TAR System Description Paper Template

Stjepanović Mateo, Žabčić Mislav, Tolić Filip

University of Zagreb, Faculty of Electrical Engineering and Computing
Unska 3, 10000 Zagreb, Croatia
`mateo.stjepanovic@fer.hr`, {`mislav.zabcic`, `filip.tolic`}`@fer.hr`

Abstract

In this paper, we tackle the problem of detecting hate speech and finding a difference between hate speech and offensive language. We know that some machine learning approaches got good results when distinguishing the hate speech from offensive language. Problem with almost all machine learning approaches is that we need a lot of data to train model well. Main idea of this work is not to fix the results of previous machine learning methods, or to outperform them, but to get similar results with much less data on which model can learn. Our results show that using even baseline semi-supervised active learning model can reduce data we need to learn model on same level.

1. Introduction

The impact that social media have on our daily lives has kept growing for the last 15 years. With the appearances of social media like Twitter, Facebook, Tumblr and many others, interactions between the users and the ability to express their opinions has never been easier. The rise in the use of hate speech, incited by the evolution of social media, was followed by the need to detect it. Facebook CEO, Mark Zuckerberg, predicts that in a 5 to 10 year period they will have the AI tools necessary to build a successful hate speech detection machine. What makes the detection of hate speech such a difficult task? To address that question we have to define hate speech. We define hate speech as communication that expresses hate towards a certain group of people based on their race, ethnicity, national origin, gender, religion, sexual orientation, etc. It often includes the use of offensive language or cuss words. It's important to differ offensive language from hate speech because people often use offensive language in their everday language. Most rap songs, which people often cite, are filled with offensive words such as b*tch, n*gga, wh*re, etc. While communicating with their closest friends, people often use cuss words as a joke with no intention of hurting the other person. Because of situations like these, we need to build a classifier that recognizes the difference between hate speech and offensive language.

We use the dataset annotated by CrowdFlower workers. The workers were asked to annotate 25000 tweets into 3 categories: hate speech (class 0), offensive language (class 1) and speech that contains neither hate speech nor offensive language (class 2). A clear imbalance in the dataset is shown in a histogram displayed in Figure 1.(Davidson et al.2017), creators of the used dataset, used logistic regression to detect hate speech. Even though their model was rather successful, they still struggled with separating hate speech from offensive language. In this paper, we explore the use of active learning to deal with the problems of separating offensive language from hate speech and solving the imbalance of the used dataset.

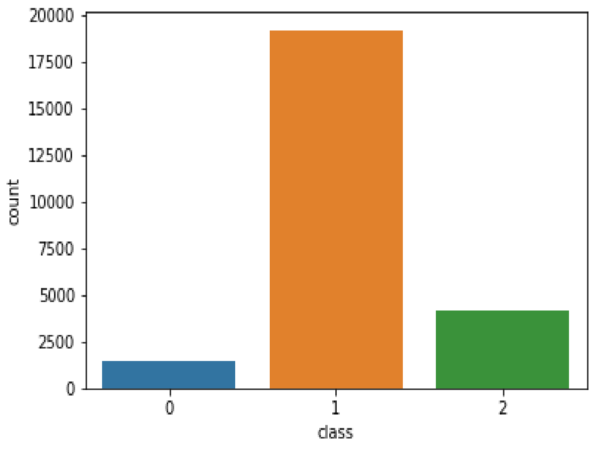

Figure 1: Histogram shows the imbalanced dataset. Tweets from dataset were extracted using *hate* words from Hatebase but not many of them contained hate speech.

2. Related work

Hate speech today is one big unresolved problem. Even companies like Facebook are still struggling with it. It is important to separate hate speech from offensive language, even so because of new laws against hate speech(Davidson et al.2017). As said in (Davidson et al.2017) offensive language is mostly used in everyday life and often doesn't do any harm. Author in that same paper tried few variants of models and decided that Logistic Regression with L2 regularization is best model right now. While this i problem yet to be solved there is no much features that surely works. In (Davidson et al.2017) they used several features to capture information about syntactic and semantic structures. Porter stammer is used to create unigram, bigram and trigram features and then used TF-IDF to put weights to them. Using NLTK they constructed Part of Speech tags as features. They showed that their model is working great, and got pretty high official metrics (precision, recall and F1 score). In introduction of their paper they stated that lexical structures often mislead model and produces false positives on hate speech. In (Waseem et al.2017) authors stated and gave example of one reason why today we can't get relevant scores or dataset. As stated problem is that there is no consensus on what falls under what category. So they proposed new separation methods. It is contained out of two questions:

- *Is the language directed towards a specific individual or entity or is it directed towards a generalized group?*
- *Is the abusive content explicit or implicit?*

Answering those questions we can differ four main classes:

- *Directed Implicit*
- *Generalized Implicit*
- *Directed Explicit*
- *Generalized Explicit*

(Chen and Lin2006) shows that one can implement feature selection methods into SVM it self. This could possibly make great future work in trying to achieve end-to-end solution for hate speech identification problem, as we encountered problem of high number of features, and high CPU and RAM requirement as result of that. In our paper we used baseline features because we didn't want to focus our selves on feature selection. Given approach in (Chen and Lin2006) gives us idea how to improve our model later on.

Problem of data size and time needed to manually label same data was always been in machine learning. That's why active learning as semi-supervised approach is developed. (Luo et al.2005) proposed new active learning model based on multi-class support vector machines. They showed that there is a way to make SVM work with probabilities and give us proper active learning model. Even though we didn't implement this model, plan is to implement it in the future and test it on state-of-the-art models. Their algorithm works as follows:

- 1. Start with an initial training set and an unclassified set.
- 2. A multi-class support vector machine is built using the current training set.
- 3. Compute the probabilistic outputs of the classification results for each data on the unclassified set. Suppose the class with highest probability is a and the class with second highest probability is b. Record the value of P(a) and P(b) for each unclassified data.
- 4. Remove the data from the unclassified set that have the smallest difference in probabilities between them (P(a) P(b)) for the two highest probability classes, obtain the correct label from human experts and add the labeled data to the current training set.
- 5. Go to 2

5

In (Yang et al.2009) authors introduced their own active learning model based on multi-class SVMs. It is little different from one before on manner that they calculate probability on each data in unlabeled pool. Given that model is computational expensive if unlabeled pool in large size. Their experiment also show that they outperform state-of-the-art models, which can only be a sign to continue further on out work to get end-to-end solution for hate speech identification.

3. Preparing data

We used dataset that was introduced in (Davidson et al.2017). They prepared the data by annotating the twitter posts with three available classes, 0 meaning it's a hate speech, 1 being an offensive language and 2 being nothing of previous two. People from CrowdFlower annotated the data by marking each tweet, they classified the tweet by majority class. Minimum of annotators for each tweet was three. As seen in Figure1 data is unevenly distributed. That has big influence on learning outcomes. Data preparation was in given order.

First we had to clear the text from all of the usual twitter stuff. Stuff like hash tag or @. Writing regular expression to automate this was a way to go. Except hash tags and special signs we also removed URL, dates, times, etc.

After cleaning text same text is switched to lower case and got tokenized. Text was stemmed using Porter stammer.

Using *NLTK* library we removed all classical stop words in English language. After tokenization 50 most frequent words are removed, that allowed us to lower size of our features. That same size was one of the biggest problems in our work. We tried doing that on our baselines and got much better results so we used that approach with active learning.

The last thing we had to do is to turn this tokens to vectors.We used TF-IDF vectorizer, from *sklearn* library, to extract features from our text. After this preparations we where ready to train our baseline models and our active learning method.

Table 1 shows the output of our data preparation. After cleaning, tokenization, etc. we got array of words that represents input post in dataset.

4. Model

As stated above problem with machine learning is that we need a lot of data to train model properly. Hate speech identification problem is that there is no relevant and good enough database. Database introduced in (Davidson et al.2017) has lack of equal distribution of data across all classes. To sidestep that problem we decided to go with semi-supervised model. So called active learning.

While this is just a brief overview and some kind of prototype to show us if active learning is good approach, we introduced just one of the baseline models for active learning. Label Spreading and Label Propagation are only two semi-supervised model that one can find in *sklearn* library. In (Luo et al.2005) and (Yang et al.2009) it is shown that active learning can be of much use when coming to problem of to few data.

Tweets before cleaning		Lmao RT @MoeMartin44 The in soles in Reebok Classics can't even handle diabetic weight Rick Ross holds
Array of words		lmao, sole, reebok, classic, can't, even, handl, diabet, weight, rick, ross, hold

Table 1: First row shows the original tweet post from dataset, the second row shows cleaned, tokenized and steammed tweet post

Idea of active learning is that in first step we take small subset out of training set, and train our model on it. In next steps we need to calculate probabilities of good estimation of model. Given these probabilities we then choose subset of that data that has the lowest probability of estimation. After that model is trained once again. This continues until maximum number of iteration is reached, metrics are satisfying, or model reached some other condition.
In our work we used 100 iterations, and lowest 100 probabilities. These number were chosen because, when tried greater number of new data in training set we would lose precision in calculating how much data we need to have the results satisfying. With these numbers we get desired results after half of iterations, and we kept running other half too see how will model behave.

Table **??** show how are metric improved through time. In table **??** we can see that optimal, and peak of model is on 60 iterations. That's about 6000 labeled data on which we train our model.

Active learning algorithm goes like this:

- *1. Select m size subset from training set*
- *2. Train label spreading model on given subset*
- *3. Predict probabilities on remaining set from training set*
- *4. Select n worst probabilities and put that data expand initial training set with these data*
- *5. Go to 2.*

In next section, section about Experiments, it is shown that our active learning model, even though it is baseline model, has a lot of potential for future work.

5. Experiments

We compare our active learning model to the model created by (Davidson et al.2017) and two different baseline models using the same features as our model. One of those models is a logistic regression and the other is an SVM. SVM and Logistic regression were evaluated using nested cross validation while our active learning model was evaluated on a regular cross validation. For our results we use those that gave us the highest F1 score for hate speech class. The results are shown in Table 3. The (Davidson et al.2017) model was trained using 10000 TF-IDF features including POS taggs and some general information features as tweet length, number of words,etc. Our 3 models used only 1000 TF-IDF features. We can see that because of the lack of features the F1 scores for hate class fall significantly compared to (Davidson et al.2017) but they are consistant for all 3 models. We assume the reason for this is that the sklearn implementation of TF-IDF vectorizer, used in this paper, takes only top used ngrams from the dataset and this results in a substantial information loss about the hate speech class because of the imbalanced dataset. To fix this problem, in future work we should either try to "cherry-pick" ngrams that show in both the hate speech class and offensive language class or use a different set of features. When comparing the F1 score for all classes (Davidson et al.2017) has an F1 score of 0.9 while our best model is logistic regression with the score of 0.85. Our SVM model has a F1 score of 0.8, which is a considerable drop compared to (Davidson et al.2017), while our active learning model's F1 score is only 0.03 behind SVM. Considering that the (Davidson et al.2017) was trained using a substantially larger set of features, it is expected to have better results. That is why we are going to compare our active learning model to our own implementations of SVM and logistic regression that use the same set of features. All our testing was done on an Intel(R) Core(TM) i5-7200U CPU with 16 GB of RAM. Table 4. shows the difference in F1 scores and time needed to train each model. As we can see the logistic regression class outperforms SVM and active learning model by a large margin. Even though the F1 scores for hate speech class are the same for all three models, the biggest difference is in detecting clean text with F1 scores varying from 0.46 to 0.7. Even though clean text has the biggest varrience between the models, the biggest impact on the total F1 score comes from the offensive language class. The reason for this is the imbalanced dataset where over 70 Even though our active learning model has worse results then the other two models, it is important to mention that it has the smallest differences in precision and recall for each class. Why is this the case? As we mentioned, we chose active learning model as a way to train a model on a much smaller dataset. The model was trained on around 8000 and tested on approximately 14000 tweets. Beacuse of that, the difference in the dataset distribution wasn't as impactfull as it was on the other two models that were trained on 20000 tweets. We believe that given a balanced dataset, all our models would perform much better, but active learning wouldn't need as much data to train it self, which is significant step forward.

6. Conclusion

Today hate speech is a big problem in the world, but detecting it might be even a bigger problem. It's hard for a human to detect the difference between hate speech and just offensive language, and teaching a machine to solve that problem is an interesting and ambitious project. We know that lexical detection methods have a low precision because hate speech can be hidden and sometimes indirect and also they classify all messages containing particular terms as hate speech so we get a lot of fake positive results. Some machine learning methods give better results

Table 2: Table shows how precision, recall and F1 score behaves on different iterations.

Iteration			0				40				60				100	
		P	R	F1		P	R	F1		P	R	F1		P	R	F1
0		0.11	0.04	0.06		0.23	0.13	0.17		0.35	0.20	0.25		0.37	0.19	0.25
1		0.80	0.91	0.85		0.82	0.90	0.86		0.82	0.91	0.86		0.82	0.91	0.86
2		0.47	0.30	0.37		0.52	0.37	0.43		0.53	0.36	0.43		0.51	0.36	0.42

Table 3: Table contains F1 scores for four different models. Column F1(hate) shows F1 scores calculated only on hate speech class while column F1(all) scores are calculated as average F1 scores on all three classes.

	F1(hate)	F1(all)
Davidson et al.	**0,51**	**0,90**
Logistic Regression	0,26	0,85
SVM	0.26	0.80
Active Learning	0.26	0.77

Table 4: Table shows F1 scores for every class individually and for all classes together for each of our models. It also contains estimated average time needed to train one of ten K folds in our cross validation.

	F1(hate)	F1(offensive)	F1(clean text)	F1(all)	Training time(minutes)
Logistic Regression	**0.26**	**0.92**	**0.70**	**0.85**	**6 min**
SVM	**0.26**	0.90	0.55	0.80	70 min
Active Learning	**0.26**	0.87	0.46	0.77	16 min

and that was our point of interest that we used when we started working on this paper. We where reading and we tried to recreate the paper (Davidson et al.2017) where they used Logistic Regression with L2 regularization as machine learning method, but when we tried to recreate the results we ended up with a model that took more than 24 hours to create. Then we realized that semi-supervised active learning method could get us similar results with less time and less data. It's a bit harder to implement active learning but we got what we wanted. We can conclude that active learning can spare us the time of model building, and also the time and resources needed to annotate big dataset, but also the results are a little bit worse when using SVM.

References

Yi-Wei Chen and Chih-Jen Lin, 2006. *Combining SVMs with Various Feature Selection Strategies*, pages 315–324. Springer Berlin Heidelberg.

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

Tong Luo, Kurt Kramer, Dmitry B. Goldgof, Lawrence O. Hall, Scott Samson, Andrew Remsen, and Thomas Hopkins. 2005. Active learning to recognize multiple types of plankton. *Journal of Machine Learning Research*, pages 589–613, April.

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 Language Online*, pages 78–84. Association for Computational Linguistics.

Bishan Yang, Jian-Tao Sun, Tengjiao Wang, and Zheng Chen. 2009. Effective multi-label active learning for text classification. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 917–926. ACM.