HAAG NLP Summarization Week 3

Michael Bock

September 2024

1 Slack Questions

What did you accomplish this week?

- Wrote documentation Clearinghouse API and created code samples for undergraduate students
- Did an experiment with Pegasus on generate very short summaries. I did an extractive-abstractive based approach where I first parsed the complaint bullet points from the document and then fed only those into Pegasus. I was able to get one sentence summaries that, although not very detailed, were factually correct. But as summaries got longer, it started repeating the same sentence

What are you planning on working on next?

- I've read about RAG, I want to try getting myself a simple seq2seq baseline to play with
- I need to send out a list of papers to the undergrads, I have about 30 papers so each undergrad will be able to review 2 or 3 of them
- Read seq2seq and Mistral papers as required by my subteam

What is blocking you from progressing?

- None

2 Abstract

Large pre-trained language models have been shown to store factual knowledge in their parameters, and achieve state-of-the-art results when fine-tuned on down- stream NLP tasks. However, their ability to access and precisely manipulate knowl- edge is still limited, and hence on knowledge-intensive tasks, their performance lags behind task-specific architectures. Additionally, providing provenance for their decisions and updating their world knowledge remain open research problems. Pre- trained models with a differentiable access mechanism to explicit non-parametric memory have so far been only investigated for extractive downstream tasks. We explore a general-purpose fine-tuning recipe for retrieval-augmented generation (RAG) — models which combine pre-trained parametric and non-parametric mem- ory for language generation. We introduce RAG models

where the parametric memory is a pre-trained seq2seq model and the non-parametric memory is a dense vector index of Wikipedia, accessed with a pre-trained neural retriever. We com- pare two RAG formulations, one which conditions on the same retrieved passages across the whole generated sequence, and another which can use different passages per token. We fine-tune and evaluate our models on a wide range of knowledge- intensive NLP tasks and set the state of the art on three open domain QA tasks, outperforming parametric seq2seq models and task-specific retrieve-and-extract architectures. For language generation tasks, we find that RAG models generate more specific, diverse and factual language than a state-of-the-art parametric-only seq2seq baseline.

Link: https://arxiv.org/pdf/2005.11401v4

2.1 Brief Analysis

Retrieval Augmented Generation(RAG) is a technique that draws a dividing line between two types of knowledge that a generative language model has. Parameteric knowledge is encoded into the parameters of a neural network during training and helps to generate semantically and gramatically correct text. Models that rely purely on parametric knowledge hallucinate and generate text that is factually incorrect. RAG builds on memory based knowledge called non-parametric knowledge. Non-parametric knowledge is stored outside the network in a database. RAG will encode a query text(like a complaint text) and then that encoding is used to find the top few documents that relate to that text. Then, the documents are appended to the initial query and this is fed into a sequence to sequence model. To me, this is somewhat reminiscent of region proposal networks increasing classification accuracy for object detectors. It also reminds be of extractive-abstractive approaches like the one I experimented with in pegasus. One thing I do not fully understand with RAG is that it claims that you don't have to retrain if you change documents, but I think this may only be the case with models that are equipped to run RAG. If I had a normal pretrained sequence to sequence model, wouldn't I need to add more inputs to accomadate the extra tokens added by RAG? Unless we decode the retrieved documents and add them to a prompt, I think you'd need to modify the network architecture. So far, RAG is the only NLP tool I've seen that claims to have higher scores on benchmarks like ROUGE than an extractive approach.

3 Scripts and Code Blocks

I've edited the clearinghouse_api folder to add markdown documentation and code examples of how to use the API. The file ch_openapi.yaml holds the spec used to generate the api. The file is pretty long(1200 lines). The general flow of the API to generate a dataset suitable for finetuning summarization models is shown in Figure 1

3.1 Pegasus

This was a largely unsucessful experiment where I tried to run lead3 and then pegasus. It was able to get one sentence summaries but broke after one sentence most of the time. I haven't collected any metrics on this yet because our validation team has not yet decided on an evaluation pipeline. The code is shown in Figure 2 and Figure 3.

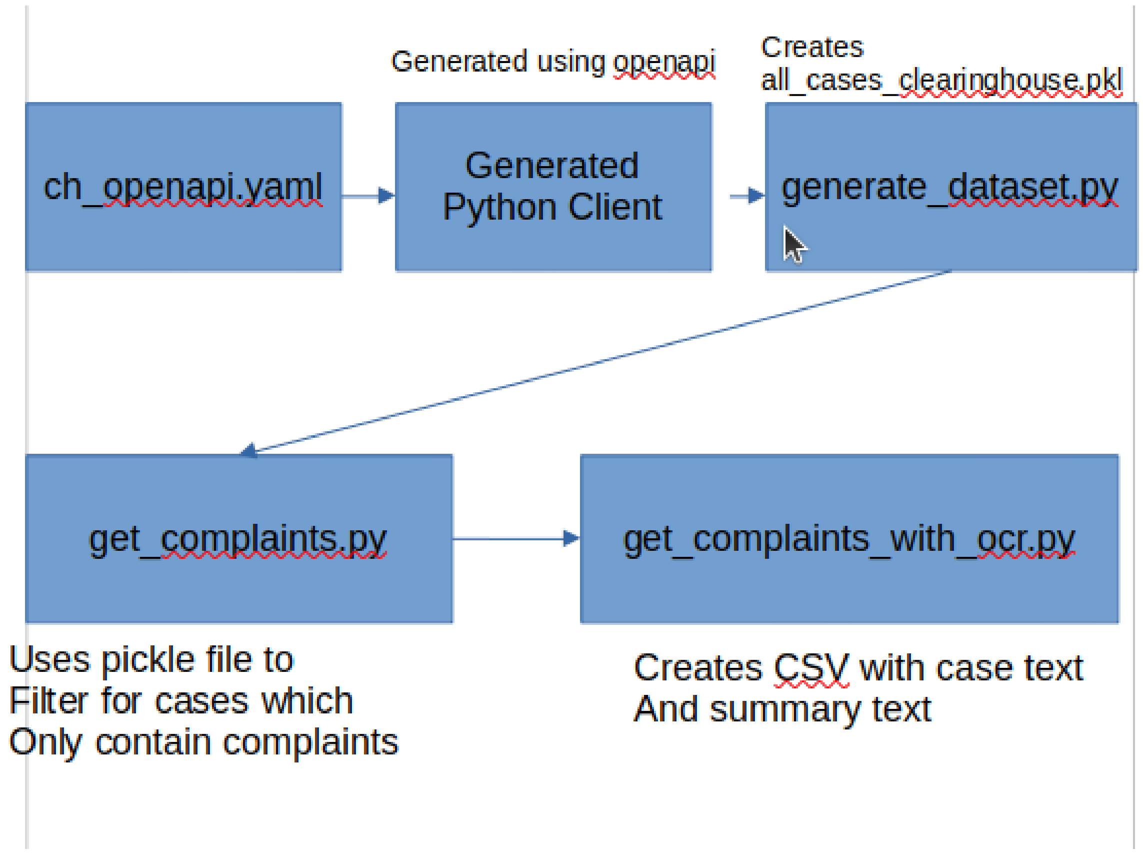
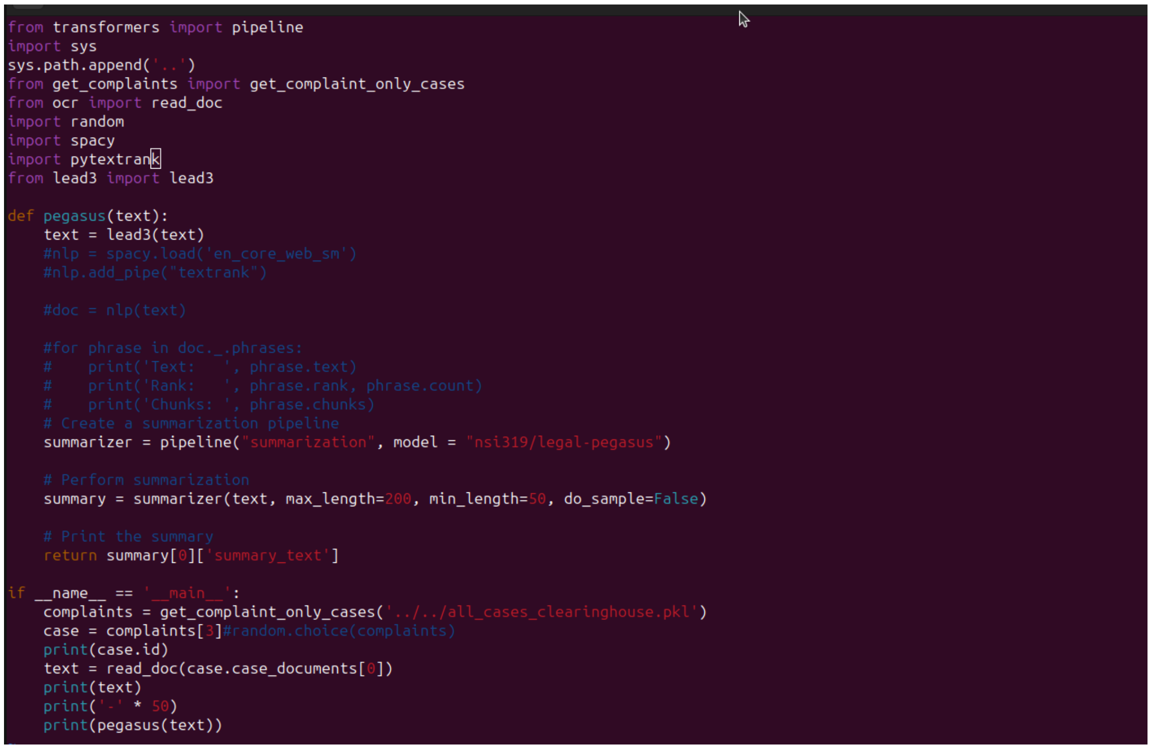
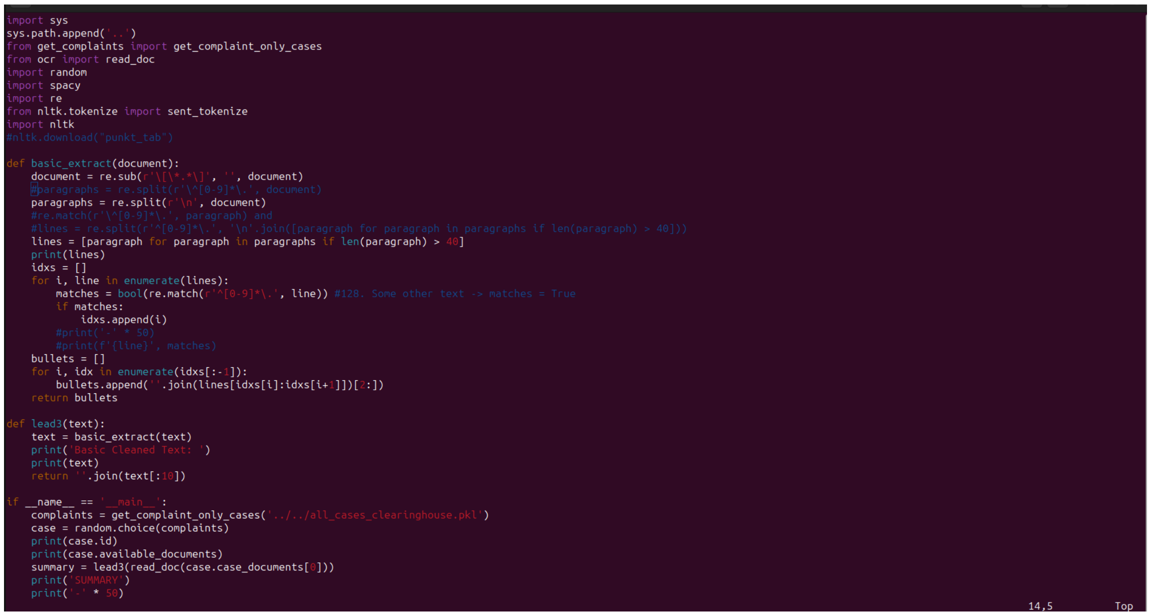

Generated using openapi
Creates all_cases_clearinghouse.pkl
ch_openapi.yaml
Generated Python Client
generate_dataset.py
get_complaints.py
get_complaints_with_ocr.py
Uses pickle file to Filter for cases which Only contain complaints
Creates CSV with case text And summary text

Figure 1: API To Summary Model Flow



```python
from transformers import pipeline
import sys
sys.path.append('..')
from get_complaints import get_complaint_only_cases
from ocr import read_doc
import random
import spacy
import pytextrank
from lead3 import lead3

def pegasus(text):
    text = lead3(text)
    #nlp = spacy.load('en_core_web_sm')
    #nlp.add_pipe("textrank")

    #doc = nlp(text)

    #for phrase in doc._.phrases:
    #    print('Text:    ', phrase.text)
    #    print('Rank:    ', phrase.rank, phrase.count)
    #    print('Chunks: ', phrase.chunks)
    # Create a summarization pipeline
    summarizer = pipeline("summarization", model = "nsi319/legal-pegasus")

    # Perform summarization
    summary = summarizer(text, max_length=200, min_length=50, do_sample=False)

    # Print the summary
    return summary[0]['summary_text']

if __name__ == '__main__':
    complaints = get_complaint_only_cases('../../all_cases_clearinghouse.pkl')
    case = complaints[3]#random.choice(complaints)
    print(case.id)
    text = read_doc(case.case_documents[0])
    print(text)
    print('-' * 50)
    print(pegasus(text))
```

Figure 2: Pegasus Extractive Abstractive Summarization



```python
import sys
sys.path.append('..')
from get_complaints import get_complaint_only_cases
from ocr import read_doc
import random
import spacy
import re
from nltk.tokenize import sent_tokenize
import nltk
#nltk.download("punkt_tab")

def basic_extract(document):
    document = re.sub(r'\[\*.*\]', '', document)
    #paragraphs = re.split(r'\^[0-9]*\.', document)
    paragraphs = re.split(r'\n', document)
    #re.match(r'\^[0-9]*\.', paragraph) and
    #lines = re.split(r'^[0-9]*\.', '\n'.join([paragraph for paragraph in paragraphs if len(paragraph) > 40]))
    lines = [paragraph for paragraph in paragraphs if len(paragraph) > 40]
    print(lines)
    idxs = []
    for i, line in enumerate(lines):
        matches = bool(re.match(r'^[0-9]*\.', line)) #128. Some other text -> matches = True
        if matches:
            idxs.append(i)
        #print('-' * 50)
        #print(f'{line}', matches)
    bullets = []
    for i, idx in enumerate(idxs[:-1]):
        bullets.append(''.join(lines[idxs[i]:idxs[i+1]])[2:])
    return bullets

def lead3(text):
    text = basic_extract(text)
    print('Basic Cleaned Text: ')
    print(text)
    return ''.join(text[:10])

if __name__ == '__main__':
    complaints = get_complaint_only_cases('../../all_cases_clearinghouse.pkl')
    case = random.choice(complaints)
    print(case.id)
    print(case.available_documents)
    summary = lead3(read_doc(case.case_documents[0]))
    print('SUMMARY')
    print('-' * 50)
                                                                                                              14,5          Top
```

Figure 3: Basic Lead3 extractive summarization

4 Documentation

See api documentation at https://github.com/Human-Augment-Analytics/NLP-Gen/tree/main/michael/api/clearinghou
Also see documentation in https://github.com/Human-Augment-Analytics/NLP-Gen/tree/main/michael/api that shows how to open the pickle file. To generate the api, do:

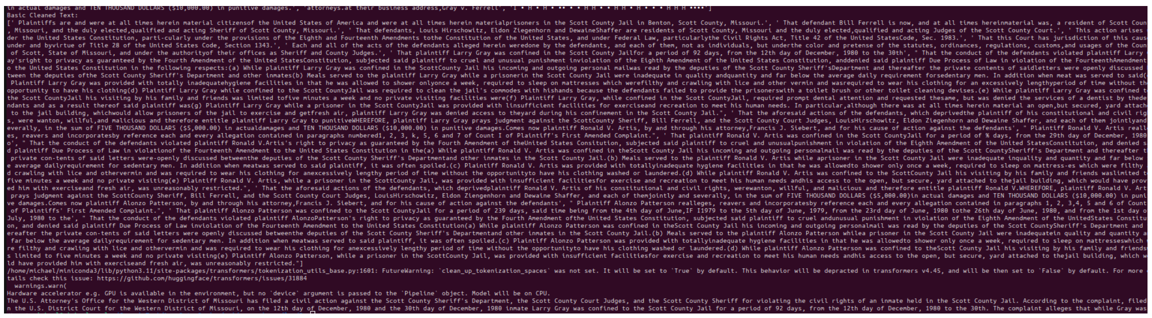
```
openapi-generator-cli generate -i ch\_openapi.yaml -g python -o clearinghouse\_api
```

5 Results Visualization

Figure 4 shows a case that has been summarized with pegasus

6 Next Week's proposal

- Research retrieval augmented generation, NER
- Research decoder-only transformer seq2seq model(Mistral), Create a test decoder only model with less than 7B parameters? That I think can help students understand creating seq2seq models and can probably assist with the NER tasking
- Create seq2seq script and see if I can add metadata from cases into the model as a form of RAG
- Send list of papers to undergraduates for review



```
in actual damages and TEN THOUSAND DOLLARS ($10,000.00) in punitive damages.', 'attorneys.at their business address,Gray v. Ferrell', 'I • H • H • •• • • H H • • H • •• • • • H H • • H H • H • • • H H H ••••']
Basic Cleaned Text:
[' Plaintiffs are and were at all times herein material citizensof the United States of America and were at all times herein materialprisoners in the Scott County Jail in Benton, Scott County, Missouri.', ' That defendant Bill Ferrell is now, and at all times hereinmaterial was, a resident of Scott Coun
, Missouri, and the duly elected,qualified and acting Sheriff of Scott County, Missouri.', ' That defendants, Louis Hirschowitz, Eldon Ziegenhorn and DewaineShaffer are residents of Scott County, Missouri and the duly elected,qualified and acting Judges of the Scott County Court.', ' This action arises
der the United States Constitution, parti-cularly under the provisions of the Eighth and Fourteenth Amendments tothe Constitution of the United States, and under Federal Law, particularlythe Civil Rights Act, Title 42 of the United StatesCode, Sec. 1983.', ' That this Court has jurisdiction of this caus
under and byvirtue of Title 28 of the United States Code, Section 1343.', ' Each and all of the acts of the defendants alleged herein weredone by the defendants, and each of them, not as individuals, but underthe color and pretense of the statutes, ordinances, regulations, customs,and usages of the Coun
 of Scott, State of Missouri, and under the authorityof their offices as Sheriff and County Judges.', ' That plaintiff Larry Gray was confined in the Scott County Jailfor a period of 92 days, from the 12th day of December, 1980 to the 30th', " That the conduct of the defendants violated plaintiff Larry
ay'sright to privacy as guaranteed by the Fourth Amendment of the United StatesConstitution, subjected said plaintiff to cruel and unusual punishment inviolation of the Eighth Amendment of the United States Constitution, anddenied said plaintiff Due Process of Law in violation of the FourteenthAmendment
o the United States Constitution in the following respects:(a) While plaintiff Larry Gray was confined in the ScottCounty Jail his incoming and outgoing personal mailwas read by the deputies of the Scott County Sheriff'sDepartment and thereafter the private contents of saidletters were openly discussed
tween the deputies ofthe Scott County Sheriff's Department and other inmates(b) Meals served to the plaintiff Larry Gray while a prisonerin the Scott County Jail were inadequate in quality andquantity and far below the average daily requirement forsedentary men. In addition when meat was served to said(
 Plaintiff Larry Gray was provided with totally inadequatehygiene facilities in that he was allowed to shower onlyonce a week, required to sleep on mattresses which werefilthy and crawling with lice and other vermin and wasrequired to wear his clothing for an excessively lengthyperiod of time without the
opportunity to have his clothing(d) Plaintiff Larry Gray while confined to the Scott CountyJail was required to clean the jail's commodes with hishands because the defendants failed to provide the prisonerswith a toilet brush or other toilet cleaning devises.(e) While plaintiff Larry Gray was confined t
the Scott CountyJail his visiting by his family and friends was limited tofive minutes a week and no private visiting facilities were(f) Plaintiff Larry Gray, while confined in the Scott CountyJail, required prompt dental attention and requested thesame, but was denied the services of a dentist by thede
ndants and as a result thereof said plaintiff was(g) Plaintiff Larry Gray while a prisoner in the Scott CountyJail was provided with linsufficient facilities for exerciseand recreation to meet his human needs. In particular,although there was at all times herein material an open,but secured, yard attache
 to the jail building, whichwould allow prisoners of the jail to exercise and getfresh air, plaintiff Larry Gray was denied access to theyard during his confinement in the Scott County Jail.", ' That the aforesaid actions of the defendants, which deprivedthe plaintiff of his constitutional and civil rig
s, were wanton, willful,and malicious and therefore entitle plaintiff Larry Gray to punitiveWHEREFORE, plaintiff Larry Gray prays judgment against the ScottCounty Sheriff, Bill Ferrell, and the Scott County Court Judges, LouisHirschowitz, Eldon Ziegenhorn and Dewaine Shaffer, and each of them jointlyand
everally, in the sum of FIVE THOUSAND DOLLARS ($5,000.00) in actualdamages and TEN THOUSAND DOLLARS ($10,000.00) in punitive damages.Comes now plaintiff Ronald V. Artis, by and through his attorney,Francis J. Siebert, and for his cause of action against the defendants', " Plaintiff Ronald V. Artis reall
es, reavers and incorporatesby reference each and every allegation contained in paragraphs numbered1, 2, 3, k, 5, 6 and 7 of Count I of Plaintiff's First Amended Complaint.", ' That plaintiff Ronald V. Artis was confined in the Scott CountyJail for a period of % days, from the 29th day of December, 1980
o', " That the conduct of the defendants violated plaintiff Ronald V.Artis's right to privacy as guaranteed by the Fourth Amendment of theUnited States Consitution, subjected said plaintiff to cruel and unusualpunishment in violation of the Eighth Amendment of the United StatesConstitution, and denied s
d plaintiff Due Process of Law in violationof the Fourteenth Amendment to the United States Constitution in the(a) While plaintiff Ronald V. Artis was confined in theScott County Jail his incoming and outgoing personalmail was read by the deputies of the Scott CountySheriff's Department and thereafter t
 private con-tents of said letters were-openly discussed betweenthe deputies of the Scott County Sheriff's Departmentand other inmates in the Scott County Jail.(b) Meals served to the plaintiff Ronald V. Artis while aprisoner in the Scott County Jail were inadequate inquality and quantity and far below
e average dailyrequirement for sedentary men. In addition when meatwas served to said plaintiff, it was often spoiled.(c) Plaintiff Ronald V. Artis was provided with totallyinadequate hygiene facilities in that he was allowedto shower only once a week, required to sleep on mattress-es which were filthy
d crawling with lice and othervermin and was required to wear his clothing for anexcessively lengthy period of time without the opportunityto have his clothing washed or laundered.(d) While plaintiff Ronald V. Artis was confined to the ScottCounty Jail his visiting by his family and friends waslimited t
five minutes a week and no private visiting(e) Plaintiff Ronald V. Artis, while a prisoner in the ScottCounty Jail, was provided with insufficient facilitiesfor exercise and recreation to meet his human needs andhis access to the open, but secure, yard attached to thejail building, which would have prov
ed him with exerciseand fresh air, was unreasonably restricted.", ' That the aforesaid actions of the defendants, which deprivedplaintiff Ronald V. Artis of his constitutional and civil rights, werewanton, willful, and malicious and therefore entitle plaintiff Ronald V.WHEREFORE, plaintiff Ronald V. Art
 prays judgment against the ScottCounty Sheriff, Bill Ferrell, and the Scott County Court Judges, LouisHirschowitz, Eldon Ziengenhorn and Dewaine Shaffer, and each of themjointly and severally, in the sum of FIVE THOUSAND DOLLARS ($5,000.00)in actual damages and TEN THOUSAND DOLLARS ($10,000.00) in puni
ve damages.Comes now plaintiff Alonzo Patterson, by and through his attorney,Francis J. Siebert, and for his cause of action against the defendants', " Plaintiff Alonzo Patterson realleges, reavers and incorporatesby reference each and every allegation contained in paragraphs 1, 2, 3,4, 5 and 6 of Count
 of Plaintiffs' First Amended Complaint.", ' That plaintiff Alonzo Patterson was confined to the Scott CountyJail for a period of 239 days, said time being from the 4th day of June,IF I1979 to the 5th day of June, 1979, from the 23rd day of June, 1980 tothe 26th day of June, 1980, and from the 1st day o
July, 1980 to the', " That the conduct of the defendants violated plaintiff AlonzoPatterson's right to privacy as guaranteed by the Fourth Amendment ofthe United States Constitution, subjected said plaintiff to cruel andunusual punishment in violation of the Eighth Amendment of the UnitedStates Constitu
on, and denied said plaintiff Due Process of Law inviolation of the Fourteenth Amendment to the United States Constitution(a) While plaintiff Alonzo Patterson was confined in theScott County Jail his incoming and outgoing personalmail was read by the deputies of the Scott CountySheriff's Department and
ereafter the private con-tents of said letters were openly discussed betweenthe deputies of the Scott County Sheriff's Departmentand other inmates in the Scott County Jail.(b) Meals served to the plaintiff Alonzo Patterson whilea prisoner in the Scott County Jail were inadequatein quality and quantity a
 far below the average dailyrequirement for sedentary men. In addition when meatwas served to said plaintiff, it was often spoiled.(c) Plaintiff Alonzo Patterson was provided with totallyinadequate hygiene facilities in that he was allowedto shower only once a week, required to sleep on mattresseswhich
re filthy and crawling with lice and othervermin and was required to wear his clothing for anexcessively lengthy period of time without the opportunityto have his clothing washed or laundered.(d) While plaintiff Alonzo Patterson was confined to theScott County Jail his visiting by his family and friends
s limited to five minutes a week and no private visiting(e) Plaintiff Alonzo Patterson, while a prisoner in the ScottCounty Jail, was provided with insufficient facilitiesfor exercise and recreation to meet his human needs andhis access to the open, but secure, yard attached to thejail building, which w
ld have provided him with exerciseand fresh air, was unreasonably restricted."]
/home/michael/miniconda3/lib/python3.11/site-packages/transformers/tokenization_utils_base.py:1601: FutureWarning: `clean_up_tokenization_spaces` was not set. It will be set to `True` by default. This behavior will be depracted in transformers v4.45, and will be then set to `False` by default. For more
tails check this issue: https://github.com/huggingface/transformers/issues/31884
  warnings.warn(
Hardware accelerator e.g. GPU is available in the environment, but no `device` argument is passed to the `Pipeline` object. Model will be on CPU.
The U.S. Attorney's Office for the Western District of Missouri has filed a civil action against the Scott County Sheriff's Department, the Scott County Court Judges, and the Scott County Sheriff for violating the civil rights of an inmate held in the Scott County Jail. According to the complaint, filed
n the U.S. District Court for the Western District of Missouri, on the 12th day of December, 1980 and the 30th day of December, 1980 inmate Larry Gray was confined to the Scott County Jail for a period of 92 days, from the 12th day of December, 1980 to the 30th. The complaint alleges that while Gray was
```

Figure 4: The Cleaned Text and The Summary(at the bottom) of a case from clearinhouse generated by pegasus

HAAG Research Report
NLP - Sentencias / NLP - Gen Team
Week 3

Víctor C. Fernández

September 2024

1 WEEKLY PROJECT UPDATES

What progress did you make in the last week?

- Have been creating code for:
  - Converting pdfs to text without data loss or words transformation.
  - Removing noise from the genereted text files such as headers, footers, repetitive symbols over the page.
  - Named Entity Recognition using SpaCy's es_core_news_lg model, grouping content in the document by type of information we may be able to use later on in the process.
- Meeting with Dr. Alexander on September 6th for an update call.
- Meeting with the NLP team on September 6th for our weekly meeting.
- Created a file containing OMSCS professors to contact for seminars.
- Cold emailed multiple professors for carrying out seminars.

What progress are you making next?

- Meeting with the NLP team on September 13th on our weekly meeting.
- Look into the use of pretrained LLMs for Information Retrieval, such as BERT or DBRX, as will most likely be what we'll need to use for extracting more precise information from the documents.

Is there anything blocking you from making progress?

No, nothing right now.

2 ABSTRACTS

1. **Title:** IR2: Information Regularization for Information Retrieval
   - **URL:** https://aclanthology.org/2024.lrec-main.810.pdf

   - **Abstract:** Effective information retrieval (IR) in settings with limited training data, particularly for complex queries, remains a challenging task. This paper introduces IR2, Information Regularization for Information Retrieval, a technique for reducing overfitting during synthetic data generation. This approach, representing a novel application of regularization techniques in synthetic data creation for IR, is tested on three recent IR tasks characterized by complex queries: DORIS-MAE, ArguAna, and WhatsThatBook. Experimental results indicate that our regularization techniques not only outperform previous synthetic query generation methods on the tasks considered but also reduce cost by up to 50%. Furthermore, this paper categorizes and explores three regularization methods at different stages of the query synthesis pipeline—input, prompt, and output—each offering varying degrees of performance improvement compared to models where no regularization is applied. This provides a systematic approach for optimizing synthetic data generation in data-limited, complex-query IR scenarios. All code, prompts and synthetic data are available at https://github.com/Info-Regularization/Information-Regularization.

   - **Summary:** The paper introduces IR2, a technique that uses Information Regularization to enhance the quality of synthetic data generation for Information Retrieval (IR) tasks, mainly involving complex queries. It aims at reducing overfitting during synthetic data creation by generating queries that have conceptual overlap with the original document but differ in phrasing and structure. The paper explores three regularization methods applied at different stages of the query synthesis pipeline: input document regularization, instruction regularization, and output query regularization. The experimental results demonstrate that these techniques outperform previous synthetic query generation methods on complex IR tasks and can even reduce costs.

   - **Relevance:** It improves information retrieval for complex queries. In the context of the NLP-Sentencias project, the judges' written decisions can be seen as complex documents, and the information we need to extract could be considered complex queries at some point. The IR2 technique could be leveraged to generate synthetic queries from these decisions, increasing training data and potentially improving the performance of the NLP tools to extract key information.

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 cleaning the generated txt files and extracting entities.

```python
def extract_case_number(text):
    case_number_pattern = r'Número\s+(?:único\s+)?de\s+caso\s+\
    (?:\(NUC\))?\s*:?\s*(\d+-\d+)'
    case_number_match = re.search(case_number_pattern, text,
    ↪   re.IGNORECASE)
    return case_number_match.group(1) if case_number_match else
    ↪   None
```

Code 1—Extract case number from text

```python
def extract_parties(doc):
    persons = set()
    organizations = set()
    for ent in doc.ents:
        if ent.label_ == "PER":
            persons.add(ent.text)
        elif ent.label_ == "ORG":
            organizations.add(ent.text)
    return {
        "persons": list(persons),
        "organizations": list(organizations)
    }
```

Code 2—Extract persons and organizations

```python
def extract_money_amounts(doc):
    for ent in doc.ents:
        if ent.label_ == "MONEY":
            money_amounts.append(ent.text)

    # Additional regex pattern for amounts that might be missed by
    ↪   SpaCy
    money_pattern = r'\$?\d{1,3}(?:,\d{3})*(?:\.\d{2})?(?:
    ↪   [a-zA-Z]+)?'
    regex_matches = re.findall(money_pattern, doc.text)

    # Combine SpaCy results and regex matches, removing duplicates
    all_amounts = list(set(money_amounts + regex_matches))
    return all_amounts
```

Code 3—Extract money amounts / numbers

```python
def extract_dates_and_events(doc):
    date_events = []
    for sent in doc.sents:
        date = None
        event = []
        for token in sent:
            if token.ent_type_ == "DATE":
                date = token.text
            elif token.pos_ == "VERB":
                event.append(token.text)
        if date and event:
            date_events.append({"date": date, "event": "
            ↪   ".join(event)})
    return date_events
```

Code 4—Extract money amounts / numbers

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 then extracted from PDF and doc files using the PyMuPDF library into txt files.
   - New text documents were then processed in order to remove headers, footers, pagination and other repetitive items in the corpus.
   - New text files were generated with the cleaned up content.
2. **Text Analysis and Feature Extraction:**
   - Named Entity Recognition was carried out to identify different parts of the documents content using SpaCy's Spanish large model es_core_news_lg.
   - New json files were generated containing the identified entities with the above indicated code.

5 SCRIPT VALIDATION (OPTIONAL)

Results were manually reviewed and thought through verifying the logic of the entities recognized within the context in the document. So far entities were correctly identified in a large percentage, but it is still not enough to be able to obtain the information we want for classifying the documents by time for case completion.

6 RESULTS VISUALIZATION

Resulting documents were shared in the private project repository given the nature of the content and the need to maintain privacy. Repository's link is the following (may only be accessed by members with authorization):
https://github.gatech.edu/calexander97/sentencias

7 PROOF OF WORK

The results obtained were reliable and stable, although they won't be enought to fully identify the context of the data we're looking for. We can extract a list of persons or organizations, but in order to know who is who and what role they are playing in the document we'll need extra processing with a different model in the line of an LLM such as BERT. Still, the text was correctly cleared removing unnecessary noise that does not add value to posterior processing.

Will keep moving forward during the coming week in order to enhance the entities extraction and start generating a data pool which we may then use to obtain insights on the differences between each case to identify what aspects affect the cases causing delays.

Week 3 | HAAG - NLP | Fall 2024

Alejandro Gomez

September 6th, 2024

1 Time-log

1.1 What progress did you make in the last week?

- Continued getting more familiar with general machine learning practices and dived into the current library we are using: spaCy. Last week I had learned to use it for Named Entity Recognition so this week I learned about model training and fine-tuning, so I attempted to fine-tune the spaCy spanish model so that I could have more control over the NER's that would output since the default outputs were not enough to meet our project objectives.

1.2 What are you planning on working on next?

- Will meet with Dr. Alexander and the NLP group to discuss our strategy moving forwards as we all ramp up and eventually converge to collaborate on building a model. Will need to identify an architecture design with the team.
- Update the website with this week's content.
- Potentially explore other models
- Learn more about fine-tuning and build a larger dataset for training and development to avoid overfitting

1.3 Is anything blocking you from getting work done?

N/A

2 Article Review

2.1 Abstract

A bottleneck in efficiently connecting new materials discoveries to established literature has arisen due to an increase in publications. This problem may be addressed by using named entity recognition (NER) to extract structured summary-level data from unstructured materials science text. We compare the performance of four NER models on three materials science datasets. The four models include a bidirectional long short- term memory (BiLSTM) and three transformer models (BERT, SciBERT, and MatBERT) with increasing de- grees of domain-specific materials science pre-training. MatBERT improves over the other two BERTBASE - based models by 1Despite relative architectural simplicity, the BiLSTM model consistently outperforms BERT, perhaps due to its domain-specific pre-trained word embeddings. Furthermore, MatBERT and SciBERT models outper- form the original BERT model to a greater extent in the small data limit. MatBERT's higher-quality predictions should accelerate the extraction of structured data from materials science literature. doi link[Tre22]

2.2 Summary

This article analyzes the effects of using different models for named entity recognition - a major component of the haag-nlp group's current effort. It alludes to the benefits of domain-specific pre-training and I've made note of this to apply this to our project - i.e. searching for a social scienes/law domain specific NER model for best results.

3 Scripts and Code Blocks

3.1 Code

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

# nlp = spacy.load("es_core_news_lg")
nlp = spacy.load("./output/model-best")

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: main script

```python
import spacy
from spacy.tokens import DocBin

nlp = spacy.blank("es")
TRAINING_DATA = [
    ("Juan P rez pag  5,000 d lares a Microsoft el 12 de abril de 2023 a las 10:00 AM.",
     [(0, 10, "PERSON"), (16, 29, "MONEY"), (32, 41, "ORG"), (45, 64, "DATE"), (71, 76, "TIME")]),

    ("Mar a G mez recibi  un bono de 3,000 euros de Google el 5 de marzo de 2022 a las 9:30 AM.",
     [(0, 11, "PERSON"), (31, 42, "MONEY"), (46, 52, "ORG"), (58, 74, "DATE"), (81, 89, "TIME")]),
]
## the DocBin will store the example documents
db = DocBin()
for text, annotations in TRAINING_DATA:
    doc = nlp(text)
    ents = []
    for start, end, label in annotations:
        span = doc.char_span(start, end, label=label)
        if span is not None:
            ents.append(span)
        else:
            print(f"Failed to create span for text: {text[start:end]}")
    doc.ents = ents
    db.add(doc)
db.to_disk("./train.spacy") #db.to_disk("./dev.spacy")
```

Listing 2: preprocessing for training

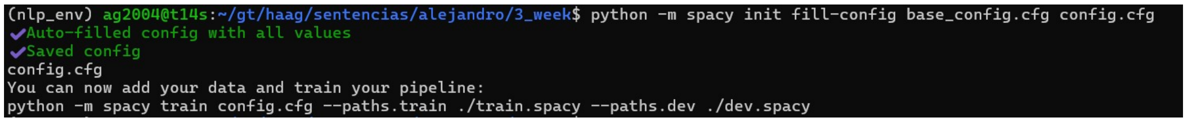
3.2 Documentation

Last week I learned to use the spaCy library but the named entities recognized by its default use were incomplete so I dug into the documentation to see how to broaden the NER. The docs instructed to fine tune the model, so my focus this week was to learn about fine tuning and how to conduct it witht the spaCy library. By following the documentation and adjusting code for my use case, I Was eventually able to fine tune a model and use it for my main script and it was successful! The model is currently overfitted, but I did demonstrate that finetuning will broaden the default NER's from just PER and ORG to PER, ORG, MONEY, and DATE.

3.3 Script Validation (optional)

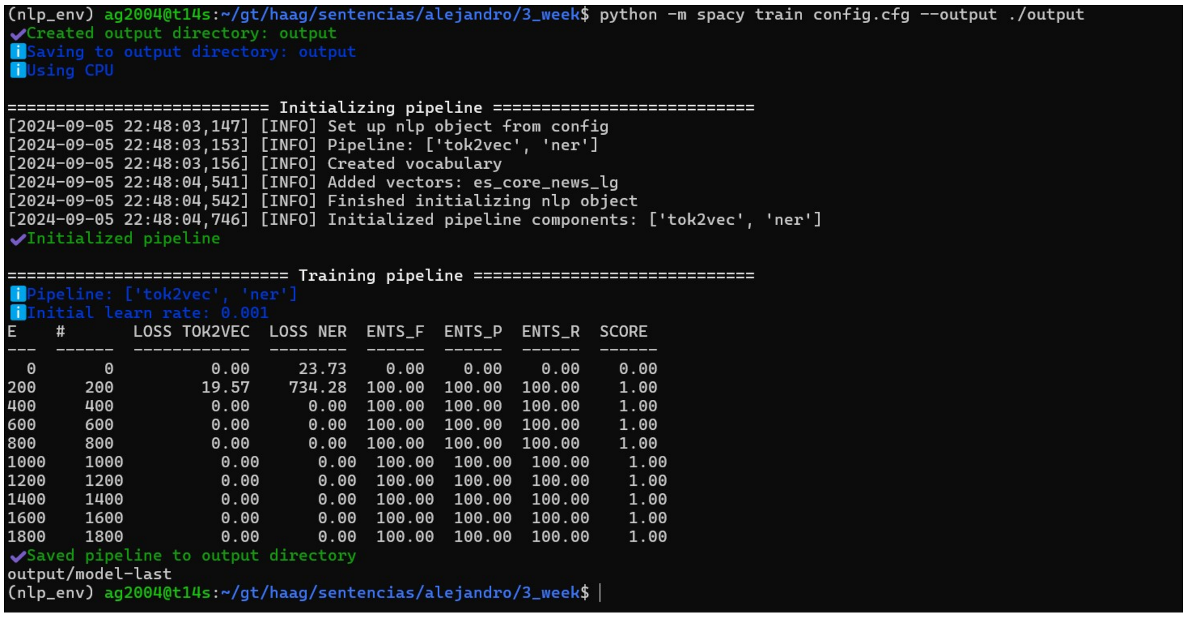
See images below

3.4 Results Visualization



```
(nlp_env) ag2004@t14s:~/gt/haag/sentencias/alejandro/3_week$ python -m spacy init fill-config base_config.cfg config.cfg
✔Auto-filled config with all values
✔Saved config
config.cfg
You can now add your data and train your pipeline:
python -m spacy train config.cfg --paths.train ./train.spacy --paths.dev ./dev.spacy
```

Figure 1: training config



```
(nlp_env) ag2004@t14s:~/gt/haag/sentencias/alejandro/3_week$ python -m spacy train config.cfg --output ./output
✔Created output directory: output
ℹSaving to output directory: output
ℹUsing CPU

=========================== Initializing pipeline ===========================
[2024-09-05 22:48:03,147] [INFO] Set up nlp object from config
[2024-09-05 22:48:03,153] [INFO] Pipeline: ['tok2vec', 'ner']
[2024-09-05 22:48:03,156] [INFO] Created vocabulary
[2024-09-05 22:48:04,541] [INFO] Added vectors: es_core_news_lg
[2024-09-05 22:48:04,542] [INFO] Finished initializing nlp object
[2024-09-05 22:48:04,746] [INFO] Initialized pipeline components: ['tok2vec', 'ner']
✔Initialized pipeline

============================= Training pipeline =============================
ℹPipeline: ['tok2vec', 'ner']
ℹInitial learn rate: 0.001
E    #       LOSS TOK2VEC  LOSS NER  ENTS_F  ENTS_P  ENTS_R  SCORE
---  ------  ------------  --------  ------  ------  ------  ------
  0       0          0.00     23.73    0.00    0.00    0.00    0.00
200     200         19.57    734.28  100.00  100.00  100.00    1.00
400     400          0.00      0.00  100.00  100.00  100.00    1.00
600     600          0.00      0.00  100.00  100.00  100.00    1.00
800     800          0.00      0.00  100.00  100.00  100.00    1.00
1000    1000          0.00      0.00  100.00  100.00  100.00    1.00
1200    1200          0.00      0.00  100.00  100.00  100.00    1.00
1400    1400          0.00      0.00  100.00  100.00  100.00    1.00
1600    1600          0.00      0.00  100.00  100.00  100.00    1.00
1800    1800          0.00      0.00  100.00  100.00  100.00    1.00
✔Saved pipeline to output directory
output/model-last
(nlp_env) ag2004@t14s:~/gt/haag/sentencias/alejandro/3_week$ |
```

Figure 2: fine tuning (overfit)

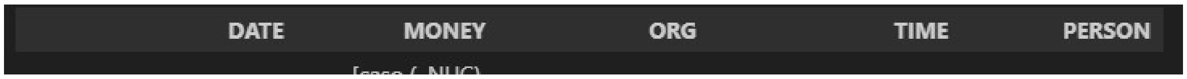


Figure 3: named entities recognized after fine tuning

3.5 Proof of Work

Scripts in GitHub Repo

4 Next Week's Proposal

- Work on the fine tuning process in order to have an accurate model for validation and testing and avoid overfitting.
- Consider exploring other models to compare outputs.
- Explore models and tools specific to the law domain.
- Work on a manually indexed dataset of reasonable size for fine tuning the data
- Update the NLP website with current records and check if the bio's are up.
- 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

[Tre22] Amalie ; Walker Nicholas ; Huo Haoyan ; Lee Sanghoon ; Cruse Kevin ; Dagdelen John ; Dunn Alexander ; Persson Kristin A. ; Ceder Gerbrand ; Jain Anubhav Trewartha, Amalie ; Trewartha. Quantifying the advantage of domain-specific pre-training on named entity recognition tasks in materials science. *Patterns (New York, N.Y.)*, 3(4):100488–1004898, 2022.

HAAG NLP Sentencias — Week 2 Report
NLP-Gen Team

Karol Gutierrez

September 6, 2024

1 Weekly Project Update

1.1 What progress did you make in the last week?

- Call with Dr. Alexander and on September 6. We introduced ourselves to a broader team of collaborators of Dr. Alexander (Jose Torres and Laura Bastidas) and planned on the specific attributes that we want to extract from the documents moving forward.
- Experimentation with SpaCy library for Name Entity Recognition (NER). Code regarding date extraction from the PDF files, analysis of occurences over time.
- Literature review for use of LLMs for domain specific texts.
- Fulfill my role as Meet Manager/Documentor by working on the tasks expected for my position.
- Start code work on LLM usages.

1.2 What are you planning on working on next?

- Experiment with existing LLMs during weekend.
- Further literature review on LLMs.
- Give proposal to team early next week to split work and decide on tools to use.
- Design benchmark to evaluate performance of different models.
- Work with team in a consolidated code base and split sections to retrieve (ordinance number, case type, plaintiff, etc).
- Continue fulfilling my role as Meet Manager/Documentor by working on the tasks expected for my position (gather notes from meetings and prepare recordings).

1.3 Is anything blocking you from getting work done?

No.

2 Literature Review

Paper: A pre-trained BERT for Korean medical natural language processing [KKL+22].

2.1 Abstract

With advances in deep learning and natural language processing (NLP), the analysis of medical texts is becoming increasingly important. Nonetheless, despite the importance of processing medical texts, no research on Korean medical-specifc language models has been conducted. The Korean medical text is highly difcult to analyze because of the agglutinative characteristics of the language, as well as the complex terminologies in the medical domain. To solve this problem, we collected a Korean medical corpus and used it to train the language models. In this paper, we present a Korean medical language model based on deep learning NLP. The model was trained using the pre-training framework of BERT for the medical context based on a state-of-the-art Korean language model. The pre-trained model showed increased accuracies of 0.147 and 0.148 for the masked language model with next sentence prediction. In the intrinsic evaluation, the next sentence prediction accuracy improved by 0.258, which is a remarkable enhancement. In addition, the extrinsic evaluation of Korean medical semantic textual similarity data showed a 0.046 increase in the Pearson correlation, and the evaluation for the Korean medical named entity recognition showed a 0.053 increase in the F1-score.

2.2 Summary

The text discusses the development of a Korean medical language model based on BERT for natural language processing (NLP) in the medical domain. By training the model on a Korean medical corpus, the pre-trained model demonstrated improved accuracies in tasks such as masked language modeling and next sentence prediction, as well as enhanced performance in semantic textual similarity and named entity recognition evaluations specific to Korean medical text analysis. The study highlights the importance of domain-specific language models, like KR-BERT, in addressing the complexities of medical terminology and language characteristics for effective text processing in the healthcare domain.

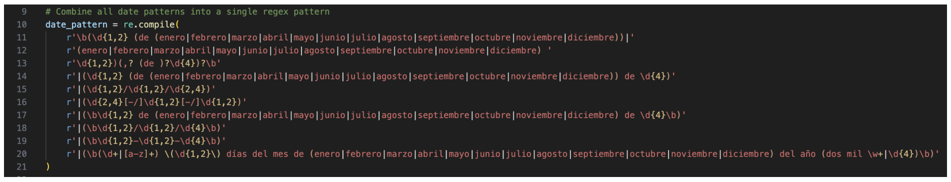
2.3 Relevance

This paper is relevant for future advancements in NLP for medical applications in Korean and can be the base for other non-English applications. It showed the importance of creating language and domain-specific models in order to achieve higher accuracy in specialized tasks. In particular, the applied methodology can be used in the Sentencias project, where we are handling legal texts in Spanish.

3 Scripts and code blocks

All the existing code is in the new private repository. Since we are handling private information from the PDF files, it was decided alongside Dr. Alexander that we should add all of our code work here from now on.

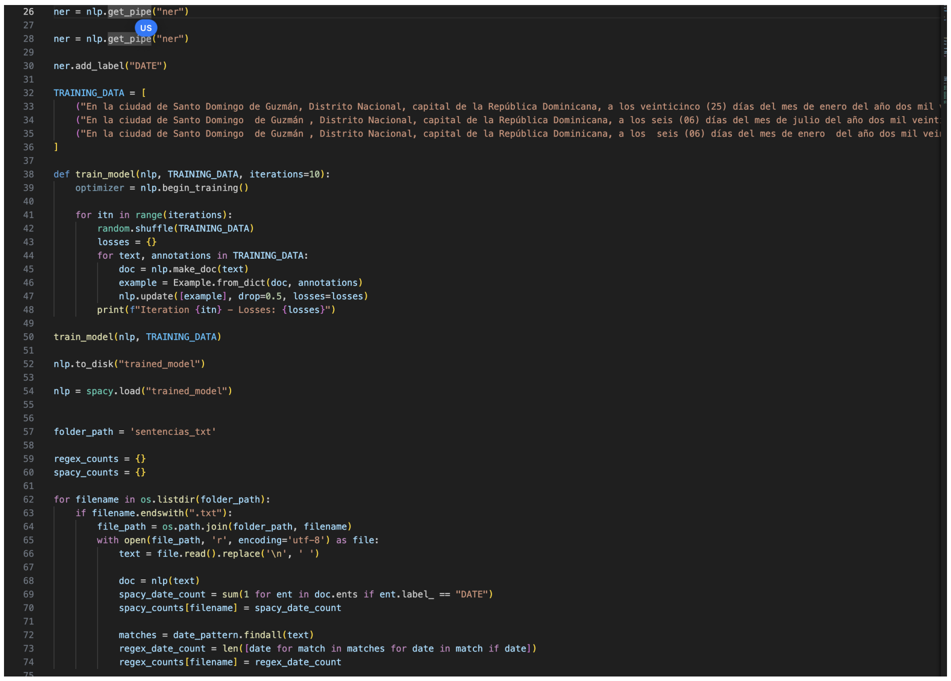
I added scripts that use the SpaCy library to try to retrieve the dates and also get specific samples to train a model. I also compare the results with the existing Regex implementation, and I expanded the regex to consider more cases for date representation.



```python
# Combine all date patterns into a single regex pattern
date_pattern = re.compile(
    r'\b(\d{1,2} (de (enero|febrero|marzo|abril|mayo|junio|julio|agosto|septiembre|octubre|noviembre|diciembre))|'
    r'(enero|febrero|marzo|abril|mayo|junio|julio|agosto|septiembre|octubre|noviembre|diciembre) '
    r'\d{1,2})(,? (de )?\d{4})?\b'
    r'|(\d{1,2} (de (enero|febrero|marzo|abril|mayo|junio|julio|agosto|septiembre|octubre|noviembre|diciembre)) de \d{4})'
    r'|(\d{1,2}/\d{1,2}/\d{2,4})'
    r'|(\d{2,4}[-/]\d{1,2}[-/]\d{1,2})'
    r'|(\b\d{1,2} de (enero|febrero|marzo|abril|mayo|junio|julio|agosto|septiembre|octubre|noviembre|diciembre) de \d{4}\b)'
    r'|(\b\d{1,2}/\d{1,2}/\d{4}\b)'
    r'|(\b\d{1,2}-\d{1,2}-\d{4}\b)'
    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)'
)
```

Figure 1: Updated regex

For the current submission we are experimenting with sample dates that can be trained using spaCy in order to retrieve the dates. However, the size of the training set is still small and the performance is not adequate. Nevertheless, it serves as a proof of concept for future development. The current code logic is explained in Figure 4, and the visualization compares the number of dates retrieved by spaCy vs the regex implementation.



```python
ner = nlp.get_pipe("ner")

ner = nlp.get_pipe("ner")

ner.add_label("DATE")

TRAINING_DATA = [
    ("En la ciudad de Santo Domingo de Guzmán, Distrito Nacional, capital de la República Dominicana, a los veinticinco (25) días del mes de enero del año dos mil
    ("En la ciudad de Santo Domingo  de Guzmán , Distrito Nacional, capital de la República Dominicana, a los seis (06) días del mes de julio del año dos mil veint
    ("En la ciudad de Santo Domingo  de Guzmán , Distrito Nacional, capital de la República Dominicana, a los  seis (06) días del mes de enero  del año dos mil vei
]

def train_model(nlp, TRAINING_DATA, iterations=10):
    optimizer = nlp.begin_training()

    for itn in range(iterations):
        random.shuffle(TRAINING_DATA)
        losses = {}
        for text, annotations in TRAINING_DATA:
            doc = nlp.make_doc(text)
            example = Example.from_dict(doc, annotations)
            nlp.update([example], drop=0.5, losses=losses)
        print(f"Iteration {itn} - Losses: {losses}")

train_model(nlp, TRAINING_DATA)

nlp.to_disk("trained_model")

nlp = spacy.load("trained_model")


folder_path = 'sentencias_txt'

regex_counts = {}
spacy_counts = {}

for filename in os.listdir(folder_path):
    if filename.endswith(".txt"):
        file_path = os.path.join(folder_path, filename)
        with open(file_path, 'r', encoding='utf-8') as file:
            text = file.read().replace('\n', ' ')

            doc = nlp(text)
            spacy_date_count = sum(1 for ent in doc.ents if ent.label_ == "DATE")
            spacy_counts[filename] = spacy_date_count

            matches = date_pattern.findall(text)
            regex_date_count = len([date for match in matches for date in match if date])
            regex_counts[filename] = regex_date_count
```

Figure 2: Code block

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.

5 Script Validation

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

6 Results Visualization

Plots of the dates retrieved using the new methods. Figure 5. AS can be seen in Figure 4, the level of dates found using the trained model is minimal compared to the regular expressions, however, this is due to the minimal size of the training set.

7 Proof of Work

All the scripts work end to end from the starting PDF files to the generated plots and printing of results. A benchmark will be included in next deliverable so we can have a measure of performance and then determine the quality of the work.

8 Next Week's Proposal

Refer to section 1.2 for details (avoid repetition).

References

[KKL+22] Youngtae Kim, Jihoon Kim, Jeong Min Lee, et al. A pre-trained bert for korean medical natural language processing. *Scientific Reports*, 12:13847, 2022.

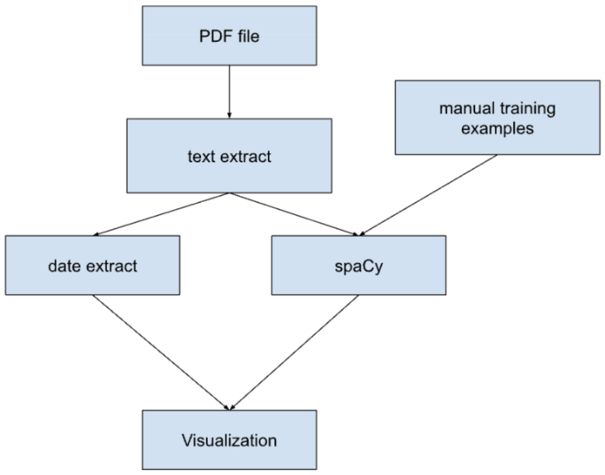


Figure 3: Code logic flow chart.

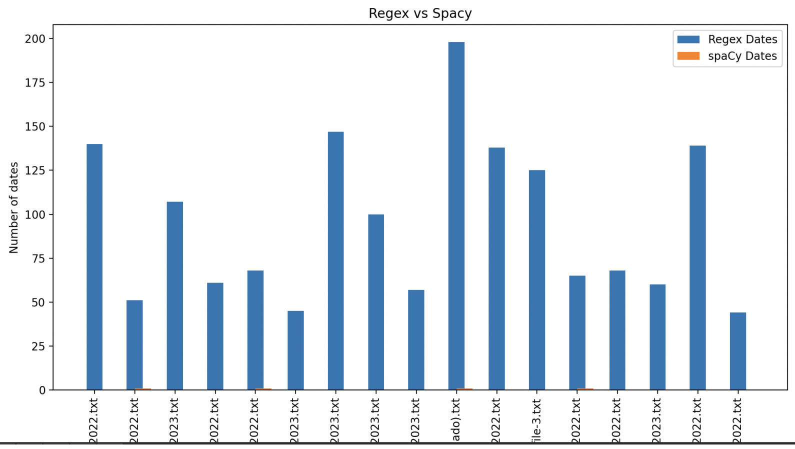


Figure 4: Dates occurrences over time.

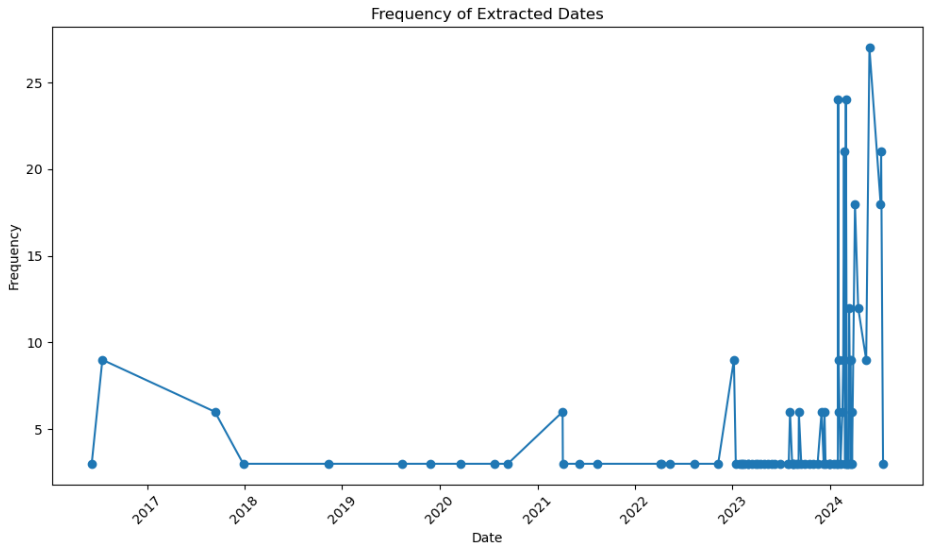


Figure 5: Dates occurrences over time.

Week 3 Research Report

Thomas Orth (NLP Summarization / NLP Gen Team)

September 2024

1. Met with VIP sub teams to discuss research and new direction for tasking
2. Created initial finetuning pipeline
3. Met with Dr. Alexander to talk about compute needs for the VIP class and short term planning

0.1 What are you planning on working on next?

1. Send Dr. Alexander forms on PACE cluster for access.
2. Clean up finetuning pipeline and pass along to VIP students to start experimenting.
3. Expand pipeline to include LongT5.

0.2 Is anything blocking you from getting work done?

1. None

1 Abstracts

- Title: LongT5: Efficient Text-To-Text Transformer for Long Sequences. Conference: ACL, Findings of the Association for Computational Linguistics: NAACL 2022
- Abstract: Recent work has shown that either (1) increasing the input length or (2) increasing model size can improve the performance of Transformer-based neural models. In this paper, we present LongT5, a new model that explores the effects of scaling both the input length and model size at the same time. Specifically, we integrate attention ideas from long-input transformers (ETC), and adopt pre-training strategies from summarization pre-training (PEGASUS) into the scalable T5 architecture. The result is a new attention mechanism we call Transient Global (TGlobal), which mimics ETC's local/global attention mechanism, but without requiring additional side-inputs. We are able to achieve state-of-the-art results on

several summarization and question answering tasks, as well as outperform the original T5 models on these tasks. We have open sourced our architecture and training code, as well as our pre-trained model checkpoints.

- Summary: This paper introduces LongT5, a new neural model that combines ideas from long-input transformers and summarization pre-training to improve performance on tasks like summarization and question answering. The model, which includes a new attention mechanism called Transient Global, achieves state-of-the-art results and surpasses the original T5 models on these tasks.
- Relevance: The presentation of metrics may be one to follow for the summarization project.

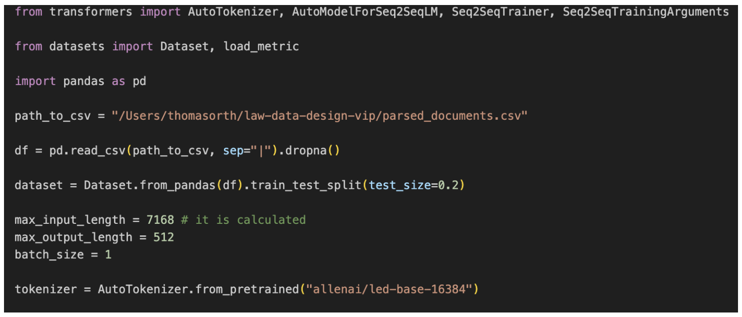
2 Relevant Info

- LED model is a proposed transformer technique from 2020 for the use in long document tasks in NLP. The paper on it can be found here.
- Legal documents are very long, making LED a good initial model to explore
- One source of data that concerns the NLP-Summarization project is data from the Clearinghouse as well as COVID data, but I have not explored that website much.

3 Scripts

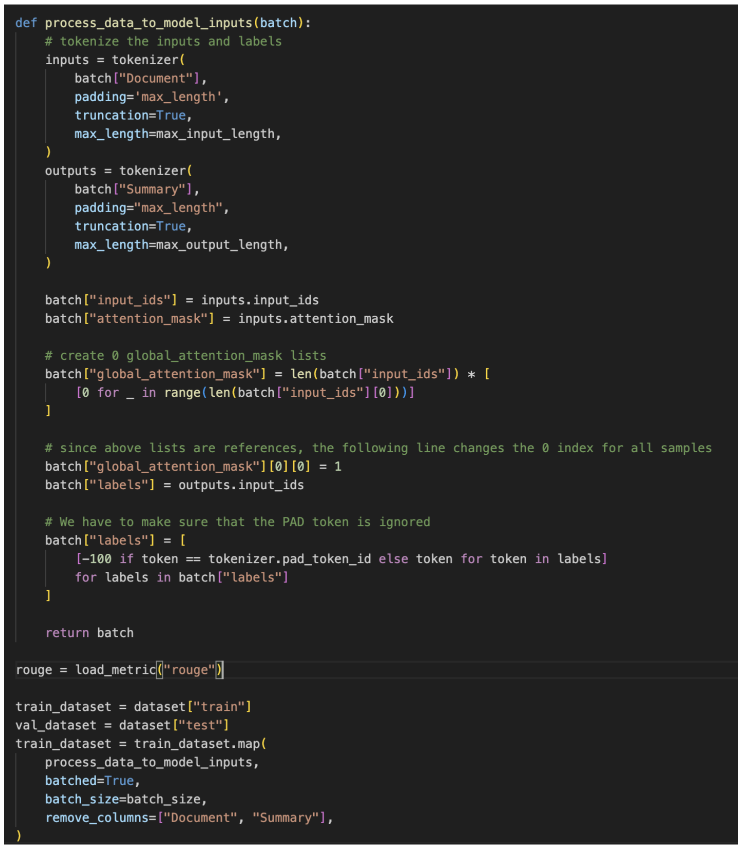
1. All scripts uploaded to https://github.com/Human-Augment-Analytics/NLP-Gen
2. Scripts were run with the following file for testing: `https://gatech.box.com/s/hv70flwkm977gky004l5vz15rpgfdmir`
3. Thomas-Orth/train_huggingface.py
   - Brief Description: This finetunes an LED model on the clearinghouse dataset in its current state
   - Status: Tested by running the pipeline to completion without issue
   - Important Code Blocks:
     (a) First block: Read in and set up the dataset
     (b) Second block: Set up train and validation dataset to be processed by tokenizer
     (c) Third Block: Configure model and train

- Screenshot of code:



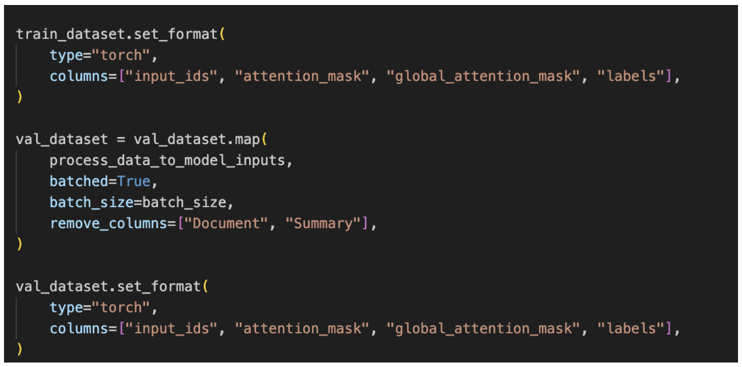
```python
from transformers import AutoTokenizer, AutoModelForSeq2SeqLM, Seq2SeqTrainer, Seq2SeqTrainingArguments

from datasets import Dataset, load_metric

import pandas as pd

path_to_csv = "/Users/thomasorth/law-data-design-vip/parsed_documents.csv"

df = pd.read_csv(path_to_csv, sep="|").dropna()

dataset = Dataset.from_pandas(df).train_test_split(test_size=0.2)

max_input_length = 7168 # it is calculated
max_output_length = 512
batch_size = 1

tokenizer = AutoTokenizer.from_pretrained("allenai/led-base-16384")
```

Figure 1: First screenshot



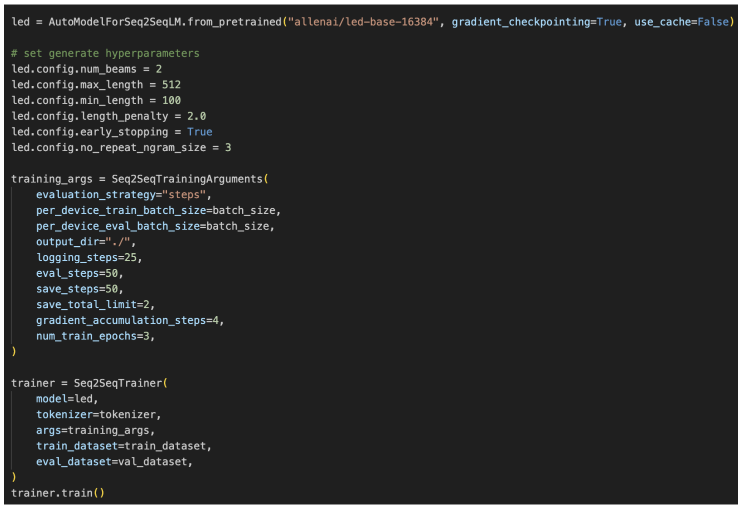
```python
def process_data_to_model_inputs(batch):
    # tokenize the inputs and labels
    inputs = tokenizer(
        batch["Document"],
        padding='max_length',
        truncation=True,
        max_length=max_input_length,
    )
    outputs = tokenizer(
        batch["Summary"],
        padding="max_length",
        truncation=True,
        max_length=max_output_length,
    )

    batch["input_ids"] = inputs.input_ids
    batch["attention_mask"] = inputs.attention_mask

    # create 0 global_attention_mask lists
    batch["global_attention_mask"] = len(batch["input_ids"]) * [
        [0 for _ in range(len(batch["input_ids"][0]))]
    ]

    # since above lists are references, the following line changes the 0 index for all samples
    batch["global_attention_mask"][0][0] = 1
    batch["labels"] = outputs.input_ids

    # We have to make sure that the PAD token is ignored
    batch["labels"] = [
        [-100 if token == tokenizer.pad_token_id else token for token in labels]
        for labels in batch["labels"]
    ]

    return batch

rouge = load_metric("rouge")

train_dataset = dataset["train"]
val_dataset = dataset["test"]
train_dataset = train_dataset.map(
    process_data_to_model_inputs,
    batched=True,
    batch_size=batch_size,
    remove_columns=["Document", "Summary"],
)
```

Figure 2: Second screenshot



```python
train_dataset.set_format(
    type="torch",
    columns=["input_ids", "attention_mask", "global_attention_mask", "labels"],
)

val_dataset = val_dataset.map(
    process_data_to_model_inputs,
    batched=True,
    batch_size=batch_size,
    remove_columns=["Document", "Summary"],
)

val_dataset.set_format(
    type="torch",
    columns=["input_ids", "attention_mask", "global_attention_mask", "labels"],
)
```

Figure 3: Third screenshot



```python
led = AutoModelForSeq2SeqLM.from_pretrained("allenai/led-base-16384", gradient_checkpointing=True, use_cache=False)

# set generate hyperparameters
led.config.num_beams = 2
led.config.max_length = 512
led.config.min_length = 100
led.config.length_penalty = 2.0
led.config.early_stopping = True
led.config.no_repeat_ngram_size = 3

training_args = Seq2SeqTrainingArguments(
    evaluation_strategy="steps",
    per_device_train_batch_size=batch_size,
    per_device_eval_batch_size=batch_size,
    output_dir="./",
    logging_steps=25,
    eval_steps=50,
    save_steps=50,
    save_total_limit=2,
    gradient_accumulation_steps=4,
    num_train_epochs=3,
)

trainer = Seq2SeqTrainer(
    model=led,
    tokenizer=tokenizer,
    args=training_args,
    train_dataset=train_dataset,
    eval_dataset=val_dataset,
)
trainer.train()
```

Figure 4: Fourth screenshot

4. Flow Diagram for Parse and Summarization scripts:

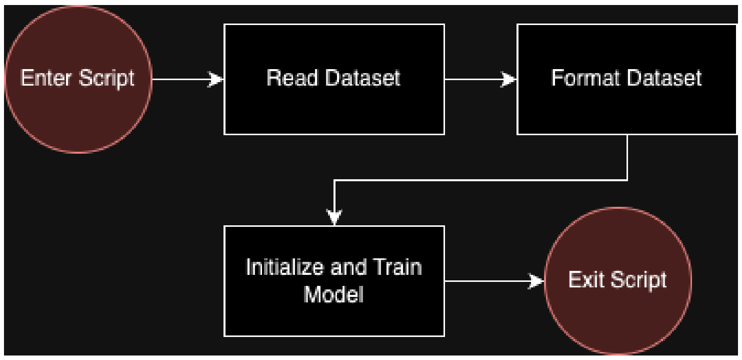


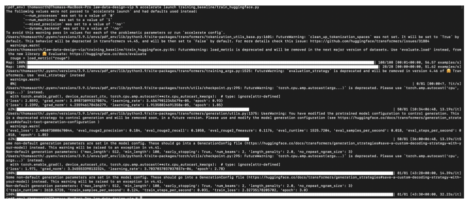
Figure 5: Flow diagram

5. Running scripts:
   (a) Download the script, huggingface.requirements.txt, and the csv from the box link.
   (b) Update the path variable at the top of the script to match where the csv is stored on your laptop.
   (c) Run: python -m pip install requirements.txt
   (d) Run: python train_huggingface.py

4 Documentation

1. Download CSV file, with two columns: Document and Summary
2. Update script to point to the CSV file
3. Train model
4. Output Model Checkpoints

5 Results



```
(pdf_env) thomasorth@Thomass-MacBook-Pro law-data-design-vip % accelerate launch training_baseline/train_huggingface.py
The following values were not passed to `accelerate launch` and had defaults used instead:
        `--num_processes` was set to a value of `0`
        `--num_machines` was set to a value of `1`
        `--mixed_precision` was set to a value of `'no'`
        `--dynamo_backend` was set to a value of `'no'`
To avoid this warning pass in values for each of the problematic parameters or run `accelerate config`.
/Users/thomasorth/.pyenv/versions/3.9.1/envs/pdf_env/lib/python3.9/site-packages/transformers/tokenization_utils_base.py:1601: FutureWarning: `clean_up_tokenization_spaces` was not set. It will be set to `True` by
default. This behavior will be depracted in transformers v4.45, and will be then set to `False` by default. For more details check this issue: https://github.com/huggingface/transformers/issues/31884
  warnings.warn(
/Users/thomasorth/law-data-design-vip/training_baseline/train_huggingface.py:54: FutureWarning: load_metric is deprecated and will be removed in the next major version of datasets. Use 'evaluate.load' instead, from
 the new library 🤗 Evaluate: https://huggingface.co/docs/evaluate
  rouge = load_metric("rouge")
Map: 100%|██████████████████████████████████████████████████████████████████| 108/108 [00:01<00:00, 56.57 examples/s]
Map: 100%|██████████████████████████████████████████████████████████████████| 28/28 [00:00<00:00, 51.62 examples/s]
/Users/thomasorth/.pyenv/versions/3.9.1/envs/pdf_env/lib/python3.9/site-packages/transformers/training_args.py:1525: FutureWarning: `evaluation_strategy` is deprecated and will be removed in version 4.46 of 🤗 Tran
sformers. Use `eval_strategy` instead
  warnings.warn(
  0%|                                                                                   | 0/81 [00:00<?, ?it/s]
/Users/thomasorth/.pyenv/versions/3.9.1/envs/pdf_env/lib/python3.9/site-packages/torch/utils/checkpoint.py:295: FutureWarning: `torch.cpu.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cpu',
args...)` instead.
  with torch.enable_grad(), device_autocast_ctx, torch.cpu.amp.autocast(**ctx.cpu_autocast_kwargs):  # type: ignore[attr-defined]
{'loss': 2.8592, 'grad_norm': 3.898730993270874, 'learning_rate': 3.45679012345679e-05, 'epoch': 0.93}
{'loss': 2.2392, 'grad_norm': 4.235946178436279, 'learning_rate': 1.91358024691358e-05, 'epoch': 1.85}
 62%|████████████████████████████████████████                                  | 50/81 [10:34<06:48, 13.19s/it]
/Users/thomasorth/.pyenv/versions/3.9.1/envs/pdf_env/lib/python3.9/site-packages/transformers/generation/utils.py:1375: UserWarning: You have modified the pretrained model configuration to control generation. This
is a deprecated strategy to control generation and will be removed soon, in a future version. Please use and modify the model generation configuration (see https://huggingface.co/docs/transformers/generation_strate
gies#default-text-generation-configuration )
  warnings.warn(
{'eval_loss': 2.406073808670044, 'eval_rouge2_precision': 0.184, 'eval_rouge2_recall': 0.1058, 'eval_rouge2_fmeasure': 0.1176, 'eval_runtime': 1525.7204, 'eval_samples_per_second': 0.018, 'eval_steps_per_second': 0
.018, 'epoch': 1.85}
 62%|████████████████████████████████████████                                  | 50/81 [36:00<06:48, 13.19s/itS
ome non-default generation parameters are set in the model config. These should go into a GenerationConfig file (https://huggingface.co/docs/transformers/generation_strategies#save-a-custom-decoding-strategy-with-y
our-model) instead. This warning will be raised to an exception in v4.41.
Non-default generation parameters: {'max_length': 512, 'min_length': 100, 'early_stopping': True, 'num_beams': 2, 'length_penalty': 2.0, 'no_repeat_ngram_size': 3}
/Users/thomasorth/.pyenv/versions/3.9.1/envs/pdf_env/lib/python3.9/site-packages/torch/utils/checkpoint.py:295: FutureWarning: `torch.cpu.amp.autocast(args...)` is deprecated. Please use `torch.amp.autocast('cpu',
args...)` instead.
  with torch.enable_grad(), device_autocast_ctx, torch.cpu.amp.autocast(**ctx.cpu_autocast_kwargs):  # type: ignore[attr-defined]
{'loss': 1.971, 'grad_norm': 3.345553398132324, 'learning_rate': 3.7037037037037037e-06, 'epoch': 2.78}
100%|██████████████████████████████████████████████████████████████████████████| 81/81 [43:28<00:00, 14.39s/it]
Some non-default generation parameters are set in the model config. These should go into a GenerationConfig file (https://huggingface.co/docs/transformers/generation_strategies#save-a-custom-decoding-strategy-with-
your-model) instead. This warning will be raised to an exception in v4.41.
Non-default generation parameters: {'max_length': 512, 'min_length': 100, 'early_stopping': True, 'num_beams': 2, 'length_penalty': 2.0, 'no_repeat_ngram_size': 3}
{'train_runtime': 2610.5728, 'train_samples_per_second': 0.124, 'train_steps_per_second': 0.031, 'train_loss': 2.32735178205702, 'epoch': 3.0}
100%|██████████████████████████████████████████████████████████████████████████| 81/81 [43:30<00:00, 32.23s/it]
(pdf_env) thomasorth@Thomass-MacBook-Pro law-data-design-vip %
```

Figure 6: Output from initial training

This is the initial output of training. I need to refine the output.

6 Proof of Results

The pipeline is based on the officially linked notebook for LED: `https://colab.research.google.com/drive/12LjJazBl7Gam0XBPy_y0CTOJZeZ34c2v?usp=sharing` and a deriavation of that work for arxiv training: `https://github.com/Bakhitovd/led-base-7168-ml`. So the pipeline itself is well thought out.

6.0.1 Known Limitations

The initial work is done with an early version of the dataset so further results with this dataset may be sub-par to start.