Word count: 1973/2000

Penalty calculation: 0%

ECE1786 – Final Report

QuickPat

Stephen He – 997632931
Tamara O'Connell – 997281132

Introduction

The goal of QuickPat is to automate patent text generation. In particular, QuickPat generates an abstract and additional claims for use in a patent application, based on a single human-authored claim that defines what is new and inventive about the invention.

A patent application is a document intended to outline a new technological invention and typically includes sections for: claims, description and drawings, and abstract.[1] The claims define the legal boundaries of the invention. The first claim is referred to herein as the "primary claim" and is typically an independent claim intended to capture the invention with the broadest definition. The description and drawings further describe and illustrate the operation or use of the invention. The abstract is a brief summary of the technical disclosure and is often based on the primary claim.

The motivation for the project is to reduce time and costs in drafting a patent application. Large language models have seen hundreds of thousands of historical patents, so these models could provide benefits like summarizing the given patent claims, generating text in the "patent-appropriate" language, and coming up with ideas to inspire new claims.

Overall Architecture

Figure 1 illustrates the QuickPat architecture. With just the primary claim and suitable prompt provided, a large language model like GPT-4 or Llama 2 will help generate text for the abstract and additional claims of the patent.

[1] https://ised-isde.canada.ca/site/canadian-intellectual-property-office/en/patents/what-patent

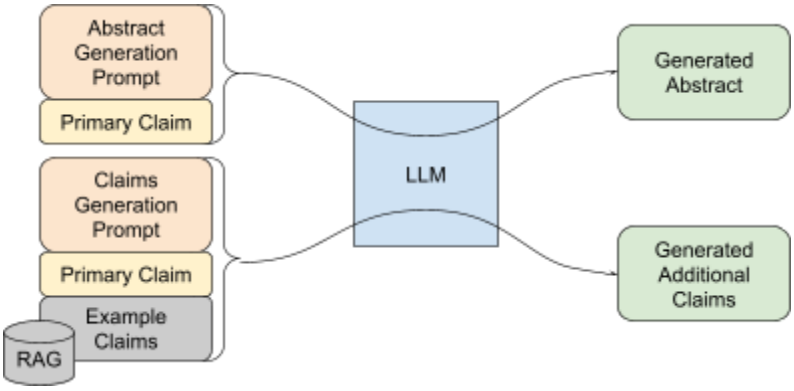

Figure 1 - QuickPat at a high-level

Background & Related Work

Various natural language processing tools have been contemplated in the domain of patents. In 2020, Google released a white paper[2] on the use of their BERT model on patents.[3] In it, the team described their fine-tuning of the BERT algorithm, trained exclusively on patent text, and used to generate contextual synonyms.

Patent text generation has been explored using GPT-like models, such as GPT-J-6B, an open-source LLM and the largest pre-trained model open to the public at the time of publication of "Evaluating Generative Patent Language Models" by Jieh-Sheng Lee in June 2022 ("J-S Lee").[4] [5] In J-S Lee, GPT-J-6B was pre-trained from scratch with US patents.

In the marketplace, PatentBots offers a service for automated generation of claim summaries and brief descriptions of the figures.[6]

Data and Data Processing

Figure 2 illustrates the data processing performed on raw XML data from the USPTO[7].

[2] https://services.google.com/fh/files/blogs/bert_for_patents_white_paper.pdf
[3] https://cloud.google.com/blog/products/ai-machine-learning/how-ai-improves-patent-analysis
[4] https://6b.eleuther.ai/
[5] https://arxiv.org/abs/2206.14578
[6] https://www.patentbots.com/patent-drafting-automation
[7] https://bulkdata.uspto.gov/

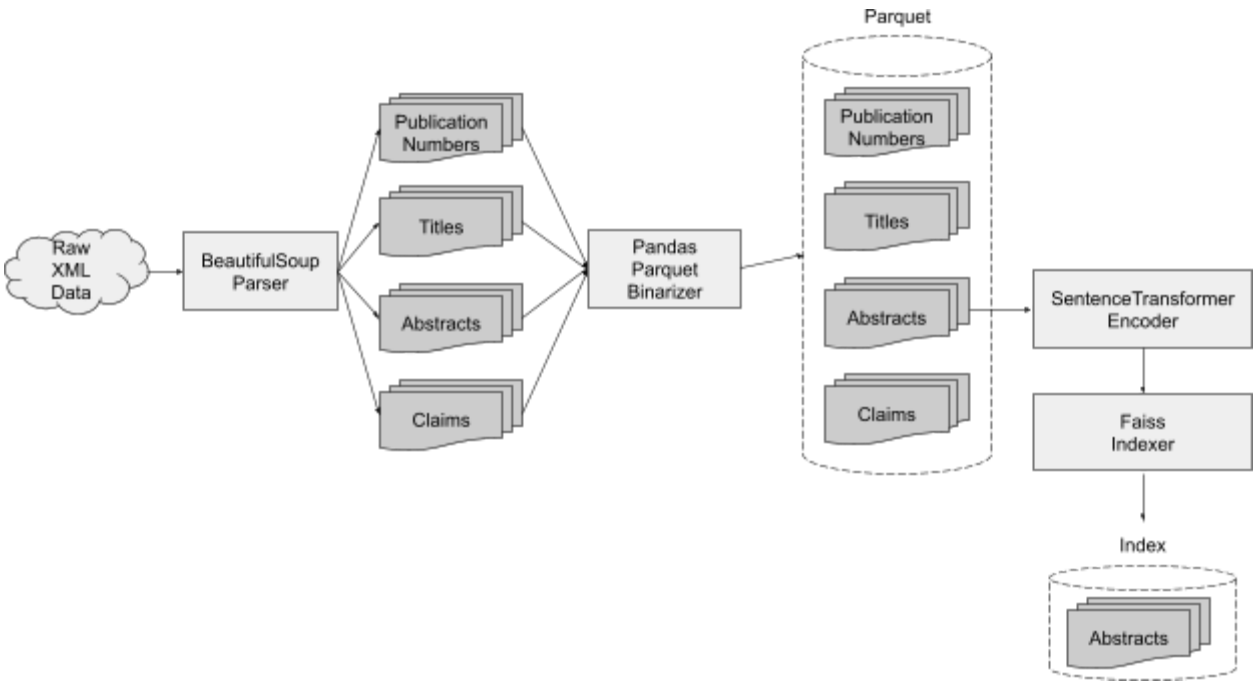

Figure 2 - Data processing

Each XML file contains one week of published US patent applications. BeautifulSoup[8] was used to parse these files and extract fields of interest. Since these files were already organized and well-labelled, minimal processing was required, such as trimming white spaces and filtering out unusable patent data.

Each file contains ~9000 patents. We extracted the following fields:

publ_number	title	abstract	claim_data
20200000001	SYSTEM FOR CONNECTING IMPLEMENT TO MOBILE...	An apparatus for connecting an implement...	[1. Apparatus for connecting an implement...
…			

Table 1 - Processed patent data

We used Pandas[9] to store and manipulate the data, and store them as Parquet and CSV files.

We randomly selected 5 entries from this database to use as our training samples.

All processed patent data are indexed using the FAISS library[10] so that they can be queried as a local database for RAG (Retrieval Augmented Generation).

[8] https://pypi.org/project/beautifulsoup4/
[9] https://pandas.pydata.org/
[10] https://faiss.ai/

Architecture and Software

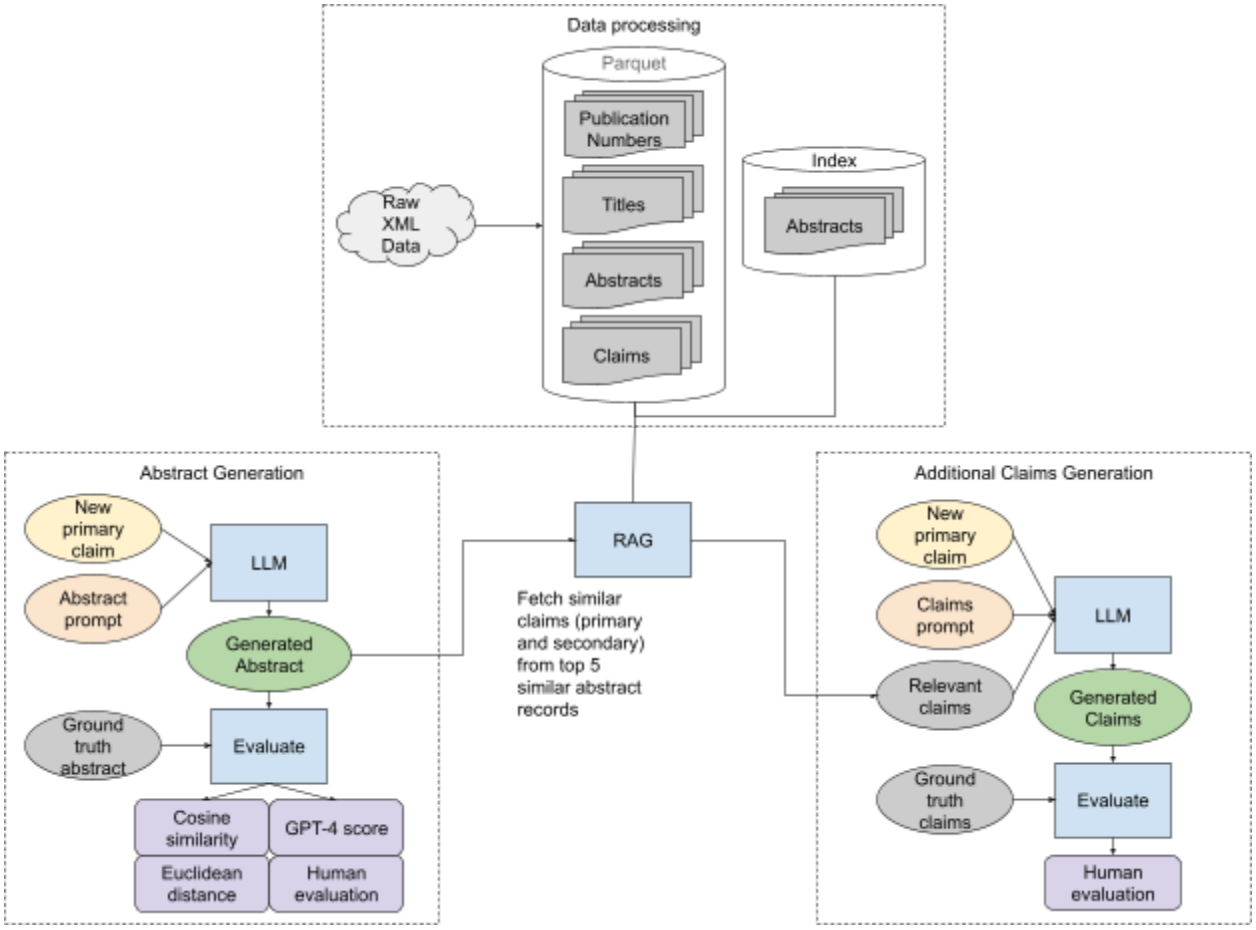

Figure 3 - QuickPat Architecture

The QuickPat architecture in Figure 3 includes:

1. Data Processing
    - As described in the "Data and Data Processing" section, above, patent data is processed and stored for access by the RAG
    - Publication numbers, titles, abstract and claims are binarized in a parquet file
    - Abstract data is also encoded and indexed in an index file
2. Retrieval Augmented Generation (RAG)
    - Each processed patent is indexed so we can identify similar patents to augment the claims prompt
    - Using SentenceTransformer[11], an embedding with 768 dimensions is generated using *paraphrase-mpnet-base-v2* for a given patent's abstract. This model was chosen for its balance of performance to speed to model size.

[11] https://www.sbert.net/

    - Each embedding maps to a patent index, so that when we find similar patents during the claims generation step, we can find the full claims efficiently.
3. Abstract Generation
    - Generate abstract using an LLM (e.g., GPT-4), based on a new primary claim and prompt engineering
        - "New primary claim" input is an existing claim from the training samples; during deployment, it would be a new claim
    - With prompt engineering, the final prompt included instructions to
        - Avoid referencing certain words
        - Avoid discussing advantages
        - Use simple and plain language
    - Evaluate Abstract
        - Evaluate the quality of generated abstracts by comparing them against the ground truth
4. Additional Claims Generation
    - Identify relevant patents based on similarity to the generated abstract, and use those patents' claims as relevant claims
    - Generate additional claims (to supplement the primary claim) using an LLM (e.g., GPT-4), based on the relevant claims and prompt engineering
    - With prompt engineering, the final prompt included instructions
        - Conform to US patent law
        - Include independent and dependent claims
        - Are ordered correctly and the claims terms are introduced correctly (antecedent basis)
        - Contain only relevant material
    - Evaluate Additional Claims
        - Evaluate the quality of the claims generated by human evaluation

Baseline Model or Comparison

For abstract generation, prompt engineering was performed to improve a baseline prompt, e.g. "Generate a patent abstract from the provided claim." Similar prompt engineering was performed for claims generation.

To evaluate how well the models and prompts are performing, we use a combination of manual human evaluation and automated evaluation.

- Automated evaluation techniques had two methods:
    - Comparing the generated text with the ground truth using cosine similarity and Euclidean distance for a simple way to quantify the results
    - Evaluation using GPT-4. For the query, we prompted the model to be a patent lawyer judging the provided generated abstract on the quality on a score from 1-5 with a rationale to elicit chain-of-thought
- For human review, the coherence of the generated abstracts and claims are qualitatively evaluated for the following criteria:

- Correct patent language, including grammar
- Capturing relevant subject matter
- Uniqueness

For claims generation, we decided the best way to evaluate the results is by using human evaluation. Unlike the abstract, which is based on the claims and does not have much variation, the claims are more freeform and unpredictable. Therefore it made less sense to compare against the ground truth claims and instead evaluate based on the qualitative criteria above.

Quantitative Results

Five abstracts were generated based on primary claims from our November 2023 training data using the engineered prompt using both GPT-4 and Llama 2. The generated abstracts were compared against the ground truth:

	GPT-4			Llama 2		
Title	**Cosine Similarity**	**Euclidean Distance**	**GPT-4 Score**	**Cosine Similarity**	**Euclidean Distance**	**GPT-4 Score**
SYSTEMS, APPARATUS, AND RELATED METHOD FOR REMOTE …	0.780	**6.666**	4	**0.828**	7.153	4
APPARATUS AND METHOD FOR EX VIVO LUNG VENTILATION …	**0.728**	7.296	4	0.608	**5.851**	4
THERAPEUTIC DERIVATIVES OF INTERLEUKIN-22	0.676	**5.557**	3	**0.777**	6.356	3
Systems and Methods for Securing License Plates	**0.608**	5.557	3	0.498	**4.568**	3
METHOD AND SYSTEM FOR INTELLIGENT MAINTENANCE …	**0.805**	7.324	4	0.724	**6.511**	4

Table 2 - Abstract evaluation

The bolded scores indicate that neither GPT-4 nor Llama 2 strongly outperformed the other, but the results from both were still very good.

Qualitative Results

In one sample (Systems and Methods for Securing License Plates), the input primary claim (a ground truth created by an expert human) read:

1. A tracking device for license plates, comprising:
   a global positioning system tracker integrated within an electronic housing; and
   wherein the electronic housing is adapted to attach to a license plate.

The following abstract was generated by GPT-4:

> The abstract discloses a tracking device designed for license plates. The device includes a global positioning system (GPS) tracker nested within an electronic housing. The electronic housing is specially created to enable attachment to a license plate. ...

Both GPT-4 and Llama 2 performed well in generating abstracts that covered the relevant inventive concepts from the primary claim, such as in the sample above, a tracking device, and housing for the license plate. In some instances, however, such as the sample above, unnecessary language was included – the self-reference to "abstract" could be omitted, since it is a redundant waste of word space.

Additional claims, including the following, were generated by GPT-4:

> ...
>
> 6. A tracking device for license plates as claimed in claim 1, wherein the electronic housing includes a power source for providing electrical power to the [GPS] tracker.
>
> ...
>
> 8. A tracking device for license plates as claimed in claim 6, wherein the power source includes a solar panel on an exterior surface of the electronic housing for charging the power source.
>
> ...

The additional claims generated by GPT-4 (less so with Llama 2), were mostly grammatically correct with suitable patent language, such as "wherein" to add additional limitations. In some instances, however, such as the samples above, the additional claims began with a new "A tracking device…" language, and instead should refer directly to claim 1 with the language "The tracking device…".

The claim scope of the additional claims covered relevant inventive features, in the sample above, that started broadly in claim 6 by introducing a power source, and in subsequent claim 8 further narrowed that power source as including a solar panel, illustrating good nesting of the claims.

The additional claims also appeared to be unique, differing in scope from at least the ground truth additional claims.

Discussions and Learnings

Prompt engineering improved results over the baseline prompt by including specific patenting requirements.

The generated additional claims provide a helpful framework for further discussions with inventors, in terms of whether the claims cover new and important features, and if such features would be practical to implement. It is also important to incorporate inventor feedback into the process, since at present most jurisdictions require human contribution for inventorship.

The relevant claims were selected from similar patents and used as example claims for few-shot prompting in generating the additional claims. They were selected based on semantically similar

abstracts as opposed to claims (more specific to the invention), this allowed us to take into account those differences and resulted in a more diverse set of additional claims generated.

Llama 2 generally performed worse than GPT-4, particularly in generating claims. It was not well equipped to handle very large input prompting, such as the few-shot additional claims, and often generated a few good claims before devolving into gibberish. Our theory is this was caused by the smaller model (7B parameters) that we used due to limitations in computer resources. However, as an open-source model, Llama 2 and its larger variants may offer future opportunities for customization or fine-tuning on patent documents, which could improve performance.

Based on human review, the generated claims used the correct patent language and covered relevant scope. Much of the relevant evaluation in the project is qualitative human evaluation, which has limitations, particularly since patent drafting is both an art and a science. It would be useful to survey multiple experts to more objectively examine the results.

To further improve the generated text and reinforce prompt instructions, a second call to GPT-4 was tested, inputting the initially-generated abstract, with a prompt to repeat some of the original instructions – making any necessary corrections and removing words such as "abstract", "invention", and "patent". The purpose of this is that these words are redundant, and unnecessarily take up word count in the resulting abstract. However, on human evaluation, this additional step ended up removing important context in the resulting revised abstract, so we decided to retain the instructions in a single original prompt (both for generating the abstract and generating the additional claims), although this requires some human review since the prompt is not always strictly followed by the GPT-4 model.

Future Work

The current RAG database used is generated from a single week of USPTO data, and will typically include data for 8,000-10,000 patent applications. The database could be expanded to include further data (full year of records), to include more similar patents to be included when generating additional claims.

Another avenue for future exploration is to generate claims for other jurisdictions that have slightly different patent rules, such as in Canada or Europe. These could be generated similarly to the US claims in the existing project, or alternatively, they could be generated directly from the generated US claims.

Individual Contributions

Each team member contributed as follows:[12]

Stephen

- Processing patent data
- RAG indexing functionality
- File management with parquet and CSV
- Llama 2 integration

[12] https://github.com/ece1786-2023/QuickPat

Tamara

- OpenAI API integration
- Abstract generation and claims generation
- Prompt engineering for both prompts
- Exploration to generation for non-US patents
- Exploration into improving the result with sequential prompting to GPT-4

Permissions

Stephen

- permission to post video: no
- permission to post final report: yes
- permission to post source code: yes

Tamara

- permission to post video: no
- permission to post final report: yes
- permission to post source code: yes