Abusive Language Detection using Syntactic Dependency Graphs

Kanika Narang
Facebook AI
kanika13@fb.com

Chris Brew*
LivePerson
christopher.brew@gmail.com

Abstract

Automated detection of abusive language online has become imperative. Current sequential models (LSTM) do not work well for long and complex sentences while bi-transformer models (BERT) are not computationally efficient for the task. We show that classifiers based on syntactic structure of the text, dependency graphical convolutional networks (DepGCNs) can achieve state-of-the-art performance on abusive language datasets. The overall performance is at par with of strong baselines such as fine-tuned BERT. Further, our GCN-based approach is much more efficient than BERT at inference time making it suitable for real-time detection.

1 Introduction

Abusive language usage in online social media is a grave issue affecting the interactions of users online. In a study conducted by Pew Research Center[1], 60% of Internet users have personally experienced harassment online. Social media websites, like Twitter and Facebook, allow users to report harassing content. However, due to the sheer volume of data, timely human curation of all reported content is not possible. Besides, there is also a need to filter these abusive content proactively. Therefore, there is an increased interest in automated detection and moderation of abusive speech in text (Waseem and Hovy, 2016; Vidgen et al., 2019).

Abusive speech is defined as an *attack* targeted towards a particular individual or entity belonging to a *protected group* (protected group may include, but are not always limited to, religious, gender or racial minorities) (ElSherief et al., 2018). Thus, abusive speech identification can be cast as a *relation extraction* problem in which the goal is to detect a "hate" or "attack" relation that links the speaker to a protected group (the object of the attack).

Current state-of-the-art methods in abusive language detection use either n-gram features (Waseem and Hovy, 2016; Davidson et al., 2017) or employ sequential deep learning models like CNN or LSTM (Zhang et al., 2018b; Badjatiya et al., 2017). These methods do not work well to capture semantic word meanings or long-range attack in text (such as long clauses or complex scoping shown in online attacks). Large pre-trained language models like BERT (Devlin et al., 2019) achieve high accuracy after fine-tuning on supervised tasks. However, these methods are computationally expensive and are, thus, unfit to be used for real-time detection.

Clark et al. (2019) observed that some attention heads of the pre-trained BERT model are learning syntactic dependencies between words such as direct objects of verbs etc. It is similar to a *dependency parser* that represents the structure of syntactic dependence between words in the sentence. Recently, Burnap and Williams (2016) and Alorainy et al. (2019) also showed that including syntactic dependency as features improves classifier performance in abusive language detection tasks. However, adding features can only provide weak supervision compared to encoding these dependence explicitly in the model. On the other hand, Zhang et al. (2018a) leveraged the dependency path between the subject and object of the sentence to achieve state-of-the-art results on *relation extraction* task on the TACRED dataset. They encoded the dependency parse graph using efficient graph convolution operations (Kipf and Welling, 2017).

However, a direct usage of Zhang et al. (2018a)'s method is not straightforward for abusive language datasets due to the complexity of the possibilities for expressing an attack in text. An attack may be

*This work was done while the author was at Facebook.
[1] https://pewrsr.ch/2XzABRo

Proceedings of the Fourth Workshop on Online Abuse and Harms, pages 44–53
Online, November 20, 2020.
https://doi.org/10.18653/v1/P17

expressed directly by either using explicit *slurs* or *curses*, or attacking a protected group explicitly. In other instances, the attack can be *implicit* such as sexist posts denigrating genders to stereotypical roles. These implicit attacks need nuanced understanding of the semantic context. However, parse structures of the text can still be useful for capturing longer-range dependencies than sequential models. For instance, in Fig 1, the sequential distance between the attack *mentally ill* and the target *Protected Group* is five, while the distance in the parse tree is only two.

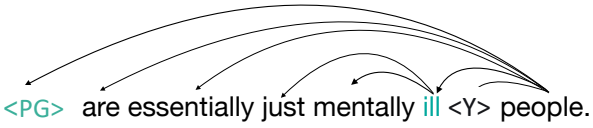


Figure 1: Dependency parse for a sample hate tweet. The target <*PG*> is closer to the attack word, *ill* in the parse tree. <PG> and <Y> replaces the targeted protected group and attack in the actual tweet.

An additional challenge in applying Zhang et al. (2018a)'s method to noisy social media text is that the text often does not have an explicit subject and object, in which case there is no dependency path between them. Or, especially in social media text, subject and object may be present but remain undetected by a parser trained on general purpose text.

Thus, we propose an adaption of the Zhang et al. (2018a) model that learns dependency-aware text representations useful for abusive language detection task. Our contributions are as follows:

- In this work, we leverage the dependency parse of the sentence to induce a graph on the text. We then augment the text embeddings of the words with their syntactic neighbors using efficient graph convolution operations (Kipf and Welling, 2017). Further, we propose a classifier based on this dependency graphical convolutional network, DepGCN, to detect abusive language online.

- We evaluate our method on the benchmark Twitter hate speech datasets. Our model outperforms the current state-of-the-art (Waseem and Hovy, 2016; Davidson et al., 2017) for abusive language detection and performs at par with strong baselines like fine-tuned BERT (Devlin et al., 2019).

- Further analysis shows that our DepGCN model is much more scalable than BERT and is complementary to the sequential models.

2 Methods

In this section, we first describe our approach to use syntactic dependencies to induce a graph on the text. We then convolve over this dependency graph using a graph convolutional framework (DepGCN) to compute a text embedding useful for abusive language detection task.

2.1 Graph representation of Text

We use the dependency parse tree to induce a graph on a sentence. Specifically, a graph $G =< V, E >$ is represented as a collection of vertices V and as a set of edges E between these vertices. To compute the graphical representation of the sentence, we treat each word as a vertex, with syntactic dependencies between words corresponding to an edge. Now, for this graph G, $\mathbf{A}$ represents the Adjacency matrix where $A_{ij} = 1$ if there is a dependency relation between word i and j and 0 otherwise. We also connect each word to itself such that $A_{i,i} = 1; \forall i \in V$. Although syntactic dependencies are directed, we treat these dependency edges as undirected, resulting in a symmetric matrix[2].

Graph Convolution Networks (GCN) are recently proposed to compute vertex embeddings in a graph by convolving over each vertex's local neighborhood (Kipf and Welling, 2017). The convolution operation for vertex i in layer k in GCN is defined as follows,

$$h_i^{k+1} = \sigma\left(\sum_{j=1}^{|V|} \tilde{A}_{ij}\mathbf{W}^k h_j^k + b^k\right) \quad (1)$$

$$= \sigma\left(\frac{\sum_{j\in\mathcal{N}(i)\cup i} \mathbf{W}^k h_j^k}{d_i} + b^k\right) \quad (2)$$

where $\tilde{\mathbf{A}} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{1/2}$ is the normalized Adjacency matrix with $\mathbf{D}$ being the degree matrix. h_i^{k+1} represents the vertex embeddings at layer $k+1$, with h_i^0 being initialized with the vertex features. In our case, we use pretrained word embeddings as the initial features. $\mathbf{W}^k, b^k$ are learnable weight and bias parameters of layer k and σ represents the ReLU function. $\mathcal{N}(i)$ represents the vertex

[2]Similar to (Zhang et al., 2018a), we observed a performance dip when using each edge direction as a separate graph.

i's neighborhood while $d_i = \sum D_i$ represents the vertex degree.

Now, assume that $\mathbf{W}^k = \mathbf{I}$ with $b^k = 0$ and $\sigma(.)$ as an identity function. The updated vertex embedding at layer $k+1$ will be,

$$h_i^{k+1} = \frac{\sum_{j \in \mathcal{N}(i) \cup i} h_j^k}{d_i} \quad (3)$$

Thus, it is easy to verify that the convolution operation updates the vertex embeddings at layer $k+1$ to be the average embeddings of the vertex's neighborhood and the vertex itself from the previous layer, k. In our dependency graph, applying graph convolution operation will augment each word's embedding with its syntactic neighbors. Thus, convolution helps to contextualize the word embeddings, where the word's syntactic relationships define the context. Notice that it is different from the sequential models (like LSTM or CNN), where the adjacent words in the sentence define the context.

Consider the sample tweet in Figure 1, in the first DepGCN layer, the attack *ill* will be augmented with its surrounding adverbs *mentally* and *just* (eq. (4)). In turn, these updated embeddings of *ill* will be propagated when computing embeddings of the noun *people* in addition to the subject <*PG*> in the next layer.

$$h_{ill} = f_{gcn}(h_{mentally}, h_{just}, h_{ill}) \quad (4)$$

$$h_{people} = f_{gcn}(h_{ill}, h_{<PG>}, h_{people}) \quad (5)$$

However, in sequential models with a fixed window, the attack *ill* will be too far from the subject <*PG*>.

Further, by stacking such k convolution layers, we can propagate the vertex embeddings to its k-hop neighborhood (Kipf and Welling, 2017). For our experiments, we did not see any further improvements after two layers. This could be because, as we are dealing with a short text, the resulting parse tree is shallow [3].

2.2 Sentence representation

In the previous section, we computed contextualized word embeddings using syntactic relationships. However, we still need to aggregate these vertex embeddings to compute a graph-level embedding (sentence in our case). In particular, we perform masked pooling over the learned word embeddings from the last layer (K) to compute a sentence embedding. We only pool over non-terminal words or intermediary nodes in the dependency parse tree (i.e. $|A_i| > 2$). We ignore the leaf words or words linked to only one other word as their word embeddings are relatively unchanged (because of fewer neighbors) after the convolution compared to other intermediary nodes with more neighbors. Thus, when we perform pooling over all the words, leaf words will skew the final result even though they are not always important.

We tried different variants of pooling (average and min), but max-pooling performed the best (eq. (6)) for our case.

$$h_G = \max_{i \in V'}(h_i^K) \; s.t. |A_i| = 2 \quad (6)$$

Further, we feed these sentence embeddings through fully connected layers followed by a sigmoid (σ) to compute the final class score (eq. (7)) for the sentence.

$$c_G = \sigma(f_{MLP}(h_G)); c_G \in \mathbf{R}^C \quad (7)$$

Here, C represents the total number of classes. The final architecture of our dependency graph convolution network (DepGCN) is depicted in Figure 2.

2.3 Embedding variants

As we deal with noisy text, there can be ill-formed words and grammatically incorrect sentences, potentially leading to incorrect parse trees. To overcome these potential errors, we feed the initial word embeddings (h_i^0) to a BiLSTM module. The BiLSTM module helps to aid in word disambiguation by encoding adjacent words in the sentence.

3 Experiments

In this section, we first describe our experimental setup, followed by the results. We then present a detailed error analysis of dependency-based model vs. a widely-used sequential model for sentence classification.

3.1 Experimental Setup

We first describe our datasets, followed by comparative baselines.

Datasets: Wiegand et al. (2019) emphasizes the difficulty of selecting representative datasets for studying abusive language. At the heart of this difficulty is the relative rarity of abusive language in large-scale user-generated text. It is not unusual for

[3] Our experiments did not show performance gain when using recent variants of GCN that uses attention (Veličković et al., 2018) or different aggregators (Xu et al., 2018).

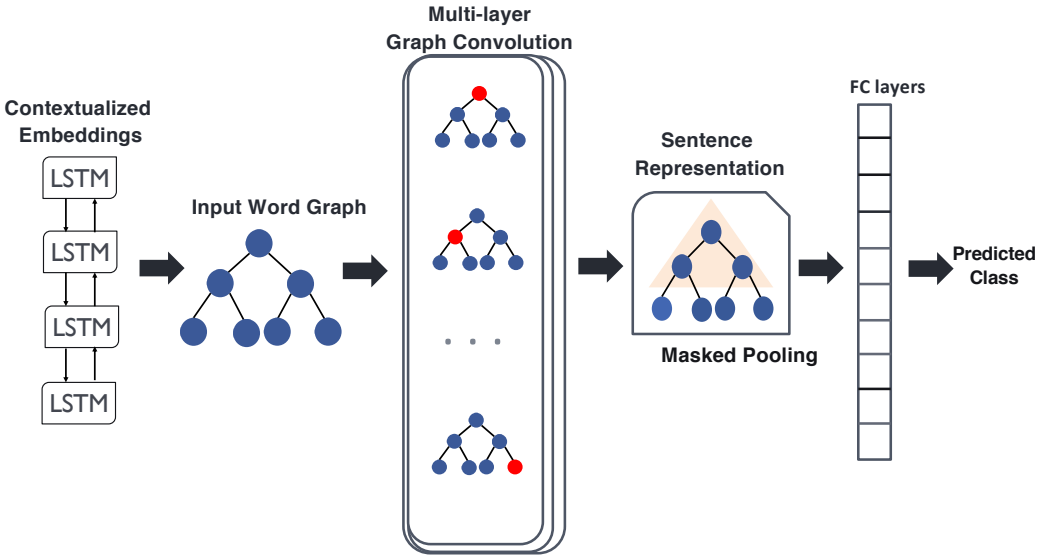


Figure 2: Overview of our proposed DepGCN model

$> 99\%$ of text to be benign. To make experiments manageable, Waseem and Hovy (2016) bootstrap data collection with queries that are indicators of possible hate speech[4]. As a result of this bootstrapping, the data collected is *not* representative of the underlying text distribution. This *data bias* applies to both the benign and the hateful categories. Davidson et al. (2017) provides an alternative dataset[5], but this too is affected by pre-filtering, with the result that the dataset also fails to represent the underlying text distribution. We also discuss the potential annotator bias (Sap et al., 2019) in these datasets, later in the Discussion section. Thus, caution is advised when interpreting the results of studies on these datasets.

Dataset	Categories		
Davidson et al. (2017)	Hate	Offensive	Benign
	1,430	19,190	4,163
Waseem and Hovy (2016)	Racism	Sexism	Benign
	1,939	3,148	11,115
Davidson extended	Hate	Offensive	Benign
	1,430	19,190	15,278

Table 1: Dataset Statistics.

For comparability, we experiment with both the datasets from Davidson et al. (2017) and Waseem and Hovy (2016). We do not claim that our results will necessarily transfer to more naturalistic settings. Table 1 lists the per class distribution in the collected dataset. Note that the Davidson et al. (2017) is highly skewed with majority of tweets being offensive. However, it is the opposite in real world settings. We thus also create a custom dataset (Davidson ext.) to mimic the real world settings, by adding benign tweets from Waseem and Hovy (2016) to Davidson et al. (2017)'s benign tweets.

Baselines: We compare against a variety of state-of-the-art approaches proposed for computing sentence embeddings. We use these embeddings to classify the tweets into abusive or not.

N-grams based approaches have shown to achieve competitive results for abusive language detection (Waseem and Hovy, 2016). In particular, the model extracts frequent N-grams from the tweets and feeds them into a logistic regression along with additional Twitter-specific features per tweet. We use tf-idf scores as features for unigram and bigram words from each tweet.

BERT models have achieved new state-of-the-art results on multiple natural language tasks such as question answering and language inference (Devlin et al., 2019). We use pre-trained $BERT_{base}$ model from the *transformers* [6] library for our experiments. Preliminary experiments showed that average pooling from the last four layers of BERT as the aggregate sentence representation performed better than the [CLS] token representation from the final layer. We append a linear layer on top of the model to compute class-wise scores. Further, we fine-tune this classifier on our training dataset.

BiLSTM is a sequential model to compute sentence embeddings useful for many downstream classification tasks (Zhou et al., 2016). We use

[4] https://bit.ly/3gN2RsD
[5] https://bit.ly/3dpoTzJ
[6] github.com/huggingface/transformers

the output of the final hidden layer in the BiLSTM as the sentence embeddings. We follow BERT in feeding the sentence embeddings to a linear layer to compute the final class-wise score.

Implementation Details: We initialized h_i^0 of each word with its Glove embeddings (Pennington et al., 2014) combined with its POS tag and NER tag given by the Stanford NLP API. We used Stanford parser[7] for extracting the dependency parse relationship between the words in a tweet. In the DepGCN model, we used a hidden dimension of 200 in both layers with ReLU non linearity. We trained with a batch size of 50 and used stochastic gradient optimizer with a learning rate of 0.3. We also used a dropout of 0.5 for both GCN layers and BiLSTM.

For the BERT classifier, we fine-tuned the model with a batch size of 16 for 6 epochs. We used a dropout of 0.1 in all layers and used Adam optimizer with a learning rate of 4e-6. For the N-gram approach, we ignored words which occur in less than 5 tweets or occur in more than 75% of the tweets to filter out uninformative words. We used $l2$ regularization in the logistic regression model and performed a grid search over different C values where C denotes regularization strength.

We implemented our model and the baselines(except N-gram) in PyTorch and run the experiments on an Nvidia Tesla V100 GPU. We used sklearn library for the N-gram baseline. We performed stratified sampling on the dataset to create an 80-10-10 split between training, dev, and test sets. The dev set is used for hyperparameter tuning while the results are reported on the test set. We used a *weighted* cross-entropy loss to counter the effect of class imbalance for all the baselines and our proposed approach. We report the class-wise F1 score with ROCAUC scores for each dataset.

3.2 Performance analysis

Table 2 reports class-wise F1 score with AUC scores for the Davidson et al. (2017) extended dataset. As expected, the bag-of-words based N-gram approach is not competitive with the best approaches. This result is reasonable, since N-grams does not take any advantage of semantic similarities between different words. More surprisingly, the powerful BERT model, even after fine-tuning, still performs slightly worse than our DepGCN model for the hate class.

[7] https://stanford.io/2zALCdz

Approach	Hate	Offensive	Benign	AUC
N-grams	0.35	0.88	0.88	0.899
BERT	0.45	0.94	0.96	**0.953**
BiLSTM	0.31	0.93	0.94	0.895
DepGCN	0.47	0.94	0.96	0.918
BiLSTM + DepGCN	**0.49**	**0.95**	**0.97**	0.945

Table 2: Class-wise F1 score and AUC of different approaches on the Davidson et al. (2017) extended dataset.

The sequential model, i.e., BiLSTM, also performs worse than our dependency-based model. As argued before, sequential models often struggle to capture long term dependencies between words while DepGCN alleviates this issue by encoding syntactic dependencies. Further, if we use BiLSTM to contextualize the embeddings before feeding it to our DepGCN model, the results are slightly improved. Note that even a slight improvement in the hate class is significant as the dataset contains limited training examples for this class (Table 1) as compared to the other classes.

Approach	Hate	Offensive	Benign	Overall	AUC
N-grams	0.46	0.94	0.84	0.89	0.899
BERT	0.42	**0.95**	**0.88**	**0.91**	**0.942**
BiLSTM	0.52	0.94	0.86	0.90	0.931
DepGCN	0.50	0.94	0.86	0.90	0.926
BiLSTM + DepGCN	**0.53**	0.94	0.87	**0.91**	0.937

Table 3: Class-wise F1 score and AUC for different approaches on the original Davidson et al. (2017) dataset.

We obtain a similar trend in the results when evaluating performance on the original Davidson et al. (2017) dataset (Table 3). The Benign class of the original dataset has systematically lower figures than the corresponding class in the extended dataset, presumably because the extended data set has a better representation of the space of possible benign examples. The Hate class is slightly easier to detect in the original dataset, even though it contains the same examples as the corresponding class in the extended dataset. This could be presumably because the classifiers expend more of their modeling capacity on the benign set. The same pattern is present to a lesser degree for the Offensive class.

BERT becomes more competitive with BiLSTM on the original dataset. BiLSTM retains a substantial ($0.52 > 0.42$) advantage over BERT on the Hate class and is close on Offensive and Benign. The sequential model, BiLSTM, also performs slightly better ($0.52 > 0.50$) than our DepGCN model for the hate class. One possible explanation

can be that the Davidson et al. (2017) dataset is full of *slurs* and *direct* hate attacks on protected groups. These *direct* attacks do not exhibit complex scoping or long-range dependencies and, thus, are well captured by the sequential models. Also, due to the heavy usage of *slurs* and noisy text, BERT performs worse as there might be many OOV tokens in the dataset.

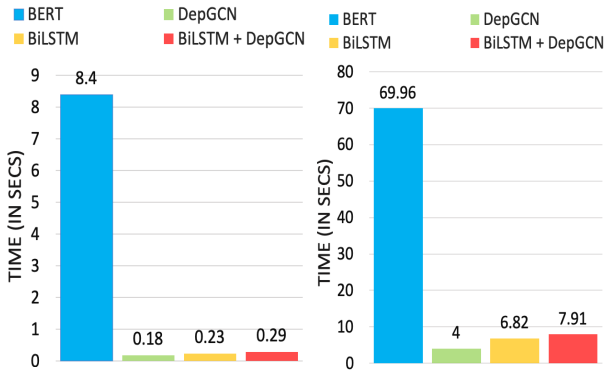
Approach	Racist	Sexist	Benign	Overall	AUC
N-grams	0.75	0.71	0.88	0.83	0.881
BERT	0.78	**0.81**	**0.91**	**0.88**	**0.945**
BiLSTM	0.72	0.71	0.89	0.84	0.917
DepGCN	0.76	0.72	0.88	0.83	0.926
BiLSTM + DepGCN	**0.78**	0.74	0.90	0.85	0.938

Table 4: Class-wise weighted F1 score and AUC for different approaches on the Waseem and Hovy (2016) dataset.

On a more nuanced dataset collected by Waseem and Hovy (2016), BERT performs the best out of the competing methods, as shown in Table 4. Our model performs competitively for racist and benign tweets while it performs worse for sexist tweets. This dataset is more nuanced as it contains more indirect or implied hate attacks (discussed in section 3.4) with the usage of fewer slurs. It seems that BERT is doing a better job of capturing the semantic meanings of these tweets.

3.3 Time Analysis

We further compare the running time of all the baseline approaches. Figure 3 shows the comparison at both training and inference time.



(a) Inference time per 1K tweets (b) Training time/Epoch

Figure 3: Time analysis of variants of our model with respect to the BERT (Devlin et al., 2019) model.

First, in Figure 3a, we plot the inference time (in secs) required by each approach per 1000 tweets. Our proposed DepGCN is the most efficient approach at inference time closely followed by BiLSTM. Adding the BiLSTM module before the DepGCN only increases the inference time slightly. On the other hand, BERT takes an order of magnitude longer than any of these approaches.

Note that, for operational reasons, the inference time does not take into account the time taken to extract the tweets' parse tree. We did the parsing step ahead of time, once, and reused the results for each experiment. A real production system would do this at inference time, adding a small time cost for each new tweet. State-of-the-art dependency parsers can currently achieve around 1000 sentences per second per CPU core (Chen and Manning, 2014; Kong and Smith, 2014). We estimate that on modern multi-CPU machines we can keep the parse cost under 0.05 seconds per 1000 tweets. This still keeps GCN methods more than competitive with BERT at inference time.

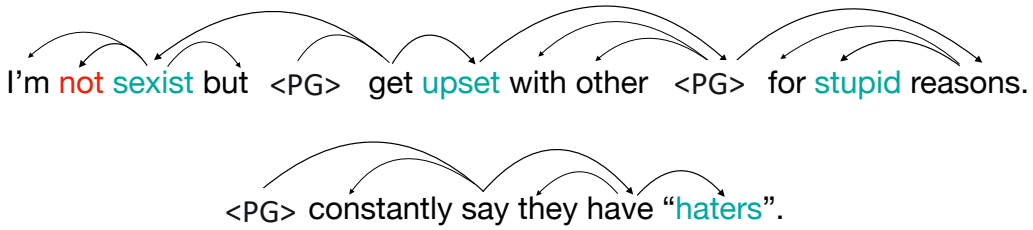
The same trend can be observed at training time too in Figure 3b. However, the jump from DepGCN to BiLSTM training time is a little higher than during inference.

In summary, our parser-based DepGCN approach is much more efficient than the BERT model. Also, including BiLSTM module to the DepGCN model leads to only a slight drop in efficiency.

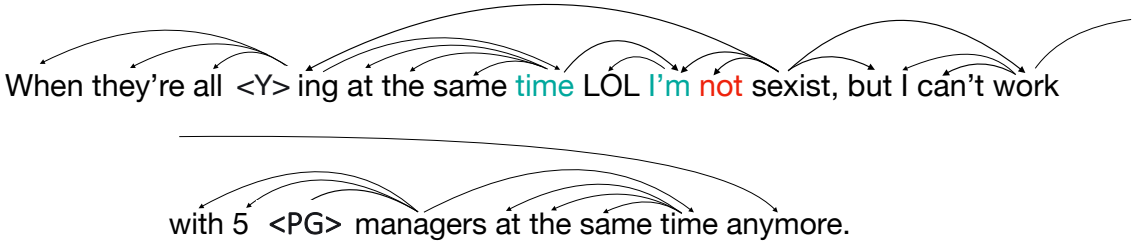
3.4 Error Analysis of Sequential vs. Dependency model

In this section, we present a detailed error analysis of the Sequential (BiLSTM) vs. Dependency (DepGCN) model. Table 5 shows the confusion matrix of BiLSTM vs DepGCN model on the Waseem and Hovy (2016) dataset. The parser-based approach is more conservative in labeling tweets as benign than the sequential approach. Specifically, sexist tweets are more probable to be misclassified as racist (7 for Dep vs. 1 for Seq) and vice versa. Alternatively, DepGCN tags much more benign tweets as abusive (Sexist/Racist) (105 for Dep vs. 36 for Seq), thus creating more false positives. However, as there is a higher cost involved in missing an abusive tweet, DepGCNs will be more effective in real-world scenarios.

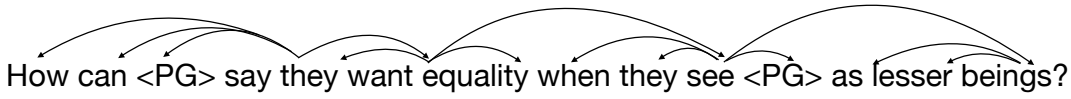
We also examined some sample tweets from the Waseem and Hovy (2016) dataset, which were erroneously classified as benign by BiLSTM but not by DepGCN and vice versa to understand the difference between these two approaches in depth.



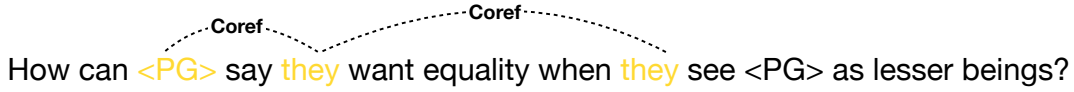
(a) Parse tree of the sexist tweet missed by LSTM.



(b) Parse tree of the sexist tweet missed by DepGCN.



(c) Parse tree of the racist tweet missed by DepGCN.



(d) Coreference resolution of the tweet.

Figure 4: Parse Tree of the sample tweets from Waseem dataset. <PG> replaces the Protected Group mentioned in the actual tweet.

	Racism		Sexism		Benign	
	Dep	Seq	Dep	Seq	Dep	Seq
Racism	7	4	11	3	7	18
Sexism	7	1	11	7	8	18
Benign	35	9	70	25	33	104

Table 5: Confusion matrix for Sequential (BiLSTM only) vs Dependency Parser (DepGCN) approach for Waseem and Hovy (2016) dataset.

Sexist tweet missed by LSTM: Following is a sample sexist tweet that is correctly classified by the DepGCN approach but missed by the BiLSTM.*"I'm not sexist but <PG> get upset with other <PG> for stupid reasons. <PG> constantly say they have haters."* Figure 4a shows the parse tree of the tweet by the Stanford parser. It is a difficult sample to classify as the author of the tweet says that he is *not* sexist but is writing offensive remarks against <PG>. The dependency tree can capture this long-range dependency and establish negative relation of "upset," "stupid," and "haters" with the "<PG>" subject.

Sexist tweet missed by DepGCN: However, DepGCN fails to capture similar nuanced sexism in another sample tweet, *"And when they're all <Y>ing at the same time LOL I'm not sexist, but I can't work with 5 <PG> managers at the same time anymore."*. Note that the sentence contains punctuation error as it is missing punctuation between the two sentences in the tweet (after *time* and before *I'm not*). This error leads to a wrong parse tree, as shown in Figure 4b. Thus, our parser-based model is sensitive to these parsing errors.

Sexist tweet missed by DepGCN: However, even if the parse tree is correct, establishing dependency relationships may not be sufficient to capture nuanced relationships in the text in some cases. For instance, the parse tree of the racist tweet, *"How can <PG> say they want equality when they see <PG> as lesser beings? "* shown in Section 3.4 is correct. However, the parse tree misses the coreference of pronoun *they* to belong to *<PG>*. In these cases, powerful language models like BERT will be able to extract these relationships.

These analyses show that both approaches have their own merits and often perform well for complementary attack types.

4 Related Work

Most of the prior work for detecting abusive speech on Twitter primarily relies on using statistical features like bag-of-words (character or word n-grams) or tf-idf features for automated detection (Waseem and Hovy, 2016; Davidson et al., 2017; Nobata et al., 2016). Bag-of-words approaches are unable to capture nuanced abusive speech as they fail to contextualize the word meanings. For instance, depending on the context, the word *gay* can be used to denote either ebullience or sexual preference. Only the latter is a candidate attack.

Recently, deep learning models are also proposed that leverage pre-trained word embeddings (Mikolov et al., 2013; Pennington et al., 2014) to capture the semantics of the tweets. These models aggregate individual word embeddings in a context-aware manner to compute tweet embeddings and later use them for classification. Earlier studies have either used the CNN (Gambäck and Sikdar, 2017; Park and Fung, 2017) or RNN (Badjatiya et al., 2017; Agrawal and Awekar, 2018) to compute these embeddings.

The syntactic structure of the text was also used to help identify the target group and the intensity of hate speech (Warner and Hirschberg, 2012; Silva et al., 2016). The primary difficulty of these works is that the space of possibly relevant rules is too large to be comprehensive. Besides, it verges on the impossible to specify a set of rules that will cover possible implicit attacks. On the other hand, Burnap and Williams (2016); Alorainy et al. (2019) proposed computational models using the dependency labels as features and reported significant gains over the bag-of-words features. Our model builds on this work and explicitly models the dependency between words using graph convolution operations.

5 Discussion

Worse performance of BERT: Our experiments did not show the superior performance of the BERT model on the abusive language datasets. We also noticed that prior literature on comparable tasks is variable, with some successes for BERT-like models but few robust trends. There are numerous reports of difficulties in training these large neural networks on the small, imbalanced annotated datasets typical of such tasks (Zampieri et al., 2019; Liu et al., 2019). The challenges are likely an effect of over-fitting and lead to inconsistent results. Remedies include careful hyperparameter tuning, early stopping, and the use of ensembles. Risch and Krestel (2020) proposed to use an ensemble of BERT models to control the variance of these large models over small datasets. Ensembling is expensive, so there remains a need for computationally efficient methods that approach the same performance. Because GCN has far fewer parameters, it is less likely to need these countermeasures against overfitting.

Experiments on the relatively larger abusive language dataset of 100K tweets (Founta et al., 2018) by Lee et al. (2018) highlighted that sequential models such as RNN perform well but are still hard to train on this dataset size. Further, Kumar et al. (2018) concluded that with optimal feature selection, classifiers like SVM and Random forest performs at par with neural networks.

Dataset quality and annotator bias: Recent works have highlighted that majority of the abusive language datasets suffer from poor quality (Wiegand et al., 2019; Vidgen and Derczynski, 2020) or show evidence that annotator decisions were inappropriately affected by surface markers of speaker race (Sap et al., 2019). We believe that corrections of these deficiencies and biases in annotation are essential for research progress in the field. Better dataset collection, however, is complementary to our computational approach. We believe our dependency parser-based approach should be able to perform competitively on future datasets. This is because the model is designed to be insensitive to the cues from unrelated single words.

Social media-specific tools: Social media text tends to be very noisy and thus, NLP tools trained on general corpus do not perform well on these datasets. However, our preliminary experiments with using pre-trained Glove embeddings [8] trained on the Twitter dataset showed a significant drop in performance. The performance drop could be because of the relatively smaller training data size of social media text used for training such models that may lead to overfitting. A possible counter approach can include pretraining these embeddings on a mix of general news corpus data along with social media text.

Similarly, our experiments with the training of Spacy parser[9] on our training data showed a drop in performance. We did not experiment with parsers

[8] nlp.stanford.edu/projects/glove/
[9] https://spacy.io/

specially built for Twitter (Kong et al., 2014) but believe that using Twitter-specific parsers might improve our results further.

6 Conclusion

In this work, we propose a novel sentence encoder that extends the graph convolutional network (GCN) to an induced graph built from syntactic dependencies in the text for abusive language detection. Our model achieved state-of-the-art performance on public hate speech twitter datasets, performing at par with strong baselines such as fine-tuned BERT.

Our DepGCN model is much more scalable than BERT and thus, can be efficiently used for real-time detection. Error analysis reveals that our model is complementary to the sequential baselines. Future work includes using an ensemble of sequential model with our dependency parser-based model. We will also extend our approach to longer text spanning multiple sentences such as posts/comments on online platforms.

References

Sweta Agrawal and Amit Awekar. 2018. Deep learning for detecting cyberbullying across multiple social media platforms. In *European Conference on Information Retrieval*, pages 141–153. Springer.

Wafa Alorainy, Pete Burnap, Han Liu, and Matthew L Williams. 2019. "the enemy among us" detecting cyber hate speech with threats-based othering language embeddings. *ACM Transactions on the Web (TWEB)*, 13(3):1–26.

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

Pete Burnap and Matthew L Williams. 2016. Us and them: identifying cyber hate on twitter across multiple protected characteristics. *EPJ Data science*, 5(1):11.

Danqi Chen and Christopher Manning. 2014. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 740–750, Doha, Qatar. Association for Computational Linguistics.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D Manning. 2019. What does BERT look at? an analysis of BERT's attention. *arXiv preprint arXiv:1906.04341*.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. In *Eleventh international aaai conference on web and social media*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Mai ElSherief, Vivek Kulkarni, Dana Nguyen, William Yang Wang, and Elizabeth Belding. 2018. Hate lingo: A target-based linguistic analysis of hate speech in social media. In *Twelfth International AAAI Conference on Web and Social Media*.

Antigoni-Maria Founta, Constantinos Djouvas, Despoina Chatzakou, Ilias Leontiadis, Jeremy Blackburn, Gianluca Stringhini, Athena Vakali, Michael Sirivianos, and Nicolas Kourtellis. 2018. Large scale crowdsourcing and characterization of twitter abusive behavior. *ICWSM*.

Björn Gambäck and Utpal Kumar Sikdar. 2017. Using convolutional neural networks to classify hate-speech. In *Proceedings of the First Workshop on Abusive Language Online*, pages 85–90, Vancouver, BC, Canada. Association for Computational Linguistics.

Thomas N Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. *International Conference on Learning Representations*.

Lingpeng Kong, Nathan Schneider, Swabha Swayamdipta, Archna Bhatia, Chris Dyer, and Noah A Smith. 2014. A dependency parser for tweets. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1001–1012.

Lingpeng Kong and Noah A Smith. 2014. An empirical comparison of parsing methods for stanford dependencies. *arXiv preprint arXiv:1404.4314*.

Ritesh Kumar, Atul Kr Ojha, Shervin Malmasi, and Marcos Zampieri. 2018. Benchmarking aggression identification in social media. In *Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying (TRAC-2018)*, pages 1–11.

Younghun Lee, Seunghyun Yoon, and Kyomin Jung. 2018. Comparative studies of detecting abusive language on twitter. In *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, pages 101–106.

Ping Liu, Wen Li, and Liang Zou. 2019. Nuli at semeval-2019 task 6: Transfer learning for offensive language detection using bidirectional transformers.

In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 87–91.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Chikashi Nobata, Joel Tetreault, Achint Thomas, Yashar Mehdad, and Yi Chang. 2016. Abusive language detection in online user content. In *Proceedings of the 25th International Conference on World Wide Web*, WWW '16, page 145–153, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.

Ji Ho Park and Pascale Fung. 2017. One-step and two-step classification for abusive language detection on twitter. In *Proceedings of the First Workshop on Abusive Language Online*, pages 41–45.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Julian Risch and Ralf Krestel. 2020. Bagging BERT models for robust aggression identification. In *Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying*, pages 55–61, Marseille, France. European Language Resources Association (ELRA).

Maarten Sap, Dallas Card, Saadia Gabriel, Yejin Choi, and Noah A. Smith. 2019. The risk of racial bias in hate speech detection. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1668–1678, Florence, Italy. Association for Computational Linguistics.

Leandro Silva, Mainack Mondal, Denzil Correa, Fabrício Benevenuto, and Ingmar Weber. 2016. Analyzing the targets of hate in online social media. In *Tenth International AAAI Conference on Web and Social Media*.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph attention networks. In *International Conference on Learning Representations*.

Bertie Vidgen and Leon Derczynski. 2020. Directions in abusive language training data: Garbage in, garbage out. *arXiv preprint arXiv:2004.01670*.

Bertie Vidgen, Alex Harris, Dong Nguyen, Rebekah Tromble, Scott Hale, and Helen Margetts. 2019. Challenges and frontiers in abusive content detection. In *Association for Computational Linguistics*.

William Warner and Julia Hirschberg. 2012. Detecting hate speech on the world wide web. In *Proceedings of the second workshop on language in social media*, pages 19–26. Association for Computational Linguistics.

Zeerak Waseem and Dirk Hovy. 2016. Hateful symbols or hateful people? predictive features for hate speech detection on Twitter. In *Proceedings of the NAACL Student Research Workshop*, pages 88–93, San Diego, California. Association for Computational Linguistics.

Michael Wiegand, Josef Ruppenhofer, and Thomas Kleinbauer. 2019. Detection of Abusive Language: the Problem of Biased Datasets. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 602–608, Minneapolis, Minnesota. Association for Computational Linguistics.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2018. How powerful are graph neural networks? In *International Conference on Learning Representations*.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019. Semeval-2019 task 6: Identifying and categorizing offensive language in social media (offenseval). In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 75–86.

Yuhao Zhang, Peng Qi, and Christopher D. Manning. 2018a. Graph convolution over pruned dependency trees improves relation extraction. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*.

Ziqi Zhang, David Robinson, and Jonathan Tepper. 2018b. Detecting hate speech on twitter using a convolution-gru based deep neural network. In *The Semantic Web*, pages 745–760, Cham. Springer International Publishing.

Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. 2016. Attention-based bidirectional long short-term memory networks for relation classification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 207–212, Berlin, Germany. Association for Computational Linguistics.