Evaluating the Robustness of SHAP in Judicial Models

Anna Deng and Mija Reddy

Abstract

AI is being used in the judicial system in early stages and has the potential to have a great impact by expediting judicial protocols. However, one of the key concerns of experts about judicial AI is regarding a lack of accountability and transparency in blackbox models, especially legal judicial prediction (LJP) models. Post-hoc interpretability techniques like SHAP (SHapley Additive exPlanations) address this by providing insight into how different features in the input data affect the decided outcomes. Still, SHAP's main vulnerability is adversarial perturbation, which is when an agent manipulates input features, makes minor adjustments or introduces random noise to attempt to change feature importance significantly. Thus, the robustness of SHAP should be investigated in light of such perturbations. SHAP has been found to be robust in precision medicine, especially for simpler models. However, there is a gap in evaluating SHAP in the more complex models. In this paper we aim to evaluate the robustness of SHAP in complex models in the judicial sector, fine-tuning and using publicly available LJP models. We compare the SHAP values of a control set of normal inputs against the SHAP Values of a set of inputs with added noise, b) perturbed inputs (simulating gradient based adversarial attacks with FGSM) or inputs with added noise. Full code developed for the project can be found at https://github.com/annadeng8/ljp-interp.

1. Introduction

1.1. Definitions

SHAP (SHapley Additive exPlanations) is a game theoretic approach to explain the output of any machine learning model. It connects optimal credit allocation with local explanations using the classic Shapley values from game theory and their related extensions

Adversarial perturbation attacks on SHAP involve manipulating the input data or model outputs to deceive SHAP's feature importance explanations. These attacks aim to make SHAP generate misleading attributions, either by hiding biases, masking critical features, or creating explanations that appear fair while the model's predictions remain adversarial or biased.

Fast Gradient Sign Method (FGSM) is a technique that uses a neural network's gradients to create adversarial inputs. It's a type of adversarial attack that can impact the reliability of machine learning models.

Legal Text includes litigation-related corpora such as dockets, opinions and court transcripts but also corpora based on patents, briefs, public financial filings, civil code, local ordinances, privacy policies, law enforcement records, congressional records and speeches. It has distinct characteristics such as specialised vocabulary, particularly formal syntax, domain-specific semantics etc., to the extent that legal language is often classified as a 'sublanguage' which makes it challenging for generic Natural Language Processing (NLP) tools to work accurately.

The **Legal Judgment Prediction (LJP)** study refers to a field of study that predicts the court's decision based on the facts given of a legal case and other information from legal text.

Natural Legal Language Processing (NLLP) is an interdisciplinary field with elements of NLP and the legal domain, focusing on applying NLP to legal text and text with legal significance. Developing novel NLP for legal text methods holds the promise of transforming the legal services sector, with the US Legal Services market alone valued at $211 billion according to US government price indices.

1.2. Problem Statement

How robust is SHAP in complex legal judicial prediction models?

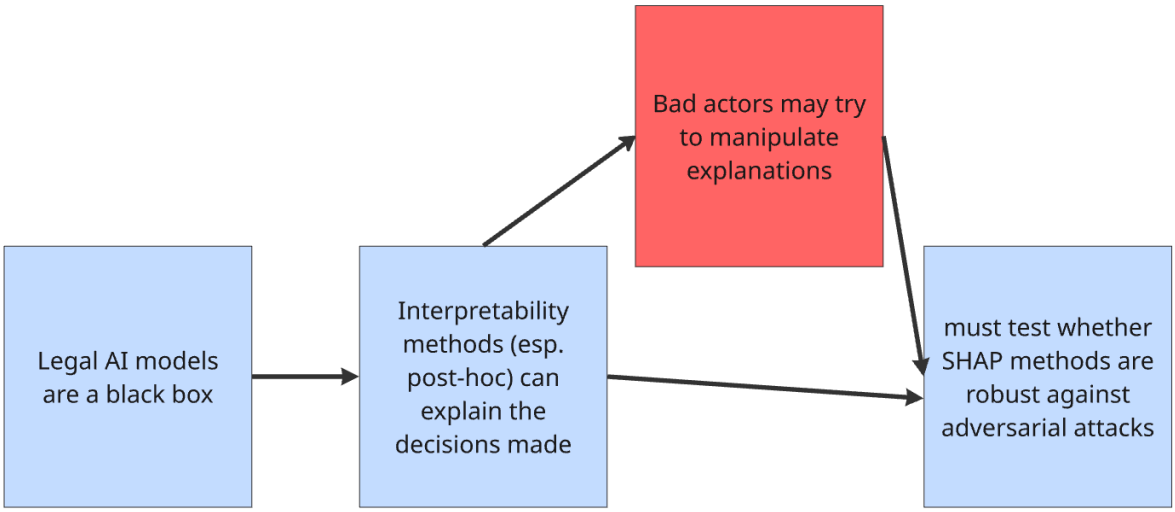
1.2.1 Importance

a) Legal Judgment Prediction Field

The field of Legal Judgment Prediction (LJP) has seen significant growth in the past decade, with over 100 papers published in the past few years alone. However, comprehensive studies have shown that only ~7% of published papers are actually predicting court decisions, highlighting the flawed and unreliable nature of the remaining experiments and emphasising their limited utility in the legal domain (Medvedeva et al., 2023). The media has long speculated about a more prevalent usage of AI-judges in making court decisions and AI-lawyers in defending human clients (Griffin, 2016). AI judges have already been implemented in China, with highly popular Internet courts available 24/7 to resolve a variety of affairs more quickly, from product reliability issues to administrative disputes (Vasdani, 2020).

In general, both the public and academics hold mixed views on the potential for AI applications to our lives. AI used in the judicial system could lead to faster decision-making and more consistency in decision-making, and it could also help lawyers process risk assessment and obtain better data-driven insights from precedents set by previous cases to help their clients be served justice. On the other hand, AI could reinforce unjust biases present in historical legal data, oversimplify complex human interactions, be susceptible to technological attacks, and overall is just not transparent in its outputs as a black-box model (Grimm et al., 2024).

While explainability tools like SHAP are available, we must be wary of the vulnerability of these tools to attacks by actors looking to modify outputs or sway both AI and humans in the litigation process. Still, post-hoc interpretability is a noncostly method of providing human-level interpretation for a process that is heavily human-centered, making it important to test existing such methods for their robustness.



b) Legal System (Broadly)

This project ultimately aims to bring change to the legal system and create an ethical basis for using AI to expedite the legal system, especially considering several people face severe negative consequences from delayed justice.
India, the country whose legal models we evaluate, has a slow and clogged judicial system due to an inadequate judge-to-population ratio, and has 50,000,000 cases pending and an estimated 245,000 cases pending for 20+ years.

1.2.2 Tractability

Our project allows for quantitative comparison of SHAP values between control and perturbed inputs, providing clear metrics for robustness assessment. This makes the robustness evaluation concrete and interpretable. Furthermore, because SHAP is model-agnostic, it can be easily and flexibly used to provide interpretation results for any model, making our framework for testing its vulnerability highly generalizable.

AI usage in the judicial system has just started and we believe efforts at this early stage will have a greater effect, and efforts in ensuring transparency and interpretability now will have a lasting impact.

1.2.3 Neglectedness

Despite its level of importance and potential for further research, the Legal Domain is largely underrepresented in the NLP literature. The legal and ethical landscape around judicial AI is still evolving, making researchers hesitant to invest in areas where certain regulatory changes might make their work inappropriate. Furthermore, due to privacy rights, many LJP models and datasets are not readily available to the public.

While research has been done on investigating the robustness of the LJP models themselves, not much has been done on investigating the robustness of interpretability techniques. Nonetheless, there has been work done on investigating SHAP's vulnerability in different applications, but the unique nature of legal texts with legal jargon make it more imperative to investigate SHAP specifically on LJP models (Zhong et al., 2020).

2. Overview of Current Literature

1. **Legal Model & InLegalBERT Interpretability**: *Pre-Trained Language Models for the Legal Domain a Case Study for the Legal Domain* (Paul et. al, 2022)

This paper on InLegalBERT, the model we use for this paper, and CaseLawBERT emphasises the importance of explainability of automated methods in the legal domain. The authors pass embeddings generated by InLegalBERT into a BiLSTM (Bidirectional Long Short Term Memory Network) higher level encoder with an attention head incorporated, and finally into a linear layer with sigmoid activation for classification. The authors compare 56 manually annotated documents with the details pertinent to the decision highlighted with the attention scores and InLegalBERT showed better alignment to the expert annotations than the previously existing LegalBERT model.

Relationship with our Work:

- Relates to interpretability in the legal domain, uses the same model.
- Does not use a model-agnostic method like SHAP, thus not generalizable.

2. Adversarial perturbation of interpretability method: *Interpretation of Neural Networks is Fragile* (Ghorbani et al., 2017)

The authors test the robustness of saliency maps against adversarial perturbation. The authors highlight that the focus of previous works is on adversarial attacks against the prediction; in contrast their work focuses on attacks on the interpretation without changing the prediction.The authors conclude that the interpretations (e.g. saliency maps) are vulnerable to perturbations. However, the authors highlight that fragility is inherent to high dimensional networks and interpretation fragility may be a potential natural consequence.

Relationship with our Work:

- Relates to adversarial perturbation, as it also focuses on attacks on the interpretation without changing the prediction, rather than the typical focus on adversarial attacks against the prediction.
- Uses images rather than text data, and uses saliency maps, which are applicable only to images. Evaluates fragility of high dimensional methods and explanations as a high level rather than focused evaluation of a single method.

3. Adversarial attacks on SHAP: *Fooling Lime and SHAP*

In this paper, the authors generate a novel framework to mask bias in models against LIME and SHAP. The authors methodology is creating “scaffolding” around the biased classifier such that its predictions on input data distribution remain biased, but its behavior on the perturbed data points is controlled to make the post hoc explanations look completely innocuous, the intuition behind this methodology being that real world data points are differentiable than artificial and perturbed datapoints.

The authors find that both LIME and SHAP are vulnerable to scaffolding, and LIME is more vulnerable than SHAP.

Relationship with our Work:

- Relates to interpretability in the legal domain, focuses on adversarial attacks against SHAP.
- Uses scaffolding attacks rather than adversarial perturbation attacks, a more complicated, and likely less probable adversarial attack than adversarial perturbation.

4. **Robustness of SHAP:** *Evaluating SHAP’s Robustness in Precision Medicine: Effect of Filtering and Normalization*

The authors evaluate the robustness of SHAP on various precision medicine models, while investigating the influence of filtering and normalisation techniques on the predictor variables. The authors aimed to identify the most 100 most important genes in predictions using SHAP, and the effect of filtering and normalisation on the explanations generated.

The authors conclude that SHAP is robust on simple ML methods such as logistic regression, though less so on more complex methods like XLGNet.

Relationship with our Work:

- Evaluates the robustness of SHAP, concludes that SHAP is robust on simple models, but not as much on complex models,
- Uses quantitative rather than text data, does not use more complex models such as BERT.

The research gap our project aims to fill:

Our project focuses on the robustness of SHAP against adversarial perturbation on complex text based legal models.

3. Overview of Post-hoc Interpretability Methods

There are broadly two categories of explainable AI (XAI) methods, model-agnostic and model-specific. **Model-agnostic methods work independently of the underlying model architecture**, often requiring post-hoc analysis. Based on model outputs, they often approximate a complex model or analyze feature contributions. On the other hand, **model-specific methods use the internal structure of a model**, giving more precise and architecture-dependent explanations at the cost of being less adaptable. Popular such methods include Grad-CAM which visualizes gradients of class scores using feature maps and Guided Backpropagation which ensures only positive, class-relevant features are propagated. **With the purpose of having a more generalizable method, we chose to use model-agnostic methods.**

The main post-hoc analysis tools we considered include Anchors, LIME (Local Interpretable Model-Agnostic Explanations), and SHAP.

3.1 Anchors

The Anchors method searches for the smallest candidate rule (input perturbation) that meets a desired prediction threshold probability for the prediction to remain unchanged. In other words, for a model f and input x, as well as a perturbed x' and precision threshold τ, an anchor A satisfies $P(f(x') = f(x) \mid A(x')) \geq \tau$ (Ribeiro 2018).

3.2 LIME

LIME is an XAI technique that trains a more interpretable surrogate model which serves to mimic the behavior of the original model based on the results of perturbed inputs passed through the original model. Then the coefficients of the surrogate model give information on the most important regions contributing to the prediction. The goal of this surrogate model g is to minimize the function

$$argmin_{g \in G} L(f, g, \pi_x) + \Omega(g)$$

where $L(f, g, \pi_x)$ is the loss function ensuring that g mimics f for perturbations around x, π_x assigns higher weights to perturbed instances in closer neighborhoods to x, and $\Omega(g)$ acts as a sparsity constraint for g to remain interpretable.

3.3 SHAP

SHAP is a game theory-based method for explaining the output of any machine learning model. It uses a cooperative game-theoretic framework to form feature coalitions, assign scores for how adding or removing a feature affects predictions called Shapley values, and then produces

visualization heatmaps to show what regions affected the prediction in which direction.

The Shapley value for a feature i can be expressed as

$$\phi_i = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!(|N|-|S|-1)!}{|N|!} (f(S \cup \{i\}) - f(S))$$

where S is a subset of features, N is the set of all features, and $f(S)$ is the model output when only features in S are accounted for.

3.4 What We Decided On

While LIME and Anchors approximate local behavior, SHAP ensures global and local interpretability (Devireddy 2025). This makes these methods good choices for image processing. Furthermore, LIME has also already been shown to be very vulnerable to scaffolding attacks, so we decided to proceed with investigating LJP models with SHAP

4. Theory of Change

4.1 Inputs

We compare the SHAP values of inputs to model against SHAP values of
a) inputs with noise added
b) inputs generated with FGSM, a gradient based adversarial method

4.2 Outcomes

Primary outcomes:

An understanding of the robustness of SHAP on complex text-based judicial models

Secondary outcomes:

- Understanding the robustness of InLegalBERT
- Understanding of the interplay of model vs explanation fragility with InLegalBERT vs SHAP
- Understanding the utility of post-hoc interpretability models

Impact:

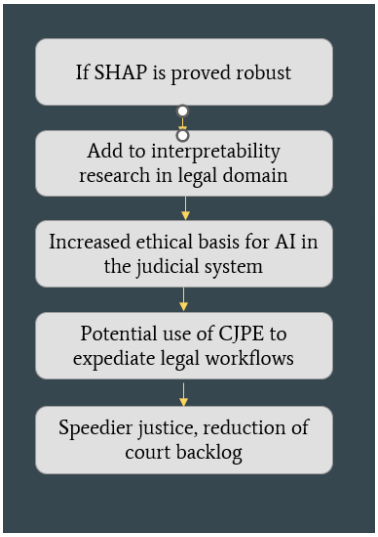
- Improving and setting a precedent for transparency in judicial AI (facilitating its use in helping over-crowded judicial sectors)

- Getting a clearer understanding about the robustness in SHAP in complex judicial models, filling a gap in research, as SHAP robustness has only been evaluated in a few areas and complexities of models.
- SHAP is also a popular technique, providing global interpretability across the data-set, as well as local interpretability for a particular decision, increasing the impact of our findings.

4.3 Scenarios

We present our **Theory of Change** for two scenarios:

If SHAP proven to be robust:

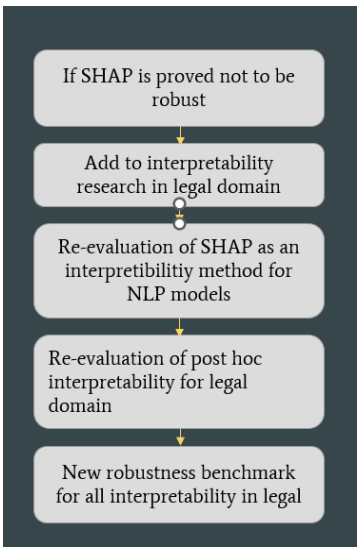


Ideally, if SHAPley is shown to be robust on legal datasets, we would add to existing Legal AI interpretability research, and help cement an ethical basis for AI use in the judicial systems.

If there were an ethical basis for AI use in the legal sector, judicials systems, especially the Indian judicial system could implement it in their ongoing initiatives to include AI in the legal process (The Indian courts have published legal datasets and models).

If successful, the result may be a far more efficient AI-assisted judicial system, which is still ethical and transparent, which will result in ensuring speedy justice for millions of people, and ease the burden on courts and judges. Additionally, as AI in institutional decision making is still in early stages in India, we may set a precedent for transparency across the institutional decision making spectrum.

If SHAP is proven not to be robust:



1. <u>SHAP's value must be reevaluated, especially against especially complex text models.</u>

SHAP is an increasingly popular interpretability method, so it is urgent to be reevaluated if it is not robust against complex text models. *"As the popularity of SHAP increases, also the number of approaches based on it or directly on Shapley values has been on the rise. In fact, ~ 3, 200 of the ~ 6,900 papers citing Lundberg and Lee (2017) are*

from 2021, an exponential increase when compared to previous years (1563, 567, and 118).”

2. A more general reevaluation of post-hoc interpretability methods and generalizable methods in the legal domain should be conducted

   If SHAP is proven not to be robust against adversarial perturbation, and since SHAP and other post-hoc methods like LIME have already been proven to not be robust against scaffolding attacks, it may be worthwhile to evaluate whether other types of interpretability methods such as model specific methods could be used, or whether ante-hoc methods could be trained to sufficient accuracy

3. A new robustness benchmark for all interpretability methods. Further interpretability methods should be proven to be more robust than SHAP.

5. Methods

5.1. Model Selection and Architecture Setup

LegalBERT is a family of BERT-based transformer models pre-trained specifically on legal texts, such as legislation, court decisions, and contracts, to better handle legal NLP tasks than general-purpose language models. The models are pre-trained on source corpora from legislation, contracts, and legal documents. When adapting to downstream tasks (e.g. LJP, contract clause classification, entity recognition), LegalBERT uses enhanced hyperparameter tuning for a broader search that yields notably better performance than the default settings used for standard BERT.

BERT to InLegalBERT

InLegalBERT is a subset of LegalBERT pretrained on Indian legal texts. Without fine-tuning, when text is passed to it (like a sentence or paragraph), the model returns contextual embeddings:

- A vector for each token (from the last hidden layer).
- A special [CLS] token vector (embedding) that summarizes the entire input.

These embeddings are then used as input to **a** downstream task-specific model.

We fine-tuned the model for the judicial outcome prediction task using the ILDC dataset that InLegalBERT was also trained on. Specifically, we took input case documents (e.g., facts, arguments), fed them into InLegalBERT to get contextual embeddings (the `[CLS]` token), added a classification head, and trained the whole system end-to-end using labeled data (e.g., accepted vs. rejected outcomes).

(InLegalBERT, created by IIT Kanpur faculty, is publicly available on GitHub (linked below))

5.2. Model Training

We trained the classifier in two ways following the decision to use a direct application of LegalBERT, a simple framework, and an enhanced framework.

1. The simple framework uses a single dropout layer + single linear layer, has the standard CrossEntropyLoss function, and takes the pooled output directly from BERT.

```python
class SentimentClassifier(nn.Module):
    def __init__(self, n_classes):
        super(SentimentClassifier, self).__init__()
        self.bert = model = AutoModel.from_pretrained("law-ai/InLegalBERT")
        self.drop = nn.Dropout(p=0.3)
        self.out = nn.Linear(self.bert.config.hidden_size, n_classes)

    def forward(self, input_ids, attention_mask):
        # Ensure BERT model returns tensors
        outputs = self.bert(
            input_ids=input_ids,
            attention_mask=attention_mask
        )
        pooled_output = outputs[1]  # Typically, the second item is the pooled output

        # Add a dropout layer
        output = self.drop(pooled_output)
        return self.out(output)
```

Simple Framework Used For Direct InLegalBERT application

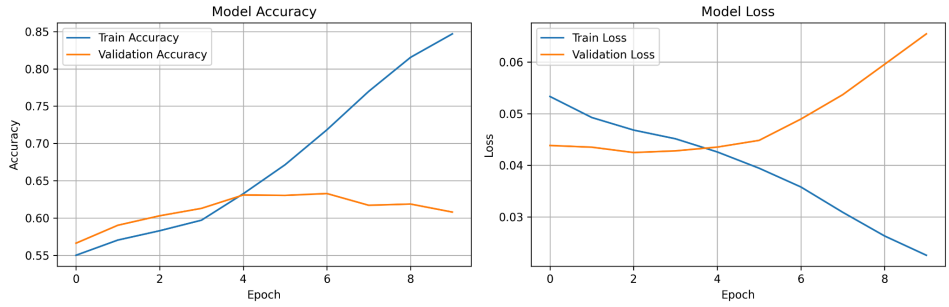
	precision	recall	f1-score	support
0	0.68	0.71	0.69	2154
1	0.54	0.50	0.52	1466
accuracy			0.63	3620
macro avg	0.61	0.61	0.61	3620
weighted avg	0.62	0.63	0.62	3620

Simple Model Training

2. Meanwhile, the enhanced multi-layer classifier uses batch and layer normalization, a custom attention pooling mechanism, text-preprocessing, and FocalLoss to account for skewed datasets. Though the enhanced framework was more complex, it did not perform better than the simpler framework.

```python
class EnhancedLegalClassifier(nn.Module):
    def __init__(self, n_classes=2, hidden_size=768, dropout_rate=0.3, num_layers=2):
        super(EnhancedLegalClassifier, self).__init__()

        # Load legal BERT model
        self.bert = AutoModel.from_pretrained("law-ai/InLegalBERT")

        # Freeze early layers
        for param in self.bert.embeddings.parameters():
            param.requires_grad = False

        # Multi-layer classifier with batch normalization
        self.classifier_layers = nn.ModuleList()
        current_size = self.bert.config.hidden_size

        for i in range(num_layers):
            next_size = hidden_size // (2 ** i)
            self.classifier_layers.extend([
                nn.Linear(current_size, next_size),
                nn.BatchNorm1d(next_size),
                nn.ReLU(),
                nn.Dropout(dropout_rate)
            ])
            current_size = next_size

        self.final_classifier = nn.Linear(current_size, n_classes)

        # Attention pooling mechanism
        self.attention_pooling = nn.MultiheadAttention(
            embed_dim=self.bert.config.hidden_size,
            num_heads=8,
            batch_first=True,
            dropout=dropout_rate
        )

        # Layer normalization
        self.layer_norm = nn.LayerNorm(self.bert.config.hidden_size)
```

Enhanced Framework Used For Direct InLegalBERT application

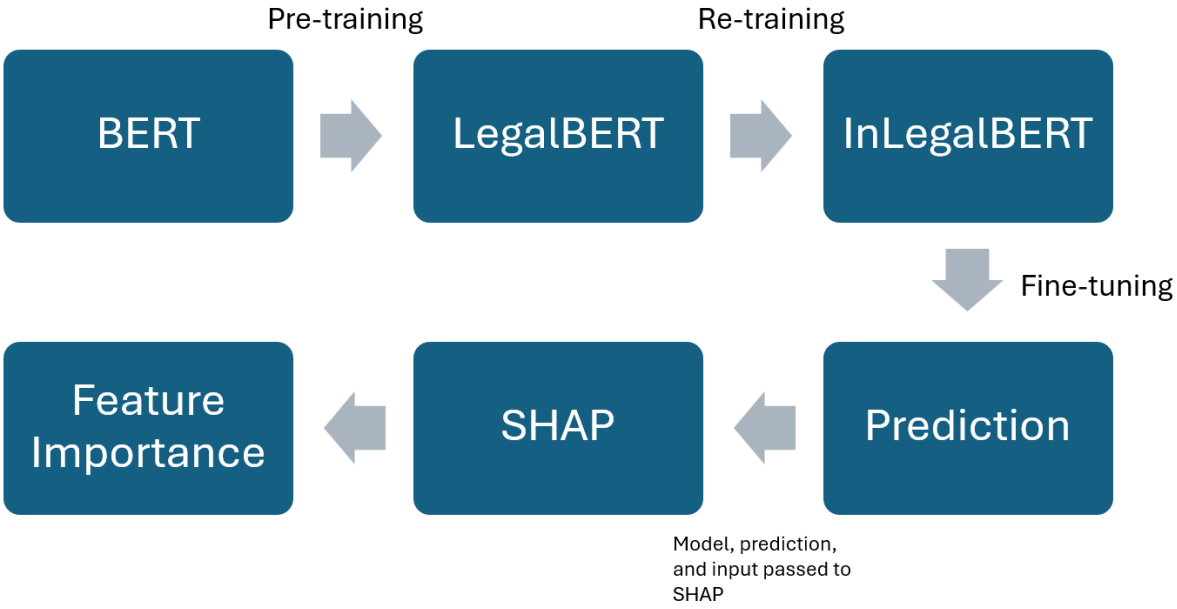


Enhanced Model Training

5.3. SHAP Application

Below are the ways ways we applied SHAP analysis to legal texts:

- **Text Masker**: Used `shap.maskers.Text(tokenizer)` to handle tokenization and masking operations consistent with our BERT-based model
- **Prediction Function**: Created a wrapper that handles tokenization, padding, and returns probability distributions
- **Token-level Attribution**: Generated SHAP values at the token level rather than character or n-gram level



Full pipeline

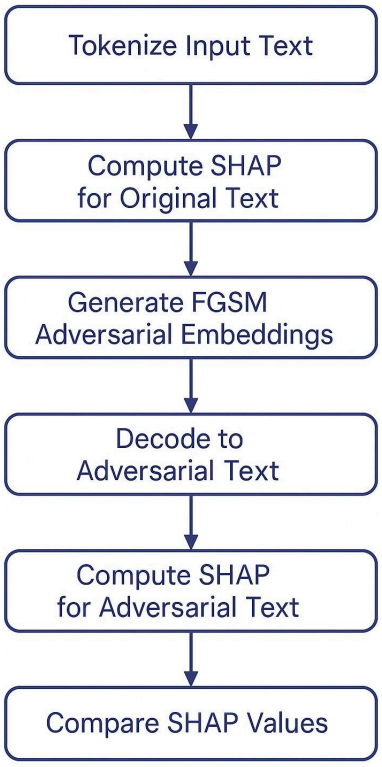
5.4 Noise Perturbations

Extensive noise vulnerability testing at different intensities for the judicial prediction model was done, from text-based manipulation to adding noise to embeddings, to assess how robust SHAP explanations and the model itself are against various types of noise attacks.

- Embedding Noise Attacks - This test injects noise directly into the BERT embeddings to see how SHAP explanations change. The types of noise used include Gaussian, uniform, and targeted. Gaussian noise (`torch.randn_like(embeddings) * noise_std`) is random noise sampled from a normal distribution, simulating random perturbations in all embedding dimensions. Uniform noise (`noise = (torch.rand_like(embeddings) - 0.5) * 2 * noise_std`) acts as noise sampled from a uniform distribution and models uniform uncertainty across embedding space. Targeted noise (`noise[..., :half_dim] = torch.randn_like(...) * noise_std`) injects Gaussian noise only into the first half of the embedding dimensions. It simulates an adversarial, structured attack that mimics feature-specific perturbation.

- Transformation Tests - Tests various text modifications (synonyms, case changes, punctuation, paraphrasing) that shouldn't change the legal meaning of the input but might still fool the model

5.5 FGSM Perturbations



5.5.1 Code for FGSM Attacks

```
import torch
import torch.nn as nn
from transformers import AutoModel, AutoTokenizer
import shap
import numpy as np
import pandas as pd

device = torch.device("cuda" if torch.cuda.is_available() else "cpu")

class InLegalBERTClassifier(nn.Module):
    def __init__(self, n_classes):
        super(InLegalBERTClassifier, self).__init__()
        self.bert = AutoModel.from_pretrained("law-ai/InLegalBERT")
        self.drop = nn.Dropout(p=0.3)
        self.out = nn.Linear(self.bert.config.hidden_size, n_classes)

    def forward(self, input_ids=None, attention_mask=None, inputs_embeds=None, token_type_ids=None): # Accept token_type_ids
        output = self.bert(input_ids=input_ids, attention_mask=attention_mask, inputs_embeds=inputs_embeds)
        return self.out(self.drop(output.pooler_output))

tokenizer = AutoTokenizer.from_pretrained("law-ai/InLegalBERT")
model = InLegalBERTClassifier(n_classes=2)
model.load_state_dict(torch.load("/content/drive/MyDrive/Colab Notebooks/best_model_state (1).bin", map_location=device))
model.eval().to(device)
```

1. Loading and initializing finetuned InLegalBERT

```
def predict_fn(texts):
    if isinstance(texts, np.ndarray): texts = texts.tolist()
    inputs = tokenizer(texts, return_tensors="pt", padding=True, truncation=True, max_length=512)
    inputs = {k: v.to(device) for k, v in inputs.items()}
    with torch.no_grad():
        logits = model(**inputs)
        probs = torch.nn.functional.softmax(logits, dim=1)
    return probs.cpu().numpy()
```

2. Creating a predict function to output the probabilities

```python
def shap_with_fgsm(text, epsilon):
    tokens = tokenizer(text, return_tensors='pt', padding=True, truncation=True, max_length=512)
    input_ids = tokens['input_ids'].to(device)
    attention_mask = tokens['attention_mask'].to(device)
    # Clean SHAP
    explainer_clean = shap.Explainer(predict_fn, shap.maskers.Text(tokenizer))
    shap_values_clean = explainer_clean([text])
    # FGSM
    embedding_layer = model.bert.embeddings.word_embeddings
    inputs_embeds = embedding_layer(input_ids).detach().clone().requires_grad_(True)
    logits = model(inputs_embeds=inputs_embeds, attention_mask=attention_mask)
    predicted_label = logits.argmax(dim=-1).detach()
    loss = torch.nn.functional.cross_entropy(logits, predicted_label)
    model.zero_grad()
    loss.backward()
    grad = inputs_embeds.grad
    adv_embeds = inputs_embeds + epsilon * grad.sign()
    # Decode adversarial tokens
    with torch.no_grad():
        token_embeddings = embedding_layer.weight.to(device)
        sim = torch.nn.functional.cosine_similarity(adv_embeds.view(-1, adv_embeds.size(-1)).unsqueeze(1),
                                                    token_embeddings.unsqueeze(0), dim=2)
        adv_token_ids = sim.argmax(dim=-1).view(input_ids.shape)
        adv_text = tokenizer.decode(adv_token_ids[0], skip_special_tokens=True)

    # Adversarial SHAP
    explainer_adv = shap.Explainer(predict_fn, shap.maskers.Text(tokenizer))
    shap_values_adv = explainer_adv([adv_text])

    return shap_values_clean[0], shap_values_adv[0], text, adv_text
```

3.1 Clean SHAP Calculation:

Creates a SHAP explainer for the original text using the `predict_fn` and a text masker with the tokenizer and `c`alculates the SHAP values for the original input text.

3.2 FGSM Attack:

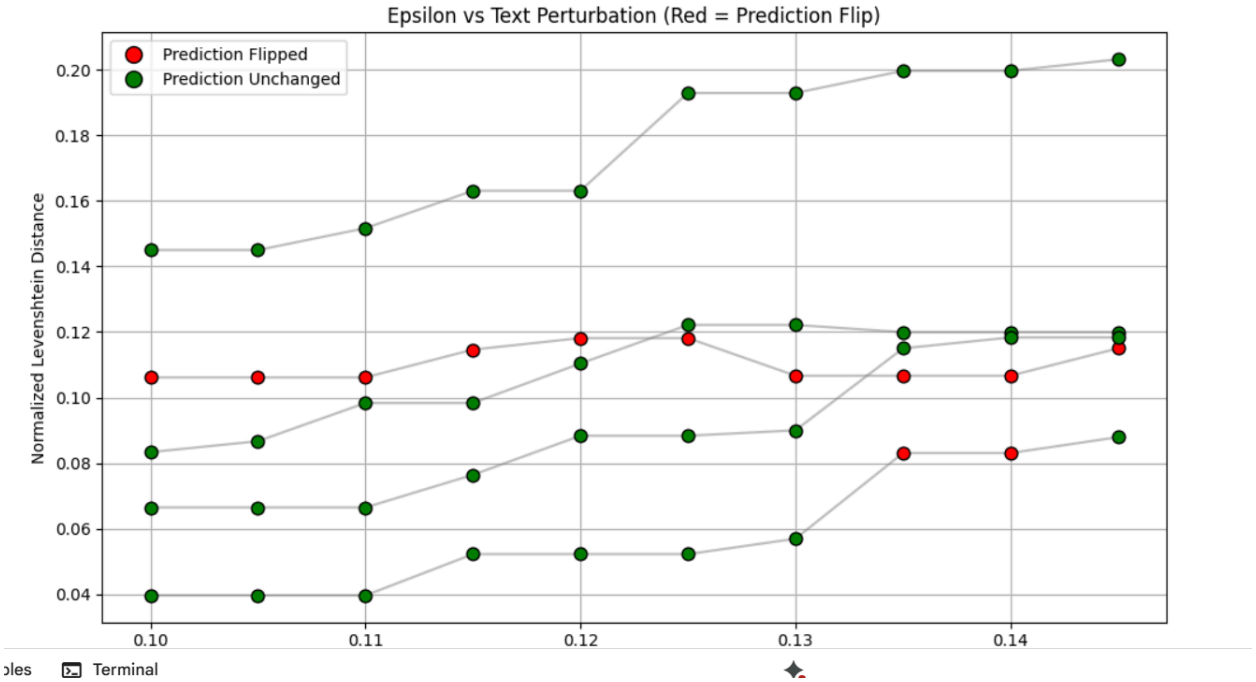
FGSM cannot be run on tokens, it gets access to the embeddings and gradients. It adds a small perturbation to the original embeddings in the direction of the sign of the gradient, scaled by `epsilon`. This perturbation is designed to maximize the loss for the original predicted class.

3.3 Decoding Adversarial Tokens:

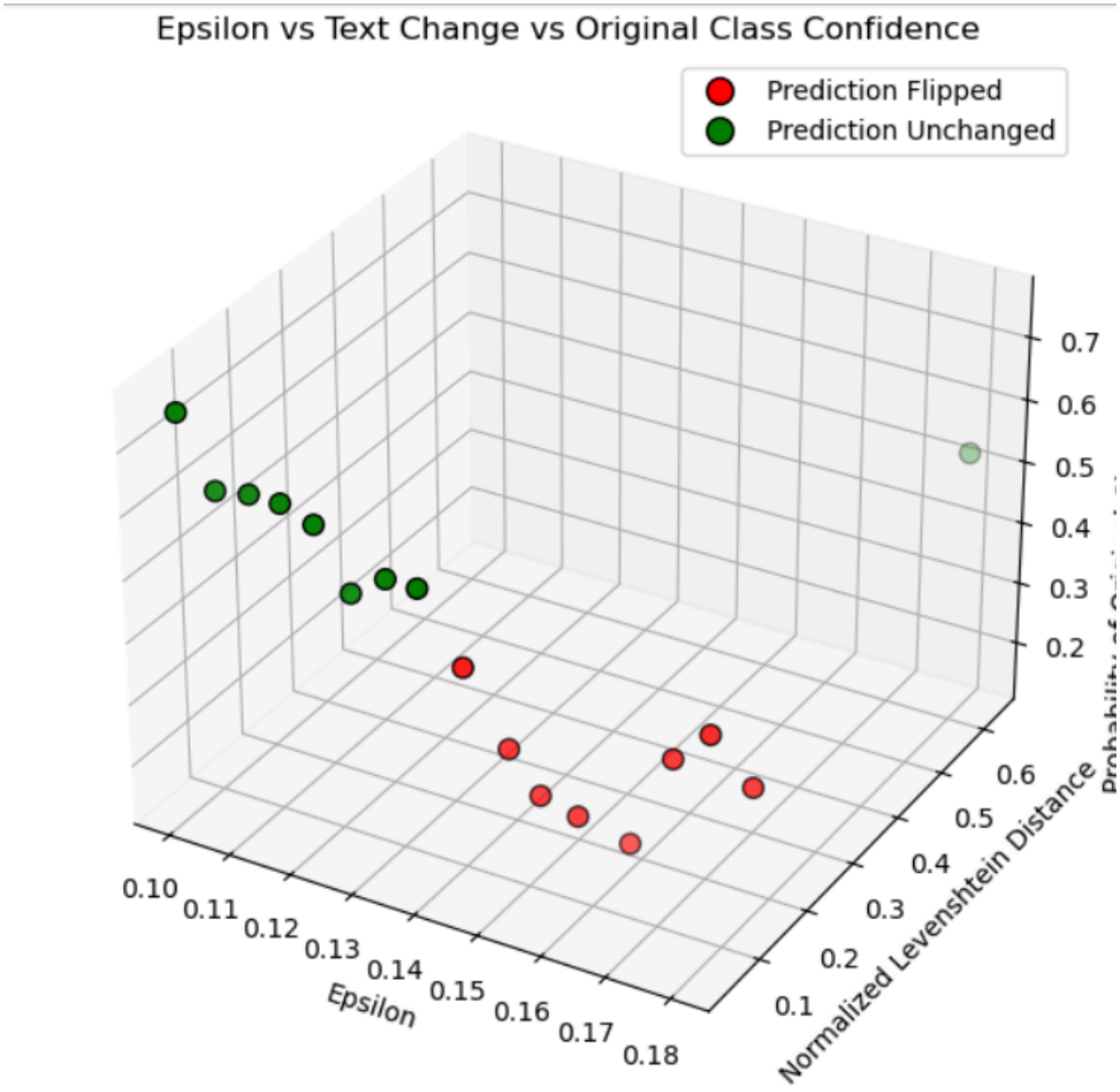
- *This part attempts to find the closest actual tokens in the vocabulary to the modified adversarial embeddings as tokens are easier to understand. Creates a SHAP explainer for the adversarial text and calculates the SHAP values for the adversarial text.*

5.5.2 Activity Cliffs

FGSM was prone to activity cliffs, wherein the adversarial text be any different from the original text until a certain point, after which it would be so different as to be completely incomprehensible.
Thus, we had to select an epsilon carefully so as to generate a slightly perturbed yet still comprehensible.
We ran a few simulations to pick an ideal epsilon



How perturbed strings in a few cases changed for different values of epsilon.



How perturbed strings and model probabilities in single case changed for different values of epsilons

(NOTE: Levenshtein distance is the number of modifications that must be made to a string to get another string, it is a measure of difference between strings, here the Levenshtein distance is between perturbed and original inputs.)

5.5.3 Comparison with the paper Evaluating SHAP's Robustness in Precision Medicine

1. We replicated the methodology of Evaluating SHAP's robustness in precision medicine by considering only the cases where the correctly predicted the unperturbed input. All graphs attached in the results section are for only the cases the model predicted correctly. However, graphs with all cases are attached in the appendix for. This was recommended by the authors to counteract low model accuracy
2. Furthermore, as in the precision medicine paper the top 100 most influential genes were calculated, we calculated the top 20 most influential tokens

5.6. Evaluation Metrics

- **Attack success rate / prediction flip rate**: percentage of samples that the model flipped its predictions for post-attack
- **Jaccard similarity / text similarity**: a measure of similarity between two sets of text. It is used to determine if clusters containing raw messages are duplicates by computing the similarity score between the sets of words associated with each cluster. It is computed using intersection / union.
- **SHAP difference**: difference in the SHAP values post attack. Specifically, it measures the average change in SHAP values (how much each token's contribution changed)
- **Confidence change**: difference in the token probabilities post attack
- **SHAP correlation / Pearson correlation**: the Pearson correlation between the absolute SHAP values of the original text and the attacked (perturbed) text. This tells us how similarly the model values each token before and after the attack. A low SHAP correlation suggests a successful attack on SHAP methods.
- **Feature consistency**: how much the top-k most important features (based on SHAP values) remain the same before and after the attacks

5.7. Alternative Methods

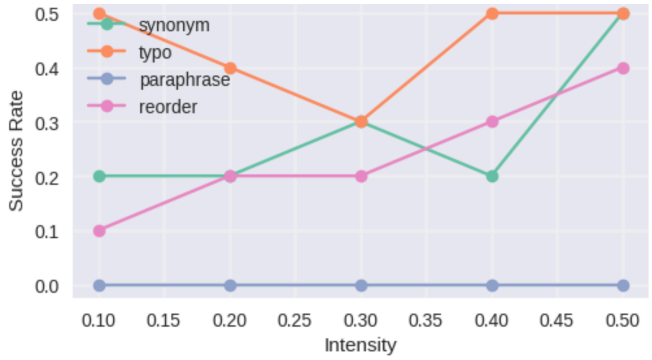
We considered two main alternatives

1. **Different model**: We initially considered using an entirely different model, InLegalBERT (Indian Legal BERT). InLegallBERT is an LLM, however we could not prompt it to output clear binary predictions, i.e. 0/1 or accept/reject for a petition/claim. Thus we could not use it for LJP tasks.
2. **Different pipeline for InLegalBERT:** We tried using the pipeline that had the highest accuracy for CJPE (Court Judgment Prediction and Explanation) tasks according to IL-TUR, InLegalBERT+BIGRU+Linear Layer. However, we were not able to replicate the 80% accuracy stated in the benchmark, and instead only got a 60% accuracy. Owing to less accuracy, as well as the complications of applying SHAP on this more convoluted pipeline, we fine-tuned InLegalBERT alone for classification.

6. Results and Analysis

6.1. Noise Attacks

6.1.1. Semantic Transformations

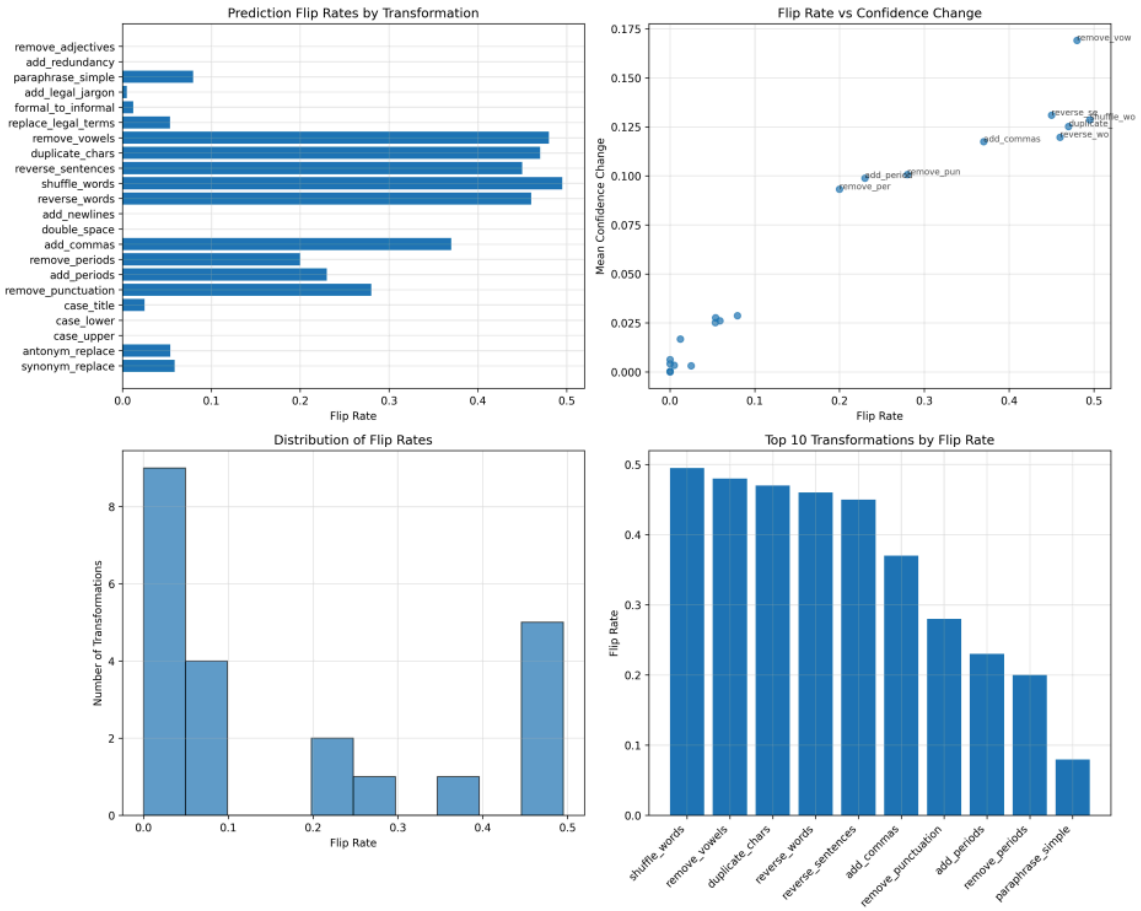


We found that the model's predictions were highly susceptible to the attacks themselves based on the number of prediction flips. However, this may be related to the model's overall low accuracy rates.

Attack Type	Success Rate	Avg Conf Change	Avg Similarity
synonym	0.280	-0.017	0.541
typo	0.440	0.019	0.166
paraphrase	0.000	0.006	0.993
reorder	0.240	-0.042	1.000

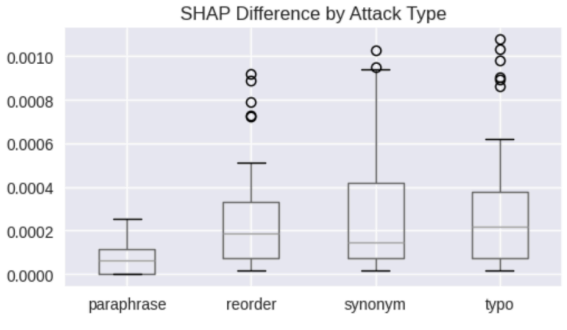
In particular, the analysis shows different vulnerability across the semantic noise attack types:

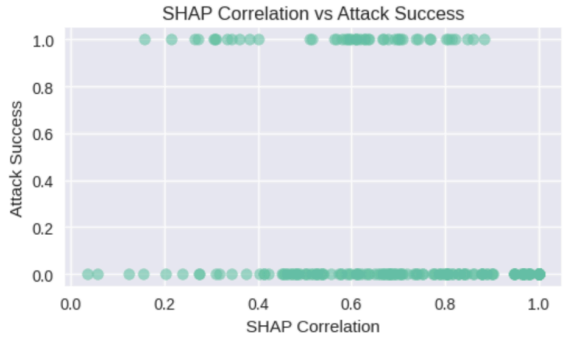
- **Typo attacks** demonstrated the highest success rate (40%), indicating that InLegalBERT is weak against character-level attacks
- **Synonym replacement** and **word reordering** attacks achieved moderate success rates (24% each), suggesting the model has some resilience to semantic-preserving transformations
- **Paraphrase attacks** showed no measurable success (0%), indicating robust performance against higher-level semantic restructuring



Graphs for Prediction Flip Rates as a Consequence of Input Transformation

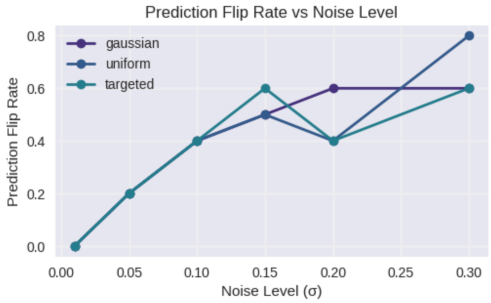
Similarly, typos and synonym attacks exhibited the greatest SHAP differences. However, there interestingly was not much correlation between attack success and SHAP correlation, and low SHAP correlation values (which would mean significant changes in SHAP reasoning post attack) were not prevalent. For all experiments, the mean SHAP correlation was at least 0.55. Furthermore, the SHAP differences in general were quite low, suggesting that SHAP was mostly resistant against the text attacks. The mean SHAP differences for all experiments was less than 0.0003.



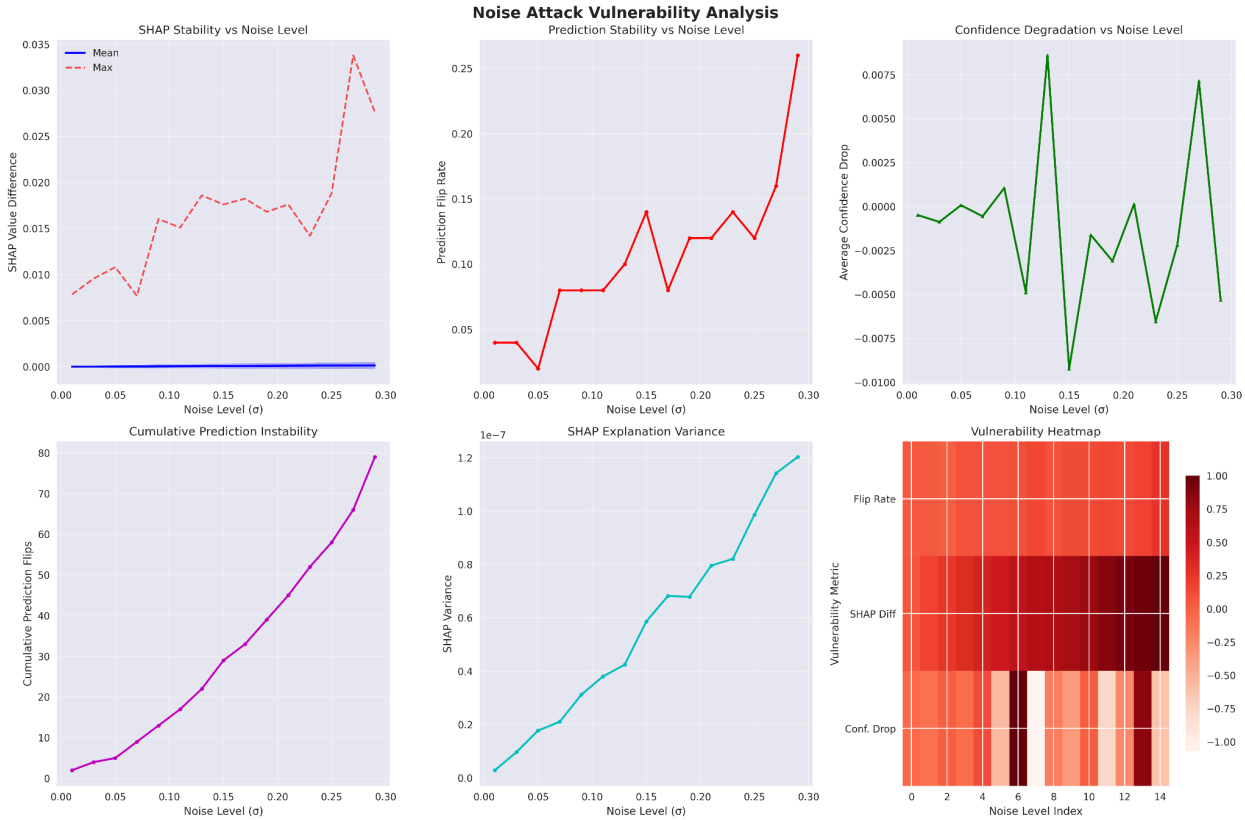


6.1.2. Embedding Noise

The noise was successful in degrading the confidence of the model, with greater noise levels influencing higher prediction flip rates.



However, it was overall ineffective in disturbing the stability of SHAP as the SHAP values barely changed for each token after the attack.



Graphs Analyzing SHAP difference and other metrics

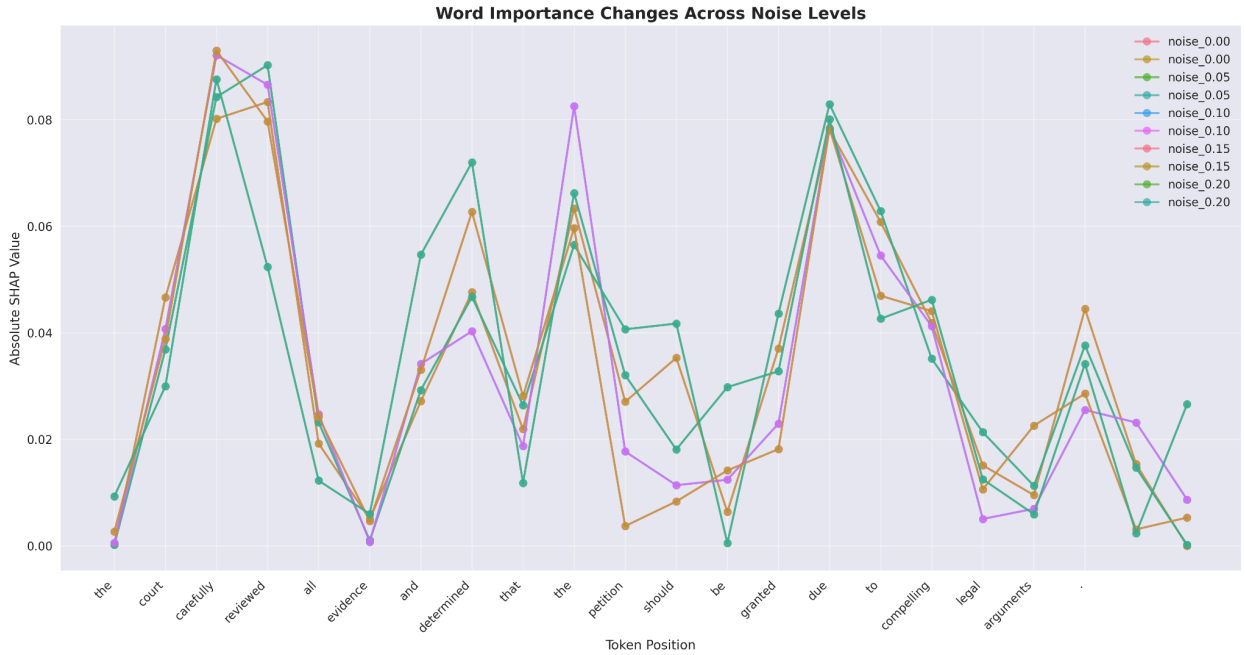
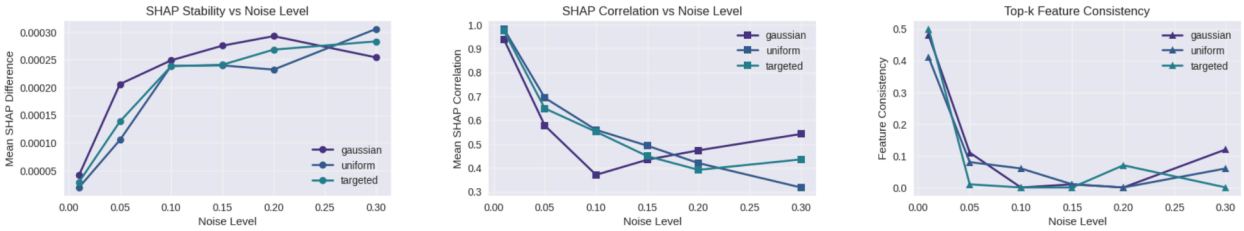


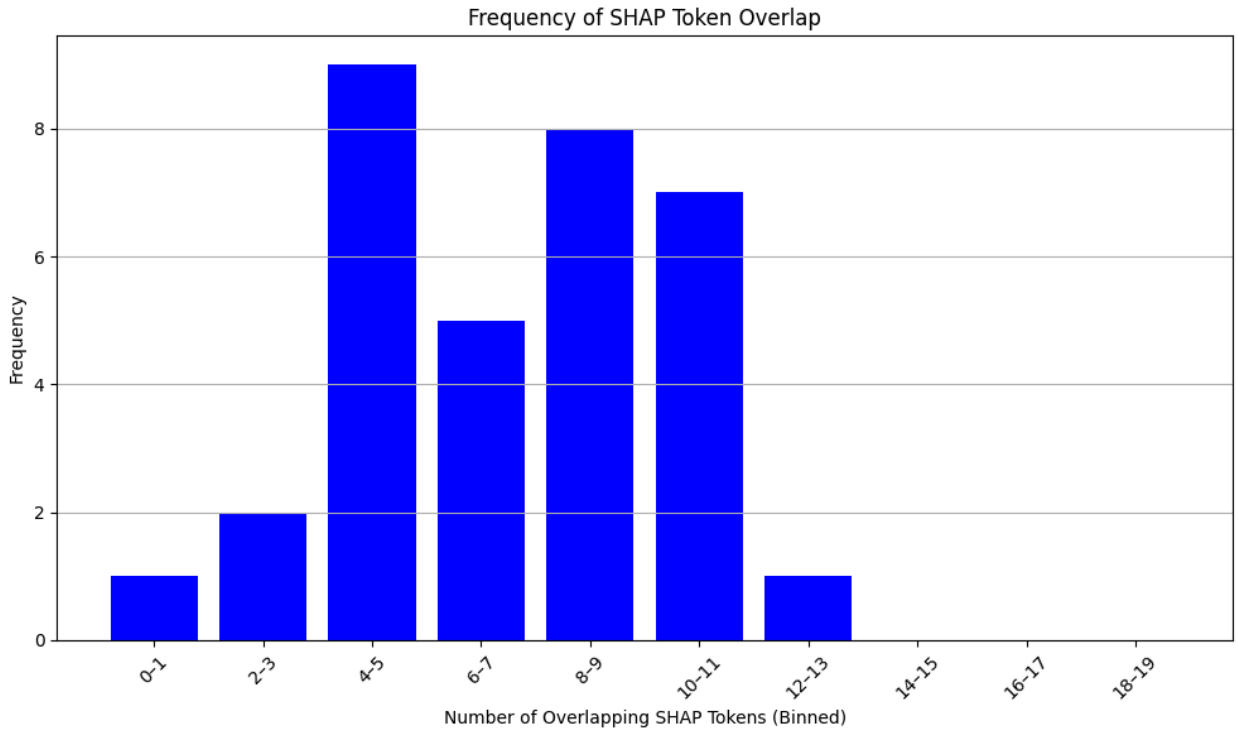
Chart of effect of noise attacks on SHAP values for sample sentence "The court carefully reviewed all evidence and determined that the petition should be granted due to compelling legal arguments."



However, in contrast to the semantic transformations, the embedding noise attacks clearly affected SHAP more. While the SHAP differences still remained under 0.003, the SHAP correlations dropped to under 0.5 when the noise level was greater than 0.2, and the top-k feature consistency was also rather low for greater noise. The latter metric could also be affected by the model's inconsistency with its predictions.

6.2. FGSM Attacks

6.2.1 Frequency of SHAP Token Overlap



This graph shows how frequently different levels of overlap occur between the top 20 most influential tokens in the original and perturbed inputs

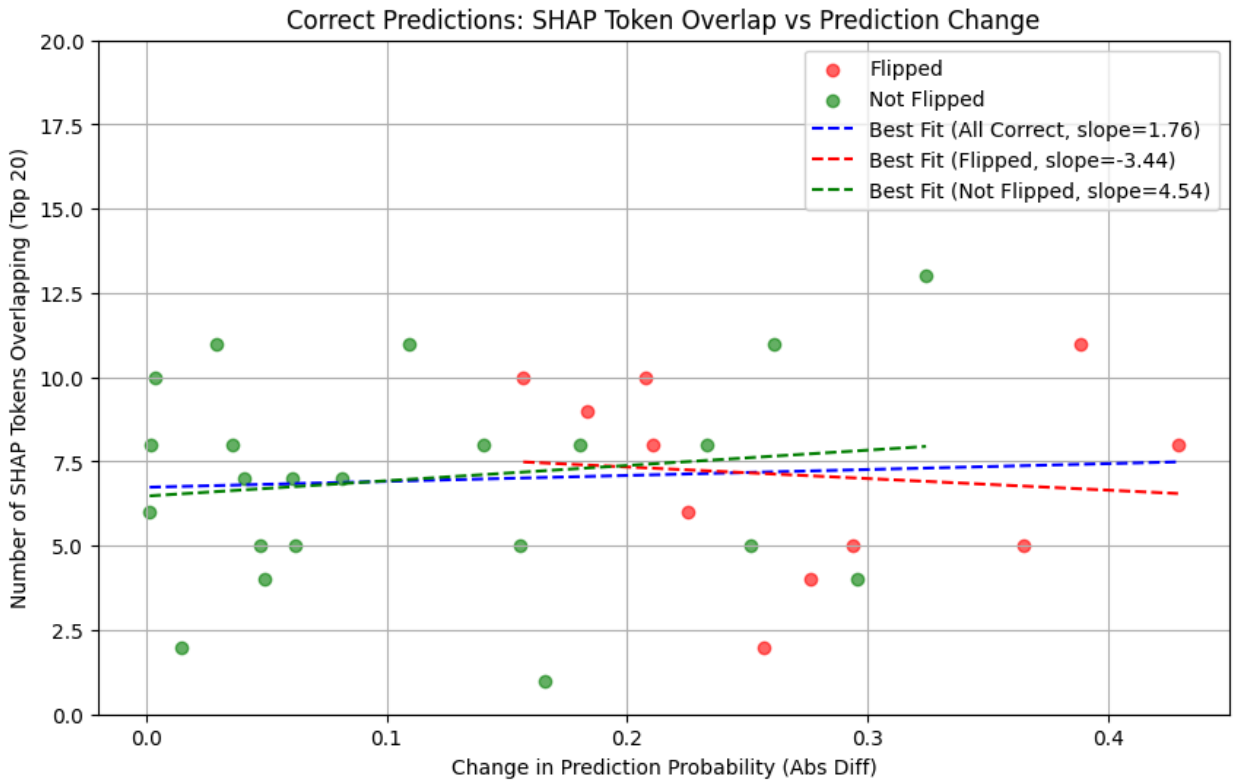
The average no of overlapping top 20 tokens is 7.01, or 35%

For comparison: In the paper Evaluating SHAP's Robustness we are attempting to replicate, the following results are obtained

Patient-specific top 100 genes from each run were selected using the SHAP score.
a) Logistic Regression outcome: the same set of 100 significant genes were identified at each run.
b) Multi-Layer Perceptron outcome: almost no common genes across 10 different runs.
c) XGBoost outcome: around 20% of the significant genes were common in 10 runs.

In comparison, we may see that though SHAP on the complex InLegalBERT is far less than robust on the simple Logistic Regression, SHAP however shows better robustness on InLegalBERT than other complex models such as the multi-layer Perceptron and XGBoost.

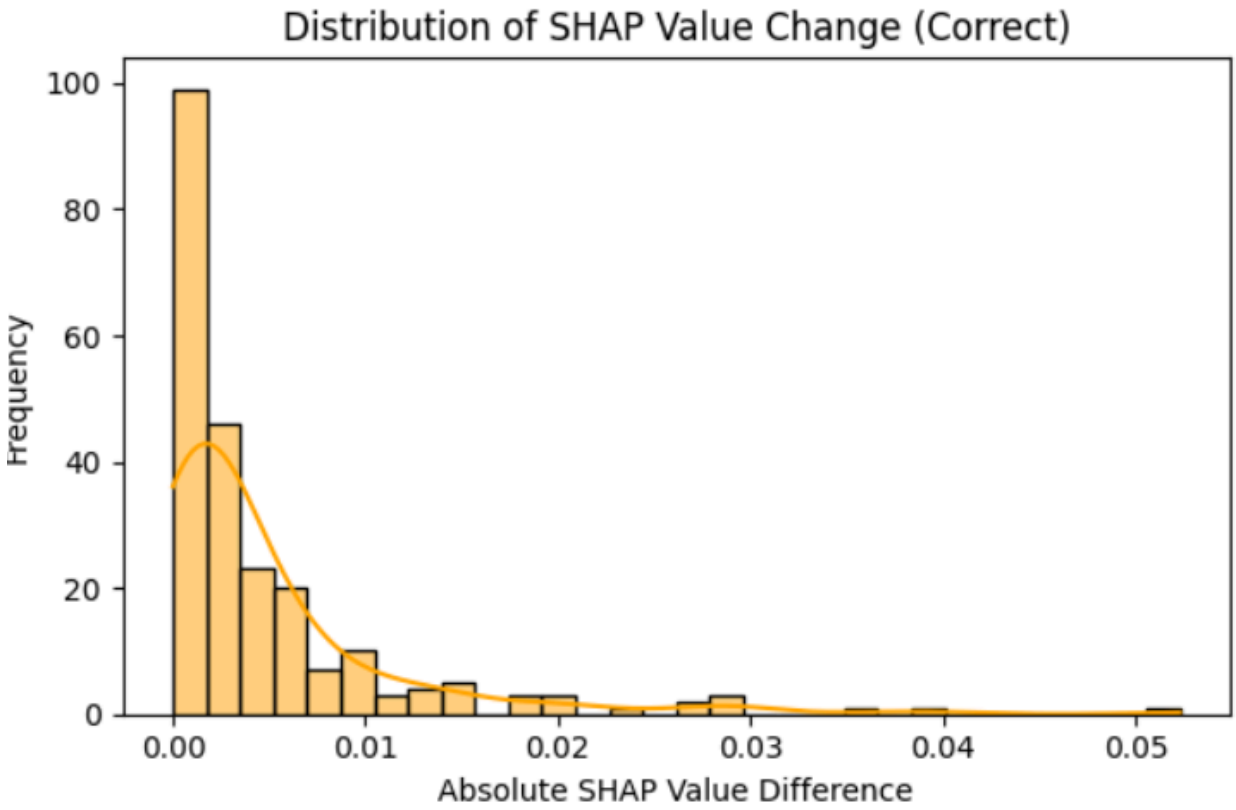
6.2.2 SHAP Token Overlap vs Prediction Change



This chart depicts the no of overlapping top 20 SHAP tokens between the unchanged and adversarial inputs versus the change in model prediction probability between the unchanged and adversarial inputs

Ideally, the number of overlapping tokens should decrease with prediction probability, ie, slope should be negative. As in, if the model shows little change, if SHAP on the model is robust, it should show little change as well.
However, here there appears to be a slight increase in the number of overlapping tokens, i.e. the slope is positive, 1.76.

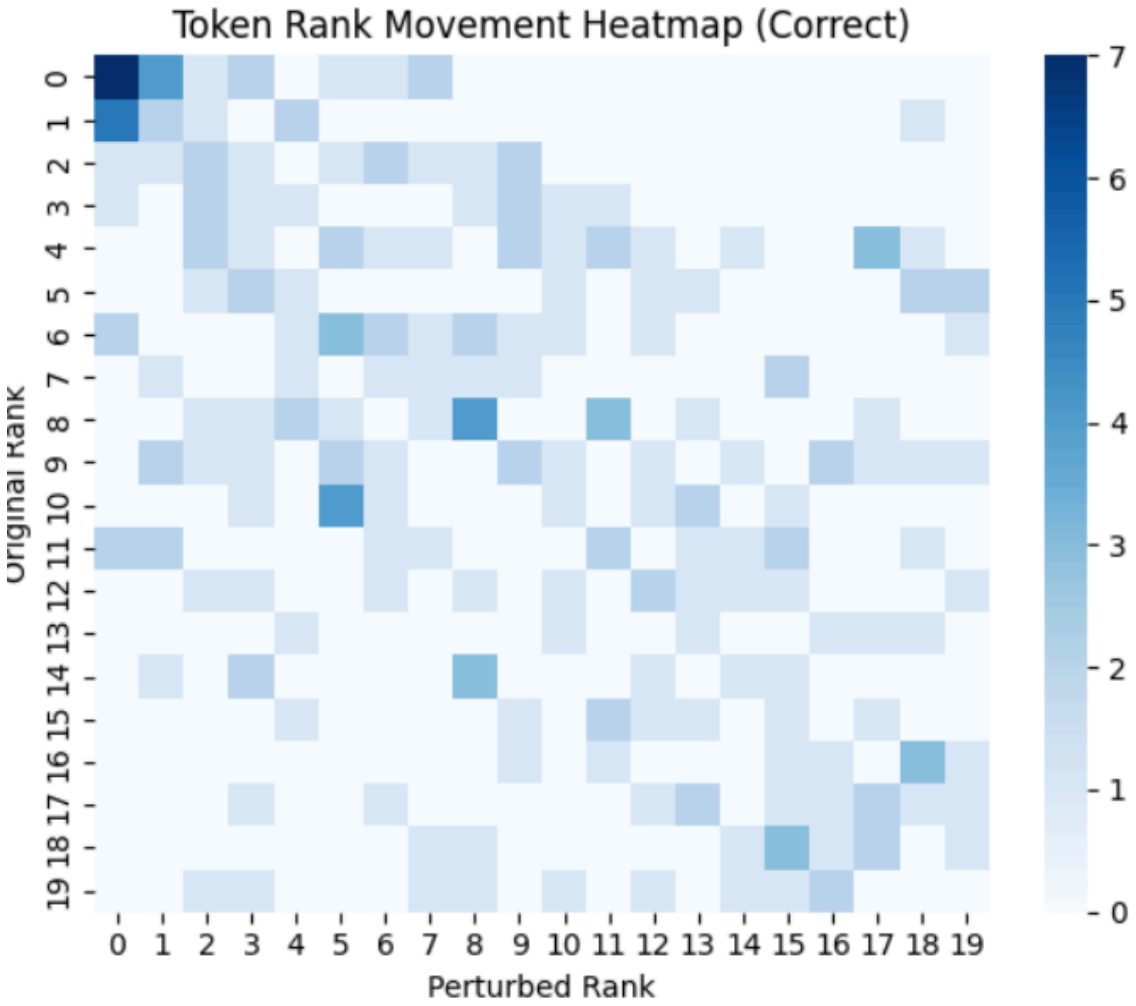
6.2.3 Distribution of SHAP Value Change



Correct] Mean SHAP Value Change: 0.0048, Median: 0.0023

This chart depicts now often Shap values of the top 20 tokens changed by a certain amount

Mean of all SHAP values: 0.0056
Mean of change in SHAP values: 0.0048
This means top-20 SHAP values change by 85% on average- a very large change.



In this chart the colour intensity at a cell i,j represents how many times a token ranked i in the original inputs shifted to rank j in the perturbed inputs.

- The “darker” the plot is along the diagonal, the more robust the model.
- The model visibly shows moderate robustness, but not strong robustness

6.2.4 Summary:

- *In comparison with other more complex models, XGBoost and* Multi-Layer Perceptron cited in *the paper Evaluating SHAP’s Robustness, SHAP on InLegalBERT seems more robust, though not very high.*
- *However, SHAP values vary greatly for the top 20 tokens after adversarial attacks*
- *Furthermore, the slope of the regression line is an area for potential future research.*

Overall, SHAP shows mixed to poor robustness against FGSM attacks

However small sample size due to computational constraints potentially limits the strength of our results.

7. Conclusion

From our statistical analysis, it is clear that SHAP is robust in LJP models to some extent against particular types of noise, partially supporting our hypothesis that SHAP is a robust post-hoc model-agnostic interpretability method. Furthermore, we observed that changes in SHAP values were not strongly correlated with prediction flips. SHAP stability degraded more slowly than prediction stability as noise increased, which implies that improving a model's predictive robustness could also enhance its interpretability stability.

However, our findings reveal that SHAP was fairly vulnerable to FGSM attacks. In particular, the vulnerability was present when the attacks were more structured and specific. Top-k feature consistency dropped sharply, SHAP value correlations fell, and feature attribution rankings fluctuated significantly even when predictions remained the same. This means SHAP explanations may not only fail in identifying correct explanations but also mislead users under attack to bogus explanations.

There is post-deployment risk in deploying LJP models because our results show that malicious or noisy inputs can degrade both accuracy and transparency in legal prediction systems. Hence, for now, our results call for more robust training or defensive techniques.

8. Discussion & Next Steps

Our result can guide researchers working in NLLP to consider building more interpretable, lightweight models without sacrificing model stability itself. Furthermore, our framework is reusable to test both the explainability and its robustness for any LJP model.
Although our results show that SHAP is vulnerable to FGSM attacks, its robustness against random noise brings hope to future efforts on making stronger interpretability methods. However, for now, it would help for people taking part in the legal system to use workflows including LJP and NLLP more carefully.

Potential Methodology Improvement Areas:

1. **Better Model Accuracy**: The low accuracy in our model training could be a huge factor for our results in the unreliability of SHAP. Though legal models typically have low accuracy, the pre-trained model's creators claim to have reached 80% AUROC using an InLegalBERT+ BiGRU+ Linear layer pipeline, we could not replicate this result, despite using the same datasets and code, and reached only about 60% accuracy. Thus, we used only InLegalBERT for classification. Though it is a common saying that there is a tradeoff between model interpretability and accuracy, a model with lower accuracy but

the same complexity has less interpretability as it may be highlighting the wrong features in its prediction process and hence these features may be more vulnerable to the attacks as the model.

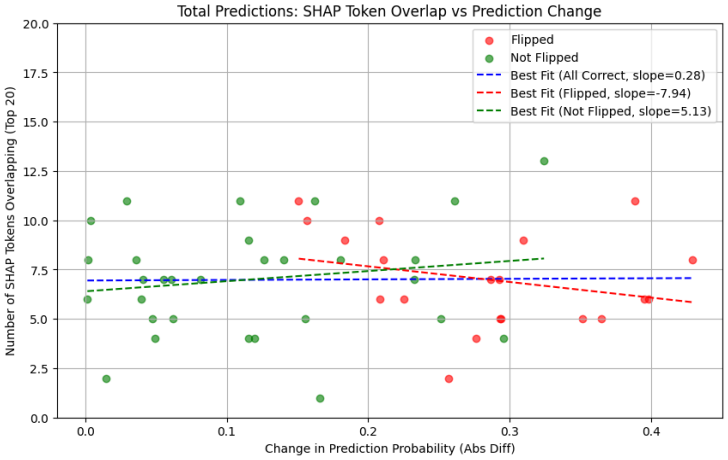
2. **FGSM Truncation:** Being very computationally intensive, FGSM requires a large amount of memory, so due to computation constraints, we could only implement FGSM on tokens on the first 600 characters. This corresponds to approximately only the first 100-120 words, however, these, according to Indian Kanoon, typically correspond to the facts and issues sections.
3. **Better dataset:** The ILDC dataset was not segmented by rhetorical roles, and contained statements about the judges final decision within the last few lines, and thus we removed the last few sentences of each case. It also contained several typos.

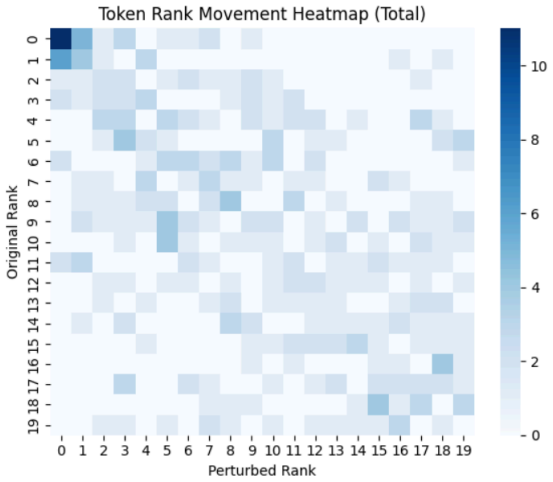
Next Steps (Specificity, formatting)

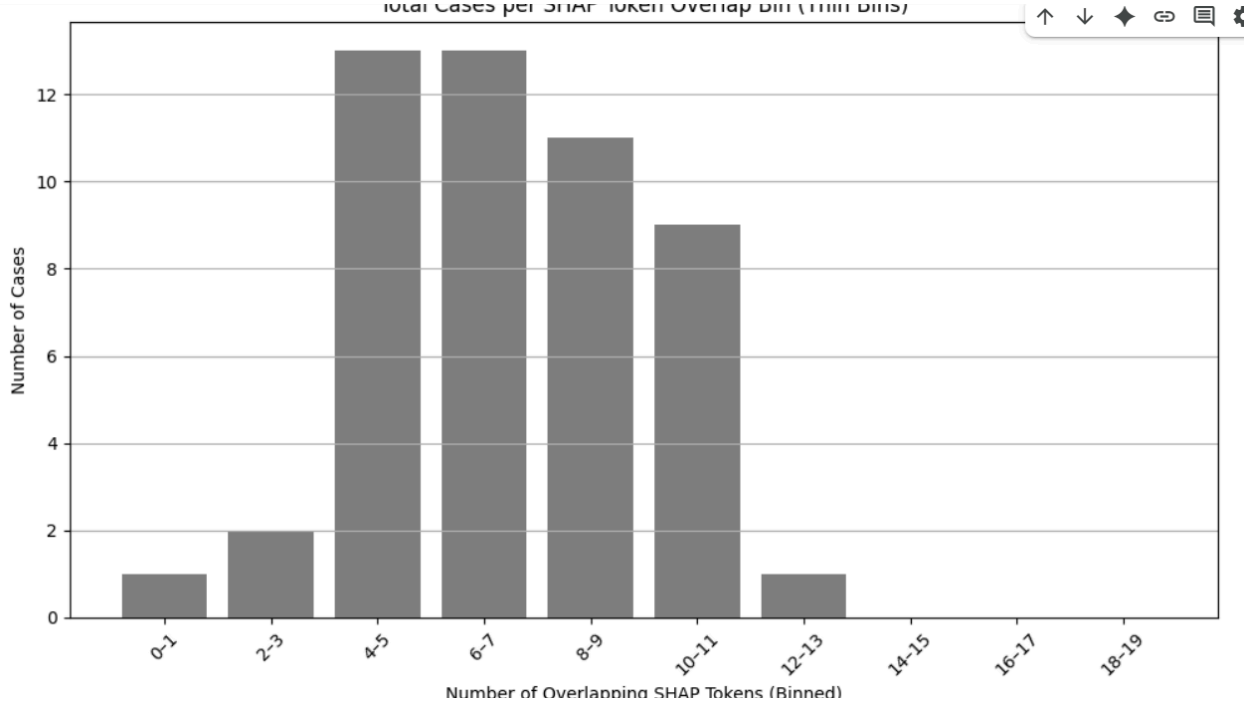
1. **Improve Model Accuracy**
   We could potentially try to get the model to a higher accuracy by retrying the InLegalBERT+BiGRU+Linear Layer pipeline with the highest accuracy according to the benchmark that we had previously tried, but could not get enough accuracy on.
2. **Other datasets/models**
   Post-fellowship, we definitely would like to test on other legal datasets and models available, including those from other countries. InLegalBERT has been proven to have increased accuracy on the ECtHR-B Dataset (European Court of Human Rights) and the UKSS Dataset (UK Supreme Court cases)
3. **Legal-trained LLMs**
   Legal-trained LLM's such as InLegalBERT could also be explored further, but as general LLMs have low interpretability, so we are hesitant about the level of interpretability a legal-specific LLM could hold.
4. **Other interpretability techniques**
   We also would like to try out the XAI techniques, both model-agnostic methods such as LIME and Anchors that we had previously ruled out, and model-specific methods such as using attention weights to see if they are more resistant to attacks compared to SHAP.

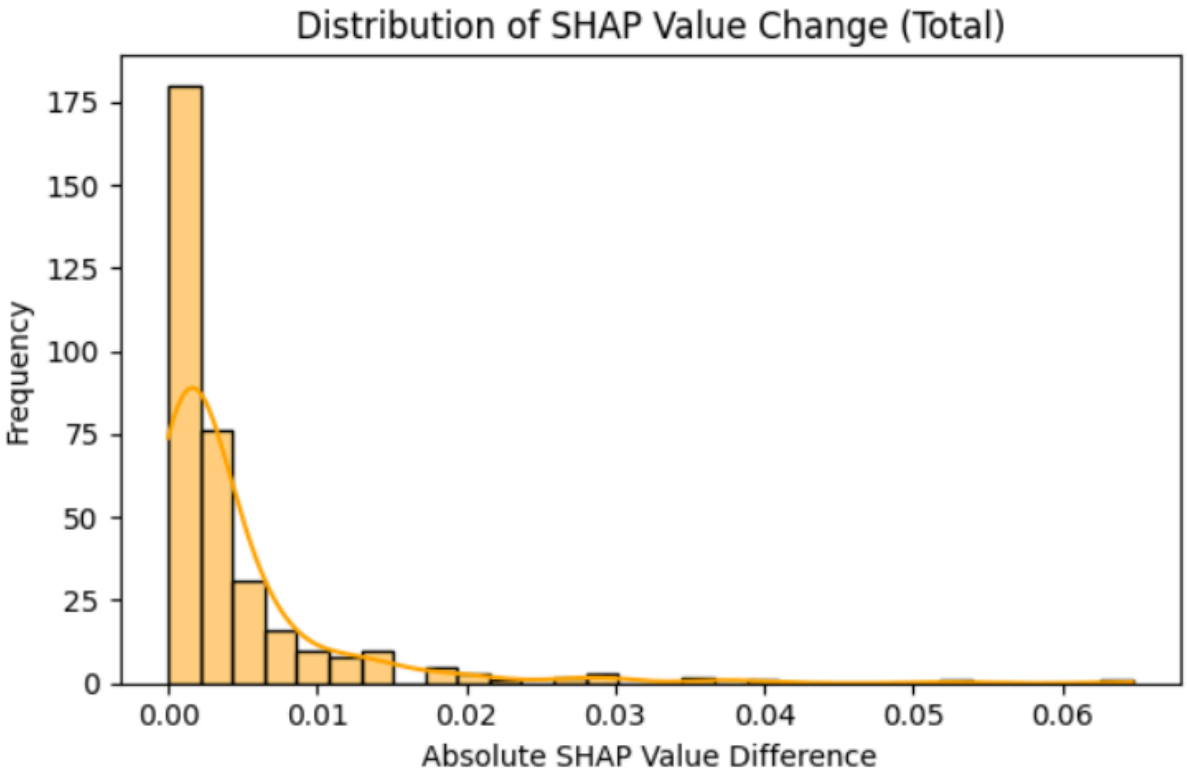
Appendices

FGSM on the entire dataset



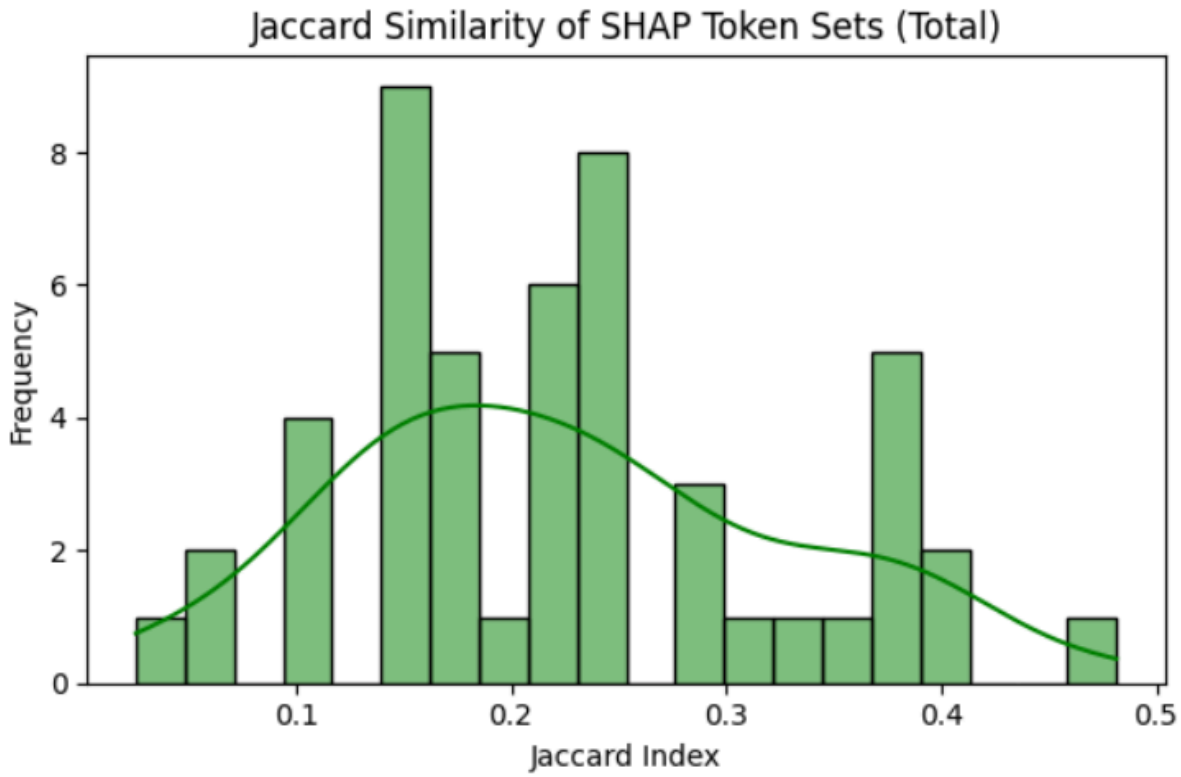




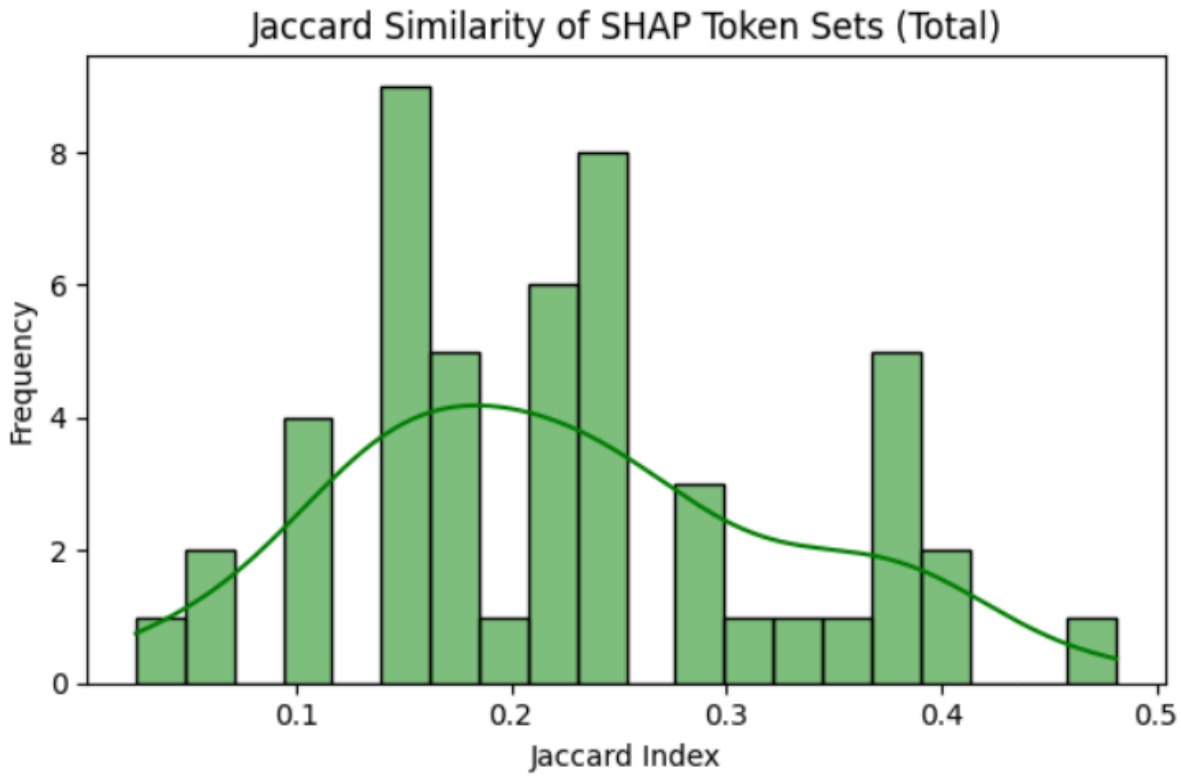



```
Total] Mean SHAP Value Change: 0.0045, Median: 0.0020
```

Jaccard Similarity Plots FGSM



[Total] Mean Jaccard: 0.2253, Median: 0.2121




```
[Total] Mean Jaccard: 0.2253, Median: 0.2121
```

Statistics

Semantic Attack:

SHAP Value Changes:

attack_type	intensity	mean	std	min	max
paraphrase	0.1	0.0001	0.0001	0.0	0.0002
	0.2	0.0001	0.0001	0.0	0.0002
	0.3	0.0001	0.0001	0.0	0.0002
	0.4	0.0001	0.0001	0.0	0.0003
	0.5	0.0001	0.0001	0.0	0.0003
reorder	0.1	0.0002	0.0002	0.0	0.0007
	0.2	0.0002	0.0002	0.0	0.0007
	0.3	0.0003	0.0003	0.0	0.0009
	0.4	0.0003	0.0002	0.0	0.0008
	0.5	0.0003	0.0003	0.0	0.0009
synonym	0.1	0.0002	0.0002	0.0	0.0007
	0.2	0.0003	0.0003	0.0	0.0010
	0.3	0.0003	0.0003	0.0	0.0009
	0.4	0.0002	0.0002	0.0	0.0007
	0.5	0.0003	0.0003	0.0	0.0010
typo	0.1	0.0003	0.0003	0.0	0.0010
	0.2	0.0003	0.0003	0.0	0.0010
	0.3	0.0003	0.0003	0.0	0.0009
	0.4	0.0003	0.0003	0.0	0.0011
	0.5	0.0003	0.0003	0.0	0.0009

SHAP Correlations:

attack_type	intensity	mean	std	min	max
paraphrase	0.1	0.9611	0.0573	0.8393	1.0000
	0.2	0.9611	0.0573	0.8393	1.0000
	0.3	0.9611	0.0573	0.8393	1.0000
	0.4	0.9219	0.0851	0.7887	1.0000
	0.5	0.9219	0.0851	0.7887	1.0000
reorder	0.1	0.7603	0.1664	0.4513	0.9723
	0.2	0.6966	0.1283	0.4836	0.8839
	0.3	0.6491	0.1438	0.3179	0.8143
	0.4	0.5802	0.2273	0.1233	0.8614
	0.5	0.5732	0.2079	0.2380	0.8591
synonym	0.1	0.6328	0.1780	0.3094	0.8884
	0.2	0.6022	0.2195	0.2021	0.8592
	0.3	0.5888	0.1566	0.4012	0.8079
	0.4	0.5479	0.2118	0.1554	0.8444
	0.5	0.5572	0.2141	0.0346	0.8901
typo	0.1	0.5249	0.1563	0.3082	0.7373
	0.2	0.5191	0.2148	0.1570	0.8020
	0.3	0.5520	0.1701	0.2135	0.7964
	0.4	0.5727	0.2357	0.0558	0.7918
	0.5	0.5609	0.1755	0.3105	0.8478

Embedding Noise Attack

SHAP Value Changes:

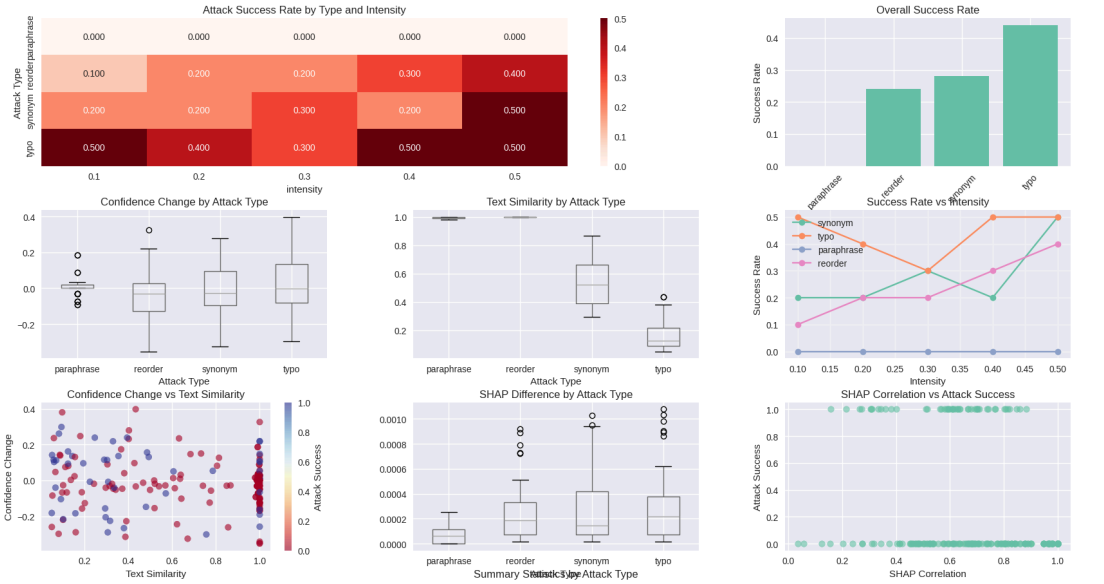
noise_type	noise_level	mean	std	min	max
gaussian	0.01	0.0000	0.0000	0.0000	0.0002
	0.05	0.0002	0.0002	0.0000	0.0005
	0.10	0.0002	0.0002	0.0001	0.0007
	0.15	0.0003	0.0002	0.0001	0.0007
	0.20	0.0003	0.0003	0.0001	0.0009
	0.30	0.0003	0.0002	0.0001	0.0007
targeted	0.01	0.0000	0.0000	0.0000	0.0001
	0.05	0.0001	0.0001	0.0000	0.0005
	0.10	0.0002	0.0002	0.0000	0.0006
	0.15	0.0002	0.0002	0.0001	0.0006
	0.20	0.0003	0.0002	0.0001	0.0007
	0.30	0.0003	0.0002	0.0001	0.0007
uniform	0.01	0.0000	0.0000	0.0000	0.0001
	0.05	0.0001	0.0001	0.0000	0.0003
	0.10	0.0002	0.0002	0.0000	0.0007
	0.15	0.0002	0.0002	0.0000	0.0007
	0.20	0.0002	0.0002	0.0001	0.0006
	0.30	0.0003	0.0002	0.0001	0.0008

SHAP Correlations:

noise_type	noise_level	mean	std	min	max
gaussian	0.01	0.9390	0.1117	0.6327	0.9983
	0.05	0.5769	0.2978	-0.0125	0.9600
	0.10	0.3703	0.2728	-0.1386	0.8304
	0.15	0.4337	0.2101	0.1058	0.7023
	0.20	0.4730	0.2818	-0.0664	0.8121
	0.30	0.5416	0.1663	0.2642	0.7929
targeted	0.01	0.9739	0.0422	0.8587	0.9991
	0.05	0.6488	0.3330	0.0975	0.9539
	0.10	0.5509	0.2106	0.3016	0.9022
	0.15	0.4488	0.2310	0.0683	0.7912
	0.20	0.3912	0.2628	-0.0799	0.7703
	0.30	0.4350	0.2119	0.1122	0.7601
uniform	0.01	0.9822	0.0332	0.8897	0.9993
	0.05	0.6932	0.3706	-0.0527	0.9778
	0.10	0.5588	0.2490	0.2219	0.9590
	0.15	0.4925	0.2708	0.0560	0.8712
	0.20	0.4194	0.2365	0.0917	0.7741
	0.30	0.3173	0.1951	0.0720	0.6109

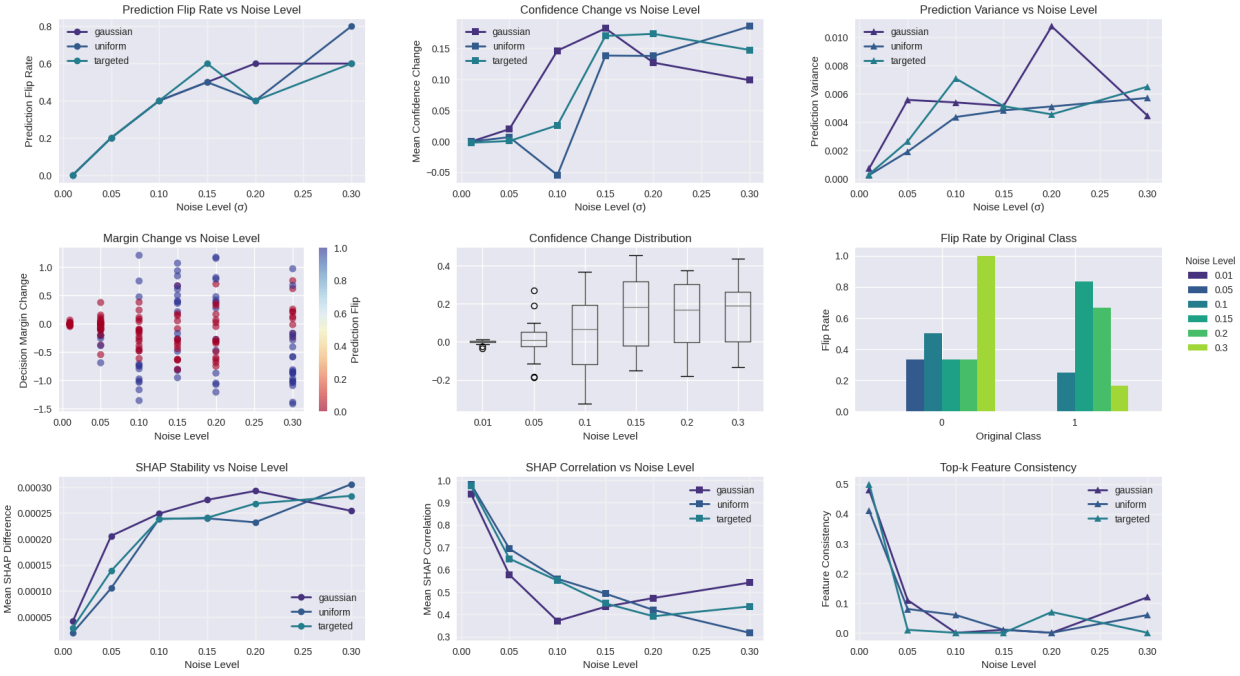
Visualizations

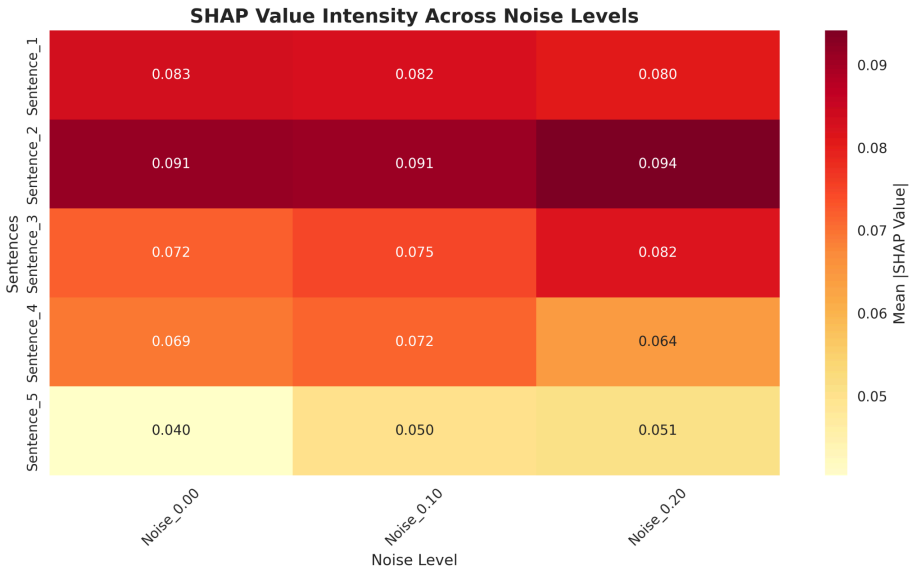
Comprehensive Text Attack Analysis Results



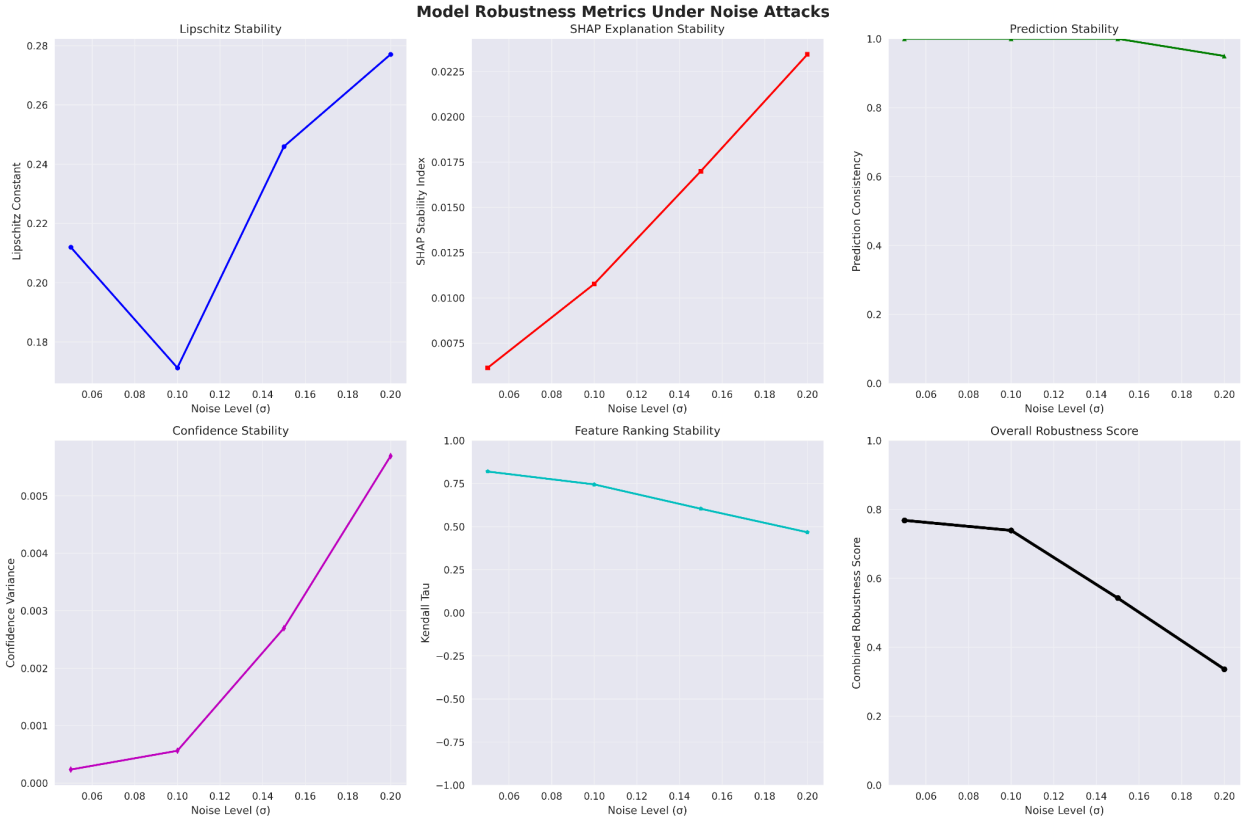
Attack Type	Success Rate	Avg Conf Change	Avg Similarity	Count
synonym	0.280	-0.017	0.541	50
typo	0.440	0.019	0.166	50
paraphrase	0.000	0.006	0.993	50
reorder	0.240	-0.042	1.000	50

InLegalBERT Embedding Noise Attack Analysis





Heatmap for SHAP Value Differences with different amounts of noise



Additional metrics for model robustness

References

Medvedeva, M., & McBride, P. (2023). *Legal Judgment Prediction: If You Are Going to Do It, Do It Right*. In *Proceedings of the Natural Legal Language Processing Workshop 2023* (pp. 73–84). Association for Computational Linguistics. https://aclanthology.org/2023.nllp-1.9.pdf ACL Anthology

Griffin, A. (2016, October 24). Robot judges could soon be helping with court cases. *The Independent*. https://www.the-independent.com/tech/ai-judge-robot-european-court-of-human-rights-law-verdicts-artificial-intelligence-a7377351.html The Independent

Paul, S., Mandal, A., Goyal, P., & Ghosh, S. (2023). Pre-trained language models for the legal domain: A case study on Indian law. In *Proceedings of the 19th International Conference on Artificial Intelligence and Law (ICAIL 2023)*. https://doi.org/10.48550/arXiv.2209.06049 arXiv

Raj, R., & Devi, V. S. (2023). Adversarially robust neural legal judgement systems. *arXiv*. https://doi.org/10.48550/arXiv.2308.00165 arXiv

Ribeiro, M. T., Singh, S., & Guestrin, C. (2018). Anchors: High-precision model-agnostic explanations. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 32, No. 1, pp. 1527–1535). AAAI Press. https://homes.cs.washington.edu/~marcotcr/aaai18.pdf Homes CS Washington

Vasdani, T. (2020, February 5). Robot justice: China's use of Internet courts. *Law360 Canada*. https://www.law360.ca/ca/articles/1750396/robot-justice-china-s-use-of-internet-courts judicature.duke.edu

Zhong, H., Xiao, C., Tu, C., Zhang, T., Liu, Z., & Sun, M. (2020). How does NLP benefit legal system: A summary of legal artificial intelligence. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (pp. 5218–5230). Association for Computational Linguistics. https://aclanthology.org/2020.acl-main.466.pdf ACL Anthology

---

Op-ed / News & Reports

Medvedeva, M. (2025, July 24). Law is ready for AI, but is AI ready for law? *Internet Policy Review*. https://policyreview.info/articles/news/ai-ready-law/2023 Internet Policy Review

Grimm, P. W., Coglianese, C., & Grossman, M. R. (2024). AI in the courts: How worried should we be? *Judicature*, 107(3). Bolch Judicial Institute, Duke Law School. https://judicature.duke.edu/articles/ai-in-the-courts-how-worried-should-we-be/ judicature.duke.edu

---

ArXiv Preprints

Devireddy, K. (2025). A comparative study of explainable AI methods: Model-agnostic vs. model-specific approaches. *arXiv*. https://doi.org/10.48550/arXiv.2504.04276 arXiv

Ghorbani, A., Abid, A., & Zou, J. (2019). Interpretation of neural networks is fragile. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 33, pp. 3681–3688). AAAI Press. https://doi.org/10.48550/arXiv.1710.10547

Work and Split-up Details:
https://docs.google.com/document/d/1MKsHN5wEH6uiydXyMHMUgdbTLyCC8XJDANR0Ja2Ljls/edit?usp=sharing