Task 6-B: Legal Named Entities Extraction (L-NER) Report

Team: Gaurav Jadhav
Shaily Preetham Kurra
Meghank Reddy Kankanala

Repo link: https://github.com/meghankreddy/NLP-SharedTask

1. ABSTRACT

Some of the judicial systems around the world are taking a lot of time to deliver a judgment due to lack of legal resources that is affecting the lives of many people. NLP models can be used to speed up this process, Identifying important keywords in a judgment can be very useful in this regard. Based on the feature of extracting a few selected entities from the data, we have structured our system. This is the problem that we try to tackle in this paper. We have a huge data set that is generated from judgment documents in India. We have trained an NER model to make sure that it can identify legal entities in a document.

2. INTRODUCTION

India has a complex judicial system that is very slow due to many reasons like low number of judges, less number of courts, appointment problems and so on. So the main objective of this task is to integrate NLP models to make the task efficient.

The legal documents have different entities like court, appellant, respondent etc. We would like to highlight these entities so that it would be easier to go through the document and reduce the resources used to filter these documents.

Similar to NER, legal NER is something that is used to identify legal entities in a document. A legal document can be split into the Preamble and judgment which are passed as input to our model to identify these entities.

There were many NER models referred in the hugging face community, and we discovered the code

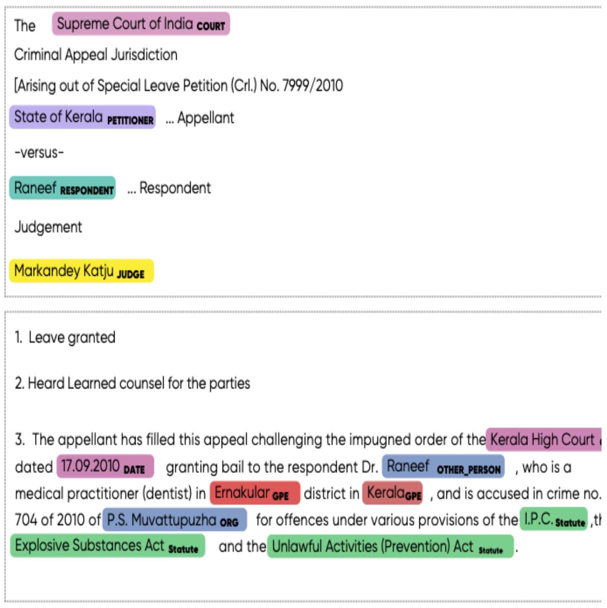
3. BACKGROUND

The input data is divided into two parts preamble and judgment which are JSON objects; they are relatively small input files with sizes 8MB and 14 MB respectively. Each input parameter consists of Id, results, text, and source.

Id: Id indicates the identity of the particular data point
Results: results consist of different entities and their information like start and end index, the entity name, and label(what the entity identifies as)
Text: this contains the entire preamble in preserved format
Source: This contains the information of the source like where the text was obtained from.
Eg: tax_districtcourts judgment
https://indiankanoon.org/doc/1556717/

The output identifies the entities and colors them accordingly for easy identification and observability

NLP REPORT


The Supreme Court of India COURT

Criminal Appeal Jurisdiction

[Arising out of Special Leave Petition (Crl.) No. 7999/2010

State of Kerala PETITIONER ... Appellant

-versus-

Raneef RESPONDENT ... Respondent

Judgement

Markandey Katju JUDGE

1. Leave granted

2. Heard Learned counsel for the parties

3. The appellant has filled this appeal challenging the impugned order of the Kerala High Court
dated 17.09.2010 DATE granting bail to the respondent Dr. Raneef OTHER_PERSON , who is a
medical practitioner (dentist) in Ernakular GPE district in Kerala GPE , and is accused in crime no.
704 of 2010 of P.S. Muvattupuzha ORG for offences under various provisions of the I.P.C. Statute ,t
Explosive Substances Act Statute and the Unlawful Activities (Prevention) Act Statute .

4. SYSTEM OVERVIEW

NLP is being used everywhere and it's a very unique thought by the Indian government to use NLP for extracting Legal named entities. Based on the feature of extracting a few selected entities from the data, we have structured our system. To briefly describe, there are multiple data (json) files both for training and testing. The structure of data is something like this

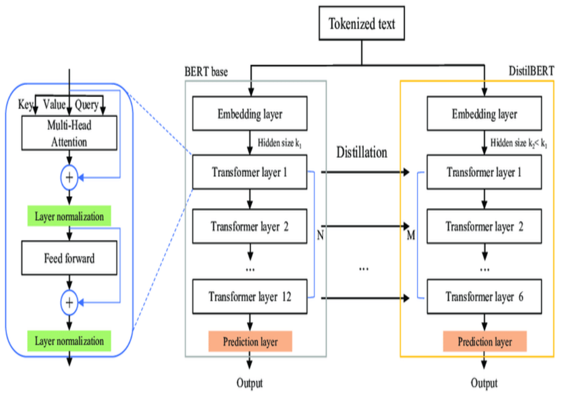
	Train	Dev
Time Range	1950 - 2017	2018 - 2022
Preamble	1560	125
Judgment	9435	949
Entities	29964	3216

Table 1: Data Count for Train and Dev Data

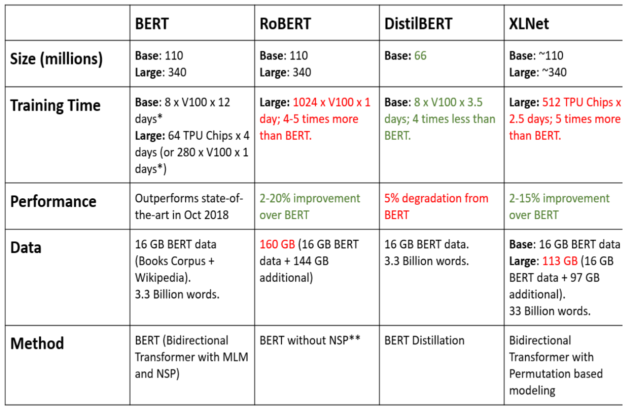
The community of NLP is vast and growing, and one of the fastest growing communities is the Hugging Face. This community and data science platform builds tools that enable users to build , train, and deploy ML models based on Open Source codes and platforms. There are many Named Entity Recognition (NER) The problem setter has guided us to one of their many models deployed on hugging face. We tried multiple models and finally chose a **DistilBERT base model (uncased).**

This model is a distilled version of the BERT base model which was pre trained in a self-supervised fashion. It is trained on raw text and is faster than the BERT model and hence this is the reason we want to go with this model. We also tried to explore some other models, but the limitations to gpu has forced us to choose smaller and faster models. Pre-training is done with 3 main objectives considering the distillation loss which basically establishes the same probability of that of the BERT model. Masked language modeling is also an integral part because it is different from traditional recurrent neural networks (RNNs) that usually see the words one after the other, or from autoregressive models like GPT which internally mask the future tokens. And finally the trained model has to generate the hidden states as close to the BERT model which is calculated by Cosine embedding loss.

The architecture of this model looks something like this.



We have chosen this over the bert-base model because of the capability of handling large datasets and faster functioning. The Bert base model and the distilBERT model has same number of layers i.e., 12-layer, 768-hidden, 12-heads, 110M parameters, but the difference is that the base size and training time is different. There is this picture from the net which gives a detailed description between various models.



	BERT	RoBERT	DistilBERT	XLNet
Size (millions)	**Base**: 110 **Large**: 340	**Base**: 110 **Large**: 340	**Base**: 66	**Base**: ~110 **Large**: ~340
Training Time	**Base**: 8 x V100 x 12 days* **Large:** 64 TPU Chips x 4 days (or 280 x V100 x 1 days*)	**Large:** 1024 x V100 x 1 day; 4-5 times more than BERT.	**Base**: 8 x V100 x 3.5 days; 4 times less than BERT.	**Large:** 512 TPU Chips x 2.5 days; 5 times more than BERT.
Performance	Outperforms state-of-the-art in Oct 2018	2-20% improvement over BERT	5% degradation from BERT	2-15% improvement over BERT
Data	16 GB BERT data (Books Corpus + Wikipedia). 3.3 Billion words.	160 GB (16 GB BERT data + 144 GB additional)	16 GB BERT data. 3.3 Billion words.	**Base**: 16 GB BERT data **Large**: 113 GB (16 GB BERT data + 97 GB additional). 33 Billion words.
Method	BERT (Bidirectional Transformer with MLM and NSP)	BERT without NSP**	BERT Distillation	Bidirectional Transformer with Permutation based modeling

Next comes the **spaCy** library which is a free advanced natural language processing library in python. It is built to understand and process large volumes of data. The problem setters are using spacy-transformers to train the baseline model. This library itself installs all the dependencies required for the model. There is a default base config file which is being specified in the spaCy library. The Legal NER model is specified in that config file with many other details. For allocating GPU we are using pytorch, transformers are spacy-transformers.

5. EXPERIMENTAL SETUP

The experimental setup took us quite a long time and a lot of thoughts to make the execution efficient. Initially we've gathered the data in the form of json files. We also have a default base configuration file which is an auto-generated partial config which is used with 'spacy train'. The initial setup starts with installing python libraries to solve this problem. We are using python 3.8, the command to install the python libraries is "pip install spacy[transformers]". This command installs all the required libraries.

The next step is to collect data for training. There are 2 training json files given in the github repo provided in the task description. The various fields in those files are annotated and collected for training. Next comes the model.

The task revolves around building a spacy model. New nlp spacy model is initialized

There is also a base config file which contains all the parameters to be passed in the model. The model name, gpu accessibility, batch size, tokenizers, epochs, etc., everything is present in the config file. The spacy model generates a new config file based on the basic configurations provided in the base config file and the command used to do this operation is "`!python -m spacy init fill-config base_config.cfg config.cfg`"

Here comes the necessity of having GPUs for training this model. We tried using the system's CPU but it is taking forever. Hence we decided to use Google Collab to get the support for GPUs. The setup was pretty easy in that, first the data files and basic config was uploaded and then we wrote a .ipynb file to run the model. We've also tried to use kaggle, but the setup was a bit complex over there.

The **DistilBERT base model (uncased)** generates 2 output folders model-best and model-last which has tokenizer, transformer model, entities to be recognized, vocab, and a config file. This folder later can be loaded to see the results.

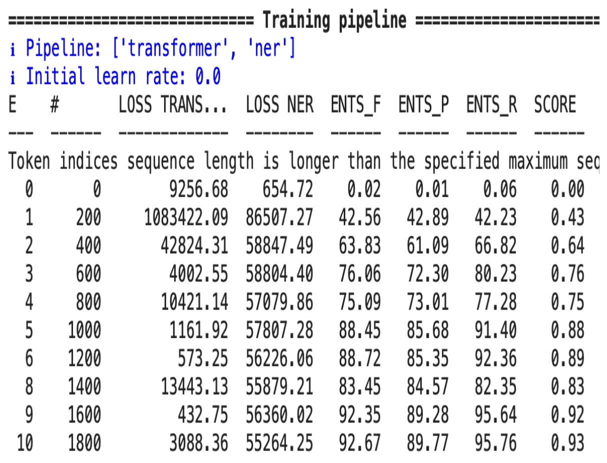
6. RESULTS

The transformer based NER model - distilbert-base-uncased which we used for training the datasets was very effective. This model generates a detailed table which contains transformer loss, NER loss, ENTS_F, ENTS_R, ENTS_P, and F1 score which can be seen in the below pictures. As there are 2 different datasets (one for preamble and one for judgment), the entities are different for both of them. These include COURT4, PETITIONER,

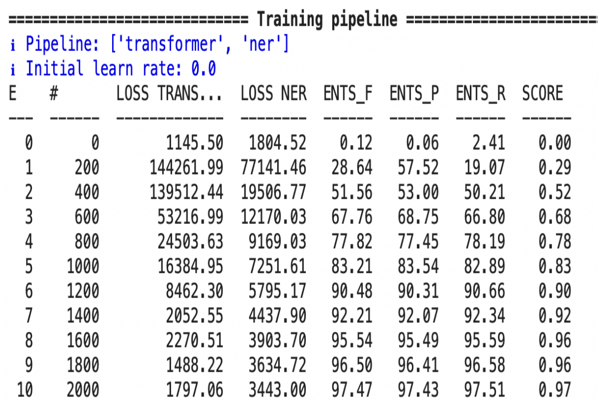
NLP REPORT

RESPONDENT, JUDGE, LAWYER, DATE, ORG, GPE, STATUTE, PROVISION, PRECEDENT, CASE NUMBER, WITNESS, OTHER PERSONS etc.,

Below are the images generated for preamble and judgment datasets respectively.



```
============================ Training pipeline =====================
ℹ Pipeline: ['transformer', 'ner']
ℹ Initial learn rate: 0.0
E    #       LOSS TRANS...  LOSS NER  ENTS_F  ENTS_P  ENTS_R  SCORE
---  ------  -------------  --------  ------  ------  ------  ------
Token indices sequence length is longer than the specified maximum seq
  0       0        9256.68    654.72    0.02    0.01    0.06    0.00
  1     200     1083422.09  86507.27   42.56   42.89   42.23    0.43
  2     400       42824.31  58847.49   63.83   61.09   66.82    0.64
  3     600        4002.55  58804.40   76.06   72.30   80.23    0.76
  4     800       10421.14  57079.86   75.09   73.01   77.28    0.75
  5    1000        1161.92  57807.28   88.45   85.68   91.40    0.88
  6    1200         573.25  56226.06   88.72   85.35   92.36    0.89
  8    1400       13443.13  55879.21   83.45   84.57   82.35    0.83
  9    1600         432.75  56360.02   92.35   89.28   95.64    0.92
 10    1800        3088.36  55264.25   92.67   89.77   95.76    0.93
```



```
============================ Training pipeline ======================
ℹ Pipeline: ['transformer', 'ner']
ℹ Initial learn rate: 0.0
E    #       LOSS TRANS...  LOSS NER  ENTS_F  ENTS_P  ENTS_R  SCORE
---  ------  -------------  --------  ------  ------  ------  ------
  0       0        1145.50   1804.52    0.12    0.06    2.41    0.00
  1     200      144261.99  77141.46   28.64   57.52   19.07    0.29
  2     400      139512.44  19506.77   51.56   53.00   50.21    0.52
  3     600       53216.99  12170.03   67.76   68.75   66.80    0.68
  4     800       24503.63   9169.03   77.82   77.45   78.19    0.78
  5    1000       16384.95   7251.61   83.21   83.54   82.89    0.83
  6    1200        8462.30   5795.17   90.48   90.31   90.66    0.90
  7    1400        2052.55   4437.90   92.21   92.07   92.34    0.92
  8    1600        2270.51   3903.70   95.54   95.49   95.59    0.96
  9    1800        1488.22   3634.72   96.50   96.41   96.58    0.96
 10    2000        1797.06   3443.00   97.47   97.43   97.51    0.97
```

We can see the F1 scores are as high as 0.93 for preamble data, and as high as 0.97 for judgment data. The losses are varying a lot, but we can also observe the ENTS scores being increased consequently.

7. REFERENCES

https://github.com/Legal-NLP-EkStep/legal_NER

https://github.com/topics/spacy

https://github.com/OpenNyAI/Opennyai

https://huggingface.co/models

https://huggingface.co/opennyaiorg/en_legal_ner_sm

https://towardsdatascience.com/whats-hugging-face-122f4e7eb11a

https://sites.google.com/view/legaleval/home?pli=1#h.fbpoqsn0hjeh

https://semeval.github.io/system-paper-template.html

https://huggingface.co/bert-large-uncased