Capstone project - CSML1030 - Final Report
NLP for variable extraction and understanding of noisy PDF documents
Project hosted by: Athennian
Duration: 8 weeks
July 2021 - September 2021

Gouri Kulkarni
gouri.kc@gmail.com

Abstract

As part of the machine learning capstone course, I had the opportunity to address a real business problem faced by a document processing firm, turn it into a scalable machine learning task and research possible solutions. The firm provides provides entity management products and services to law firms. Entity minute books have historically been stored by law firms on behalf of their clients in binders following traditional filing practices. The clients are law firms that have migrated from legacy systems to their SaaS solution. The cloud-based work space enables streamlining of day-to-day processes and keep transactions organized. Majority of the work done at law firms is based off reading, assimilating and analyzing information from documents. An entity is a client whose books are managed by the law firm. The size of data handled per entity varies. So does the size of the law firm. The number of entities managed by a law firm can range from 50 to upto 25,000. Contract review is the process of thoroughly reading a contract to understand the content, It is one of the most repetitive and most tedious jobs performed by law firm associates and is also expensive and inefficient use of skills.The same data points are to be identified in every document. A list of data points is shown below. Automation involves developing new workflows and using data driven approaches to read, analyse and interpret documents. Generalizing the process to work on any input document cannot be handled with traditional coding.

Law firms that are on the journey to becoming paperless already have scanned PDF copies of minute books. Manually summarizing and extracting key information from PDFs involves a lot of time and effort. Rule based coding approaches to analysing PDFs can provide insights and reports, but are not scalable.

In this paper, I have explored some state of the art NLP approaches to generalizing the task of extracting key information and building relationships to develop workflows. The best approach would be the one which provides accurate results and satisfies key metrics.Through using Pre Trained models and Fine Tuning, we can achieve results aligned to the use case and domain.

(LateX template borrowed from NeurIPS 2019)

Variable Category	Variable Name
Basic Entity Data	Current Entity Name
Basic Entity Data	Entity Names forming Amalgamation (optional)
Basic Entity Data	Previous Entity Name - (optional)
Basic Entity Data	Entity Type
Basic Entity Data	Corporation Number
Basic Entity Data	Jurisdiction
Basic Entity Data	Entity Formation Date
Basic Entity Data	Registered Office
Share Classes	Share Class Name
Share Classes	Voting Rights (True/False)
Share Classes	Votes per Share
Share Classes	Status
Share Classes	Authorized Amount Unlimited (True/Fase)
Share Classes	Authorized Amount
Share Classes	Share Class Creation Date
Share Classes	Certificate Code
Share Classes	Share Class Termination Date
Articles	Min Directors
Articles	Max Directors
Articles	Unlimited Directors (True/False)
Articles	Transfer Restrictions Texts
Articles	Other Provisions
Articles	Restrictions

The variables of interest , which a law firm associate would repetitively look for in minute books, and which need to be extracted using a generalizable solution

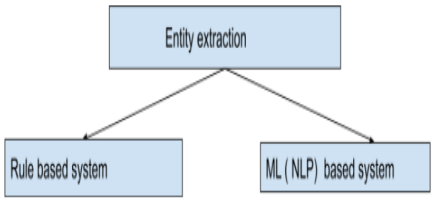
1 Introduction

The amount of information and data collected by lawyers for each client keeps growing over the years. Human capacity being limited when it comes to consuming information, it is paramount to extract only useful information from the unstructured data. The collection and analysis of large amounts of data is becoming increasingly important for lawyers to improve their businesses and processes, such as the introduction of new services. Minute books held by lawyers were historically stored in paper files, and are available in scanned format today. The data from the scanned documents has to be manually entered into the client's entity management system. Scanned documents are not machine readable and require dedicated analysis methods, such as text extraction, object recognition, and natural language processing. PDFs consist of many different document elements such as tables, figures or text sections and many more unknowns depending on the age and source of the document.Pretrained NLP models are widely available to tackle a variety of text data.

A model that has learned from entities recognized from the training data can predict entities on test data. Unlike a rule based approach where adding or modifying rules could need a lot of analysis, a machine learning approach can be scaled and generalized to extract entities from any given document after it has learned from the training data. Automatic named entity extraction is the process of extracting entities from an input document. The model can only get as good as the data it is trained on. The problem of extracting key variables is converted in to a supervised learning problem by using annotated text. In other words, we can call it a data driven approach to recognition and extraction of elements of interest from PDF documents. Approaches for annotating text or generating the training data are discussed, as well as text extraction methods, annotation tools and approaches, their pros and cons. Models go through several iterations before deployment. After deployment too, versions evolve as the data or use case changes. The chosen model can be deployed as a task on the entity management software and used to generate summaries and reports for each entity. Another possible application could be to automatically populate the tabs on the entity management software once documents are received for a new entity. Once into the entity management software, the data can be used to generate digital versions of the same documents using templates.

2 Data

Data collection
Data for this project is readily available in the form of scanned PDFs provided by the clients (law firms).
The average size of each document is about 200 pages and can be more or less depending on the transactions and client history.
In order to make the PDF data retrievable and analyzable, we need to extract relevant document elements as in a formato required by downstream models and store them to a database.
Due to the age of records, documents could have beverage stains, faded sunspots, loosening pages, and wrinkles which makes the documents noisy and make extraction
Text mining
Text mining combines statistics, linguistics and machine learning to create models that learn from training data and can predict results on new information based on previous experience.
A textual datapoint for us is the entire document, as we are trying to extract elements from a minutebook and develop relationships so that they can be put together to tell a story from the transactions discovered through NLP.



Rule based systems are easy to understand and improve, however hard to scale and generalize. ML based systems need training data that is correctly tagged.

Preprocessing
In NLP, the goal of preprocessing is to prepare text by cleaning the data and removing noise.
Some essential steps are lowercasing, stopwords removal, special characters preprocessing and tokenization. In our case, stop word removal and cleaning may erase important information and will not be conducted regardless of the method used.
One preprocessing method is text extraction or conversion.
Text extractors do not capture information from tables and figures accurately. Extraction is prone to errors. Open source text extraction tools are readily available. Features of each tool range from providing metadata to functionalities such as page splitting, ordering which can be used to standardize the PDFs for further processing and a pipeline can be developed.
Another method is to convert the PDF documents into images and use OCR for extraction of key elements. OCR is the process of getting type or handwritten documents into a digitized format. OCR makes previously static content editable, searchable and much easier to share.
The result of pre processing will be the input for future algorithms.
Text embedding /feature engineering
The pre processing output can only be used by algorithms if it is in the proper numerical form and the features are useful enough. Useful features have to be developed. Of the various frameworks available for deep learning, Pytorch by Facebook and TensorFlow by Google are most popular and host their hub of pre trained models ready for fine tuning and solving downstream tasks. Feature engineering is the most time consuming task and iterative in the ML life cycle.

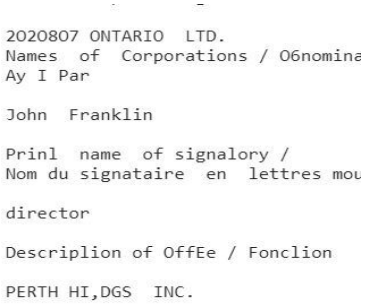


Figure 1: *This figure shows results of text extraction using textract. Overall , the quality of text extracted using various tools has been low . Certain text is not recognized by the extractors.*

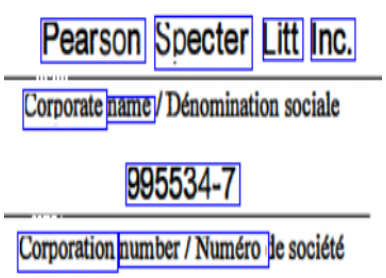


Figure 2: *This figure shows results of text extraction using OCR bounding boxes.*

3 Training data

Turning a problem into a machine learning task involves transforming the available data into a format baseline or proposed models can use to make predictions.
Translating the problem into and NLP task that would possibly solve the problem should be addressed with domain experts and UX.
How should the final solution look like?
Will framing it as a specific NLP task provide a better solution?
Information retrieval, Named entity recognition, Relation extraction, Document classification, Annotation, Topic modelling, Keyword extraction, Question answering, Question Generation are some translatable NLP tasks.
Annotation is the process of generating the training data suitable to the selected task.
Today, there are several online annotation tools both open source and commercial available to label text to use as training data for downstream models. Through annotation, we extract key features out of the text. Once data has been annotated, it can be split into train,test and validation sets. Some available annotation tools are : Scale, Label, Prodigy, Mechanical Turk, CVAT, Doccano, tagtog, UBIAI.
We have some documents and want to extract information. How do we decide which NLP annotation tool or library to use? Comparing NLP tools is not a straightforward exercise. Each tool may extract a different set of entity types, classify the same entity a little differently, or provide output in different formats, thus preventing direct comparison.
By annotating preprocessed text, we are labeling it so that the chosen downstream model can learn from it. Open source and commercial annotation tools are available that can output annotations for use in formats required by various NLP libraries.
Before proceeding with annotation, a gold standard needs to be set by domain experts , who could be the end user/s of the SaaS software for which the model is being developed.

Similar to how medical images are labelled by several doctors prior to the final diagnosis, document can be annotated by several annotators at the same time and inter annotator tools can give a clear sense of which annotator is doing a right job with the data. This can help prevent bias from creeping into the training data and also help develop the workflow and linking of entities. Open source tools may or may not have that feature and is something that should be considered when evaluating annotators.
Some points to consider when choosing an annotator tool are :
-annotation on the span, character or OCR bounding box
-format of the output suitable to use in various libraries, APIs and schemes such as Amazon Comprehend, Spacy, OCR, Stanford CORE NLP, IOB
-the function of entity linking
-the output from the annotator tools must be general enough like json format so that it can be incorporated into any other task
-how will the problem be framed into an NLP task - will a question answering model be suitable ?
-features available to streamline the process and help the team increase efficiency
Manual annotators will likely make numerous mistakes. Most annotation firms guarantee a maximum error percentage. The annotation scheme may need to be defined and redefined over and over again in the ML lifecycle, which makes selecting a third party annotation service difficult. Problems and errors in annotation are easier to detect and fix in house rather than when outsourced. It is a tedious and costly process.
In house annotation for developing training data to use in future modeling projects is important, as model results can only be as good as the data used, and KPI definitions used might change. Further, model versions evolve and different ML frameworks require training data in specific defined formats.

4 Preliminary work, Baseline model

A new paradigm in natural language processing (NLP) is to select a pre-trained model and then fine-tune the model with new data similar to the specific task.
Research started with first extracting text from the PDFs using open source text extraction tools and analyzing the quality of output, usage features and functions, using nltk and spaCy to visualize the sparsity of text and frequency of words.
Having identified the NLP task (NER) spaCy was the first preference in developing the baseline. It is designed specifically for production use and helps build applications that process and understand large volumes of text using several pretrained models with transformer capability. It can be used to build information extraction or natural language understanding systems. It has components for named entity recognition. The spaCy API has pipelines, dependency parsers and matchers that can be used to develop annotated training data suited to the downstream model. Annotations can be developed on text input.

spaCy can be used for natural language understanding tasks. It is production ready. spaCy requires raw text as input, and has great linguistic features.

A baseline approach was taken using spaCy ver 2.2.4 and the small English model which is optimized for CPU. It does not use word vectors.
The pretrained model has an F- score of 84 for NER. Our text is predominantly English but the text is specific to the legal or business domain.
The main task of extracting key variables was converted into a NER task.spaCy's NER model provides incremental parsing with bloom embeddings and residual CNNs.
NER is the foundational task over which other methods (for relationship building) can be developed.The text being sparse and not consisting of complete sentences, POS tagging is not required. Default NER labels do not suit our task, and custom labels need to be defined. spaCy also has a medium and large model which both

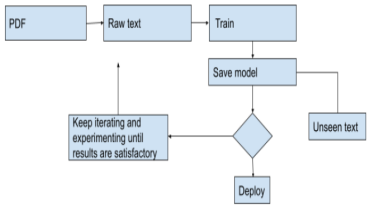


Figure 3: *The iterative process*

use word vectors. However, they are still optimized for CPU. The larger models require more space and time to implement. The models are also application based as opposed to research based.
The transformer pipeline is roberta based and has a higher F1 score (90).

The goal of this initial exercise was to determine if spaCy is appropriate for our use case and explore possible use to generate annotated training data for downstream tasks. The baseline model was developed to extract one variable using the small English model. The first step was to parse the input text and create patterns using PhraseMatcher. Ideally, if the solution worked, an entire list of phrases could have been provided in a list. The PhraseMatcher was then used to find the terms in the text, create matched spans and append to a list which eventually make up the training data or annotations. A NER pipeline with the annotated label was added to the NLP object created using sample minute book text. Through this exercise and iterations, it was found that text extractors such as PyPDF2 , fitz, textract do not extract desired text from forms.
Testing the model on unseen data worked. However, it was noticed that the text extractors were unable to extract certain key variables from the documents. Text extractors are generally accurate, but the PDFs being noisy, none were able to give good results.
In NER, Precision is the fraction of retrieved documents relevant to the query.
Recall is the fraction of relevant documents that are successfully retrieved.
Having developed a base model, and tested it on new data, the model was evaluated using F1 score, which is the harmonic mean of precision and recall and is more appropriate to the NER case.

The approach can be used to create annotated training data in spaCy format for use in further entity/variable extraction. spaCy can also be used to develop applications and automate the generation of annotations. Due to CPU restrictions and poor quality of extracted text, this approach was not pursued.

It is possible to use word vectors (transfer learning) to import knowledge into the current pipeline so that the model is able to generalize better from the annotated examples. To use transfer learning, we need a few annotated examples of what we are trying the predict.

There are several open source and commercial annotation tools available today. Labeling text data accurately is crucial to solving supervised NLP problems. By labelling training data, we will have have converted our problem of extracting entities from unstructured text into a supervised learning problem.

5 Proposed method -1

Pretrained model on IIT-CDIP and fine tuned on form, receipt and image datasets
ML framework and algorithm: PyTorch 1.9.0+cu102, LayoutLMTokenClassifier

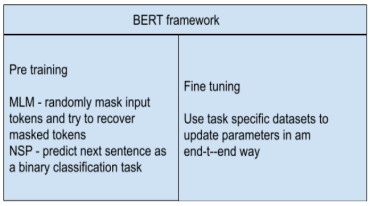


Figure 4: *The BERT framework*

pre trained on IIT-CDIP and fine tuned on form, receipt and image datasets [5]
Pretrained models achieve state of the art results on text, however they focus on text level manipulation and neglect layout and document style. In our use case, layout, sequence and order of text are critical in developing relationships. It is possible to develop techniques for automatically reading, understanding and analyzing business documents.
The proposed method is based on a recent paper[https://arxiv.org/abs/1912.13318], Pretraining of Text and Layout for Document Image Understanding.
Although several models have been developed for table detection in PDFs (CNNs), improving the accuracy of document layout analysis (Faster R-CNN, Mask-R-CNN), extracting semantic structures from document images (pre trained NLP models), combining textual and visual information for information extraction (Graph Convolutional Networks) , they do have some limitations.
1. relying on a few human labeling samples without fully exploring the possibility of using large-scale unlabeled training data.
2. they use either pre trained CV models or NLP models but not a combination of both.
LayoutLM is inspired by BERT and adds two types of embedding over the text and position embeddings
1. 2d position embedding which captures the relationship among tokens
2. image embedding which captures appearance features.
The model itself is pre trained on over 6 million scanned documents (IIT-CDIP).

BERT like models take text as input. The accuracy of the text extracted from the PDF minute books is not great due to the noisiness of the content. A model can only get as good as the data it is trained on. This is the reason why the baseline model developed using spaCy pre trained language models was not pursued.
LayoutLM uses two more types of features over BERT which are useful to us in understanding minute books. Layout information is treated as a parsing problem.
1.Document layout information- the relative position of words in a document contribute a lot to the semantic representation. For example, the corporation name is more likely to appear to the right or below the key "Name of Corporation".
2.Visual information- the whole image can indicate the document layout. Identifying styles such as bold, underline, italics could bring more meaning to plain text
Model Architecture
Using BERT architecture as the backbone , which has position embedding, this model adds two new input embeddings : a 2D position embedding and an image embedding.
Position embedding: model word position in a sequence (possible in BERT)
2D position embedding: model the relative spatial position in a document by considering the document page as a coordinate system with the top left origin. The model has four position embedding layers and two embedding lookup tables .
Image embedding: The image embedding layer represents image features in language representation. With bounding boxes, the image is split into several pieces. Image pieces from a Faster R-CNN are used as the token image embeddings.

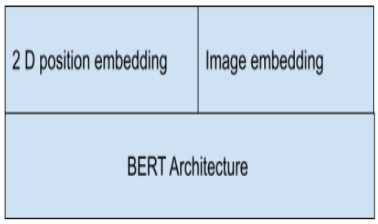


Figure 5: *LayoutLM architecture*

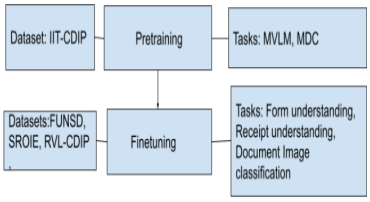


Figure 6: *The model has learned from Pretraining and Finetuning tasks*
The LayoutLM model was compared with two SOTA pretrained models BERT and RoBERTa. It outperforms these pre training baselines.

The downloaded model is pre-trained on two tasks:
1. Masked Visual Language Model
2. Multi label Document Classification.
MVLM learns the language representation with clues of 2D position embeddings and text embeddings. Document tags from the training dataset IIT-CDIP were used to supervise the pre training process so that the model can cluster the knowledge from different domains and generate better document level clssification (MDC) .
Fine tuning tasks : The pretrained model is fine tuned on three document image understanding tasks - form understanding, receipt understanding and document image classification. The form and receipt understanding tasks the BIESO tag scheme is used. Predictions are made for each token using sequence labelling to detect each type of entity in the dataset.

Fine tuning tasks : The pretrained model is fine tuned on three document image understanding tasks - form understanding, receipt understanding and document image classification. Form and receipt understanding tasks use the BIESO tag scheme. Predictions are made for each token using sequence labelling to detect each type of entity in the dataset.

Experiments conducted have also shown that as the model is trained on more epochs and larger data sizes, the precision, recall and F1 scores increase. We can conclude that pre training technique can be researched and used to achieve our goal of extracting variables of interest and developing relationships for downstream automation tasks.
As discussed, the annotation strategy depends on the chosen NLP task

6 Proposed method -2

Many specialized domains remain untouched by deep learning, as large labeled datasets require expensive expert annotators. Labelled data is a bottleneck within the legal domain. The Contract Understanding Atticus Dataset (CUAD) was created with dozens of legal experts from The Atticus Project and consists of over 13,000 annotations 510 commercial legal contracts. It identifies 41 types of

Modality	Model	Precision	Recall	F1	#Parameters
Text only	$BERT_{BASE}$	0.5469	0.671	0.6026	110M
	$RoBERTa_{BASE}$	0.6349	0.6975	0.6648	125M
	$BERT_{LARGE}$	0.6113	0.7085	0.6563	340M
	$RoBERTa_{LARGE}$	0.678	0.7391	0.7072	355M
Text + Layout MVLM	$LayoutLM_{BASE}$ (500K, 6 epochs)	0.665	0.7355	0.6985	113M
	$LayoutLM_{BASE}$ (1M, 6 epochs)	0.6909	0.7735	0.7299	113M
	$LayoutLM_{BASE}$ (2M, 6 epochs)	0.7377	0.782	0.7592	113M
	$LayoutLM_{BASE}$ (11M, 2 epochs)	0.7597	0.8155	0.7866	113M
Text + Layout MVLM+MDC	$LayoutLM_{BASE}$ (1M, 6 epochs)	0.7076	0.7695	0.7372	113M
	$LayoutLM_{BASE}$ (11M, 1 epoch)	0.7194	0.7780	0.7475	113M
Text + Layout MVLM	$LayoutLM_{LARGE}$ (1M, 6 epochs)	0.7171	0.805	0.7585	343M
	$LayoutLM_{LARGE}$ (11M, 1 epoch)	0.7536	0.806	0.7789	343M
Text + Layout + Image MVLM	$LayoutLM_{BASE}$ (1M, 6 epochs)	0.7101	0.7815	0.7441	160M
	$LayoutLM_{BASE}$ (11M, 2 epochs)	**0.7677**	**0.8195**	**0.7927**	160M

Table 1: Model accuracy (Precision, Recall, F1) on the FUNSD dataset

Figure 7: *Model accuracy on FUNSD dataset*

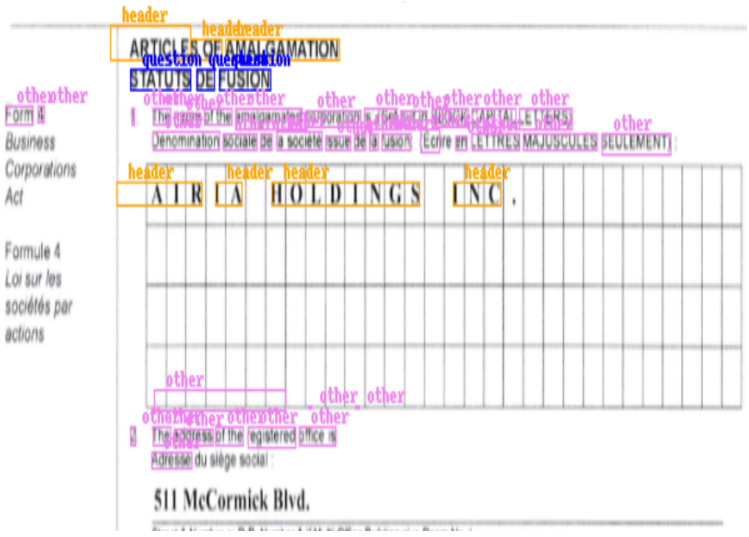


Figure 8: *Inference on sample from a minute book using current annotations. Replacing the FUNSD dataset with out own data and labelled annotations suited to our use case can yield desired results.*

legal clauses that are considered important in contact review in connection with a corporate transaction, including mergers acquisitions, etc. It highlights salient portions of a contract that are important for a human to review.
Requirements: HuggingFace Transformers library , PyTorch
The research is outlined in the paper [https://arxiv.org/abs/2103.06268]
The model is based on the AutoModelForQuestionAnswering class which is trained specifically for question-answering tasks. This can be applied to our use case by making the model identify certain parts of the document by asking it a question (e.g. "What is the Corporation name?). Questions can be framed by domain experts by working on the list of key variables.
Pretrained models that can be used are ROBERTA-base (100M parameters) , ROBERTA-large (300 M parameters) , or DEBERTA-V2-xlarge (900M parameters). Documents and associated queries must be in json format.
The way QA models work is that question and document are concatenated (separated by a special token), tokenized (i.e preparing the input so the model can understand it), and then fed into the model. The model will then provide two outputs: the start logits and the end logits. The start logits describe the probability for each word in the string to be the beginning of the answer for the question. Similarly, the end logits describe the probabilities for each word to be the end of the answer. In order to get the best prediction from the model, our task is to pick the start and end tokens with the highest probabilities.
This method has a limitation. The model has a 512 token limit (on the tokens generated by the tokenizer). Through initial EDA on the minute books, we found that the density of words in each document is sparse, as opposed to contracts. In order to generalize the model over any input document, the limitation needs to be addressed. Each document has to be split up into several parts of 512 words before developing features.

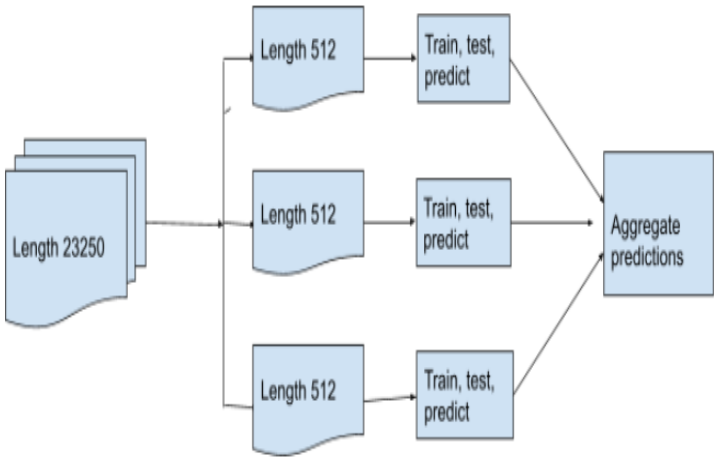


Figure 9: *Splitting documents into smaller sized documents that are within the 512 limit and aggregating individual model results*

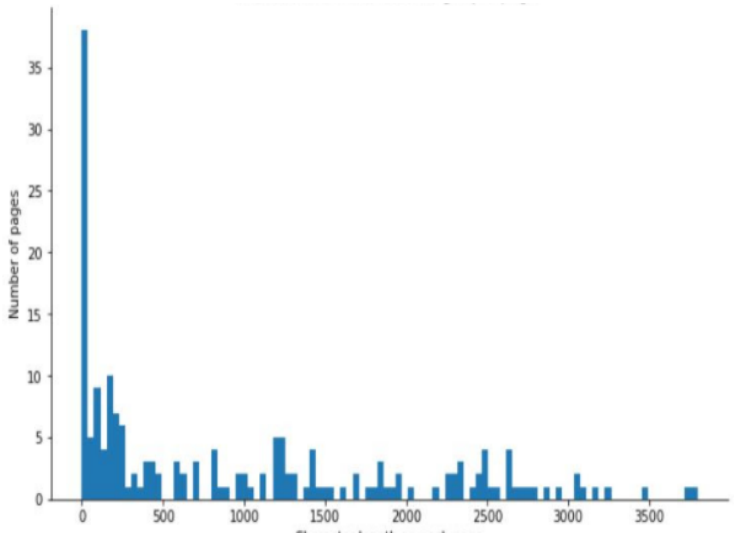


Figure 10: *Distribution of character length per page in a sample minute book. The average size of a minute book is 200 pages. Most pages do not contain useful information.*

In order to develop a strategy for splitting the document, input from domain experts and end users and the current human approach to actually read an analysing a minute book will be required.

Given a business problem ,Assist the firm with a business problem

7 Discussion and Future Work

Through weekly discovery sprints with the client, the given challenge was framed as a machine learning problem and possible NLP tasks and requirements were identified.
Researching state of the art approaches to generalizing the task of extracting key information and relationships helped the firm get a head start into developing strategies for product enhancement.

The proposed methods can work favourably with proper annotations prepared for the use case.
As is obvious, the output from the pre trained and fine tuned models depends on the strategy used to label the training data.
Developing custom metrics for the use case , and measuring the success of different models and repeating with variations is important.
Some points to consider:
1.Does the model correctly answer questions based on the text?
2.Is it cost effective?
3.Metrics such as accuracy, effort, cost of generating training data should be discussed with domain experts.
4.In setting up metrics, it is important to have a good annotation pipeline and error analysis pipeline. Annotation pipelines must include gold standards and inter annotator agreements, which require input from domain knowledge experts.
5.Chosen metrics will differ depending on the input data (raw text vs images).
6.Annotation tools charge users based on the number of documents and number of variables to be extracted. Due to the varying sizes of minute books, it is hard to estimate annotation costs. Plus, there is manual work involved. A model chosen today may not serve the purpose tomorrow as desired out requirements change. Developing annotations in house can help with defining custom metrics and changing links and relationships as the system workflow evolves.
7.The UX and user value of the model.
8.In the context of scanned minute books, the time to analytics refers to the total time that elapses between identifying pages that contain relevant information, collection of data ,storing it ,preparing it for machine learning and extracting different elements.
9.Regardless of the method chosen, identifying failures is the first step in figuring out ways to improve model performance which could be adding more training data, changing the pipeline steps, or fixing annotations.
10.Developing models to solve problems is an iterative process.

Machine Learning models can be incorporated into the current SaaS Entity Management System software to solve many problems and also automate workflows at every level.

NLP frameworks and model architectures are evolving at a fast pace.
Through this project, I have attempted to conduct initial research into the problem and possible use of ML to achieve the desired output.

I would like to thank Prof. Saber Amini (York University) for his supervision, support and mentorship during the Capstone Project and Ariel Browne, Ari Tegshee, Kenny Laurin and Grant Taylor (Athennian) for hosting the Capstone Project, providing invaluable insights and data. It was my pleasure working with the team to collectively brainstorm and propose solutions. Future work on the proposed solutions and additional research will be hosted on github.