Augment to Prevent: Short-Text Data Augmentation in Deep Learning for Hate-Speech Classification

Georgios Rizos*
Konstantin Hemker*
georgios.rizos12@imperial.ac.uk
konstantin.hemker@googlemail.com
[1]GLAM - Group on Language, Audio and Music, Imperial College London

Björn Schuller
bjoern.schuller@imperial.ac.uk
[1]GLAM - Group on Language, Audio and Music, Imperial College London
[2]ZD.B Chair of Embedded Intelligence for Health Care and Wellbeing, University of Augsburg

ABSTRACT

In this paper, we address the issue of augmenting text data in supervised Natural Language Processing problems, exemplified by deep online hate speech classification. A great challenge in this domain is that although the presence of hate speech can be deleterious to the quality of service provided by social platforms, it still comprises only a tiny fraction of the content that can be found online, which can lead to performance deterioration due to majority class overfitting. To this end, we perform a thorough study on the application of deep learning to the hate speech detection problem: a) we propose *three text-based data augmentation techniques* aimed at reducing the degree of class imbalance and to maximise the amount of information we can extract from our limited resources and b) we apply them on a selection of top-performing deep architectures and hate speech databases in order to showcase their generalisation properties. The data augmentation techniques are based on a) synonym replacement based on word embedding vector closeness, b) warping of the word tokens along the padded sequence or c) class-conditional, recurrent neural language generation. Our proposed framework yields a significant increase in multi-class hate speech detection, outperforming the baseline in the largest online hate speech database by an absolute *5.7 %* increase in Macro-F1 score and *30 %* in hate speech class recall.

KEYWORDS

online hate speech detection, short text data augmentation, class imbalance

ACM Reference Format:
Georgios Rizos, Konstantin Hemker, and Björn Schuller. 2019. Augment to Prevent: Short-Text Data Augmentation in Deep Learning for Hate-Speech Classification. In *The 28th ACM International Conference on Information and KnowledgeManagement (CIKM '19), November 3–7, 2019, Beijing, China.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3357384.3358040

*Both authors contributed equally to this research.

ACM ISBN 978-1-4503-6976-3/19/11…$15.00
https://doi.org/10.1145/3357384.3358040

1 INTRODUCTION

Due to the stark increase of hateful content on the internet [3, 24, 34], hate speech classification has become a subject of growing interest for industry and academia [6, 42, 44]. Since abusive textual content is only a small fraction of the total online user generated content, any collected representative sample is expected to exhibit high class imbalance (especially if we want to model multiple types of hate speech), which is a well-known factor for the underperformance of machine learning methods. Currently, the best performing methods in terms of accuracy for hate speech detection utilise neural word embeddings (e. g., the Word2Vec method [23]) and deep learning, first introduced to this domain in [2]. However, recent studies [6, 47] fail to achieve an equally impressive increase in the detection performance of the hate speech class, which is a serious limitation since this is the class of interest. *To this end, we must design a solution that takes the class imbalance into account, yet manages to reap all the benefits from using deep neural networks in what is in its core, a natural language processing (NLP) task.*

To this end, the idea of enriching a dataset with perturbed replicas of its samples has been very successful in other domains (e. g., standard practice in image classification [19, 38], audio-visual affect recognition [40], environmental sound classification [33], speaker language identification [18] and 3D pose estimation [29]), however such *data augmentation has been underexplored in NLP.* The reason for that is that whereas techniques such as flipping and rotation on images will result in new, valid images with similar semantic information, they are not transferrable to text, as they would break correctness of syntax and grammar and even alter the meaning of the original sentence. Similarly, whereas noise injection is popular in audio signal data augmentation [13, 17, 40], it is not directly applicable to text as word and character tokens are categorical.

In this paper, we show that if adapted to the domain of short-text hate speech detection, data augmentation can yield significant improvements as well. *We propose three data augmentation techniques, tailored specifically for text*: a) synonym replacement of any word in a sentence according to cosine similarity above a threshold between pre-trained word embedding vectors on a large corpus, b) warping the sentence along the padded sequence by repeating word tokens and c) recurrent neural language generation of new sentences. All these methods are class-conditional, meaning that we are certain of the label of the newly generated samples. Moreover, the two former methods can be viewed as generating perturbed versions of the original sentence that preserve the meaning thereof. This way data sparsity is reduced and the model is able to learn a more robust

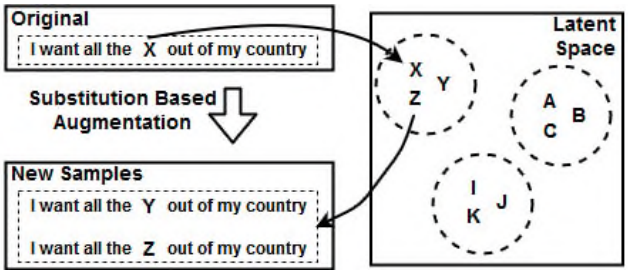

Figure 1: Motivation for leveraging neural word embeddings for generating new samples: the sentence is an example of hate speech, regardless of the victimised target group or the particular name or slur used for the denotation thereof.

association between the meaning of the original sentence and the label instead of trying to memorise specific keywords. This concept is illustrated in Figure 1, exemplified by hate speech detection.

The achievements of this paper are the following:

- We introduce three data augmentation techniques for text and *we demonstrate their capability for generalisation by applying them on three hate speech databases and four top performing deep models from an extensive comparative study using fully connected, convolutional and recurrent layers (henceforth FCNN, RNN and CNN).* Compared to the non augmented baselines, we observe an overall Macro-F1 improvement up to $\sim 5.7\,\%$ across all network topologies and recall improvement of the minority class of up to $\sim 30\,\%$.
- We achieve significant improvement over previous deep architectures applied on the task: We show that an optimised for the task deep model consisting of Global Vector (GloVe) embeddings followed by stacked CNN and gated recurrent unit (GRU) RNN layers outperforms other deep learning architectures [47] by and absolute of $0.5\,\%$ Macro-F1 score and $1.3\,\%$ hate recall.
- The recall measure of the minority class (hate speech) is improved by an absolute of $\sim 30\,\%$, compared with a previous pure deep learning baseline which is the largest contribution to the prediction of the class of interest.

Our experiments are performed on three online hate-speech databases (see Section 3) that we refer to as: a) **HON** [6] (hate speech - offensive language - neutral), b) **RSN-1** [44] (racism - sexism - neutral) and c) **RSN-2** [42] (classes as previous). We provide an implementation of the data augmentation methods we propose and all the methods included in the architecture comparison in the project's GitHub page[1].

2 RELATED WORK

Finding a common definition of hate speech is difficult due to its subjective nature and country-dependent legislative differences: while in the United States of America most types of hate speech are protected under free speech provisions, in most other Western countries hate speech is illegal and defined in legal terms as expressions that "target minority groups that could promote violence or social disorder" [6]. In many previous studies, hate speech detection has been formulated as a binary classification problem [2, 21, 41] which unfortunately disregards subtleties in the definition of hate speech, e. g., implicit versus explicit or directed versus generalised hate speech [43] or different types of hate speech (e. g., racism and sexism) [42, 44]. The study performed in [6] was the first one in which a distinction was made between hate speech and otherwise offensive language. An example of the latter could be a direct quotation of a another person's racist/misogynistic/homo-/trans-phobic comment followed by a condemnation of said comment. Another example would be chatter among friends of the same racial/social background including terms that coming from an outsider and used in a different context or with malicious intent would be inappropriate. The label structure can even be hierarchical, as in a study in which the authors have applied a Hierarchical Conditional Variational Autoencoder [28] on a database collected by pulling tweets from 40 hate groups belonging to 13 hate speech types. *The three databases we utilise in our experiments follow a multi-class paradigm.*

Using lexical resources such as WordNet[2], as well as extracting and enhancing surface features of comments is the most straightforward approach to hate speech detection [11, 35]. Such methods utilise a dictionary of loaded terms that are assigned an 'offensiveness' weight and used in order to infer the total score of the entire sentence. *Such a word-level approach, however, fails to take into account the context of a sentence into account. Utilising a fixed vocabulary also does not allow for considering common alternative spellings or misspellings of hate terms.* The authors of the study performed in [6] performed a comparison among a range of traditional machine learning techniques such as support vector machines, logistic regression, and Naïve Bayes classifiers on a three-class hate speech database. They manage to achieve a seemingly high overall accuracy of $90\,\%$, but *do not address the issue of class imbalance, which we propose is the reason for the comparatively lower recall of* $61\,\%$ *on the hate speech class.* Deep learning approaches have been found to be more effective both on binary classification [2, 8], as well as multi-class classification [10, 25, 47] in this context, possibly due to the inclusion of word semantic information by the utilisation of distributional word embedding techniques [4, 7, 23, 26, 27] pretrained on large text corpora. *However, there is no discussion of either the class imbalance problem in a limited database or the recall in the hate speech class. Here, we replicate the state-of-the-art deep method proposed in [47] and show that there is much room for improvement, both by means of tailored neural architectures and embeddings and by the utilisation of much needed data augmentation.*

Whereas in image and sound processing the flipping, cropping and noise injection data augmentation techniques are commonly applied [9, 18, 29, 33, 38, 40], such techniques are either not directly applicable to text due to the categorical nature of word and character tokens or because they change the meaning of the original sentence. There are certain languages with which this is not a significant problem and a variation of such techniques have been successful [30], but English (which is the language we focus on this study) is not one of them. In a study performed in [16], the authors performed word dropout, an idea that was shown in a later study [45] to be very incosistent with respect to the improvement

[1] https://github.com/glam-imperial/TextAug

[2] https://wordnet.princeton.edu/documentation

it yields. In [45], the same holds with many choices for additive noise (similar to dropout) on neural embedding vectors where, in fact, the performance occasionally deteriorates. Looking into data augmentation methods that generate new samples before inputting them into a learning model, the closest text-based data augmentation method to the ones we designed for this study is the one proposed in [46] in which existing words in a sentence are selected based on a geometric distribution and replaced with a suggested external thesaurus synonym. Although this technique is shown to be effective, there is a major problem in using it in a hate speech context. As lexical approaches are likely to miss common colloquial alternative spellings and misspellings and are expected to have many out-of-vocabulary words, this makes this method hard to replicate in a meaningful manner. Our proposed substitution method replaces words based on very specific criteria based on neural word embedding similarity (such as Word2Vec [23]) and are aware of POS-tags [5] in order to preserve, as much as possible, the semantics of the original sentence.

3 HATE SPEECH DATA

In our experiments we utilise three Twitter databases, all of which treat the problem of online hate-speech detection as being multi-class and furthermore exhibit high class imbalance. Throughout the paper we refer to these as: a) **HON** [6], b) **RSN-1** [44] and c) **RSN-2** [42]. The *HON* database makes the separation between hate speech, offensive language and a neutral class. The latter two (*RSN-1* and *RSN-2*) include a different axis of hate-speech differentiation by making a distinction between racist and sexist comments, and a neutral class. In the case of *RSN-2*, there were also comments that were annotated as both racist and sexist, but in this study, we simplify this multi-label problem into a multi-class problem in order to homogenise our experimental setup across databases and as such, work only with samples from the racism, sexism and neutral classes. A summary of the database sizes and class proportions is included in Table 1. In the cases of *RSN-1* and *RSN-2* the number of samples reported corresponds to the number of tweets that were still available at the time we attempted to fetch them using the Twitter API[3], as the authors of the papers in which they were introduced provide a list of tweet ids.

The *HON* database comprises around 25,000 labeled tweets, which were obtained using the hatebase[4] lexicon to filter tweets for common abusive terms, and then pulling all the tweets from all the users selected (approximately 85 million). The 25,000 tweets were a random sample taken from the overall pool of tweets and were labeled using CrowdFlower[5]. The database is available online[6].

In order to validate the generalisability of our framework, we opted to also perform experiments on the *RSN-1* and *RSN-2* databases as well. We used the Twitter API in order to fetch the tweets corresponding to the tweet ids provided by the authors of [42, 44]. The annotation was performed using CrowdFlower for the *RSN-1* and manually by a mix of professional and amateur annotarors for the *RSN-2* database. The tweet ids and corresponding labels of these databases are both available online[7].

Table 1: Overview of datasets. Class names are *H*ate, *O*ffensive and *N*either for the first database and *R*acism, *S*exism and *N*either for the other two.

Dataset	Class (in %)			No. Samples
	H	O	N	
HON [6]	5.7	77.4	16.7	24,783
	Class (in %)			**No. Samples**
	R	S	N	
RSN-1 [44]	0.3	26.1	73.4	11,299
RSN-2 [42]	1.3	11.8	86.3	6,310

4 SHORT TEXT DATA AUGMENTATION

Since we cannot use the data augmentation techniques that are used in non-NLP domains, we want to develop new ones that ideally satisfy three desiderata - they must:

- change the input to the neural network (new sample).
- be class-conditional, such that no manual labeling is required (same class).
- produce perturbed versions of the original sentence samples (same meaning - see Figure 1).

The methods we propose in Sub-Section 4.1 satisfy all three, whereas the method described in Sub-Section 4.3 only the first two.

4.1 Substitution Based Augmentation

In order to make relevant substitutions of words with synonyms without using tools external to neural networks (e. g., a thesaurus as in [46]) we make use of pre-trained neural word embeddings because they allow us to determine the relative similarity between each word in the vocabulary space of a text corpus. We assume we are given a training sequence of words $\{w_t\}$ with corresponding embedding vectors $\{\mathbf{v_t}\}, \forall t \in \{1:T\}$. For each center word w_t, we predict the surrounding context words w_o within a radius m. For example, for Word2Vec [23] we want to maximise the probability of any context word given the current center word $w_{\hat{o}}$, while minimising the probability of a random word from the vocabulary (i. e., negative sampling):

$$J_t(\theta) = \log \sigma(\mathbf{v_t}^\top \mathbf{v_o}) + \sum_{\hat{o} \sim P(w)} \log \sigma(-\mathbf{v_t}^\top \mathbf{v_{\hat{o}}}), \tag{1}$$

where $P(w)$ is a distribution that places higher sampling probabilities on less frequent words, θ is the entire array of embedding vectors and σ a nonlinear scoring function. This representation tends to become more precise the larger the training corpus, as different words would have been seen in a broader range of contexts. The utilisation of neural embeddings is not only useful in substitution based data augmentation, as it is a central part of the main deep prediction model.

[3] https://developer.twitter.com/en/docs.html
[4] https://www.hatebase.org/connect_api
[5] https://www.figure-eight.com
[6] https://github.com/t-davidson/hate-speech-and-offensive-language
[7] https://github.com/zeerakw/hatespeech

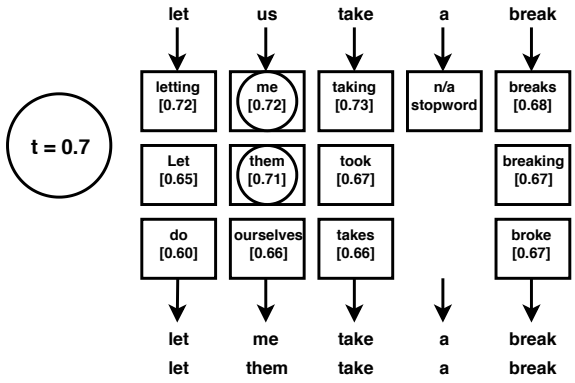

Figure 2: Perturbed copies of an original short text are generated based on high cosine similarity and POS-tag match. We can see that 'letting' exceeds the similarity threshold, but is a gerund, while 'let' is a regular verb.

The first data augmentation technique, which we refer to as **ThreshAug**, builds on the definition of a similarity measure between the equal-size vector representations denoted by v_i in Equation 1). We denote by $c \in [0, 1]$ the cosine similarity between two vectors $\mathbf{v_i}$ and $\mathbf{v_j}$:

$$c = \frac{\mathbf{v_i}^T \cdot \mathbf{v_j}^T}{\|\mathbf{v_i}\| \cdot \|\mathbf{v_j}\|}, \tag{2}$$

where $\|\mathbf{v}\|$ is the norm of the vector $\mathbf{v}$. For each word w_i of the input word sequence, this method describes a substitution that is determined by two factors. Firstly, any potential replacement word must exceed the cosine distance threshold t, where $t \in [0, 1]$ *and* it must match the POS-tag assigned to the word. The intuition behind the inclusion of both the above requirements is that two words must have been seen in sufficiently equal contexts such that one can be replaced with the other without changing the sentence semantics. A usage example of this method is illustrated in Figure 2.

The strength of the above method is that it encourages the downstream task to place lower emphasis on associating single words with a label and instead place higher emphasis on capturing similar sequential patterns, i. e., the context of hate speech (see Figure 1). Finally, we note that it makes sense to align the embedding model used for substitution based augmentation with the one used in the deep text classifier - we show proof of that in Sub-Section 5.5.

4.2 Word Position Augmentation

The second technique, which we refer to as **PosAug**, is based on the observation that sample sentence lengths vary significantly and as such, due to zero-based left padding, most elements in training batches consist of zeroes. This has an impact in the means by which neural networks perceive data and we hypothesise that by shifting and warping the sentence (i. e., the word tokens) within the confines of the padded sequence, we can receive meaningfully perturbed versions of the original samples with the same semantics and class.

4.3 Neural Generative Augmentation

After sequential machine learning models such as recurrent neural networks gained popularity within the NLP community, RNNs were also applied to natural language generation (NLG) [37]. The main idea behind using RNNs for NLG is to first train a text model on the class-conditional corpus subset that we want to augment, a method that we refer, henceforth, as **GenAug**. In order to generate, we prime our model with a random start word from the vocabulary and attempt to predict the next word in the sequence based on the previous sequence and the model weights. More formally, given a sequence of embedding vectors $\{\mathbf{v_t}\}$, the network uses the returned sequence of its output probability vectors $\{\hat{\mathbf{v}}_\mathbf{t}\}$ to obtain a sequence of predictive distributions [37]. In other words, the RNN predicts the next element in the sequence based on the *entire* previous sequence ($P(\mathbf{v_{t+1}}|\{\mathbf{v_t}\})$):

$$P(\mathbf{v_{t+1}}|\{\mathbf{v_t}\}) = \text{softmax}(\hat{\mathbf{v}}_{\mathbf{t+1}}), \tag{3}$$

$$\text{softmax}(\hat{\mathbf{v}}_{\mathbf{t+1}})^{(j)} = \frac{\exp(\hat{\mathbf{v}}_{\mathbf{t+1}}^{(j)})}{\sum_k \exp(\hat{\mathbf{v}}_{\mathbf{t+1}}^{(k)})}, \tag{4}$$

where the superscript (j) denotes the j-th element of the vector. For longer texts, it may make sense to restrict the length of the previous sequence considered for generation, however for short text sequences, such as tweets as in our study, we hypothesise that the entire sequence may be relevant.

We design a network which takes an input of N words and converts each into a 100-dimensional word embedding vector. The $N \times 100$ embedding matrix is used as an input for long short-term memory (LSTM) recurrent layers set in parallel, with 128 hidden units each. The separate outputs of each RNN layer are concatenated, outputting a $N \times 356$ feature matrix, which is fed into a final FCNN layer. The final output represents a probability distribution over each word in the vocabulary. Once this model is trained and the weights are stored, this technique is significantly faster than the ones based on substitution.

5 EXPERIMENTS

Across all databases that we use in our study, we consider a tweet to be a sample with a corresponding label and our goal is to predict one class label per tweet. All the reported results in the following Sub-Sections are averaged across 20 trials. We used a 50-25-25 train-valid-test split scheme everywhere (a 10 % validation split was used in [6]). We opt to report Macro-F1, which places equal emphasis on each class, regardless of its size.

We have found that it is very important to use regular downsampling for all cases in order to prevent the model from almost exclusively predicting only the majority class. At every epoch we use a different downsampled version of the training set as follows: we include all the samples of the minority class, but for the other classes we sample without replacement a number of data samples that is equal to the number of samples of the minority class. This way we ensure that the model receives training samples from each class at the same average frequency and also has the entire training set available to it over the entire training course.

5.1 Deep Model Implementation

In order to maximise the predictive power of our base model before applying data augmentation and to showcase the generalisability of our proposed data augmentation methods, we perform an extensive comparative study among deep architecture variants. We compare among the following word embedding methods: a) **Word2Vec** [23] trained on the hate speech database training set, b) **Google** refers to a Word2Vec model trained[8] on the three billion word Google News corpus, c) **GloVe** [26] trained[9] on the CommonCrawl corpus and d) **FastText** [4] trained[10] on the Wikipedia corpus. All distributional embedding techniques were implemented using the `Gensim` library[11]. The POS-tags were created using the `NLTK` implementation of the Brill Tagger[12].

As for the rest of the deep model, we compare among three different sub-models using fully connected, convolutional and recurrent layer (henceforth FCNN, RNN and CNN) combinations to optimise sentence context learning. The **EMB+CNN+Dense** model takes in the word embedding matrix of 100×300 dimensions to a 1-dimensional CNN layer. We utilised a CNN layer with 32 filters and filter width of 24 which output a 100×32 convolved feature representation. The features are then passed into a max pooling layer that undersamples at a rate of 4. The 25×32 feature space is then flattened to a 600-dimensional vector and passed into an FCNN layer with a rectified linear unit (ReLU) activation function that outputs 25 hidden units. The output FCNN layer is the only layer using softmax activation. As for the **EMB+CNN+LSTM** and **EMB+CNN+GRU** architectures, we add an LSTM RNN or a GRU RNN with 100 hidden units, respectively, before two FCNN layers, as shown in Figure 3. *EMB* is a placeholder for each embedding choice.

5.2 Baseline Reproduction

We identified two past studies that report results on the *HON* database, one being the paper where it was introduced [6] and the other is the first paper in which a deep model was applied on this specific database [47]. The latter approached the dataset as a binary task and has achieved a Micro-F1 score of 94%. The former is an application of logistic regression with L2 regularisation on a set of features that includes extraction of n-grams using a POS-tagger which are treated to term frequency - inverse document frequency (TF-IDF) post-processing, as well as reading ease and sentiment [15] features. The latter is a *CNN+GRU* submodel on top of pre-trained Word2Vec vector embeddings. Our *Google+CNN+GRU* topology is a deeper variant of this method. The baselines are summarised in Table 2.

5.3 Surpassing Competition on *HON* Database

The results of the neural embedding and architecture comparative study are summarised in Table 3 for the largest (*HON*) database. For the best architecture per neural embedding choice, we re-execute the experiment with data augmentation from our best proposed technique, *PosAug*. For the best methodology after augmentation, we additionally apply *ThreshAug* after threshold parameter optimisation (see Sub-Section 5.6) and report results with the best parameter (i. e., 0.70) (see the GloVe+CNN+LSTM+BestAug method). *We see that the application of PosAug brings a consistent improvement over the performance of any base deep model without augmentation in both Macro-F1 and hate class recall in the respective ranges of* $4.6 - 5.4\,\%$ *and* $25.2 - 27.2\,\%$. Our best method (after parameter optimisation) outperforms our variant implementation of *DeepBase* (*Google+CNN+GRU*) for an absolute of 5.7 % in Macro-F1 score and 30 %.

We now focus on the results without data augmentation and we can see in Table 3 that the *EMB+CNN+LSTM* and *EMB+CNN+GRU* topologies seem to be more suitable for capturing sequential word patterns as they outperform *EMB+CNN+Dense* by approximately $1 - 2\,\%$ in Macro-F1 for all embedding choices, with the exception of Word2Vec trained on the database, where *Word2Vec+CNN+GRU* is the best performer in that block. We see a slight positive trend in the results when the *FastText* and *GloVe* embeddings are used.

A noteworthy observation is that our reproduction of *DeepBase* can be improved even further by the choice of a more appropriate

Table 2: Baseline classifiers previously applied on the *HON* database. Reported results are from the original corresponding studies. Micro-F1 and hate recall scores in (%)

Topology	Micro-F1	Hate Recall
LogRegBase [6]	91.0	61.0
DeepBase [47]	94.0*	n/a*

* Binary classification task - hate recall was not reported.

Table 3: Comparison of network topologies on the *HON* database in (%). Each block pertains to a different neural embedding matrix and the reported results correspond to three deep topologies and the best topology per block, with the *PosAug* method applied as well. The final block also contains the best result achieved in the paper, using a combination of *PosAug* and *ThreshAug*.

Topology	Macro-F1	Hate Recall
Word2Vec+CNN+Dense	66.5	15.9
Word2Vec+CNN+LSTM	68.6	20.3
Word2Vec+CNN+GRU	68.7	21.0
Word2Vec+CNN+GRU+PosAug	73.3	48.2
FastText+CNN+Dense	67.6	18.8
FastText+CNN+LSTM	68.4	19.8
FastText+CNN+GRU	68.6	20.4
FastText+CNN+GRU+PosAug	73.4	46.3
Google+CNN+Dense	67.3	17.6
Google+CNN+LSTM	68.4	19.8
Google+CNN+GRU	68.4	19.6
Google+CNN+LSTM+PosAug	73.8	46.9
GloVe+CNN+Dense	67.7	18.8
GloVe+CNN+LSTM	68.8	21.1
GloVe+CNN+GRU	68.8	21.0
GloVe+CNN+LSTM+PosAug	73.6	46.3
GloVe+CNN+LSTM+BestAug	**74.1**	**49.6**

[8] https://github.com/mmihaltz/word2vec-GoogleNews-vectors
[9] https://nlp.stanford.edu/projects/glove
[10] https://github.com/facebookresearch/fastText/blob/master/pretrained-vectors.md
[11] https://radimrehurek.com/gensim
[12] https://www.nltk.org

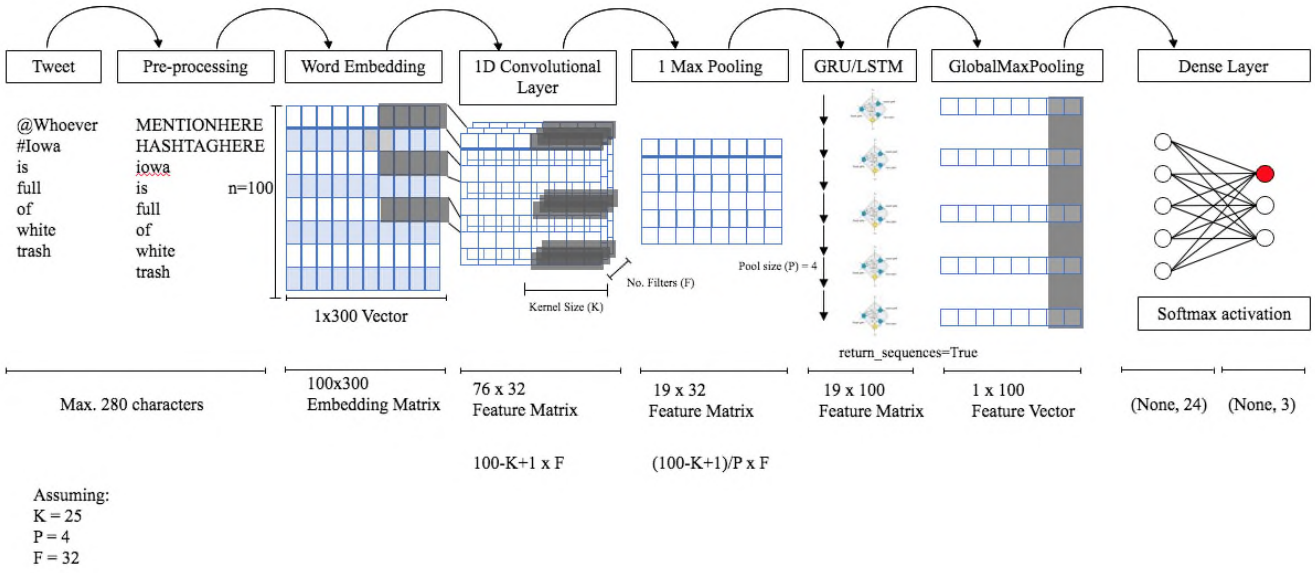

Figure 3: Overview of entire computational process for a single tweet. The deep topology depicted is the best performing family of a neural Embedding + CNN + GRU/LSTM RNN.

neural embedding model. Specifically, we see in Table 3 that the GloVe method is consistently the best choice with respect to F1 score both reported scores across topologies.

5.4 Data Augmentation Experiments

We now take the best performing neural architecture with respect to each embedding from Table 3 and we perform a comparison among the different data augmentation techniques. The results of this comparison can be found in Table 4. We used aligned embeddings and the *ThreshAug* threshold-based examples used a cosine similarity threshold of 0.7.

We notice that the *PosAug* method is very effective, bringing a consistent improvement to the baseline models. All the models had approximately 5 % and 26 − 28 % absolute improvement in Macro-F1 and hate recall scores. The *GenAug* method exhibited overall slightly worse results. Finally, the application of *ThreshAug* yielded an improvement of around 0.3 − 1 % and 2 % increase in Macro-F1 and hate recall, respectively for the *GloVe* and *FastText* embeddings. Interestingly, the two *Word2Vec* methods (pretrained and trained on the dataset) yielded worse results with *ThreshAug*.

5.5 Aligning Embeddings

We will now show in Table 5 that *the embedding used by the model and that by the augmentation technique should be the same to achieve the best possible outcome from the augmentation.* If the embeddings are not the same, the model may be ill-adapted to make predictions on the replacement words which may be close in one representation, but not in the other. We compare the performance of the top four deep topologies between using the Google Word2Vec embeddings and aligned embeddings for augmentation using ThreshAug with threshold parameter equal to 0.7.

The results confirm the initial hypothesis, by showing that the alignment of neural word embeddings in the topology and the augmentation technique yields the highest performance, for the two embeddings that were shown to be a good base for *ThreshAug*, i. e., *FastText* and *GloVe*.

5.6 Threshold Value Robustness

A central question in using *ThreshAug* for data augmentation is how sensitive the method is with respect to the threshold value that the cosine similarity of a similar word must exceed in order to be replaced when generating a new short-text sample. We perform another set of experiments on the *HON* database using the *GloVe+CNN+LSTM* model in which we first apply *PosAug and then* vary the threshold value of *ThreshAug* and we summarise the results in Table 6. *We see that the best value is 0.70, which yields the best F1 score and hate class recall in the paper.*

5.7 Validation Across Databases

In order to show that this technique is not only effective for this particular learning task, but is useful for other learning tasks as well, we ran the same augmentation technique on the *RSN-1* and *RSN-2* databases, as well, *which are smaller and also exhibit higher class imbalance* (0.3 % and 1.3 % minority class samples, respectively). The 'Baseline' in the results shown in Table 7 refers to the highest classification in the papers the databases were introduced. In Table 7, we applied the *GloVe+CNN+GRU* method with no augmentation, *PosAug*, *GenAug* and *ThreshAug* augmentation and report the results. As the database was created some time ago, a lot of the tweets could not be queried anymore, as they are likely to have been removed. Note that the baseline model for the *RSN-1* database [44] only used n-grams and no deep learning architecture, thus

Table 4: Performance comparison of data augmentation methods in %. We compare the best deep topology per neural embedding type. *ThreshAug* was performed using the same embedding as the model and a threshold parameter of 0.7.

Model	No augment		GenAug		PosAug		ThreshAug	
	Macro-F1	*Hate Recall*	*Macro-F1*	*Hate Recall*	*Macro-F1*	*Hate Recall*	*Macro-F1*	*Hate Recall*
Word2Vec+CNN+GRU	68.7	21.0	67.7	19.1	73.3	48.2	46.0	10.0
FastText+CNN+GRU	68.6	20.4	68.2	20.7	73.4	46.3	69.6	22.3
Google+CNN+LSTM	68.4	19.8	67.7	18.7	73.8	46.9	65.1	16.0
GloVe+CNN+LSTM	68.8	21.1	67.0	17.0	73.6	46.3	69.1	23.0

Table 5: Performance comparison of the *ThreshAug* data augmentation technique using aligned embeddings versus using the Google embedding on the *HON* database (in %). The threshold parameter used was 0.7.

Model	Google W2V		Aligned	
	Macro-F1	*Hate Recall*	*Macro-F1*	*Hate Recall*
Word2Vec+CNN+GRU	65.4	17.2	46.0	10.0
Fasttext+CNN+GRU	65.3	16.4	69.6	22.3
Google+CNN+LSTM	65.1	16.0	65.1	16.0
GloVe+CNN+LSTM	64.9	16.0	69.1	23.0

Table 6: Threshold value variation comparison for the *ThreshAug* method using the *GloVe+CNN+LSTM* topology on the *HON* database in (%).

Threshhold	Macro-F1	Hate Recall
0.50	71.6	42.7
0.60	71.6	42.3
0.70	**74.1**	**49.6**
0.80	73.7	46.8
0.90	73.6	47.2

Table 7: Effectiveness of augmentation techniques across databases using the *GloVe+CNN+LSTM* topology (in %).

Model	RSN-1		RSN-2	
	Micro-F1	*Macro-F1*	*Micro-F1*	*Macro-F1*
Baseline	73.9	n/a	91.2	n/a
No Augmentation	82.3	50.7	86.2	31.7
PosAug	80.8	48.3	84.8	31.2
ThreshAug	82.3	50.5	86.4	31.7
GenAug	82.1	51.0	83.8	31.4

achieving a much lower classification performance. *We also achieve competitive results as two recent methods on the RSN-1 database: an ensemble method* [48] *and one* [20] *that uses the Simple Word Embedding Model* [36] *that have achieved a maximum of* 77.8 % and 86.0 %, *respectively. We can surmise that improved data augmentation is vital in order to improve results on such limited datasets.*

6 DISCUSSION

6.1 Neural Networks for Hate Speech Detection

We have proposed a methodology that surpasses the current state-of-the-art in short text hate speech detection. We have conducted a comparative study among various neural embeddings and deep architectures. Hereby, we found that embedding techniques that are pre-trained on a larger text corpus are more effective than those just trained on the training data of the training exercise. These embeddings tend to be better at learning common alternative spellings of words that users may use in order to circumvent blacklisted abusive terms (e. g., 'b$tch', 'assh*le', etc.). After the embedding, we have found that a deep model that comprises a CNN, an RNN (LSTM or GRU) and two FCNN layers is the best choice. This kind of architecture has been shown to be very powerful in past studies on speech recognition [31, 32] and multi-modal speech affect recognition [39]. The best model with augmentation outperformed our re-implementation of a previous deep model by 5.7 % Macro-F1. This improvement mainly stems from *a much better hate speech recall, which has increased by* 30.0 %.

Within our comparisons, we found that the GloVe embeddings were the best performers when used as the base of a deep model. We surmise that the hypothesis made in [26] that GloVe manages to combine the best of counting approaches (e. g., n-gram) and distributional approaches, is a possible explanation for their effectiveness in a deep classification task. After successful augmentation with *PosAug* and *ThreshAug*, we found that the augmentation method dominates in the results and the choice of embedding is a lesser contribution. That being said, our substitution based method was successful only when based on pretrained *FastText* and *GloVe*, possibly indicating that baseline distributional approaches are not suitable for identifying informative word replacements.

Finally, in Table 3, we report the hate speech class recall score, something that has not been given the attention it requires in past deep learning approaches to hate speech detection [47]. We note that even for our two best performing methods *without augmentation*, the measure is significantly less impressive than the F1 score. We would like to perform one more step and report the confusion matrix in Table 8, in which the performance in each class is clearer. *We hypothesise that this phenomenon is caused by the class imbalance problem in this database and was the main motivator for the application of short-text data augmentation.*

Table 8: Confusion Matrix for the *GloVe+CNN+LSTM* model on the *HON* database (without data augmentation).

		Predicted		
		H	O	N
Actual	H	0.20	0.68	0.12
	O	0.02	0.95	0.03
	N	0.01	0.12	0.87

6.2 Data Augmentation Tailored to Short Text

The three of the short text data augmentation methods we proposed perform in a varying way across different deep architectures and databases in both Macro-F1 and minority class recall. *PosAug* was by far the best performing method in the *HON* dataset and *ThreshAug* yielded slight improvement that was orthogonal to *PosAug*, as the combination of the two methods has given the best results. Moreover, the alignment of pre-trained neural embeddings in both the deep model and the replacement method is vital for letting *ThreshAug* perform as best as it can. It is important, then, that the entire system has the same internal model of language.

Data augmentation did not only improve the accuracy, but also made the training of the model less prone to being dominated by the majority class samples. By augmenting the minority class with more samples, hate speech patterns can be extracted more easily. We applied our methods on two other Twitter, multi-class ('racism', 'sexism' and 'neutral'), hate speech datasets. Both datasets also had much stronger class imbalance (see Table 1 making them a good fit for validating the techniques. The results show that the framework is not yet able to generalise in cases of such degree of imbalance in two out of three classes or perhaps in higher resolution hate speech sub-modelling (i. e., racism and sexism) and further work needs to be performed to identify the reason.

We also performed a parameter sensitivity comparison for the combination of *PosAug* and *ThreshAug* and we saw that for small degrees of augmentation (i. e., high threshold parameter) there is noticeable improvement over the baseline. We also note that there is a slightly decreasing trend in performance for values both lower and higher than the optimal. It is important to understand the implications of this phenomenon: When the threshold is very low, we may be able to create a large number of new samples (many replacement words will exceed the threshold), but the samples may be semantically different (high quantity, low quality). On the other hand, if the threshold is too high, the new samples created will most likely be semantically similar (high quality), but too few to achieve the highest possible improvement (not enough quantity).

6.3 Data Augmentation Samples

In Table 9 we show some samples generated using the *ThreshAug* based synonym-replacement for data augmentation. The examples show that the grammatical structure of the sentence does not change because we check for POS-tag equality and as such it is mostly a word-for-word replacement of true synonyms (e. g., 'confused' to 'bewildered', interchangeable due to being adjectives of the same POS tag). *Most importantly, a multitude of similar hate speech words or phrases (e. g., 'white trash', 'out of my country', etc.) are present in the generated tweets, which is very important in order for the rest of the neural architecture to learn hate speech patterns.* Turning to generative augmentation, we show in Table 10 some sample tweets for each class generated with *GenAug*.

The lightly worse performance of *GenAug* compared to *ThreshAug* may be caused by the fact that many of the created samples do not actually make sense from a semantic (or even syntactic) point of view. This is because the samples are completely artificially generated, utilising only a minimal priming for the RNN (as opposed to the substitution methods which provide variations of the original sentences, with similar contexts). For example, one sample that was created for the neutral class was *'just trash away from season man'*. Although this sentence could be considered as grammatically and syntactically acceptable in an online short-text publication platform such as Twitter, it does not make sense, which may lead to the subsequent modelling of unnecessary patterns by our network if a lot of similar quality samples are created. That being said, we do not expect all actual instances of (hate speech) content found in online short-text platforms to be prime examples of high syntactic (or even semantic) quality standards, which is the reason we believe *GenAug* was a sensible hypothesis to test.

Regarding the augmentation performed by *PosAug*, even though the new samples do not resemble actual sentences found in the wild, we consider them to be the same as the original, albeit shifted and warped in a uniform way within the available sequence length. Early experiments on varying the length of the sequence did not indicate a difference in the results.

6.4 Limitations

The generative augmentation technique is limited by the quality of the data, because the generation of new samples can only be as good as the initial dataset. For example, if there are only a hundred samples in the initial training set that trains the *GenAug* weights, this technique is likely to perform relatively poorly. Therefore, we pose that the generative technique should only be attempted when there are relatively large sample sizes available. In any case, as the generative approach generates completely new sequence samples instead of variations of the original samples with slightly altered contexts it does not exploit what we propose is the main strength of the embedding-based replacement augmentation methods: the fact that by training on slight variations of original samples, we let the model learn the pattern of hate speech utterances, instead of focusing on single words that may be associated with an offensive context (but may not be part of hate speech). Another limitation of our conception of *GenAug* is that it requires training one generative model per class and as such the class imbalance problem is inherited here as well. One possible idea for improvement would be to train one model on all available data with vector-based input conditioning.

The current implementation of the synonym replacement data augmentation technique is relatively slow. For each word, the program has to search through the entire vocabulary to find its most similar words, which is an expensive operation. This is not a limitation to the results, but limits the ability to perform extensive parameter optimisation with any confidence for the threshold replacement method, because one would need to fully re-augment the full dataset for each threshold level and then re-train the models.

Table 9: Generated tweets by the *ThreshAug* model. The method is able to find synonyms and also alternative spellings.

Method	Initial Sentence	Augmented version
ThreshAug	*these people should get out of my country*	*these people ought to get out of My nation*
ThreshAug	*you be confused as shit*	*you be bewildered as sh_*_t*

Table 10: Generated tweets by the *GenAug* model. We note that the generated samples may lack in syntactical and grammatical correctness.

Tweet	Class
I hate rooms full of white people they are trash	Hate
The south is full faggot	Hate
bitch what the hell lmfaooo	Offensive
retweet: I swear I love my friends with these hoes	Offensive
np me else wins trust in the bbc	Neutral
I just accidentally ate the pot brownie	Neutral

Another limitation to the findings of this work is imposed by the subjectivity of the topic, as different people have different beliefs about what is offensive and what is hate speech. This subjectivity is reflected by the high annotator disagreement in the *HON* database. As each sample was labelled by at least three people, different coders have differing opinions on what the true label of a sample should be. The fact that even amongst humans there is a relatively high annotator disagreement (81 % of samples of the hate speech class did not exhibit consensus with respect to the annotations) indicates that by attempting to model samples assuming they are properly labelled might introduce errors due to label noise due to the lack of an objective definition.

7 CONCLUSIONS & FUTURE WORK

In summation, in this paper we set out to work on the problem of online hate speech detection in short text format, motivated by the scarcity of previous studies that address the class imbalance problem in online platforms and as such, relevant databases used in research, as well as the scarcity of data augmentation techniques for text. We first attempted to maximise, as far as possible, the performance of a deep network topology, making sure to examine the recall on the minority classes. We have shown that the utilisation of pre-trained neural embeddings on large corpora (especially the Word2Vec and GloVe models pre-trained on the Google News and Common Crawl corpora, respectively, were the best fit for the embedding layer) introduces word context that is outside a limited task-specific database. Most importantly, we found that we can utilise these pre-trained embeddings in a vector similarity based word replacement technique in order to augment the database within the deep text classification framework such that the class imbalance proportions are decreased and also the deep model is encouraged to capture the patterns/context of hate speech instead of learning to associate 'loaded' words with the hate speech label. We have achieved an absolute total of 5.7 % increase in macro-F1 score and up to 30 % in hate speech class recall in the largest online hate speech database. In order to encourage reproducibility we have described the means by which one can get access to all databases and pre-trained neural embeddings and uploaded all code in the GitHub page of the project[13].

As for future work on data augmentation for text, we believe that there are several avenues worth exploring. For example, we believe that the replacement methods could be automated further, e. g., by training a model that predicts which words should be replaced and to also learn word specific similarity thresholds instead of a global one. Even though the generative method we proposed in this paper was not the best performing data augmentation method, we believe that the exploration of generative adversarial networks for data augmentation (e. .g, see [1]) for text can be very promising, as is research in defence against adversarial attacks [12]. For future research, it would be interesting to see how the data augmentation techniques we have proposed perform on longer text inputs or to Natural Language Processing tasks other outside the domain of online hate speech detection. On another matter, we have seen that pre-trained neural embeddings on large corpora for data augmentation can be very helpful in providing synonyms, alternative spellings and misspellings to the words present in a sample short text. Even though FastText was not the best performing word embedding in our comparative study, we believe that sub-word level neural embeddings as in [22] are worth exploring in the short-text hate speech detection task. Finally, we also consider addressing the fact that there is also high annotator disagreement present in the annotation of the samples by utilising Bayesian neural networks (e. g., see [14]) to model this label variance.

ACKNOWLEDGMENTS

This work was supported by the UK Economic & Social Research Council through the research Grant No. HJ-253479 (ACLEW). Georgios Rizos was funded by the Imperial College President's PhD Scholarship scheme. The authors would also like to thank Siyuan Shen for his help on testing and debugging.

REFERENCES

[1] Antreas Antoniou, Amos Storkey, and Harrison Edwards. 2017. Data augmentation generative adversarial networks. *arXiv preprint arXiv:1711.04340* (2017).
[2] 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*. International World Wide Web Conferences Steering Committee, 759–760.
[3] Berit Brogaard. 2016. The number of hate crimes rose in 2016. https://www.psychologytoday.com/us/blog/the-superhuman-mind/201612/hate-crimes-spurned-group-based-hatred. [Accessed: 2018-06-05].
[4] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. Enriching Word Vectors with Subword Information. *arXiv preprint arXiv:1607.04606* (2016).
[5] Eric Brill. 1992. A simple rule-based part of speech tagger. In *Proceedings of the 3rd conference on Applied natural language processing*. Association for Computational Linguistics, 152–155.
[6] Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. [n. d.]. Automated Hate Speech Detection and the Problem of Offensive Language. In *Proceedings of the 11th International AAAI Conference on Web and Social Media*.

[13] https://github.com/glam-imperial/TextAug

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

[8] Nemanja Djuric, Jing Zhou, Robin Morris, Mihajlo Grbovic, Vladan Radosavljevic, and Narayan Bhamidipati. 2015. Hate speech detection with comment embeddings. In *Proceedings of the 24th international conference on world wide web*. ACM, 29–30.

[9] Bin Dong, Zixing Zhang, Björn W. Schuller, et al. 2016. Empirical Mode Decomposition: A Data-Enrichment Perspective on Speech Emotion Recognition. *Emotion and Sentiment Analysis* (2016), 71.

[10] Björn Gambäck and Utpal Kumar Sikdar. 2017. Using Convolutional Neural Networks to Classify Hate-Speech. *Proceedings of the 1st workshop on abusive language online* (2017).

[11] Njagi Dennis Gitari, Zhang Zuping, Hanyurwimfura Damien, and Jun Long. 2015. A lexicon-based approach for hate speech detection. *International Journal of Multimedia and Ubiquitous Engineering* 10, 4 (2015), 215–230.

[12] Tommi Gröndahl, Luca Pajola, Mika Juuti, Mauro Conti, and N Asokan. 2018. All You Need is" Love": Evading Hate-speech Detection. *arXiv preprint arXiv:1808.09115* (2018).

[13] Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, et al. 2014. Deep speech: Scaling up end-to-end speech recognition. *arXiv preprint arXiv:1412.5567* (2014).

[14] José Miguel Hernández-Lobato and Ryan Adams. 2015. Probabilistic backpropagation for scalable learning of bayesian neural networks. In *Proceedings of the International Conference on Machine Learning*. 1861–1869.

[15] CJ Hutto and Eric Gilbert. 2014. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Proceedings of the 8th International Conference on Weblogs and Social Media*.

[16] Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daumé III. 2015. Deep unordered composition rivals syntactic methods for text classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Vol. 1. 1681–1691.

[17] Navdeep Jaitly and Geoffrey E Hinton. 2013. Vocal tract length perturbation (VTLP) improves speech recognition. In *Proceedings of ICML Workshop on Deep Learning for Audio, Speech and Language*, Vol. 117.

[18] Gil Keren, Jun Deng, Jouni Pohjalainen, and Björn W Schuller. 2016. Convolutional Neural Networks with Data Augmentation for Classifying Speakers' Native Language.. In *Proceedings of the 17th International Speech Communication Association INTERSPEECH*. 2393–2397.

[19] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. 1097–1105.

[20] Rohan Kshirsagar, Tyus Cukuvac, Kathleen McKeown, and Susan McGregor. 2018. Predictive Embeddings for Hate Speech Detection on Twitter. *arXiv preprint arXiv:1809.10644* (2018).

[21] Irene Kwok and Yuzhou Wang. 2013. Locate the Hate: Detecting Tweets against Blacks. In *Proceedings of the 27th AAAI Conference on Artificial Intelligence*. 1621–1622.

[22] Yashar Mehdad and Joel Tetreault. 2016. Do Characters Abuse More Than Words?. In *Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*. 299–303.

[23] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[24] Nadia Naffi. 2017. Online Hate Speech in Canada is up 600 percent. https://theconversation.com/the-trump-effect-in-canada-a-600-per-cent-increase-in-online-hate-speech-86026/. [Accessed: 2018-06-10].

[25] Ji Ho Park and Pascale Fung. 2017. One-step and two-step classification for abusive language detection on twitter. *arXiv preprint arXiv:1706.01206* (2017).

[26] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*.

[27] Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365* (2018).

[28] Jing Qian, Mai ElSherief, Elizabeth Belding, and William Yang Wang. 2018. Hierarchical cvae for fine-grained hate speech classification. *arXiv preprint arXiv:1809.00088* (2018).

[29] Grégory Rogez and Cordelia Schmid. 2016. Mocap-guided data augmentation for 3d pose estimation in the wild. In *Advances in Neural Information Processing Systems*. 3108–3116.

[30] Gözde Gül Şahin and Mark Steedman. 2019. Data Augmentation via Dependency Tree Morphing for Low-Resource Languages. *arXiv preprint arXiv:1903.09460* (2019).

[31] Tara N Sainath, Oriol Vinyals, Andrew Senior, and Haşim Sak. 2015. Convolutional, long short-term memory, fully connected deep neural networks. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*. IEEE, 4580–4584.

[32] Tara N Sainath, Ron J Weiss, Andrew Senior, Kevin W Wilson, and Oriol Vinyals. 2015. Learning the speech front-end with raw waveform CLDNNs. In *Proceedings of the 16th Annual Conference of the International Speech Communication Association*.

[33] Justin Salamon and Juan Pablo Bello. 2017. Deep convolutional neural networks and data augmentation for environmental sound classification. *IEEE Signal Processing Letters* 24, 3 (2017), 279–283.

[34] Sam Petulla and Tammy Kupperman and Jessica Schneider. 2017. Hate Crimes Spurned By Group-Based Hatred. https://edition.cnn.com/2017/11/13/politics/hate-crimes-fbi-2016-rise/index.html. [Accessed: 2018-06-25].

[35] Anna Schmidt and Michael Wiegand. 2017. A Survey on Hate Speech Detection using Natural Language Processing. In *Proceedings of the 5th International Workshop on Natural Language Processing for Social Media*. 1–10.

[36] Dinghan Shen, Guoyin Wang, Wenlin Wang, Martin Renqiang Min, Qinliang Su, Yizhe Zhang, Ricardo Henao, and Lawrence Carin. 2018. On the use of word embeddings alone to represent natural language sequences. (2018).

[37] Ilya Sutskever, James Martens, and Geoffrey E Hinton. 2011. Generating text with recurrent neural networks. In *Proceedings of the 28th International Conference on Machine Learning*. 1017–1024.

[38] Toan Tran, Trung Pham, Gustavo Carneiro, Lyle Palmer, and Ian Reid. 2017. A bayesian data augmentation approach for learning deep models. In *Advances in Neural Information Processing Systems*. 2797–2806.

[39] George Trigeorgis, Fabien Ringeval, Raymond Brueckner, Erik Marchi, Mihalis A Nicolaou, Björn Schuller, and Stefanos Zafeiriou. 2016. Adieu features? end-to-end speech emotion recognition using a deep convolutional recurrent network. In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*. IEEE, 5200–5204.

[40] Panagiotis Tzirakis, George Trigeorgis, Mihalis Nicolaou, Björn W. Schuller, and Stefanos Zafeiriou. 2017. End-to-end multimodal emotion recognition using deep neural networks. *IEEE Journal of Selected Topics in Signal Processing* 11, 8 (2017), 1301–1309.

[41] William Warner and Julia Hirschberg. 2012. Detecting hate speech on the world wide web. In *Proceedings of the 2nd Workshop on Language in Social Media*.

[42] Zeerak Waseem. 2016. Are you a racist or am i seeing things? annotator influence on hate speech detection on twitter. In *Proceedings of the 1st workshop on NLP and computational social science*. 138–142.

[43] Zeerak Waseem, Thomas Davidson, Dana Warmsley, and Ingmar Weber. 2017. Understanding abuse: a typology of abusive language detection subtasks. *arXiv preprint arXiv:1705.09899* (2017).

[44] 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*. Association for Computational Linguistics, San Diego, California, 88–93.

[45] Dongxu Zhang and Zhichao Yang. 2018. Word embedding perturbation for sentence classification. *arXiv preprint arXiv:1804.08166* (2018).

[46] Xiang Zhang and Yann LeCun. 2015. Text understanding from scratch. *arXiv preprint arXiv:1502.01710* (2015).

[47] Ziqi Zhang, David Robinson, and Jonathan Tepper. 2018. Detecting Hate Speech on Twitter Using a Convolution-GRU Based Deep Neural Network. In *European Semantic Web Conference*. Springer, 745–760.

[48] Steven Zimmerman, Udo Kruschwitz, and Chris Fox. 2018. Improving Hate Speech Detection with Deep Learning Ensembles. In *Proceedings of the 11th International Conference on Language Resources and Evaluation*.