HAAG NLP Summarization Week 2

Michael Bock

August 2024

1 Slack Questions

What did you accomplish this week?

- Met with VIP subteams to begin API/ Data Acquisition and Summarization efforts
- Created API client and provided data to summarization teams, although I still need to concatenate all files and find somewhere it can be put
- Took Victor's OCR code and put it into the api to generate dataset
- Scraped clearinghouse.net

What are you planning on working on next?

- I've found the Civil Rights Clearinghouse's previous models, run those on the dataset I've created
- Concatenate results from clearinghouse.net scraping
- Work with summarization subteam to begin reviewing and implementing summarization models for summarizing complaints

What is blocking you from progressing?

- None

2 Abstract

Neural sequence-to-sequence models have provided a viable new approach for abstractive text summarization (meaning they are not restricted to simply selecting and rearranging passages from the origi- nal text). However, these models have two shortcomings: they are liable to reproduce factual details inaccurately, and they tend to repeat themselves. In this work we pro- pose a novel architecture that augments the standard sequence-to-sequence attentional model in two orthogonal ways. First, we use a hybrid pointer-generator network that can

copy words from the source text via pointing, which aids accurate repro- duction of information, while retaining the ability to produce novel words through the generator. Second, we use coverage to keep track of what has been summarized, which discourages repetition. We apply our model to the CNN / Daily Mail sum- marization task, outperforming the current abstractive state-of-the-art by at least 2 ROUGE points.

Link: https://arxiv.org/pdf/1704.04368

2.1 Brief Analysis

Pointer Generator networks are built on sequence to sequence models. Sequence to sequence models generate the next word of text given the previous words. One way I've learned to think of these networks is by thinking of them in 3 pieces: an encoder which maps words to numbers to form a vocabulary, a decoder which extracts features from that embedding text, and finally a classifier which produces a distribution that represents the next word. Pointer generator networks introduce copying words from a source document into the summary to the sequence to sequence framework. It does this by adding words from the source text into the distribution outputted by the final classifier in the model. It assigns a probability of generation P_{gen} by linearly combining a context vector, the decoder state, and the input and sigmoiding that output. Then the classification head's softmax scores are adjusted according to $P(w) = P_{gen}P_{vocab}+(1-P_{gen})(\Sigma w_i a_i)$. The first term is the softmax scores output by the model. The second term is the number of times a given word appears in the text. Pointer generator networks showed improved performance over other abstractive summarizers but couldn't outperform basic extractive summarizers. The authors speculate that this is a flaw in the ROUGE metric. Extractive summarizers have better ROUGE scores because they directly copy from the text and ROUGE finds how many identical n-grams there are between the summary and the source text.

3 Scripts and Code Blocks

I have uploaded the clearinghouse api client and the generator script to Law Data and Design VIP Git and to the HaaG git.

4 Documentation

See api documentation at https://github.com/Human-Augment-Analytics/NLP-Gen/tree/main/michael/api/clearinghouse_api

5 Results Visualization

https://www.youtube.com/watch?v=AKDTtSTe5XQ

6 Next Week's proposal

- Filter dataset for cases with only complaints and provide to summarization team
- Run previous summarizers on filtered dataset
- Read on PRIMERA, RAG, and BART
- Communicate with Tom on model baseline pipeline

HAAG Research Report
NLP - Sentencias / NLP - Gen Team
Week 2

Víctor C. Fernández

August 2024

1 WEEKLY PROJECT UPDATES

What progress did you make in the last week?

- Carried out the kick off call with Dr. Alexander and have now set goals for the coming weeks
- Met with the NLP Gen team on Monday, agreeing on a meeting time for the coming weeks.
- Read through papers on identifyin relationship between people and for Named Entity Recognition and Retrieval.
- Implemented logic for extracting text from pdfs and docs into usable txt files.

What progress are you making next?

- Meeting with Dr. Alexander on September 6th for an update meeting.
- Meeting with the NLP team on September 6th on our weekly meeting.
- Further develop data ingestion process to filter out headers, footers and any other irrelevant information common to all documents.
- Identify best SpaCy data for Named Entity Recognition in legal Spanish text.
- Carry out Named Entity Recognition on the outcoming text to identify the required key entities for our further investigation.
- Investigate the option of using some pretrained model for extracting key data (e.g. BERT, DBRX, etc.).

Is there anything blocking you from making progress?

No, nothing right now.

2 ABSTRACTS

1. **Title:** NERetrieve: Dataset for Next Generation Named Entity Recognition and Retrieval
   - **URL:** https://aclanthology.org/2023.findings-emnlp.218v2.pdf

   - **Abstract:** Recognizing entities in texts is a central need in many information-seeking scenarios, and indeed, Named Entity Recognition (NER) is arguably one of the most successful examples of a widely adopted NLP task and corresponding NLP technology. Recent advances in large language models (LLMs) appear to provide effective solutions (also) for NER tasks that were traditionally handled with dedicated models, often matching or surpassing the abilities of the dedicated models. Should NER be considered a solved problem? We argue to the contrary: the capabilities provided by LLMs are not the end of NER research, but rather an exciting beginning. They allow taking NER to the next level, tackling increasingly more useful, and increasingly more challenging, variants. We present three variants of the NER task, together with a dataset to support them. The first is a move towards more fine-grained—and intersectional—entity types. The second is a move towards zero-shot recognition and extraction of these fine-grained types based on entity-type labels. The third, and most challenging, is the move from the recognition setup to a novel retrieval setup, where the query is a zero-shot entity type, and the expected result is all the sentences from a large, pre-indexed corpus that contain entities of these types, and their corresponding spans. We show that all of these are far from being solved. We provide a large, silver-annotated corpus of 4 million paragraphs covering 500 entity types, to facilitate research towards all of these three goals.

   - **Summary:** The paper defines a dataset to improved Named Entity Recognition (NER). The authors argue that advancements in large language models (LLMs) present an opportunity to redefine NER, moving from simple entity recognition to more complex tasks like zero-shot retrieval of all mentions of a given entity type within a large corpus. The NERetrieve dataset, containing 4.29 million paragraphs annotated with 500 diverse entity types, aims to facilitate research in this direction. The paper also highlights the limitations of current NER models, particularly in handling fine-grained, hierarchical,

and intersectional entity types.

- **Relevance:** The NERetrieve dataset, with its focus on fine-grained and intersectional entity types, could be directly applicable to extracting key information from judges' written decisions.

2. **Title:** Characterizing Interactions and Relationships between People
   - **URL:** https://aclanthology.org/D18-1470.pdf

   - **Abstract:** This paper presents a set of dimensions to characterize the association between two people. We distinguish between interactions (when somebody refers to somebody in a conversation) and relationships (a sequence of interactions). We work with dialogue scripts from the TV show Friends, and do not impose any restrictions on the interactions and relationships. We introduce and analyze a new corpus, and present experimental results showing that the task can be automated.

   - **Summary:** The paper presents a framework for characterizing interactions and relationships between people in dialogues. It introduces a set of dimensions to describe the nature of these interactions and relationships, including aspects like cooperation, intensity, and intimacy. The authors annotate a corpus of dialogues from the TV show "Friends" with these dimensions and conduct experiments to show that these dimensions can be automatically predicted using machine learning models.

   - **Relevance:** Its focus on characterizing relationships between people could be adapted to analyze interactions between judges, lawyers, and other parties involved in the legal cases. The dimensions used in the paper, such as cooperation vs. competition and equal vs. hierarchical, could provide insights into the dynamics of legal proceedings and potentially identify factors that contribute to court congestion.

3 SCRIPTS AND CODE BLOCKS

All scripts have been uploaded to https://github.com/Human-Augment-Analytics/NLP-Gen/blob/main/victor.

The following functions are the core parts of the script in the provided folder that is intended for extracting the text from the sentencias pdfs and new docs.

```python
def extract_text_from_pdf(pdf_path, output_location):
    """
    Extracts text from a PDF file and saves it to a .txt file.

    Args:
        pdf_path: The path to the PDF file.
        output_location: The directory where the output .txt file should be saved.
    """

    doc = fitz.open(pdf_path)
    text = ""

    for page_num in range(doc.page_count):
        page = doc[page_num]
        text += page.get_text()

    # Close the document
    doc.close()

    # Create output file name based on the PDF file name
    output_file_name = os.path.splitext(os.path.basename(pdf_path))[0] + ".txt"
    output_file_path = os.path.join(output_location, output_file_name)

    # Write cleaned text to the output file
    with open(output_file_path, 'w', encoding='utf-8') as txt_file:
        txt_file.write(text)
```

Listing 1—Extract text from pdfs

```python
def extract_text_from_doc(doc_path, output_location):
    """
    Extracts text from a DOC file and saves it to a .txt file.

    Args:
        doc_path: The path to the DOC file.
        output_location: The directory where the output .txt file should be saved.
    """
    # If it's a .doc file, convert it to .docx first
    if doc_path.endswith('.doc'):
        doc_path = convert_doc_to_docx(doc_path)

    doc = docx.Document(doc_path)
```

```
    text = ""
    for paragraph in doc.paragraphs:
        text += paragraph.text + "\n"

    # Create output file name based on the DOC file name
    output_file_name = os.path.splitext(os.path.basename(doc_path))[0] + ".txt"
    output_file_path = os.path.join(output_location, output_file_name)

    with open(output_file_path, 'w', encoding='utf-8') as txt_file:
        txt_file.write(text)
```

Listing 2—Extract Text from doc or docx

```
def process_file_or_folder(single_file=None, folder=None, output_location="output_texts"):
    """
    Processes a single file or all files in a folder, extracting text and saving it to .txt files.

    Args:
        single_file: The path to a single file (optional).
        folder: The path to a folder containing files (optional).
        output_location: The directory where the output .txt files should be saved (default: "output_texts").
    """

    if single_file:
        file_extension = check_file_extension(single_file)
        if file_extension == 'pdf':
            extract_text_from_pdf(single_file, output_location)
            print(f"Text extracted from {single_file} and saved to {output_location}")
        elif file_extension == 'doc':
            extract_text_from_doc(single_file, output_location)
            print(f"Text extracted from {single_file} and saved to {output_location}")
        else:
            print(f"Unsupported file type: {single_file}")

    elif folder:
        if not os.path.exists(output_location):
            os.makedirs(output_location)

        for file_name in os.listdir(folder):
            file_path = os.path.join(folder, file_name)
            if os.path.isfile(file_path):
                file_extension = check_file_extension(file_path)
```

```
                if file_extension == 'pdf':
                    extract_text_from_pdf(file_path, output_location)
                    print(f"Text extracted from {file_name} and saved to {
    output_location}")
                elif file_extension == 'doc':
                    extract_text_from_doc(file_path, output_location)
                    print(f"Text extracted from {file_name} and saved to {
    output_location}")
```

Listing 3—Function for either individually loading or bulk loading documents in pdf or doc/docx format

4 DOCUMENTATION

1. **Data Collection and Preprocessing:**
   - A set of judicial decisions (sentencias) in pdf and doc format was obtained from Dr. Alexander, originating from the National School of the Judiciary in the Dominican Republic.
   - Text was extracted from PDF and doc files using the PyMuPDF library instead of previous pypdf library.
   - The extracted text was stored in a new folder and shared with the team for future usage and avoid requiring the need to process the pdf from the start again.
   - Text was extracted in the most similar format as possible to the original document. Ensuring as much similarity as possible and reducing the noise in the txt file.
   - Additionally, these new documents will then be preprocessed to remove some preexisting noise such as headers and footers in the documents, to ensure the proceeding steps are based on the core content of the sentencias.
2. **Text Analysis and Feature Extraction:**
   - Once previous part has been fully achieved for all documents, will proceed onto the text analysis and features extraction.

5 SCRIPT VALIDATION(OPTIONAL)

Script was validated by picking 2 samples from the documents and verifying that at least 90% of the text was in the same format and structure as the original document. This validation led to switch from using pypdf library to using PyMuPDF, since the first was splitting words in multiple locations of the corpus

into separate words, increasing the complexity of the following steps.

Additionally, as security measure, a posterior data processing step may be carried out based on nltk library to correct any remaining issue after the first extraction.

6 RESULTS VISUALIZATION

Resulting documents were shared in basecamp with the team in order for everyone to have access and be able to use them in any future steps. Folder link is the following (may only be accessed by members with authorization): https://3.basecamp.com/5835116/buckets/38747799/vaults/7763406910

7 PROOF OF WORK

The results obtained were reliable and stable for a first data extraction. It is currently prepared for pdf and docx documents, being doc files converted to the latest docx format before being extracted. There were minor details that could cause issues in future processing steps such as words being split into multiple words unnecessarily, but these were solved by using a more robust data extraction library for pdf documents, PyMuPdf. After multiple iterations, text was extracted in a stable manner, allowing it to serve as reference text for further processing.

In general terms, a simple text extraction mechanism was created that can now allow for further development towards extracting relevant entities from the documents.

Will keep moving forward during the coming week in order to have extracted more relevant data from the documents, as pointed out by Dr. Alexander during our last meeting.

Week 2 | HAAG - NLP | Fall 2024

Alejandro Gomez

August 30, 2024

1 Time-log

1.1 What progress did you make in the last week?

- Had a kickoff meeting with Dr. Charlotte Alexander where we discussed the project objective, team roster, and the current (plus incoming) available data (i.e. PDF) for the NLP sentencias subgroup.
- Met with the larger NLP group to discuss current progress, upcoming implementations, and suggested libraries
- In the prior week I tried to familiarize myself with a Python environment, which consisted of setting up conda, jupyter notebooks, and common libraries such as numpy,pandas, and matplotlib, so this week I tried to actually get a small program running with the spaCy library for NLP. I was able to iterate through the sentencias documents and retrieve the named entities. They don't align fully with the project requirements so the project script will require further refinement. Nevertheless, I've familiarized myself more with Python in scientific applications, setting up an environment with new dependencies, and the spaCy library for NLP.

1.2 What are you planning on working on next?

Dr. Alexander shared some named entities that we should be targeting for our extraction and summarizations of the Dominican Republic Judicial documents, so I'll work on trying to extract those entities from the available dataset while leveraging the spacy library.

1.3 Is anything blocking you from getting work done?

No current blockers.

2 Abstract

The effort of providing a brief and fluent summary of a business meeting while preserving vital information content and overall meaning is known as business meeting summarizing. Initially, we convert recorded meetings, interviews, speeches, lectures, and other audio streams into text documents in the proposed work. After that, speaker text is divided into segments. Minutes of the Meeting are generated automatically using the proposed work.We also focused on using the Rev-AI Speech-to-Text API to better the voice to text conversion of a specific recorded audio file while making sure that the summarised text that has been converted from speech to text provides a specific and accurate meaning that is not only understandable to everyone but also covers all of the recorded file's important points doi link[AJ22]

3 Scripts and Code Blocks

3.1 Code

```python
import os
import spacy
import pandas as pd

nlp = spacy.load("es_core_news_sm")

sentencias_entities = {}

for filename in os.listdir(data_files_path):

   with open(os.path.join(data_files_path, filename), 'r') as f:
      sentencias_doc = f.read()

   doc = nlp(sentencias_doc)

   # Find named entities, phrases and concepts
   for entity in doc.ents:
      if entity.label_ not in sentencias_entities:
         sentencias_entities[entity.label_] = [entity.text]
      else:
         sentencias_entities[entity.label_].append(entity.text)

pd.DataFrame([sentencias_entities])
```

Listing 1: python code

3.2 Documentation

In this script, I iterated through the directory containing the sentencias files and analyzied each file. The method was Named Entity Recognition (NER) which is an NLP method that can categorize the tokens gathered from each analyzed document. After meeting with Dr. Alexander, our team was instructed to find the entities Initial experiment is to extract/identify the following:

- Names of the parties (individuals, companies) : PERSON, ORG
- Amounts of money : MONEY
- Dates (and ideally the event that happened on that date) : DATE, TIME

These are the entities I identified which would align most with the assignment's criteria; however, they are not the default entities that the program found. This will require further work next week to reach closer alignment. Scripts in GitHub Repo

3.3 Results Visualization

	ORG	MISC	LOC	PER
0	[REPÚBLICA, PODER, REPÚBLICA, Unidad de Primer...	[JUDICIAL, Sentencia laboral núm. 0374-2024-SS...	[JUDICIAL, SANTIAGO \n \nSentencia, República ...	[Benigno Filomeno de Rojas, Santiago Rodríguez...

Figure 1: dataframe

3.4 Proof of Work

This was an initial prototype for working with the spaCy NLP library.

4 Next Week's Proposal

- Refine the script to search for the specific entities as directed by Dr. Alexander
- Update the NLP website with contact info and current work
- Update powerpoint slide to share with my team and meet during our scheduled time to compare our current approaches and future implementations on how to retrieve results that meet the current project objective

References

[AJ22] Preetam Hegde Navin Singhaniya Aryan Jha, Sameer Temkar. Business meeting summary generation using nlp. *ITM Web of Conferences*, 44(1):3063, 2022.

HAAG NLP Sentencias — Week 2 Report
NLP-Gen Team

Karol Gutierrez

August 30, 2024

1 Weekly Project Update

1.1 What progress did you make in the last week?

- Call with Dr. Alexander and team on August 26. It was agreed that for the next two weeks our focus should be on extracting dates and key elements/actors from the sentencias files.
- Weekly 1 hour meeting with team to discuss our progress, show results and further work.
- Creation of OneNote to store all the documentation and notes related to our team, setting it as the shared source of knowledge for the work we will be doing. This was communicated with the team on Slack.
- Creation PowerPoint slides with our weekly updates, as well as a shared folder where we will store all the presentations. This was also shared with the team.
- Uploading meeting recording to YouTube as unlisted videos and shared links with Webmaster.
- Literature review for feature extraction in pdf files.
- Fulfill my role as Meet Manager/Documentor by working on the tasks expected for my position.
- Start coding work regarding dates and key elements from the pdf files.

1.2 What are you planning on working on next?

- Complete code to extract dates and key elements from pdf texts, as this was the milestone for agreed with Dr. Alexander during these two weeks.
- Call with Dr. Alexander and team on September 6 to show results.
- Further literature reviews for NLP models that can be applied to our project.
- Continue fulfilling my role as Meet Manager/Documentor by working on the tasks expected for my position.

1.3 Is anything blocking you from getting work done?

No.

2 Literature Review

Paper: Layout-aware text extraction from full-text PDF of scientific articles [RPH+12].

2.1 Abstract

Background: The Portable Document Format (PDF) is the most commonly used file format for online scientific publications. The absence of effective means to extract text from these PDF files in a layout-aware manner presents a significant challenge for developers of biomedical text mining or biocuration informatics systems that use published literature as an information source. In this paper we introduce the 'Layout-Aware PDF Text Extraction' (LA-PDFText) system to facilitate accurate extraction of text from PDF files of research articles for use in text mining applications.

Results: Our paper describes the construction and performance of an open source system that extracts text blocks from PDF-formatted full-text research articles and classifies them into logical units based on rules that characterize specific sections. The LA-PDFText system focuses only on the textual content of the research articles and is meant as a baseline for further experiments into more advanced extraction methods that handle multi-modal content, such as images and graphs. The system works in a three-stage process: (1) Detecting contiguous text blocks using spatial layout processing to locate and identify blocks of contiguous text, (2) Classifying text blocks into rhetorical categories using a rule-based method and (3) Stitching classified text blocks together in the correct order resulting in the extraction of text from section-wise grouped blocks. We show that our system can identify text blocks and classify them into rhetorical categories with Precision1 = 0.96present an evaluation of the accuracy of the block detection algorithm used in step 2. Additionally, we have compared the accuracy of the text extracted by LA-PDFText to the text from the Open Access subset of PubMed Central. We then compared this accuracy with that of the text extracted by the PDF2Text system, 2 commonly used to extract text from PDF. Finally, we discuss preliminary error analysis for our system and identify further areas of improvement.

Conclusions: LA-PDFText is an open-source tool for accurately extracting text from full-text scientific articles. The release of the system is available at http://code.google.com/p/lapdftext/.

2.2 Summary

The paper explains the development of LA-PDFText, a tool for extracting text from scientific PDF articles. The authors created this tool to solve problems caused by the complex formats of scientific PDFs, which make it hard to extract text correctly. LA-PDFText works in three steps: first, it detects blocks of text using layout analysis, then it classifies these blocks into logical units like sections and headings, and finally, it stitches them together to keep the narrative of the article. The system was tested using precision, recall, and F1 scores, showing superiority over existing tools like PDF2Text in keeping the structure and clarity of the text. The study shows that LA-PDFText can be very useful for tasks like text mining and organizing scientific information.

2.3 Relevance

This is relevant to our project because the proposed method is a solution for accurately extracting and structuring text from complex PDFs, which aligns with our goal of extracting key information from judges' written decisions (sentencias) and structuring court data. In particular, using a rule-based approach is something that can be applied to our case. By leveraging these techniques, we can improve the accuracy and reliability of the text extraction process in legal documents.

3 Scripts and code blocks

All the existing code is in the following repository.

I added two scripts, one that uses the existing library PyPDF2 to convert the pdf content into txt files. The second script uses the txt files to retrieve dates from the texts using regular expressions shown in the code section in Figure 1. At the end, I plot the occurrences of dates over time across all the documents.

The current code logic is explained in Figure 2.

```
6
7  # Regexs for dates in Spanish
8  date_patterns = [
9      r"\b\d{1,2} de (enero|febrero|marzo|abril|mayo|junio|julio|agosto|septiembre|octubre|noviembre|diciembre) de \d{4}\b",
10     r"\b\d{1,2}-\d{1,2}-\d{4}\b",
11     r"\b(\d+|[a-z]+) \(\d{1,2}\) días del mes de (enero|febrero|marzo|abril|mayo|junio|julio|agosto|septiembre|octubre|noviembre|diciembre) del año (dos mil \w+|\d{4})\b",
12 ]
13
```

Figure 1: Code block.

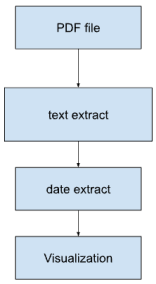

Figure 2: Code logic flow chart.

4 Documentation

The documentation is present in the README.md file in the repository. Refer to the repository to get the most updated instructions on how to run the code.

An addition to this week is adding a .gitignore file with a rule to prevent pushing the .pdf and .doc documents into the public repository, since these documents contain sensitive and personal information.

4.1 README.md

```
# nlp-dr
...
...
...

### Run the code

Use ‘python extract_dates_example.py‘.
```

5 Script Validation

It doesn't apply at this point of the development of the project.

6 Results Visualization

Using our case-based approach we can extract relevant dates from the cases and plot them over time, as shown in Figure 3.

7 Proof of Work

At this time the code runs but it is still a prototype. It can successfully retrieve some of the dates from the files but more analysis is required to test accuracy.

8 Next Week's Proposal

Refer to section 1.2 for details (avoid repetition).

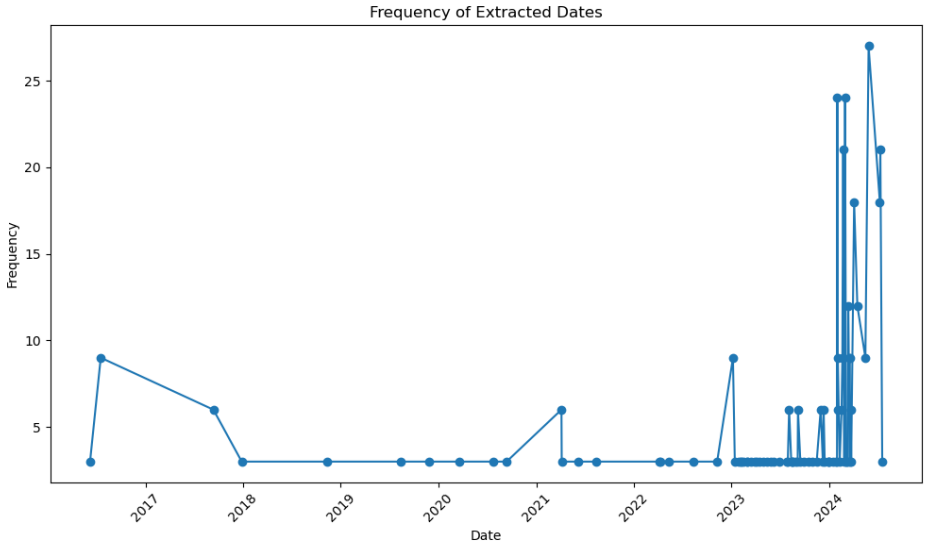

Figure 3: Dates occurrences over time.

References

[RPH+12] C. Ramakrishnan, A. Patnia, E. Hovy, et al. Layout-aware text extraction from full-text pdf of scientific articles. *Source Code for Biology and Medicine*, 7, 2012.

Week 2 Research Report

Thomas Orth (NLP Summarization / NLP Gen Team)

August 2024

1 Weekly Project Updates

1.1 What progress did you make in the last week?

1. Met with VIP sub teams to kickstart summarization project
2. Performed Literature review of summarization methods
3. Performed review of tools for usage on this project

1.2 What are you planning on working on next?

1. Meet with VIP team to provide next steps based on Literature Review.
2. Begin creating model baseline pipeline with small dataset from the clearinghouse in anticipation of more documents.
3. Support OCRing as needed.
4. Begin looking into PACE cluster.

1.3 Is anything blocking you from getting work done?

1. None

2 Abstract Review

- Name: Legal Case Document Summarization: Extractive and Abstractive Methods and their Evaluation. Conference: Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)
- Link: https://aclanthology.org/2022.aacl-main.77/

- Abstract: Summarization of legal case judgement documents is a challenging problem in Legal NLP. However, not much analyses exist on how different families of summarization models (e.g., extractive vs. abstractive) perform when applied to legal case documents. This question is particularly important since many recent transformer-based abstractive summarization models have restrictions on the number of input tokens, and legal documents are known to be very long. Also, it is an open question on how best to evaluate legal case document summarization systems. In this paper, we carry out extensive experiments with several extractive and abstractive summarization methods (both supervised and unsupervised) over three legal summarization datasets that we have developed. Our analyses, that includes evaluation by law practitioners, lead to several interesting insights on legal summarization in specific and long document summarization in general.

- Summary: This paper explores how different types of summarization models perform when summarizing legal case documents, which are typically long. The study includes experiments with various extractive and abstractive summarization methods and evaluates their effectiveness with input from law practitioners.

- Relevancy: This gives a comprehensive review of different methods used for legal summarization.

3 Scripts

No code due to focus on literature and tools review.

4 Documentation

No code due to focus on literature and tools review

5 Results

The summarization of my review is below

1. Transformers
   (a) T5
      - Paper: https://arxiv.org/pdf/1910.10683
      - High Level Overview from here: T5, or Text-to-Text Transfer Transformer, is a Transformer based architecture that uses a text-to-text approach. Every task – including translation, question answering, and classification – is cast as feeding the model text as input and training it to generate some target text. This

allows for the use of the same model, loss function, hyperparameters, etc. across our diverse set of tasks. The changes compared to BERT include: adding a causal decoder to the bidirectional architecture, replacing the fill-in-the-blank cloze task with a mix of alternative pre-training tasks.

- Implementation: Provided by Huggingface. Docs linked here.
- Paper(s) using model / referencing the model for legal summarization:
  - `https://arxiv.org/html/2404.00594v1`
  - `https://www.sciencedirect.com/science/article/abs/pii/S0957417423020730`
- Shortcomings: By default, not good for long documents. Need to use an extension of the model such as `https://huggingface.co/docs/transformers/en/model_doc/longt5`.

(b) Pegasus

- Paper: https://arxiv.org/pdf/1912.08777
- High Level Overview from here: PEGASUS proposes a transformer-based model for abstractive summarization. It uses a special self-supervised pre-training objective called gap-sentences generation (GSG) that's designed to perform well on summarization-related downstream tasks. As reported in the paper, "both GSG and MLM are applied simultaneously to this example as pre-training objectives. Originally there are three sentences. One sentence is masked with [MASK1] and used as target generation text (GSG). The other two sentences remain in the input, but some tokens are randomly masked by [MASK2]."
- Implementation: Provided by Huggingface. Docs linked here.
- Paper(s) using model / referencing the model for legal summarization:
  - `https://link.springer.com/chapter/10.1007/978-981-99-0085-5_46`
  - `https://arxiv.org/pdf/2206.10883`
  - `https://aclanthology.org/2022.aacl-main.77/`
- Shortcomings: A good number of papers show that pegasus is not good for legal document summarization but at least one asserts it works well. So not throwing it out of the baseline but probably not a promising method.

(c) BART

- Paper: https://arxiv.org/abs/1910.13461
- High Level Overview from here: BART is a denoising autoencoder for pretraining sequence-to-sequence models. It is trained by (1) corrupting text with an arbitrary noising function, and (2)

learning a model to reconstruct the original text. It uses a standard Transformer-based neural machine translation architecture. It uses a standard seq2seq/NMT architecture with a bidirectional encoder (like BERT) and a left-to-right decoder (like GPT). This means the encoder's attention mask is fully visible, like BERT, and the decoder's attention mask is causal, like GPT2.

- Implementation: Provided by Huggingface. Docs linked here.
- Paper(s) using model / referencing the model for legal summarization:
  - `https://arxiv.org/pdf/2206.10883`
  - `https://aclanthology.org/2022.aacl-main.77/`
- Shortcomings: Not known yet due to mix of papers having BART as the best model vs others who don't.

(d) Longformer (LED)

- Paper: https://arxiv.org/abs/2004.05150
- High Level Overview from here: Longformer is a modified Transformer architecture. Traditional Transformer-based models are unable to process long sequences due to their self-attention operation, which scales quadratically with the sequence length. To address this, Longformer uses an attention pattern that scales linearly with sequence length, making it easy to process documents of thousands of tokens or longer. The attention mechanism is a drop-in replacement for the standard self-attention and combines a local windowed attention with a task motivated global attention.
- Implementation: Provided by Huggingface. Docs linked here.
- Paper(s) using model / referencing the model for legal summarization:
  - `https://arxiv.org/pdf/2206.10883`
  - `https://aclanthology.org/2022.aacl-main.77/`
  - `https://arxiv.org/html/2404.00594v1`
- Shortcomings: During an initial test, the largest variation provided by huggingface hallucinated details during an initial test for zero shot learning. However, this could be due to the pretraining data that was used by the author.

(e) Primera

- Paper: https://arxiv.org/abs/2110.08499
- Abstract: We introduce PRIMERA, a pre-trained model for multi-document representation with a focus on summarization that reduces the need for dataset-specific architectures and large amounts of fine-tuning labeled data. PRIMERA uses our newly proposed pre-training objective designed to teach the model to connect and aggregate information across documents. It also

uses efficient encoder-decoder transformers to simplify the processing of concatenated input documents. With extensive experiments on 6 multi-document summarization datasets from 3 different domains on zero-shot, few-shot and full-supervised settings, PRIMERA outperforms current state-of-the-art dataset-specific and pre-trained models on most of these settings with large margins.

- Implementation: Provided by Huggingface. Docs linked here.
- Paper(s) using model / referencing the model for legal summarization:
  - `https://arxiv.org/pdf/2206.10883`
  - `https://arxiv.org/html/2404.00594v1`
- Shortcomings: Not known until done testing

2. LLMs

   (a) Mixtral
      i. Paper: https://arxiv.org/abs/2401.04088
      ii. Summary: LLM using the same architecture as Mistral 7b but each layer has 8 feedforward blocks
      iii. OSS or closed source: OSS
      iv. Availability: Ollama and Huggingface

   (b) Llama 3
      i. Paper:https://arxiv.org/pdf/2407.21783
      ii. Summary: Llama 3 is a set of foundational models, trained on huge amounts of data to achieve SOTA performance across tasks. Architecture isn't described in paper but model checkpoints released.
      iii. OSS or closed source: OSS
      iv. Availability: Ollama and Huggingface

   (c) Phi 3
      i. Paper: https://arxiv.org/abs/2404.14219
      ii. Summary: Phi-3 is an attempt to take an LLM and make it smaller for less computation being used.
      iii. OSS or closed source: OSS
      iv. Availability: Ollama and Huggingface

   (d) GPT-4
      i. Paper: https://arxiv.org/abs/2303.08774
      ii. Summary: OpenAI's proprietary model that is one of the best performing LLMs out there currently. Architecture details unknown.
      iii. OSS or closed source: Closed

iv. Availability: API call using OpenAI package

(e) Claude 3.5 Sonnet

i. Paper: `https://www-cdn.anthropic.com/fed9cc193a14b84131812372d8d5857f8f304c52/Model_Card_Claude_3_Addendum.pdf`

ii. Summary: Anthropic's latest model released that has similar performance to GPT-4

iii. OSS or closed source: Closed

iv. Availability: API call using anthropic's software.

3. Tools

(a) Langchain: Tool used for LLM workflows that will allow for summarization techniques like map-reduce.

(b) Huggingface: Library providing model checkpoints, training setup and inference tools for AI work

(c) DSPY: Library for use with LLMs to do programming of prompts instead of prompt engineering

(d) Ollama: Tool that serves out LLMs for testing locally through quantized models.