Learning the Rulebook: Challenges Facing NLP in Legal Contexts

Mark S. Krass
Department of Political Science
Stanford University
Stanford, CA 94305
mkrass@stanford.edu

Abstract

Despite the facial simplicity of the task, common deep learning tools used to classify text documents perform poorly in the setting of legal documents. I implement four families of Convolutional Neural Networks and train them on a dataset of approximately 300,000 decisions by the Board of Veterans' Appeals (BVA). The objective is to predict whether the decisions by the BVA will be (a) not appealed or (b) appealed, and then within the category of appeals, whether the decision will be (i) affirmed, (ii) reversed or remanded (indicating error), or (iii) whether the appeal will be dismissed. I argue that the poor performance of the classifiers can be explained by three roadblocks: hierarchical labelling, embedding weakness, and abstract decision rules.

1 Introduction

Why did a legal document receive a particular classification probability from a CNN? Well-established approaches typically answer this question by propagating the gradient of the classifier back to individual words within the input document. Unfortunately, in long documents such as legal opinions or legal briefs, word-by-word gradient scores present at least two well-documented problems. First, writers cannot change long documents on the basis of individual word scores: They think in larger units (e.g., sentences or sections). Second, writers typically seek more qualitative advice about their work.

While the focus of this project was initially intended to be on improving these challenges in explanation, I ultimately focused on a narrower issue instead: Making good predictions. The difficulty of accurately predicting the decisions of higher courts on the basis of lower court opinions forms the first step in the process of trying to detect judicial error. I argue that the application of text-based deep learning methods to legal reasoning presents challenges poorly addressed by current standard models in NLP. In particular, I document the poor performance of CNNs in predicting the outcome of legal cases conditional on a prior court's judgment. Specifically, I show that a simple 1-layer CNN, as well as models inspired by AlexNet and GoogLeNet, achieve worse-than-expected results on relatively simple text classification tasks. I use experimental results to argue for three roadblocks to the use of NLP and deep learning in law: first, the unique hierarchical structure of outcomes; second, the linguistic quirk of legal adversarialism; and third, the challenge of using acontextually trained embeddings.

1.1 Setting

Before proceeding to a discussion of the NLP context, it may be useful to consider some substantive information about the setting in which these classifications are taking place. The decisions I study

32nd Conference on Neural Information Processing Systems (NIPS 2018), Montréal, Canada.

come from the Board of Veterans' Appeals, an agency dedicated to processing claims by wounded war veterans that they were wrongly denied access to disability payments.

Until the Veterans' Judicial Review Act was signed into law on November 18, 1988, decisions by the BVA could not be reviewed by any other court.[1] Lawmakers changed that: It created an appeals court, called the Court of Appeals for Veterans' Claims (CAVC), the sole purpose of which is to review BVA decisions for legal error. Congress' purpose in creating CAVC was to ensure that errors by decision-makers at beginning stages of the process could be progressively filtered out. In this way, this paper's focus on predicting the fate of a case on appeal can be seen as a form of error detection: the goal of each successive decision-maker in this administrative court system is to find out when *error* has been made by a previous decision maker.

The life cycle of a typical benefits claim begins when a veteran submits claim to their local VA office. If a veteran's claim is denied, or if the veteran disagrees with their disability rating, the veteran has one year to file a Notice of Disagreement (NOD). Upon the filing of the NOD, the VA's Regional Office is required to prepare an authoritative "Statement of the Case" listing the reasons for its initial decision and the evidence it used to generate those reasons. The Statement of the Case is then provided to the claimant.

If the claimant disagrees with the VA's reasons for its decision, she may file a Substantive Appeal with the Board of Veterans' Appeals (BVA). The BVA issues most decisions on the basis of a paper record alone. But the veteran is entitled to an in-person or video hearing if she wishes to have one and is entitled to submit additional evidence to support her claim. Finally, veterans who disagree with the BVA's determination may appeal to the CAVC. The CAVC is the final stop for most cases, especially cases where a disagreement centers on factual concerns or the application of law to facts. Article III courts have no jurisdiction to reconsider specific benefit determinations; they may rule only on challenges to the validity of the statutes or regulations underlying BVA or CAVC decision.

Once they receive a case file, both the BVA and the CAVC may take one of three kinds of action. They can reverse or vacate the order of the previous decision maker and remand the matter to be resolved differently. They can deny the appeal and simply leave the previous decision in place. Or they can remand the matter without deciding on the underlying substantive question with orders to develop or reconsider certain facts and then review the decision. At the CAVC in 2017, about 37% of cases were affirmed in part and reversed or vacated in part; about 21% were reversed and remanded; about 18% were simply remanded; and about 12% were affirmed.

1.2 Related Work

The application of NLP to legal text might straightforward—and imminent. That is certainly received wisdom among lawyers. To name just one example, Volokh (2019) breathlessly predicts that deep learning-based text generation models will soon allow for replacement of human judges by machines.[2]

Like Narcissus to Echo, the deep learning community has paid relatively attention to the legal domain as compared to, say, image classification. Ditto NLP: for perhaps understandable reasons, a small-label classification problem has not attracted great interest. Indeed, even those contributions that have been made to date largely avoid the difficult predictive task at the heart of legal reasoning.

One body of work has focused on predicting the conclusion of a legal document from other language *in the same document.* Aletras et al. (2016)(3) predicts the outcomes of cases in the European Court of Human Rights by using the portion of those same cases that does not contain the decision. Others conceptualize the task as one of document summarization. Zhong et al. (2019)(4) use iterative masking to identify the sentences in a judicial opinion most closely correlated with its conclusion. Finally, Grabmair (2017)(5) implements a formalism designed to model the balancing of parties' interests in order to predict outcomes of trade secret disputes, using a similar experimental paradigm (predicting the conclusion of a text from its non-conclusory language).

[1] *See* Veterans' Judicial Review Act, Pub. L. No. 100-687, 102 Stat. 4105 (1988).

[2] Indeed, this assumption is more like an entreaty. The appetite for legal applications for NLP is enormous. Chen (2019, p. 36)(1) proposes that deep-learning predictions of case outcomes can be used to measure judge bias or fairness. But such enthusiasm rests on the presumption that deep learning will "reliably yield[] opinions that we view as sound," that is, will reach the same bottom-line decisions as a reliable human judge would (Volokh 2019:1138).(2)

Another research agenda uses legal texts merely as a setting from which to extract formal argument or rhetorical structures. Walker (2018)(6) summarizes the well-developed field of argumentation mining, which has used legal texts to study the attribution of statements to particular arguments or rhetorical devices. Similarly, Ashley Walker (2013)(7) propose a framework for extracting argument structures from legal test for the purpose of generation. Savelka et al. (2017)(8) focus on predicting sentence boundaries in legal text.

1.3 Building expectations from Other NLP research on classification.

The use of convolutional neural networks (CNNs) to perform text classification is well-documented. The most famous cases in which simple CNNs have been quite successful include sentiment classification on short text snippets (e.g., Deriu et al. (2016)(9), Ruder et al. (2016)(10), both focused on twitter data). Nonetheless, these studies offer helpful indications of where problems may lie for legal texts.

First, Deriu et al. point out that the use of context-specific pretrained embeddings can be quite impactful in the model's performance. Embeddings trained on American legal texts (which one might think important given the proliferation of terms of art) were not readily available at the time of writing.

2 Approach

As described above, the basic task I aimed to achieve was to classify each BVA opinion into one of five dispositions: unknown, not appealed, appealed-affirmed, appealed-reversed/remanded, and appealed-dismissed.

2.1 Data Wrangling

One unexpectedly challenging step in the process was wrangling the data into a format that could be efficiently read and processed during training. I obtained the 300,000 opinions[3] from a colleague's database and then partitioned them into training, testing, and validation datasets using a standard 70:20:10 split ratio. Because the opinions were stored remotely in a mongoDB, the efficiency of the training algorithm was severely impacted (as each batch of files was streamed one-at-a-time from the remote server). I transferred the files to the Azure server implemented a process using Pytorch's DataLoader paradigm that permitted batch processing while conserving memory.

2.2 Pre-Processing

Decisions vary dramatically in length: the shortest decision I observed was 678 words long; the longest was over 4,000 words long. Thus, after de-casing every word and stripping documents of punctuation and numbers, I trimmed each document down to a maximum document length (M), which I set at 2,000 words, a figure close to the median document size. As I explain below, I find the maximum document length to be a significant parameter and finding the optimal trimming size is a significant task for future work.

Following the paradigm suggested in our coursework, I then converted each document to a tensor of size $M \times E$, where E is the size of a pretrained embedding. I used GLoVE embeddings in all of the implementations I report below.[4] I also regularized the text by dropping words not found in at least three documents, which left a vocabulary of 55,876 word types.

2.3 Modelling

After discussions with the course team and given the average document size, I decided to begin with a CNN-based model paradigm. To that end, I used three standard baseline architectures to build my classifier. For each model, I used a dropout layer immediately before feeding the activations from the convolutional detection layers into the fully-connected linear layer; tuning suggested that overfitting

[3]These had been previously scraped by a collaborator, Prof. Matthias Grabmair, and I do not deserve credit for the scraping step!

[4]For more information on GLoVE, see Pennington et al. (11).

	F1 Micro (Across Observations)	F1 Macro (Across Labels)
Simple 1 layer	0.937	0.164
1 inception layer	0.535	0.123
AlexNet	0.942	0.162
2 labels, 1 layer	0.928	0.162
2 labels, 1 inception layer	0.528	0.119
1 layer with oversampling	0.535	0.167
3 layer with oversampling	0.542	0.168
1K Document Size	0.936	0.164
3K Document Size	0.931	0.163

was an extremely serious issue, so I chose a high dropout rate of 0.5. In order of complexity, the architectures I implemented were as follows:

1. The first modelling paradigm was a vanilla CNN with a single convolutional unit consisting of a 1-dimensional convolutional layer, a ReLU nonlinearity, and a global max pooling step. I tuned the kernel size of the 1-D CNN on the validation data and found that $k = 4$ achieved (relatively) good results. Similarly, I tuned a number of different configurations for the feature map and found that setting 256 output channels was a reasonable place to begin.
2. Next, I implemented a simple model inspired by the GoogLeNet Inceptor unit.(12). Here I ran three convolutional layers with kernel sizes 3, 4, and 5, with 256 output channels each, and then placed each of them into a unit with a ReLU nonlinearity and a max-pooling step. I then fed the result of this concatenated model into a fully-connected linear layer.
3. Finally, I implemented a lightly adapted version of AlexNet, which included five convolutional units (with kernel sizes 11, 5, 3, 3, and 3) and two fully-connected linear layers.

In each case, I used weighted cross-entropy loss on the results. I weighted the minority classes (i.e., all of the subcategories under 'appeal'), significantly higher to account for the model's tendency to predict only majority class labels.

3 Experiments

The baseline classification accuracy of each model is presented in Table 1. As indicated in the discussion above, each of the results reflects the test results from a model subjected to 10 epochs of training on 201,375 training examples with a decaying learning rate beginning at 0.03 and a dropout rate of 0.4. Unless otherwise indicated, documents were trimmed to 2,000 words and were encoded using pretrained, 100-dimensional GLoVE embeddings.

In the top row, we see the results on a 5-label training task with the three different model configurations described above. One striking feature of the results that is immediately obvious is the enormous difference between the so-called 'macro' F1 score (a term I borrow from `scikit-learn`, and which refers to the average of the F1 score for each label, with zero assigned to labels for which the classifier made no predictions) and the 'micro' F1 score, which is the same as accuracy in this case. The severe class imbalance problem is immediately apparent from these results: with the exception of the single-layer CNN trained on 2,000 word document excerpts, all of the models achieve something like 93% accuracy—effectively (though not exactly) simply guessing the majority class every time. In contrast, the F1 'macro' score, i.e., the average accuracy within each individual class, is quite poor. Perhaps the only significant variation I observed was that. The GoogLeNet-inspired 'inception layer' models performed worse than their peers in virtually every case.

With that said, Figures 1 and 2 do indicate that convergence on this model occurred much faster for some models than for others. Figure 1 shows that AlexNet reached a low development perplexity—that is, fit the validation set—much more quickly than did the simpler 1-layer or inception layer models. Similarly, Figure 2 shows that models trained to solve a two-label class problem rather than

the original 5-layer problem converged more quickly, though again Table 1 shows that they ultimately suffered from highly invariant predictions.

Figure 1: AlexNet converges faster to the optimal model.

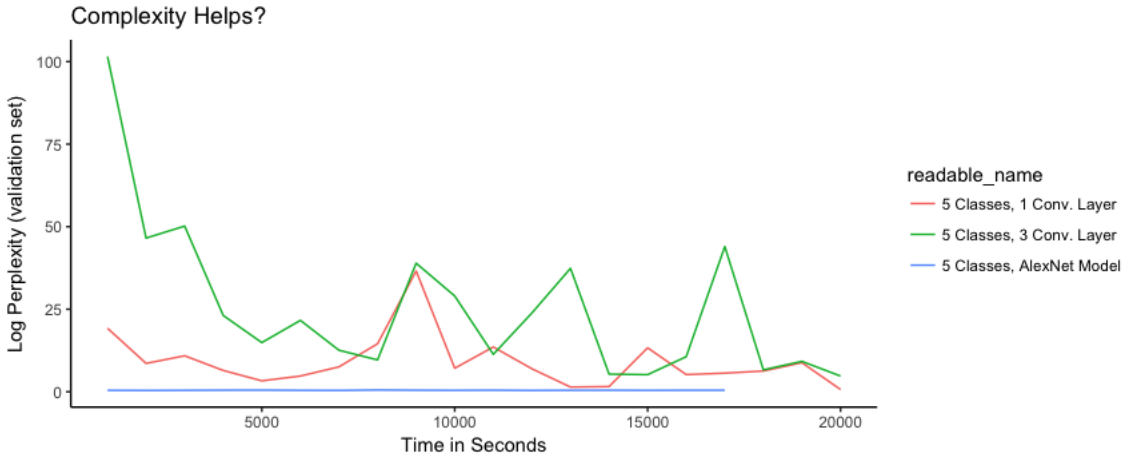
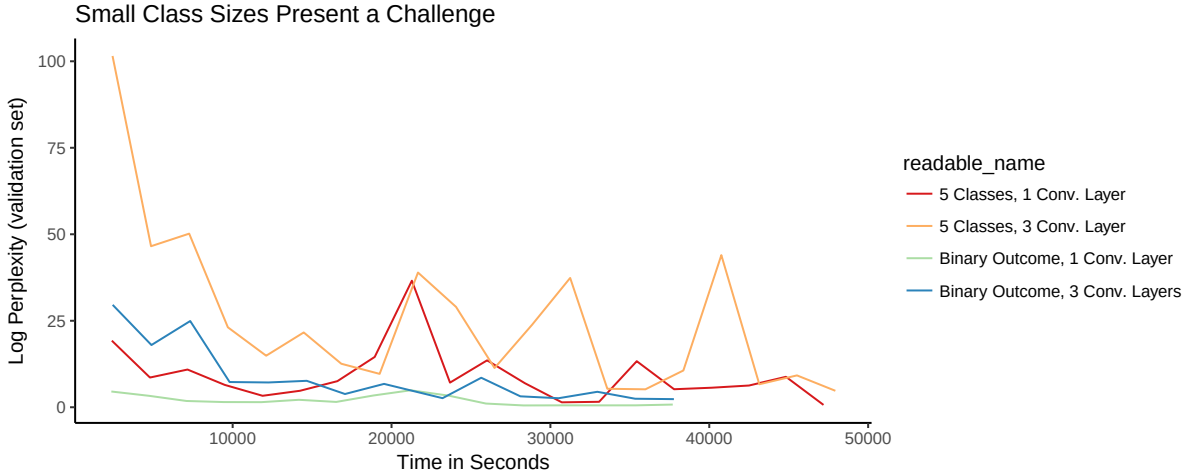


Figure 2: Simplifying the outcome variable speeds convergence, though it does not substantially change the results.



3.1 Oversampling Unsuccessful

To help solve the class imbalance problem, one experiment I implemented involved oversampling the minority class. Instead of randomly sampling batches of documents from my training pool, I created a tool to force a minimum level of batch purity, that is, to generate a sample in which at least some minimum proportion of the documents belonged to the minority class.

Table 1 displays the results of this experiment for a batch purity of 0.2. Remarkably, forcing the models to learn from the minority class more frequently did little to change the important F1-Macro results; while models did start guessing the minority class much more frequently, they did so with very little success.

Not reported in Table 1 were experiments I did to modify the loss function by penalizing failures to guess the minority class more harshly. Like oversampling, these efforts resulted in little improvement of the F1 macro score; they encouraged seemingly random but more frequent predictions of minority label categories.

3.2 Difficult to Gain Qualitative Purchase

In order to try to understand the nature of the errors being made by the models, I implemented a method for propagating 'relevance scores,' that is, unnormalized log probability scores, to individual neurons. I used the layerwise relevance propagation method proposed by (13) to create decision rules for how the log probability scores were propagated by layer, and applied it to the AlexNet model weights to try to understand the source of the issue.

Unfortunately, the model's convergence on an essentially trivial and invariant decision rule—never guess the minority class—meant that LRP was extremely uninformative (as the method correctly returned a salience score of zero for each input neuron).

4 Discussion

Why did the results fro this project turn out to be far worse than expected? In this section, I reflect on three possibilities.

Of course, the first possible reason to mention is a simple data or programming error. I exhaustively checked my code for mistakes and attempted to verify that every stage of the process was working as expected, but given poor learning outcomes such as these the risk of a simple coding or data error is high. In future iterations of this project I will rebuild my framework and be sure to verify that my model is working as expected.

Assuming, however, that the null result is not an artefact of mere error, three features of this task strike me as potentially challenging for NLP-based deep learning methods.

First, I think I could have done more with the hierarchical structure of the labels. While switching to binary labels was less helpful than expected, in the future an appropriate loss function should cluster losses, so that guessing a reversal when the right label is an affirmance is penalized less harshly than failing to predict an appeal.

Second, one significant vulnerability in the setup of this project was the use of generic, pre-trained GLoVE embeddings. One unique feature of legal reasoning is the use of terms of art in speech. Unlike Wikipedia or Reuters or Twitter, all sources of major pretrained embedding dictionaries, legal texts are targeted at a specific subset of readers and the most important words in a document are also often the ones with the most specialized meaning. For example, one recent case in which the BVA was reversed by CAVC (Kraft v. Wilkie)[5] hinged on the question whether a form used by the VA to process claims was 'ambiguous' or not. Thus a human reader would probably apply a great deal of attention to the word 'ambiguous,' a term with important legal meaning. But one might imagine that 'ambiguous' has a more benign meaning in a pretrained embedding. Indeed, the closest neighbour to 'ambiguous' in the Word2Vec projector on Tensorflow is 'vague' (though that is followed by 'misleading' which is closer to the legal meaning).

Finally, it may simply be difficult to learn to 'read' legal documents for error on a word-by-word basis because the definition of error depends on the application of an abstract legal concept. To illustrate the importance of this fact, consider two identical cases that resulted in virtually identical lower court opinions, separated by only one factor: one decision was issued several years after the other. In many areas of law, the simple passage of time would require the application of a new set of rules (i.e., a statute of limitations) that might dramatically change the outcome of the case. Thus, issuing the same opinion for the same facts several years later might be an error. While the setting in which my data arose is not typically characterized by subtle reasoning of this kind, it is possible that legal rules generally are challenging to derive from text alone.

References

[1] D. Chen, "Judicial analytics and the great transformation of american law," *Artificial Intelligence and the Law*, no. 27, pp. 15–42, 2019.

[2] E. Volokh, "Chief justice robots," *Duke Law Journal*, pp. 1135–1192, 2019.

[5] Link here: `https://efiling.uscourts.cavc.gov/cmecf/servlet/TransportRoom?servlet=ShowDoc&dls_id=01205673890&caseId=98501&dktType=dktPublic`

[3] N. Aletras, D. Tsaparanasis, D. Preotiuc-Pietro, and V. Lampos, "Predicting judicial decisions of the european court of human rights: a natural language processing perspective," *Peer J. Comp. Sci.*, 2016.

[4] L. Zhong *et al.*, "Automatic summarization of legal decisions using iterative masking of predictive sentences," *(under review)*, 2019.

[5] M. Grabmair, "Predicting trade secret case outcomes using argument schemes and learned quantitative value effect tradeoffs," *Proc. ICAIL*, 2017.

[6] V. R. Walker, "The need for annotated corpora from legal documents, and for (human) protocols for creating them: The attribution problem," 2018.

[7] V. R. Walker and K. D. Ashley, "Toward constructing evidence-based legal arguments using legal decision documents and machine learning," 2013.

[8] J. Savelka, "Sentence boundary detection in adjudicatory decisions in the united states," 2017.

[9] J. D. et al., "Swisscheese at semeval-2016 task 4: Sentiment classification using an ensemble of convolutional neural networks," 2016.

[10] P. G. Sebastian Ruder and J. Breslin, "Insight-1: Convolutional neural networks for sentiment classification and quantification," 2016.

[11] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, 2014.

[12] C. Szegedy, W. Liu, Y. Jia1, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," 2017.

[13] L. Arras, F. Horn, G. Montavon, K.-R. Müller, and W. Samek, "What is relevant in a text document?: An interpretable machine learning approach," *PLOS ONE*, vol. 12, p. e0181142, 2017.