A NLP Approach to Understanding Patent Acceptance Criteria

Stanford CS224N Custom Project

Ryan Kearns
Department of Computer Science
Stanford University
kearns@stanford.edu

Sauren Khosla
Department of Computer Science
Stanford University
sauren@stanford.edu

Benjamin Wittenbrink
Department of Computer Science
Stanford University
witten@stanford.edu

1 Key Information to include

- TA mentor: Lucia Zheng
- External collaborators (if no, indicate "No"): No
- External mentor (if no, indicate "No"): Mirac Suzgun <msuzgun@stanford.edu>
- Sharing project (if no, indicate "No"): No

Abstract

Patent applications and acceptances are a useful domain for assessing the state of innovation across various fields, including biomedical sciences, artificial intelligence, software services, and more. The number of patents filed per year has nearly doubled since 2000 with over 650,000 patent filed in the 2020 fiscal year. Until now, no large-scale corpus of patent filings exists for ML and NLP practitioners to leverage. The Harvard USPTO Patent Dataset (HUPD), consisting of over 4.5 million English-language utility patent applications filed between 2004 and 2018 is the first example of such a corpus. Unlike other, smaller but similar corpora, this dataset contains the inventor-submitted versions of patent applications as opposed to the final versions of granted patents, allowing for the usage of NLP techniques at the time of filing. Taking advantage of this rich data, we vary the metadata inputs to a number of NLP models to conduct an ablation study on the binary classification of filed patents (i.e. acceptance or rejection). Our best metadata-augmented model achieves 63.32% binary classification accuracy, outperforming the best language models from the HUPD paper [1] as well as our baseline models. Yet, for some text fields our best model still cannot outperform bag-of-words models, likely due to specific qualitative linguistic features of these fields.

2 Introduction

Patents are essential for assessing the level of technological innovation across and within modern fields, providing a valuable source of information for evaluating growth, activity, and transformation in emerging and traditional industries. The number of patents filed per year to the United States Patent and Trademark Office (USPT) has nearly doubled since 2000, with over 650,000 patents filed in the 2020 fiscal year alone, making the study of patents more important than ever before.

Yet, the absence of a “large-scale, well-structured, and distilled patent dataset" has been a significant impediment to this goal, making computational, quantitative research approaches to understanding patent acceptances incredibly difficult. Consequently, the machine learning (ML) and natural language processing literature (NLP) on this topic is very sparse. However, the introduction of the Harvard USPTO Patent Dataset (HUPD) addresses this deficit, providing comprehensive information on more than 4.5 million patent application documents from 2004 to 2018, substantially more than the pre-existing patent datasets (see Appendix Figure 6 for an overview of these) [1].

The HUPD provides an all-encompassing and holistic view of patent applications at the time of filing through their lifespan, allowing for flexibility and control in developing ML and NLP models. In particular, unlike other patent datasets, the HUPD has access to both patents that were accepted and rejected alike in the condition in which they were first filed, thereby enabling the binary classification analysis of patent decisions. Solving such a problem would provide insight into the importance of various components and data fields when filing a patent application. Moreover, the dataset contains critical information that is otherwise not easily available, such as filing data, IPC codes, examiner information, and more. This metadata provides an opportunity to enhance ML and NLP models alike as they feed off of the additional data; further, this allows for the visualization and analysis of acceptances and rejections based on completely new data, providing insight into the relative importance of each field.

3 Related Work

Prior to the introduction of the HUPD, the existing patent datasets in the NLP literature were designed for two primary tasks: patent subject classification (see Larkey, 1999; Chu et al. 2018; Devlin et al., 2019; Zaheer et al. 2020) and patent summarization (see Sharma et al. 2019). For example, the dataset introduced in Sharma et al. (2019) [2], is a collection of all the successful patents stored within the Google Patents Public Dataset. This dataset contains strictly accepted patents and contains fewer metadata and text fields than the HUPD. Indeed, to the best of our knowledge (and the authors of the HUPD study), the HUPD was the first paper within NLP to “introduce the patent decision classification task" and discuss the patterns in patent decisions from a textual perspective

[1]. Consequently, this work draws inspiration from and works to extend the original HUPD study. The authors similarly tackle the binary classification problem and run baseline tests with a variety of different methods, including traditional statistical methods as well as state-of-the-art Transformer models.

In order to integrate metadata information into our neural model, we looked within the NLP literature for similar architectures. Ostendorff et al. 2019 proposes a way to enrich BERT hidden states with knowledge graph embeddings for classifying book genres [3]. The authors create an intermediate representation formed by contatenating BERT's final hidden state output with knowledge graph embeddings. They then pass this representation through a projection layer and a 2-layer multilayer perceptron (MLP) before a prediction head. Their results show a general improvement in performance over baseline BERT sequence classification.

4 Approach

4.1 Main Approach

Our main approach in this paper uses a natural language model to encode the patent text stream and combines it with a neural network embedding of the patent metadata. We then compare this metadata-augmentation approach to the base language models used in the original HUPD paper [1].

4.1.1 Natural Language Model

For our natural language model, we utilize pre-trained DistilBERT (`distilbert-base-uncased`) and RoBERTa (`roberta-base`) models available through HuggingFace [1]. We then finetune these models to fit our task. While the reference paper only focused on the abstract and the set of claims, we make use of the additional rich text fields, such as description and summary.

We make use of the BERT class of models for their pre-trained robustness and the textual nature of the various data fields for our classification task. They are particularly good at tasks in which word context is important, returning different embeddings for words depending on their context, which is critical for the dissection of patent applications. In particular, DistilBERT, while maintaining approximately 95% of standard BERT's performance, runs 60% faster using 40% less parameters, leaving us with more time for training more models in the ablation study. Finally, RoBERTa, which is more robust than BERT due to making use of more training data, introduces dynamic masking and removes next sentence prediction (NSP) from BERT's pre-training to improve the training procedure. This, in theory, increases the accuracy with which we predict patent acceptances.

However, since some of the text fields exceed the maximum input sequence length for the BERT family of language models of 512 tokens – for instance, the claims field has an average of 1,272 tokens – we also implement a Longformer model. The Longformer was introduced in Beltagy et al., 2020 as a way to apply Transformer architectures to long text sequences [4]. Transformer self-attention in models like BERT or GPT scales quadratically with sequence length, making it computationally infeasible for long sequences. The Longformer, by contrast, implements local windowed attention together with global attention on pre-selected indices. Since the number of selected indices is typically signicantly less than the sequence length, the overall complexity is still $O(n)$. HuggingFace provides a Longformer implementation from Allen AI that supports a maximum sequence length of 4,096 tokens.[2]

4.1.2 Metadata Augmentation

Each patent comes with 20 metadata fields in addition to the actual text of the patent. Many of these fields are irrelevant for learning as they're unique to each individual patent. Other fields may carry signal, but we initially did not know which would be helpful for classification. In order to select relevant metadata fields, we did data exploration on the distributions of accepted and rejected patents. Specifically, we observed that acceptance criteria varies substantially over time and by field as well

[1]For DistilBERT, see `https://huggingface.co/docs/transformers/model_doc/distilbert`; for RoBERTa, see `https://huggingface.co/docs/transformers/model_doc/roberta`.

[2]`https://huggingface.co/allenai/longformer-base-4096`

as based on the chosen patent examiner – see Figure 1 for acceptances rates broken down by year and examiner – we worked to introduce these additional variables in our model.

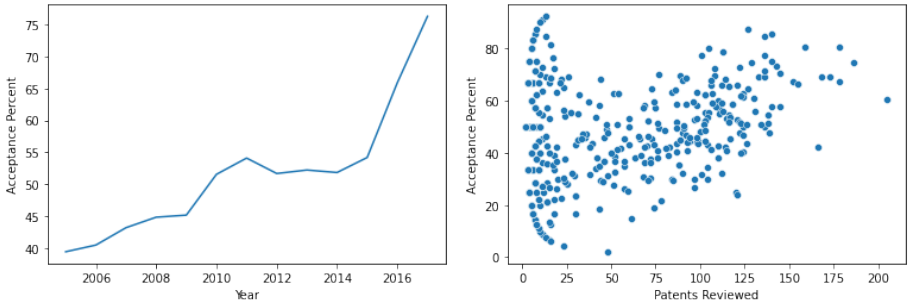

Figure 1: Acceptance rates by year and by examiner. These figures are created using the balanced sample as described below. The acceptance percent is defined as the number of accepted patents over the sum of accepted and rejected patents, times 100.

Inspired by [3], our metadata-augmented models learn 128-dimensional embeddings for each field of interest. Embeddings are fit from scratch using the same training corpus as is used for finetuning. We concatenate these embedding outputs to the final hidden state output of a language model to form an intermediate representation, which has shape $(\text{batch size}, 2 * 128 + 768 = 1024)$. Next, we learn a 2-layer MLP with 728-dimensional hidden linear layers, which takes the intermediate representation as input. Finally, a 2-class prediction head outputs the probability that the patent was accepted.

In addition to the learned dense representations, we calculate a target mean-imputation encoding for the examiners, that is, given the categorical examiner x and target decision y variables, we replace each distinct examiner j in x with its conditional mean of y: $\bar{y}_j = \sum_{i=1}^{n} \frac{1_{x_i=j} y_i}{1_{x_i=j}}$. Examiners with fewer than 10 patents are replaced with the overall, unconditional mean. To avoid data leakage, all estimates are obtained "out-of-fold", i.e. for a given project in fold 1, the mean encoding is calculated based on the average of the project outcome in folds 2-5.

To understand the effect of metadata augmententation, we perform an ablation study where we iteratively added embeddings for patent examiners and patent filing dates by year. Due to compute constraints, we used only DistilBERT for our language model when ablation testing. From the HUPD paper we note that DistilBERT, BERT, and RoBERTa perform within a percentage-point accuracy of one another on the classification task [1]. It remains as future work to see if BERT or RoBERTa could better leverage the metadata-enhanced architecture to improve performance on the task. Figure 2 shows a diagram of the metadata-enhanced model architecture.

4.2 Baseline

We utilize two sources for our baseline. First, we compare the performance of our preferred models against naive, non-natural language classification methods, such as logistic regression or Naive Bayes. In the reference paper, these methods, in particular Bernoulli Naive Bayes, actually outperformed the natural language models on the claims text stream.

Second, we treat the best performing models from the original HUPD paper as baseline results to improve upon. Specifically, the authors focused on the abstract and claims text streams, where they obtained their best results from a fine-tuned DistilBERT (61.83% accuracy) and a Bernoulli Naive Bayes model (64.37%).

5 Experiments

5.1 Data

We are using the Harvard University Patent Dataset (HUPD), which contains approximately 4.5 million patent applications [1]. A patent formatted to the USPTO specifications contains the following sections:

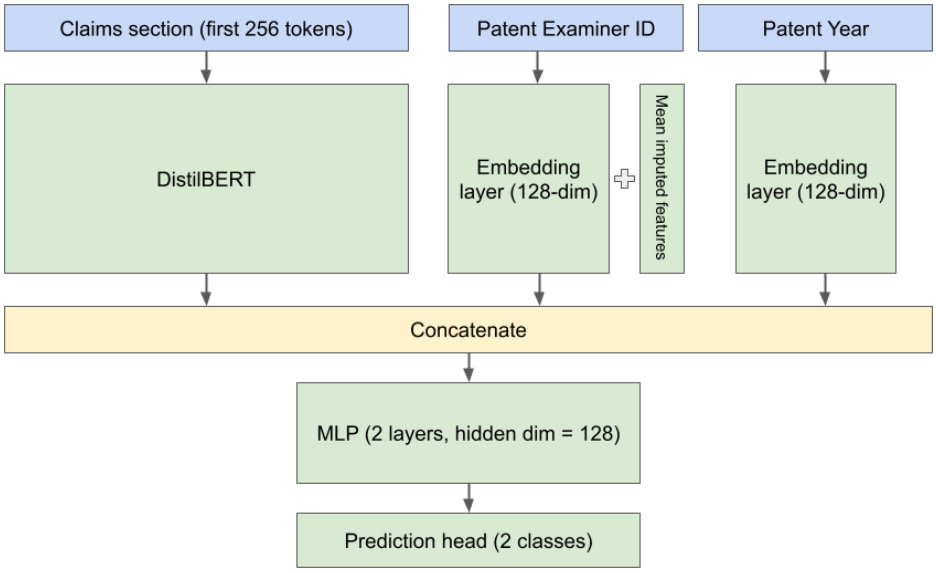

Figure 2: Architecture of our Metadata-Augmented Model.

1. **Abstract**: A brief summary of the invention in broad terms, not exceeding 150 words.
2. **Background**: Describes the general context in which the invention takes place.
3. **Description**: The main technical disclosure of the patent, describing the invention in detail.
4. **Summary**: A condensed version of the description field.
5. **Claims**: The most important part of the patent from a legal standpoint, which sets out the limits of intellectual property claims for the invention.

For each application, the entire text is available, broken into abstract, background, claims, description, and summary fields. Of these, the main text fields are the abstract (average 132 tokens), the claims (1,272), the background (627), the summary (918), and the description (11,856). Metadata for each application also includes the application invention type, the application examiner's full name, the International Patent Classification (IPC) code, which classifies the application according to the technical fields to which it pertains. Of particular interest to us is the binary version of the *decision* class, which has the continuation application phase removed as in the original paper. The USPTO permits a lengthy appeals process where patents may be accepted after an initial rejection; for simplicity we ignore these cases. See Figure 3 for a visual representation of these fields.

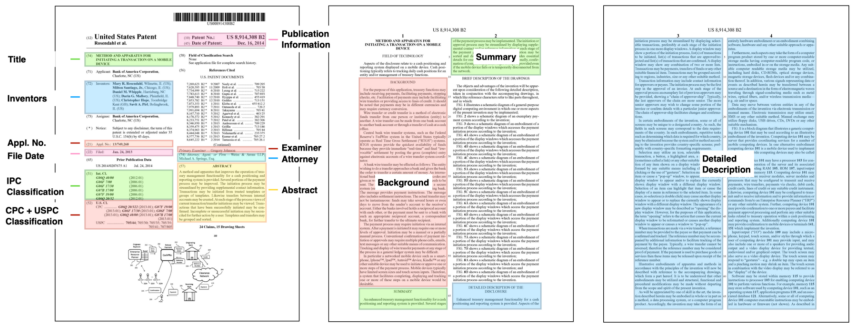

Figure 3: Example of a patent application in our dataset.

Given that the full panel of data is approximately 370 GB, far exceeding the available disk space on our virtual machines, in addition to CPU and GPU-RAM limitations, we selected a subset of patent

applications according to the International Patent Classification (IPC) category code “G06F-17/30". This is the largest category code in the data, including about 30,000 patents from 2005 to 2017, representing all patents for inventions relating to “information retrieval; database structures therefore" [5]. We chose this category as it was the largest and the acceptance dynamics match that of the greater sample well, as can be seen in Figure 4.

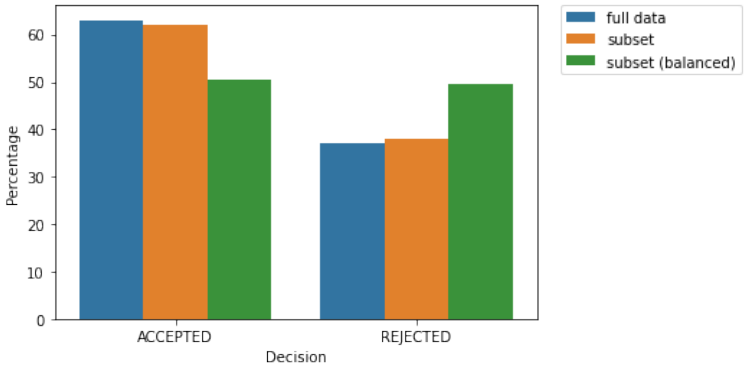

Figure 4: Acceptance rates by sample.

As we are completing a binary classification task, we proceed with our analysis on a perfectly balanced version of this subset. This allows us to justifiably compare our accuracy to a 50% chance baseline, as well as to the models from the HUPD paper, where the authors also balanced their data [1]. Thus, this final sample includes approximately 13,000 accepted and rejected claims, respectively.

5.2 Evaluation method

For training, we use binary cross entropy loss with mean reduction, defined as

$$l(x, y) = \sum_{n=1}^{N} \frac{1}{\sum_{n=1}^{N} w_{y_n}} l_n$$

such that l_n is given by

$$l_n = -w_{y_n} \log \frac{\exp(x_{n,y_n})}{\exp(x_{n,0}) + \exp(x_{n,1})}$$

where x is the input, y is the target, w is the weight, and N represents the batch size. After balancing our data sample above, we do not impose any weighting, thus $w = 1$ for all n. The `pytorch` implementation of this loss is used.[3] We use binary accuracy to evaluate our predictions. In addition, we also calculate a confusion matrix post-prediction.

5.3 Experimental details

We run our experiments using the Azure setup provided by the CS 224N course, which means a Standard NC6s v3 machine (6 vCPUs, 112 GB RAM, running one NVIDIA Tesla V100 GPU with 16 GB of memory). We jointly develop model code using a Github repository forked from the HUPD paper.

For our language models, we adopted open-source code from HuggingFace with the following high-level specifications:

1. RoBERTa: 12 layers, 768 hidden size; 12 heads; 125 million parameters;
2. DistilBERT: 6 layers; 768 hidden size; 12 heads; 66 million parameters.

Other hyperparameters will be the HuggingFace defaults unless specified otherwise. For our Naive Bayes bag-of-words model we adapted the Bernoulli Naive Bayes implementation from `sklearn`,[4]

[3] See `https://pytorch.org/docs/stable/generated/torch.nn.CrossEntropyLoss.html`.
[4] `https://scikit-learn.org/stable/modules/naive_bayes.html`

with $\alpha = 1$. We trained all language models with a batch size of 64, except where CUDA memory constrained us – we point these cases out explicitly in our results tables. We fine-tuned for 10 epochs. We used `AdamW` to set the learning rate,[5] and reduced the default parameters for more stable updates (`lr=2e-5`, `eps=1e-8`). Vocabulary size was limited to 10,000 for all models, and tokenizer max length was 256 except where stated otherwise.

5.4 Results

Table 1 compares our best results with baseline models from the HUPD paper (applied on our sample).

Table 1: Comparison of our best models to the baselines from [1].

	Model	**Batch Size**	**Validation Accuracy (%)**	
			Abstract	**Claims**
Baseline	DistilBERT	64	60.82	61.83
	RoBERTa	32	60.74	61.76
	Logistic Regression	64	59.31	59.31
	Bernoulli Naive Bayes	1	61.54	**64.37**
Metadata-Augmented	DistilBERT	64	**63.08**	63.32

Table 2 shows the results of ablation studies we conducted to understand the effect of metadata augmentation. As we expected, adding metadata improves on the performance of language models alone. The more metadata we added, the better the models did, with marginally diminishing returns. We observe that our feature engineering efforts to encode the imputed mean of each field also brought small improvement.

Regrettably, our Longformer implementation was unable to stably learn to a high accuracy, so we omit the results. Due to CUDA memory constraints, we had to reduce our batch size to 8 in order to train. This resulted in an infeasible learning setting given our limited Azure credits. Future work should explore the performance of the Longformer with sufficient computing resources.

Finally, Table 3 compares the results of our best model, DistilBERT with full metadata-augmentation, on the five main text inputs against Naive Bayes. The model performs best on all fields other than the claims section. Once again, given the long nature of these sections – particularly the description – we would have wished to take advantage of the Longformer implementation here, but ran into memory issues. In addition, we were unable to train the metadata-augmented DistilBERT with a batch size of 64 and a maximum input sequence of 512. As a result, we were forced to truncate these input streams after the 256th unique token, discarding valuable information. In comparison, as Naive Bayes was implemented as a bag-of-words model with a 10,000 vocabulary size, it was able to ingest much more of the text than the language models.

Table 2: Ablation studies to understand the effect of metadata augmentation. Every model was run with a max input length of 256 for 10 epochs. The best validation accuracy for each model is reported.

NLP Model	**Metadata**	**Validation Accuracy (%)**	
		Abstract	**Claims**
DistilBERT	None	60.82	61.83
	Examiner ID	62.01	62.38
	Examiner ID + Year	62.11	63.30
	Examiner ID + Year + Imputed Mean	**63.08**	**63.32**
None	Examiner ID + Year	57.59	
None	Examiner ID + Year + Imputed Mean	58.32	

[5] See `https://huggingface.co/docs/transformers/main_classes/optimizer_schedules#transformers.AdamW`

Table 3: Performance of main model across different input text streams compared to Naive Bayes.

Model	Validation Accuracy (%)				
	Abstract	**Claims**	**Description**	**Summary**	**Background**
DistilBERT Examiner ID + Year + Imputed Mean	**63.08**	63.32	**62.46**	**62.28**	**62.34**
Bernoulli Naive Bayes	61.54	**64.37**	61.05	61.49	59.31

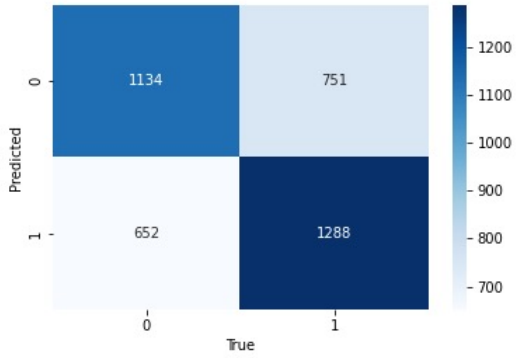

Figure 5: Confusion matrix for our best performing model, with 63.32% classification accuracy.

6 Analysis

Figure 5 shows the confusion matrix of our best model. We note that misclassification is somewhat balanced, with a comparable number of false positives (652) as false negatives (751). We are slightly more accurate on ground-truth accepted patents ($\frac{1288}{1288+652} = 66.39\%$) than rejected patents ($\frac{1134}{1134+751} = 60.16\%$). Related, our precision ($\frac{1288}{1288+652} = 66.39\%$) is better than our recall ($\frac{1288}{1288+751} = 63.16\%$) for identifying accepted patents.

In our result section, we noticed that the bag-of-words models were only able to outperform language model-based approaches even with metadata augmentation on the claims text, but not the rest of the application. We think this result can be explained by some of the qualitative linguistic features of the claims section:

- **Technical jargon**: Patent language tends to include long sequences of technical words, which is unusual for most English text. An example sentence from a patent for "modified human growth hormones" reads: "Stimulated T-cell proliferation is measured using 3H-thymidine (3H-Thy) and the presence of incorporated 3H-Thy assessed using scintillation counting of washed fixed cells." We suspect that bag-of-words models can outperform pretrained language models in these cases since technical words like "thymidine" are likely not seen often during pretraining tasks.
- **Legal jargon**: Patents are legal documents, so the language therein is not free-formed. Instead, it's crafted carefully for legal interpretation, resulting in abnormal English language. Previous work, such as Limsopatham 2021, identify difficulties in legal document classification using langauge models, which supports this thesis [6].
- **Non-grammatical format**: Patent claims especially are filled mostly with enumerated lists, English descriptions of tabular data, and generally many sentence fragments. We suspect BERT's masking pretraining task does not prepare it well for language of this form.
- **Long sequences**: As we stated in 5.1, the claims, background, and description sections of patents are long on average. Claims average 1,272 tokens, backgrounds average 627 tokens, and descriptions average 11,856 tokens. All of these lengths are longer than BERT's maximum sequence length of 512. Bag-of-words models, by contrast, can support much larger "sequence" sizes by ignoring sequences altogether and just taking as many words as the maximum vocabulary size allows. Future work should explore whether this limitation

holds for the Longformer model as well, which can support attention on entire claims and background sections.

Given these results, we suspect that future work could improve on our results with an ensemble model combining bag-of-words, metadata embeddings, and language model encodings of other text sections, such as the abstract or the description.

7 Conclusion

Taking advantage of the novel HUPD dataset, we specified a metadata-augmented natural language model that was able to out-perform the baseline language models from the reference paper on all text fields. Specifically, for the abstract field, this model performed the best relative to all other models, highlighting the importance of incorporating metadata into the language model. However, on the claims section, our model was not able to surpass the bag-of-words model. We argue that this can be explained by distinctive attributes of this text field: the high frequency of technial and legal jargon as well as the non-grammatical format and long sequences. Moreover, due to memory and compute constraints, we were forced to truncate input text sequences (at 256 tokens) and were unable to train the Longformer method on a reasonable batch size. We believe that future work – less constrained by these technical specs – should take advantage of the rich nature of this text, specifying models with the capacity for much longer text streams and/or ensemble methods that are able to combine insights across these text streams.

Finally, while this paper specifically focused on the area of US patent applications, the possibilities unlocked from the introduction of this dataset have wide-reaching implications. The obtained fine-tuned natural language model can be used to enable transfer learning to other highly-structured governmental application contexts. For instance, such a model could be generalizable to the context of scientific grant proposals, where researchers must submit an application to a US governmental agency in order to receive funding. Moreover, the findings from the patent application acceptance model have the potential to impact the larger literature on innovation, e.g. what makes a successful product, and, outside of a research context, inform future innovators on how to prepare more successful filings.

References

[1] Mirac Suzgun, Suproteem K Sarkar, Luke Melas-Kyriazi, Scott Kominers, and Stuart Shieber. The harvard uspto patent dataset: A large-scale, well-structured, and multi-purpose corpus of patent applications. 2021.

[2] Eva Sharma, Chen Li, and Lu Wang. BIGPATENT: A large-scale dataset for abstractive and coherent summarization. pages 2204–2213, July 2019.

[3] Malte Ostendorff, Peter Bourgonje, Maria Berger, Julián Moreno Schneider, Georg Rehm, and Bela Gipp. Enriching BERT with knowledge graph embeddings for document classification. *CoRR*, abs/1909.08402, 2019.

[4] Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The long-document transformer. *CoRR*, abs/2004.05150, 2020.

[5] International patent classification (ipc) 2015. 2015.

[6] Nut Limsopatham. Effectively leveraging BERT for legal document classification. In *Proceedings of the Natural Legal Language Processing Workshop 2021*, pages 210–216, Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

A Appendix

Figure 6: Figure from the HUPD paper, comparing their dataset to existing patent datasets for NLP.

Dataset	# Docs	Title	Abst	Appl	Exam	Invt	PD	Claims	Bkgd	Dsc	PCs	Years	Primary Purpose
WIPO-alpha	75,250	✓	✓		✓	✓	✓	✓	✓	✓	✓	< 2002	Classification
CLEF-IP (2011)	1,500,000	✓	✓			✓	✓	✓		✓	✓	< 2009	Retrieval+Classification
USPTO-2M	2,000,147	✓	✓					✓			✓	2006-2015	Classification
BIGPATENT	1,341,362	✓	✓							✓		1971-2018	Summarization
Ours (AIPD)	**4,518,263**	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	**2004-2018**	**Multi-Purpose**

Table 4: Model performance on a smaller sample. Because of the aforementioned compute struggles, we chose an even smaller sample and ran a lot of models on this sample to better understand their dynamics. This is why performance here is slightly higher. We think that natural language models were better able to rote memorize in this case.

Metadata-Augmented	Model	Batch Size	Validation Accuracy (%)				
			Abstract	Claims	Description	Summary	Background
Yes	DistilBERT	64	**63.29**	66.08	**67.34**	64.34	**64.34**
No	DistilBERT	64	62.94	65.08	66.74	**65.39**	64.34
No	RoBERTa	32	50	65.74	66.44	65.34	50
No	Logistic Regression	64	54.55	56.29	55.24	59.79	60.84
No	Bernoulli Naive Bayes	1	62.24	**67.03**	66.78	63.99	60.48