Pre-trained Contextual Embeddings for Litigation Code Classification

Max Bartolo
UCL
m.bartolo@cs.ucl.ac.uk

Kamil Tylinski
Mishcon de Reya LLP
kamil.tylinski@mishcon.com

Alastair Moore
Mishcon de Reya LLP
alastair.moore@mishcon.com

Abstract

Models for a variety of natural language processing tasks, such as question answering or text classification, are potentially important components for a wide range of legal machine learning systems. These tasks may include examining whole legal corpora, but may also include a broad range of tasks that can support automation in the digital workplace. Importantly, recent advances in pre-trained contextual embeddings have substantially improved the performance of text classification across a wide range of tasks. In this paper, we investigate the application of these recent approaches on a legal time-recording task. We demonstrate improved performance on a 40-class J-code classification task over a variety of baseline techniques. The best performing single model achieves performance gains of 2.23 micro-averaged accuracy points and 9.39 macro-averaged accuracy points over the next best classifier on the test set. This result suggests these techniques will find broad utility in the development of legal language models for a range of automation tasks.

1 Introduction

Legal data comes in a variety of different forms, from contracts and legal documents containing technical language, to the variety of correspondence between client and solicitor (from email to transcripts), to billing and enterprise performance management (EPM) systems used to support the business of law.

Narrative text	J-code
Working on response from <ORG></ORG> and statutory review.	JE20
Preparing documents for meeting with <PERSON></PERSON>	JH30
Attendance on client, email exchange	JJ70

Table 1: Example narrative text for the classification task. Given a sentence of text describing the actions completed by the lawyer, assign a label based on a discrete J-codes label set. J-codes are time-recording codes introduced to comply with requirements under the UK Civil Procedure Rules. The process of redaction, highlighted, is discussed in Section 3.3.

Developing systems that can support the automation of a variety of tasks across the digital workplace involves working with heterogeneous data, with different quantities of labelled data (for the purposes of supervised learning) of variable quality. For this reason, practitioners are increasingly turning to more indirect ways of injecting weak supervision signals into their models (Ratner et al., 2017). Recent work on multitask learning (Ratner et al., 2019) has developed an approach to deep learning architectures that learn massive multitask models with different heads adapted for different tasks.

A traditional approach to text classification tasks is to create a linear classifier (Logistic regression or Support Vector Machine) on sentences presented as bag of words. The main disadvantage of this method is its inability to share parameters among classes and features (Joulin et al., 2017). Alternatively, the problem be approached by means of neural networks (Zhang et al., 2015), where transformer architectures has proven to be more appropriate for a wide variety of tasks, not only text classification (Vaswani et al., 2017; Dai et al., 2019).

Importantly, incorporating pre-trained contex-

In: Proceedings of the First International Workshop on AI and Intelligent Assistance for Legal Professionals in the Digital Workplace (LegalAIIA 2019), held in conjunction with ICAIL 2019. June 17, 2019. Montréal, QC, Canada.

tual embeddings (Peters et al., 2018; Radford et al., 2018; Devlin et al., 2018) has led to impressive performance gains across many natural language processing tasks such as question answering, natural language inference, sequence labelling and text classification. Models with access to pre-trained language knowledge currently provide state-of-the-art results on the GLUE benchmark[1] tasks and also outperform human baselines in some cases. The GLUE benchmark consists of nine natural language understanding tasks (e.g., natural language inference, sentence similarity, etc.). Each comes with its own unique set of examples and labels, ranging in size from 635 training examples (WNLI) to 393k (MNLI) (Wang et al., 2018).

However, legal text (whether it contains technical language or simple correspondence) tends to differ from the text corpora on which these state-of-the-art language models are trained, such as Wikipedia and BookCorpus. In this paper, towards the goal of developing large multitask models for different legal applications, we first demonstrate the successful use of pre-trained language models transferred to a legal domain task.

We focus on the task of litigation code classification, illustrated in Table 1, which is an important sub-task in legal time-recording and for preparing bills of costs for assessment by the courts. We base our approach on fine-tuning BERT (Bidirectional Encoder Representations from Transformers), a transformer-based language representation model, (Devlin et al., 2018) and our evaluation shows that a single pre-trained model achieves significant performance gains over the next best classifier on the test set.

2 Related Work

Text classification is a category of Natural Language Processing (NLP) tasks with real-world applications such as spam detection, fraud identification (Ngai et al., 2011), and legal discovery (Roitblat et al., 2010). Formally, it is about assigning a Boolean value to each pair of $\langle d_j, c_i \rangle \in \mathcal{D} \times \mathcal{C}$ (Sebastiani, 2002), where $\mathcal{D}$ in our example is a domain of narrative documents and $\mathcal{C} = \left\{c_1, ..., c_{|C|}\right\}$ a set of J-Codes such that we obtain a decision value for each narrative document d_j being classed as c_i.

[1] https://gluebenchmark.com/leaderboard

Classification tasks require large quantities of training data, but in many domain-specific applications the construction of a large training set is very costly and requires the use of experts to label data. The use of pretrained embeddings allows models to obtain linguistic knowledge from very large auxiliary corpora, often reduce the amount of task-specific training data required for good performance.

Recent approaches to natural language processing have revolved around neural methods for inferring probability distributions over sequences of words, referred to as language modelling (LM), using deep learning architectures. Recurrent Neural Network (RNN) based language models, owing largely to their capacity for learning sequential context, have been extensively researched (Mikolov et al., 2019; Chelba et al., 2013; Zaremba et al., 2014; Wang and Cho, 2015; Jozefowicz et al., 2016) despite various challenges (Merity et al., 2017; Yang et al., 2017). The sequential nature of RNN-based models precludes parallelization within training examples which makes scaling to long sequence lengths and large corpora challenging. The Transformer architecture, relying on stacked self-attention and pointwise, fully-connected layers, allows for significantly more parallelization (Vaswani et al., 2017).

One approach to developing deep architectures for specific language tasks has been to exploit feature representations learned from large datasets of general purpose data such as Wikipedia. These pre-trained approaches are now key components in many natural language applications (Mikolov et al., 2013). These concepts have also been extended to the legal domain, including the creation of the `Law2Vec` legal word embeddings, which is likely to accelerate the progress in this research area (Chalkidis and Kampas, 2019).

There are generally two strategies for applying pre-trained language models to downstream tasks: *feature-based* and *fine-tuning*. The feature-based approach, as was used in ELMo (Peters et al., 2018), learns a fixed representation, or feature space, on a large text corpus. More specifically, ELMo develops a coupled forward LM and backward LM approach as well as a linear combination of the hidden representations stacked above each input word for each end task, and markedly improves performance over just using the top LSTM layer representation.

The fine-tuning approach, as demonstrated in ULMFiT (Howard and Ruder, 2018) and GPT (Radford et al., 2018), introduce minimal task-specific parameters and are adapted for downstream tasks simply by re-learning the weights in one or more layers of the deep architecture.

In this paper, we build upon the recent release of BERT (Devlin et al., 2018), which makes use of a masked language model for its pre-training objective to learn a deep bidirectional language model. We develop our approach by fine-tuning the pre-trained parameters for the downstream legal time-recording classification task.

Text classification in the legal space has included research in court ruling predictions (Sulea et al., 2017) and legal deontic modality classification (Neill et al., 2017), but the incorporation of pre-trained contextual embeddings remains relatively unexplored.

3 Litigation Code Classification

3.1 Overview

The task is a 40-class classification problem where the labels are litigation J-Codes. The J-codes set are one set of the Uniform Task Based Management System (UTBMS) codes used to classify legal services performed by a legal vendor in an electronic invoice submission[2].

The background of the J-code-set originates from the Review of Civil Litigation Costs in England and Wales (Nelson and Jackson, 2014). A key recommendation of the review was that a new format for bills of costs be standardized to increase both the transparency of costs assessed by the courts, and the consistency in the way costs are presented to judges.

The new format, designed to be produced and analyzed in digital workflows, resulted in a set of discrete J-Codes that are used to categorize work undertaken. There are three hierarchical levels of granularity. The highest level is the `Phase`. Examples include *Pre-Action* work and *Disclosure* corresponding to J-code `JC00` and `JF00` respectively. The intermediate level of generality is the `Task`. Each `Phase` has a finite and limited number of `Tasks` assigned to it. For example, the *Issue / Statements of Case* phase (`JE00`) includes the lower tier tasks of *Review of Other Party/Opponents' Statement of Case* (`JE20`) and *Amendment of Statement of Case* (`JE40`). An example of the distribution of J-codes used in the evaluation can be seen in Figure 1. The lowest tier is `Action`, but we do not use this granularity in this study. `Actions` specify how the work is done, `Tasks` inform of what is being done and are further grouped by `Phases`. The detailed explanation of the J-codes structure can be found in (Nelson and Jackson, 2014).

3.2 Motivation

This classification task is important in the context of legal digital workflows because it allows law firms to extract value from billing data. Organizing work by `Phase` and `Task` facilitates more effective budgeting, particularly as alternative fee arrangements become more prevalent, and increases transparency across different clients and matters.

Automating `Phase-Task` code classification also reduces administrative burden upon lawyers, who may each record thousands of time entries involving these codes annually. Furthermore, the adoption of UTBMS codes can be inconsistent within industries or even a given firm, with some lawyers delegating their task-based coding or assigning blocks of time entries to the same code. In these cases, automation is likely to improve the quality of data collected and allow for inter-department comparative analyses.

Moreover, it is possible for time entries to be entered just once into a solicitor's system (including `Task` and `Activity` codes) and then used in a variety of different reporting applications, from the client, to the court to the normal administrative functions of finance and tax.

Lastly, the nature of billing data in an industry characterized by time-based charging, means it is likely to be a key source of data in any multimodal multitask system supporting task automation in the digital workplace.

All of the above emphasize the importance of accuracy, when assigning the codes. There are also financial incentives, as any incorrect entries may be impossible to recover from the other side or not approved by the court. Additionally, the time fee earners spend amending and checking the codes has to be written off and does not provide any benefit to the law firm. Thus, automated code

[2] A similar set of codes have previously been developed in the United States. Here the codes have been developed to provide a common language for e-billing, under which both the law firm and the client have systems using a common code set for respectively the delivery and analysis of bills - commonly referred to as L-codes.

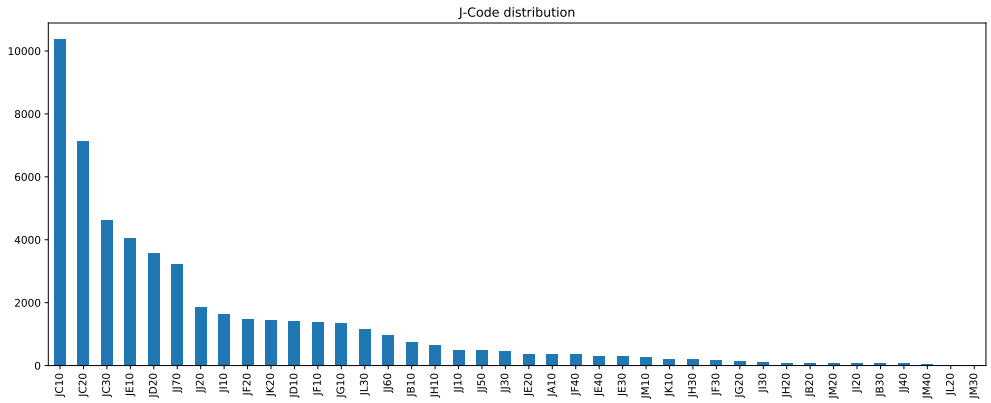

Figure 1: Histogram to show the distribution of J-codes. The long tail demonstrates the class imbalance in this dataset. This is to be expected as time entries, aggregated by type of work performed, mean that multiple time entries could result from the services performed in a single day on a single matter.

assignment can lead to significant improvements in productivity, even if the output requires to be reviewed by the legal professional.

3.3 Data

The data is a collection of narratives from a legal firm's proprietary set spanning more than 1500 matters and 300 timekeepers. Due to its sensitive nature, the data has been anonymized using a Named Entity Recognition (NER) algorithm that identifies and redacts the names of people, organizations, and locations, among other entity types in the form of a word mask. This algorithm combines machine learning based on linguistic features with stricter pattern-based exclusions. Another effect of preprocessing data with the NER algorithm is to ensure a higher degree of model generalisability, since it is not trained based on specific proper nouns which may be present in the vocabulary at training time but not at test time. This can be seen in Figure 2 where we can see high mask counts for MASK_PERSON and MASK_ORG.

The data has been cleaned by a heuristic whereby blocks of time entries from the same timekeeper assigned almost exclusively to the same phase-task code combination were excluded. Despite this process, classes in the data set remain relatively imbalanced, with about one third of entries assigned to the most common phase code and one fifth of entries assigned to the most common task code.

The data set consists of $51,948$ examples split into training, development, and testing sets using $80\%/10\%/10\%$ split ratios respectively.

3.4 Evaluation Metrics

This is a multi-class classification problem, with significant class imbalance so we evaluate on both micro-averaged accuracy and macro-averaged accuracy in a one-vs-all setting.

The micro-averaged accuracy is computed by aggregating to contributions of all the classes to compute the average by taking the number of correct predictions divided by the total number of examples.

The macro-averaged accuracy considers the computation of the accuracy for each individual class independently (class average), followed by taking the average across classes (hence treating all classes equally). This is useful for understanding how the system performs across each class despite the limited data points for particular classes.

4 Models

To demonstrate any improved performance from the use of pre-trained contextual embeddings on this domain specific task we benchmark performance against a variety of different baseline models.

4.1 Random Baseline

The random baseline simply predicts a random class for any given data point. As such, we expect the micro-averaged accuracy to be roughly $\frac{1}{num\_classes}$.

4.2 Majority Baseline

We present a majority baseline which predicts the most common class (`JC10`) for any given data

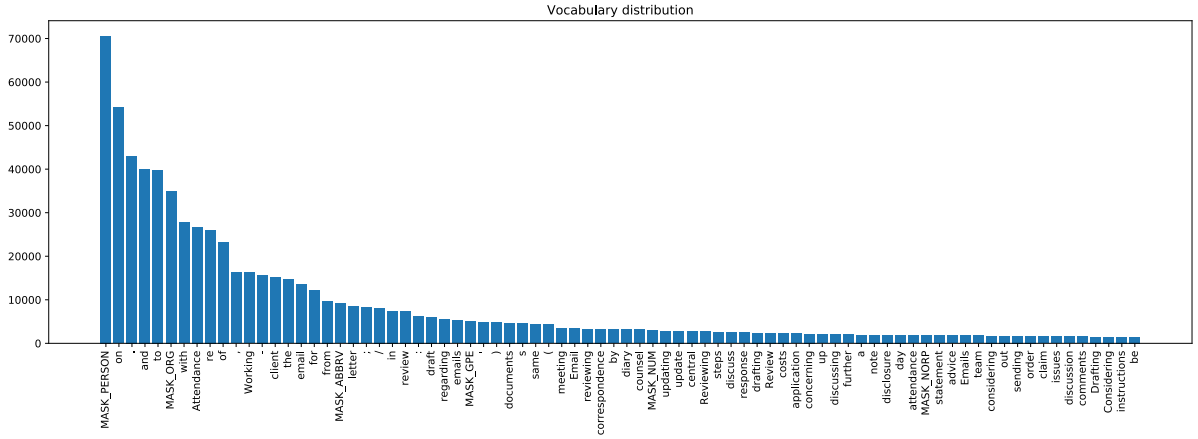

Figure 2: Histogram to show the distribution of vocabulary, including word masks. We can see that the person (MASK_PERSON) and organization (MASK_ORG) masks are more frequent.

point.

4.3 Surface Logistic Regression

We featurise the narratives to the surface models by normalising the input narratives and converting to a Bag-of-Words (BOW) sparse representation. In addition, we also experiment with character and word tokenisation, removal of stopwords and TF-IDF feature reweighting but observe best performance on bigram-enhanced BOW features tokenised at word level while retaining stopwords. A logistic regression model is applied to the featurised input in a one-versus-rest multi-class scheme and an L2 weight regularisation penalty.

4.4 XGBoost Baseline

As a final baseline, we use the scalable gradient-boosting implementation *XGBoost* (Chen and Guestrin, 2016), which has been used on various text classification tasks with strong performance results based on additive tree-based optimisation. As with the logistic regression baseline, we performed pre-processing based on stopword-removal, TF-IDF weighting, and n-gram selection. We also experimented with lemmatisation and case standardisation to achieve highest model performance.

4.5 BERT Models

We work with the HuggingFace[3] PyTorch implementation of BERT (Bidirectional Encoder Representations from Transformers) model and run various fine-tuning experiments. BERT is designed to learn deep bidirectional representations by jointly conditioning on both left and right context in all layers through a masked language model objective. Pre-trained BERT representations are publicly available for download and can be fine-tuned with just one task-specific output layer to create state-of-the-art models for a wide range of tasks (Devlin et al., 2018). We experiment with the uncased and cased versions of pre-trained $BERT_{BASE}$ which is a 12-layer transformer architecture with a hidden size of 768 and 12 self-attention heads adding up to 110 million parameters, and the uncased version of $BERT_{LARGE}$ which is a 24-layer transformer architecture with a hidden size of 1024 and 16 self-attention heads adding up to 340 million parameters, both trained on a combined BookCorpus and Wikipedia corpus of 3.3 billion words on 4×4 and 8×8 TPU slices respectively for 4 days.

We fine-tune the models on an AWS `p2.xlarge` instance running a single NVIDIA K80 GPU. We adapt the BERT fine-tuning mechanism for single sentence classification tasks to the matter classification task.

4.6 Chronology-enhanced models

In principle, any production system for time-recording can take account of additional information to support the classification task. The J-codes set has ordinal structure resulting from the progression of `Phases` and `Tasks` during the case, and any specific time-entries also have temporal structure that can be exploited.

As a result of this, we can significantly im-

[3] https://github.com/huggingface/pytorch-pretrained-BERT

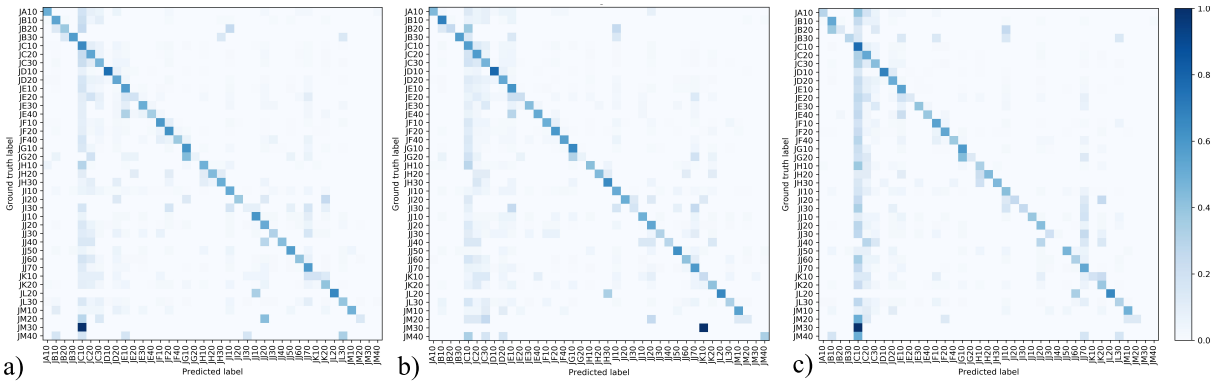

Figure 3: Confusion matrices. a) $BERT_{BASE}$ (Uncased) b) $BERT_{LARGE}$ (Uncased) c) *XGBoost*. We can see that $BERT_{LARGE}$ is better at classifying class $JC10$, particularly against $JM30$.

prove model performance by incorporating features based on the set of codes typically associated with a user or matter. Therefore, we include a chronology-enhanced XGBoost model in our analysis to set any performance improvements in context.

Care is taken to verify that the model behavior is not to simply repeat the last code on a given matter by setting chronology-based features to zero, obtaining predictions from the chronology-enhanced model, and confirming that the difference in micro-accuracy is not greater than five percent relative to the purely text-based model.

5 Results and Discussion

Results for the different models are presented in Table 2. We observe substantial performance improvements of BERT models over the text-based baselines as well as the *XGBoost* text-based model, particularly with regards to macro-accuracy.

The best performing BERT single model achieves performance gains of 2.23 micro-averaged accuracy points and 9.39 macro-averaged accuracy points over the *XGBoost* text-only classifier on the test set. This is likely to have a strong effect on user experience of a production system as it indicates substantially better performance on less common classes. It also demonstrates the effectiveness of pre-trained methods to incorporate prior knowledge and learn on low-resource data, despite the linguistic differences between the pre-trained and legal domains.

We also perform an in-depth error analysis, including visual inspection of different model predictions and confusion matrices (see Figure 3) to understand which classes the models commonly mistake for others. We find that both the *XGBoost* text-based model and the $BERT_{BASE}$ model commonly predict the most common class `JC10` (*Factual Investigation: Work required to understand the facts of the case including instructions from the client and the identification of potential witnesses*) when the ground truth is `JM30` (*Hearings: Includes preparation for and attendance at hearings for directions and interim certificate applications as well as the detailed assessment itself*). We also observe that all text-based models have difficulty distinguishing between `JG10` (*Taking, preparing and finalising witness statement(s)*) and `JG20` (*Reviewing Other Party(s)' witness statement(s)*). It is likely that this can be explained to some extent by the text anonymisation.

We can also see that there are different error patterns between the BERT and *XGBoost* models and therefore we are likely to be able to improve performance in a production system using an ensemble approach. Furthermore, in addition to the `Task` level results above, results on the `Phase` level are encouraging for use in production, with a micro-accuracy rate of 90.40 percent for the chronology-enhanced *XGBoost* model. In some cases, such data is already sufficiently granular to derive actionable firm budgeting insights and an improvement over existing manual methods.

6 Conclusion and Future Work

Recent empirical improvements due to transfer learning with language models have demonstrated that rich, unsupervised pre-training is an integral part of many language understanding systems. Here we present experiments and analysis of state-of-the-art models based on deep pre-trained con-

Model	Micro Acc. (%)	Macro Acc. (%)
Random Baseline	2.02	2.26
Majority Baseline	19.96	2.50
Surface Random Forest	42.66	28.49
Surface Logistic Regression	45.87	32.30
Surface Logistic Regression (enhanced with bigram features)	51.78	39.30
XGBoost	53.15	36.65
BERT Base (Uncased)	55.17	44.28
BERT Base (Cased)	**55.38**	**46.04**
BERT Large (Uncased)	54.17	45.25
XGBoost (Chronological features)	77.11	61.51

Table 2: Results of the models on the test set. We can see increased performance over baseline models.

textual embeddings applied to the task of litigation code classification. We show that BERT fine-tuned to the 40-class matter classification task provides substantial performance gains over our best-performing baseline.

One area to explore further is to incorporate these chronology-based features into a BERT-centric approach. For example, one approach could be to learn contextual embeddings for text over temporal set of J-codes. Another could be to ensemble the predictions of purely chronology-based model with the BERT output.

We achieve our primary goal of demonstrating that there is the capability to transfer pre-trained language knowledge from a general corpus to the legal domain task, with improved performance.

Notwithstanding this fine-tuning result, in future work we intend to extend this by learning contextualised representations from legal corpora, a direction that has achieved some success in other domains (Lee et al., 2019) and which could be applied across a wide variety of tasks in the legal domain.

Moreover, although we have explored use of multi-task learning framework, we have only demonstrated performance on a single legal task. Future work will likely include extending this analysis to a set of legal benchmark tasks that include natural language inference tasks (similar to GLUE) on publicly available legal datasets.

Given the relatively high degree of class imbalance present in `Phase` and `Task` codes, as well as the level of legal expertise involved in distinguishing closely related or rarer options, this classification problem lends itself well to human-in-the-loop machine learning. Such an active learning platform would involve feeding timekeeper-validated data back into the model for near-real-time retraining. This method of data collection may also achieve scale conducive to learning contextualised legal-corpora representations mentioned above.

Acknowledgements

We thank Edwin Zhang and Brandon Hill at Ping Inc. for their assistance in the data preparation, baseline modeling, and chronology enhancements.

References

Ilias Chalkidis and Dimitrios Kampas. 2019. Deep learning in law: early adaptation and legal word embeddings trained on large corpora. *Artificial Intelligence and Law*, 27(2):171–198.

Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, and Phillipp Koehn. 2013. One Billion Word Benchmark for Measuring Progress in Statistical Language Modeling. *arXiv preprint arXiv:1312.3005*.

Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A Scalable Tree Boosting System. *https://arxiv.org/abs/1603.02754*.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime G. Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context. *arXiv preprint arXiv:1901.02860*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of Tricks for Efficient Text Classification. *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431.

Rafal Jozefowicz, Oriol Vinyals, Mike Schuster, Noamand Shazeer, and Yonghui Wu. 2016. Exploring the limits of language modeling. *arXiv preprint arXiv:1602.02410*.

Jinhyuk Lee, Wonjin Yoon1, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2019. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, 1.

Stephen Merity, Nitish Shirish Keskar, and Richard Socher. 2017. Regularizing and optimizing LSTM language models. *arXiv preprint arXiv:1708.02182*.

Tomas Mikolov, Martin Karafiat, Lukas Burget, Jan Cernocky, and Sanjeev Khudanpur. 2019. Recurrent neural network based language model. *INTERSPEECH*, 2:1045–1048.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *NIPS*.

James O' Neill, Paul Buitelaar, Cecile Robin, and Leona O' Brien. 2017. Classifying Sentential Modality in Legal Language: A Use Case in Financial Regulations, Acts and Directives. *Proceedings of the 16th Edition of the International Conference on Articial Intelligence and Law (ICAIL '17)*.

David Nelson and Jackson. 2014. EW-UTBMS Civil Litigation J-Code Set Overview and Guidelines.

EWT Ngai, Yong Hu, YH Wong, Yijun Chen, and Xin Sun. 2011. The application of data mining techniques in financial fraud detection: A classification framework and an academic review of literature. *Decision Support Systems*, 50(3):559–569.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws.com/openai-assets/researchcovers/languageunsupervised/language understanding paper.pdf*.

Alexander Ratner, Stephen H. Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher RÃl'. 2017. Snorkel: Rapid training data creation with weak supervision. *Proceedings of the VLDB Endowment*, 11(3):269–282.

Alexander Ratner, Braden Hancock, Jared Dunnmon, Frederic Sala, Shreyash Pandey, and Christopher Re. 2019. Training Complex Models with Multi-Task Weak Supervision. *AAAI*.

Herbert Roitblat, Anne Kershaw, and Patrick Oot. 2010. Document categorization in legal electronic discovery: computer classification vs. manual review. *Journal of the Association for Information Science and Technology*, 61(1):70–80.

Fabrizio Sebastiani. 2002. Machine Learning in Automated Text Categorization. *ACM Computing Surveys*, 34(1):1–47.

Octavia-Maria Sulea, Marcos Zampieri, Shervin Malmasi, Mihaela Vela, Liviu P. Dinu, and Josef van Genabith. 2017. Exploring the use of text classification in the legal domain. *Proceedings of 2nd Workshop on Automated Semantic Analysis of Information in Legal Texts*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. *31st Conference on Neural Information Processing Systems (NIPS 2017)*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Tian Wang and Kyunghyun Cho. 2015. Larger-context language modelling. *arXiv preprint arXiv:1511.03729*.

Zhilin Yang, Zihang Dai, Ruslan Salakhutdinov, and William W. Cohen. 2017. Breaking the softmax bottleneck: A high-rank RNN language model. *arXiv preprint arXiv:1711.03953*.

Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. 2014. Recurrent Neural Network Regularization. *arXiv preprint arXiv:1409.2329*.

Xiang Zhang, Junbo Zhao, and Yann Lecun. 2015. Character-level Convolutional Networks for Text Classification. *Advances in Neural Information Processing Systems 28*, pages 649–657.