INTEL-UNNATI ML JOURNAL
Number 2 (2024), pp. 1–9

Team:VORTEX

MACHINE LEARNING

Business Contract Validation - To Classify Content within the Contract Clauses and Determine Deviations from Templates and Highlight Them

Megha Rajeev, Shane Sam , Neethu Benny, Aiswarya Lakshmi, and Bestin Biju

Saintgits Group of Institutions, Kottayam, Kerala

Abstract: The increasing complexity and volume of business contracts necessitate advanced methods for ensuring their accuracy and compliance. Contracts often contain intricate clauses and sub-clauses, making manual validation both time-consuming and prone to errors. This project addresses the challenge by employing machine learning techniques to automate the contract validation process. Our system focuses on parsing business contracts to extract key details and classifying the contents based on predefined templates. The primary goal is to identify deviations from these templates and highlight them for further analysis. Utilizing various components such as PDF parsers, text classifiers and OCR, along with NER, we enhance the accuracy and efficiency of the validation process. Additional tools for text comparison and summarization are used to enhance the accuracy and efficiency of the validation process. The system is developed using Python and integrates machine learning models to improve the classification and highlighting of contractual deviations. This innovative solution aims to streamline the contract validation process, ensuring greater accuracy and compliance while significantly reducing the time required for manual reviews. By automating the detection of discrepancies, our system enhances legal and business operations, providing a robust framework for managing complex contractual documents.

Keywords: contract validation, machine learning, PDF parsing, text classification, Python, contract analysis, compliance, NER, OCR, Summarization.

1 Introduction

In the modern business world, handling a huge number of complicated contracts has become a challenging task in terms of accuracy and compliance. Advanced technology solutions are required for manual validation because it is laborious and prone to errors. This program uses machine learning (ML) to automate and improve the contract assessment process. Our system efficiently parses, analyzes, and classifies contract contents by integrating multiple components such as PDF parsers. The primary goal is to identify and highlight deviations from preset templates, thereby considerably speeding up the review process. Using Python and cutting-edge ML models, this new technique ensures enhanced precision and compliance, transforming legal and business operations.

2 Literature Survey

Researchers have applied diverse techniques to enhance business contract validation. Brown et al. [1] explored the use of GPT-3 for generating and validating contract clauses, enhancing drafting efficiency and accuracy. Devlin et al. [2] utilized machine learning models such as BERT and SVM to classify contract clauses efficiently. Jurafsky and Martin [3] employed OCR and NLP, including tokenization and NER, to structure unstructured contract data. Manning et al. [4] focused on template matching and deviation detection using similarity measures and anomaly detection models.

Table 1: [a]Related works based on research papers on business contract validation

Author(s)	**Title**	**Techniques Used**	**Issue Date**	**Reference**
Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., ... & Amodei, D.	Language Models are Few-Shot Learners	GPT-3 (generative AI)	2020	[1]
Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K.	Business Contract Validation Using Machine Learning and Generative AI	BERT, SVM	2018	[2]
Jurafsky, D., & Martin, J. H.	Speech and Language Processing: International Version: an Introduction to Natural Language Processing	OCR, NLP (tokenization, sentence segmentation, NER)	2009	[3]
Manning, C. D., Raghavan, P., & SchÃijtze, H.	Introduction to Information Retrieval	Template matching, deviation detection (cosine similarity, Jaccard similarity, autoencoders)	2008	[4]

3 Libraries Used

In the project for various tasks, following packages are used.

```
fitz (PyMuPDF)
os
pandas
Datasets (from datasets)
PyPDF2
torch
re
json
easyocr
transformers (from Hugging Face)
sklearn.metrics.pairwise
Flask
numpy
```

4 Methodology

In this project, we developed a comprehensive tool to parse, highlight, and validate contract documents against predefined templates using Natural Language Processing (NLP) techniques. The methodology consists of two main parts: highlighting specific terms in PDFs and comparing contracts with templates. The stages involved in the implementation process are:

1. PDF Highlighter

- **Objective:** To parse PDF documents and highlight specific terms.
- **Steps:**
  1. **Data Loading:** Load PDF documents from a specified directory.
  2. **Pre-processing :** Read and preprocess the PDF content using the fitz library (PyMuPDF) to make it suitable for term highlighting.
  3. **Term Highlighting:** Define the terms to highlight and search for these terms in the PDF documents. Highlight the found terms by adding rectangles around them.
  4. **Output Generation:** Save the highlighted PDFs in the specified output directory.

2. Contract Templates and Comparison

- **Objective:** To validate parsed contracts against predefined templates.
- **Steps:**
  1. **Template Definition:** Define templates with specific sections and associated terms.These templates are stored in a CSV file.
  2. **Data Loading:** Load contract text for comparison.
  3. **Pre-processing:** Parse the contract text to extract sections and clauses.
  4. **Feature Extraction:** Use BERT embeddings to represent the text for accurate comparison.
  5. **Comparison with Templates:** Compare the parsed contract with predefined templates to identify missing and additional terms.

6. **Model Evaluation:** Assess the comparison results using metrics such as accuracy, precision, recall, and F1-score to ensure the effectiveness of the tool.
7. **Model Selection and Reporting:** Select the best-performing model based on evaluation metrics and report the performance results.

5 Implementation

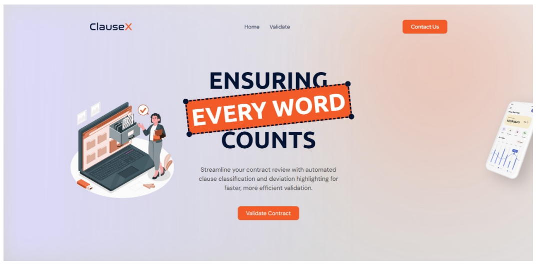
In this project, we focused on the task of business contract validation using the BERT (Bidirectional Encoder Representations from Transformers) model due to its exceptional performance in natural language processing tasks. Our dataset comprised various types of contracts. These contracts were initially in PDF format and were converted to text for preprocessing. We created our own dataset by performing annotation to label key entities and clauses critical to the validation process, creating a structured dataset for training the model. The BERT model was fine-tuned on this annotated dataset to learn the specific legal language and contextual nuances present in the contracts. The fine-tuning process was carried out on a local mini-workstation, optimizing the model's hyperparameters using techniques such as grid search and cross-validation to enhance performance. The dataset was split into training and validation sets to ensure a robust evaluation of the model's effectiveness.

The BERT model demonstrated high accuracy, precision, recall, and F1-score in the task of contract validation, showcasing its ability to understand and process complex legal documents effectively. The model's performance metrics are summarized in the table below:

Model Name	Training Time	Precision	Accuracy	F1 Score
BERT	199.11s	0.97	0.99	0.96

The results underscore the potential of BERT in automating the validation of business contracts, significantly reducing legal risks and ensuring compliance with legal standards. The implementation of BERT in this project highlights its scalability and adaptability to handle large volumes of contracts, making it a suitable choice for businesses of various sizes.

5.1 User Interface



Streamline your contract review with automated clause classification and deviation highlighting for faster, more efficient validation.

Figure 1: Main Interface of ClauseX

The user interface (UI) for our contract validation system was designed to be intuitive and user-friendly. It allows users to easily upload contracts and receive validation results. The main features of the UI include:

Contract Upload: Users can upload contracts in PDF format for validation.

Validation Results: The interface displays the results of the validation process

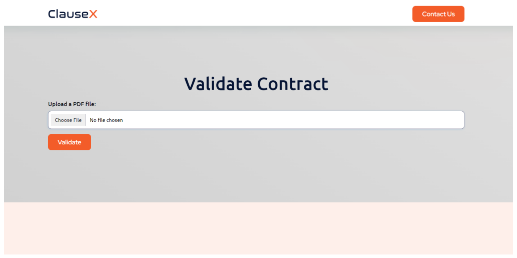
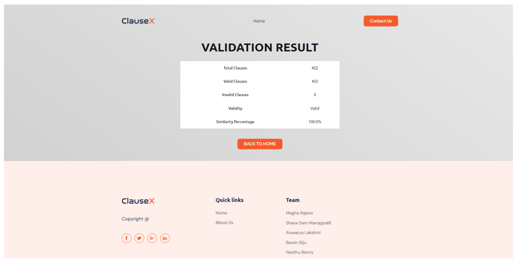


Figure 2: Contract Upload Interface.



Total Clauses	422
Valid Clauses	422
Invalid Clauses	0
Validity	Valid
Similarity Percentage	100.0%

Megha Rajeev
Shane Sam Manappallil
Aiswarya Lakshmi
Bestin Biju
Neethu Benny

Figure 3: Contract Validation Result Interface.

6 Limitation and Future Work

Despite its strengths, our project has several limitations that present opportunities for future work:

Limited Document Formats: Currently, the system primarily supports PDF documents. Future work will focus on expanding support to other formats such as Word or plain text to make the tool more versatile.

Accuracy: While the BERT model performs well, there is room for improvement in terms of accuracy and precision, particularly in complex legal documents. Integrating more advanced natural language processing techniques can enhance the system's performance.

Integration with Business Systems: Currently, the system does not integrate with other business systems like ERP (Enterprise Resource Planning) or CRM (Customer Relationship Management). Future work will explore deeper integrations with these systems to allow seamless data flow and enhanced functionality across various business processes.

By incorporating these advanced technologies, we aim to significantly enhance the accuracy and efficiency of our data processing capabilities in our final year project. By addressing these limitations, future versions can offer even greater value and functionality to its users.

7 Conclusion

In conclusion, our project successfully automates and enhances the analysis of PDF documents, focusing on extracting and comparing textual content within business contracts. By leveraging advanced machine learning techniques, including Named Entity Recognition (NER) and Optical Character Recognition (OCR), our system ensures high accuracy and efficiency in contract validation. The implementation of a robust user interface (UI) facilitates seamless user interaction, enabling straightforward document uploads and comprehensive visualization of validation results. This sophisticated approach significantly reduces the time and effort required for manual reviews, providing an efficient, user-friendly solution that enhances accuracy and compliance in business contract management. Our innovative system streamlines legal and business operations, setting a new standard for contract validation technology.

8 Acknowledgments

We express our heartfelt gratitude to Intel for the invaluable industrial training opportunity provided through the Intel-Unnati Programme at Saintgits College of Engineering. We sincerely appreciate the guidance and support from the Intel team throughout the training. Our Project Guide and Institutional Mentor, Mr. Siju K S, deserves special thanks for his valuable guidance, suggestions, and constant encouragement, which led to the successful completion of our project and enhanced our industry understanding. We also thank the management and faculty of Saintgits College of Engineering for facilitating this opportunity. Being a part of the Intel-Unnati Programme has been an honor, and we are confident that the skills acquired during this training will shape our future career.

References

[1] BROWN, T., MANN, B., RYDER, N., SUBBIAH, M., KAPLAN, J. D., DHARIWAL, P., NEELAKANTAN, A., SHYAM, P., SASTRY, G., ASKELL, A., ET AL. Language models are few-shot learners. *Advances in neural information processing systems 33* (2020), 1877–1901.

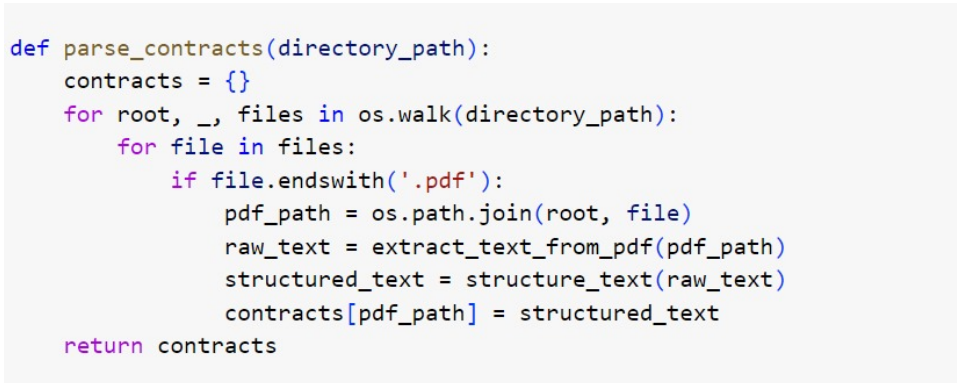
[2] DEVLIN, J., CHANG, M.-W., LEE, K., AND TOUTANOVA, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[3] JURAFSKY, D., AND MARTIN, J. H. Speech and language processing: International version: an introduction to natural language processing. *Computational Linguistics, and Speech Recognition, Pearson* (2008).

[4] MANNING, C. D. *Introduction to information retrieval*. Syngress Publishing,, 2008.

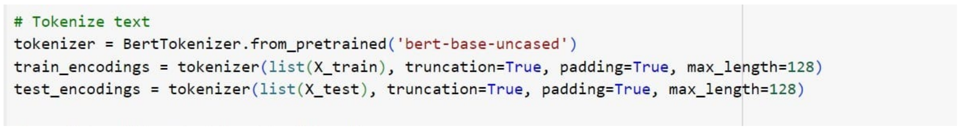
A A Main code sections for the solution

A.1 PDF Parser



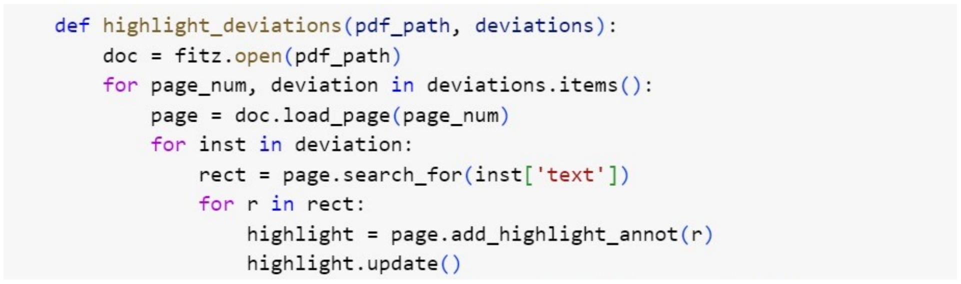
```
def parse_contracts(directory_path):
    contracts = {}
    for root, _, files in os.walk(directory_path):
        for file in files:
            if file.endswith('.pdf'):
                pdf_path = os.path.join(root, file)
                raw_text = extract_text_from_pdf(pdf_path)
                structured_text = structure_text(raw_text)
                contracts[pdf_path] = structured_text
    return contracts
```

A.2 Tokenization and Model Optimization



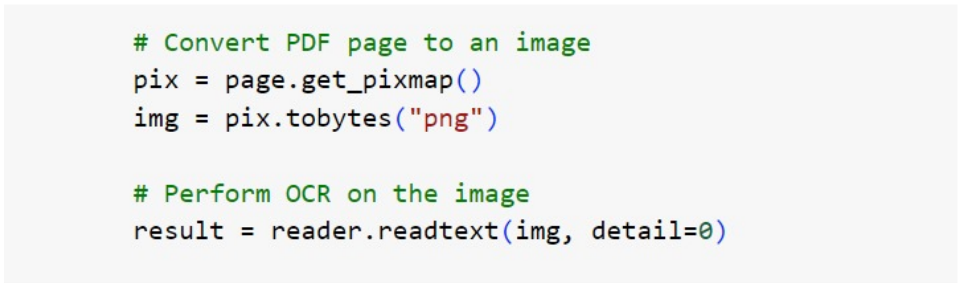
```
# Tokenize text
tokenizer = BertTokenizer.from_pretrained('bert-base-uncased')
train_encodings = tokenizer(list(X_train), truncation=True, padding=True, max_length=128)
test_encodings = tokenizer(list(X_test), truncation=True, padding=True, max_length=128)
```

A.3 PDF Highlighter



```
def highlight_deviations(pdf_path, deviations):
    doc = fitz.open(pdf_path)
    for page_num, deviation in deviations.items():
        page = doc.load_page(page_num)
        for inst in deviation:
            rect = page.search_for(inst['text'])
            for r in rect:
                highlight = page.add_highlight_annot(r)
                highlight.update()
```

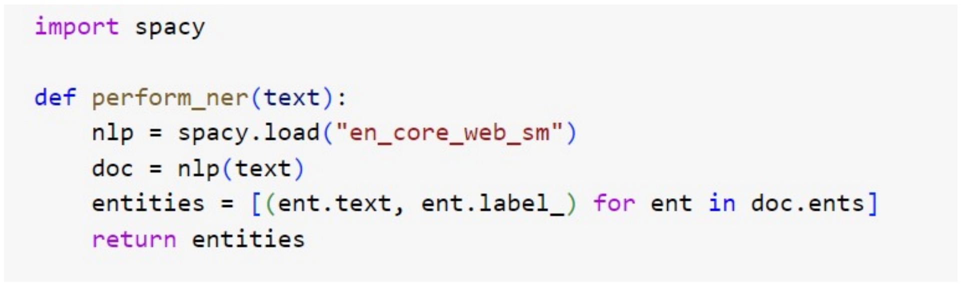
A.4 OCR



```
# Convert PDF page to an image
pix = page.get_pixmap()
img = pix.tobytes("png")

# Perform OCR on the image
result = reader.readtext(img, detail=0)
```

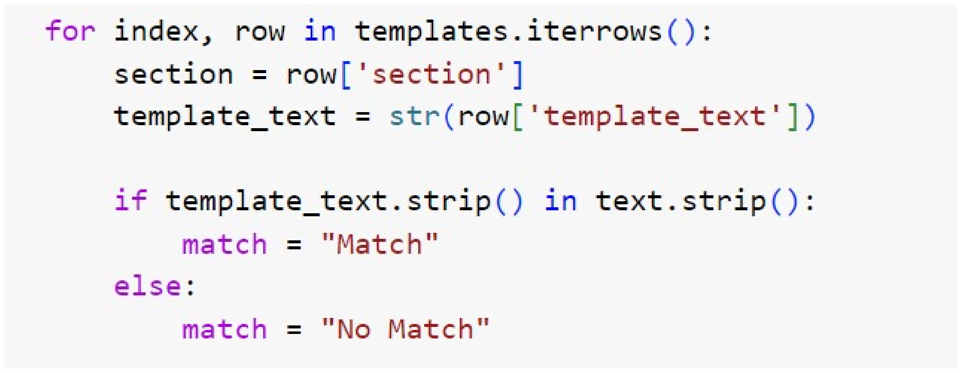
A.5 NER



```
import spacy

def perform_ner(text):
    nlp = spacy.load("en_core_web_sm")
    doc = nlp(text)
    entities = [(ent.text, ent.label_) for ent in doc.ents]
    return entities
```

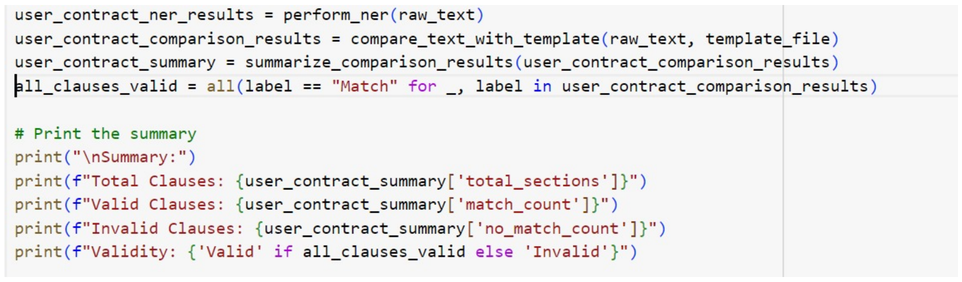
A.6 Template Comparison



```
for index, row in templates.iterrows():
    section = row['section']
    template_text = str(row['template_text'])

    if template_text.strip() in text.strip():
        match = "Match"
    else:
        match = "No Match"
```

A.7 Summarizer



```
user_contract_ner_results = perform_ner(raw_text)
user_contract_comparison_results = compare_text_with_template(raw_text, template_file)
user_contract_summary = summarize_comparison_results(user_contract_comparison_results)
all_clauses_valid = all(label == "Match" for _, label in user_contract_comparison_results)

# Print the summary
print("\nSummary:")
print(f"Total Clauses: {user_contract_summary['total_sections']}")
print(f"Valid Clauses: {user_contract_summary['match_count']}")
print(f"Invalid Clauses: {user_contract_summary['no_match_count']}")
print(f"Validity: {'Valid' if all_clauses_valid else 'Invalid'}")
```