National College of Ireland

Configuration Manual

MSc Research Project
MSc in Data Analytics

Atharva Sanjay Bhalilkar
Student ID: 23299371

School of Computing
National College of Ireland

Supervisor: Prof. Jaswinder Singh

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name: Atharva Sanjay Bhalilkar

Student ID: 23299371

Programme: MSc Data Analytics **Year:** 2025

Module: MSc Research Project

Lecturer: Prof. Jaswinder Singh

Submission Due Date: 11/08/2025

Project Title: Interpretable Multi-Label Rule Extraction from Financial Regulatory Texts Using Transformer Models: A Comparative Study on EURLEX

Word Count: 1700 **Page Count:** 10

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Atharva Sanjay Bhalilkar

Date: 11/08/2025

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Atharva Sanjay Bhalilkar
Student ID: 23299371

1 Introduction

This configuration guide can be described as a guide that covers the setting up, implementation, and repeating the Extracting Rule Representations from Financial Regulatory Documents using Transformer Models project. It records the hardware and software setup, Python packages, data preparation steps, and the pipeline developed used in the implementation, all the way to pre-processing, evaluation and interpretability. The project uses transformer-based language models, mainly LegalBERT, that are fine-tuned over multi-label classification on the EURLEX data pre-filtered to the 100 most common EuroVoc labels.

This is done to compare transformer-based classification with a baseline of a simple rule-based classifier when evaluating the performance in terms of precision, recall, F1 scores, interpretability (using SHAP and LIME) and scalability. Using this manual, other researchers can replicate those experiments and comprehend every phase of the pipeline and use the framework on other comparable classification of legal or financial documents.

2 System Configuration

2.1 Software & Coding Tools

The project developed and implemented was on a hybrid environment:

- **Google Colab** – for GPU-accelerated model training and interpretability analysis.
- **VS Code (local)** – for preprocessing, feature engineering, and evaluation scripts.
- **Python 3.10** – used across all environments.

Google Colab was selected considering the possibility to access to NVIDIA T4 GPU A hardware, which decreased training times significantly. Processing and debugging was performed in VS Code, and dependency management was performed by a Conda virtual environment (***minimal_env***).

2.2 Libraries Used

```python
# Core Utilities
import os
import numpy as np
import pandas as pd
from tqdm import tqdm
import joblib

# Visualization
import matplotlib.pyplot as plt
import seaborn as sns

# Machine Learning & Preprocessing
from sklearn.preprocessing import MultiLabelBinarizer
from sklearn.metrics import classification_report, f1_score, precision_score, recall_score, confusion_matrix
from sklearn.manifold import TSNE
import umap

# NLP & Transformers
import torch
from torch.nn import BCEWithLogitsLoss
from transformers import (
    AutoTokenizer,
    AutoModelForSequenceClassification,
    Trainer,
    TrainingArguments
)

# Interpretability
import shap
from lime.lime_text import LimeTextExplainer
```

Figure 1: Used Libraries / Packages

Packages / Libraires Purpose Overview:

- **os, tqdm, joblib** – File management, progress tracking, and model/data serialization.
- **numpy, pandas** – Data manipulation, numerical computations, and CSV/JSON handling.
- **matplotlib, seaborn** – Data visualisation (confusion matrices, label distributions, dimensionality reduction plots).
- **scikit-learn** – Multi-label binarization, evaluation metrics, and dimensionality reduction (t-SNE).
- **umap-learn** – Alternative dimensionality reduction for visualising high-dimensional embeddings.
- **torch** – PyTorch backend for transformer training.
- **transformers** – Tokenization, model loading, fine-tuning, and evaluation of LegalBERT and other transformer models.
- **shap, lime** – Post-hoc interpretability to explain model predictions.

3 Setting up the Environment

In this project, the code was developed and data preprocessed locally in Visual Studio Code in an Anaconda environment, although model training and analysis of model interpretability require computationally intensive processing which was done in Google Colab on an

NVIDIA T4 GPU. This was a hybrid solution that gave the flexibility of local debugging and speed and scalability of cloud-based GPU acceleration.

Google Colab is specifically well suited to resource-intensive processing like transformer model fine-tuning, because they provide free or low-cost access to GPU-accelerated hardware, without the requirement to have a high-spec locally located machine in order to typically run training. As we did in our case, training on the T4 GPU Ampere reduced the training time by up to 10x against the running on CPU.

Application of the T4 GPU was in:

- Hands-On Tuning of LegalBERT model
- Executing interpretability scripts based on SHAP- and LIME-based models

3.1 Local Environment Setup (VS Code + Anaconda)

Step 1: Install Anaconda and Visual Studio Code.

Step 2: Open a terminal in VS Code and create the Conda environment

```
conda create -n minimal_env python=3.10
conda activate minimal_env
```

Step 3: Install the required dependencies

```
pip install transformers datasets scikit-learn shap lime matplotlib seaborn tqdm
joblib umap-learn
```

Step 4: Select the fin-nlp environment in VS Code (Command Palette → Python: Select Interpreter)

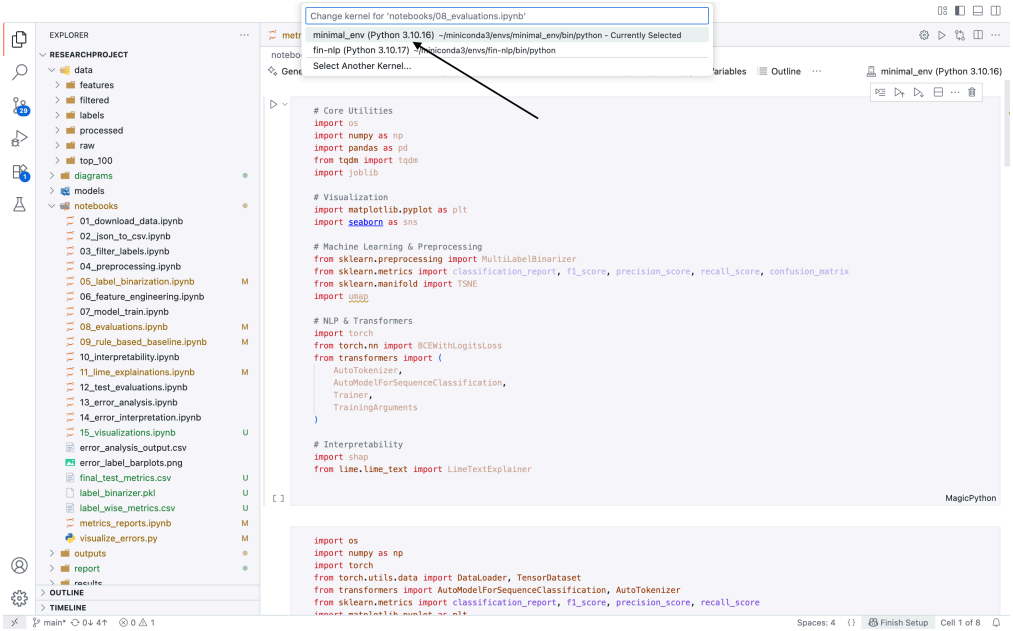

Figure 2: Select Interpreter

Step 5: Run preprocessing, feature engineering, and baseline scripts locally.

3.2 Cloud Environment Setup (Google Colab)

Step 1: Go to: [Google Colab]

Step 2: Create a New Notebook

Step 3: Navigate to **Runtime → Change runtime type**

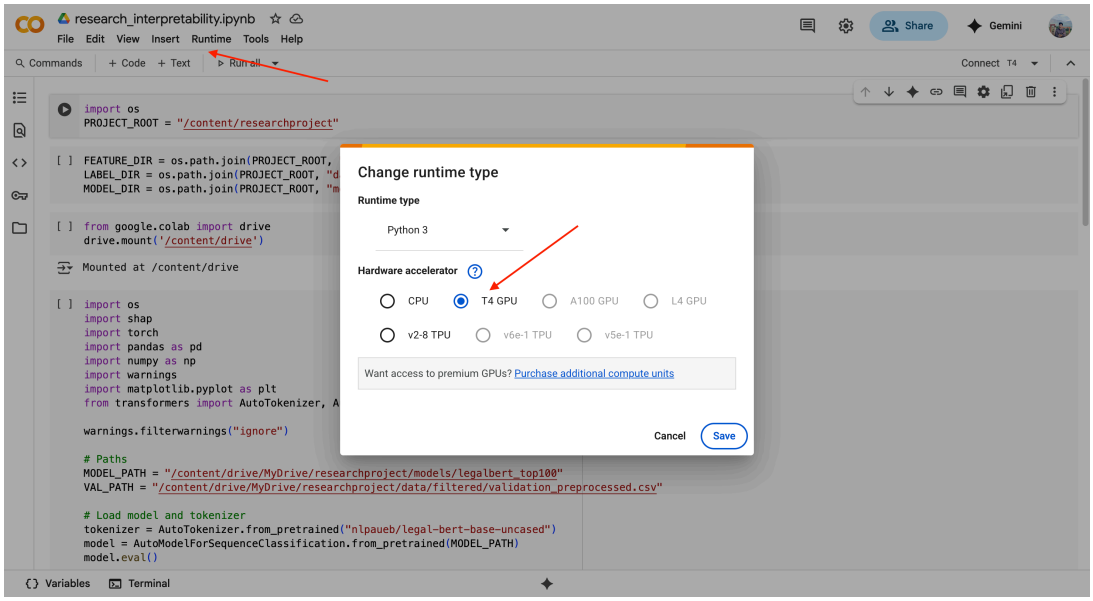

Figure 3: Change Google Colab Runtime Type

Step 4: Set

Runtime Type: Python3
Hardware Accelerator: CPU to T4 GPU

Step 5: Mount Google Drive if accessing project files stored there

! from google.colab import drive
drive.mount('/content/drive')

Step 6: Install all the required libraries and dependencies

!pip install transformers datasets scikit-learn shap lime umap-learn

4 Dataset – EURLEX57K

4.1 Dataset Source

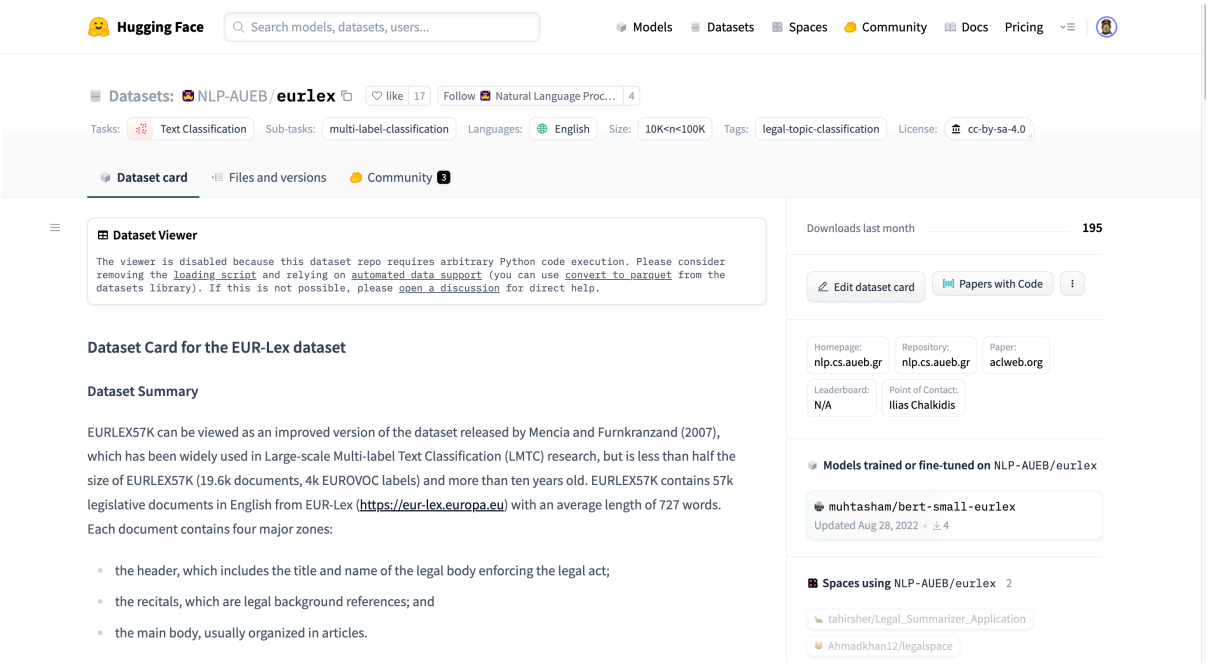

Figure 4: HuggingFace Dataset

Step 1: The data was collected through the Hugging Face Datasets library which uses the official EURLEX repository (Dataset Link)

4.2 Dataset Format

Each record in the raw JSON files follows this structure

```
{
 "celex_id": "32014R0517",
 "text": "full text of the legal document",
 "labels": ["0406", "1206", "1421"]
}
```

Where:

- celex_id uniquely identifies the legal act.
- text contains the merged title and full legal text.
- labels is a list of EuroVoc concept codes.

4.3 Preprocessing Steps

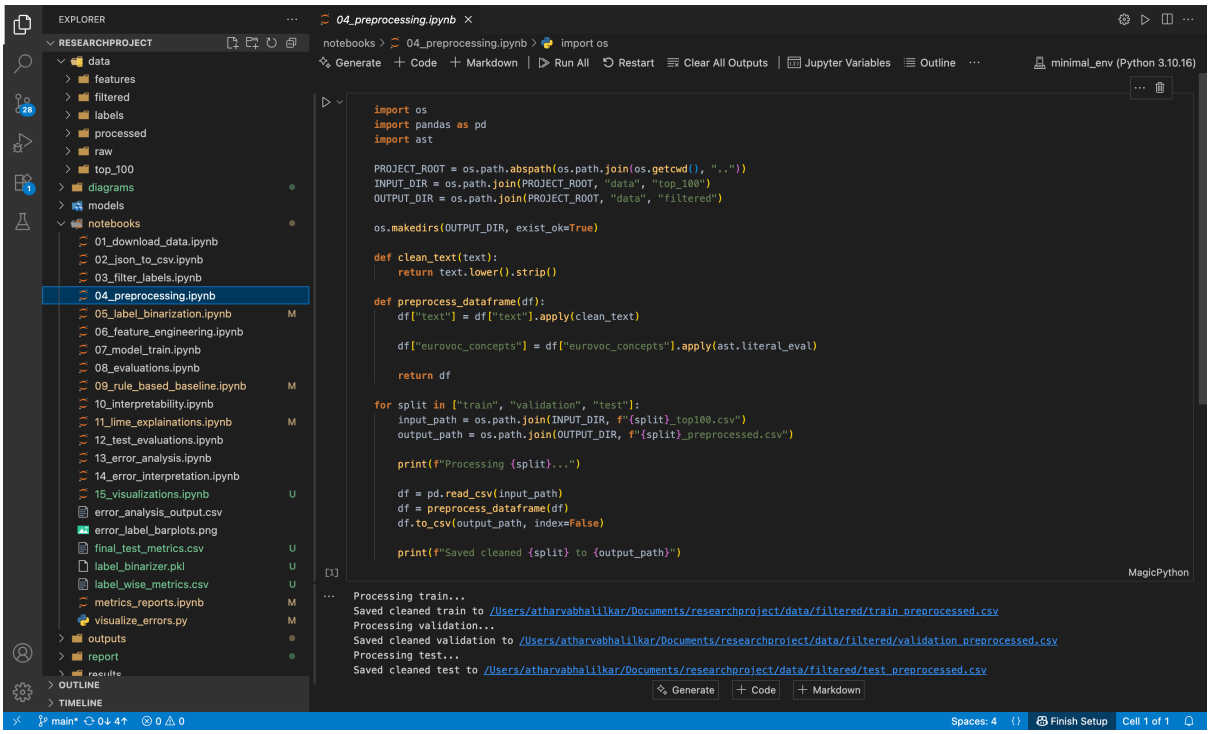

Figure 5. Preprocessing Workflow for EURLEX Dataset

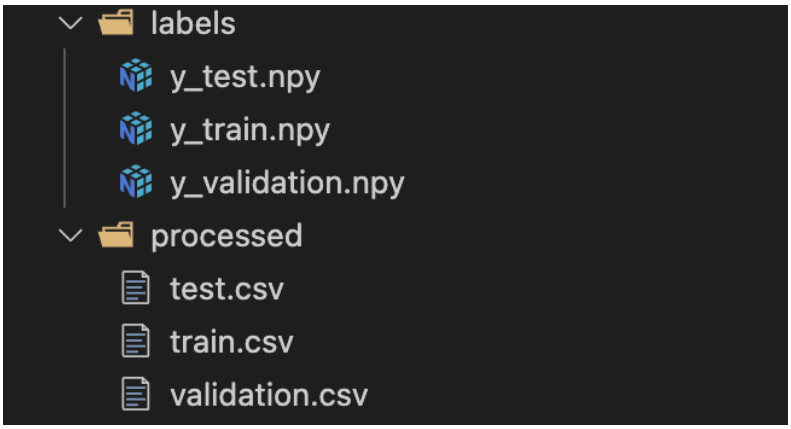

Figure 6. Sample Data Generation

- The major steps involved in preprocessing the text:

    1. Merged **title** and **body** fields into a single text column.
    2. Lowercased text for consistency.
    3. Removed special characters and extra spaces.
    4. Converted labels from lists of strings to binary vectors using `MultiLabelBinarizer`.

- Saved / Generated Files:
    - Preprocessed CSV files: `train_preprocessed.csv`, `validation_preprocessed.csv`, `test_preprocessed.csv`
    - Binary label arrays: `y_train.npy`, `y_validation.npy`, `y_test.npy`
    - Label binarizer: `label_binarizer.pkl`

4.4 Data Understanding

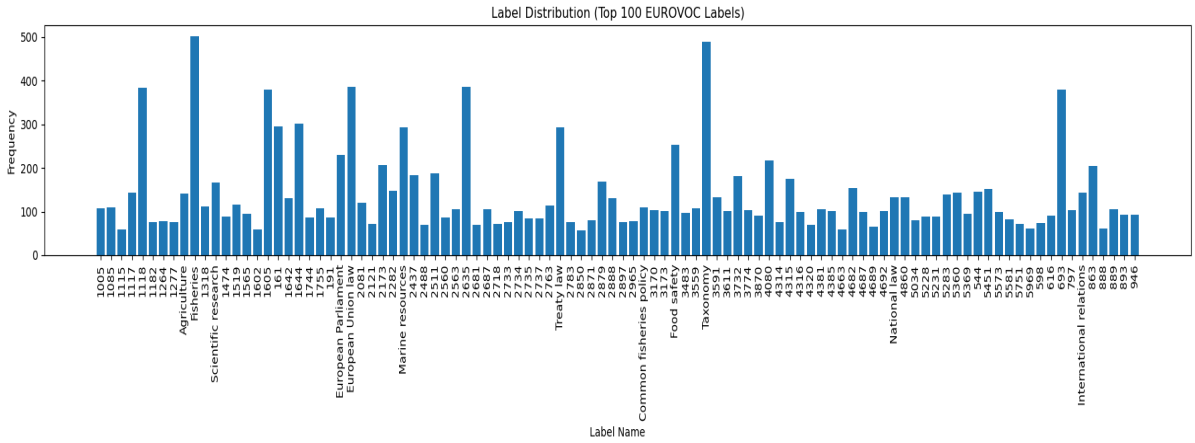

Figure 7. Label Frequency Distribution (After filtering top 100 labels)

The EURLEX dataset first consisted of more than 19,000 legal documents tagged with more than 4,000 EuroVoc tags. This yielded an extremely skewed distribution of labels where a few categories appeared extremely often whereas others ended up appearing only a few times. These matching biases can distort the operation of the model from the point of view of training and undermine the accuracy of forecasting by less frequent labels.

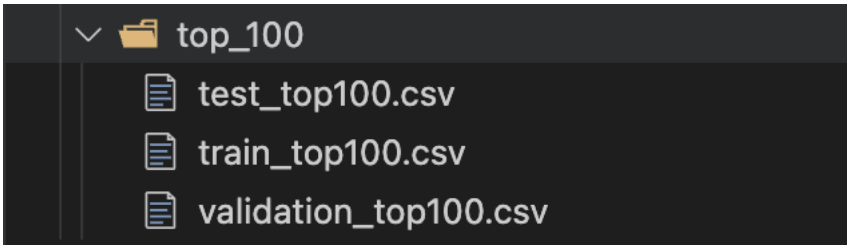

Figure 8. Filtered Labelled Documents

Upon filtering to the top 100 most frequent labels, the dataset was left with about 15,000 training documents, 2,000 test documents and 2,000 validation documents. This labelling process allowed obtaining a more balanced representation of labels at the expense of maintaining a short data set that can be useful when training transformer-based models in a GPU-enabled computational environment.

5 Models & Evaluation

The project will be organized in three key sections in order to compare performance of different transformer architectures to be used in multi-label classification of legal documents based on EURLEX data.

5.1 Modelling

Phase 1 – Transformer Fine-Tuning
The top-100-label subset of the EURLEX against which transformer model such as LegalBERT was fine-tuned. The weights pretraining was loaded, and the last classification was adjusted to the multi-label output by using the sigmoid activation function. Hugging Face tokenizers tokenized the token texts and supported a maximum sequence length of 512 tokens.

Phase 2 – Baseline: Rule-Based Classifier
Traditional rule-based baseline was performed with below keyword match and regular expression matching hurt the concepts of EuroVoc. This was used as a standard on the performance of the transformer models.

Phase 3 – Interpretability and Error Analysis
To explain the predictions of labels, interpretability analysis of the top-scoring transformer was done by using SHAP and LIME. Also, errors were executed to test the shortcomings of the misclassified samples, and one of these shortcomings is confusion among closely related EuroVoc categories.

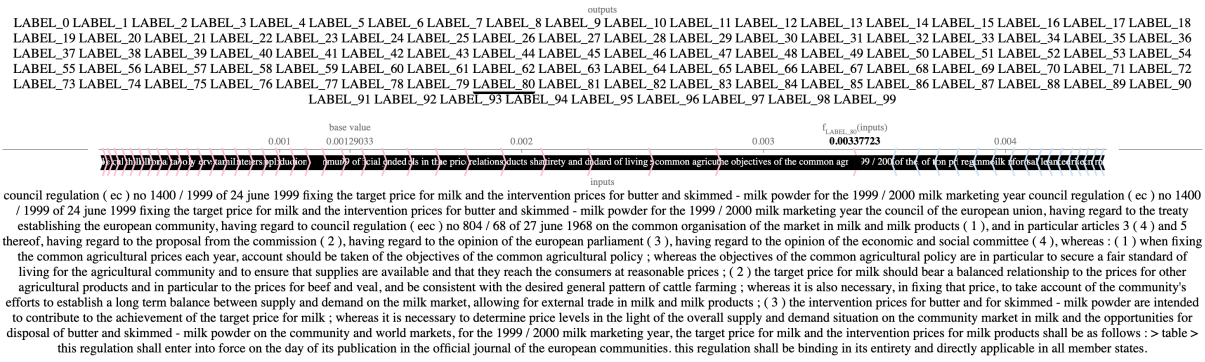

Figure 9. Example Results : SHAP Summaries (Phase 3)

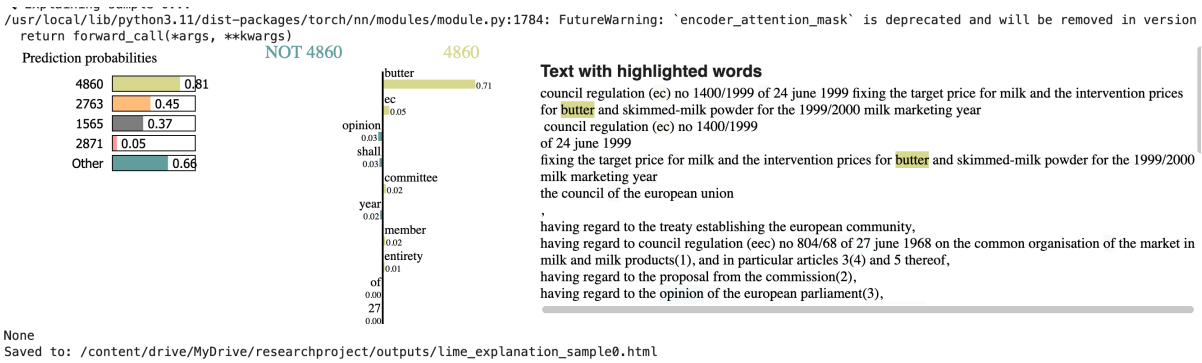

Figure 10. Example Results : LIME Explainations (Phase 3)

5.2 Model Evaluation

For assessing the performance of the models, some quantitative and qualitative measures were combined to give a comprehensive evaluation of the models. Compared to, the quantitative assessment involved calculating micro-averaged F1-scores, micro-averaged precision, and recall, measured overall in the data, as well as on a per-label basis. The performances were measured using a baseline comparison of rule-based classifier and fine-tuned transformer models. SHAP summaries plots as well as LIME were created to increase interpretability where the most relevant tokens related to each prediction are highlighted. Lastly, a confusion matrix and plot of label-wise error rates were also analysed to determine tendencies of misclassification and possible areas of further enhancements in the model.

Metric	Transformer (LegalBERT)
F1 Score (Micro)	0.7774
F1 Score (Macro	0.6993
Precision (Micro)	0.8817
Recall (Micro)	0.6951

Figure 11. LegalBERT Evaluation Metrics

Metric	Score
F1 Score (Micro Avg)	0.2972
F1 Score (Macro Avg)	0.1836
Precision (Micro Avg)	0.3412
Recall (Micro Avg)	0.2703

Figure 12. Rule-Based Classifier Evaluation Metrics

References

Chalkidis, I., Fergadiotis, M., Malakasiotis, P., Aletras, N. and Androutsopoulos, I., 2021. MultiEURLEX – A multi-lingual and multi-label legal document classification dataset for zero-shot cross-lingual transfer. *Journal of Artificial Intelligence Research*, 72, pp.481–520. Available at: https://doi.org/10.1613/jair.1.12753

Chalkidis, I., Fergadiotis, M., Malakasiotis, P. and Androutsopoulos, I., 2020. LEGAL-BERT: The muppets straight out of law school. arXiv preprint arXiv:2010.02559. Available at: https://arxiv.org/abs/2010.02559

Lundberg, S.M. and Lee, S.I., 2017. A unified approach to interpreting model predictions. *Advances in Neural Information Processing Systems*, 30. Available at: https://arxiv.org/abs/1705.07874

Ribeiro, M.T., Singh, S. and Guestrin, C., 2016. "Why should I trust you?": Explaining the predictions of any classifier. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp.1135–1144. Available at: https://doi.org/10.1145/2939672.2939778