The hypothesis acts as a query. For a selected document, the top-k sentences most semantically similar to the hypothesis are retrieved based on cosine similarity in the embedding space.
3. **Natural Language Inference (NLI)**: The hypothesis and each of the retrieved sentences are passed as input pairs to a BERT-based NLI model. This model predicts the relationship (entailment, contradiction, or neutral) between the hypothesis and the retrieved sentences.
4. **Evidence Identification**: The sentence that supports the hypothesis with the highest probability (entailment) is considered the strongest evidence for the claim. This sentence is selected based on its classification probability from the NLI model.
5. **Result Aggregation**: The NLI results for the top-k sentences are used to produce a final classification for the ContractNLI task. This can involve aggregating scores or selecting the most relevant classification based on downstream requirements.

This approach combines **semantic retrieval** for selecting evidence and **deep learning-based classification** for analyzing relationships, ensuring scalability and accuracy in identifying contractual obligations and relationships.

Evaluation:

Table 4: Evaluation Metrics

Metric	**Value**
Accuracy	0.614
Precision	0.6377
Recall	0.6141
F1 Score	0.5869

8 Conclusion

8.1 Baseline Approaches

Whole Document NLI: This approach is effective for short documents but suffers from limitations due to 512-token truncation. It is best suited for cases where critical information is located at the beginning of the document.
Span TF-IDF: A lightweight and interpretable baseline for evidence identification. The SVM variant outperforms cosine similarity in distinguishing relevant spans but lacks the ability to capture complex semantic relationships.

8.2 Span-Based Approaches

NLI with Spans: By focusing on specific sections of the document, this approach improves NLI accuracy. It is particularly effective for long contracts with clear span boundaries.
EI with Spans: This method uses binary classification for fine-grained evidence extraction. It is suitable for tasks requiring high-confidence evidence identification.
SpanNLI-BERT: The most effective span-based approach, leveraging dynamic context segmentation. It excels at joint NLI and EI, especially in scenarios with dense or complex evidence requirements.

8.3 Span-Free Approaches

LSTM: Processes entire documents without truncation, making it suitable for very long contracts. However, it underperforms transformer models in terms of precision and F1 score.
RAG: Combines semantic retrieval with NLI, making it scalable for large datasets. Its performance depends heavily on the quality of the retrieval mechanism.

8.4 Key Recommendations

- **SpanNLI-BERT**: Recommended for tasks requiring both NLI and EI with high accuracy and robust performance.
- **RAG**: A scalable alternative for large-scale evidence retrieval and NLI tasks.
- **Baseline Models**: Suitable for quick benchmarks or resource-constrained environments.