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 specifically considers the problem of identifying evidence of gender based discrimination in Fijian court records for cases of gender based violence (GBV). The problem of GBV in Fiji is serious and well documented, with 64% of Fijian women facing 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 restoration of social harmony [10]. Our work deals with finding evidence that cultural norms have manifested as discriminatory practices in Fijian 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 handful of domain experts working for a foreign nongovernmental organization applying their best judgement after the fact. 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. In Section 2 we provide more details of the corpus. In Section 3 we conduct baseline studies using three somewhat sophisticated shallow classifers. In Section 4 we train convolutional neural networks with two different embedding schemes. In Section 5 we turn our attention to variety of transformer models. In Section 6 we provide a brief treatment of explainability.

2 The Corpus

Our data are provided by the Institute 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 divide the corpus into a training set of 647 documents and a validation/test set of 162 documents. Faced with a paucity of labeled documents, we weighed the benefits and costs of further dividing the corpus into training, validation, and test sets. Every model presented below uses intermediate predictions on the validation set to adjust learning rates dynamically. Building a validation set from the training data impaired training, in some instances significantly. We surmise that our corpus is so small that model results are sensitive to reducing the size of the training set. Proceeding, therefore, with a unitary validation we are careful not to claim that our models have been rigorously tested on held out data.

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 a degree of analysis.

Table 1: Characteristics of the corpus

Subset	Documents	Avg words	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: Comparison of baseline models

Model	Precision	Recall	F1	Accuracy
Max class	0.60	1.0	0.75	0.60
LR	0.85	0.76	0.80	0.78
Fasttext	0.77	0.81	0.79	0.75
NBSVM	0.77	0.89	0.82	0.77

3 Baseline Studies

A model that predicts the majority training set class yields an accuracy of 0.60 on our test set. Moving beyond this simplest case, we want to understand what can be accomplished with a relatively sophisticated class of shallow models before exploring more interesting deep architectures.

3.1 Logistic Regression with Trainable Embeddings

Our first model is a simple logistic regression model with an embedding layer that encodes a 50,000 word vocabulary of unigrams as two-dimensional embeddings. The model computes embeddings for the first 1,000 tokens in each document and then sums those embeddings row-wise to yield inputs to a two-element softmax layer.

The logistic regression model is sensitive to initialization but yields surprisingly good performance. Performance for all three models considered in this section is provided in Table 2. We found, that expanding the model to include bi-grams and increasing the input length decreased performance.

3.2 A Fasttext-like model

The Fasttext architecture [9] consists of a trainable embedding layer that processes the first 1,000 unigram tokens in each document into 64-dimensional embeddings. Dropout is applied to the result to reduce overfitting. The embeddings are then averaged and normalized to form a single 64-dimension vector which is used as input to a two-dimensional softmax classifier. The NBSVM model was a good deal more sensitive to initialization than was the logistic regression model. Optimizing hyperparameters led us to limit input lenghts to 1,000 unigram tokens. Expanding beyond unigrams or increasing the input length did not yield better results.

3.3 NBSVM

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 model generated from our training data is relatively insensitive to initializations. Best performance was achieved for inputs of 8,000 unigram word tokens. While we do not doubt the authors' claim that the model generally performs better when applied to bigrams than unigrams, we did not find this to be the case when applied to our data.

4 Convolutional neural networks (CNNs)

CNNs apply convolutional filters to inputs to learn features. Kim demonstrated that a simple convolutional model with pre trained word embeddings can achieve excellent results when applied to classifying sentences. We explored the use of CNNs with two embedding schemes.

4.1 A CNN with trainable embeddings

We implemented a version of Kim's architecture with a trainable embedding layer in Keras with the ktrain package. This architecture, shown in figure (x), includes a randomly initialized 100- dimension word embedding layer at the bottom of the model. 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 are then concatenated and passed through a 64-neuron dense layer. Overfitting is a significant challenge, likely owing to the relatively small size of the data set. Applying dropout with a probability of 0.4 helps manage this challenge. We trained the model using the triangular learning rate policy proposed by Smith [11] with a maximum learning rate of 1e-2.

Fast training - roughly 64 second per iteration using an NVIDIA P6000 GPU - allowed a 0-order parameter space investigation to tune hyperparameters. The model produced encouraging results, as detailed in Table 3 when applied to the validation set. While the model's performance was locally insensitive to changes in hyperparameters, we must again emphasize that our study is hampered by having insufficient data to provide a true hold-out test 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. This is a natural connection that can be drawn from the training set as a perpetrator's status in the family is considered by Fijian courts. The majority of words in the vocabulary that we explored, however, did not have meaningful associations encoded in their embeddings except tangentially (daughter

Table 3: Comparison of CNN models

Embeddings	Precision	Recall	F1	Acc.
Trainable	0.85	0.72	0.78	0.76
Ioya2Vec	0.78	0.85	0.76	0.78

was most closely associated with care, for instance).

4.2 A 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/test set 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.

We used Ioya2Vec as the embedding layer for a CNN. The tuned 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. 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

For these models, DistilBert and Bert pre-trained models were used to provide embeddings with a linear layer added for classification. Sanh et al [3] provide a methodology, DistilBert, that is almost as performant as BERT but with lower computational cost. BERT (base) consists of 12 layers and 12 attention layers where DistilBert is half that. All experiments were run with a batch size of 8 (to conserve memory), a learning rate of 10e-6, Adam optimizer and 3 epochs. Training data was split 90/10 for training and validation.

As expected the results for the truncation methods did not provide good results. 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. Mean pooling fared slightly better than other models, including LSTM. Overall the results are disappointing with most of the models predict-

ing 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. We present results in Table 4 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 5.

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 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.

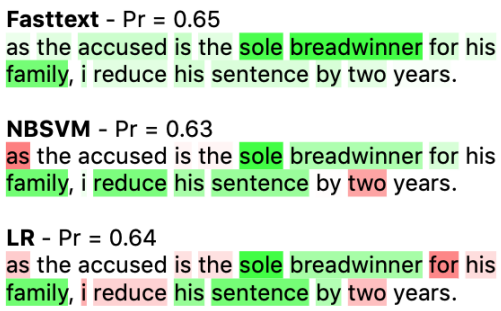

Fig. 1: Application of the LIME algorithm to predictionss generated by a variety of models

Green text in Figure 1 indicates words that support the prediction that the text contains evidence of gender based discrimination. The deeper the green, the stronger the positive assication. 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 examination 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,

Table 4: BERT, DistilBERT Methods

Method	Head	Tail	Mean Pooling	Max Pooling	LSTM
Avg Loss - Train	0.79	0.73	0.83	0.86	0.80
Avg Loss - Val	0.61	0.76	0.70	0.69	0.75
Accuracy	0.58	0.58	0.65	0.58	0.58

Table 5: 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

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 preprint 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 preprint arXiv:1602.04938v3*, 2016.