# Cross-Domain Detection of Abusive Language Online

**Mladen Karan** and **Jan Šnajder**
Text Analysis and Knowledge Engineering Lab
Faculty of Electrical Engineering and Computing, University of Zagreb
Unska 3, 10000 Zagreb, Croatia
{mladen.karan,jan.snajder}@fer.hr

## Abstract

We investigate to what extent the models trained to detect general abusive language generalize between different datasets labeled with different abusive language types. To this end, we compare the cross-domain performance of simple classification models on nine different datasets, finding that the models fail to generalize to out-domain datasets and that having at least some in-domain data is important. We also show that using the frustratingly simple domain adaptation (Daume III, 2007) in most cases improves the results over in-domain training, especially when used to augment a smaller dataset with a larger one.

## 1 Introduction

Abusive language online (Waseem et al., 2017) is an increasing problem in modern society. Although abusive language is undoubtedly not a new phenomenon in human communication, the rise of the internet has made it concerningly prevalent. The main reason behind this is the cloak of relative anonymity offered when commenting online, which lowers the inhibitions of individuals prone to abusive language and removes some of the social mechanisms present in real life that serve to protect potential victims. Moreover, this type of psychological violence can occur at any time and regardless of the physical distance between the persons involved. While abusive language online can probably never be weeded out entirely, its effect can certainly be lessened by locating abusive posts and removing them before they cause too much harm. Training supervised machine learning models to recognize abusive texts and alert human moderators can make this process much more efficient. However, retaining humans in the loop is crucial, since blindly relying on model predictions would in effect turn every false positive prediction into infringement of free speech. This would defeat the initial purpose of using machine learning models to facilitate a free and civilized online discussions.

Detecting abusive language online is a subject of much ongoing research in the NLP community. Different studies have zeroed in on different types of abusive language (e.g., aggressive language, toxic language, hate speech) and have yielded a number of different datasets collected from various domains (e.g., news, Twitter, Wikipedia). However, from a practical perspective – if one simply wishes to build a classifier for detecting general abusive language in a given domain – the question arises as to which of these datasets to use for training. More generally, the question is to what extent abusive language detection transfers across domains, and how much, if anything, can be gained from a simple domain adaptation technique that combines the source and the target domain.

This paper investigates the question to what extent abusive language detection can benefit from combining training sets and sharing information between them through domain adaptation techniques. Our contribution is twofold. First, we compare the cross-domain performance of simple classification models on nine different English datasets of abusive language. Second, we explore whether the framework of frustratingly simple domain adaptation (FEDA) (Daume III, 2007) can be applied to improve classifier performance, in particular for smaller data sets. In addition, we show how a simple post-hoc feature analysis can reveal which features are specific to a certain domain and which are shared between two domains. We make our code and links to the used datasets available online.[1]

## 2 Related Work

A bewildering plethora of different types of abusive language can be found online. Some of the

[1] http://takelab.fer.hr/alfeda



*Proceedings of the Second Workshop on Abusive Language Online (ALW2)*, pages 132–137
Brussels, Belgium, October 31, 2018. 

types dealt with in related work include but are not limited to sexism, racism (Waseem and Hovy, 2016; Waseem, 2016), toxicity (Kolhatkar et al., 2018), hatefulness (Gao and Huang, 2017), aggression (Kumar et al., 2018), attack (Wulczyn et al., 2017), obscenity, threats, and insults. A typology of abusive language detection subtasks was recently proposed by Waseem et al. (2017).

Traditional machine learning approaches to detecting abusive language include the naive Bayes classifier (Kwok and Wang, 2013; Chen et al., 2012; Dinakar et al., 2011), logistic regression (Waseem and Hovy, 2016; Davidson et al., 2017; Wulczyn et al., 2017; Burnap and Williams, 2015), and support vector machines (SVM) (Xu et al., 2012; Dadvar et al., 2013; Schofield and Davidson, 2017). The best performance is most often attained by deep learning models, the most popular being convolutional neural networks (Gambäck and Sikdar, 2017; Potapova and Gordeev, 2016; Pavlopoulos et al., 2017) and variants of recurrent neural networks (Pavlopoulos et al., 2017; Gao and Huang, 2017; Pitsilis et al., 2018; Zhang et al., 2018). Some approaches (Badjatiya et al., 2017; Park and Fung, 2017; Mehdad and Tetreault, 2016) also rely on combining different types of models.

In this paper we explore combining different datasets from different domains to improve model performance. This idea is well established in the machine learning community under the name of transfer learning; we refer to (Weiss et al., 2016; Lu et al., 2015) for overviews. The work closest to ours is (Waseem et al., 2018), where multi-task learning is used to build robust hate-speech detection models. Our approach is very similar, but we consider more datasets and use a simpler, more easily interpretable transfer learning scheme.

## 3 Datasets

For our study we use nine publicly available datasets in English; Table 1 summarizes their main characteristics. For reasons of efficiency and comparability, we use a fixed split on each of the datasets into a train, development, and test portions. We respected the official splits where they were provided. As we are interested in detecting the presence of general abusive language, rather than in discerning among its many subtypes, we binarize the labels on all datasets into *positive* (abusive language) and *negative* (not abusive language). We do this by labeling all classes typeset in bold in Table 1 as *positive* and all other classes as *negative*. There are two exceptions to this rule. First, on the Kol dataset, we consider as *positive* those examples for which at least one annotator gave a rating higher than 1. Second, on the Kaggle dataset, which uses a multilabeling scheme, we consider as *positive* all instances annotated with at least one of the six harmful labels, and as *negative* all instances without any labels. We perform only the very basic preprocessing by lowercasing all words and lemmatizing them using NTLK (Loper and Bird, 2002).

While these modifications to original datasets make a comparison to previous work difficult, they allow a direct comparison across the datasets and a straightforward application of FEDA.

[2]Available at https://tinyurl.com/y7qmd8lm

## 4 Exp. 1: Cross-Domain Performance

The goal of this experiment is to asses how well the models trained on a particular dataset of abusive language perform on a different dataset. The differences in performance can be traced back to two factors: (1) the difference in the types of abusive language that the dataset was labeled with and (2) the differences in dataset sizes. In this work we observe the joint effect of both factors.

### 4.1 Experimental Setup

We use a linear Support Vector Machine (SVM), which has already been successfully applied to the task of abusive text classification (Schofield and Davidson, 2017). The main motivation for using an SVM, rather than more complex deep learning models, is that in this study we favor model interpretability, even if this means sacrificing performance.[3] Having interpretable models makes it easier to identify the biases that the models might have learned from data and how domain adaptation affects such biases. While, from a practical perspective, we might want to retain those biases for the sake of improving performance, it is important that we are aware that they exist, and thus have the option to correct them if necessary.

For the same reason, we rely on the most simple text representation with unigram counts, which makes it possible to directly correlate word salience to feature weights obtained from the SVM.

[3]We acknowledge that there are other competitive and yet interpretable models, such as deep neural networks with attention mechanisms. We leave the investigation of such models for future work.

| | Source | Train | Dev | Test | BR | Labels |
|---|---|---|---|---|---|---|
| Kol (Kolhatkar et al., 2018) | Newspaper | 730 | 104 | 209 | .627 | toxicity rating: 1,**2**,**3**,**4** |
| Gao (Gao and Huang, 2017) | Fox news | 1069 | 153 | 306 | .284 | non-hateful, **hateful** |
| TRAC (Kumar et al., 2018) | Twitter | 11999 | 3001 | 916 | .566 | non-aggressive, **covertly-aggressive**, **openly-aggressive** |
| Was2 (Waseem, 2016) | Twitter | 4836 | 691 | 1382 | .153 | none, **racism**, **sexism**, **both** |
| Was1 (Waseem and Hovy, 2016) | Twitter | 11835 | 1691 | 3381 | .319 | none, **racism**, **sexism** |
| Wul1 (Wulczyn et al., 2017) | Wikipedia | 69526 | 23160 | 23178 | .146 | non-aggressive, **aggressive** |
| Wul2 (Wulczyn et al., 2017) | Wikipedia | 69526 | 23160 | 23178 | .134 | non-attack, **attack** |
| Wul3 (Wulczyn et al., 2017) | Wikipedia | 95692 | 32128 | 31866 | .115 | non-toxic, **toxic** |
| Kaggle [2] | Wikipedia | 143614 | 15957 | 63978 | .101 | **toxic**, **severe_toxic**, **obscene**, **threat**, **insult**, **identity_hate** |

Table 1: Nine abusive language datasets: the source, the number of instances in the train, development, and test set, positive instance base rate (BR), and label sets. We mapped the boldface labels to the positive label.

When an SVM model trained on dataset $X$ is applied to dataset $Y$, we first train the model on training set $X$ optimizing the hyperparameter $C$ in the range $\{2^{-10}, ..., 2^{6}\}$ to maximize performance on the development set of $X$. We then train the SVM with the optimal hyperparameters on the union of training and development sets of $X$, and then use the model to label the test set of $Y$, obtaining the final score. We measure the performance using the standard two-class F1 score.

### 4.2 Results

Results are given in Table 2. The rows correspond to different training sets, while the columns correspond to different test sets. For each test set, the best performance is shown in bold. The diagonal cells correspond to the cases of in-domain model testing. For each model $X$ tested on each out-domain dataset $Y$ (off-diagonal cells), we test the statistical significance between that model's in-domain and out-domain performance using a two-tailed bootstrap resampling test at $\alpha = 0.05$.

Expectedly, most models perform best on the in-domain test sets. Exceptions are the Wikipedia-based data sets, where the model trained on Kaggle performs the best on all test sets. This can be attributed to the an overlap that exists between these data sets: Wul1 and Wul2 contain almost identical texts, Wul3 has 68% overlap with them and Kaggle has 1.5% and 3% overlap with Wul1/Wul2 and Wul3, respectively. We mark in gray the corresponding portion of the tables, and refrain from drawing any conclusions from this data.

Another observation is that the performance on out-domain data sets is considerably lower. When applying models to a different test set the performance often drops by more than 50% of F1 score, which indicates that the models do not generalize well to different datasets. In cases when the size of $X$ is small compared to the size of $Y$, the training portion of $X$ will also be smaller than the training portion of $Y$, and it could be argued that the drop in performance is simply due to the model having less training data. However, considerable performance drops are also observable when going from a large $X$ to a small $Y$, which suggests that the gains from having more training instances in $X$ are counterbalanced by the domain differences between $X$ and $Y$, and the net result is a loss in performance. Our experiments thus show that having a smaller dataset for a particular domain of abusive language is better than having a very large dataset from a different one. In the following experiment we explore whether a large dataset from a different domain can still be leveraged in a different way.

## 5 Exp. 2: Domain Adaptation

### 5.1 Experimental Setup

We investigate the potential of applying domain adaptation to augment the original domain with the information from a different domain. To this end, we employ the FEDA framework (Daume III, 2007), which works by copying features several times to account for different domains, allowing the model to learn domain-dependent weights for each feature.

Let the dataset from the original domain be denoted as $O$ and the data set from an augmentation domain as $A$. We generate a joint train set as a union of train sets of $O$ and $A$ by keeping three copies of each feature: (1) a general copy, which is unaltered for instances from both domains, (2) an $O$-specific copy, which is set to 0 for all instances not from $O$, and (3) an $A$-specific copy, which is set to 0 for all instances not from $A$. In the same

| | Kol | Gao | TRAC | Was2 | Was1 | Wul1 | Wul2 | Wul3 | Kaggle |
|---|---|---|---|---|---|---|---|---|---|
| Kol | **0.816** | 0.423 | 0.475* | 0.280* | 0.479* | 0.251* | 0.233* | 0.204* | 0.178* |
| Gao | 0.362* | **0.493** | 0.249* | 0.181* | 0.399* | 0.184* | 0.177* | 0.168* | 0.139* |
| TRAC | 0.697* | 0.421 | **0.548** | 0.283* | 0.484* | 0.307* | 0.288* | 0.259* | 0.225* |
| Was2 | 0.088* | 0.023* | 0.091* | **0.680** | 0.174* | 0.108* | 0.109* | 0.098* | 0.094* |
| Was1 | 0.432* | 0.348* | 0.238* | 0.421* | **0.739** | 0.236* | 0.229* | 0.208* | 0.186* |
| Wul1 | 0.092* | 0.118* | 0.316* | 0.190* | 0.236* | 0.701 | 0.718 | 0.746 | 0.602* |
| Wul2 | 0.115* | 0.078* | 0.318* | 0.248* | 0.226* | 0.694 | 0.710 | 0.757 | 0.613* |
| Wul3 | 0.139* | 0.159* | 0.320* | 0.263* | 0.316* | 0.773* | 0.784* | 0.753 | 0.626* |
| Kaggle | 0.054* | 0.132* | 0.296* | 0.249* | 0.300* | **0.782*** | **0.800*** | **0.860*** | **0.640** |

Table 2: Results of cross-domain model performance. Rows are the training datasets and columns are the test datasets. The best performance for each test set (column) is shown in bold. "*" indicates statistical significance at significance level $\alpha = 0.05$ with respect to the diagonal cell.

| | None | Kol | Gao | TRAC | Was2 | Was1 | Wul1 | Wul2 | Wul3 | Kaggle |
|---|---|---|---|---|---|---|---|---|---|---|
| Kol | **0.816** | – | 0.654* | 0.775* | 0.605* | 0.615* | 0.627* | 0.651* | 0.622* | 0.605* |
| Gao | 0.493 | 0.500 | – | 0.460 | **0.534** | 0.507 | 0.441 | 0.415 | 0.463 | 0.455 |
| TRAC | 0.548 | 0.548 | 0.554 | – | 0.567 | 0.568 | **0.575** | 0.573 | 0.557 | 0.565 |
| Was2 | 0.680 | 0.703 | 0.661 | **0.730*** | – | 0.711 | 0.706 | 0.714* | 0.715* | 0.724* |
| Was1 | 0.739 | 0.744 | 0.743 | **0.755*** | 0.743 | – | 0.749 | 0.747 | 0.749 | 0.752* |
| Wul1 | 0.701 | 0.701 | 0.699 | 0.708* | 0.701 | 0.699 | – | 0.701 | **0.717*** | **0.717*** |
| Wul2 | 0.710 | 0.709 | 0.709 | 0.719* | 0.710 | 0.715 | 0.716* | – | 0.734* | **0.736*** |
| Wul3 | 0.753 | 0.753 | 0.753 | 0.758* | 0.752 | 0.754 | 0.764* | 0.763* | – | **0.788*** |
| Kaggle | 0.640 | 0.640 | 0.640 | **0.643*** | 0.639 | 0.639 | 0.640 | 0.640 | 0.638 | – |

Table 3: FEDA domain adaptation results. Rows correspond to original datasets and columns to augmentation datasets. The best performance for each original dataset (row) is shown in bold. "*" indicates statistical significance at significance level $\alpha = 0.05$ against the "None" column, which is equivalent to the diagonal of Table 2.

way we generate joint development and test sets. The intuition behind why this effectively leads to domain adaptation is that it allows the underlying machine learning model to differentiate features (words) that are generally useful from those that are useful in only one of the domains. Consequently, it can better learn the similarities and differences of the domains and how to exploit them to maximize performance. For example, a word such as *moron* is almost universally abusive in all domains and would generalize well. On the other hand, a word like *fruit* is almost always completely non-abusive except in specific domains where it might denote a derogatory slang for a homosexual person.

As before, the SVM is trained on the joint training set, with model selection on the joint development set. The model is then trained using optimal hyperparameters on the union of joint training and joint development set and applied to the joint test set. Note that the joint test set contains test instances from both $O$ and $A$. We evaluate the model only on the test instances from $O$, as the goal is to determine whether augmentation with $A$ improves performance on the dataset from the original domain $O$.

### 5.2 Results

Results are given in Table 3. Each row represents an original domain dataset and each column an augmentation domain dataset. The "None" column corresponds to the results obtained using no augmentation. We use two-tailed bootstrap resampling with $\alpha = 0.05$ to test the statistical significance of each result to the one on the same original dataset without augmentation. The main observation is that for most datasets FEDA leads to performance improvements, and for six out of nine datasets there is at least one augmentation dataset which gives a statistically significant performance improvement. For the five smallest datasets, (Kol, Gao, TRAC, Was1, and Was2) domain adaptation improves the performance on four, and for two the improvements are statistically significant. These results indicate that domain adaptation has the potential to improve results on smaller datasets. Augmenting Wul1, Wul2, and Wul3 with Kaggle yields considerable improvements, which again can be attributed to the overlap between these datasets. An exception is Kol, on which models do not benefit from FEDA. The possible reasons for this might be its small size or high base rate.

| No FEDA | General | Was2 | TRAC |
|---|---|---|---|
| anti_feminazi_movement | motherfucker | feminazi | fuck |
| feminazi | cocksucker | west | idiot |
| feminazi_front | dickhead | howtospotafeminist | asshole |
| models | douchebag | feminazi_front | ass |
| howtospotafeminist | cunt | blondes | bitch |
| prowomanchoice | assholes | anti_feminazi_movement | motherfucker |
| raging | fuckers | coon | cocksucker |
| blondemoment | fuckhead | killerblondes | dickhead |
| prove | feminazi | asian | shit |
| adorable | coward | hold | douchebag |

Table 4: Top 10 features by SVM weights for the Was2 data set without FEDA and with FEDA using TRAC as the augmentation dataset and three feature variants (General, Was2-specific, and TRAC-specific)

### 5.3 Feature Analysis

FEDA offers a convenient way to analyze which features are generic and signal abusive language in both domains, and which are specific to each. The former features will have high merit for their general copies, while the latter will have high merit for domain-specific copies. In Table 4 we list the top 10 features for the case where we observed the highest improvement: Was2 as the original and TRAC as the augmentation dataset. The results show that the model does indeed learn to differentiate between the sexism/racism domain of Was2 and the aggression focused domain of TRAC, while also learning the general features useful on both datasets.

When not using FEDA, the most indicative features are, expectedly, focused mostly on the sexism/racism aspects of the Was2 dataset. However, when introducing the augmentation domain TRAC dataset, which focuses on aggressive/non-aggressive texts, the features discern between different aspects of abusive language. Words in the *General* column of Table 4 are indeed generally abusive words and can be viewed as indicative of the abusive class for both datasets. On the other hand, the domain-specific features reflect the specific properties of each dataset. For the Was2 dataset these include words correlated with sexism or racism (but not useful for aggression detection on TRAC) such as *feminazi*. On the TRAC dataset domain-specific features are words that are indicative of aggression (but not of sexism/racism in the Was2 dataset), such as *shit*.

## 6 Conclusion

We compared the performance of abusive language classifiers across datasets from different sources and types of abusive language. We found that the models considered do not generalize well to different-domain datasets, even when trained on a much larger out-domain data. This indicates that having in-domain data, even if not much of it, is crucial for achieving good performance on this task. Furthermore, the experiments have shown that frustratingly simple domain adaptation (FEDA) in most cases improves the results over in-domain training, especially when smaller datasets are augmented with a larger datasets from a different domain.

We found FEDA to be a useful tool to compare the differences between various domains of abusive language and believe that related techniques might lead to new interesting insights into the phenomenon of abusive language.

## References


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

Pete Burnap and Matthew L Williams. 2015. Cyber hate speech on Twitter: An application of machine classification and statistical modeling for policy and decision making. *Policy & Internet*, 7(2):223–242.

Ying Chen, Yilu Zhou, Sencun Zhu, and Heng Xu. 2012. Detecting offensive language in social media to protect adolescent online safety. In *Privacy, Security, Risk and Trust (PASSAT), 2012 International Conference on and 2012 International Confernece on Social Computing (SocialCom)*, pages 71–80. IEEE.

Maral Dadvar, Dolf Trieschnigg, Roeland Ordelman, and Franciska de Jong. 2013. Improving cyberbullying detection with user context. In *Advances in Information Retrieval*, pages 693–696. Springer.

Hal Daume III. 2007. Frustratingly easy domain adaptation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 256–263.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated Hate Speech Detection and the Problem of Offensive Language. In *Proceedings of ICWSM*.

Karthik Dinakar, Roi Reichart, and Henry Lieberman. 2011. Modeling the detection of textual cyberbullying. In *The Social Mobile Web*, pages 11–17.

Björn Gambäck and Utpal Kumar Sikdar. 2017. Using Convolutional Neural Networks to Classify Hate-speech. In *Proceedings of the First Workshop on Abusive Language Online*, pages 85–90.

Lei Gao and Ruihong Huang. 2017. Detecting online hate speech using context aware models. In *Proceedings of Recent Advances in Natural Language Processing (RANLP)*, pages 260–266, Varna, Bulgaria.

Varada Kolhatkar, Hanhan Wu, Luca Cavasso, Emilie Francis, Kavan Shukla, and Maite Taboada. 2018. The SFU opinion and comments corpus: A corpus for the analysis of online news comments.

Ritesh Kumar, Aishwarya N. Reganti, Akshit Bhatia, and Tushar Maheshwari. 2018. Aggression-annotated corpus of Hindi-English code-mixed data. In *Proceedings of the 11th Language Resources and Evaluation Conference (LREC)*, pages 1425–1431, Miyazaki, Japan.

Irene Kwok and Yuzhou Wang. 2013. Locate the hate: Detecting tweets against blacks. In *Twenty-Seventh AAAI Conference on Artificial Intelligence*.

Edward Loper and Steven Bird. 2002. NLTK: The natural language toolkit. In *Proceedings of the ACL-02 Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics - Volume 1*, ETMTNLP '02, pages 63–70, Stroudsburg, PA, USA. Association for Computational Linguistics.

Jie Lu, Vahid Behbood, Peng Hao, Hua Zuo, Shan Xue, and Guangquan Zhang. 2015. Transfer learning using computational intelligence: a survey. *Knowledge-Based Systems*, 80:14–23.

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*, pages 299–303.

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

John Pavlopoulos, Prodromos Malakasiotis, and Ion Androutsopoulos. 2017. Deep learning for user comment moderation. In *Proceedings of the First Workshop on Abusive Language Online*, pages 25–35.

Georgios K Pitsilis, Heri Ramampiaro, and Helge Langseth. 2018. Detecting offensive language in tweets using deep learning. *arXiv preprint arXiv:1801.04433*.

Rodmonga Potapova and Denis Gordeev. 2016. Detecting state of aggression in sentences using CNN. *arXiv preprint arXiv:1604.06650*.

Alexandra Schofield and Thomas Davidson. 2017. Identifying hate speech in social media. *XRDS: Crossroads, The ACM Magazine for Students*, 24(2):56–59.

Zeerak Waseem. 2016. Are you a racist or am I seeing things? Annotator influence on hate speech detection on Twitter. In *Proceedings of the first workshop on NLP and computational social science*, pages 138–142, Austin, Texas.

Zeerak Waseem, Thomas Davidson, Dana Warmsley, and Ingmar Weber. 2017. Understanding Abuse: A Typology of Abusive Language Detection Subtasks. In *Proceedings of the First Workshop on Abusive Langauge Online*.

Zeerak Waseem and Dirk Hovy. 2016. Hateful symbols or hateful people? Predictive features for hate speech detection on Twitter. In *Proceedings of the NAACL student research workshop*, pages 88–93, San Diego, California.

Zeerak Waseem, James Thorne, and Joachim Bingel. 2018. Bridging the gaps: Multi task learning for domain transfer of hate speech detection. In *Online Harassment*, pages 29–55. Springer.

Karl Weiss, Taghi M Khoshgoftaar, and DingDing Wang. 2016. A survey of transfer learning. *Journal of Big Data*, 3(1):9.

Ellery Wulczyn, Nithum Thain, and Lucas Dixon. 2017. Ex machina: Personal attacks seen at scale. In *Proceedings of the 26th International Conference on World Wide Web*, pages 1391–1399. International World Wide Web Conferences Steering Committee.

Jun-Ming Xu, Kwang-Sung Jun, Xiaojin Zhu, and Amy Bellmore. 2012. Learning from bullying traces in social media. In *Proceedings of the 2012 conference of the North American chapter of the association for computational linguistics: Human language technologies*, pages 656–666. Association for Computational Linguistics.

Ziqi Zhang, David Robinson, and Jonathan Tepper. 2018. Detecting hate speech on Twitter using a convolution-GRU based deep neural network. In *Lecture Notes in Computer Science*. Springer Verlag.