Detecting Evidence of Gender Discrimination in Fijian Court Documents

Chris Sexton and Greg Tozzi
School of Information
University of California, Berkeley
email: cjsexton, greg.tozzi@ischool.berkeley.edu

Abstract goes here.

1 Introduction

This study considers the problem of identifying evidence of gender-based discrimination in Fijian court records for cases involving gender-based violence (GBV). The problem of GBV in Fiji is serious and well documented, with 64% of Fijian women reporting having experienced violence by an intimate partner. The challenge of holding perpetrators accountable is sometimes confounded by an emphasis on the maintenance of the social order through blanket acceptance of traditional practices—referred to as *vakaturaga*. Traditional practices that make finding justice for survivors of GBV challenging include the patriarchal ordering of society—(*matanitu*)—and the practice of making atonement between men—(*bulubulu*). *Bulubulu* involves making an offer of items of some value as a means of securing forgiveness and restoring of social harmony, but it is typically carried out by men and does not address the need and right of survivors to seek justice [10]. Our work deals with finding evidence in texts generated by the Fijian judicial system that cultural norms have manifested as discriminatory practices in the courts.

The social benefits of detecting gender-based discrimination in court documents are clear. Our natural language processing interest in studying this problem stems from the nature of the documents in question. The court documents we examine in this study are relatively few in number, are of of arbitrary length, and are hand-coded by a small number of domain experts working for an international nongovernmental organization. The small number of labeled documents (under 1,000), the wide range of document lengths (over 10,000 words), and likely variations in applying legal principles on the part of the domain experts who coded the data render the problem of classifying instances of gender based discrimination decidedly non-trivial.

We approached this problem wanting to know if state-of-the-art text classification methods offered substantially better performance over more established shallow and deep learning methods for this particular, somewhat ill-formed data set. We explore the corpus in Section 2. In Section 3 we conduct baseline studies using a somewhat sophisticated shallow classifier. In Section 4 we train convolutional neural networks with two different embedding schemes and apply them to the corpus. In Section 5 we turn our attention to a variety of transformer models. In Section 6 we consider the challenges specific to this corpus and discuss our attempts—frequently unsuccessful—to mitigate them. Section 7 deals with the problem of explainability.

2 The Corpus

Our data are provided by the International Center for Advocates against Discrimination (ICAAD), a non-governmental organization that conducts research and policy advocacy to combat structural discrimination. The data consist of 13,384 court documents from the Republic of Fiji. The documents cover a variety of matters. A subset of 809 documents involving cases of gender-based violence (GBV) are manually labeled with metadata indicating whether the document contains evidence of gender-based discrimination and, if it did, the form that the discrimination took. In general, appeals to outmoded cultural practices and ideas around gender roles that are used to justify a reduction in a perpetrator's sentence are labeled as instances of discrimination.

We apply an 80/10/10 split and divide the corpus into a training set of 647 documents and test and validations sets of 81 documents each. Faced with a paucity of labeled documents, we weighed the benefits of removing the hold-out set entirely to maximize data available during model training. Since many of our methods use validation loss during training as a trigger to adjust the learning rate, we determined that not presenting performance data on a hold out set would unacceptably limit the strength of our conclusions.

The lengths of documents in the corpus varies substantially as is shown in Table 1. Labels are applied to the entire document rather than the sentence level, so determining where offending text lies in the positively-labeled documents requires some degree of analysis.

The 809 documents related to GBV in the corpus contain

Table 1: Characteristics of the corpus

Subset	Documents	Avg length	95%	99%
Complete	13,384	2,218	5,989	11,259
Train	647	1,526	4,005	7,619
Test	162	1,414	3,320	7,203

Table 2: Prevalence of discrimination factors

Factor	Training	Validation
Customary practices	0.25	0.25
Gender stereotypes	0.38	0.32
Other factors	0.28	0.31
Overall positive	0.58	0.59

labels for three sub-classes of gender discrimination. The *customary practices* label identifies instances in which methods of informal arbitration or restitution are improperly taken into account by the court. The *gender stereotypes* label indicates that the court has placed undue weight on perceived aspects of the victim's gender in forming its decision. The third label is the catch-all *other factors*. The prevalence of each factor in the training and validation sets are shown in Table 2. Note that documents can be assigned more than one label.

The corpus also contains a variety of document types. These include decisions, rulings, sentencing documents, and judgements. Each document type serves a different purpose, and some are specific to different parts of the judiciary. Sentences, for instance, are specific to trial courts while judgements are issued by appellate courts. The document types exhibit different general structures and lengths. The length of the documents in the training set disaggregated by document type are presented in Figure 1.

3 Baseline Study - NBSVM

To establish a performance baseline for binary classifcation, we examined classification performance using the naive Bayes support vector machine (NBSVM) classifier implemented in the ktrain library [14]. The NBSVM model [12] seeks to balance the performance of multinomial naive Bayes and support vector machines for text classification. The former are better suited for smaller sections of text while the latter perform better with longer sequences. The model generates embeddings from naive Bayes log-count ratios and uses these as input to a support vector machine model. The embeddings are fixed during training.

The NBSVM performed best with a maximum input length

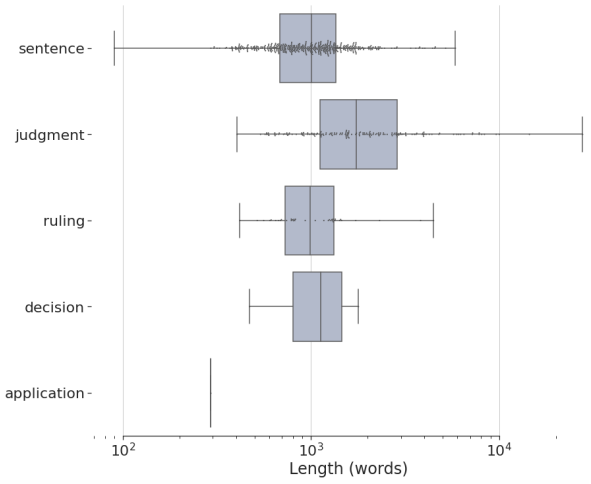

Fig. 1: Distribution of document lengths disaggregated by document type

Table 3: NBSVM Performance

Precision	Recall	F1	Accuracy
0.77	0.89	0.82	0.77

of 8,000 tokens. We conducted training using ktrain's `autotrain` function with a maximum learning rate of 10^{-4}. The `autotrain` function automatically decreases the learning rate when it encouters a plateau in the validation loss. Model performance was consistent for consecutive runs and is summarized in Table 3.

Wang and Manning found that model performance increased when they trained with bigrams. We did not find this to be the case for our corpus. We ran related studies with a Fasttext-like model and a logistic regression model with trainable embeddings. Both models returned performance similar to, but a bit worse than, the NBSVM model. However, these models returned a wider variation of results than did the NBSVM model, likely due to sensitivity to initalizations.

4 Convolutional Neural Networks (CNNs)

Kim demonstrated that relatively simple CNNs can achieve excellent results for sentence classification tasks [15]. We explored the performance of CNNs on the task of detecting gender discrimination in our corpus using two models similar to Kim's. The baseline architecture is shown in Figure 2. The unusual use of a two-element softmax classification head for a binary classification problem allows us to use the ktrain package and does not materially affect the model's results. The model accepts 5,000 input word tokens per document, enough to ingest over 95% of the documents in the corpus in their entirety. Thirty-two filters each are learned for kernel sizes ranging from two to six. The maximum activations of each filter—160 in all—are then concatenated and passed through a 64-neuron dense layer.

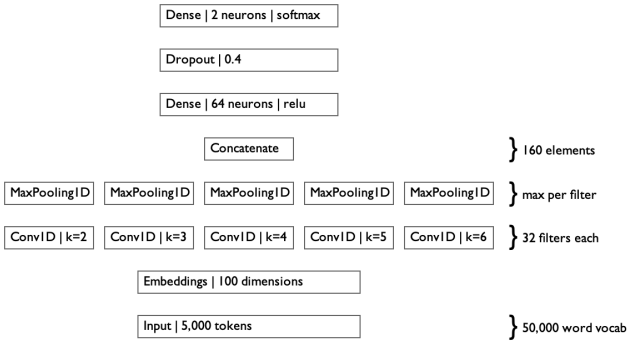

Fig. 2: Baseline CNN architecture showing a 100-dimensional embedding layer for the case of trainable embeddings

Table 4: Comparison of CNN models

Set	Precision	Recall	F1	Acc.
Trainable embeddings				
Validation	0.77	0.89	0.82	0.77
Test	0.68	0.89	0.77	0.72
Ioya2Vec				
Validation	0.66	0.82	0.73	0.64
Test	0.72	0.80	0.76	0.70

4.1 CNN with trainable embeddings

We implemented the model shown in Figure 2 which includes a randomly initialized 100-dimension word embedding layer at the bottom of the model. Overfitting is a significant challenge, likely owing to the relatively small size of the data set which allowed the model easily to learn features unique to individual documents in the training set. We managed overfitting by applying dropout regularization with a probability of 0.4 to the final dense hidden layer. We also applied an implementation of the the triangular learning rate policy proposed by Smith [11] with a maximum learning rate of 10^{-3}. The training routine decreased the learning rate in response to plateaus experienced in the validation loss and applied early stopping when the model was no longer able to produce improvements in validation loss.

Training took roughly 64 second per iteration using an NVIDIA P6000 GPU. This allowed a 0-order parameter space investigation to tune hyperparameters. The best model of this class underperformed the NBSVM as detailed Table 4. Performance on the test set closely matched that returned for the validation set.

The model encoded only a handful of meaningful semantic relationships in the learned embeddings. For instance, *bread* is closest to *breadwinner* in terms of cosine similarity. The terms *breadwinner* and *bread winner* are used interchangeably in several court documents, in the context of the traditional head of a Fijian household. The majority of words in the vocabulary that we explored, however, did not have meaningful associations encoded in their embeddings except tangentially (*daughter* was most closely associated with *care*, for instance).

Noting that that models we built all had similar performance but tended to vary in individual predictions, we built a 10-learner bagged ensemble of CNNs [16]. The ensemble model's performance was substantially worse than that of the single model reported in Table 4.

4.2 CNN with domain-specific embeddings

We used the full corpus of Fijian legal texts to build custom Word2Vec-style embeddings using the skip gram model provided by the gensim package. The resulting loya2Vec (loya being Fijian for lawyer) embeddings cover a vocabulary of nearly 33,000 words. While the texts used to construct the embeddings include those in our training set, we purposefully excluded the texts contained in our validation and test sets from the embedding construction process.

As one might expect, semantic relationships are better captured in Ioya2Vec then they are in the embeddings learned in the previous section. As we had hoped, these relationships are well tailored to the corpus' domains. *Bread* and *breadwinner* are closely related in these embeddings, but so are *Suva* and *Lautoka* (both Fijian cities) and *Sydney* and *Honolulu*.

Starting from the previous CNN architecture, we built the Ioya2Vec mapping into a new embedding layer. As in the case of the CNN with trainable embeddings, the fixed-embedding model uses 32 filters each for kernel sizes two through six. The model's head consists of a 64-neuron dense layer to which a dropout probability of 0.2 is applied. We were surprised to find that the Ioya2Vec-based CNN's performance was not as good as was the model with a smaller trainable embedding layer as shown in Table 4. We strongly suspect that this inversion of expected results is a consequence of the key characteristics of our data set—its small overall size and wide range of document lengths.

5 Transformers

Bidirectional Encoder from Transformers [1] represents the recent development in deep learning techniques for NLP tasks. BERT has been trained on vast amounts of text and as such the BERT pre-trained model can be used with transfer learning by simply applying an additional layer to support the task. The Huggingface library provides pre-trained BERT models which can be fine tuned for text classification. Huggingface also provides variants of BERT such as DistilBert and Longformer which are used in these experiments. For classification tasks the pretrained model and a simple linear layer is added and all parameters are jointly fine-tuned. An alternative approach is to simply use the word embeddings as features and feed these into a simpler downstream

model.

BERT is designed to run against relatively small sequences of text, with a maximum length of 512 word pieces. Typically sentences or documents longer than this are truncated, potentially losing valuable signal information. Given the long form length of documents in the corpus, we implement a number of strategies designed to overcome the token limit.

Firstly, we replicate two of the methods described by Sun et al [6], specifically truncation methods and hierarchical methods. For the truncation approach input documents are truncated to provide smaller inputs, either by taking the first 512 or last 512 tokens. The hierarchical approach divided each document in $k = L/n$ segments which are input into the BERT model where n represents a specified max token length. The representation of each segment is the hidden state of the [CLS] tokens of the last layer (prior to classification). Mean Pooling and Max Pooling are used to combine the representations of each segment.

Secondly we implement the method described by Pappagari et al [5], this takes a similar approach to the hierarchical method, however instead of pooling, the segment outputs are stacked into a sequence which serves as input into a small (100 dimensional) LSTM layer. This is fed into two fully connected layers, a 30 dimensional RELU and then a softmax for the final classes.

The third and final approach is an implementation of Longformer, as described by Beltagy et al [4]. Longformer has been trained on larger documents and can work with a word piece count of up to 4096. Longformer is not described in detail here, in summary it reduces the complexity of the self-attention component by sparsifying the full self-attention matrix according to an attention pattern, this is applied on a fixed sliding window. To improve task specific results, global attention is applied to specific input locations, in the case of classification this is the [CLS] token.

Originally training was done against pre-processed (lower case, removal of special tokens) and unprocessed text. There was no difference in the results as the Bert Tokenizer was able to account for both. Therefore all results are presented from the un-processed input text using cased transformers.

5.1 Truncation, Hierarchical and Recurrent Methods

We used DistilBert and Bert pre-trained models to provide embeddings with a linear layer added for classification. Sanh et al [3] provide a methodology, DistilBERT, that performs nearly as well as BERT but with lower computational cost. BERT (base) consists of 12 layers and 12 attention layers where DistilBert consists of 6 layers and 6 attention layers. We conducted experiments with a batch size of eight—to conserve memory—over 3 epochs using the Adam optimizer and a learning rate of 2^{-5}. This learning rate generally performed the best across BERT models. Training data was split 90/10 for training and validation. BERT models were trained using the hugginface library with one additional layer for classification. We add a linear layer on top of the Bert Pooler for classification.

Table 5: Results of applying truncation; three experiments each were conducted using head and tail tokens with batch size = 8, epochs = 3, and learning rate = 2^{-5}

Method	Acc.	Precision	Recall	F1
Head	0.64	0.69	0.64	0.62
Tail	0.74	0.75	0.74	0.74

Table 6: BERT, DistilBERT Methods

Method	Mean Pooling	Max Pooling	LSTM
Avg Loss - Train	0.83	0.86	0.80
Avg Loss - Val	0.70	0.69	0.75
Accuracy	0.65	0.58	0.58

Truncation methods on BERT were quite effective; using the last 512 tokens provided better results than using the head 512 tokens. This is not surprising as the text in which a judge enumerates mitigating factors—factors which frequently contain evidence of gender-based discrimination—generally appears toward the end of sentencing documents and judgements. The documents in the corpus have already been pre-processed to remove non-useful header and tail information, so truncating the documents has a significant impact on signal processing. Results returned from using the head and tail tokens are provided in Table 5.

For the hierarchical models, mean pooling fared slightly better. Overall the results are disappointing with most of the models predicting all test cases as positive, achieving an unrealistic recall score of 1. It is likely that there is simply not enough documents to train on, and the mean pooling is a better methodology than truncating to divide up the input. More work is needed to investigate further.

5.2 Longformer Method

The Longfomer approach is different from the previous method in that it uses a different method for attention. For pretraining they utilize RoBERTa [2] checkpoint, adding minimal changes required for attention (Beltagy et al. 2020). We utilize Longformer base consisting of 12 hidden layers and the Huggingface LongformerForSequenceClassification library which includes a Linear layer for classification.

Longformer was fine-tuned on two separate systems, one with a single NVidia K80 GPU and one with a single NVIDIA P100. Memory limitations prevented us from utilizing the full 4096 token segment length, even with 1024 tokens, the K80 was only able operate with a batch size of 1 (i.e. Stochastic gradient descent) whereas the P100 could

operate a batch size up to 4. Multiple experiments were conducted with various learning rates, batch sizes and epochs. In all experiments a larger batch size was preferable over a larger segment length, and 3 epochs performed adequately. All Logformer methods were promising providing better results than the BERT/DistilBERT truncation methods. Results are provided in Table 7.

For each experiment, the data was divided into 562 training samples, 65 validation samples and 162 test (holdout) samples. All experiments run with dropout 0.2 and attention dropout 0.2

Interestingly increasing batch size and token length has a slight improvement on F1and recall but at the cost of precision. Considering the BERT models generally prefer to predict all positive results (higher recall) we need to be careful in evaluation. The practical implication of a false positive outweigh a false negative when predicting sentencing judgements relating to gender based discrimination.

With more powerful compute it would be interesting to see if these results can be improved upon with the recommended batch size (16 or 32) and by utilizing more of the document text.

6 Why was this hard? The challenge of heterogeneous data

None of our modeling efforts produced stunning results, and we were left to ponder the features of this data set that made classification so challenging. We considered the possibility that the BERT- and DistilBERT-based models suffered from their inability to ingest more than 512 tokens. Of note, 512 tokens generally maps to far fewer than 512 words as the BERT family tokenizers handle unfamiliar words by splitting them. As described above, we had tried steering BERT at different parts of the documents, but the results were unsatisfying.

The CNN models are able to ingest far more text than our transformer models, so we used these to search for the maximum signal in the documents. To do this, we used the CNN with trainable embeddings trained on documents with a length of 5,000 tokens. We divided the validation set documents into 500-word snippets in rolling windows starting every 100 words. We then ran infrencing on these snippets. We plotted the accuracy on the batch of snippets against the snippet start location expressed in tokens from the start of the document. We repeated this procedure with the documents in reverse order such that the resulting plot was of the accuracy on the batch of snippets plotted against the starting position referenced to the end of the documents. The results, provided in figure (x) suggests that the portions of the documents containing text that yields a positive prediction are mostly located at the TODO.

Suspecting that the structure of a document may be a function of the document's type, we repeated the above procedure with subsets of the validation set grouped by document type the documents by type. The results, presented in figure (x), suggest that the location of the text in a document that leads to a positive prediction is associated with the document's type. The sentencing documents examined have signal near the beginning and end of the documents, while the signal in the judgement documents examined is in the middle. There were very few TODO documents, so it is difficult to draw conclusions about their structure, but those examined had text roughly 1,500 words from the beginning and end that provided the best predictions.

We tried building a classifier to take advantage of our understanding of the structure of the various document types. We used a representative sample of 500 tokens for each document. For sentences, we took a 500-token snippet starting at the first token. For judgements, rulings, and decisions, we took our snippet starting at the 1,500th token. We used the existing CNN with trainable embeddings model as the base of this location-aware classifier. It under-performed the simple CNN.

A problem with drawing strong conclusions about the observations above is that the filters that the model learns are highly dependent on initializations. To demonstrate this, we trained another instance of the CNN with trainable embeddings trained using the same training methodology as above. This model's performance is very similar to the previous model's, but the locations where it finds evidence to support a positive classification in the sentencing documents is quite different, as demonstrated in figure (x).

7 Considering Explainability

While not a formal accusation of judicial misconduct, labeling a court document as containing evidence of bias certainly carries some weight. Before applying a model to the task of potentially impugning the character of a sitting judge, we should provide a qualitative assessment of our predictions. The LIME algorithm [13] provides a method for doing this via automated perturbation analysis. In the case of text, LIME computes the sensitivity of a prediction to the input text by iterativly removing a single word from the input text and repeating the prediction.

As an illustrative example, we consider the sentence *As the accused is the sole breadwinner for his family, I reduce his sentence by two years.* Taken by itself, and applying our understanding of ICAAD's concerns about Fijian jurisprudence, we expect that this sentence would be classified as containing evidence of gender based discrimination due to its appeal to the accused's patriarchal role in his family.

Green text in Figure 3 indicates words that support the prediction that the text contains evidence of gender based discrimination. The deeper the green, the stronger the positive association. Red text similarly indicates negative associations. We see certain similarities in how the models treat the text. All three of the simple models recognize words associated with appeals to family status with evidence of gender based discrimination. Perhaps more interesting is an exami-

Table 7: Longformer Experiments

Max Len	Epochs	Batch Size	LR	Experiments	Avg Acc	Avg Precision	Avg Recall	Avg F1
512	5	8	2e-5	5	0.70	0.71	0.81	0.76
512	3	8	2e-5	1	0.69	0.71	0.81	0.76
512	5	8	4e-5	4	0.69	0.70	0.81	0.75
512	3	8	2e-5	2	0.71	0.72	0.84	0.78
512	3	12	2e-5	1	0.76	0.67	0.89	0.77
1024	3	6	2e-5	1	0.69	0.69	0.87	0.77

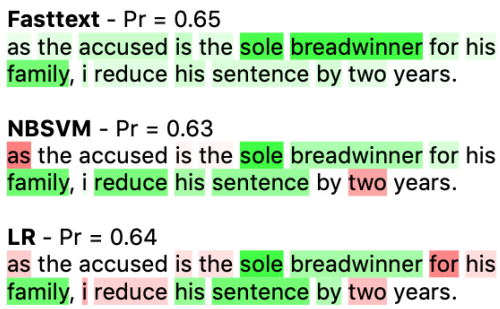

Fig. 3: Application of the LIME algorithm to predictions generated by a variety of models

nation of the differences, especially in the phrase *I reduce his sentence by two years*. Also noteworthy is the association of the word *accused* in the Fasttext model with discrimination.

References

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

[2] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Bert: RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv preprint arXiv:1907.11692*, 2019.

[3] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

[4] Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The Long-Document Transformer. *arXiv preprint arXiv:2004.05150*, 2020.

[5] Raghavendra Pappagari, Piotr Żelasko, Jesús Villalba, Yishay Carmiel, and Najim Dehak. Hierarchical Transformers for Long Document Classification. *arXiv preprint arXiv:1910.1078*, 2019.

[6] Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. How to Fine-Tune BERT for Text Classification? *arXiv preprint arXiv:1905.05583*, 2019.

[7] Guy Hadash, Einat Kermany, Boaz Carmeli, Ofer Lavi, George Kour, and Alon Jacovi. Estimate and replace: A novel approach to integrating deep neural networks with existing applications. *arXiv preprint arXiv:1804.09028*, 2018.

[8] Armand Joulin, Edouard Grave, Piotr Bojanowski and Tomas Mikolov. Bag of Tricks for Efficient Text Classification, 2016; arXiv:1607.01759.

[9] George Kour and Raid Saabne. Fast classification of handwritten on-line Arabic characters. In *Soft Computing and Pattern Recognition (SoCPaR), 2014 6th International Conference of*, pages 312–318. IEEE, 2014.

[10] Lynda Newland. Villages, Violence and Atonement in Fiji. In A. Biersack, M. Jolly and M. Macintyre (Eds.), *Gender Violence and Human Rights*, Australian National University Press, 2017.

[11] Leslie N. Smith. Cyclical Learning Rates for Training Neural Networks. *arXiv:1506.01186v6*, 2017.

[12] Sida Wang and Christopher Manning. Baselines and Bigrams: Simple, Good Sentiment and Topic Classification. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 90–94. Association for Computational Linguistics, 2012.

[13] Marco Ribeiro, Sameer Singh and Carlos Guestrin. "Why Should I Trust You?" Explaining the Predictions of Any Classifier. *arXiv:1602.04938v3*, 2016.

[14] Arun Maiya. ktrain: A Low-Code Library for Augmented Machine Learning. *arXiv:arXiv:2004.10703*, 2020.

[15] Yoon Kim. Convolutional Neural Networks for Sentence Classification. *arXiv:1408.5882*, 2014.

[16] Ian Goodfellow, Yoshua Bengio and Aaron Courville. *Deep Learning*. pp 249-251. MIT Press, 2016.