Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, pages 76–81
with a Shared Task on Offensive Language Detection.
Language Resources and Evaluation Conference (LREC 2020), Marseille, 11–16 May 2020

Arabic Offensive Language Detection with Attention-based Deep Neural Networks

Bushr Haddad*, Zoher Orabe*, Anas Al-Abood*, Nada Ghneim**
*Damascus University
Damascus, Syria
**AlSham Private University
Damascus, Syria
{bushr.haddad, zoherorabe999, anasabood3}@gmail.com
n.ghneim@aspu.edu.com

Abstract

The abusive content on Arabic social media such as hate speech, sexism, racism has become pervasive, and it has a lot of negative psychological effects on users. In this paper, we introduce our work aiming to detect Arabic offensive language and hate speech. We present our two deep neural networks Convolutional Neural Network (CNN) and Bidirectional Gated Recurrent Unit (Bi-GRU) used to tackle this problem. These models have been further augmented with attention layers. In addition, we have tested various pre-processing and oversampling techniques to increase the performance of our models. Several machine learning algorithms with different features have been also tested. Our bidirectional GRU model augmented with attention layer has achieved the highest results among our proposed models on a labeled dataset of Arabic tweets, where we achieved 0.859 F1 score for the task of offensive language detection, and 0.75 F1 score for the task of hate speech detection.

Keywords: Abusive Language, Text Mining, Arabic Language, Social Media Mining, Deep Learning, Convolutional Neural Network, Gated Recurrent Unit, Attention Mechanism, Machine Learning.

1. Introduction

The internet and social media provide people with a range of benefits and opportunities to empower themselves in a variety of ways. There are millions of people using social media platforms to maintain social connections and support networks that otherwise would not be possible. All of these benefits led to a huge growth of social media interactions in the last few years. Arabic language has a very high rate of growth in social networking usage. Based on the Arab social media report (Salem, 2017), the average rate of using Arabic language in social media reaches 55% in 2017.

With the massive increase of the social connections, there has also been an increase of abusive language that should be detected and eliminated from these networks, due to its negative impacts on users. This paper has been prepared for the competition of OSACT4 shared task on offensive language detection (Mubarak et al., 2020). The competition was divided into two sub-tasks; sub-task A (offensive language detection) and sub-task B (hate speech detection). Offensive language is defined as any implicit or explicit insult or attack against other people, or any inappropriate language, while hate speech[1] is defined as any abusive speech targeting individuals (a politician, a celebrity, etc.) or particular groups (a gender, a religion, a country, etc.).

Hate speech is known to be complex and ambiguous because it was not just a words identification. (Zhang and Luo, 2019) showed that detecting a hateful content is a challenging task compared to non-hateful content due to their lack of unique, discriminative linguistic features. On the other hand, Arabic language is known to be difficult and ambiguous, the Arabic content on social media is noisy with different dialects, and most Arabic users do not care about using correct grammar, or spelling. All of these factors made these tasks nearly impossible in the past to detect and identify using conventional features widely adopted in many language-based tasks.

Based on (Al-Hassan and Al-Dossari, 2019), offensive language detection task depends mainly on text mining approaches such as NLP and machine learning algorithms.

In the rest of this paper, a brief of related works are summarized in section 2. In section 3, we represent our data preparation process, then our proposed models are presented in section 4. In section 5, a brief discussion on the results is addressed. At the end, a short summary and insights for the future are presented.

2. Related Works

Different researches have addressed both offensive language detection, and hate detection subjects. (Cambray and Podsadowski, 2019) evaluated their model on OffensEval 2019 English dataset and presented their best model as a bidirectional LSTM; followed by a two-branch bidirectional LSTM and GRU architecture (macro F1 of 73% for offensive language detection task and 61% for targeted hate speech detection).

(Mubarak, Darwish, and Magdy, 2017) have created a list of 288 of Arabic obscene words and other list of 127 of hashtags. They used this list in addition to patterns to collect Arabic abusive tweets from Twitter API during 2014. They classified tweet users into two groups, namely: those who authored tweets that did not include a single obscene word from list words (clean group) and those who used at least one of the words in list at least once (obscene

[1] https://www.dictionary.com/browse/hate-speech

group). They computed unigram and bigram counts in both of them and computed the Log Odds Ratio (LOR) for each word unigram and bigram that appeared at least 10 times. (Alakrot, Murray, and Nikolov, 2018) have collected a dataset of 15,050 comments from YouTube and labelled them manually by three annotators. This dataset was collected in July 2017. They applied some preprocessing operations on the dataset, and then applied SVM classifier on tf-idf features with different methods for text normalizing (macro F1 of 82%). (Mohaouchane, Mourhir, and Nikolov, 2019) have used the same YouTube dataset. They used Word2Vec embeddings and trained different neural networks models namely: convolutional neural network (CNN), bidirectional long short-term memory (Bi-LSTM), Bi-LSTM with attention mechanism, and combined CNN and LSTM. The CNN model achieved the highest accuracy (87.84%), precision (86.10%), and F1 score (84.05%) among other models.

Several works have investigated the problem of hate speech detection in English language. (Zhang and Luo, 2019) Firstly: they demonstrated that hateful content exhibits a 'long tail' pattern compared to non-hate, and secondly: they proposed two deep neural networks, CNN and GRU, to identify specific types of hate speech. They outperformed the previous state of the art methods by 5 percentage points in macro-average F1. (Gambäck and Sikdar, 2017) evaluated CNN model on various word embeddings, and achieved their best score (F1 score of 78%) with CNN model trained on Word2Vec word embeddings. (Badjatya et al, 2017) evaluated several neural architectures on a 16 K annotated tweets benchmark dataset. Their best setup involved a two-step approach using a short-term word-level memory (LSTM) model, tuning GLoVe or randomly initializing word embedding, and then training a gradient boosted decision tree (GBDT) classifier on the average of the tuned embedding in each tweet. They achieved the best results using randomly initialized embeddings (macro F1 of 93%).

In Arabic language there was a limited number of works in this area. (Mulki et al., 2019) constructed a Levantine hate speech and abusive dataset from Twitter. (Haddad, Mulki, and Oueslati, 2019) constructed a Tunisian hate and abusive speech dataset. (Albadi, Kurdi, and Mishra, 2018) built a lexicon of Arabic terms related to religion abuse along with hate score, the labeled dataset is then used to train several classification models using lexicon-based, n-grams-based, and deep-learning based approaches. Their best model achieved 0.84 area under receiver operating characteristic curve (AUROC).

3. Data Preparation

The main dataset used in this work, is the one that was firstly presented at OffensEval 2020. This dataset contains 10000 tweets, only 5% of tweets are labeled as hate speech while 19 % of the tweets are labeled as offensive and the other 81% as inoffensive tweets. The data has been given by the following format: a tweet followed by a label indicating its class {OFF/HS, NOT_OFF/NOT_HS}, all hate speech tweets considered to be offensive language, but not vice versa. The dataset was divided into 70% *train* data, 10% *validation* data, and the rest 20% *test* data.

3.1 Data Preprocessing

This dataset run through a series of pre-processing steps in order to get the most normalized language form. Twitter data is known for its unstructured and unformed language. So, making a good preprocessing steps will results in a much better text representation. As a first step, we removed non-Arabic words, diacritization, punctuations, emoticons and some other stopwords, while we replaced some words with their simplified Arabic equivalent, (example: "URL" will be substituted with "يورل"). We intend to study the effect of the emoticons in a future work. Normalization step was also applied (example, replacing "ة" with "ه", "ى" with "ي", "[أإا]" with "ا"). In addition, elongated and some consecutive repetitive characters that people usually write on their dialect speech are converted back to their original form (example: "هههههه" was be converted to "هه", and "غووول" to "غول"). This step is very important as some Arabic speakers tend to repeat and elongate some characters on their dialect speech.

3.2 Data Balancing

After preprocessing, and as the provided dataset is imbalanced, we applied different methods to balance out the classes for better model performance. Researches shows that classifiers trained on imbalanced dataset may tend to have a high number of false negatives (offensive tweets which are misclassified as inoffensive tweets) and thus a lower recall (Mohaouchane, Mourhir, and Nikolov, 2019). Such detection systems are preferable to identify offensive language even if it sometimes mistakes inoffensive language as offensive. Because the number of inoffensive language exceeds the number of offensive language, so it is preferable to have a higher recall comparing to a higher precession. A lot of ways have been used previously trying to balance out the data like loss function weighting (Cui et al., 2019), down sampling and oversampling. For our case and for subtask A we used an external augmenting technique by adding some offensive and inoffensive comments from an already constructed Arabic dataset collected from YouTube comments (Alakrot, Murray, and Nikolov, 2018) (as YouTube have a similar type of language to Twitter). Thus, augmenting our data from this YouTube comments data guarantee the compatibility of the language added with our given data. We have achieved a balanced dataset that contains approximately the same number of offensive and inoffensive samples.

For subtask B, we did not use the same technique used for sub-task A. Different reasons were behind this decision: insufficient hate speech examples in these datasets (only 468 tweets) (Mulki et al., 2019), some datasets are specific for one kind of hate speech, like religious hate speech (Albadi, Kurdi, and Mishra, 2018), and some datasets are specific for one or more Arabic dialectical form (example: Tunisian (Haddad, Mulki, and Oueslati, 2019) or Levantine (Mulki et al., 2019)). However, for future works, we intend to test augmenting our given data with the combination of all the existing hate speech datasets. Instead, we used the random oversampling technique by shuffling the words into hate speech tweets to create new samples. This method repeated many times over the undersampling class (hate

speech) until each class in the dataset is represented apporoximately equally. Table 1 presents the number of tweets in each category before and after balancing.

	Before	After
Offensive	1330	7184
Inoffensive	5670	8705
Subtask A total	7000	15889
Hate Speech	361	7486
Not Hate Speech	6639	6639
Subtask B total	7000	14125

Table 1: Number of samples before and after balancing

3.3 Data Representation

The main idea of data representation is to represent words as feature vectors. Each entry in a word vector stands for one hidden feature inside the word meaning. Word embedding is one of the best data representation neural network depends on. Word embedding can reveal semantic or syntactic dependencies. We used the publically available Word2Vec Arabic model (AraVec) (Mohammad et al., 2017) that supports two types of words embeddings skipGram and CBOW, each of which has been trained on one of three datasets: tweets, Wikipedia articles, or web pages. AraVec also provides multiple dimensions for its word vectors. The choice of words embeddings used in this work is the vectors that was trained on the twitter dataset with the skipGram architecture. The choice of twitter model is to be compatible with the language used in the given dataset and also to ensure a huge cover of the dialectical words found on the tweets. We have further checked the overlap between our balanced dataset and the AraVec Twitter model. Table 2 shows the number of tokens that has been found on the final balanced datasets, number of dataset tokens found on the AraVec Twitter model and the percentage of overlapping between them.

N. of tokens	Balanced Dataset	AraVec Twitter	Overlapping
Sub-task A	40562	33777	83%
Sub-task B	24504	21592	88%

Table 2: Number of tokens found on the balanced datasets, dataset tokens found on the AraVec Twitter model, and the percentage of overlapping.

4. Proposed Approaches

Before introducing our attention based models. We will introduce two deep neural models; convolutional neural network and Gated Recurrent unit, and then augment these models with an attention layer, and finally compare the attention models with the original versions.

4.1 Convolutional Neural Network (CNN)

Although the main purpose of creating CNN was to convolve over image data, CNN have recently been used a lot in document classification, and experiment on textual data has shown improvements in multiple tasks (Kim, 2014). In this work, we use relatively the same model presented in (Kim, 2014), (Mohaouchane, Mourhir, and Nikolov, 2019), and (Gong et al., 2016), with some parameters' changes (number of filters and filter sizes). The first layer of this model is an embedding layer (represents a lookup table for words already in the table, and others are initialized with random weights, and tuned jointly while learning). The second layer contains a number of filters with different filter sizes to capture different contextual features. Then, a Max-pooling layer was used to capture the most important features. After that, all feature vectors is concatenated together in order to be passed for a fully connected layer of one neuron. The output layer is responsible of classifying the tweet into a positive class (offensive/ hate speech) or a negative class (inoffensive/ not hate speech). We refer to this model as CNN. This model is shown at figure 1.

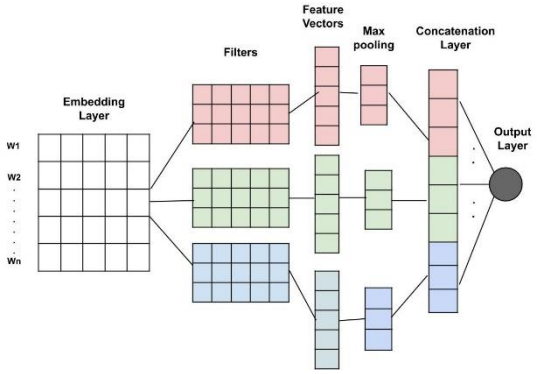

Figure 1: CNN Model

4.2 Bidirectional Gated Recurrent Unit (Bi-GRU)

Bidirectional GRUs are a type of bidirectional recurrent neural networks with only the input and forget gates. It allows for the use of information from both previous time steps and later time steps to make predictions about the current state. We stack two layers of bidirectional GRU on top of each other, followed by a fully connected layer of one neuron to predict the output. We refer to this model as Bi-GRU, figure 2 shows the model.

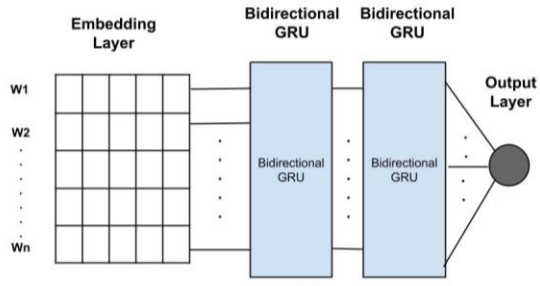

Figure 2: Bi-GRU Model

4.3 Convolution Neural Network with Attention (CNN_ATT)

Certainly, some words on the sentence plays more important role than others and some words are more important than others to the class of the tweet. (Bahdanau, Cho, and Bengio, 2014) was the first to present this type of attention in seq2seq model to improve machine translation model. After that, attention-based neural networks have

been used in various tasks and achieved promising performance, such as (Gong et al., 2016) retweet prediction, (Xu et al., 2015) image captioning, (He and Golub, 2016) question answering, and so on.

In this section, we present a CNN neural network model augmented with an attention layer. Using an attention layer after the max pooling layer can learn which max pooled feature vectors are most important and thus learn which n-grams are most important for classification. So, after max pooling the feature vectors, they are stacked above each other and fed into an attention layer to learn the most important feature vectors. We refer to this model as CNN_ATT, Figure 3 shows the model.

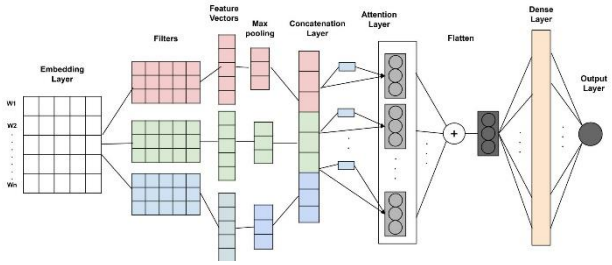

Figure 3: CNN_ATT Model

4.4 Bidirectional Gated Recurrent Unit with Attention (Bi-GRU_ATT)

Although Bidirectional Recurrent model has achieved a very good results in many tasks, they still treat all steps as equal. In this model, we stack the vectors of all computed steps, and then we calculate the score function of each step and then implement a folding layer to generate a context vector indicating the importance of each step vector. This is followed by one dense layer of 64 neurons with Relu activation function (to increase the nonlinear property of this model). After that, we add a fully connected layer with one neuron of a sigmoid activation function as the output layer. This model is referred as Bi-GRU_ATT and shown in figure 4.

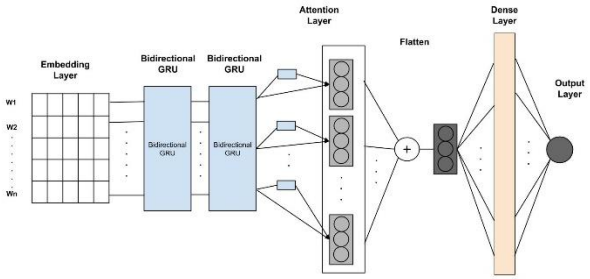

Figure 4: Bi-GRU_ATT Model

4.5 Basic machine Learning Models

We compared our proposed model with three basic machine learning classifiers (Ridge, SVM, and Logistic Regression). The Ridge classifier used RMSE and l2 penalty on both bag of word features (Bow_Ridge) or Tf-idf features (Tf-idf_Ridge). The SVM classifier was trained on bag of word (Bow_SVM) or tf-idf features (Tf-idf_SVM). In this method, we conducted a grid search to obtain the best parameters of SVM kernel. The Logistic Regression classifier was trained on bag of word features (Bow_LR) or tf-idf features (Tf-idf_LR).

5. Experimental Results

For Subtask A and B, we used the Adam optimizer, to adapt the learning rate and optimize the training of the neural networks. For both subtasks, we used the binary cross-entropy loss function. We also used an early stopping strategy based on a long-term moving-average of the F1 score evaluated at the end of every epoch.

Number of filters, recurrent units and neurons in dense layers have been optimized using a grid search. We used filter sizes of 1, 2, 3, and 4 to capture unigram, bigram, trigram, and quad-gram features. These filters are not organized in a sequential order, but rather in parallel to each other as shown in Fig. 1 and Fig. 3.

We used an early stopping strategy to determine the number of epochs that should be used. For other hyper-parameter optimization, we performed a manual twerking over successive runs on the validation set.

As a result, we found that the following parameters yielded the best validation performance based on our experiment.

- Maximum Tweet length= 100
- Filter sizes = [1,2,3,4]
- Number of Filters = 64
- Recurrent unit in Bidirectional GRUs' = 128, 64
- Neurons number in dense layers = 64
- Dropout rate = 0.2 and 0.3
- Batch size = 512
- Number of epochs = 5

To initialize the word vectors, the publicly available AraVec word vectors were used (Mohammad et al., 2017) (skipGram model from Twitter, with 1,476,715 tokens). The dimension of the vectors used is 100. Table 2 shows a good overlapping with our data, and for words that are not found in the vocabulary of pre-trained words, we initialized them with random vectors and tuned them while training.

Hereafter, we present our *validation* results. Table 3 and 4 lists the results of using various baseline machine learning classifiers (Bow_LR, Tf-idf_LR, Bow_Ridge, Bow_SVM, Tf-idf_Ridge, and Tf-idf_SVM).

Subtask A	Avg. Acc	Avg. P	Avg. R	Avg. F1
Bow_LR	0.86	0.77	0.79	0.78
Tf-idf_LR	0.85	0.75	0.79	0.77
Bow_Ridge	0.880	0.8	0.78	0.79
Tf-idf_Ridge	0.905	**0.85**	0.8	**0.83**
Bow_SVM	0.872	0.78	0.77	0.78
Tf-idf_SVM	**0.906**	**0.85**	**0.81**	**0.83**

Table 3: Performance of baseline models on subtask A

Subtask B	Avg. Acc	Avg. P	Avg. R	Avg. F1
Bow_LR	0.491	0.49	0.45	0.36
Tf-idf_LR	**0.492**	**0.51**	0.55	**0.37**
Bow_Ridge	0.463	0.5	0.49	0.35

Tf-idf_Ridge	0.462	**0.51**	**0.57**	0.36
Bow_SVM	0.391	0.49	0.46	0.31
Tf-idf_SVM	0.386	0.5	0.52	0.31

Table 4: Performance of baseline models on subtask B

Comparing the models of Tf-idf and Bow features, we can see that tf-idf features is relatively better than Bow features, which may be due to the tf-idf's ability to determine how relevant a given word is in a particular document. We can also observe that Tf-idf_SVM achieved a high performance on subtask A, this may be because SVM generalized better with nonlinear kernel that has been found by a grid search over its parameters.

Table 5 and 6 shows the comparison of the proposed models (CNN, Bi-GRU, CNN_ATT, and Bi-GRU_ATT) when evaluated on subtask task A and B respectively.

Subtask A	**Avg. Acc**	**Avg. P**	**Avg. R**	**Avg. F1**
CNN	0.92	0.63	0.84	0.85
CNN_ATT	0.92	0.86	0.85	**0.86**
Bi-GRU	0.91	0.85	**0.86**	0.85
Bi-GRU_ATT	**0.93**	**0.91**	0.83	**0.86**

Table 5: Performance of proposed models on subtask A

Subtask B	**Avg. Acc**	**Avg. P**	**Avg. R**	**Avg. F1**
CNN	0.91	0.63	0.78	0.67
CNN_ATT	0.9	0.63	**0.84**	0.67
Bi-GRU	0.92	0.65	0.78	0.69
Bi-GRU_ATT	**0.93**	**0.66**	0.79	**0.7**

Table 6: Performance of proposed models on subtask B

We can observe that baseline models was less efficient compared to our proposed models in both subtasks. The proposed models have increased the F1 measure of the baseline models on subtask B (from 37% to 70%). We also can observe that models augmented with attention layer can achieve a better performance than the models without the attention layer. The improvement is in the order of 1 to 2% in Recall and F1 score. However, attention layer has achieved a significant improvement in Precision, which means that attention layer helps in detecting the right offensive and hate speech words and thus raising precision. Bi-GRU_ATT achieved the highest accuracy, Precision and F1 score for both subtasks A and B, and outperformed the CNN models. This may be due to the fact that GRU models have more information of text sequence dependencies and order that CNN models does not have. These features seems to be very important for such tasks as shown in (Zhang and Luo, 2019).

Table 7 shows the results of our best model (Bi-GRU_ATT) evaluated on the *test* data.

Bi-GRU_ATT	**Avg. Acc**	**Avg. P**	**Avg. R**	**Avg. F1**
Subtask A	0.91	0.88	0.83	0.85
Subtask B	0.95	0.75	0.74	0.75

Table 7: *Test* performance of Bi-GRU_ATT on subtasks A and B

We observe that Bi-GRU_ATT performance on *test* data - for both subtasks is close to performance on the *validation* data, which is a good indication of a good generalization of the model. We can also notice that general performance on sub-task B is less efficient than performance on sub-task A, due to the hard separation of hate speech from other instances of offensive language.

6. Conclusion

In this paper, we tackle the problem of offensive language and hate speech detection. We proposed our methods for data preprocessing and balancing, and then we presented our Convolutional Neural Network (CNN) and bidirectional Gated Recurrent Unit (GRU) models used. After that, we augmented these models with attention layer. The best results achieved was using the Bidirectional Gated Recurrent Unit augmented with attention layer (Bi-GRU_ATT). Comparing the Precision results of models without attention layers and models with attention layer reveals that attention layer enabled our model to effectively select the relevant input series to the output class, and thus raising Precision score. Future work will consider the same problem working with both character-level and word-level features. Another improvement of models could be using LSTM instead of GRU to capture long range dependencies in tweets, which plays a big role in offensive language and hate speech detection tasks.

7. Bibliographical References

Alakrot, A., Murray, L., & Nikolov, N. S. (2018). Dataset construction for the detection of anti-social behaviour in online communication in Arabic. Procedia Computer Science, 142, 174-181.

Alakrot, A., Murray, L., & 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., & Mishra, S. (2018, August). 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) (pp. 69-76). IEEE.

Al-Hassan, A., & Al-Dossari, H. (2019). Detection of hate speech in social networks: a survey on multilingual corpus. In 6th International Conference on Computer Science and Information Technology.

Badjatiya, P., Gupta, S., Gupta, M., & Varma, V. (2017, April). Deep learning for hate speech detection in tweets. In Proceedings of the 26th International Conference on World Wide Web Companion (pp. 759-760).

Bahdanau, D., Cho, K., & Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473.

Cambray, A., & Podsadowski, N. (2019). Bidirectional Recurrent Models for Offensive Tweet Classification. arXiv preprint arXiv:1903.08808.

Cui, Y., Jia, M., Lin, T. Y., Song, Y., & Belongie, S. (2019). Class-balanced loss based on effective number of samples. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 9268-9277).

Gambäck, B., & Sikdar, U. K. (2017, August). Using convolutional neural networks to classify hate-speech. In

Proceedings of the first workshop on abusive language online (pp. 85-90).
Golub, D., & He, X. (2016). Character-level question answering with attention. arXiv preprint arXiv:1604.00727.
Haddad, H., Mulki, H., & Oueslati, A. (2019, October). T-HSAB: A Tunisian Hate Speech and Abusive Dataset. In International Conference on Arabic Language Processing (pp. 251-263). Springer, Cham.
Kim, Y. (2014). Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882.
Mohaouchane, H., Mourhir, A., & Nikolov, N. S. (2019, October). Detecting Offensive Language on Arabic Social Media Using Deep Learning. In 2019 Sixth International Conference on Social Networks Analysis, Management and Security (SNAMS) (pp. 466-471). IEEE.
Mubarak, H., Darwish, K., & Magdy, W. (2017, August). Abusive language detection on Arabic social media. In Proceedings of the First Workshop on Abusive Language Online (pp. 52-56).
Mubarak, H., Darwish, K., Magdy, W., Elsayed, T., & Al-Khalifa, H. (2020). Overview of OSACT4 Arabic Offensive Language Detection Shared Task. Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools (OSACT), 4.
Mulki, H., Haddad, H., Ali, C. B., & Alshabani, H. (2019, August). L-HSAB: A Levantine Twitter Dataset for Hate Speech and Abusive Language. In Proceedings of the Third Workshop on Abusive Language Online (pp. 111-118).
Salem, F. (2017). Social media and the internet of things towards data-driven policymaking in the Arab world: potential, limits and concerns. The Arab Social Media Report, Dubai: MBR School of Government, 7.
Soliman, A. B., Eissa, K., & El-Beltagy, S. R. (2017). Aravec: A set of arabic word embedding models for use in arabic nlp. Procedia Computer Science, 117, 256-265.
Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhudinov, R., ... & Bengio, Y. (2015, June). Show, attend and tell: Neural image caption generation with visual attention. In International conference on machine learning (pp. 2048-2057).
Zhang, Q., Gong, Y., Wu, J., Huang, H., & Huang, X. (2016, October). Retweet prediction with attention-based deep neural network. In Proceedings of the 25th ACM international on conference on information and knowledge management (pp. 75-84).
Zhang, Z., & Luo, L. (2019). Hate speech detection: A solved problem? The challenging case of long tail on twitter. Semantic Web, 10(5), 925-945.