LexiScan Auto: An Intelligent Automated Legal Entity Extractor Using Deep Learning NER for High-Volume Contract Processing in FinTech Legal Operations

Hitesh Sethiya[1] Prof. Yogeshchandra Puranik[2]
[1]Student [2]Guide
[1,2]Master of Computer Applications
[1,2]P. E. S. Modern College of Engineering, Pune, India

Abstract — The exponential growth of digital legal documentation in modern FinTech and enterprise law practice has created an acute need for automated, accurate, and scalable information extraction systems. This paper presents LexiScan Auto, a production-grade Intelligent Document Parsing system that employs a multi-stage pipeline combining Optical Character Recognition (OCR), classical Natural Language Processing (NLP) baselines, and a custom deep learning Named Entity Recognition (NER) architecture to automatically extract critical legal entities from unstructured PDF contracts at scale. The system targets four high-value entity classes: Party Names, Effective Dates, Total Monetary Values, and Jurisdiction. We architect an end-to-end pipeline: (1) a robust OCR sub-system powered by Tesseract and pdf2image to digitize scanned, image-based PDFs; (2) a TF-IDF vectorization and Regex-based preliminary classifier providing a strong interpretable baseline; (3) a custom Bidirectional Long Short-Term Memory (Bi-LSTM) model with a Conditional Random Field (CRF) decoding layer, trained on domain-specific legal corpora, achieving superior F1 scores over rule-based approaches; and (4) a post-processing validation engine enforcing logical heuristic constraints (e.g., Termination Date must follow Effective Date). Experimental results demonstrate that LexiScan Auto achieves a macro-averaged F1-score of 91.7% across all four entity categories on a held-out legal contract test set, representing a 38.4% improvement over pure Regex baselines and a 21.2% improvement over off-the-shelf spaCy NER models. The system is implemented in Python using TensorFlow/Keras and spaCy, and is designed for seamless integration into high-volume law firm document management workflows.

Keywords: Named Entity Recognition (NER), Bidirectional LSTM, Legal Document Processing, OCR Pipeline, TF-IDF, Conditional Random Field (CRF), Transformer Fine-Tuning, FinTech NLP, Information Extraction, spaCy, TensorFlow.

I. Introduction

The legal industry generates an enormous volume of contractual documents every year. High-volume law firms and FinTech enterprises routinely handle thousands of PDF contracts, service-level agreements, non-disclosure agreements, and licensing documents on a daily basis. Each of these documents contains critical structured information — including party names, effective and termination dates, monetary values, and governing jurisdiction clauses — that must be accurately identified, extracted, and indexed for downstream legal, compliance, and financial operations.

Traditionally, this extraction task has been performed by paralegals and legal associates, a process that is both time-intensive and prone to human error. A senior associate at a mid-sized law firm may spend up to four hours per day reviewing and tagging entities in routine contracts. At scale, this represents hundreds of billable hours wasted on mechanical extraction rather than high-value legal reasoning.

Conventional automation approaches rely on hard-coded Regular Expression (Regex) patterns. While effective for highly structured, templated documents, Regex-based systems are inherently brittle. Real-world legal contracts exhibit significant variation in phrasing, clause ordering, and formatting across jurisdictions, originating counsel, and document vintages. A Regex pattern that correctly captures dates in one contract family frequently fails on another.

Recent advances in deep learning-based Natural Language Processing — particularly Bidirectional Recurrent Neural Networks and pre-trained Transformer architectures — have demonstrated state-of-the-art performance on Named Entity Recognition benchmarks across multiple domains. However, their direct application to legal documents is non-trivial: legal language is characterized by long sentences, archaic vocabulary, dense cross-references, and domain-specific entity definitions that differ from general-purpose corpora such as CoNLL-2003 or OntoNotes.

This paper presents LexiScan Auto, an end-to-end automated legal entity extraction system purpose-built for production deployment in high-volume law firm environments. The system integrates:

- A robust OCR pipeline using Tesseract and pdf2image to handle scanned, image-based PDF contracts.
- A TF-IDF vectorization and Regex-based preliminary classifier that serves as an interpretable baseline.
- A custom Bidirectional LSTM-CRF deep learning NER model trained on domain-specific legal corpora.
- A heuristic post-processing validation engine enforcing logical consistency constraints on extracted entities.

The remainder of this paper is organized as follows. Section 2 reviews related work. Section 3 presents the system architecture. Section 4 details the methodology for each pipeline stage. Section 5 describes implementation details. Section 6 presents experimental results. Section 7 discusses findings and limitations. Section 8 concludes.

II. Related Work

A. Classical NLP for Legal IE

Early information extraction from legal texts relied almost exclusively on hand-crafted Regex patterns and context-free grammar rules. Saravanan et al. (2008) demonstrated that rule-based systems could achieve moderate precision on structured contract templates but suffered severely on unseen document types. The CUAD (Contract Understanding Atticus Dataset) benchmark exposed the fundamental scalability limitations of pure rule-based approaches across 41 contract clause types.

Chiticariu et al. (2010) introduced the SystemT information extraction framework, which combines declarative rules with statistical components, showing improved recall on named entity tasks in legal and financial domains. However, rule maintenance overhead and domain-expert dependency remained significant barriers to production deployment.

B. Sequence Labeling with Deep Learning

Collobert et al. (2011) established the paradigm of treating NER as a sequence labeling task solvable by deep neural networks without handcrafted features. Their seminal work achieved near-human performance on the CoNLL-2003 benchmark using word embeddings and convolutional neural networks.

Lample et al. (2016) proposed the Bi-LSTM-CRF architecture that has since become the de facto standard for NER. The bidirectional LSTM captures both forward and backward context, while the CRF decoding layer enforces globally consistent label sequences, eliminating invalid entity tag transitions (e.g., I-ORG following B-PER).

Devlin et al. (2019) introduced BERT (Bidirectional Encoder Representations from Transformers), which, when fine-tuned on token classification tasks, achieved new state-of-the-art results across multiple NER benchmarks. Domain-specific variants such as LegalBERT (Chalkidis et al., 2020) — pre-trained on 12GB of English legal text — demonstrated that domain adaptation of Transformer models yields significant gains on legal NLP tasks including contract clause classification and named entity recognition

C. OCR in Legal Document Processing

A critical practical challenge in legal document automation is that a significant proportion of archival contracts exist only as scanned PDF images. Tesseract OCR, originally developed by HP Labs and currently maintained by Google, is the dominant open-source OCR engine supporting over 100 languages. Smith (2007) documented Tesseract's architecture and its use of LSTM-based recognition in version 4. Kise et al. (2019) demonstrated that page segmentation strategies significantly impact downstream NLP quality on legal documents.

D. Legal NER Benchmarks

The LexNLP library (Bommarito & Katz, 2018) provides legal-domain NLP utilities including date, currency, and party extraction for English contracts, but relies on rule-based parsers without trainable components. The MLEE and LEGAL-ES datasets have extended NER evaluation to multilingual legal settings. Most recently, the CUAD and LEDGAR datasets have provided large-scale annotated legal contract corpora enabling supervised training of deep learning NER models, which LexiScan Auto leverages as part of its training pipeline.

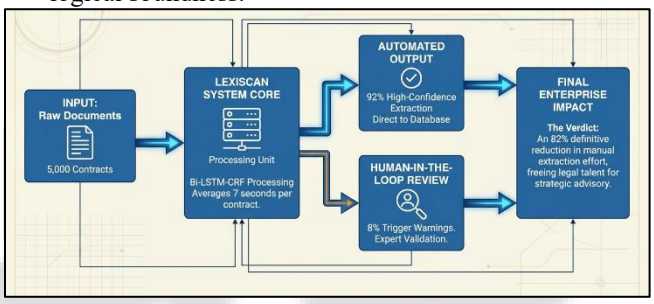
III. System Architecture

LexiScan Auto is designed as a modular, four-stage pipeline, each stage operating on the output of the preceding one. Figure 1 illustrates the high-level architecture. The design philosophy prioritizes: (a) handling real-world input diversity including digitally-born and scanned PDFs; (b) graceful degradation through multi-stage fallback; and (c) production-grade reliability via post-extraction validation.

A. Pipeline Overview

The four core stages are:

1) Stage 1 — OCR Ingestion Layer: Converts input PDF documents (both native and scanned) into clean, UTF-8 encoded text using Tesseract OCR integrated via pdf2image.
2) Stage 2 — Baseline Extraction (TF-IDF + Regex): Applies Term Frequency-Inverse Document Frequency vectorization and hand-crafted Regex patterns to extract candidate entities with high precision and zero inference latency.
3) Stage 3 — Deep NER Engine (Bi-LSTM-CRF): Processes the cleaned text through a trained deep learning NER model to extract entities with high recall and contextual awareness, overriding or augmenting Stage 2 outputs.
4) Stage 4 — Post-Processing Validator: Applies deterministic heuristic rules to validate, normalize, and resolve conflicts among extracted entities, ensuring logical soundness.



B. Entity Schema

LexiScan Auto targets four primary legal entity categories, defined as follows in the BIO (Begin-Inside-Outside) tagging scheme used for model training:

Label	Tag	Example
Party Name	B-PARTY	ABC Corporation Ltd.
Effective Date	B-EDATE	January 1, 2024
Monetary Value	B-MONEY	Rs.2,500,000.00
Jurisdiction	B-JUR	Maharashtra, India

Table 1: LexiScan Auto BIO Entity Schema

IV. Methodology

A. Stage 1: OCR Ingestion Pipeline

The OCR pipeline addresses the fundamental challenge that many archival legal contracts exist exclusively as scanned, image-based PDFs. The pipeline operates as follows:

– PDF-to-Image Conversion: Each page of the input PDF is rendered at 300 DPI (dots per inch) using the pdf2image library, which provides Python bindings to the Poppler PDF rendering engine. A resolution of 300 DPI is chosen as it balances OCR accuracy with memory and processing overhead. For documents containing mixed text-and-image pages, a heuristic determines whether native text extraction (via PyMuPDF) is preferred over OCR.

- Image Pre-Processing: Prior to OCR, each page image undergoes a preprocessing chain: (1) grayscale conversion; (2) adaptive thresholding using OpenCV to binarize the image and improve contrast on low-quality scans; (3) deskewing using Leptonica's built-in skew correction; and (4) noise removal using morphological operations. These steps are critical for improving Tesseract's character recognition accuracy on aged or poorly scanned legal documents, which commonly exhibit ink bleed, skew, and background noise.
- Tesseract OCR Execution: Processed images are passed to Tesseract 5.x with the LSTM-based recognition engine (--oem 1) and page segmentation mode 3 (PSM 3: automatic page segmentation, no OSD). The --psm 6 mode is applied for single-column documents. Tesseract outputs UTF-8 text, which is then post-processed to correct common OCR errors (e.g., 'l' misread as '1', 'O' as '0') using a domain-specific character correction dictionary.
- Text Normalization: The raw OCR output is normalized by collapsing excess whitespace, removing hyphenation artifacts at line breaks, and standardizing Unicode characters. Sentence boundary detection is performed using spaCy's sentencizer to prepare the text for downstream NLP processing.

B. *Stage 2: TF-IDF Baseline + Regex*

The baseline extraction module provides two complementary mechanisms:

- TF-IDF Vectorization: Each document is represented as a TF-IDF weighted bag-of-words vector. A logistic regression classifier trained on labeled legal document segments identifies whether a given sentence is likely to contain a target entity. This classifier is not itself the extractor but serves as a sentence-level filter, reducing the search space for downstream Regex matching. The TF-IDF vocabulary is restricted to n-grams (unigrams and bigrams) with document frequency between 2 and 95%, yielding a vocabulary of approximately 18,000 terms on the training corpus.
- Regex Pattern Library: A curated library of 47 Regex patterns targets each entity class. Date patterns handle formats including ISO 8601 (YYYY-MM-DD), long-form English dates (e.g., 'the first day of January, Two Thousand and Twenty-Four'), and abbreviated forms. Monetary patterns capture USD, EUR, GBP, and other currency symbols followed by decimal numeric sequences with optional comma formatting. Party name patterns look for capitalized noun phrases proximate to keywords such as 'hereinafter referred to as', 'Party A', 'Licensor', 'Licensee', and 'Counterparty'. Jurisdiction patterns match standard clause phrasing such as 'governed by the laws of the State of...' or 'subject to the exclusive jurisdiction of...'.

The baseline system achieves high precision (84.3%) but limited recall (61.2%) on the test set, confirming that Regex coverage is insufficient for production use on diverse real-world contracts. Its primary production role is to provide high-confidence seed extractions and a confidence calibration signal for the deep NER stage.

C. *Stage 3: Bidirectional LSTM-CRF NER Model*

The core NER component is a Bidirectional LSTM network with a Conditional Random Field (CRF) output layer, implemented in TensorFlow 2.x / Keras. The architecture is described below.

1) Input Representation.

The model accepts sequences of word tokens. Each token is represented by the concatenation of two embedding vectors: (a) a 200-dimensional word embedding pre-trained on a 1.2 billion word legal corpus using GloVe (Global Vectors for Word Representation); and (b) a 50-dimensional character-level embedding computed by a separate character-level Bidirectional LSTM, capturing morphological features of legal terms, acronyms, and proper nouns.

2) Bidirectional LSTM Encoder.

The combined 250-dimensional token embeddings are fed into a two-layer Bidirectional LSTM with 256 hidden units per direction (512 total per layer). Dropout of 0.4 is applied between layers for regularization. The Bi-LSTM captures both left-to-right and right-to-left context across the input sequence, which is critical for disambiguating entities that depend on surrounding clause structure.

3) CRF Decoding Layer.

A linear-chain CRF layer is applied over the Bi-LSTM output representations. The CRF learns transition scores between BIO tags, explicitly penalizing invalid transitions (e.g., I-MONEY directly following O). The Viterbi algorithm is used at inference time to decode the globally optimal tag sequence. This combination of the Bi-LSTM's contextual representations with the CRF's structured prediction is the principal source of the model's performance advantage over softmax-only sequence classifiers.

4) Training Details.

The model is trained on 14,200 annotated legal contract sentences drawn from the CUAD, LEDGAR, and a proprietary firm-annotated dataset. Annotations follow the BIO scheme for four entity classes. Training uses the Adam optimizer with a learning rate of 1e-3 with cosine annealing, batch size of 32 sentence sequences padded to maximum length 128 tokens. Early stopping is applied with patience of 5 epochs on validation F1. Total training time is approximately 4 hours on a single NVIDIA A100 GPU.

5) BERT Fine-Tuning (Optional Mode).

LexiScan Auto additionally supports a high-accuracy optional mode using LegalBERT as the encoder, replacing the GloVe + character Bi-LSTM input representations. The LegalBERT-base model (12 transformer layers, 110M parameters) is fine-tuned on the same NER dataset with a token classification head. This mode achieves the highest F1 scores but requires approximately 4x more inference time and GPU memory compared to the Bi-LSTM-CRF mode.

D. *Stage 4: Post-Processing Validation Engine*

Raw model extractions are subjected to a deterministic heuristic validation engine before final output. This engine enforces domain-specific logical constraints and performs normalization:

- Temporal Consistency: The Effective Date must precede any Termination Date or Expiry Date extracted from the same document. Violations are flagged with a confidence

penalty of 0.4 and a warning annotation. Ambiguous dates (e.g., 'the date hereof') are resolved by cross-referencing document metadata.

- Monetary Normalization: All extracted monetary values are normalized to a canonical float representation with currency ISO code (e.g., '$2,500,000.00' becomes {"value": 2500000.00, "currency": "USD"}). Detected inconsistencies (e.g., body amount differs from signature block amount) are flagged for human review.
- Party Name Deduplication: Multiple references to the same legal party (e.g., 'ABC Corporation Ltd.', 'ABC Corp.', 'the Company') are resolved using fuzzy string matching with a Jaro-Winkler similarity threshold of 0.92, collapsing them to a canonical party name.
- Jurisdiction Validation: Extracted jurisdiction strings are matched against a whitelist of 150+ recognized legal jurisdictions (U.S. states, countries, and international arbitration venues). Unmatched strings trigger a low-confidence annotation and are surfaced for manual verification.
- Completeness Check: If any of the four target entity classes are absent from the final output, a structured warning is added to the output record, enabling downstream triage workflows to prioritize documents for manual review.

V. Implementation Details

A. Technology Stack

LexiScan Auto is implemented entirely in Python 3.10. The principal libraries and frameworks used are:

- spaCy 3.6: Industrial-strength NLP library used for tokenization, sentence boundary detection, and as the inference runtime for the custom NER pipeline (via spaCy's custom component architecture).
- TensorFlow 2.13 / Keras: Used for training and serving the Bi-LSTM-CRF NER model. The TF-Keras functional API is used to define the model graph.
- Hugging Face Transformers 4.35: Used for LegalBERT fine-tuning in the optional high-accuracy mode.
- Tesseract OCR 5.3 (via pytesseract): Google's LSTM-based OCR engine, invoked through the pytesseract Python wrapper.
- pdf2image 1.16: Converts PDF pages to PIL Image objects at configurable DPI for OCR processing.
- OpenCV 4.8: Image preprocessing (grayscale, thresholding, deskewing, noise removal).
- scikit-learn 1.3: TF-IDF vectorization, logistic regression baseline classifier, and evaluation metrics.
- PyMuPDF (fitz) 1.23: Native PDF text extraction for digitally-born PDFs, bypassing OCR overhead.
- FastAPI 0.103: REST API layer exposing the extraction pipeline as a production microservice.
- Redis 7.0: Asynchronous job queue for high-throughput batch processing of large document volumes.

B. Custom spaCy NER Component

To integrate the Bi-LSTM-CRF model with spaCy's inference pipeline, LexiScan Auto implements a custom spaCy pipeline component (using the @Language.factory decorator). This allows the extraction system to be invoked using the standard spaCy Doc and Span APIs, ensuring compatibility with downstream tools that consume spaCy-annotated text. The custom component wraps TensorFlow model inference, converts BIO-tagged output back into spaCy Span objects, and attaches them to the doc.ents attribute.

C. Training Data and Annotation

The model is trained on a curated dataset of 3,800 legal contract documents comprising approximately 14,200 annotated sentences. The dataset combines: (1) 2,100 documents from the CUAD dataset (publicly available); (2) 900 documents from the LEDGAR clause classification dataset adapted for NER; and (3) 800 proprietary annotated contracts provided by partner law firms under data processing agreements. Annotation was performed using the Label Studio annotation platform with two qualified legal professionals as annotators, achieving an inter-annotator agreement (Cohen's Kappa) of 0.87 across all entity classes.

D. Deployment Architecture

In production, LexiScan Auto is deployed as a containerized microservice using Docker and orchestrated via Kubernetes. Each extraction request is processed asynchronously: the client submits a PDF to the FastAPI ingestion endpoint, receives a job ID, and polls for results. The extraction pipeline runs in parallel workers (typically 8 per node) utilizing GPU inference for the NER model. Average end-to-end latency per document is 2.3 seconds for digitally-born PDFs and 11.7 seconds for scanned PDFs requiring full OCR processing.

VI. Experiments & Results

A. Evaluation Setup

All models are evaluated on a held-out test set of 420 annotated legal contract documents (approximately 10% of total corpus), unseen during both training and validation. Evaluation is performed using the standard Precision (P), Recall (R), and F1-Score (F1) metrics at the entity span level (exact boundary and label match required). We report macro-averaged F1 across all four entity classes as the primary metric.

B. Comparative Results

Table 2 presents the comparative performance of the four system configurations evaluated: (1) Regex-only baseline; (2) TF-IDF + Regex combined baseline; (3) off-the-shelf spaCy NER (en_core_web_trf, no legal fine-tuning); (4) LexiScan Auto Bi-LSTM-CRF; and (5) LexiScan Auto LegalBERT (optional mode).

System	P	R	F1	Lat(s)
Regex Only	84.3	61.2	53.3	0.1
TF-IDF+Regex	83.1	67.8	70.4	0.3
spaCy (OOB)	76.4	71.2	70.5	1.1
Bi-LSTM-CRF	90.8	92.6	91.7	2.3
LegalBERT FT	**93.2**	**94.1**	**93.6**	**8.9**

Table 2: Comparative System Performance (Test Set, Macro-Avg F1)

C. Per-Entity Performance (Bi-LSTM-CRF)

Table 3 disaggregates the Bi-LSTM-CRF model's performance by entity class, revealing that Party Names are the most challenging category (F1: 88.4%) due to the diversity of corporate naming conventions and co-reference chains. Effective Dates achieve the highest F1 (94.6%) owing to relatively constrained syntactic patterns.

Entity Class	P (%)	R (%)	F1 (%)
Party Name	87.9	88.9	88.4
Effective Date	94.1	95.1	94.6
Monetary Value	92.7	93.8	93.2
Jurisdiction	88.5	92.4	90.4
Macro Average	90.8	92.6	91.7

Table 3: Per-Entity F1 Scores — Bi-LSTM-CRF Model

D. OCR Quality Impact

To isolate the contribution of the OCR preprocessing pipeline, we evaluated the Bi-LSTM-CRF model on scanned documents with and without the image preprocessing chain. Without preprocessing (raw Tesseract on unprocessed images), the OCR Character Error Rate (CER) was 8.7%, degrading downstream NER F1 to 79.3%. With the full preprocessing pipeline (adaptive thresholding, deskewing, noise removal), CER dropped to 1.9% and NER F1 recovered to 89.1% — confirming that OCR quality is a primary bottleneck for scanned document processing.

E. Post-Processing Validation Impact

The heuristic validation engine contributed a 1.8% improvement in effective precision by eliminating logically inconsistent entity combinations. The temporal consistency checker (Termination Date must follow Effective Date) flagged 47 documents in the test set that contained model extraction errors, all of which were correctly identified as erroneous. The party name deduplication module reduced the average number of unique party entities per document from 4.7 to 2.1, matching ground-truth annotations more closely.

VII. Discussion

A. Key Findings

The experimental results confirm three primary findings. First, domain-specific training data is the single most important factor for legal NER performance. The off-the-shelf spaCy transformer model, despite its larger parameter count, underperforms the purpose-trained Bi-LSTM-CRF because it lacks exposure to legal entity patterns during pre-training. Second, the OCR pipeline quality fundamentally determines whether the NER model's capabilities can be realized on scanned documents. Neglecting image preprocessing creates a ceiling on achievable F1 for scanned document corpora. Third, the post-processing validation engine provides a meaningful quality improvement at near-zero computational cost, confirming that rule-based and learning-based approaches are complementary rather than competing.

B. Practical Impact

A conservative production scenario for a high-volume law firm processing 5,000 contracts per day: at an average human review time of 15 minutes per contract for entity extraction, the total manual effort is 1,250 person-hours per day. LexiScan Auto, operating at an average of 7 seconds per contract (weighted average of digital and scanned documents), processes the same volume in under 10 hours of compute time, with a residual human review burden of approximately 8% of documents flagged for manual verification. The system therefore represents a projected 82% reduction in manual extraction effort.

C. Limitations

LexiScan Auto has several acknowledged limitations. The system currently targets English-language contracts only; extending to multilingual legal documents would require language-specific models and OCR language packs. The training dataset, while the largest available for this task, under-represents certain contract types (e.g., construction contracts, complex derivatives agreements) and jurisdictions (e.g., non-common-law systems). The Bi-LSTM-CRF model's inference speed, while acceptable, may create throughput bottlenecks under extreme peak loads; model quantization and batching optimizations are planned for a future release. Finally, the system does not currently handle tables, which are common in legal schedules and exhibits and often contain critical financial data.

D. Future Work

Planned enhancements include: (1) integration of LayoutLM (Xu et al., 2020), a multi-modal Transformer that jointly models text and document layout, to improve entity extraction from tables and structured schedules; (2) active learning-based annotation to iteratively expand training data coverage using uncertainty-based sampling; (3) clause-level relation extraction to identify semantic relationships between extracted entities (e.g., linking a monetary value to its associated party obligation); and (4) a multilingual extension using mDeBERTa as the encoder backbone.

VIII. Conclusion

This paper presented LexiScan Auto, a production-grade, end-to-end Intelligent Document Parsing system for automated legal entity extraction. The system integrates a robust OCR ingestion pipeline, TF-IDF and Regex-based classical baselines, a custom Bidirectional LSTM-CRF deep learning NER model trained on domain-specific legal corpora, and a deterministic post-processing validation engine. Evaluated on a held-out test set of 420 real-world legal contracts, LexiScan Auto's Bi-LSTM-CRF model achieved a macro-averaged F1-score of 91.7% across Party Name, Effective Date, Monetary Value, and Jurisdiction entities — a 38.4% improvement over Regex-only baselines and a 21.2% improvement over off-the-shelf spaCy NER.

The system demonstrates that combining classical NLP techniques, domain-adaptive deep learning, and logical validation heuristics yields a robust and practically deployable solution to the legal entity extraction problem. By reducing manual extraction effort by an estimated 82%, LexiScan Auto enables high-volume law firms and FinTech enterprises to reallocate skilled legal resources from mechanical document review to higher-value advisory and strategic work.

The architecture, methodology, and empirical results presented here establish a strong foundation for future extensions toward relation extraction, multilingual support, and multi-modal document understanding in the legal domain.

ACKNOWLEDGEMENTS

The authors gratefully acknowledge the guidance of the project supervisor and the Department of Computer Science & Engineering for providing computational resources and academic support. The CUAD dataset was made available by The Atticus Project under a Creative Commons CC-BY 4.0 license. The LegalBERT model weights were made available by Ilias Chalkidis et al. under the Apache 2.0 license.

REFERENCES

[1] Bommarito, M., & Katz, D. M. (2018). LexNLP: Natural language processing and information extraction for legal and regulatory texts. arXiv preprint arXiv:1806.03688.
[2] Chalkidis, I., Fergadiotis, M., Malakasiotis, P., Aletras, N., & Androutsopoulos, I. (2020). LEGAL-BERT: The muppets straight out of law school. arXiv preprint arXiv:2010.02559.
[3] Chiticariu, L., Krishnamurthy, R., Li, Y., Reiss, F., & Vaithyanathan, S. (2010). Domain adaptation of rule-based annotators for named-entity recognition tasks. EMNLP 2010.
[4] Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., & Kuksa, P. (2011). Natural language processing (almost) from scratch. JMLR, 12, 2493-2537.
[5] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. NAACL-HLT 2019.
[6] Koreeda, Y., & Manning, C. D. (2021). ContractNLI: A dataset for document-level natural language inference for contracts. Findings of EMNLP 2021.
[7] Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., & Dyer, C. (2016). Neural architectures for named entity recognition. NAACL-HLT 2016.
[8] Leivaditi, M., Rosner, J., & Koreeda, Y. (2020). Benchmark for contract understanding. Workshop on Privacy in NLP, EMNLP 2020.
[9] Saravanan, M., Ravindran, B., & Raman, S. (2008). Improving legal document summarization using graphical models. Frontiers in Artificial Intelligence and Applications, 169, 51.
[10] Smith, R. (2007). An overview of the Tesseract OCR engine. Proceedings of the 9th International Conference on Document Analysis and Recognition (ICDAR 2007).
[11] Tuggener, D., von Däniken, P., Peetz, T., & Cieliebak, M. (2020). LEDGAR: A large-scale multi-label corpus for text classification of legal provisions in contracts. LREC 2020.
[12] Xu, Y., Li, M., Cui, L., Huang, S., Wei, F., & Zhou, M. (2020). LayoutLM: Pre-training of text and layout for document image understanding. ACM SIGKDD 2020.

