Detecting hate speech on Twitter using a convolution-GRU based deep neural network

Conference Paper · March 2018

CITATIONS
113

READS
4,084

3 authors, including:

Ziqi Zhang
The University of Sheffield
59 PUBLICATIONS **943** CITATIONS
SEE PROFILE

Jonathan Tepper
Perceptronix Ltd
33 PUBLICATIONS **421** CITATIONS
SEE PROFILE

Detecting Hate Speech on Twitter Using a Convolution-GRU Based Deep Neural Network

Ziqi Zhang*[1], David Robinson[2], and Jonathan Tepper[2]

[1] University of Sheffield, UK
[2] Nottigham Trent University, UK
ziqi.zhang@sheffield.ac.uk, david.robinson2015@my.ntu.ac.uk,
jonathan.tepper@ntu.ac.uk

Abstract. In recent years, the increasing propagation of hate speech on social media and the urgent need for effective counter-measures have drawn significant investment from governments, companies, and empirical research. Despite a large number of emerging scientific studies to address the problem, a major limitation of existing work is the lack of comparative evaluations, which makes it difficult to assess the contribution of individual works. This paper introduces a new method based on a deep neural network combining convolutional and gated recurrent networks. We conduct an extensive evaluation of the method against several baselines and state of the art on the largest collection of publicly available Twitter datasets to date, and show that compared to previously reported results on these datasets, our proposed method is able to capture both word sequence and order information in short texts, and it sets new benchmark by outperforming on 6 out of 7 datasets by between 1 and 13 percents in F1. We also extend the existing dataset collection on this task by creating a new dataset covering different topics.

1 Introduction

In recent years, the exponential growth of social media such as Twitter and community forums has been increasingly exploited for the propagation of hate speech and the organisation of hate based activities [1,4]. The anonymity and mobility afforded by such media has made the breeding and spread of hate speech - eventually leading to hate crime - effortless in a virtual landscape beyond the realms of traditional law enforcement. In the UK, there has been significant increase of hate speech towards the migrant and Muslim communities following recent events including leaving the EU, and the Manchester and the London attacks [12]. This correlates to record spikes of hate crimes, and cases of threats to public safety due to its nature of inciting hate crimes, such as that followed the Finsbury van attack [3]. Surveys and reports also show the rise of hate speech and related crime in the US following the Trump election [18], and the EU where 80% of young people have encountered hate speech online and 40% felt attacked or threatened [2].

* The work was carried out when the author was at the Nottingham Trent University.

Social media companies (e.g., Twitter, Facebook) have come under pressure to address this issue [13], and it has been estimated that hundreds of millions of euros are invested every year [10]. However, they are still being criticised for not doing enough. This is largely because standard measures involve manually reviewing online contents to identify and delete offensive materials. The process is labour intensive, time consuming, and not sustainable in reality [10,26].

The pressing need for scalable, automated methods of hate speech detection has attracted significant research using semantic content analysis technologies based on Natural Language Processing (NLP) and Machine Learning (ML) [4,7,9,10,16,17,20,23,24,26,25,27]. Despite this large amount of work, it remains difficult to compare their performance [21], largely due to the use of different datasets by each work and the lack of comparative evaluations. This work makes three contributions to this area of work. First, we use a CNN+GRU (convolutional neural network, gated recurrent unit network) neural network model optimised with dropout and pooling layers, and elastic net regularisation for better learning accuracy. Compared to existing deep learning models that use only CNNs, the GRU layer also captures sequence orders that are useful for this task. Second, we create a public dataset of hate speech by collecting thousands of tweets on the subjects of religion and refugees, which extends currently available datasets [7,26,25] by both quantity and subject coverage. Third, we test our model against several baselines and also previously reported results on all existing public datasets, and set new benchmark as we show that our model outperforms on 6 out of 7 datasets by as much as 13% in F1. We also undertake error analysis to identify the challenges in hate speech detection on social media.

The remainder of this paper is structured as follows. Section 2 reviews related work; Section 3 introduces the CNN+GRU model; Section 4 describes our datasets, followed by experiments in Section 5 and conclusion in Section 6.

2 Related Work

2.1 Terminology

Recent years have seen an increasing number of research on hate speech detection as well as other related areas. As a result, the term 'hate speech' is often seen mixed with other terms such as 'offensive', 'profane', and 'abusive languages', and 'cyber bullying'. To distinguish them, we identify that hate speech 1) targets individual or groups on the basis of their characteristics (targeting characteristics); 2) demonstrates a clear intention to incite harm, or to promote hatred; 3) may or may not use offensive or profane words. For example:

`'Assimilate? No they all need to go back to their own countries. #BanMuslims Sorry if someone disagrees too bad.'`

In contrast, `'All you perverts (other than me) who posted today, needs to leave the O Board. Dfasdfdasfadfs'` is an example of abusive language, which often bears the purpose of insulting individuals or groups, and can include hate speech, derogatory and offensive language [17]. `'i spend my money how i`

`want bitch its my business'` is an example of offensive or profane language, which is typically characterised by the use of swearing or curse words. `'Our class prom night just got ruined because u showed up. Who invited u anyway?'` is an example of bullying, which has the purpose to harass, threaten or intimidate typically individuals rather than groups.

In the following, we cover state of the art in all these areas with a focus on hate speech. Our experiments will only involve hate speech, due to both dataset availability and the focus of this work.

2.2 State of the Art

State of the art primarily casts the problem as a supervised document classification task [21]. These methods can be divided into two categories: one relies on manual feature engineering that are then consumed by algorithms such as SVM, Naive Bayes, and Logistic Regression [4,7,9,16,24,26,25,27] (**classic methods**); the other represents the more recent deep learning paradigm that employs neural networks to automatically learn multi-layers of abstract features from raw data [10,17,20,23] (**deep learning methods**).

Classic methods. [21] summarised several types of features used in the state of the art. *Simple surface features* such as bag of words, word and character n-grams have shown to be highly predictive in hate speech detection [4,16,23,24,26,25], as well as other related tasks [17,28]. Other surface features can include URL mentions, hashtags, punctuations, word and document lengths, capitalisation, etc [7,17]. *Word generalisation* includes the use of word clusters [24], and techniques such as topic modelling [27,28] and word embedding learning [9,17,23] that learn low-dimensional, dense feature vectors for words from unlabelled corpora. Such word vectors are then used to construct feature vectors of messages. *Sentiment analysis* makes use of the degree of polarity expressed in a message [4,7,11,23]. *Lexical resources* are often used to look up specific negative words (such as slurs, insults, etc.) in messages as their presence can be predictive features [4,11,17,27]. *Linguistic features* utilise syntactic information such as Part of Speech (PoS) and certain dependency relations as features [4,7,11,28]. For example, [4] noted that 'othering phrases' denoting a 'we v.s. them' stance are common in hate speech. *Meta-information* refers to data about messages, such as gender identity of a user associated with a message [26,25], or high frequency of profane words in a user's post history [27]. In addition, *Knowledge-Based features* such as messages mapped to stereotypical concepts in a knowledge base [8] and *multimodal information* such as image captions and pixel features [28] were used in cyber bully detection but only in very confined context [21].

In terms of the classification algorithms, Support Vector Machines (SVM) is the most popular algorithm [4,7,16,24,27]. Others can include as Naive Bayes [7,16], Logistic Regression [7,9,16,26,25], and Random Forest [7,27].

Deep learning based methods employ Deep artificial Neural Networks (DNN) to learn abstract feature representations from input data through its multiple

stacked layers for the classification of hate speech. The input can take various forms of feature encoding, including many of those used in the classic methods. However, the input features are not directly used for classification. Instead, the multi-layer structure learns new abstract feature representations that are used for learning. For this reason, deep learning based methods focus on the design of the network topology, which automatically extracts useful features from a simple input feature representation. Note that this excludes those methods [9,16] that use DNN to learn word or text embeddings and subsequently apply another classifier (e.g., SVM) to use such embeddings as features for classification. Instead, we focus on DNN methods that perform the classification task itself.

To the best of our knowledge, methods belonging to this category include [1,10,20,23], all of which use simple word and/or character based one-hot encoding as input features to their models, while [23] also use word polarity. The most popular network architectures are Convolutional Neural Network (CNN) and Recurrent Neural Networks (RNN), typically Long Short-Term Memory network (LSTM). In the literature, CNN is well known as an effective network to act as 'feature extractors', whereas RNN is good for modelling orderly sequence learning problems [19]. In the context of hate speech classification, intuitively, CNN extracts word or character combinations [1,10,20] (e.g., phrases, n-grams), RNN learns word or character dependencies (order information) in tweets [1,23].

While each type of network has been used for hate speech classification, no work has explored combining both structures for this task. In theory, CNN+RNN networks are powerful structures to capture order information between features extracted by CNNs. In practice, they are found to be more effective than structures solely based on CNNs or RNNs in tasks such as gesture [22] and activity recognition [19], and Named Entity Recognition [5]. In hate speech classification, we hypothesise that a CNN+RNN type of structure can be more effective as it may capture co-occurring word n-grams as useful patterns for classification, such as the pairs `(muslim refugees, deported)` and `(muslim refugees, not welcome)` in the sentence `'These muslim refugees are not welcome in my Country they should all be deported ...'`. However, practically, it remains a question as to what extent such orderly information is present in short messages, and how can we optimise such a network for classification accuracy.

2.3 Datasets

It is widely recognised that a major limitation in this area of work is the lack of comparative evaluation [21]. The large majority of existing works were evaluated on privately collected datasets for different problems. [17] claimed to have created the largest datasets for abusive language by annotating comments posted on Yahoo! Finance and News. The datasets were later used by [16]. However, they are not publicly available. Also, as we illustrated before, abusive language can be different from hate speech. Currently, the only publicly available hate speech datasets include those reported in [7,10,20,26,25], which are all Twitter-based. [26] annotated 16,914 tweets into 'sexism', 'racism' and 'neither'. The corpus was collected by searching for tweets containing frequently occurring terms (based

on some manual analysis) in tweets that contain hate speech and references to specific entities. It was then annotated by crowd-sourcing over 600 users. The dataset was later expanded in [25], with about 4,000 as new addition to their previous dataset. This dataset was then annotated by two groups of users to create two different versions: domain experts who are either feminist or anti-racism activist; and amateurs that are crowd-sourced. Later in [10], the authors merged both expert and amateur annotations in this dataset by using majority vote; and in [20], the dataset of [26] was merged with the expert annotations in [25] to create a single dataset. [7] annotated some 24,000 tweets for 'hate speech', 'offensive language but not hate', and 'neither'. These were sampled from tweets filtered using a hate speech lexicon from Hatebase.org.

3 Methodology

3.1 Pre-processing

Given a tweet, we start by applying a light pre-processing procedure described below based on that reported in [7] to normalise its content. This includes: 1) removing the characters `| : , ; & ! ? \`; 2) applying lowercase and stemming, to reduce word inflections; and 3) removing any tokens with a document frequency less than 5, which reduces sparse features that are not informative for learning. Empirically, this was found to lead to better classification accuracy.

Further, we also normalise hashtags into words, so `'#refugeesnotwelcome'` becomes `'refugees not welcome'`. This is because such hashtags are often used to compose sentences. We use dictionary based look up to split such hashtags.

3.2 The CNN+GRU architecture

Our CNN+GRU network is illustrated in Figure 1. The first layer is a word embedding layer, which maps each text message (in generic terms, a 'sequence') into a real vector domain. To do so, we map each word onto a fixed dimensional real valued vector, where each element is the weight for that dimension for that word. In this work, we use the word embeddings with 300 dimensions pre-trained on the 3-billion-word Google News corpus with a skip-gram model[3] to set the weights of our embedding layer. We also constrain each sequence to be 100 words which is long enough to encode tweets of any length, truncating long messages and pad the shorter messages with zero values.

The embedding layer passes an input feature space with a shape of 100×300 to a drop-out layer with a rate of 0.2, the purpose of which is to regularise learning to avoid overfitting. Intuitively, this can be thought of as randomly removing a word in sentences and forcing the classification not to rely on any individual words. The output feeds into a 1D convolutional layer with 100 filters with a window size of 4, padding the input such that the output has the same length as the original input. The rectified linear unit function is used for activation.

[3] https://github.com/mmihaltz/word2vec-GoogleNews-vectors

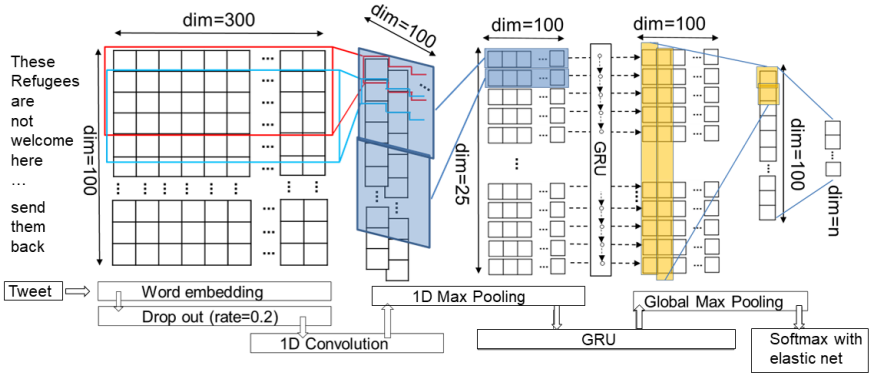

Fig. 1. The CNN+GRU network architecture. This diagram is best viewed in colour.

This convolves the input feature space into a 100×100 representation, which is then further down-sampled by a 1D max pooling layer with a pool size of 4 along the word dimension, producing an output of shape 25×100. Each of the 25 dimensions can be considered an 'extracted feature'.

These extracted features then feed into the GRU layer, which treats the feature dimension as timesteps and outputs 100 hidden units per timestep. The key difference between a GRU and an LSTM is that a GRU has two gates (reset and update gates) whereas an LSTM has three gates (namely input, output and forget gates). Thus GRU is a simpler structure with fewer parameters to train. In theory, this makes it faster to train and generalise better on small data; while empirically it is shown to achieve comparable results to LSTM [6]. Next, a global max pooling layer 'flattens' the output space by taking the highest value in each timestep dimension, producing a 1×100 vector. Intuitively, this selects the highest scoring features to represent a tweet. Finally, a softmax layer takes this vector as input to predict probability distribution over all possible classes (n), which will depend on individual datasets. The softmax activation is also regularised using the elastic net regularisation that linearly combines the L_1 and L_2 norms, but is designed to solve the limitations of both [29].

We use the categorical cross entropy loss function and the Adam optimiser to train the model. The first is empirically found to be more effective on classification tasks than other loss functions including classification error and mean squared error [15], and the second is designed to improve the classic stochastic gradient descent (SGD) optimiser and in theory combines the advantages of two other common extensions of SGD (AdaGrad and RMSProp) [14].

Model parameters. As described above, many of our model parameters are based on empirical findings reported previously, default values or anecdotal evidence, except the batch size and epoches for training which we derive from developmental data (to be detailed later). Arguably, these may not be the best settings for optimal results, which are always data-dependent. However, we show

later in experiments that the model already obtains promising results even without extensive data-driven parameter tuning.

Comparison against similar DNNs. Our network architecture is similar to those in [5,19,22]. The differences include: 1) we use a GRU instead of an LSTM for the reasons stated before; 2) we add a drop-out layer to regularise learning and a global max pooling layer to extract features from the GRU; 3) we use elastic net to regularise the output from the softmax layer. [5] also used bi-directional LSTM in the Named Entity Classification task, while [19,22] stacked multiple convolutional layers to extract hierarchical features in image processing. We do not use such complex models as we show that our simple CNN+GRU is already performing well in this task and is able to benefit from both convolutional features and order information captured by GRU, which confirms our hypothesis. For the same reason, we build our model on only word embeddings although many suggest that character-level features can be more effective. We show later in experiments that such a structure is very powerful that it even outperforms DNN models based on character embeddings.

4 Dataset creation

As introduced before, the only publicly available hate speech datasets include that of [7], and [26,25] that are also used to create variants used in [10,20]. While [7] classify hate speech in general without identifying the targeting characteristics, [26,25] collected data for sexism and racism. In this work, we create a different dataset by collecting tweets discussing refugees and Muslims, which were media focus during the time of writing due to various recent incidents [12,3]. All tweets are annotated for two classes: hate and non-hate, firstly by a computational linguistic researcher and then cross-checked by a student researcher. Disputed annotations were then discussed and corrected to ensure both agree with the correction. The general annotation guideline in [26] were followed.

To collect the data, we follow the mainstream approach [26] starting with an initial search for tweets containing common slurs and terms used pertaining to targeting characteristics, then manually identify frequently occurring terms in tweets that contain hate speech and references to specific entities (frequent keywords), then further filter the tweets with these frequent keywords.

Specifically, we started with using the Twitter Streaming API to collect tweets containing any of the following words for a period of 7 days: `muslim, islam, islamic, immigration, migrant, immigrant, refugee, asylum`. This created a corpus of over 300,000 tweets (duplicates and retweets removed), from which we randomly sampled 1,000 for annotation (batch 1). However, it was found that tweets annotated as hate speech were extremely rare ($< 1\%$). Therefore, we manually inspected the annotations and further filtered the remaining tweets (disjoint with batch 1) by the following words found to be frequent for hate speech: `ban, kill, die, back, evil, hate, attack, terrorist, terrorism, threat, deport`. We then sampled another 1,000 tweets (batch 2)

from this collection for annotation. However, the amount of true positives was still very low (1.1%). Therefore we created another batch (batch 3) by using the Twitter Search API to retrieve another 1,500 tweets with the following hashtags considered to be strong indicators of hate speech: `#refugeesnotwelcome`, `#DeportallMuslims`, `#banislam`, `#banmuslims`, `#destroyislam`, `#norefugees`, `#nomuslims`. The dataset however, contains over 400 tweets after removing duplicates, and about 75% were annotated as hate speech. Finally we merge all three batches to create a single dataset, which we make public to encourage future comparative evaluation[4].

5 Experiment

5.1 Datasets

We use a total of 7 public datasets including ours for evaluation, as shown in Table 1. To our knowledge, this is by far the most comprehensive collection of hate speech datasets used in any studies.

Dataset	**#Tweets**	**Classes** (#tweets)	**Targeting characteristics**
WZ-L	16,093	racism (1,934) sexism (3,149) neither (11,010)	racism, sexism
WZ-S.amt	6,594	racism (123) sexism (1,073) both (15) neither (5,383)	racism, sexism
WZ-S.exp	6,594	racism (85) sexism (777) both (35) neither (5,697)	racism, sexism
WZ-S.gb	6,594	racism (90) sexism (911) both (27) neither (5,564)	racism, sexism
WZ-LS	18,625	racism (2,012) sexism (3,769) both (30) neither (12,810)	racism, sexism
DT	24,783	hate (1,430) non-hate (23,353)	general
RM	2,435	hate (414) non-hate (2,021)	refugee, Muslim

Table 1. Statistics of datasets used in the experiment

WZ-L is the larger dataset created in [26]. **WZ-S.amt** is the smaller dataset created in [25], annotated by amateurs; while **WZ-S.exp** is the same dataset annotated by experts. In [10], the authors took the WZ-S.amt and WZ-S.exp datasets to create a new version by taking the majority vote from both amateur and expert annotations where the expert was given double weights. We follow the same practice and in case of tie, we take the expert annotation. We refer to this dataset as **WZ-S.gb**. Note that although WZ-S.amt, WZ-S.exp and WZ-S.gb datasets contain the same set of tweets, state of the art results are found to be

[4] Find out at: `https://github.com/ziqizhang/chase/tree/master/data`

quite different on some of them (see Section 5.4), suggesting that the annotations can be different. Further, [20] combined the WZ-L and the WZ-S.exp datasets into a single dataset and in case of duplicates, we take the annotation from WZ-L. We refer to this dataset as **WZ-LS**. All datasets only contain the tweet IDs, some of which have been deleted or made private at the time of writing and therefore, the numbers in Table 1 may be different from the original studies.

DT refers to the dataset created in [7]. It also contains tweets annotated as 'abusive (but non-hate)'. In this work, we set such annotations to be 'non-hate' so the dataset contains only two classes. Finally, our dataset on refugees and Muslims is referred to as **RM**.

5.2 Baseline and comparative models

Baselines. We create a number of baselines. **First**, we use the SVM model described in [7][5]. Each tweet is firstly pre-processed using the procedure described in Section 3.1. Next, following the original work, a number of different types of features are used as below. We refer to these as the **Basic** feature set:

- Surface features: word unigram, bigram and trigram each weighted by TF-IDF; number of mentions, and hashtags[6]; number of characters, and words;
- Linguistic features: Part-of-Speech (PoS) tag unigrams, bigrams, and trigrams, weighted by their TF-IDF and removing any candidates with a document frequency lower than 5; number of syllables; Flesch-Kincaid Grade Level and Flesch Reading Ease scores that to measure the 'readability' of a document
- Sentiment features: sentiment polarity scores of the tweet, calculated using a public API[7].

Extending on this, we add additional surface based features as follows and refer to these as the **Enhanced** feature set:

- number of misspellings within a tweet: we check the pre-processed tweet against both a UK and US English dictionaries, then calculate the ratio between the number of misspelled words and the number of all words.
- number of emoji's uses regular expression to find tokens matching the format of an emoji from an original tweet and return a number.
- number of special punctuations uses regular expression to count the numbers of question and exclamation marks as they can be used as an expletive.
- percentage of capitalisation computes the percentage of characters that are capitalised within the tweet.
- hashtags: the lowercase hashtags from tweets.

[5] Code: `https://github.com/t-davidson/hate-speech-and-offensive-language`
[6] Extracted from the original tweet before pre-processing which splits hashtags.
[7] https://github.com/cjhutto/vaderSentiment

We use **SVM** to refer to the model using all Basic features identified above, and **SVM+** as the one using the Enhance feature set. Notice that the SVM baseline is also the model used in [7].

Second, we create another two baseline by modifying our CNN+GRU network. First, we remove the drop-out and the global max pooling layers and the elastic net regularisation, to create **CNN+GRU$_\mathbf{B}$**; second, we further remove the GRU layer to create a CNN only network **CNN**. While the first allows us to evaluate the effect of the modifications on a basic CNN+GRU structure, the second allows us to evaluate whether GRU does capture useful order information from short messages such as tweets. We apply all these baselines on all seven datasets and compare the results against our model, **CNN+GRU**.

State of the art. In addition to the baselines, we also compare our results against those reported figures in [26,25,10,20] on the corresponding datasets.

5.3 Implementation, parameter tuning, and evaluation metrics

We used the Python Keras[8] with Theano backend [9] and the scikit-learn[10] library to implement all models[11]. For each dataset, we split it into 75:25 to use 75% in a grid-search to tune batch size and learning epoches using 5-fold cross-validation experiments, and test the optimised model on the 25% held-out data. We report results using the standard Precision (P), Recall (R), and F1-measures.

5.4 Results and discussion

Table 2 compares our model against the baselines as well as state of the art on F1[12] (on an 'as-is' basis where available, indicated by citations) on each of the seven datasets. The highest figures are highlighted in **bold**.

Baselines performance. Among the four baselines, neural network based methods consistently outperformed SVM and SVM+, by as much as 9% on the WZ-LS dataset. For SVM+, adding enhanced features leads to incremental improvement over SVM, as we notice increase in F1 on 6 out of 7 datasets. Comparing CNN+GRU$_\mathrm{B}$ against CNN, it shows that adding the GRU recurrent layer does bring further improvement as we notice a 1% percent improvement on five out of seven datasets. While the relatively small improvement could be due to the short text nature of tweets, the consistent gain in F1 suggests that GRU still captures useful order information in addition to CNNs.

CNN+GRU performance. Compared against baselines, our CNN+GRU model achieves the highest F1 on all datasets. Compared against CNN+GRU$_\mathrm{B}$, the

[8] `https://keras.io/`
[9] `http://deeplearning.net/software/theano/`
[10] `http://scikit-learn.org/`
[11] Code available at `https://github.com/ziqizhang/chase`
[12] We use micro-average to be comparable to previously reported results.

Dataset	SVM	SVM+	CNN	CNN+ GRU_B	CNN+ GRU	State of the art
WZ-L	0.74	0.74	0.80	0.81	**0.82**	0.74 Waseem [26], best F1
WZ-S.amt	0.86	0.87	0.91	**0.92**	**0.92**	0.84 Waseem [25], Best features
WZ-S.exp	0.89	0.90	0.90	0.91	**0.92**	0.91 Waseem [25], Best features
WZ-S.gb	0.86	0.87	0.91	0.92	**0.93**	0.90 Gamback [10], best F1
WZ-LS	0.72	0.73	0.81	0.81	0.82	0.82 Park [20], WordCNN 0.81 Park [20], CharacterCNN **0.83** Park [20], HybridCNN
DT	0.87	0.89	**0.94**	**0.94**	**0.94**	0.87 SVM, Davidson [7]
RM	0.86	0.89	0.90	0.91	**0.92**	0.86 SVM, Davidson [7]

Table 2. F1 result of the proposed method against baselines and state of the art. The best figures are highlighted in **bold**. The 'Best features' is a setting reported in [25], where their method used the combination of the best performing features.

improved model sees further incremental improvement on 5 datasets by adding dropout, max pooling and elastic net regularisation. Compared against SVM and SVM+ baselines, CNN+GRU makes an improvement between 2 and 9 percent. Compared to previously reported results in the state of the art, our CNN+GRU model obtained an improvement of: 7% on WZ-L, 1% on WZ-S.exp, 8% on WZ-S.exp, 13% on WZ-S.gb, 7% on DT, and 6% on RM. On the WS-LS dataset, our model outperforms [20] on their character-only CNN models, losing 1% to their hybrid model that combines both word and character features. Similarly, [10] also used character features in their CNN model. As discussed before, the literature generally acknowledges that character-based features are more effective than word-based. Hence taking into account the above results, we argue that the better results obtained by our CNN+GRU models using only word-based features is due to the superiority in the network architecture.

5.5 Error analysis

We carry out further studies to analyse the errors made by the classifiers to understand the challenges for this task. To do so, on each dataset and for each class, we identify the tweets where all three neural network models (CNN+GRU, CNN+GRU_B, CNN) predicted incorrectly. Then in Table 3 we show the distribution of such errors across all classes for each dataset. The figures show that, on the WZ-S.amt, WZ-LS and WZ-L datasets, the classifiers make about the same number of errors on the 'racism' and 'non-hate' tweets, and twice the errors on the 'sexism' tweets. If we factor in the number of tweets per class on these datasets (see Table 1), it is obvious that the classifiers make relatively more errors on predicting any categories of hate tweets than non-hate. The situation is similar on the WZ-S.exp and WZ-S.gb datasets, and intensifies on the DT and RM datasets where only 'hate' and 'non-hate' are considered. Interestingly, increasing training data does not always reduce errors proportionally. For example, comparing the WZ-S.amt against the WZ-L datasets in Table 1, the

number of 'sexism' tweets increases from 16% of all instances to 20% and the number of 'racism' tweets increases from 2% to 12%. However, the classifiers did not seem to benefit very much as errors have increased by 8% on 'sexism' tweets and dropped only 3% on 'racism' tweets (Table 3).

	Racism	**Sexism**	**Both**	**Non-hate**	**Hate**
WZ-S.amt	25%	46%	3%	26%	N/A
WZ-S.exp	12%	52%	8%	28%	N/A
WZ-S.gb	14%	50%	7%	29%	N/A
WZ-LS	23%	52%	1%	23%	N/A
WZ-L	22%	54%	N/A	24%	N/A
DT	N/A	N/A	N/A	6%	94%
RM	N/A	N/A	N/A	30%	70%

Table 3. Distribution of errors over all classes in each dataset.

We further analyse the vocabularies of each class on each dataset. For each class, we calculate two statistics. **Instance-2-Uwords, I2U** ratio divides the number of instances (tweets) of the class by the number of unique words found in that class. This measures on average, the number of instances sharing at least one word. Intuitively, a higher number suggests that instances of the class are more likely to have overlap in words. We hypothesise that this translates to overlap in features, i.e., the features of instances of this class are dense; and therefore, these instances may be easier to classify. **Uwords-2-Class, U2C** ratio divides the number of unique words found *only* in that class (i.e., the words must not be present in other classes) by the number of unique words in that class (i.e., the words can be present in other classes at the same time). Intuitively, if a class has a lot of words that are unique to itself, it may have many unique features that makes it easier to classify. Table 4 shows the two ratios for different datasets.

	Racism		**Sexism**		**Both**		**Non-hate**		**Hate**	
	I2U	U2C	I2U	U2C	I2U	U2C	I2U	U2C	I2U	U2C
WZ-S.amt	.25	0	.84	0	.12	0	3.1	.22	-	-
WZ-S.exp	.26	.003	.70	.006	.19	0	3.3	.32	-	-
WZ-S.gb	.28	.003	.77	.008	.19	0	3.2	.29	-	-
WZ-LS	1.0	.004	1.4	.008	.16	0	3.8	.11	-	-
WZ-L	1.1	.004	1.3	.008	-	-	3.5	.11	-	-
DT	-	-	-	-	-	-	6.3	.45	.71	0
RM	-	-	-	-	-	-	2.2	.31	.65	.009

Table 4. Analysis of Instance-2-Uwords (I2U) and Uwords-2-Class (U2C) ratios.

Firstly, comparing 'non-hate' against any other classes of hate tweets, 'non-hate' tweets have much higher I2U and U2C scores. For example, on the DT

dataset, on average, more than 6 tweets will share at least one word (I2U=6.3), and 'non-hate' tweets have 45% of words that are unique to that class. Comparatively the figures are much lower for the 'hate' tweets on this dataset. Intuitively, this may mean that 'non-hate' tweets have much higher overlap in their features and they also have much more unique features compared to 'hate'. Both make 'non-hate' easier to predict, hence explaining the significantly higher errors on the 'hate' class on this dataset. Again the situation is similar on other datasets. On WZ based datasets, the relatively lower I2U and U2C values for 'racism' than 'sexism' also suggests that the first is a harder class to predict. This may explain the observation before that the increase in data on this class did not proportionally translate to improvement in classification accuracy.

We manually analysed a sample of 200 tweets from the WZ-S.amt, DT and RM datasets, covering all classes to identify tweets that are difficult to classify. **Non-distinctive features** appear to be the majority of cases. For example, one would assume that the presence of the phrase `'white trash'` or pattern `'* trash'` is more likely to be a strong indicator of hate speech than not, such as in `'White bus drivers are all white trash...'`. However, our analysis shows that such seemingly 'obvious' features are also prevalent in non-hate tweets such as `'... I'm a piece of white trash I say it proudly'`. As we show previously in Table 4, hate tweets in our datasets lack unique features to distinguish themselves from non-hate tweets. The second example does not qualify as hate speech since it does not 'target individual or groups' or 'has the intention to incite harm', which is indeed often very subtle to identify from lexical or syntactic levels. Similarly, **subtle metaphors** are often commonly found in false negatives such as `'expecting gender equality is the same as genocide'`. **Out of embedding vocabulary (OOV) words** are words that are frequent in hate tweets, but are not included in the word embedding model we used. For example, `'faggot'` in the hate tweet `'I'm just upset they got faggots on this show'` is an OOV word in the Google embedding model. This raises the question whether using a different or multiple word embedding models to reduce OOV can further improve learning accuracy. **Questioning or negation** is often found in false positives such as `'I honestly hate the term 'feminazi' so much. Stop it'`. Further, expression of **Stereotypical views** such as in `'... these same girls ... didn't cook that well and aren't very nice'` is also common in false negative sexism tweets. These are difficult to capture because they require understanding of the implications of the language.

6 Conclusion and future work

The propagation of hate speech on social media has been increasing significantly in recent years and it is recognised that effective counter-measures rely on automated data mining techniques. This work makes several contributions to this problem. First, we introduced a method for automatically classifying hate speech on Twitter using a deep neural network model combining CNN and GRU that are found to empirically improve classification accuracy. Second, we conducted

comparative evaluation on the largest collection of public datasets and show that the proposed method outperformed baselines as well as state of the art in most cases. Our results make new reference for future comparative studies. Third, we created and published another hate speech dataset, complementing existing ones by focusing on religion (Muslim) and refugees. Finally, our analysis shows that the presence of abstract concepts such as 'sexism', 'racism' or 'hate' is very difficult to detect if solely based on textual content. But the task may potentially benefit from knowledge about social groups and communication modes.

We will explore future work in a number of ways, such as other network structures to extracting different features; explore different word embeddings' effect on learning; and integrate user-centric features, such as the frequency of a user detected for posting hate speech and the user's interaction with others.

Acknowledgement. Part of this work was conducted during the SPUR project funded by the Nottingham Trent University. We also thank Qian Wang, a student funded by the Nuffield Foundation for data analysis in this work.

References

1. P. Badjatiya, S. Gupta, M. Gupta, and V. Varma. Deep learning for hate speech detection in tweets. In *Proceedings of the 26th International Conference on World Wide Web Companion*, pages 759–760, 2017.
2. BBCNews. Countering hate speech online, Last accessed: July 2017, http://eeagrants.org/News/2012/.
3. BBCNews. Finsbury park attack: Son of hire boss held over facebook post, Last accessed: May 2017, http://www.bbc.co.uk/news/uk-wales-40347813.
4. P. Burnap and M. L. Williams. Cyber hate speech on twitter: An application of machine classification and statistical modeling for policy and decision making. *Policy and Internet*, 7(2):223–242, 2015.
5. J. P. Chiu and E. Nichols. Named entity recognition with bidirectional lstm-cnns. *Transactions of the Association for Computational Linguistics*, 4:357–370, 2016.
6. J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NIPS 2014 Deep Learning and Representation Learning Workshop*, 2014.
7. T. Davidson, D. Warmsley, M. Macy, and I. Weber. Automated hate speech detection and the problem of offensive language. In *Proceedings of the 11th Conference on Web and Social Media*. AAAI, 2017.
8. K. Dinakar, B. Jones, C. Havasi, H. Lieberman, and R. Picard. Common sense reasoning for detection, prevention, and mitigation of cyberbullying. *ACM Trans. Interact. Intell. Syst.*, 2(3):18:1–18:30, Sept. 2012.
9. N. Djuric, J. Zhou, R. Morris, M. Grbovic, V. Radosavljevic, and N. Bhamidipati. Hate speech detection with comment embeddings. In *Proceedings of the 24th International Conference on World Wide Web*, pages 29–30. ACM, 2015.
10. B. Gambäck and U. K. Sikdar. Using convolutional neural networks to classify hate-speech. In *Proceedings of the First Workshop on Abusive Language Online*, pages 85–90. Association for Computational Linguistics, 2017.
11. N. D. Gitari, Z. Zuping, H. Damien, and J. Long. A lexicon-based approach for hate speech detection. *International Journal of Multimedia and Ubiquitous Engineering*, 10(10):215–230, 2015.

12. Guardian. Anti-muslim hate crime surges after manchester and london bridge attacks, Last accessed: July 2017, https://www.theguardian.com.
13. Guardian. Zuckerberg on refugee crisis: 'hate speech has no place on facebook', Last accessed: July 2017, https://www.theguardian.com.
14. D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference for Learning Representations*, 2015.
15. J. D. McCaffrey. Why you should use cross-entropy error instead of classification error or mean squared error for neural network classifier training, Last accessed: Jan 2018, https://jamesmccaffrey.wordpress.com.
16. Y. Mehdad and J. Tetreault. Do characters abuse more than words? In *Proceedings of the SIGDIAL 2016 Conference*, pages 299–303, Los Angeles, USA, 2016. Association for Computational Linguistics.
17. C. Nobata, J. Tetreault, A. Thomas, Y. Mehdad, and Y. Chang. Abusive language detection in online user content. In *Proceedings of the 25th International Conference on World Wide Web*, pages 145–153, 2016.
18. A. Okeowo. Hate on the rise after trump‘s election, Last accessed: July 2017, http://www.newyorker.com/.
19. F. J. Ordóñez and D. Roggen. Deep convolutional and lstm recurrent neural networks for multimodal wearable activity recognition. *Sensors*, 16(1), 2016.
20. J. H. Park and P. Fung. One-step and two-step classification for abusive language detection on twitter. In *ALW1: 1st Workshop on Abusive Language Online*, Vancouver, Canada, 2017. Association for Computational Linguistics.
21. A. Schmidt and M. Wiegand. A survey on hate speech detection using natural language processing. In *International Workshop on Natural Language Processing for Social Media*, pages 1–10. Association for Computational Linguistics, 2017.
22. E. Tsironi, P. Barros, C. Weber, and S. Wermter. An analysis of convolutional long short-term memory recurrent neural networks for gesture recognition. *Neurocomput.*, 268(C):76–86, Dec. 2017.
23. F. D. Vigna, A. Cimino, F. Dell'Orletta, M. Petrocchi, and M. Tesconi. Hate me, hate me not: Hate speech detection on facebook. In *Proceedings of the First Italian Conference on Cybersecurity*, pages 86–95, 2017.
24. W. Warner and J. Hirschberg. Detecting hate speech on the world wide web. In *Proceedings of the Second Workshop on Language in Social Media*, LSM '12, pages 19–26. Association for Computational Linguistics, 2012.
25. Z. Waseem. Are you a racist or am i seeing things? annotator influence on hate speech detection on twitter. In *Proc. of the Workshop on NLP and Computational Social Science*, pages 138–142. Association for Computational Linguistics, 2016.
26. Z. Waseem and D. Hovy. Hateful symbols or hateful people? predictive features for hate speech detection on twitter. In *Proceedings of the NAACL Student Research Workshop*, pages 88–93. Association for Computational Linguistics, 2016.
27. G. Xiang, B. Fan, L. Wang, J. Hong, and C. Rose. Detecting offensive tweets via topical feature discovery over a large scale twitter corpus. In *Conference on Information and Knowledge Management*, pages 1980–1984. ACM, 2012.
28. H. Zhong, H. Li, A. Squicciarini, S. Rajtmajer, C. Griffin, D. Miller, and C. Caragea. Content-driven detection of cyberbullying on the instagram social network. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, IJCAI'16, pages 3952–3958. AAAI Press, 2016.
29. H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society. Series B (Statistical Methodology)*, 67(2):301–320, 2005.