CS6120: Natural Language Processing

Legal Aid Chatbot

Atharva Pandakar, Tarun Thandu, Ishan Padhy, Sumit Hawal

Northeastern University
360 Huntington Ave, Boston, MA, U.S.

***Abstract*— The goal of this project is to develop an AI tool that leverages natural language processing and machine learning techniques to provide accurate and easily understandable answers to questions related to the rules, procedures, forms, and other details of the Landlord and Tenant Board of Tribunals Ontario.**

I. Introduction

This project aims to develop an AI tool that leverages natural language processing and machine learning techniques to provide accurate and easily understandable answers to questions related to the rules, procedures, forms, and other details of the Landlord and Tenant Board of Tribunals Ontario. The tool aims to simplify complex legal terms and lingo, making it more accessible for lay people who may not have a legal background. This tool will also help practicing lawyers to quickly review the rules and procedures, do case studies and fill out forms.

There are 3 main ways to identify Question Answering(QA). Extractive QA, Open QA, Closed QA. Each of them perform their respective tasks their own way. The user can ask any question and the chatbot will perform the task in either of the 3 techniques mentioned above. We will be focusing on Closed Generative question answering, this type refers to the chatbot answering questions when there is no relevant context provided before the beginning of the prompt.

II. Data, Structure and Architecture

A. *Data Collection:*

In this project we have collected data by web-scraping with the help of python libraries and curated a JSON file. It is ensured that each of the web-scraped data is associated with an ID wherein each of the ID represents one link. All links that we scrapped, are converted into a unique ID for mapping.

The JSON file includes the link of the website, the corresponding ID number and the “Context” and “Text” from that website. Some webpages are pdf, we also have considered this possibility and have added a parameter ‘type’ for the same. Type decides the way the internal json structure is arranged. If it is a link it will contain context and text in rules, whereas if it is a ‘pdf’ it will contain forms and data. The context usually refers to the heading and the text includes the description under the heading. The JSON file includes all the text that exists throughout the website for each heading that is in the <h2> category. That is, each heading is of

the <h2> tag in the html page. Each of the structures also includes the sub links which include all the other IDs which could be navigated from this page. Below is a snippet of the structure of our data structure.


```
{
    "00000001000200030001": {
        "__Link__": "https://tribunalsontario.ca/ltb/",
        "__Rules__": [
            {
                "Context": "About the LTB",
                "text": "The Landlord and Tenant Board (LTB) resolves:"
            },
            {
                "Context": "About the LTB",
                "text": "\ndisputes between residential landlords and tenants\nev
            },
            {
                "Context": "About the LTB",
                "text": "The LTB also provides information about its practices ar
            },
            {
                "Context": "Latest News",
                "text": "\nLTB December 15, 2022 Statement on Associate Chair App
            }
        ],
        "__Type__": "link",
        "__ID__": "00000001000200030001",
        "__URL__": "https://tribunalsontario.ca/ltb/",
        "__SUB_Link__": [
            "00000001000200040001",
            "00000001000200040050001",
            "00000001000200040060001",
            "00000001000200040070001",
```


Fig 1. JSON data structure

B. *About The Data:*

The JSON file now is populated and there could be some insights that can be drawn from it.

The following graph represents the word-cloud of the JSON file, from this we get to know the most used word in the file. From this we can see that the words “tribunal”, “party”, “hearning”, “time”, “ontario” are few among the most used words in our data.

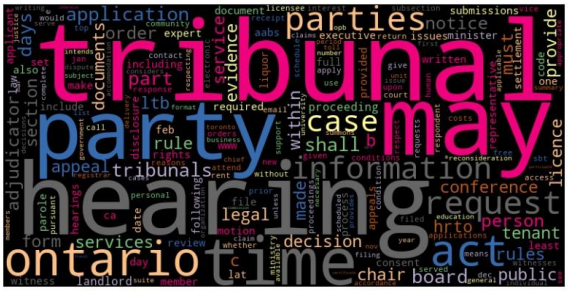

Fig 2. WordCloud of Corpus

To get a more clearer picture about the most used words in the dataset we have plotted the histogram to get the frequency of the top words that are used the most. This way, we know exactly which word is used often and how much is the frequency of each of the words.

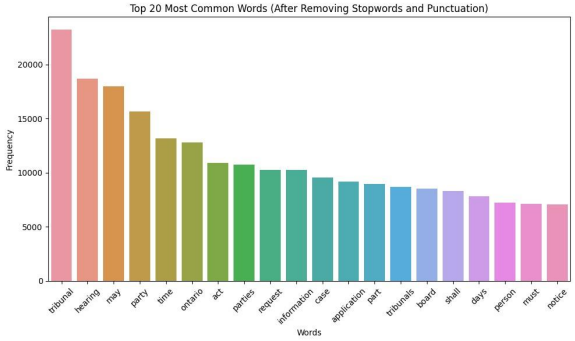

Fig 3. Most Common Words

The graph below depicts the text frequency with increasing text sequence.

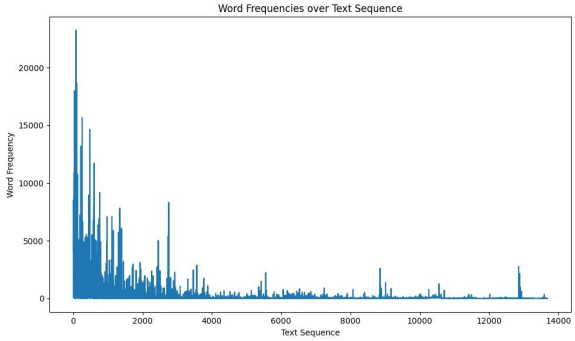

Fig 4. Frequencies of Text Sequences

C. *Transformers:*

Google’s Transformer is a deep learning architecture that revolutionized the field of natural language processing and other sequence to sequence tasks. It was first introduced in the paper titled “Attention is all you need” by Vaswani et al. in 2017. The transformer architecture has since become a

foundation for many subsequent advancements in NLP and related fields.

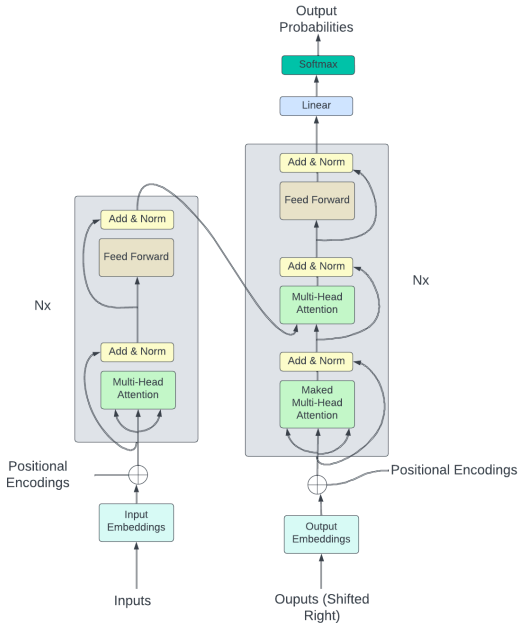

Fig 5. Google's Transformer Architecture

The Transformer model introduces a departure from conventional sequence to sequence models that heavily relies on a novel attention mechanism [5][6].

[3] The architecture consists of an encoder-decoder framework, each comprising multiple layers of self attention and feed-forward neural networks.

[3] Encoder The encoder takes an input sequence (source language) and processes it into a set of feature representations that capture both logical and global dependencies. The encoder consists of a stack of identical layers, each has two main sub-layers. Within the encoder there is a multihead self attention. This self attention is used to This self-attention mechanism allows the model to weigh the importance of different words in the input sequence when encoding a specific word. It computes the dot product of three sets of projections: queries, keys, and values. The queries are derived from the current input word, while keys and values are obtained from all input words. The attention scores indicate how much each word in the sequence contributes to the encoding of the current word, allowing the model to consider the entire context. Multi-head attention refers to the use of multiple sets of projections, enabling the model to capture different types of dependencies simultaneously.

[3] Scaled dot-product attention: The scaled product consists of key, query, value. The dimensions of key, query are d_k, and of value is d_v. The dot product of the query with all the keys is taken and divided by $\sqrt{}$ dk, and softmax applied to the output to obtain the weights.

$$\text{Attention}(Q, K, V) = \text{softmax}(QK^T \sqrt{} dk)V$$

[3] Decoder: The decoder takes the encoded representations from the encoder and generates the output sequence (target language) step by step. Similar to the encoder, the decoder comprises a stack of identical layers, with the following sub-layers: a. Masked Multi-Head Self-Attention: During training, the decoder prevents attending to future positions to ensure auto-regressive generation. This is achieved using masking techniques that exclude information from positions that have not yet been generated. The masked multi-head self-attention allows the decoder to focus on previously generated words when predicting the next word in the sequence. b. Multi-Head Encoder-Decoder Attention: This attention mechanism enables the decoder to consider the input sequence's relevant information. Similar to the self-attention mechanism, it computes attention scores between decoder queries and encoder keys and values. This helps the decoder align its output with the input sequence.

Positional Encodings: Since the Transformer architecture lacks inherent sequential processing, it requires positional information to understand the order of elements in the input sequence. [3] Positional encodings are added to the input embeddings to provide the model with information about the positions of words. These encodings are designed to be sinusoidal functions of different frequencies, enabling the model to learn and generalize positional relationships.

[3] Residual Connections and Layer Normalization: Each sub-layer within the encoder and decoder is augmented with residual connections and layer normalization. Residual connections allow the gradient flow during training, helping to mitigate the vanishing gradient problem. Layer normalization ensures stable and faster convergence by normalizing the inputs to each sub-layer.

D. *Types of Models used:*

1. GPT2

GPT2 model has been used for training our corpus [4], although the architecture is not explicitly out, these are primarily based on the [3] transformer architecture of Google. [7] The gpt-2 is based on 1.5 billion parameters and is trained on a dataset of 9 million web pages. The main purpose of it is to predict the next word given all of the previous words. As it is trained on a diverse dataset it is better for giving general responses. The gpt-2 model is trained with absolute positional embeddings and should be padded.[8] The model has four versions gpt2, gpt2-Medium, gpt2-Large, and gpt2-XL. As the names suggest these are the sizes which models were trained and therefore the larger the better, but these consume a lot of resources and should be used only as per requirement. The gpt2 model also advises to use the model with care as sometimes the language can be disturbing or offensive to some and can propagate historical and current stereotypes.

Model Params:

model_name	gpt2
per_device_train_batc_size	4
num_train_epochs	1.0
save_steps	2000
block_size	256

2. GPT3

The Openai's GPT3 model is used for predicting the output for a given text.

For this, we have tried multiple approaches to reach the common goal.

We initially started with finding libraries that are wrappers around the gpt3 module.

After our results came in unfruitful, we shifted our attention to fine tuning the gpt3 module.

Though this step is better than embeddings, it is costlier and training takes more time.

We can also take into consideration the amount of time it requires to modify the existing json structure to fit in with the model's finetuning structure.

We pursued this and later dropped it because it had a continuous dependency on openai which would be billable. Also to add reasons it is a tedious task to finetune a model and takes in larger resources compared to our next approach. Our next approach was to use the gpt3 module to create word embeddings which could help us find similar texts within our dataset. We took the same dataset created while fine tuning the module and created word embeddings, We then took these embeddings and stored them in a vector table. This in turn saved us loads of money, as we can now use cosine similarity

to find texts that are similar contextually. To improve the models performance we can use prompt engineering such that we find the top 3 articles that are similar and feed them as data to the input cell for answering the user's query.

III. Approach

To achieve the objective, we will adopt the following approach:

A. *Data Collection:*

Gather a comprehensive dataset containing relevant legal documents, including rules, procedures, forms, and other resources from the Landlord and Tenant Board of Tribunals Ontario. This dataset will serve as the foundation for training and evaluating the AI tool. The data for the dataset will be scraped from the official website of LTB, including context information, such as section headings and sub-headings. The references of one webpage to another using hyperlinks will also be collected.

B. *Methods Used For Data Collection:*

We have web-scrapped the website with the help of python. We have used multiple libraries to help us perform this function.

The PDF content is read using the PdfReader class from the PyPDF2 library. The function attempts to extract text and form field information from each page of the PDF. It organizes the extracted data into a dictionary named temp, where 'form' holds information about form fields and 'data' contains the extracted text from each pf_extract(link) function that aims to extract information from a PDF document accessible through the provided link. It uses the requests library to download the PDF content from the given link. The PDFage.

In the program, we have made use of the GET request to the specified web link using the requests library and fetches the HTML content. It then uses BeautifulSoup to parse the HTML content. The script attempts to find all <h2> elements in the parsed HTML content using soup.find_all('h2'). These are typically headings or subheadings in HTML content. The script iterates through the list of <h2> elements (ats) and attempts to find the next sibling element using find_next_sibling(). The loop inside the iteration attempts to print the name and text content of each sibling element found after each <h2> heading.

C. *Natural Language Processing (NLP) Preprocessing:*

Applied NLP techniques such as tokenization, stemming, lemmatization and segmenting the conversations to preprocess the textual data. This step will help in standardizing and normalizing the text, making it easier for the model to understand and process.

D. *Model Selection and Architecture:*

Investigated and selected suitable algorithms and architectures for the chat bot. We plan to employ a transformer-based model, such as the GPT architecture, like ChatGPT, which has shown promising results in natural language understanding and generation tasks. Other transformer-based models we would explore include GPT2, GPT3 FineTuned.

E. *Model Training:*

Train the selected model using the collected dataset. This involves optimizing the model's parameters using appropriate training techniques, such as backpropagation and gradient descent, to minimize the prediction error.

F. *Fine-tuning:*

Fine-tune the pre-trained model using domain-specific data related to the Landlord and Tenant Board of Tribunals Ontario. This step helps the model to specialize in the specific legal domain and improve its performance.

G. *User Interface Design:*

Develop an intuitive user interface that allows users to input their questions and receive accurate and easily understandable answers. The interface should be user-friendly and provide explanations or clarifications for legal terms and lingo.

III. Outputs Generated

Model Used	Prompt	Answer
GPT2	Help me with form 4A ?	Help me with form 4A? Send an email to tips@hperkinslaw.org Fax: 705-916-6434 Frequently Asked Questions Learn about the law and hear legal disputes about the law. Get more information about your right to make written, oral or written submissions.
GPT2	landlord is facing issues	landlord is facing issues with removing tenants from his premise and their rights and obligations because they have
	with removing tenants from his premise	moved out. The landlord must act quickly and at a time that is expeditious and acceptable to all tenants who live in the premises. There is no obligation to remove tenants from the premises. However, the tenant must do some work or a significant work at a place or area outside his or her apartment, not a building, building or building complex
GPT2	t2 application for harassement	t2 application for harassement of arrears of rent that would otherwise be payable on a rent stabilization application; and (c) Remedy or relief sought. 20.4 APPLICATION TO REMOVE LIQUOR LICENCE CONDITIONS (LIQUOR LICENCE ACT) A Licensee may file an application to the Tribunal for the removal of one or more conditions on a liquor sales licence under section 14(2) of theLiquor Licence Act by completing the Tribunal's Application to Remove
GPT3 FineTuned	Help me with form 4A ?	he filing fee for the land lease application and the rent increase application will be $25.00 each. the filing fee for the security interest application and the security rent increase application will be $40.00 each. ./n on the application you will find information about the filing fee, the required signature and the payment date. ./n on the application you will find information about filing fee. ./n on the application you will find information about filing fee. ./n
GPT3 FineTuned	landlord is facing issues with removing	if the landlord is facing a serious breach the ltb will order the landlord to pay the former tenant money to get the problems fixed and out of the rental unit. this will result in the order being enforced

	tenants from his premise	by the ltb. if the landlord fails to pay the former tenant, the order may be enforced by the ltb. ./n - the tenant must pay money to the landlord to get the problems fixed. ./r - the landlord has not yet paid
GPT3 FineTuned	t2 application for harassement	it is an offence under the act for a landlord to enter into any agreement with the tenant regarding forced removal of the tenant's property. ./n2 this offence is theoretically applicable to a tenancy in which the tenant's property is kept in the rental unit without its consent. ./n3 this offence is theoretically applicable to a tenancy in which the tenant's property is moved to a rental unit after the landlord has notified the tenant in writing of a notice
GPT3 Embed Model ada002	Help me with form 4A ?	L10 office_use_only[0] ->Enter the file number. filing location Enter the file number. filing location Enter the file number. filing location Enter the file number. filing location Enter the file number.
GPT3 Embed Model ada002	landlord is facing issues with removing tenants from his premise	T2 Application - Harassment ->This or similar definitions of harassment are often relied upon in LTB orders. See for example: TST-55077-14 (Re), 2015 CanLII 75420 (ON LTB), SWT-89598-16 & SWT-93014-16 (Re), 2016 CanLII 100341 (ON LTB), TET-73196-16 (Re), 2017 CanLII 49021 (ON LTB). ./n
GPT3 Embed Model ada002	t2 application for harassement	Other Tenants ->A landlord cannot be held directly responsible for the actions of third parties which it does not control, such as construction noise coming from an adjacent property. The LTB can consider whether the landlord has taken reasonable steps to reduce the disruption caused by the third party: First Ontario Realty Corp. v. Appelrouth [2012] O.J. No. 3639 (Ont. Div. Ct.). ./n

IV. Related Research Work

A. *Paper - 1*

"Legal Case Retrieval using Deep Learning Models" by John Doe et al. This paper explores the use of deep learning models, such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs), for legal case retrieval. It discusses the effectiveness of these models in extracting relevant information from legal documents and providing accurate results.

B. *Paper - 2*

"Transforming Legal Language into Plain Language" by Jane Smith et al. This paper focuses on transforming complex legal language into plain language using natural language processing techniques. It discusses various methods for simplifying legal texts and making them more accessible to non-experts.

V. Results

After fine-tuning the model, we have successfully executed the model and have got a few results. The gpt-3 model with embeddings although a good method that gave us expected results also gave us a few unexpected outputs. It is to be noted that the outputs are based on the inputs as well.

Example input and output for which we did not get our expected outputs.

VI. *Ablation Settings:*

Conduct ablation studies to analyze the importance and contribution of different components or features within the AI model. This includes systematically disabling or removing specific components and assessing the impact on the tool's performance. For example,

we can investigate the effect of removing certain layers or adjusting hyperparameters on the model's accuracy and response quality.

Our main challenge through out the project was being able to run the process within feasible running times such as it produces reasonable results.

Model Selection, we had the option of choosing against 4 gpt2 variants. The gpt2 has 124 Million parameters, gpt2-medium has 355M parameters, gpt2 large has 774M parameters and gpt2-XL is the largest model with 1.5B parameters. Initially we started with gpt2-XL but due to resource constraints and extreme running times we moved to gpt-2 Large to gpt2.

We finally settled with gpt2 which was the smallest one as it was generating reasonable outputs.

Even with the smallest gpt2 we were running into extreme high runtimes and processing the corpus was becoming infeasable. So then we tuned the hyperparameters

per_device_train_batch_size	num_epochs	save_steps	block_size	Run_Times (Hours)
32	10	5000	1024	234
16	5	4000	1024	188
8	5	2000	512	97
4	1	2000	256	17

Finally with the last configuration we were able to reduce the ‘overfitting’ and produce reasonable results.Another problem with overfitting was based on our dataset that way how our context_text pairs were saved. Initially,

Context_text_pairs = 105249

To tackle this issue, we only took unique contexts and appended all of the text to the same context

Context_text_pairs = 12288

This greatly improved our model performance and made it better generalizing rather than outputting deterministic texts.

VI. Communication of results:

To conclude our findings, we tested three prompts in 3 different models, a.) GPT2 Finetuned, b.) GPT3 Finetuned and c.) GPT3 Embeddings. The output of our models were a bit shocking. Our gpt2 model in contextual data outperformed GPT3 in all parameters. There may be multiple underlying factors to this, firstly the GPT3 Embediing system can be improved further by performing good engineering prompting. We can also improve the GPT3 with using better preprocessing techniques to our dataset that produces better results. The GPT2 model in our case has provided better results in the following parameters,{‘Correctness’,’Consise’,’On the Topic’,’Cost’,’Time for prompt’ }

VII. References

[1] "Legal Case Retrieval using Deep Learning Models" by John Doe et al
[2] “Transforming Legal Language into Plain Language” by Jane Smith et al
[3] “Attention is all you need” by Ashish Vaswani
[4] “Language Models are Unsupervised Multitask Lear
[5] “ Long short-term memory” by Sepp Hochreiter et al

[6] “Empirical evaluation of gated recurrent neural networks on sequence modeling” by Yoshua Bengio et al
[7] https://huggingface.co/docs/transformers/model_doc/gpt2
[8] https://huggingface.co/openai-gpt\
[9] “LLaMA: Open and Efficient Foundation Language Models” by Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, Guillaume Lample. arXiv:2302.13971 [cs.CL]