NLP_UIOWA at Semeval-2021 Task 5: Transferring Toxic Sets to Tag Toxic Spans

Jonathan Rusert
Department of Computer Science
University of Iowa
Iowa City, IA, USA
jonathan-rusert@uiowa.edu

Abstract

We leverage a BLSTM with attention to identify toxic spans in texts. We explore different dimensions which affect the model's performance. The first dimension explored is the dataset the model is trained on. Besides the provided dataset, we explore the transferability of 5 different toxic related sets, including offensive, toxic, abusive, and hate sets. We find that the solely offensive set shows the highest promise of transferability. The second dimension we explore is methodology, including leveraging attention, employing a greedy remove method, using a frequency ratio, and examining hybrid combinations of multiple methods. We conduct an error analysis to examine which types of toxic spans were missed and which were wrongly inferred as toxic along with the main reasons why they occurred. Finally, we extend our method via ensembles, which achieves our highest F1 score of 55.1.

1 Introduction

Social media platforms (e.g. Twitter) benefit from a welcoming environment. Toxicity introduced by users can drive new (or existing) users away. Therefore, it is beneficial for platforms to moderate the amount of toxic language allowed. Often human moderators are employed to address this. However, as the numbers of users increase on a platform, it can be difficult for human moderation as the sheer amount of content generated increases as well. Thus platforms employ automated assistance (Terdiman, 2018).

While automated toxic language classifiers have made strides at determining if text is toxic (Yenala et al., 2018; Zhang et al., 2018), much work is still left for humans who are reviewing the text. In large texts, the reason for toxicity might only be one or two words, however, the moderator will have to scan the entire text to find the toxicity. Repeating this process across many texts is cumbersome. Automated systems could assist human moderators further by noting which areas in the text are toxic, thus allowing the human to quickly check text for toxicity. Finding these toxic areas is the task that is proposed by (Pavlopoulos et al., 2021). Specifically, given English toxic text, determine where the "toxic spans" occur, where a "toxic span" is the character positions in the text where the toxicity appears. For example, in "your an idiot, this is a tax based on a lie", the toxic span would be [8, 9, 10, 11, 12] or the character positions of "idiot". Note that not all toxic texts have toxic spans. For example, "Religious people no longer have a place in this country." is a toxic phrase, yet no specific words are themselves toxic. More examples can be found in Table 1.

Text	Toxic Span
What a jerk!	[7, 8, 9, 10]
"Trump is up to the task" LMAO, you are an idiot	[44, 45, 46, 47, 48]
Canada: Institutionalised mediocrity.	[ ]
Contrary to popular belief not all criminals are stupid. Fortunately, there are plenty like this moron that are.	[49, 50, 51, 52, 53, 54, 98, 99, 100, 101, 102]

Table 1: Examples of toxic spans in toxic texts. The numbers indicate the character position of the toxic span. Empty brackets indicate no specific span.

To find toxic spans in text, we employ a Bidirectional Long Short-term Memory model (BLSTM) with attention. We examine several indirect methods which align with the BLSTM, including using the attention layer for information. While the provided dataset allows for direct methods, i.e. methods which directly use the given spans for training, this span format is new to this task. Thus many previously developed toxic datasets would need to be converted to be useful to a direct model (since they are binary labeled sets, not span labeled sets). However, the indirect methods we explore can leverage previously developed datasets, which opens up research possibilities. Therefore, we examine how well toxic texts other than the provided training data transfer to this task. An additional benefit is that since toxicity can be subjective (Aroyo et al., 2019), examining how different datasets transfer to this task will help illuminate the similarities and differences between definitions of toxicity. Finally, we provide our code and outputs for reproducibility[1].

2 Approach

We expand on the approach we use to identify toxic spans. We examine both different toxic based datasets as well as different BLSTM based methods.

2.1 Base Architecture

We leverage a bidirectional LSTM (BLSTM) with attention as our base architecture. The BLSTM has a hidden layer size of 200. We use glove embeddings[2] (Pennington et al., 2014) of size 200 trainined on Wikipedia and Gigaword corpora. We train our system over 10 epochs with a batch size of 64.

2.2 Toxic Datasets

We examine the provided training set as well 5 different toxic-based data sets. All are in English to match the provided test set. Each additional dataset is explicitly toxic or in a related area (e.g. offense). The dataset chosen can affect performance on the test data as data can vary in their definitions of toxicity. The datasets examined are:

Toxic Train: The training set provided by the task organizers. This set had no non-toxic examples thus we pulled non-toxic examples from the Kaggle Toxic set at a 1-to-1 ratio. This results in a training set size of 15,878.

OLID: (Zampieri et al., 2019a) A dataset composed of offensive and non-offensive tweets. It was used in both OffensEval 2019 (Zampieri et al., 2019b) and OffensEval 2020 (Zampieri et al., 2020) (offense language classification tasks). The training set is composed of 13,240 tweets.

Kaggle Toxic: Comments from Wikipedia's talk page edits, presented in the Kaggle toxic classification challenge[3], labeled as toxic or not toxic (as well as further dividied into subcategories). The training set is 159,571 comments.

Founta: (Founta et al., 2018) A dataset of tweets annotated via the CrowdFlower platform. The labels are hateful, abusive, or none. For training, we combine hateful and abusive into a single toxic category. The training set is of size 85,966.

Davidson: (Davidson et al., 2017) Annotated tweet dataset composed of offensive, hatespeech, and neither tweets. Our training set combines offense and hate into a single toxic label and is of size 24,783.

Gab Hate: (Kennedy et al., 2018) A dataset from Gab social network[4]. Each post is annotated by at least 3 annotators as either hateful or not. Similar to prior research, our positive (toxic) labeled are those for which the majority of annotators agree. The training set is of size 27,665.

2.3 Methods

We leverage previously examined methods in areas such as robustness (Hsieh et al., 2019) and style transfer (Xu et al., 2018; Wu et al., 2019). Each method stems from the BLSTM, but some can be adapted for other machine learning systems.

Attention: Previous work has used attention for robustness attacks (Hsieh et al., 2019) and style token masking (Xu et al., 2018). Following these, we leverage the attention layer in the BLSTM. Each token (word) has an attention score used to generate the class label (toxic, non-toxic). We use this score to determine if a given token is toxic. We label each token with an attention score of greater than the average attention score as part of the toxic span.

Frequency Ratio: In theory, a toxic token should exist more frequency in toxic texts, compared to non-toxic texts. Following previous research (Wu et al., 2019), we generate a frequency score for each token:

$$s_c(u,a) = \frac{count(u, D_a) + \lambda}{(\sum_{a' \in A, a' \neq a} count(u, D_{a'})) + \lambda} \tag{1}$$

where u is the token, a is an attribute (e.g. toxic or non-toxic), D_a represents the texts with attribute a,

[1]Code: https://github.com/JonRusert/semeval-2021-task-5

[2]https://nlp.stanford.edu/projects/glove/

[3]https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge

[4]https://gab.com/

		Toxic Datasets						
		Train	OLID	Kaggle-Toxic	Founta	Gab	Davidson	Average
Methods	Attention	50.1	50.1*	39.4	35.2	17.4	22.8	35.8
	Greedy Remove	38.3	44.7	43.3	41.9	20.8	18.6	34.6
	Frequency Ratio	41.5	45.5	22.5	44.3	19.7	8.0	30.3
	Simple Hybrid	38.4	45.5	41.7	39.6	17.7	21.7	34.1
	Recall Hybrid	37.9	45.2	38.4	34.6	17.8	22.7	32.8
	Precision Hybrid	**50.7**	49.8	44.6	42.6	20.8	19.0	**37.9**
	Average	42.8	**46.8**	38.3	39.7	19.0	18.8	
	Top System	70.8						

Table 2: Full Results on the provided Toxic Span Test Set. Results are F1 scores. * - Version submitted to task organizers. Top System is the highest result submitted by another team to the task organizers.

and $count(u, D_a)$ represents the number of times u appears in D_a. The score ($s_c(u, a)$) is then measured against a threshold to determine whether the token is representative of that attribute. For our method, we chose a threshold of 5, such that, if $s_c(u, a) >= 5$, then u is considered part of the toxic span.

Greedy Remove: Also seen as "Greedy select" (Hsieh et al., 2019) or "Important Score" (Lee, 2020) in previous research, greedy remove determines the amount each token is contributing to the text by removing each token one-by-one (each token is replaced after drop is noted) and noting the drop in toxic probability. The token with the highest probability is removed (i.e. added to the toxic span) and then the BLSTM classification is checked. If the text is no longer considered toxic, the toxic span is returned, otherwise the next token with the highest drop is removed and it is added to the toxic span. This process is continued until the text is no longer considered toxic by the BLSTM.

Simple Hybrid: We combine the greedy remove and attention approaches. We first obtain the spans from the greedy remove. If the span is empty (i.e. "[]"), then we choose the spans found by the attention method, otherwise we choose the spans obtained from the greedy remove method.

Recall Hybrid: Similar to the simple hybrid, however, we aim to maximize recall. We obtain spans from both the greedy remove and attention approaches and return the union of the two sets.

Precision Hybrid: Similar to the simple hybrid, however, we aim to maximize precision. We obtain spans from both the greedy remove and attention approaches and return the intersection of the two sets.

3 Experimental Results

3.1 Experiments

We train corresponding BLSTMs on each of the 6 presented datasets (Section 2.2). We test the methods of each BLSTM on the provided toxic test set. The test set contains 3,180 texts. The score used is the F1 for a given text t, where A_i is the system and G is ground truth, and S^t is the set of toxic spans:

$$F_1^t(A_i, G) = \frac{2 \cdot P^t(A_i, G) \cdot R^t(A_i, G)}{P^t(A_i, G) + R^t(A_i, G)} \quad (2)$$

$$P^t(A_i, G) = \frac{|S^t_{A_i} \cap S^t_G|}{|S^t_{A_i}|} \quad (3)$$

$$R^t(A_i, G) = \frac{|S^t_{A_i} \cap S^t_G|}{|S^t_G|} \quad (4)$$

When S^t_G is empty and $S^t_{A_i}$ is empty, then $F_1^t(A_i, G) = 1$, otherwise $F_1^t(A_i, G) = 0$. Finally, to obtain a single score, the F1's are averaged across texts.

3.2 Results

We examine the performance of various combinations of approaches on the test data. Full results are found in Table 2.

Provided Train Data + Precision Hybrid Approach achieves highest F1. Our highest achieved F1 score, on the test data, is 50.7. This is achieved via a BLSTM trained on the provided training data (Toxic Train) which uses the precision hybrid approach to find toxic spans. The next highest was from both the OLID BLSTM and Toxic Train BLSTM both using attention and achieving an F1 of 50.1.

Missed Spans
See a shrink you *pathetic* <troll>.
<Total> *rubbish*! The birther bit was started by Crooked Hillary and perpetuated by Zero his own damn self.
and he is to *stupid* to know it. The guy is a total <poser> and a *fool*
Along with 'fat, dumb and happy'. Or maybe '<stupid> is as *stupid* does'?
<Your an> *idiot*. The top 1% will get the biggest break while the rest of us struggle to pay rent.
In this case, anyway, I have bad news for you. <URStupid>
Sick <f**ks> like this should just be shot. What a waste of skin, air, and taxpayers money.
Falsely Detected Spans
Yes....it is to be expected....oddly enough it happens every time he opens his mouth and says <dumb> >thin<gs.
””””>Trump< is up to the task”” LMAO, you are an id*iot*>,”<
You are a *simpleton*, who is full of racial >hatred<. That is too bad. You make America a worse place.
I always knew you were a >dirty< cop.
By the way, Nazis were and are fascists so if you are one, your a *fascist*. >Scum< like this and Trumpler need to be run out of our country and given to ISIS to use as chew toys
McCain is a >pig<.
LNG is much better for the environment than diesel, yet these >stupid< hippies complain.

Table 3: Examples of missed spans and falsely detected spans. <Red> indicates missed tokens. *Blue* indicate correctly identified spans. >Green< indicates wrongly asserted spans. Note that **, <>, >< are not originally in text and are included for accessibility.

OLID shows strongest transferability. The OLID trained BLSTM achieved the highest F1 scores across all methods versus the other transfer sets, with an average of 46.8 (range: 44.7 - 50.1). The next highest set was Founta (avg: 39.7, range: 34.6 - 44.3). Surprisingly, Kaggle-Toxic (avg: 38.3, range: 22.5 - 44.6) arrives at third highest. This is surprising because the goal of the task is to identify "toxic" spans and Kaggle is toxic data, while OLID is offense, and Founta is abuse/hate data. The Gab and Davidson BLSTMs perform poorly in comparison only achieving average F1s of 19.0 and 18.8 respectively.

Precision Hybrid Averages Highest F1 Score. On average, the precision hybrid method outperforms other methods across datasets. The precision hybrid approach averages a F1 score of 35.4 (range: 19.0 - 49.8). The attention approach is second highest (avg: 35.8, range: 17.4 - 50.1), followed by the greedy remove (avg: 34.6, range: 18.6 - 44.7). This points to precision hybrid as a solid approach when the transfer strength is unknown.

Indirect methods pale in comparison to more direct systems. Although we examined various combinations of systems, our highest F1 (50.7) is still far less than the top submitted system (HITSZ-HLT) 70.8. This difference is most likely due to our system aiming to find toxic spans indirectly, rather than directly training on the spans.

4 Error Analysis

To determine how to improve our system, we examine the spans commonly missed in the test set as well as spans commonly predicted wrongly. We examine errors in our top system (Toxic Train - Precison Hybrid).

4.1 Spans Missed

We find our system misses spans for several reasons. We present examples in Table 3.

Translation from tokens to character positions. One reason for missed spans is the difficulty between translating toxic tokens to toxic spans. As our methodology finds specific tokens toxic, it needs to back-translate that to character positions in the original text. However, the text is pre-processed to help our system analyze it and therefore problems occur when lining up the characters.

Toxic words are part of a phrase. Another reason for missed spans is they are part of a toxic phrase and others part of the phrase are more toxic. For example, "Total rubbish" is marked as entirely toxic, however, our system marks only "rubbish".

Spans Missed			
Translation	Part of Phrase	Overshadowed by more toxic words	Non Standard Words
75	8	25	2
Spans Falsely Detected			
Non-Toxic becoming Toxic	Translation	Unclear Why Not Toxic	Toxic word in not toxic usage
19	23	22	30

Table 4: Frequencies of errors on 100 sampled texts. Note that multiple errors can exist in one text. Also, not all sampled texts have one of the prominent errors.

Another example is our system marking “pathetic” correctly but not “troll” in “pathetic troll”. Our system misses that these non-toxic words (e.g. “total”) can become toxic when combined in a phrase.

Certain words overshadowed by more toxic words. Some of the tokens which remain (e.g. “poser”) seem to be overshadowed by a stronger toxic word. For example, “and he is to stupid to know it. The guy is a total poser and a fool”, is marked as having three toxic spans (stupid, poser, fool), however, our system labels “stupid” and “fool” as toxic spans but not “poser”. This is most likely because our system views “stupid/fool” as more toxic words and thus assigns higher attention scores than for “poser”.

Toxic words are non standard words. Since the BLSTM uses word embeddings, non standard words are missed by our system. For example, “In this case, anyway, I have bad news for you. URStupid”, “URStupid” is marked as toxic, but it is a non standard word.

4.2 Spans Falsely Detected

Table 3 also contains examples of wrongly predicted spans by our system. We’ve identified reasons for these errors as well.

Non-toxic words become associated with toxicity in training set. Depending on the training set, our model can learn non-toxic words as toxic if they occur more frequently with the toxic label compared to the non-toxic. An interesting example is “trump” being marked as a toxic span, which means the training data provides “trump” in enough toxic texts for our system to associate that with toxicity.

Translation to character indexing. Similar to missing gold spans, our system’s predicted spans are often shifted off of the correct token. For example, in the phrase “dumb things”, our system correctly identifies that “dumb” is a toxic span, but during translation back to character index, it is shifted 4 characters to the right, so instead our system says “thin” (part of “things”) is the toxic span. This issue causes many similar misses.

Toxicity is not always clear. Another reason for wrongly identified spans is that the toxicity isn’t always clear. That is, words which appear to be toxic, are marked as non toxic in the gold labels. For example, in “I always new you were a dirty cop.”, calling someone a “dirty cop” is arguably toxic, and our system marks “dirty” as toxic, but the gold standard labels indicate there is no toxic span in the text. This could be attributed to toxicity being a subjective idea (Aroyo et al., 2019).

Toxic words used in non-toxic ways. A final prominent reason for falsely predicted spans is words that are sometimes toxic, appear in non toxic sentences. For example, in “... Gotta ask, you got back flow preventer on your plumbing or is that some of the stupid you can’t fix?”, our classifier tags “stupid” as toxic, but it is not. This category has an overlap as well with the previous unclear category since toxicity can be subjective. Where the annotators might label a word are being used in a non-toxic way, we might label it as toxic and therefore say it is unclear why the span is not counted correctly.

4.3 Quantifying Errors

To obtain an understanding of the frequency of the different types of errors we randomly sample 100 texts which contained missed spans and 100 texts which contained falsely predicted spans. The results can be found in Table 4.

For missed spans, we see the translation from tokens to character positions causing the largest problems, as it occurs in 75 out of 100 samples. Words overshadowed by more toxic words was the next highest error appearing in 25 of 100 samples.

For falsely detected spans, the errors are more evenly spread out. Toxic words used in non-toxic manners caused the most false predictions (30 out of 100), followed by the translation from tokens to spans (23 out of 100). These errors highlight the importance of the translation step, and where our system could benefit the most for improvement in the future.

5 Leveraging Multiple Views via Ensemble of Datasets

Ensemble	F1
Train + OLID + Kaggle-Toxic	54.0
Train + OLID + Kaggle-Toxic + Founta	49.6
Train + OLID + Founta	53.6
All	39.3
Train + Kaggle-Toxic + Founta	50.7
OLID + Kaggle-Toxic + Founta	49.5
Train + OLID-Att + Kaggle-Toxic	**55.1**
Top Solo (Toxic Train - Prec. Hybrid)	50.7

Table 5: F1 scores of ensemble models. All systems use the Precision Hybrid method, except for OLID-Att which uses the attention method. Train = Toxic Train

As toxicity can vary, it follows that different toxic datasets can identify different aspects of toxic spans. We explore this idea further by creating a simple majority voting ensemble model. The ensemble model takes in the predictions of multiple models trained on different datasets, then indices are included if a simple majority votes for the index. A summary of the performance of these models is found in Table 5. Due to its performance, the Precision Hybrid method is used in all instances except OLID-Att which leverages the attention method.

Ensemble outperforms top solo model. The top ensemble combination achieves an F1 score of 55.1, which is higher than the top solo F1 of 50.7. This ensemble consists of top scoring solo datasets and their top scoring respective methods: Toxic train - Precision Hybrid, OLID - Attention, and Kaggle-Toxic - Precision Hybrid. This higher score helps demonstrate how the various toxic-based datasets can bring useful perspectives to the solution.

Lower systems reduce performance of ensemble. While an ensemble can increase performance, not all toxic-based datasets help the model. While the top three solo datasets achieve an F1 score of 54, adding Founta reduces it to 49.6. Adding the lowest performing sets (Gab and Davidson), reduces F1 to 39.3. Thus the datasets used in the ensemble require much consideration.

6 Related Work

We provide a brief look at work related to ours and highlight recent related ideas.

Toxic Language: In recent years toxic language classification has seen much focus. Organized tasks for offensive language classification were seen multiple years in a row in GermEval (Wiegand et al., 2018), OffensEval 2019 (Zampieri et al., 2019b), and OffensEval 2020 (Zampieri et al., 2020). Additionally, hate and abuse have been the focus of classification (Founta et al., 2018) (Davidson et al., 2017). For toxicity specifically, Karan and Šnajder (2019) examine detecting threads that could lead to toxic language in the future. van Aken et al. (2018) present specific challenges that remain for toxic classification.

Methodology: Our work draws from both robustness and text style transfer work. Robustness work looks at attacks of adversaries against classification (and other) systems. Hsieh et al. (2019) look at the robustness of attention based systems. Their "greedy-select" method for selecting words to change in obfuscation, is similar to our "greedy-remove" method. Style transfer often focuses on transferring text from one style to another. To change the style, some authors first remove or mask words which indicate the initial style from the text. Wu et al. (2019) accomplish this by leveraging both a frequency ratio method (same as ours) and attention. Though, the methodology is similar, we examine it in terms of toxic span identification where it has not been done before.

7 Conclusion

We leveraged various BLSTM methods to identify toxic spans. We find that a combination of a "Greedy-Remove" and "Attention" based methods with a focus on precision produces the highest overall results.

We examined how well various toxic related datasets are able to transfer to this task. We find that OLID, an offensive labeled dataset, transfers the highest, achieving an F1 only 0.6 lower than the provided dataset. Furthermore, we found that combining the viewpoints of the datasets in an ensemble model can increase the performance (F1 to 55.1).

Finally, we examined the errors our system made and the causes. We found that many errors were made when translating from tokens to character index. Future work would improve on this translation step to tighten the span indication.

References

Betty van Aken, Julian Risch, Ralf Krestel, and Alexander Löser. 2018. Challenges for toxic comment classification: An in-depth error analysis. In *Proceedings of the 2nd Workshop on Abusive Language Online (ALW2)*, pages 33–42, Brussels, Belgium. Association for Computational Linguistics.

Lora Aroyo, Lucas Dixon, Nithum Thain, Olivia Redfield, and Rachel Rosen. 2019. *Crowdsourcing Subjective Tasks: The Case Study of Understanding Toxicity in Online Discussions*, page 1100–1105. Association for Computing Machinery, New York, NY, USA.

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. *ICWSM*.

Antigoni Founta, Constantinos Djouvas, Despoina Chatzakou, Ilias Leontiadis, Jeremy Blackburn, Gianluca Stringhini, Athena Vakali, Michael Sirivianos, and Nicolas Kourtellis. 2018. Large scale crowdsourcing and characterization of twitter abusive behavior. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 12.

Yu-Lun Hsieh, Minhao Cheng, Da-Cheng Juan, Wei Wei, Wen-Lian Hsu, and Cho-Jui Hsieh. 2019. On the robustness of self-attentive models. In *Annual Meeting of the Association for Computational Linguistics (ACL)*.

Mladen Karan and Jan Šnajder. 2019. Preemptive toxic language detection in Wikipedia comments using thread-level context. In *Proceedings of the Third Workshop on Abusive Language Online*, pages 129–134, Florence, Italy. Association for Computational Linguistics.

Brendan Kennedy, Mohammad Atari, Aida M Davani, Leigh Yeh, Ali Omrani, Yehsong Kim, Kris Coombs Jr, Shreya Havaldar, Gwenyth Portillo-Wightman, Elaine Gonzalez, et al. 2018. The gab hate corpus: A collection of 27k posts annotated for hate speech. *PsyArXiv. July*, 18.

Joosung Lee. 2020. Stable style transformer: Delete and generate approach with encoder-decoder for text style transfer.

John Pavlopoulos, Léo Laugier, Jeffrey Sorensen, and Ion Androutsopoulos. 2021. Semeval-2021 task 5: Toxic spans detection (to appear). In *Proceedings of the 15th International Workshop on Semantic Evaluation*.

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

D. Terdiman. 2018. Here's How Facebook Uses AI To Detect Many Kinds Of Bad Content.

Michael Wiegand, Melanie Siegel, and Josef Ruppenhofer. 2018. Overview of the germeval 2018 shared task on the identification of offensive language.

Xing Wu, Tao Zhang, Liangjun Zang, Jizhong Han, and Songlin Hu. 2019. Mask and infill: Applying masked language model for sentiment transfer. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 5271–5277. International Joint Conferences on Artificial Intelligence Organization.

Jingjing Xu, Xu Sun, Qi Zeng, Xiaodong Zhang, Xuancheng Ren, Houfeng Wang, and Wenjie Li. 2018. Unpaired sentiment-to-sentiment translation: A cycled reinforcement learning approach. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*.

Harish Yenala, Ashish Jhanwar, Manoj K Chinnakotla, and Jay Goyal. 2018. Deep learning for detecting inappropriate content in text. *International Journal of Data Science and Analytics*, 6(4):273–286.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019a. Predicting the Type and Target of Offensive Posts in Social Media. In *Proceedings of NAACL*.

Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. 2019b. SemEval-2019 task 6: Identifying and categorizing offensive language in social media (OffensEval). In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 75–86, Minneapolis, Minnesota, USA. Association for Computational Linguistics.

Marcos Zampieri, Preslav Nakov, Sara Rosenthal, Pepa Atanasova, Georgi Karadzhov, Hamdy Mubarak, Leon Derczynski, Zeses Pitenis, and Çağrı Çöltekin. 2020. SemEval-2020 Task 12: Multilingual Offensive Language Identification in Social Media (OffensEval 2020). In *Proceedings of SemEval*.

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*, pages 745–760. Springer.