

# HybEx-Law: A Hybrid Neural-Symbolic System for Legal Aid Eligibility Determination

Mohit Kumar\*, Diya Ravishankar\*, Yuganshu Kumar\* and Sudhakaran G†

\*Electronics and Computer Engineering

†School of Electronics Engineering (SENSE)

Vellore Institute of Technology, Chennai, Tamil Nadu

Emails: {dravishankar04, mohit.kr1103, yuganshu.yk}@gmail.com, sudhakaran.g@vit.ac.in

**Abstract**—This paper presents HybEx-Law, a system that blends neural models with rule-based reasoning to determine whether someone qualifies for legal aid. The framework brings together five components: a multitask Transformer model (EnhancedLegalBERT), a standalone eligibility classifier, a domain classifier, a graph-based reasoning module (LegalGAT), and a Prolog-driven rule engine. These are coordinated by the IntelligentHybridPredictor, which calibrates their confidence levels using isotonic regression and combines their outputs through a weighted ensemble. We tested 17 different model combinations through a structured ablation study. The results show a clear advantage when all five components are used together. This full hybrid setup reaches an F1-score of 0.985 and a precision of 98.48%, outperforming the best four-model neural ensemble, which reaches an F1-score of 0.982. The study also highlights the contribution of the symbolic module: although it performs the weakest on its own (F1: 0.825), it consistently catches rule-specific errors that the neural models overlook, functioning as an effective safety layer for rare but important cases.

**Index Terms**—Legal AI, hybrid neural-symbolic, legal aid, knowledge graph, graph neural networks, natural language processing, symbolic reasoning, Prolog, explainable AI

## I. INTRODUCTION

Deciding who is entitled to legal assistance is essential for ensuring meaningful access to justice, particularly for groups that face social or economic disadvantages. In India, these eligibility rules are laid out in the Legal Services Authorities Act of 1987, which considers factors such as income level, demographic category, including women, children, and individuals from SC/ST communities and the nature of the legal issue.

Reviewing each application manually takes considerable time and resources, and the process often results in inconsistent outcomes. These delays place added pressure on legal aid institutions, making it clear that there is a need for tools that can speed up eligibility checks without compromising accuracy or transparency.

HybEx-Law was designed to address this need. It evaluates legal aid eligibility by interpreting a user's natural-language query [2]. Instead of depending exclusively on deep learning models, which can be opaque, or solely on rule-based systems, which may be too rigid, the framework combines both approaches into a unified system. It uses neural models, including variants of LegalBERT [3] to understand free-form text and pairs them with a symbolic reasoning layer. This layer includes a deterministic

rule engine built with SWI-Prolog [5], a legal knowledge graph processed through a Graph Attention Network (LegalGAT), and several specialized Transformer-based classifiers.

After receiving a user's query and its contextual details, the system produces a structured JSON output. This output offers a clear eligibility decision, provides a calibrated confidence score, and outlines the reasoning steps involved, ensuring transparency throughout the process.

The following are this work's primary contributions:

- 1) We present HybEx-Law, a five-component hybrid architecture that combines symbolic logic and neural models to facilitate legal reasoning.
- 2) We create an automated orchestration system that uses a scaled ensembling process to combine model outputs and perform confidence calibration.
- 3) We perform a thorough ablation study across 17 model combinations and show that by employing symbolic rules as a reliable edge case safeguard, the hybrid "expert ensemble" consistently outperforms neural-only approaches.

The structure of HybEx-Law is described in detail, along with the functions of each component, the training and fine-tuning processes, and the outcomes of extensive experiments conducted across a wide evaluation set.

## II. RELATED WORK

The use of AI in the legal sector is rife with challenges, mainly the requirement for high precision and entire explainability. Our work is based on three different areas of research.

### A. AI in Legal Tech

The focus of traditional legal AI was on rule-based systems and, more recently, on information retrieval and document analysis [7], including explainable case-based decision-support tools and legal search interfaces that surface explicit rationales rather than opaque scores [11], [13]. The use of deep learning has grown exponentially; models such as BERT [3] and domain-specific variants like LEGAL-BERT [10] and benchmark suites such as LexGLUE [12] have been applied to tasks like contract review, court outcome prediction, and legal document classification [14], [15]. However, these systems struggle with tasks requiring multi-step or transparent reasoning, because they often cannot provide justifications that are clearly tied to

specific statutes or regulations, which is critical when deciding whether a person should receive legal aid [11], [16], [20].

### B. Hybrid Neural-Symbolic Approaches

Purely neural systems are often weak on transparent reasoning or verifiability, while purely symbolic systems can be brittle and struggle with natural language ambiguity. Hybrid Neural-Symbolic (NeSy) approaches aim to close this gap [8], often by combining neural perception modules with symbolic components that encode rules, constraints, or logical programs. Recent work has explored hybrid agents for legal research and compliance checking, where neural models retrieve or classify documents and symbolic layers verify obligations against formal rule bases [7], [15], [16]. Our approach is in line with modular NeSy architectures [8], applying a neural front-end that extracts structured information which is then passed to a symbolic back-end for reasoning, similar in spirit to contemporary hybrid retrieval-and-reasoning pipelines in legal AI [11], [15]. Unlike systems that focus primarily on reinforcement-learning-based control [9], HybEx-Law operates by constructing explicit, sequential logical steps and prioritizes deterministic reasoning over purely probabilistic inference.

### C. Knowledge Graphs and GNNs in Law

The legal arena is not determined only by statutes, but by the complicated relationships between them, so it is well suited for representation as a Knowledge Graph (KG). Recently, legal knowledge graphs have been used to find relevant case law, represent statutory networks, and support case law analytics and compliance analysis [7], [18], [19]. Graph Neural Networks (GNNs) provide a powerful way of analyzing these relationships and performing inference over graph-structured representations of legal concepts, authorities, and factual entities [18]. A GNN can capture complex patterns within a legal graph that may be too intricate or too numerous to encode manually as rules, such as higher-order dependencies among statutes, precedents, and case facts [18], [19]. The novelty of HybEx-Law lies in its integration: the GNN does not replace the symbolic logic system but instead complements it and joins forces with other components, including the rule-checking module, in a hybrid ensemble that balances structured reasoning with data-driven predictions.

## III. SYSTEM ARCHITECTURE

### A. Overview

The HybEx-Law system features a hybrid and modular architecture comprising neural, graph, and symbolic reasoning components as shown in Fig. 1. The system consists of 5 key factors orchestrated by a central **IntelligentHybridPredictor**.

The workflow is as follows:

- 1) A **Data Preprocessor** normalizes the raw query and structured metadata (e.g., income, social category), performs entity extraction, and tokenizes the text.
- 2) The processed inputs are **sent in parallel** to the five predictive models, which are organized into two distinct paths (see Fig. 1): a **Text Path** (for neural text models)

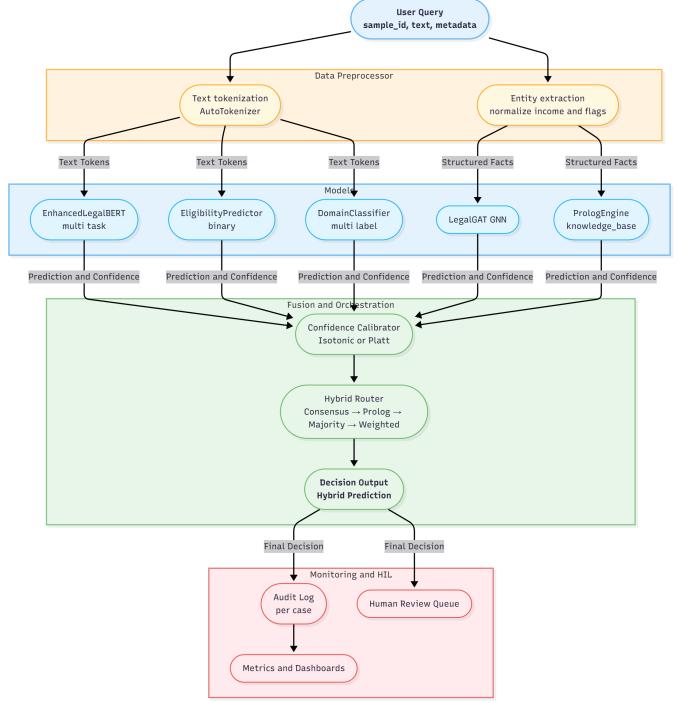


Fig. 1. The HybEx-Law System Architecture. A query is processed by the Data Preprocessor, then fed in parallel to five neural and symbolic models. The Fusion & Orchestration layer arranges and merges these inputs to produce a final decision, which is then sent to monitoring and review queues.

and a **Structured Facts Path** (for GNN and Prolog models).

- 3) The **Hybrid Orchestrator** ingests the outputs (logits or probabilities) from all five models.
- 4) It applies **Confidence Calibration** (e.g., Isotonic Regression) to **align** the confidence scores.
- 5) It combines the calibrated scores using **Ensemble Logic** (e.g., weighted averaging) or applies a priority override.
- 6) It produces a final **Decision Output** (HybridPrediction) that includes a **reasoning trace**, an uncertainty score, and a "requires review" flag.
- 7) This output is logged.

### B. Component Models

The predictive power of the system comes from five distinct components, each designed to capture different aspects of the legal query.

- 1) **EnhancedLegalBERT**: A Transformer that is the major textual model. It performs multiple tasks including eligibility prediction, domain classification, and uncertainty estimation.
- 2) **EligibilityPredictor**: A one-purpose, independent, binary classifier trained on its own eligibility. This offers a broad range of second opinion to the multi-task model.
- 3) **DomainClassifier**: This is a single purpose independent classifier trained only on domain classification (Transformer backbone). This offers a broad range of second opinion to the multi-task model.

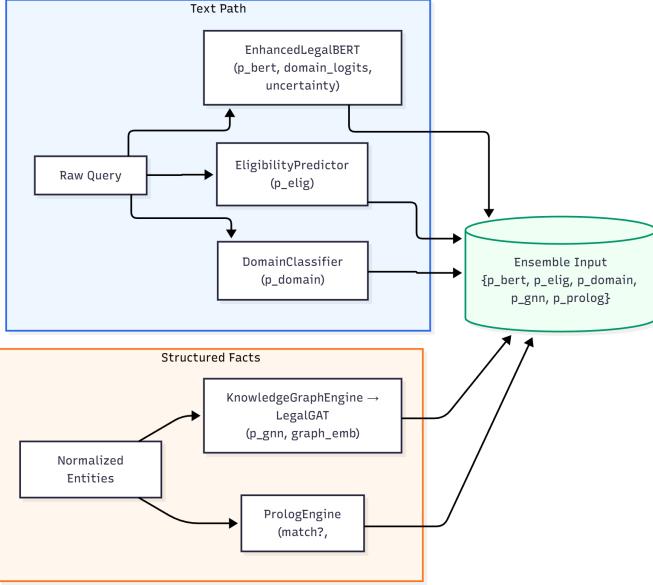


Fig. 2. The Model ensemble dependency map shows the text path in blue and structured facts path in orange, which are fed into the ensemble.

- 4) **KnowledgeGraphEngine (LegalGAT):** This engine constructs a predicate graph of the Prolog knowledge base and adds entity nodes of the case. A Graph Attention Network (GAT), or LegalGAT, is learned. to make predictions of eligibility based on these graph-based features, e.g. to capture relationships (e.g. income thresholds) that text models may miss.
- 5) **PrologEngine:** The engine loads a deterministic body of law rules. It checks the entities that are extracted with these rules. When a rule is similar, it answers a high-confidence decision with a trace of audit-able reasoning.

### C. Hybrid Orchestrator and Calibrator

The **IntelligentHybridPredictor** serves as the central coordinating unit of the system, responsible for bringing together the outputs of all five components. Its prediction workflow is outlined below, along with the key functions it performs:

- **Prolog:** The predictor first consults the PrologEngine. When the engine finds a deterministic, high-confidence rule match, it can be set to override the neural ensemble, effectively treating the symbolic module as a domain specialist for those cases.
- **Confidence Calibration:** Because raw neural logits do not directly correspond to reliable probability values, the system applies a **ConfidenceCalibrator** trained on a validation split to convert each model's output into a calibrated probability using methods such as isotonic regression or Platt scaling.
- **Ensemble Weighting:** After calibration, the predictor computes a final probability by applying learned ensemble weights (e.g.,  $W_{bert}$ ,  $W_{ggn}$ ,  $W_{prolog}$ ). These weights are optimized to minimize validation loss and reflect the relative reliability of each component.

- **Uncertainty Flagging:** The system then evaluates the confidence of the final prediction. If the probability lies close to the decision boundary (0.5) or if there is substantial disagreement among the component models, an uncertainty score is raised and the case is flagged for further human review.

## IV. DATASET

Reliable prediction of legal aid eligibility requires data that reflects both the statutory boundaries defined in the Legal Services Authorities Act and the wide variety of ways individuals describe their legal issues. Since no publicly available, large-scale Indian dataset exists with detailed eligibility labels, domain tags, and structured entities, we built a synthetic yet realistic dataset tailored to this task. The dataset is generated through a controlled pipeline that incorporates the legal provisions of the Act, common Indian legal scenarios, and rule-based annotations created manually for accuracy.

### A. Synthetic but Realistic Legal Scenarios

The core of the dataset is produced by a module, which defines legal templates for five domains: legal aid, family law, consumer protection, fundamental rights, and employment law. For each domain, the generator contains rich text templates and variable grids that encode realistic Indian situations (for example, income disputes, domestic violence, RERA disputes, discrimination, and banking fraud), and then instantiates thousands of concrete queries by sampling combinations of income, social category, relationship status, geographic context, and specific legal problems. This design allows the dataset to cover common patterns as well as rare edge cases such as borderline income thresholds, vulnerable categories, and complex multi-factor situations.

Each synthetic example is stored as a structured *LegalTrainingExample* object with (i) a natural language query; (ii) one or more target domains; (iii) an eligibility label; (iv) auxiliary fields such as extracted entities, Prolog facts, demographics, case complexity, and priority level. The label for legal aid eligibility is not arbitrary: it is generated by a *LegalEligibilityValidator* that encodes strict rules mirroring the income ceilings and vulnerable-category provisions of the Legal Services Authorities Act and its recent amendments. The validator includes explicit logic for zero income, vulnerable groups (for example, women, disabled persons, widows, senior citizens, single parents, and transgender persons), and social-category-specific annual income thresholds, and can also “repair” noisy labels by re-validating each sample against these rules when building the final dataset.

### B. Post-processing and Entity Extraction

To make the dataset useful for both neural models and the symbolic Prolog engine, all generated samples are post-processed by a dedicated preprocessing pipeline. First, the generator writes raw examples to JSON, along with Prolog facts that directly encode structured predicates such as income brackets, social category, and vulnerability indicators. Next,

a separate preprocessing script splits the data into training, validation, and test sets and normalizes the fields expected by the downstream components, ensuring that every example has consistent keys for the hybrid architecture.

Crucially, the final entity fields used by the models are not taken blindly from the synthetic templates. Instead, the *ComprehensiveEntityExtractor* re-parses every query string and re-computes entities directly from the text. The extractor uses targeted regular expressions for income, age, and other numeric fields, as well as pattern-based rules for gender, social category (SC, ST, OBC, BPL, general), and vulnerable statuses such as disability, senior citizen, widowhood, single parenthood, and transgender identity. It overwrites any pre-filled synthetic entities in all three splits, thereby enforcing that entity extraction quality reflects what is realistically possible from user text rather than what the generator “knows” internally.

### C. Coverage and Statistical Balance

The generator is designed to provide both breadth and balance across legal domains and difficulty levels. For each domain, it maintains domain-specific template banks and variation routines that create multiple levels of case complexity (for example, “medium,” “high,” and “very high”) and priority categories (for example, routine vs urgent). Enhanced sampling routines ensure that a significant portion of the dataset focuses on challenging configurations such as income values near statutory thresholds, overlapping eligibility grounds, and multi-domain queries that mention, for instance, both family law and legal aid in a single narrative. The script also tracks statistics such as the proportion of eligible versus not-eligible cases, complexity distribution, and priority distribution, and saves these to a separate statistics file so that class balance and covariate coverage can be inspected and, if necessary, adjusted when curating the final training and evaluation sets.

### D. Why Synthetic Data is Necessary and Useful

Constructing this dataset synthetically is not a purely technical convenience but a legal and practical necessity. Real legal-aid case files are sensitive, often contain personally identifiable information, and are rarely released in sufficiently large, labeled, and machine-readable form. By encoding the Legal Services Authorities Act and domain knowledge into templates and rule-based validators, the generator creates queries that closely resemble real-world narratives from different socio-economic groups and regions, while avoiding the privacy and consent issues that accompany live case data. At the same time, the label generation process is fully auditable: every eligibility label is backed by an explicit reasoning trace (for example, income compared with the correct category-specific threshold or presence of a vulnerable status), which aligns well with the goals of explainable legal AI.

Synthetic data also lets the system observe important but low-frequency phenomena that might be rare in a limited institutional archive, such as edge cases exactly at the income boundary, uncommon vulnerable groups, or complex cross-domain disputes. Because the eligibility labels are derived

from deterministic, human-written rules and then cross-checked by the *LegalEligibilityValidator*, the training data provides a clean signal about how the hybrid system should behave in legally well-understood situations, while the diversity of the templates exposes the neural and graph components to the variety of language and factual combinations expected in real deployments. This combination of rule-grounded labels and realistic textual variation makes the synthetic dataset a strong stand-in for restricted real data and is fundamental to the demonstrated performance and robustness of HybEx-Law.

## V. METHODOLOGY

### A. Hybrid Prediction Pipeline

The core logic of the system is encapsulated in the hybrid prediction pipeline implemented by the Intelligent Hybrid Predictor. This procedure details the flow from raw case data to a final, calibrated hybrid prediction object.

First, the pipeline queries the Prolog-based reasoning engine. If a high-confidence deterministic rule is found and configured to override, the process returns immediately with a Prolog override decision. If not, the pipeline runs the graph neural network and all neural text models in parallel. The raw outputs (logits or confidences) from all components are then converted to probabilities and passed through a confidence calibration module (for example, an isotonic regression model) to place them on a uniform scale. These calibrated probabilities are combined using learned ensemble weights to produce a final ensemble probability. The system then calculates an uncertainty score and sets a review-required flag if the final probability is too close to the 0.5 decision boundary or if model disagreement is high. Finally, it returns a comprehensive hybrid prediction record containing the final decision, confidence scores, and review status.

### B. Model Architectures

1) *EnhancedLegalBERT* (*Multi-task*): This is the core textual model. Its backbone is a generic transformer-based encoder (for example, a Legal-BERT model). The architecture implements configurable freezing of the bottom  $N$  encoder layers to preserve general legal language understanding during finetuning. Instead of simple classification-token pooling, it uses attention-based pooling over token outputs. It is trained with the multi-task loss shown in (1):

$$L = \lambda_{\text{elig}} L_{\text{elig}} + \lambda_{\text{domain}} L_{\text{domain}} + \lambda_{\text{conf}} L_{\text{conf}} \quad (1)$$

where  $L_{\text{elig}}$  is the binary cross-entropy loss for the eligibility prediction task,  $L_{\text{domain}}$  is the binary cross-entropy loss for the multi-label domain classification task, and  $L_{\text{conf}}$  is an auxiliary loss used to train the uncertainty estimation head;  $\lambda_{\text{elig}}$ ,  $\lambda_{\text{domain}}$ , and  $\lambda_{\text{conf}}$  are non-negative weights that control the relative importance of each task in the shared encoder’s optimization.

2) *LegalGAT (GNN)*: The knowledge-graph engine first parses the Prolog rule base to extract predicate nodes (for example, predicates representing gender status or income thresholds) and implication edges linking rule bodies to rule heads. For each case, it creates a subgraph by adding entity nodes that represent case-specific facts and connecting them to the predicate nodes they satisfy. The LegalGAT model, a two-layer Graph Attention Network, learns to perform graph-level classification on this subgraph to predict eligibility. This allows it to learn relational patterns (e.g., income + jurisdiction interactions) that are explicit in the graph.

3) *PrologEngine (Symbolic)*: The Prolog-based reasoning component loads a human-authored knowledge base of deterministic legal rules. These rules define absolute constraints and entitlements, such as categorical eligibility for minors or applicants holding below-poverty-line cards. When the engine finds a match, it returns a structured reasoning record that marks the case as eligible with maximum confidence and includes specific rule citations, providing complete explainability for that subset of cases.

### C. Training and Calibration

1) *Neural Model Training*: All neural models (the multi-task transformer, the standalone eligibility classifier, and the domain classifier) are trained independently using an AdamW optimizer with a linear warmup and decay schedule. Training the classifiers with different augmentations or hyperparameters encourages "prediction diversity," which is highly beneficial for the ensemble.

2) *GNN Training*: The LegalGAT graph neural network is trained on graph-level labels (eligible versus not eligible) using a cross-entropy loss and a global mean pooling readout layer.

3) *Confidence Calibration*: Once all models have been trained, their raw probability outputs are gathered on a held-out validation set. During the calibration phase, an isotonic regression model (or a logistic regression model for Platt scaling) is fitted separately for each component. This step is essential for effective ensembling because it aligns the confidence values across models—ensuring, for example, that a predicted probability of 0.8 from the GNN represents the same level of certainty as a 0.8 output from EnhancedLegalBERT.

## VI. EXPERIMENTS AND RESULTS

### A. Ablation Study Setup

To measure the contribution of each component, we performed an exhaustive ablation study, evaluating all 17 possible singleton, pair, triplet, and N-way combinations of the five core models. We report Accuracy, Precision, Recall, and F1-Score on the held-out test set. The ensemble logic for all combinations uses calibrated probabilities and optimized weights.

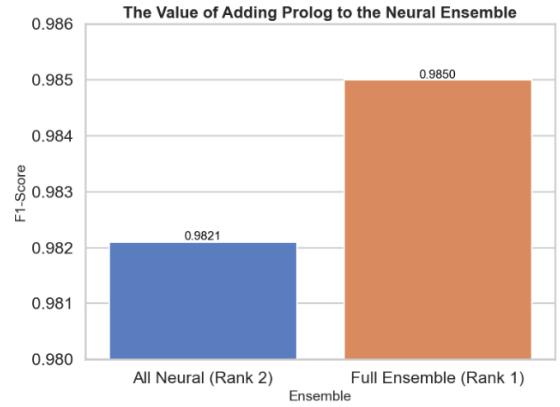


Fig. 3. The Value of Adding Prolog. The worst-performing standalone model (Prolog) provides the final performance boost when added to the best neural-only ensemble, proving its role as a "specialist."

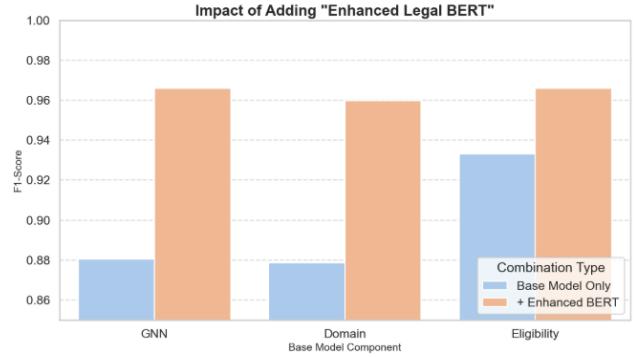


Fig. 5. Impact of Adding EnhancedLegalBERT to other base models. The performance of GNN, Domain, and Eligibility models is dramatically improved when ensembled with the powerful multi-task model.

### B. Ablation Results

As summarized in Table I, the full five-component hybrid ensemble achieves the highest F1-score, followed by the best neural-only configuration and then mixed three-component hybrids. The table also shows that each base model contributes complementary strengths, with the symbolic Prolog component providing a disproportionate benefit when combined with the neural models despite having the weakest standalone performance.

### C. Analysis of Results

The ablation study provides several critical insights into the system's behavior and the value of its hybrid design.

1) *Core Insight: The Ensemble*: The most significant finding is the performance jump from Rank 2 to Rank 1.

- **Rank 2 (15\_all\_neural)**: The best *neural-only* ensemble (all 4 neural models) achieves an F1-score of 0.9821.
- **Rank 17 (01\_prolog)**: by itself is the *worst-performing* model in the entire study, with an F1-score of 0.8252.

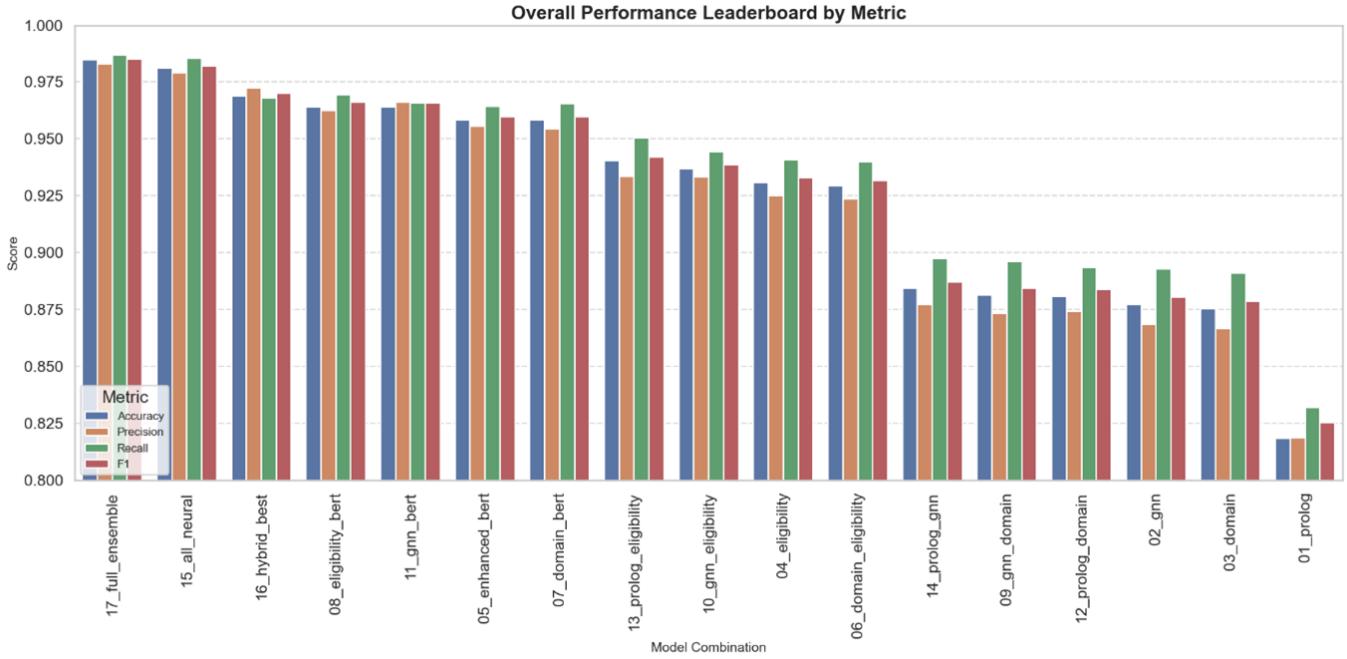


Fig. 4. Single Model Component Performance (Ranked by F1-Score). EnhancedBERT is clearly the most critical component, while Prolog is the weakest standalone component.

TABLE I  
ABLATION STUDY: PERFORMANCE OF 17 MODEL COMBINATIONS (RANKED BY F1-SCORE)

Rank	Combination ID	F1-Score	Accuracy	Precision	Recall	Models Used
1	17_full_ensemble	<b>0.9850</b>	<b>98.48%</b>	98.31%	<b>98.70%</b>	(All 5) Prolog + GNN + Domain + Eligibility + Enhanced BERT
2	15_all_neural	0.9821	98.13%	97.90%	98.53%	(All 4 Neural) GNN + Domain + Eligibility + Enhanced BERT
3	16_hybrid_best	0.9701	96.88%	97.23%	96.79%	Prolog + GNN + Eligibility Predictor
4	08_eligibility_bert	0.9660	96.40%	96.26%	96.94%	Eligibility Predictor + Enhanced BERT
5	11_gnn_bert	0.9659	96.40%	96.61%	96.58%	GNN + Enhanced BERT
6	05_enhanced_bert	0.9599	95.84%	95.55%	96.44%	Enhanced BERT (only)
7	07_domain_bert	0.9598	95.84%	95.42%	96.55%	Domain Classifier + Enhanced BERT
8	13_prolog_eligibility	0.9419	94.04%	93.36%	95.03%	Prolog + Eligibility Predictor
9	10_gnn_eligibility	0.9388	93.69%	93.33%	94.43%	GNN + Eligibility Predictor
10	04_eligibility	0.9329	93.07%	92.51%	94.08%	Eligibility Predictor (only)
11	06_domain_eligibility	0.9317	92.93%	92.35%	94.00%	Domain Classifier + Eligibility Predictor
12	14_prolog_gnn	0.8872	88.43%	87.72%	89.74%	Prolog + GNN
13	09_gnn_domain	0.8845	88.15%	87.31%	89.62%	GNN + Domain Classifier
14	12_prolog_domain	0.8837	88.08%	87.41%	89.34%	Prolog + Domain Classifier
15	02_gnn	0.8804	87.73%	86.83%	89.27%	GNN (only)
16	03_domain	0.8787	87.53%	86.67%	89.10%	Domain Classifier (only)
17	01_prolog	0.8252	81.84%	81.87%	83.18%	Prolog (only)

- **Rank 1 (17\_full\_ensemble):** Adding the "worst" model (Prolog) to the "second-best" model (all-neural) creates the #1 model, boosting the F1-score to 0.9850.

As shown in Fig. 3, this demonstrates that Prolog acts as a critical **"specialist"** or **"safety net."** It is a poor general-purpose predictor, but it flawlessly handles the specific, rule-based edge cases it was designed for, correcting rare but critical mistakes made by the powerful neural ensemble.

#### 2) Component Value Analysis:

- **EnhancedLegalBERT:** As shown in Fig. 4, enhanced bert is, by a wide margin, the best-performing single model

(F1: 0.9599). It is the core of the system's success. Fig. 5 further shows how adding EnhancedLegalBERT to the other base models (GNN, Domain, Eligibility) provides a massive performance boost in all cases.

- **EligibilityPredictor:** This is the strong second-best single model (F1: 0.9329). Its synergy with EnhancedLegalBERT is excellent, suggesting they learn different, complementary features.
- **LegalGAT:** The GNN is weak on its own (F1: 0.8804) but provides a strong boost when paired with BERT. This proves it successfully extracts relational patterns from the

structured entities (income, dates, etc.) that the text-only models miss.

3) *Precision-Recall Trade-off*: Fig. 6 visualizes the "personality" of each model combination. The top-performing ensembles (17, 15, 16) are clustered in the top-right, balancing high precision and high recall. In contrast, the prolog model is in the bottom-left, demonstrating poor performance. This plot confirms that the hybrid ensemble successfully combines the strengths of its components to achieve a superior balance.



Fig. 6. Precision-Recall Trade-off for all 17 combinations. Point size is scaled by F1-score. The top-ranked models are clustered in the optimal top-right corner.

4) *Ensemble Synergy Analysis*: Fig. 7 and Fig. 8 show clear evidence of ensemble synergy. As the number of models in the ensemble increases (Fig. 7), the median F1-score and the overall performance ceiling rise. Furthermore, Fig. 8 shows a clear progression: "Neural-Only Ensembles" outperform "Single Models," and the "Hybrid Ensembles" (which include Prolog) achieve the highest median performance and the top overall score.

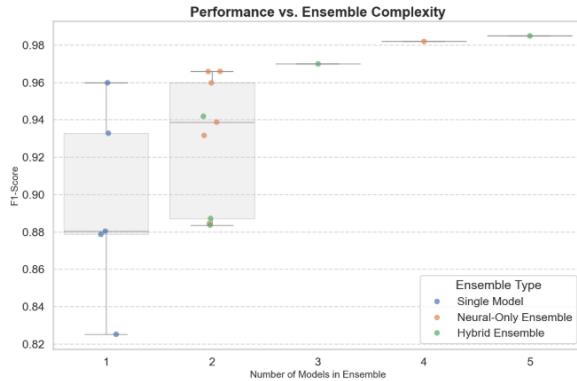


Fig. 7. Performance vs. Ensemble Complexity. Shows a clear upward trend where adding more models generally increases F1-score, with the 5-model ensemble performing best.

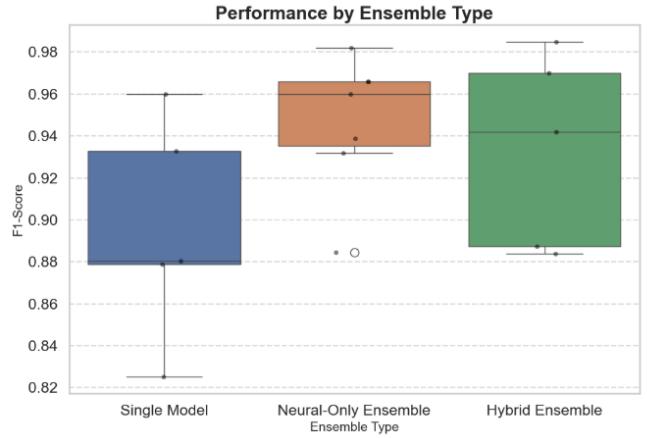


Fig. 8. Performance by Ensemble Type. This box plot shows that Hybrid Ensembles achieve the highest median F1-score and best overall performance, followed by Neural-Only Ensembles, and finally Single Models.

## VII. MONITORING AND RISK MANAGEMENT

A key requirement for a production-ready legal AI system is the ability to handle uncertainty and provide robust, auditable decisions without relying on continuous human intervention. As shown in Fig. 9, the system's output stage is designed to quantify uncertainty, log decisions, and monitor performance over time.

The hybrid prediction object from the prediction pipeline is not just a final answer; it is the start of a monitoring pipeline.

- 1) **Audit Log:** Every decision, along with its full trace (per-component scores, calibrated probabilities, and applied rules), is stored in an audit log for later inspection and compliance checks.
- 2) **Uncertainty Quantifier:** The system calculates uncertainty, primarily by checking the final probability's margin from the 0.5 decision boundary and the entropy (disagreement) among component models.
- 3) **Risk Flags:** If the margin is below a review-margin threshold or entropy is above a threshold, the system sets a risk flag to **True** and tags the decision as high-risk in the log and dashboards.
- 4) **Automated Monitoring & Metrics:** All decisions feed into monitoring dashboards that track system health, calibration error, and fairness metrics (e.g., expected calibration error, Brier score, and per-group F1-scores) over time, enabling periodic audits and retraining without requiring a human in the loop at inference time.

## VIII. CONCLUSION AND FUTURE WORK

This paper presented HybEx-Law, a five-component hybrid neural-symbolic system designed for legal aid eligibility determination. Through a comprehensive set of experiments, including a 17-configuration ablation study, we showed that the hybrid "specialist ensemble" consistently outperforms its neural-only counterparts.

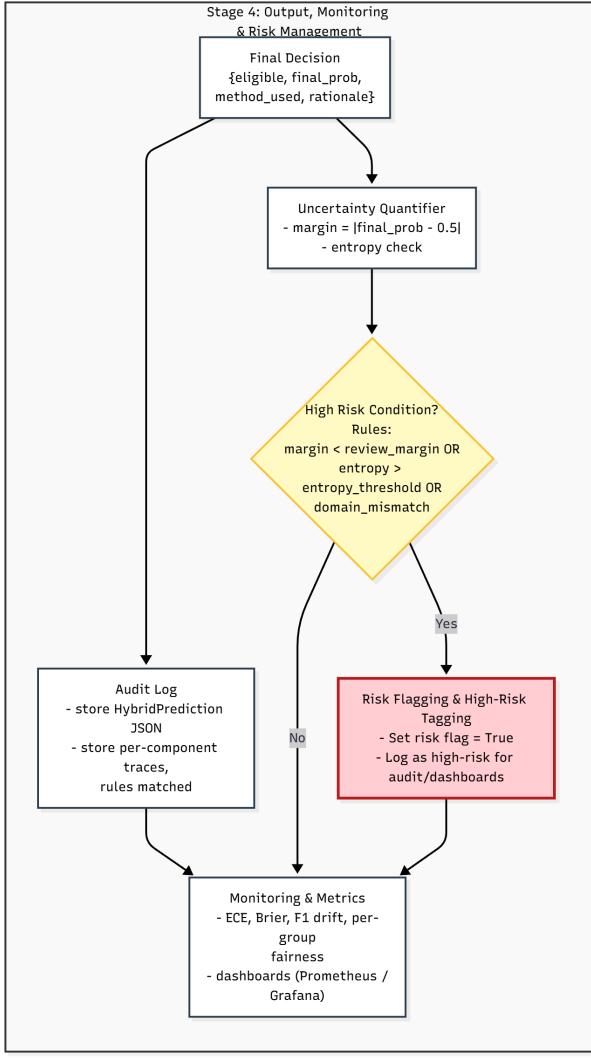


Fig. 9. Stage 4: Output, Monitoring, and Risk Management process. Uncertain decisions are automatically detected, logged, and surfaced through monitoring dashboards.

A central observation is that the full hybrid model ( $F1: 0.985$ ) exceeds the performance of the best neural-only ensemble ( $F1: 0.982$ ) due to the addition of the symbolic Prolog engine. Although the Prolog component is the weakest individual model, it acts as a deterministic safety layer that catches rule-governed errors that neural models occasionally overlook. By combining the contextual strengths of Transformer-based models such as EnhancedLegalBERT, the relational reasoning capabilities of LegalGAT, and the exactness of symbolic logic, HybEx-Law provides a more reliable and explainable decision framework.

Moving forward, we plan to broaden the scope of the Prolog rule base to capture additional legal edge cases, conduct deeper fairness and bias assessments across different population groups, and deploy a Human-in-the-Loop (HIL) interface with legal aid organizations to evaluate the system in real-world conditions.

## ACKNOWLEDGMENT

We would like to acknowledge the sources and organizations that supported this work. The legal foundation of the system is grounded in the Legal Services Authorities Act, 1987 [1]. The neural models were implemented using the Hugging Face Transformers framework [4], and the symbolic reasoning layer relies on the SWI-Prolog community's well-established inference engine [5]. We further extend our thanks to NALSA [6], the Department of Justice, and various legal aid practitioners for providing authoritative eligibility thresholds and valuable domain insights.

## REFERENCES

- [1] *Legal Services Authorities Act, 1987*, Act No. 39 of 1987, Parliament of India.
- [2] Mohit Kumar, Yuganshu Kumar, Diya Ravishankar, "HybEx-Law" 2025. [Online]. Available: <https://github.com/mk12002/HybEx-Law>
- [3] A. Vaswani, et al., "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998-6008.
- [4] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, and J. Brew, "HuggingFace's Transformers: State-of-the-art Natural Language Processing," *ArXiv*, vol. abs/1910.03771, 2019.
- [5] J. Wielemaker, T. Schrijvers, M. Triska, and T. Lager, "SWI-Prolog," *Theory and Practice of Logic Programming*, vol. 12, no. 1-2, pp. 61-92, 2012.
- [6] National Legal Services Authority (NALSA), India. [Online]. Available: <https://nals.gov.in/>
- [7] Z. Sun, K. Zhang, W. Yu, H. Wang, and J. Xu, "Logic rules as explanations for legal case retrieval," *arXiv preprint arXiv:2403.01457*, 2024. [Online]. Available: <https://arxiv.org/abs/2403.01457>
- [8] M. van Bekkum, et al., "Modular Design Patterns for Hybrid Learning and Reasoning Systems," *arXiv*, 2021.
- [9] De Smet, Lenert, Gabriele Venturato, Giuseppe Marra, and Luc De Raedt, "Neurosymbolic Reinforcement Learning With Sequential Guarantees," *arXiv*, 2024.
- [10] Chalkidis, M. Fergadiotis, P. Malakasiotis, N. Aletras, and I. Androutsopoulos, "LEGAL-BERT: The Muppets straight out of Law School," in Findings of the Association for Computational Linguistics: EMNLP 2020, 2020, pp. 2898–2904.
- [11] J. Collenette et al., "Explainable AI tools for legal reasoning about cases," *Artificial Intelligence*, 2023. [Online]. Available: <https://doi.org/10.1016/j.artint.2023.103882>
- [12] M. Chalkidis et al., "LexGLUE: A Benchmark Dataset for Legal Language Understanding in English," in *Advances in Information Retrieval*, 2022.
- [13] A. Galassi, M. Lippi, and P. Torroni, "Attention, please! A critical review of explainable artificial intelligence in natural language processing," *Artificial Intelligence*, 2021.
- [14] P. Henderson et al., "Legal NLP: An Empirical Study of Case Outcome Classification," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 2017.
- [15] R. Kalra et al., "HyPA-RAG: A Hybrid Parameter-Adaptive Retrieval-Augmented Generation System for Legal AI," in *NAACL Industry Track*, 2025. [Online]. Available: <https://aclanthology.org/2025.nacl-industry.79>
- [16] P. Hacker, "A legal framework for AI training data—from first principles to legal policies," *International Data Privacy Law*, 2021.
- [17] Z. Sun et al., "Logic rules as explanations for legal case retrieval," *arXiv preprint arXiv:2403.01457*, 2024.
- [18] X. Zhang and D. Wang, "Legal Knowledge Graphs and Their Applications in Case Law Analytics," *Artificial Intelligence and Law*, 2022.
- [19] S. Panagis et al., "Knowledge Graphs in the Legal Domain: A Survey," in *Proceedings of JURIX*, 2020.
- [20] K. Terzidou, "Generative AI systems in legal practice: quality, competence and confidentiality," *International Journal of Law in Context*, 2025.