Detecting Hate Speech on Twitter

Taurean Parker and Caroline Roper

December 2018

1 Introduction

Our research studies text classification in the context of detecting hate speech, prejudice, and other offensive rhetoric within tweets. The research presented here is the first phase of a larger research project to be completed by NYU's Social Justice Lab. The larger purpose of our research is to investigate the "Ideological-Conflict Hypothesis", which suggests that liberals and conservatives express similar levels of intolerance toward 'ideologically dissimilar and threatening groups'[1]. The authors of the "Ideological-Conflict Hypothesis" argue that the "psychological underpinnings of intolerance and prejudice — e.g., biased reasoning, the desire to promote cherished values, and perceived social threat — are not exclusive to people on either end of the political spectrum...both liberals and conservatives express similar levels of intolerance and prejudice towards ideologically dissimilar and threatening groups."

The Social Justice Lab disagrees with the notion that prejudice is symmetrical across the ideological spectrum. The authors of "The Ideological-Conflict Hypothesis" primarily relied on experimental data, but social media data presents an opportunity to analyze trends at a larger scale. With this in mind, the Social Justice Lab is conducting a study of hate speech, prejudice, and offensive language expressed by Twitter users on both sides of the political spectrum.

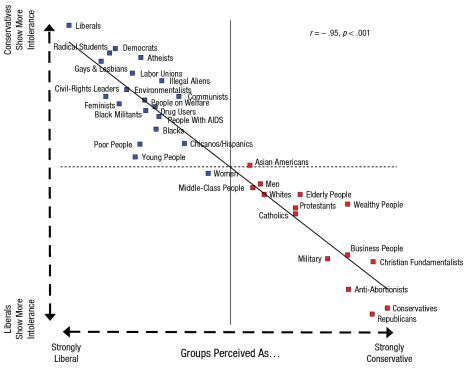

Figure 1: Representation of Ideological-Conflict Hypothesis[1]

In order for the Social Justice Lab to complete their research, they first needed data about the presence of threats, obscenity, hate speech, name-calling, stereotypes, and negative prejudice in tweets authored by left and right-leaning Twitter users. These categories are not considered mutually exclusive or dependent on each other. Annotators working with the Social Justice Lab have annotated 7k tweets with these categories, and there are about 734k remaining tweets to be annotated using machine learning methods.

The complete definition of each of the labels is found in the excerpt from the coding manual in the appendix. Below is an abbreviated paraphrase from the annotator coding manual of the most important labels:

- Threat - language containing a threat to use physical violence, or intimidation, directed at an individual or a group
- Obscenity - an offensive word or phrase that is considered inappropriate to use in professional settings
- Hate Speech - a communication that carries no meaning other than the expression of hatred for some group
- Name-Calling - language directed at an individual or group that is mocking, insulting, demeaning, or embarrassing
- Stereotypes - false or misleading generalizations about groups held in a manner that renders them largely, though not entirely, immune to counter-evidence
- Negative Prejudice - "[a] feeling, favorable or unfavorable, toward a person or thing, prior to, or not based on, actual experience", an antipathy based on group-based generalizations and stereotypes

It is also necessary to label tweets as to whether they were non-English and whether they contained pornography in order to eliminate any biases in the data by limiting the scope to English-language, non-pornographic tweets.

In this research, we implement several text classification strategies, from "bag of words" models such as SVM, to Neural Networks such as LSTM, to transfer learning techniques such as the ULMFiT and BERT. We apply each strategy to each of the labels independently.

Our training, validation, and test sets come from the 7k annotated tweets. We will also discuss the process of applying the models to the 734k tweets in the full research dataset.

2 Related work

We were inspired by domain-specific literature on the subject of hate speech detection and also by literature on transfer learning.

2.0.1 Hate Speech Detection

Davidson et al[2] studied bag-of-words models in the context of hate speech detection. They note that, when attempting to detect hate speech, especially using bag-of-words approaches, many false positives occur in the presence of offensive language. They compare several multiclass classification models that identify hate-speech, offensive language, and neither. They implement logistic regression, naive Bayes, decision trees, random forests, and linear SVMs, and found that linear SVMs performs the best. Their research provides reasoning for our choice of linear SVM as our baseline model.

Another SVM-based approach, the research of Hasanuzzaman et al[3] uses a SVM with text-based features and demographic characteristics. The intuition is that words have a different meaning and connotation depending on the identity of the speaker. To obtain the gender and age of the Twitter user, authors use a model trained on data from 70k Facebook users that predicts gender and age. To obtain the Twitter user's location, they use the location information from the stored user's profile. Each word is represented as a concatenation of the word embedding and the embeddings representing the demographic variables for age, gender, and location, and then fed to a linear SVM. They find that the demographic variables substantially boost performance.

Badjatiya et al[4] compare a number of methods for detecting hate speech on Twitter, including SVM, gradient boosted decision trees, convolutional neural networks, and LSTM models. They use a dataset of 16k tweets labeled as racist, sexist, or neither. All neural network approaches outperformed all linear mod-

els. Closely related to our own research, they achieve an f-score of 0.81 using an LSTM initialized with publicly available Stanford NLP GLoVe embeddings. They have a larger dataset than our own, and their classes are also less imbalanced (12% labeled racist, 21% labeled sexist), which may explain why they achieved higher performance with a similar architecture. Nevertheless, their research demonstrates that neural network approaches tend to outperform linear models, and that pre-trained vectors also boost performance, both findings we applied in our own research.

Zeerak Waseem[5] studied the effects of annotator influence by comparing the performance of models trained on expert annotations with those trained on amateur annotations. They define expert annotators as feminists and anti-racism activists, and use CrowdFlower to gather amateur annotators. They find low inter-annotator agreement among amateurs annotators, and their best model trained on expert annotations achieved an f-score of 91.19 compared to 83.88 for amateur annotators. Their research underscores the importance of annotator quality and suggests that it may be difficult to compare hate speech detection models across different datasets and annotation approaches.

Though we did not explore a convolutional neural network approach in our own work, research at the Norwegian University of Science and Technology identified that this could be a successful strategy [6]. Using a convolutional neural network and a dataset of a similar size to our own (6.7k annotated tweets), they achieved an f-score of 0.78 using a CNN initialized from pre-trained word2vec vectors.

3 Methods

3.1 Data

Researchers from the Social Justice Lab oversaw the process of annotating a subset of the tweets to use as training data.

The tweets were collected from the Twitter API using searches for certain keywords. These keywords correspond to groups that may be targets of hate speech and they include groups who are perceived as left-leaning and right-leaning, based on prior research.

The appendix contains an exhaustive list, but below we provide examples:

- Perceived as left-leaning: left-wing, underprivileged, non-believers, black people, queer

- Perceived as neither left-leaning nor right-leaning: millenials, teenagers, latinas, latinos, middle income
- Perceived as right-leaning: executive, navy, fascists, Christians, veterans

There were 7k labeled tweets randomly selected from the 734k tweets and assigned to 3 or 4 annotators. The annotators were each trained by the same researchers using the same manual, but due to turnover, the same raters were not available for the entirety of the labelling process.

We reserved 20% (1.4k) of the tweets as a test set to evaluate final performance. Of the remaining 5.6k tweets, 20% (1.1k) were used as validation, leaving $\tilde{4}$.5k tweets for training.

To create the labels, we relied on majority voting, where if 2 annotators agreed that the tweets contained hate speech, obscenity, etc. then it was labeled as belonging to the positive class. The prevalence of each class in the training data is summarized below:

Label	Percentage
hatespeech	2.4%
namecalling	7.3%
negpreudice	13.3%
noneng	8.3%
obscenity	4.8%
porn	0.9%
stereotypes	8.0%
threat	2.8%

3.2 Text Processing Techniques

In rare cases, the set contained identical tweets because of retweets, but because the retweets are not guaranteed to be annotated in an identical way, we preserved the each tweet in order to train the model on all available annotator choices.

We experimented with two strategies for hyperlinks and @ mentions - removing them completely or replacing them with generic tokens 'x_user_mention' and 'x_url'. Other preprocessing choices were standard - lowercasing, tokenizing, and removing html remnants.

3.3 Support Vector Machines

Support Vector Machines is a popular supervised method for classification. In addition to adopting the text processing techniques above, we removed common stop-words from the corpus and then applied term frequency–inverse document

frequency, a statistical method that adjusts the count occurrences of words based on the frequency within the different text documents. We explored different regularization techniques and created n-grams from from size 1 to size 5.

3.4 LSTM

Our LSTM model is similar to the architecture described in Badjatiya et al. We initialized our embeddings from two sets of pre-trained vectors. The first is a a set of GLoVe vectors made available by Pennington et. al[7]. We chose the embeddings pre-trained on 2B tweets for high domain relevance. The second is a set of vectors we trained on our own data, using open-source code[8]. We trained for 50 epochs on all 734k tweets in our data, with no minimum requirement for the number of times the word appears in the corpus. When the model initializes the embeddings, it first looks for each word among the publicly available pre-trained vectors, then, if not found, it finds the word among our own pre-trained vectors.

The model has one LSTM layer and a dropout layer with $p = 0.20$ for regularization.

3.5 ULMFiT

The ULMFiT model[9] was developed by researchers at fast.ai, a research institution dedicated to making deep learning more accessible. The ULMFiT model sought to apply techniques used in transfer learning for computer vision to the NLP discipline.

The ULMFiT model consists of three stages:

1. General-domain LM pre-training, which uses unlabeled data from a general domain.
2. Target task LM fine-tuning, which uses data from the same distribution as the labeled data, but which may be unlabeled.
3. Target task classifier fine-tuning, which uses the labeled data.

The general-domain language model is pretrained on Wikitext-103, which consists of 28,595 preprocessed Wikipedia articles.

Each phase of the model has a similar architecture, an LSTM-AWD[10]. The LSTM-AWD (ASGD Weight Dropped) is an LSTM with additional innovations that primarily serve to regularize the model.

The LSTM-AWD uses averaged stochastic gradient descent (ASGD) as its optimizer. Like SGD, ASGD computes a stochastic gradient at each step, but

rather than returning the weights resulting from the last step, the algorithm returns an average of the weights over the last few steps. The interval over which to take an average is computed in a dynamic way depending on the validation perplexity.

The LSTM-AWD also uses weight-dropping, which differs from the most common way that dropout is applied. Dropout temporarily removes some randomly selected units from a network to promote independence rather than co-adaptation between feature maps.

To differentiate between weight-dropping and a more traditional application of dropout, we review the LSTM architecture. The mathematical formulation of the LSTM is below:

$i_t = \sigma(W^i x_t + U^i h_{t-1})$
$f_t = \sigma(W^f x_t + U^f h_{t-1}$
$o_t = \sigma(W^o x_t + U^f h_{t-1}$
$\widetilde{c_t} = tanh(W^c x_t + U^c h_{t-1})$
$c_t = i_t \cdot c_t + f_t \cdot + \widetilde{c}_{t-1}$
$h_t = o_t \cdot tanh(c_t)$

It's common for LSTM-based models to have a dropout layer that operates on the last hidden state of the LSTM h_t prior to mapping the hidden layer to the outputs. Our standalone LSTM implements this strategy.

Weight-dropping however, is dropout applied within the recurrent connections of the RNN. Weight-dropping acts on the hidden-to-hidden weight matrices within the LSTM, $[U^i, U^f, U^0, U^c]$ in the mathematical formulation above.

The ULMFiT model uses a 3-layer LSTM-AWD for each phase - language modeling, language fine-tuning, and text classification. The diagram below illustrates this.

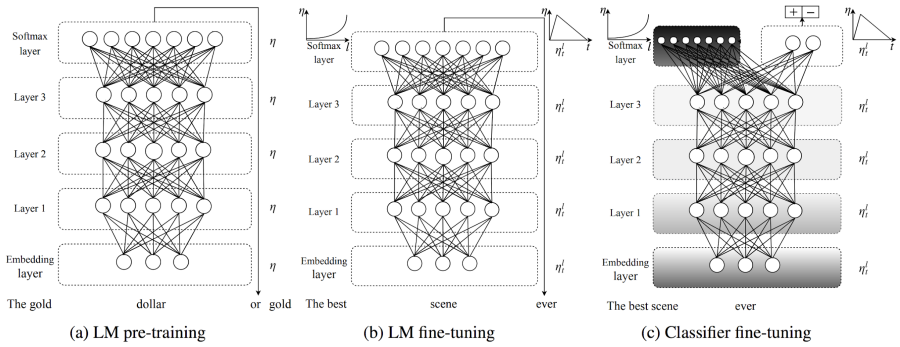

Figure 2: ULMFiT diagram[9]

The ULMFiT model has been successfully applied to the problem of hate speech detection. Rother and Rettberg[11] applied ULMFiT to the problem of hatespeech detection in a very similar configuration to our own. Using a dataset of only 5k labeled tweets, 303k unlabeled tweets, and the publicly available model pre-trained on Wikipedia data, they achieved an average validation f-score of 0.8. They also implemented a few improvements on Howard et. al's original version of the ULMFiT model. They used a learning rate finder rather than relying on Howard et. al's fixed "discriminative learning rate" schedule. They also greatly reduced the dropout during the language-medium model phase (testing multipliers of 0.7 and 0.3) and greatly increased it during the classifier stage, using a multiplier of 1.8. (Increasing the classifier dropout multiplier did not improve performance in our case.)

For our implmentation of the ULMFiT model, we used publicly available code released by the creators of ULMFiT[12]. We used the pre-trained language models available here. We ultimately used only the forward models. We fine-tuned the language model on all 734k tweets, before training the classifier on the labeled tweets.

3.6 BERT

Bidirectional Encoder Representations from Transformers or BERT is an innovative state-of-the-art language representation model [13]. Developed by researchers at Google AI Language, BERT creates a "deep bidirectional representation" of language - meaning that the representation of the language is contextualized, with each word conditioned on the words to its left and its right. A traditional language model is built by optimizing an objective function that seeks to accurately predict the next word, given the preceding context. BERT instead randomly "masks" words and seeks to predict the masked word given the context on the left and on the right.

The BERT model is "deeply bidirectional," which differs from other similar language models. There is a bidirectional version of the ULMFiT model; it's possible to pretrain both a forward and a backward language model, train two classifiers independently, and average the classifier prediction. (In our implementation, we didn't use bi-directionality as it doubles the computational costs.) BERT, in contrast, creates one united bidirectional model rather than ensembling two independent models trained in each direction.

BERT's authors contrast their approach to that of OpenAI GPT and ELMo. Though there are differences between the models, ELMo and ULMFiT both involve independently training left-to-right and right-to-left LSTMs. ELMo uses the concatenation of the LSTMs from each direction to generate features for downstream tasks. OpenAI GPT doesn't use bi-directionality.

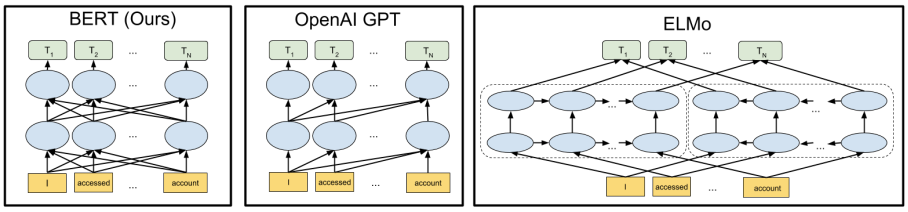

Figure 3: BERT diagram[13]

BERT is also pre-trained in such a way to build sentence and paragraph-level comprehension. BERT is trained on the task of seeking to predict the next sentence. This particular innovation is probably not as crucial for classifying Twitter data due to its brevity, but BERT is intended to serve as a basis for a wide variety of NLP tasks.

BERT uses units call transformers, as originally implemented by Vaswani et al[14]. The transformer is an alternative to convolutional and recurrent architectures that builds on the concept of multi-head attention. Traditional attention mechanisms in sequence-to-sequence models establish a correspondence between units of the input and units of the output. Multi-head attention can relate parts of a single sequence to each other, within either the input or the output.

The BERT model also represents language as word parts, not just full word tokens. So for example, it divides the word "mongering" into "mon" "ger" "ing." This isn't particularly novel, as techniques such as bipartite encodings of words have become common in NLP research, but it is especially relevant to Twitter data. Twitter data often contains mispellings and abbreviations, and hashtags are often comprised of several words combined without a space, so tokenizing only on words properly divided by spaces is limited.

Each sequence is represented as a series of token embeddings (the word or word part itself), segmentation embeddings (which identify which sentence the token belongs to), and position embeddings (which identify the order of the token within the sequence). Each sequence also begins with a special classification embedding, which is used for aggregate sequence representation.

We use BERT for sequence classification, which applies a classifier on top of the BERT Model. The classifier itself is extremely simple and shallow, relying extensively on the language understanding of BERT. The final hidden state of the first token in the special classification embedding provides a pooled representation of the sequence. The classifier applies dropout to this pooled representation, followed by a linear layer mapping between the result to the possible classes.

To implement our version of the BERT model, we used the publicly available PyTorch code. Although the original authors of BERT used TensorFlow, they have formally endorsed the PyTorch implementation, and experiments have verified that it produces identical results.[15]. We start from the publicly available BERT model, pretrained on the BooksCorpus (800M words) and English Wikipedia (2,500M words).There are two publicly available versions of the BERT model, a large and a base. The large version has 16 attention heads and 24 layers, while the base version has 12 attention heads and 12 layers.

Below are the model specifications in greater detail. These all pertain to the pre-trained BERT, not our fine-tuned classifier:

Parameter Large	Definition	Base
attention-probs-dropout-prob	The dropout ratio for the attention probabilities	0.1
hidden-act	The non-linear activation function in the encoder and pooler	"gelu"
initializer range	The standard deviation of the normal distribution initializer for initializing weight matrices	0.02
max-position-embeddings	The maximum sequence length that this model might ever be used with	512
vocab-size	Size of vocabulary	30,522

4 Results

4.1 SVM

Before exploring deep learning methods on the Twitter data, we hypertuned a few different SVM models as a first step to understand the relationship between

the different offensive speech labels. The table below represents our results. Overall, the SVM model produced sufficient results for the hatespeech and obscenity categories, but performed terribly at predicting stereotypes, threats, and negpredudice. The difference in performance between two different SVM models is partly due to sparse representation of our dataset, and L1 regularization typically performs better for such representations. This is evident in our results, as for most of the offensive speech categories, L1 outperformed L2 regularization.

Label	SVM(L1 Penalty)	SVM(L2Penalty)
Hatespeech	.670	.615
Namecalling	.144	.335
Negprejudice	.262	.367
Noneng	.337	.645
Porn	.495	.600
Obscenity	.703	.634
Stereotypes	.169	.221
Threat	.144	.198

4.2 LSTM

Below are the results of parameter tuning for the LSTM model, predicting hate speech. When we "remove" the mention and the url, we remove all @mentions and urls from the model. When we "replace" them, we use "x_mention" and "x_url" as tokens in place of the specific mentions and urls. Increasing the hidden size seems to be the most positively impactful hyperparameter choice.

mention + url	seq length	vocab	hidden size	batch size	f1
remove	25	10k	200	32	0.615
remove	30	10k	200	32	0.653
replace	25	10k	200	32	0.640
replace	30	20k	200	32	0.667
replace	40	10k	200	32	0.682
replace	40	12k	200	32	0.696
replace	40	12k	400	16	0.744
replace	40	12k	400	32	0.750

For the other labels, only the batch size was further tuned, using a model with a sequence length of 40, a vocabulary of 12k words, and a hidden size of 400.

label	batch size 32	batch size 16
hatespeech	0.750	0.744
namecalling	0.493	0.468
negprejudice	0.462	0.468
noneng	0.920	0.941
obscenity	0.615	0.727
porn	0.500	0.667
stereotypes	0.376	0.388
threat	0.359	0.312

4.3 ULMFiT

Below are the results for tuning the ULMFiT model. The first two language models were trained with the default parameters. Then two more language models were trained with a larger vocabulary, a much shorter bptt, and higher dropout. The back-propogation through time parameter controls how many words the model conditions on to predict the next word. The default value of 70 would usually span more than one tweet, so we shortened it to only 5 words. The classifier overfits quickly and more than 3 epochs tends to reduce performance. A lower learning rate is also helpful.

Below is a table summarizing the language models fine-tuned:

ID	Dropout	Epochs	BPTT	Vocab
A	1	20	70	30k
B	1	35	70	30k
C	0.70	20	10	60k
D	0.70	20	5	60k
E	0.70	7	5	60k

Below is a table summarizing the classifiers trained:

LM ID	Classifier ID	bptt	dropout	epochs	learning rate	batch	f1
A	1	70	1	1	0.01	64	0.640
A	2	70	1	2	0.01	64	0.642
B	3	70	1	3	0.01	64	0.667
A	4	70	1	5	0.01	64	0.654
C	5	40	1	3	0.01	64	0.667
D	6	40	1	3	0.01	64	0.702
D	7	40	1	4	0.01	64	0.610
D	8	40	0.70	3	0.01	64	0.655
E	9	40	0.70	3	0.005	64	0.681
D	10	40	0.70	3	0.005	64	0.634
E	11	40	0.70	3	0.005	32	0.710

Classifiers 5 and 6 are identical except for the bptt of the underlying language model. The classifier using language-model D, fine-tuned with a bptt of 5, performed better than the classifier using language model C, fine-tuned with a bptt of 10.

Classifiers 9 and 10 are identical except for the length of time that the underlying language model was trained. Language model D was trained for 20 epochs while language model E was trained for 7 epochs. Counter-intuitively, the language model trained for less time produced the higher-performing classifier.

We tuned primarily with respect to the hate speech label. The performance on all labels is below. These results correspond to the parameters of classifier 11.

label	score
hatespeech	0.710
namecalling	0.307
negprejudice	0.326
noneng	0.909
obscenity	0.594
porn	0.640
stereotypes	0.258
threat	0.364

4.4 BERT

Below are the results for tuning the BERT model. The creators of BERT recommend experimenting with batch sizes of 16 and 32, learning rates of $5e^{-5}$, $3e^{-5}$ and $2e^{-5}$, and epochs 3 and 4. We've run 6 of the 12 possible combinations, and also experimented with choosing a smaller batch size and learning rate than BERT's authors would typically recommend.

All models in the results below use the large pre-trained BERT model. An undefined f-score occurs when no correct positive class predictions are made. Because our classes are very imbalanced, this often indicates that the model didn't predict any incidences of the positive class. The original authors of the BERT paper note that, although the large version of the BERT model tends to be more accurate, even on small datasets, sometimes fine-tuning is unstable and produces de-generate results. The iterations in which no correct positive-class predictions were made demonstrate this.

epochs	learning rate	batch size	f1
4	$3e^{-5}$	32	undefined
4	$2e^{-5}$	32	0.714
4	$2e^{-5}$	16	undefined
3	$3e^{-5}$	32	undefined
3	$2e^{-5}$	32	0.682
3	$2e^{-5}$	16	0.732
3	$2e^{-5}$	8	0.727
3	$1e^{-5}$	16	0.681

The tuning results suggest that 3 epochs, a learning rate of $2e^{-5}$, and a batch size of 16 perform well. However, when we ran this tuned model on the other labels, we encountered many degenerate results from using the large model with a small dataset. Obscenity, namecalling, negative prejudice, and nonenglish all produced undefined f-scores. The original authors of BERT circumvented the issue of degenerate results by experimenting with several random initializations until one version succeeded. This strategy would be somewhat impractical for us, so we instead examined the validation scores using the same parameters on the "base" version of the model.

label	f-score
hatespeech	0.721
namecalling	0.553
negprejudice	0.493
noneng	0.966
obscenity	0.741
porn	0.700
stereotypes	0.330
threat	0.423

4.5 Model Comparison

The results of the best-performing model in each category are below.

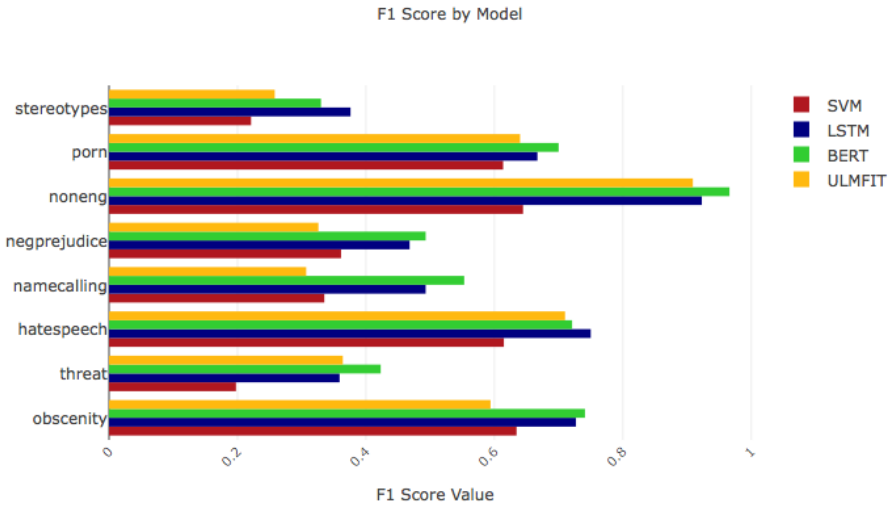

Figure 4: F1 diagram

We make the following observations:

- We achieved significant improvement over our baseline model in every category.
- ULMFiT didn't achieve the highest performance for any label.
- LSTM had the highest performance on hate speech.
- BERT had the highest performance on 6 of the 8 labels, and the highest average performance across labels.

BERT performs slightly lower than the LSTM on the hate speech and stereotypes labels, but it outperforms the LSTM on all other labels. The gains from transfer learning methods were somewhat modest. Twitter data differs greatly from the general domain datasets that ULMFiT and BERT were trained on.

Another consideration as it relates to the LSTM is that the f-score fluctuates greatly as it trains, and the f-scores reported for the LSTM are only the highest f-score observed during the training process. The performance of the LSTM is highly reliant on stopping training at the correct point. As currently implemented, early stopping prevents us from training on the validation set, and the dependence on accurate early stopping suggests that there may be a performance gap between the validation score and the test score.

In contrast, because the BERT model achieves strong performance with a fixed number of epochs, we expect to see less of a performance gap between the validation and test scores, and in fact we may see some improvement between the validation scores and the test scores from being able to also train on the validation data.

In summary, we chose BERT as our final model due to:

- Globally better performance than the SVM and ULMFiT on all labels.
- Better performance than the LSTM on almost all labels, including obscenity, name-calling, threat, and negative prejudice.
- Test scores from BERT may actually be higher than validation scores due to the opportunity to use more training data, without the need for a holdout set for early stopping.

4.6 Final Results

Our final test scores for the BERT model are below:

label	f-score
hatespeech	0.727
namecalling	0.583
negprejudice	0.530
noneng	0.932
obscenity	0.803
porn	0.762
stereotypes	0.365
threat	0.438

These metrics provide our best estimate of how the model will perform when applied to the 734k unlabeled tweets.

4.7 Model performance on Full Dataset

We also sought to identify a model that scales effectively when labeling the full dataset of 734k tweets. Predicting all 734k tweets based on the trained BERT model takes under 5 hours for all labels on a GPU.

In addition, the Social Justice Lab requested a metric such as a ranking or probability for each tweet, beyond the classification itself. The BERT model outputs raw scores after a linear mapping to the classes. A softmax of these scores produces the label probabilities, which we provided to further contextualize the classifications.

5 Conclusions And Future Work

Overall, the BERT model has shown the most promising results in identifying different forms of offensive speech. With more time and computational resources, it would be enlightening to examine how the performance would improve if the BERT model were trained from scratch on social media data. The current performance, however, is more than sufficient for the needs of the Social Justice Lab, particularly as it pertains to hate speech, obscenity, name-calling, and negative prejudice. Labels with lower performance may be excluded from the analysis if needed at the discretion of the Social Justice Lab, and the research will contribute to the growing understanding of how best to define and study those labels.

These results will be used in part of a larger study to understand the ideological differences in how conservatives and liberals express offensive speech. We're honored to contribute to such an important body of research.

6 Bibliography

References

[1] Mark J. Brandt, Christine Reyna, John R. Chambers, Jarret T. Crawford, and Geoffrey Wetherell. The ideological-conflict hypothesis: Intolerance among both liberals and conservatives. *Current Directions in Psychological Science*, 23(1):27–34, 2014.

[2] Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. Automated hate speech detection and the problem of offensive language. In *Proceedings of the 11th International AAAI Conference on Web and Social Media*, ICWSM '17, pages 512–515, 2017.

[3] Mohammed Hasanuzzaman, Gaël Dias, and Andy Way. Demographic word embeddings for racism detection on twitter. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 926–936. Asian Federation of Natural Language Processing, 2017.

[4] Pinkesh Badjatiya, Shashank Gupta, Manish Gupta, and Vasudeva Varma. Deep learning for hate speech detection in tweets. In *Proceedings of the 26th International Conference on World Wide Web Companion*, WWW '17 Companion, pages 759–760, Republic and Canton of Geneva, Switzerland, 2017. International World Wide Web Conferences Steering Committee.

[5] Zeerak Waseem. Are you a racist or am i seeing things? annotator influence on hate speech detection on twitter. pages 138–142, 01 2016.

[6] Björn Gambäck and Utpal Kumar Sikdar. Using convolutional neural networks to classify hate-speech. 2017.

[7] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *EMNLP*, volume 14, pages 1532–1543, 2014.

[8] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Global vectors for word representation. `https://github.com/stanfordnlp/GloVe`.

[9] Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. In *ACL*. Association for Computational Linguistics, 2018.

[10] Stephen Merity, Nitish Shirish Keskar, and Richard Socher. Regularizing and optimizing LSTM language models. *CoRR*, abs/1708.02182, 2017.

[11] Kristian Rother and Achim Rettberg. Ulmfit at germeval-2018: A deep neural language model for the classification of hate speech in german tweets. 09 2018.

[12] Sebastian Ruder. Imdb scripts. `https://github.com/fastai/fastai/tree/master/courses/dl2/imdb_scripts`, 2018.

[13] 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.

[14] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017.

[15] Thom Wolf and Victor Sanh. Pytorch pretrained bert. `https://github.com/huggingface/pytorch-pretrained-BERT`, 2018.

7 Supplement

7.1 Keywords Used to Create Dataset of Tweets

Perceived as left-leaning	Not perceived as left or right	Perceived as right-leaning
Underprivileged	millennials	executive
non-believers	teenagers	navy
black people	latinas	fascists
queer	latinos	christians
african american	middle income	veterans
homosexuals	middle class	coast guard
impoverished	caucasian	marines
progressives	chicanas	anti-choice
immigrants	crackers	elderly
immigration	youth	spoiled
drug addicts	asians	businessmen
undocumented		trust fund babies
radicals		evangelicals
hippies		capitalist
refugees		pentecostal
transgender		fundamentalists
unions		presbyterian
trade unions		army
niggers		pro-life
refugee crisis		right-wing
Godless		baptists
HIV-positive		lutherans
LGBT/LGBTQ		soldiers
civil rights activists		rich
greens		armed forces
community organizers		senior citizens
student activists		methodists
tree-huggers		rightists
black history month		adventists
destitute		the 1%
SJW		AntiSJW

7.2 Full Label Definitions Provided to Annotators

- Threat; Intimidation; Violence: The World Health Organization (WHO) defines violence as " the intentional use of physical force or power, threatened or actual, against oneself, another person, or against a group or community, which either results in or has a high likelihood of resulting in injury, death, psychological harm, maldevelopment, or deprivation" . A tweet meets the criteria for this label if there is language threat containing a threat to use physical violence , or intimidation , which is defined legally as " to make fearful or to put into fear" (i.e. scare tactics). Such threats of the use of violence or intimidation should be directed at an individual or a group .

- Obscenity, Vulgarity: An offensive word or phrase that is considered inappropriate to use in professional settings. A good heuristic to use is that if the word would be censored out of the "censored" version of the song, it is considered obscene or vulgar. Anything that would be, in PG terms, covered up by symbols to censor the language falls under this category as well (e.g. f**k or b@%! or a**hole). Note that if the words are censored through symbols, they do NOT get a rating of 1 . Only spelled out obscene or vulgar terms should get a rating of 1 for this category.

- Hate Speech; Racial, Ethnic, and Religious Slurs: Hate speech is a communication that carries no meaning other than the expression of hatred for some group. Racial and ethnic slurs are labels used to describe a racial, ethnic, or religious group that carry an inherently hateful and derogatory meaning. Legally , hate speech is defined as: "a communication that carries no meaning other than the expression of hatred for some group, especially in circumstances in which the communication is likely to provoke violence. It is an incitement to hatred primarily against a group of persons defined in terms of race, ethnicity, national origin, gender, religion, sexual orientation, and the like. Hate speech can be any form of expression regarded as offensive to racial, ethnic and religious groups and other discrete minorities or to women."

- Name-calling; Humiliation: Language directed at an individual or group that is mocking, insulting, demeaning, or embarrassing. This includes dehumanizing language. Wikipedia 's entry on name-calling includes the following: "Name calling is abusive or insulting language referring to a person or group, a verbal abuse." Examples could range from sissy or fag (as used in a derogatory manner, potentially related to group membership) to pig or ape or cow (comparisons to animals meant in a derogatory way).

- ALL CAPS: If at least half of the tweet is in ALL CAPS, the tweet should be labeled as ALL CAPS. If less than half of the tweet (e.g., one word) is in ALL CAPS, the tweet should not be labeled as ALL CAPS.

- Group-Based Stereotypes : According to Lawrence Blum (2004), " stereotypes are false or misleading generalizations about groups held in a manner that renders them largely, though not entirely, immune to counterevidence " (p. 251). Examples of salient [cultural] stereotypes include (Blum, 2004, p. 252):

  - Jews as greedy, wealthy, scholarly;
  - Blacks as violent, musical, lazy, athletic, unintelligent;
  - Women as emotional, nurturant, irrational;
  - Asian-Americans and Asians as good at math and science, hardworking, a 'model minority';
  - Irish as drinking too much;
  - English as snooty;
  - Poles as stupid; etc.

- Negative Prejudice : The essence of this work is to look at the manifestations of expressed prejudice in online speech, therefore we resort to the most classical, grounding definition of prejudice within social psychology. The first, and still most prominent definition of prejudice came from Gordon Allport, who defined it as " [a] feeling, favorable or unfavorable, toward a person or thing, prior to, or not based on, actual experience " (Allport, 1954/1990, p. 6); more specifically, he operationalizes (negative) prejudice as " an aversive or hostile attitude toward a person who belongs to a group, simply because he belongs to that group, and is therefore presumed to have the objectionable qualities ascribed to the group " (p. 7). Prejudice, thus, represents negative attitudes, or, in Allport's terms, an antipathy based on group-based generalizations and stereotypes (Allport, 1954/1990; Dovidio, Hewstone, Glick, Esses, 2010) [See above operational definition of stereotypes].

- Pornography: The purpose of this category is to label purely pornographic material and filter it out. Pornography is explicitly sexual material intended for sexual stimulation. If the tweet is pornographic, you should put a "1" for the pornography category and "0" for the other categories. This code should only be used if the material is clearly just advertisements for pornography. A tweet that contains sexual words, sexually objectifying language, or derogatory language with sexual overtones (e.g., using a word such as "cunt", "fuck", "rape", or "whore") is not necessarily pornographic, but may meet the criteria for other categories (e.g., obscenity, violence).

- Non-English Language Only: Any tweet that is entirely written in a non-English language. If the tweet is non-English only, you should put a "1" for the non-English only category and "0" for the other categories. This label

should NOT be used if a tweet contain a mix of English and non-English language.

References for Label Definitions

Allport, G. W. (1954/1990). The nature of prejudice. Reading, MA: Addison-Wesley.

Blum, L. (2004). Stereotypes and stereotyping: A moral analysis. Philosophical Papers, 33 (3), 251-289.