Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, pages 86–90
with a Shared Task on Offensive Language Detection.
Language Resources and Evaluation Conference (LREC 2020), Marseille, 11–16 May 2020

Multitask Learning for Arabic Offensive Language and Hate-Speech Detection

Ibrahim Abu-Farha[1] and Walid Magdy[1,2]
[1] School of Informatics, The University of Edinburgh
Edinburgh, United Kingdom
[2] The Alan Turing Institute
London, United Kingdom
i.abufarha@ed.ac.uk, wmagdy@inf.ed.ac.uk

Abstract

Offensive language and hate-speech are phenomena that spread with the rising popularity of social media. Detecting such content is crucial for understanding and predicting conflicts, understanding polarisation among communities and providing means and tools to filter or block inappropriate content. This paper describes the SMASH team submission to OSACT4's shared task on hate-speech and offensive language detection, where we explore different approaches to perform these tasks. The experiments cover a variety of approaches that include deep learning, transfer learning and multitask learning. We also explore the utilisation of sentiment information to perform the previous task. Our best model is a multitask learning architecture, based on CNN-BiLSTM, that was trained to detect hate-speech and offensive language and predict sentiment.

Keywords: Arabic, hate-speech, offensive language

1 Introduction

Social media platforms provide a versatile medium for people to communicate with each other, share ideas and express opinions. These user-driven platforms have a challenge when it comes to controlling the content being fed into them. People have different intentions, while some might use these platforms for their intended purposes, others might be sharing inappropriate content such as pornographic images or racist speech towards others. Detecting such content is very important for these platforms. For example, it is necessary to add filtration features to hide adult-only content in order to protect children (Mubarak et al., 2017). Also, such detection systems are important to provide real time monitoring of the content being fed to these platforms, which could be promoting hate crimes or racism against various groups of people. An early detection of this phenomena could help in preventing the escalation from speech to actions (Waseem and Hovy, 2016). Platforms such as Twitter, Facebook and YouTube are putting effort into fighting the spread of hate-speech, racism and xenophobia on their platforms. Thus, having robust detection systems is extremely important (Waseem and Hovy, 2016). According to Cambridge Dictionary[1] hate-speech is defined as "public speech that expresses hate or encourages violence towards a person or group based on something such as race, religion, sex or sexual orientation". There has been a large amount of studies on how to automatically detect hate-speech, offensive language and obscene content. The approaches vary from using word-lists, syntactic and semantic features to deep learning models.

This paper is a description of our submission (SMASH team) to the shared task of offensive language and hate-speech detection (Mubarak et al., 2020). This task is a part of the Open-Source Arabic Corpora and Corpora Processing Tools (OSACT4) workshop. In this paper, we explore various approaches to detect hate-speech and offensive language, which include deep learning, transfer learning and multitask learning. Our best model is a multitask learning architecture that was trained to detect both hate-speech and offensive language.

2 Related Work

There has been some work on tasks related to hate-speech and offensive language detection, especially in English. Most of the work view the problem as a classification task where the goal is to assign a specific label to a given input. Early work on such task includes (Yin et al., 2009), where the authors used multiple features such as n-grams and sentiment to train a classifier for harassment detection. In their work, they used a manually labelled dataset of discussion threads. Razavi et al. (2010) proposed a multilevel classifier to detect offensive messages, in which they utilised a set of manually collected words and phrases. Nobata et al. (2016) proposed a machine learning based approach for abusive language detection. They utilised multiple features such as n-grams, linguistic features, syntactic features and word embeddings. They also created a new dataset using comments from Yahoo Finance and News. Davidson et al. (2017) built a corpus of tweets that contain hate-speech. In their work, they utilised a lexicon of hate-speech keywords in order to collect relevant tweets. The collected data was labelled into three classes: hate-speech, offensive language and neither. They built a classifier to detect hate-speech, their analysis showed that raciest and homophobic are likely to be classified as hate-speech. In (Malmasi and Zampieri, 2017), the authors trained an SVM classifier for hate-speech detection, the classifier relies on n-gram based features. In 2019, OffensEval (Zampieri et al., 2019) was introduced to be part of SemEval. This competition has multiple sub-tasks such as offensive language detection, offence categorisation and offence target identification.

Regarding Arabic, there were few attempts to approach the problems of offensive language and hate-speech detection. These include the work of Mubarak et al. (2017), where the

[1]https://dictionary.cambridge.org/dictionary/english/

authors proposed a method to automatically expand word lists for obscene and offensive content. In their work, they created an initial list of seed words, which was used to collect a set of tweets. From the collected tweets, they extracted the patterns that are used to express offensiveness. They followed that with manual assessment and used other resources to create the final word-list. In another work (Mubarak and Darwish, 2019), the authors used the word-list as a seed to create a training set, which they used to experiment with and create an offensive language detector. Alakrot et al. (2018) proposed a method to detect abusive language, they used SVM with n-gram features for the classification where they achieved an F1-score of 0.82. They collected their own dataset from YouTube comments. Albadi et al. (2018) created a dataset of religious hate-speech discussions on Twitter, they used this data to train an RNN based classifier for automatic detection of hate-speech, they achieved 0.84 Area under the ROC curve. The authors also used their dataset to create multiple hate-speech lexicons. Haidar et al. (2017) experimented with cyberbullying detection, where they utilised a dataset that they collected from Facebook and Twitter. They used n-gram features and experimented with multiple classifiers such as Naive Bayes and SVM.

3 Dataset Description

The dataset is the same one provided in SemEval 2020 Arabic offensive language task. It consists of 10,000 tweets labelled for offensive language and hate-speech. The annotation assumes that a tweet is offensive if it contains an insult, attack or inappropriate language. While a tweet is assumed to contain hate-speech if it was directed towards a group or an entity. Table 1 shows the statistics of the training and development sets. The test set, which contains 2000 tweets, was not released for evaluation purposes.

Label	**Training Set**	**Development Set**
Hate-Speech	361	44
Non hate-speech	6,639	956
Offensive	1,410	179
Non-offensive	5,590	821

Table 1: Dataset statistics.

4 Proposed Models

In this section, we provide details of the different steps and models we used in the experiments.

4.1 Data Preprocessing

This step is important in order to clean data from unnecessary content and transform it into a coherent form, which can be processed and analysed easily. Since we are using Mazajak's word embeddings (Abu Farha and Magdy, 2019), we used the same steps used by the authors as follows:

- Letter normalisation: unifying the letters that appear in different forms. We replace { إ ، أ ، آ } with {ا}, {ة} with {ه} and {ى} with {ي} (Darwish et al., 2014).
- Elongation removal: removing the repeated letters which might appear specially in social media data (Darwish et al., 2012).
- Cleaning: removing unknown characters, diacritics, punctuation, URLs, etc.

4.2 Text Representation

In this step, we transform textual data into a representation that can be used for the task we are aiming to accomplish. There are different ways to represent textual information, in our implementation we use word embeddings. Word embeddings are a dense vector representation of the words, we utilised the word embeddings provided by (Abu Farha and Magdy, 2019). These are skip-gram word2vec embeddings, which were built using a corpus of 250M tweets.

4.3 Models

This section provides details of the different approaches and models tested for the different tasks.

BiLSTM

Long short-term memory (LSTM)(Hochreiter and Schmidhuber, 1997) networks are quite powerful at capturing relations over sequences. However, they capture the dependencies in one direction, and sometimes they might lose important information, here where bidirectional LSTMs (BiLSTM) are useful. BiLSTMs are two LSTMs where each one goes over the input in a different direction. This configuration allows the network to have a representation of the whole sequence at any point. The output of the LSTM is passed to a dense layer with softmax activation which emits the final output.

CNN-BiLSTM

This architecture consists of a convolutional neural network (CNN) followed by a BiLSTM network. Such architecture is commonly used in the literature for text classification tasks. The benefits of such architecture is that the CNN has the capability to capture patterns and correlations within the input. The CNN would work as a feature extractor and these features are fed into a BiLSTM network which captures dependencies within these features.
This architecture consists of a 1D convolutional layer followed by a max-pooling layer, then the BiLSTM part. Finally, we have a dense layers followed by the output layer.

Transfer Learning

Transfer learning has been a turning point in the field of computer vision which led to huge improvements and breakthroughs. In the last couple of years, the research in natural language processing (NLP) has caught up with the introduction of pre-trained language models such as Elmo(Peters et al., 2018) and ULMFit(Howard and Ruder, 2018). The introduction of Bidirectional Encoder Representation from Transformers (BERT) (Devlin et al., 2019) led to a revolution in the NLP world. BERT-based models achieved state-of-the-art results in many tasks. In the proposed architecture, we utilise a pre-trained language model and fine tune it for a specific task, i.e. transfer learning. In our experiments we use the multilingual BERT which was

trained on 104 languages. It utilises a vocabulary of 110K WordPeice tokens. BERT's architecture consists of 12 layers with 768 hidden units in each of them, and 12 attention heads.
In the experiments we fine tune BERT to be used for classification. This is done by adding a fully connected layer and a softmax layer after the the pre-trained model. Then the model is trained for a small number of epochs to adjust the weights for the specific task.

Multitask Learning
In multitask learning (MTL), the objective is to utilise the process of learning multiple tasks in order to improve the performance on each of them (Caruana, 1997). These tasks are usually related and have some common aspects between them. Thus, having the model to learn these tasks would give it the ability to utilise some cues from one task to improve the other. MTL has been used to improve many NLP tasks such as syntactic chunking and POS-tagging (Søgaard and Goldberg, 2016), even BERT (Devlin et al., 2019) was built using multitask learning settings.
In this architecture, we utilise that the data is labelled for both tasks, hate-speech and offensive language. Based on the given definitions of the tasks and the annotated data, we can assume that if a sentence contains hate-speech, it is offensive. Thus, we try to utilise this correlation and train the model for both tasks at once (the model is called MTL).

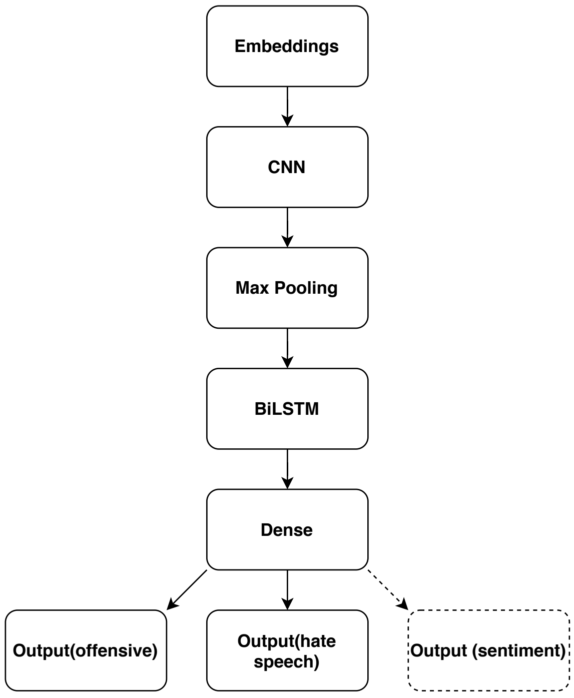

Figure 1: CNN-BiLSTM architecture in multitask learning configuration.

To extend this idea, we decided to incorporate more information through adding sentiment information. The reason for this is that offensive language or hate-speech are usually sentimental and express a negative emotion towards the target. In order get the sentiment labels, we used Mazajak sentiment analyser (Abu Farha and Magdy, 2019). With the sentiment labels added as an objective, the new model (MTL-S) learns to predict three labels, sentiment, hate-speech, and offensive language.
An issue that might occur is that when we use the sentiment from another system, we might be propagating some of error and uncertainty to the new model. In order to reduce such uncertainty in the sentiment labels, we had two variants of the experiment. In the first one, we used the labels returned from Mazajak as they are. In the second, we masked the sentiment to be negative if the sentence was originally labelled as hate-speech or offensive language. The notion behind this experiment is that a hate-speech or offensive content are always bearing negative sentiment, but those might be expressed in an indirect way which could result in an incorrect sentiment label, this model is called (MTL-S-N). In this architecture, we utilised the CNN-BiLSTM architecture, the only difference is that we have a forking before the output layer to accommodate for the different outputs as shown in Figure 1.

5 Performance and Evaluation

5.1 Experimental Setup

In the implementation, we used Python as the programming language for all the experiments. For the deep learning experiments, *Keras* (Chollet and others, 2015) was used on top of *Tensorflow* (Abadi et al., 2015) back-end. For all the experiments, *ReLU* activation and *Adam* optimiser with learning rate of 0.0001 were used. Table 2 shows the hyper-parameters for each Architecture. Regarding the experiments with BERT, we utilised HuggingFace's Transformers library (Wolf et al., 2019). We used the provided *BertForSequenceClassification* implementation along with BertAdam optimiser. We trained the models for 4 epochs with learning rate of $1e{-}5$. The maximum sequence length was set to the maximum length seen in the training set.

Hyper-parameter	BiLSTM	CNN-BiLSTM
#LSTM cells	128	128
recurrent dropout	0.2	0.2
dropout	0.2	0.2
#filters	-	300
filter size	-	3
pooling size	-	2
#hidden units	-	128

Table 2: Hyper-parameters used for each architecture.

5.2 Results and evaluation

We experimented with the different models mentioned previously. To have an initial measure of the performance, we used two different baselines. One where we always assign the majority label (baseline-1), this was done for both tasks. The second baseline (baseline-2) was Multinomial Naive Bayes (MNB) trained on unigram and bigram TF-IDF representation of the input. The evaluation metric for both tasks is macro-average F1-score. Table 3 shows the results on the development set. From the table, it is noticeable that the multitask learning models (MTL and MTL-S-N) achieved the best results on the development set. This shows that the extra information learned through learning multiple objectives was effective to improve the performance. It is noticeable that the BERT based model achieved relatively lower results compared to the other models. This is due to the fact that BERT has a limited vocabulary and was trained on the Arabic Wikipedia, which is in modern standard Arabic (MSA). Thus, BERT was not able to effectively handle the dialectal content within the dataset. In general, the models are better in detecting offensive language than hate-speech. This is due to the small number of training examples of hate-speech data.

Model	OFF	HS
baseline-1	0.450	0.490
baseline-2	0.490	0.390
BiLSTM	0.896	0.671
BERT	0.857	0.719
CNN-BiLSTM	0.901	0.702
MTL	0.899	**0.737**
MTL-S	0.899	0.712
MTL-S-N	**0.904**	0.730

Table 3: Macro F1 scores achieved on the development set for hate-speech (HS) and offensive language (OFF) detection tasks.

5.3 Submission Results

These tasks are part of a shared competition organised in OSACT4 (Mubarak et al., 2020), where we participated as SMASH team. For each task, we submitted the best performing model as our primary submission, the results are shown in Table 4. For the hate-speech task we submitted the MTL model which achieved a macro F1-score of 0.76 on the test set (ranked 6th out of 13). For the offensive language task, we submitted the MTL-S-N model which achieved and F1-score of 0.877 on test set (ranked 5th out of 14).

Model	OFF	HS
MTL	-	0.76
MTL-S-N	0.877	-

Table 4: Macro F1 scores achieved by the best models on the test set for hate-speech (HS) and offensive language (OFF) detection tasks.

6 Conclusion and Future Work

In this work, we presented our system to perform hate-speech and offensive language detection. The experiments show that using a multitask learning setting was extremely useful due to the high correlation between the two tasks. We also explored the effect of adding sentiment information, which proved to be useful. This is explained by the fact that hate-speech and offensive content always bear negative sentiment. Thus, sentiment information is correlated with hate-speech and offensive language. In the future, we hope to improve the results through the utilisation of other resources such as lexicons and experimenting with more multitask learning settings.

Acknowledgements

This work was supported by the D&S Programme of The Alan Turing Institute under the EPSRC grant EP/N510129/1.

7 References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Abu Farha, I. and Magdy, W. (2019). Mazajak: An online Arabic sentiment analyser. In *Proceedings of the Fourth Arabic Natural Language Processing Workshop*, pages 192–198, Florence, Italy, August. Association for Computational Linguistics.

Alakrot, A., Murray, L., and Nikolov, N. S. (2018). Towards accurate detection of offensive language in online communication in arabic. *Procedia computer science*, 142:315–320.

Albadi, N., Kurdi, M., and Mishra, S. (2018). Are they our brothers? analysis and detection of religious hate speech in the arabic twittersphere. In *2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 69–76. IEEE.

Caruana, R. (1997). Multitask learning. *Machine learning*, 28(1):41–75.

Chollet, F. et al. (2015). Keras. `https://keras.io`.

Darwish, K., Magdy, W., and Mourad, A. (2012). Language processing for arabic microblog retrieval. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 2427–2430. ACM.

Darwish, K., Magdy, W., et al. (2014). Arabic information retrieval. *Foundations and Trends® in Information Retrieval*, 7(4):239–342.

Davidson, T., Warmsley, D., Macy, M., and Weber, I. (2017). Automated hate speech detection and the problem of offensive language. In *Eleventh international aaai conference on web and social media*.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June. Association for Computational Linguistics.

Haidar, B., Chamoun, M., and Serhrouchni, A. (2017). A multilingual system for cyberbullying detection: Arabic content detection using machine learning. *Advances in Science, Technology and Engineering Systems Journal*, 2(6):275–284.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Howard, J. and Ruder, S. (2018). Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia, July. Association for Computational Linguistics.

Malmasi, S. and Zampieri, M. (2017). Detecting hate speech in social media. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pages 467–472, Varna, Bulgaria, September. INCOMA Ltd.

Mubarak, H. and Darwish, K. (2019). Arabic offensive language classification on twitter. In *International Conference on Social Informatics*, pages 269–276. Springer.

Mubarak, H., Darwish, K., and Magdy, W. (2017). Abusive language detection on Arabic social media. In *Proceedings of the First Workshop on Abusive Language Online*, pages 52–56, Vancouver, BC, Canada, August. Association for Computational Linguistics.

Mubarak, H., Darwish, K., Magdy, W., Elsayed, T., and Al-Khalifa, H. (2020). Overview of osact4 arabic offensive language detection shared task. 4.

Nobata, C., Tetreault, J., Thomas, A., Mehdad, Y., and Chang, Y. (2016). Abusive language detection in online user content. In *Proceedings of the 25th International Conference on World Wide Web*, WWW '16, page 145–153, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.

Peters, M., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana, June. Association for Computational Linguistics.

Razavi, A. H., Inkpen, D., Uritsky, S., and Matwin, S. (2010). Offensive language detection using multi-level classification. In *Canadian Conference on Artificial Intelligence*, pages 16–27. Springer.

Søgaard, A. and Goldberg, Y. (2016). Deep multi-task learning with low level tasks supervised at lower layers. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 231–235, Berlin, Germany, August. Association for Computational Linguistics.

Waseem, Z. and Hovy, D. (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, June. Association for Computational Linguistics.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., and Brew, J. (2019). Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Yin, D., Xue, Z., Hong, L., Davison, B. D., Kontostathis, A., and Edwards, L. (2009). Detection of harassment on web 2.0. *Proceedings of the Content Analysis in the WEB*, 2:1–7.

Zampieri, M., Malmasi, S., Nakov, P., Rosenthal, S., Farra, N., and Kumar, R. (2019). 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, June. Association for Computational Linguistics.