Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, pages 48–52
with a Shared Task on Offensive Language Detection.
Language Resources and Evaluation Conference (LREC 2020), Marseille, 11–16 May 2020
© European Language Resources Association (ELRA), licensed under CC-BY-NC

Overview of OSACT4 Arabic Offensive Language Detection Shared Task

Hamdy Mubarak[1], Kareem Darwish[1], Walid Magdy[2], Tamer Elsayed[3], Hend Al-Khalifa[4]
[1] Qatar Computing Research Institute, HKBU, Doha, Qatar
[2] School of Informatics, University of Edinburgh, Edinburgh, UK
[3] Qatar University, Doha, Qatar
[4] Information Technology Department, KSU, Riyadh, KSA
{humbarak, kdarwish}@hbku.edu.qa, wmagdy@inf.ed.ac.uk, telsayed@qu.edu.qa, hendk@ksu.edu.sa

Abstract

This paper provides an overview of the offensive language detection shared task at the 4th workshop on Open-Source Arabic Corpora and Processing Tools (OSACT4). There were two subtasks, namely: Subtask A, involving the detection of offensive language, which contains unacceptable or vulgar content in addition to any kind of explicit or implicit insults or attacks against individuals or groups; and Subtask B, involving the detection of hate speech, which contains insults or threats targeting a group based on their nationality, ethnicity, race, gender, political or sport affiliation, religious belief, or other common characteristics. In total, 40 teams signed up to participate in Subtask A, and 14 of them submitted test runs. For Subtask B, 33 teams signed up to participate and 13 of them submitted runs. We present and analyze all submissions in this paper.

Keywords: OSACT, Arabic Offensive Language, Arabic Hate Speech, Shared Task, CodaLab

1. Introduction

Offensive speech (vulgar or targeted offense), as an expression of heightened polarization or discord in society, has been on the rise. This is due in part to the large adoption of social media platforms that allow for greater polarization. The OSACT4 shared task provides a platform to bring researchers from around the world to tackle the detection of offensive and hate speech in the realm of Arabic social media. The shared task has two subtasks. Subtask A involves detecting offensive language, which contains explicit or implicit insults or attacks against individuals or groups and includes vulgar and inappropriate language. Subtask B is concerned with detecting hate speech, which contains insults or threats targeting specific groups based on their nationality, ethnicity, race, gender, political or sport affiliation, religious belief, or other common characteristics. The goal of the shared task is to aid research on the identification of offensive content and hate speech in Arabic language Twitter posts. The shared task attracted a large number of participants. In all, 40 and 33 teams signed up to Subtasks A and B respectively. From them, 13 teams submitted test runs to both subtasks, and only one team submitted runs to Subtask A. Of those teams, 11 submitted system description papers (Abdellatif and Elgammal, 2020; Abu Farha and Magdy, 2020; Abuzayed and Elsayed, 2020; Alharbi and Lee, 2020; Djandji et al., 2020; Elmadany et al., 2020; Haddad et al., 2020; Saeed et al., 2020; Hassan et al., 2020; Husain, 2020; Keleg et al., 2020).

The highest achieved F1 scores for Subtasks A and B were 90.5 (Accuracy = 93.9, Precision = 90.2, and Recall = 90.9) (Hassan et al., 2020) and 95.2 (Accuracy = 95.9, Precision = 95.2, and Recall = 95.9) (Husain, 2020) respectively.

2. Dataset

Subtasks A and B used the SemEval 2020 Task 12 Arabic offensive language dataset (OffensEval2020, Subtask A), which contains 10,000 tweets that were manually annotated for offensiveness (labels: OFF or NOT_OFF). The subtasks used the same OffensEval2020 training (70% of all tweets), dev (10%), and test (20%) splits.

The tweets were extracted from a set of 660k Arabic tweets containing the vocative particle يا ("yA" – O) from April 15 to May 6, 2019. Based on different random samples of tweets, offensive tweets represented less than 2% of of tweets. However, when considering tweets having one vocative particle, the ratio increased to 5%. This particle is mainly used for directing speech to a person or a group. Moreover, when considering the tweets with two vocative articles, the probability of finding offensive tweets increased to 20%. An example offensive statement is يا مقرف يا جبان للأسف هذه تسمى خسة ("yA mqrf yA jbAn h*h tsmY Ksp" – You disgusting coward. This is called wickedness)[1]. Annotation was performed by a native speaker of Arabic with good understanding of several Arabic dialects. Random samples of 100 tweets (50 offensive and 50 non-offensive) were judged by additional three annotators, and the inter-annotator agreement between them was 0.92 (using Fleiss's Kappa coefficient), which validates the quality of data annotation and indicates that judging the offensiveness of tweets is not difficult in many cases. Offensive tweets containing insults or threats targeting a group based on their nationality, ethnicity, race, gender, political or sport affiliation, religious belief, or other common characteristics, were annotated as Hate Speech (labels: HS or NOT_HS). An example tweet containing hate speech is: الله يقلعكم يالبدو يا مجرمين ("Allh yqlEkm yAlbdw yA mjrmyn" – May Allah remove you O Bedouin. You are

[1] We provide Arabic examples, their Buckwalter transliteration, and English translation.

criminals). The distribution of the labels in the dataset is shown in Table 1.

Label	Train	Dev	Test	Total	∼ %
NOT_OFF	5,590	821	1,598	8,009	80%
OFF	1,410	179	402	1,991	20%
NOT_HS	6,639	956	1,899	9,494	95%
HS	361	44	101	506	5%

Table 1: Distribution of labels for Subtasks A and B

Subtask A was concerned with detecting offensive language in general, while Subtask B was concerned with detecting hate speech. Both subtasks used the same train/dev/test splits. For all tweets, some light preprocessing was performed, where user mentions were replaced with @USER, URLs were replaced with URL, and empty lines were replaced with <LF>. The data of Subtask B is more imbalanced than Subtask A data as only 5% of the tweets are labeled as hate speech, while 20% of the tweets are labeled as offensive.

3. Task Settings and Evaluation

Given the strong imbalance between the number of instances in the different classes across Subtasks A and B, we used the macro-averaged F1-score ($\mathcal{F}$) as the official evaluation measure for both subtasks. Macro-averaging gives equal importance to all classes regardless of their size. Other secondary evaluation measures that we used where Precision ($\mathcal{P}$) and Recall ($\mathcal{R}$) on the positive class (offensive or hate speech tweets) as well as the overall Accuracy ($\mathcal{A}$).

Subtasks were hosted on CodaLab platform at the following competition links:
Subtask A: `https://competitions.codalab.org/competitions/22825`
Subtask B: `https://competitions.codalab.org/competitions/22826`

Participants were allowed to submit up to 10 test runs, and they were asked to specify two submissions as their official runs, which would be scored and put on the leaderboard. If official runs are not specified, the latest submissions from each team were considered as official. We gave teams the freedom to describe the differences between their systems in their papers. The idea behind this is to allow teams to examine the effectiveness of different setups on the test set. Macro-average F1 ($\mathcal{F}$) of the first submission is the official score for Subtasks A and B.

We received 43 submissions for Subtask A including 3 failed ones (e.g. incorrect format). For Subtask B, we received 41 submissions including 11 failed ones. Competitions were open from Jan. 21, 2020 until Feb. 19, 2020, and the test sets were available starting on Feb. 13, 2020.

Table 2 lists the names of participating teams and their affiliations.

4. Methods and Results

Most teams performed some data preprocessing, which typically involved character normalization, removal of punctuation, diacritics, repeated letters, and non-Arabic tokens. One team performed extensive preprocessing including normalizing emoticons (translate their English description to Arabic), dialectal to MSA conversion, word category identification (ex. dog, monkey, etc. are mapped to ANIMAL), removal of dialectal stopwords, and hashtag segmentation (Husain, 2020) leading to the best results in Subtask B. As for learning methods, the teams used traditional Machine Learning (ML) techniques, such as SVM and logistic regression, and Deep Neutral Networking (DNN) approaches, such as Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN) including LSTM, biLSTM, and GRUs with and without attention, and fine tuning of contextual embeddings such as BERT and AraBERT.

The highest ranking submissions used an ensemble of different learning methods that combined both traditional ML and DNN approaches. Most teams used similar setups for both subtasks, and two teams chose to use multitask learning (Abu Farha and Magdy, 2020; Djandji et al., 2020).

Table 3 briefly lists the preprocessing and learning methods used by different teams. Tables 4 and 5 list the results of all the teams for Subtasks A and B ranked by F1-measure ($\mathcal{F}$). Per the rules of the shared task, we judged up to two runs for every team (first submission and second submission). As shown in Tables 4 and 5, first submission from all teams always beat their second submission (column $\mathcal{F}$), meaning that best performing systems on the dev set also performed best on the test set as well.

5. Conclusion

This paper presented an overview of the OSACT4 shared task on offensive language and hate speech detection in the Arabic Twitter sphere. The most successful systems in the shared task performed Arabic specific preprocessing, with the winning system for hate speech detection performing extensive preprocessing, and an ensemble of different machine learning approaches, with the winning system for offensive language detection using an ensemble of SVM trained on character-level n-grams and pretrained embeddings (Mazajak) as well as different DNN setups that use FastText, CNN+RNN, and contextual embeddings (multilingual BERT).

6. References

Abdellatif, M. and Elgammal, A. (2020). Sentiment analysis of imbalanced arabic data using ulmfit. *OSACT*, 4.

Abu Farha, I. and Magdy, W. (2020). Multitask learning for arabic offensive language and hate-speech detection. *OSACT*, 4.

Abuzayed, A. and Elsayed, T. (2020). Quick and simple approach for detecting hate speech in arabic tweets. *OSACT*, 4.

Team	Affiliation	Subtasks
Abeer (Abuzayed and Elsayed, 2020)	Islamic University of Gaza, Palestine	1,2
aialharbi (Alharbi and Lee, 2020)	University of Birmingham, UK	1,2
alisafaya	Koç University, Turkey	1,2
alt (Hassan et al., 2020)	Qatar Computing Research Institute, Qatar	1,2
AMR-KELEG (Keleg et al., 2020)	Faculty of Engineering, Ain Shams University, Egypt	1,2
Bushr (Haddad et al., 2020)	Damascus University, Syria	1,2
elmadany (Elmadany et al., 2020)	University of British Columbia, NLP Lab, Canada	1,2
fatemah (Husain, 2020)	Kuwait University, Dep. of Information Science, Kuwait	1,2
hassaansaeed (Saeed et al., 2020)	University of Antwerpen, Belgium	1,2
iaf7 (Abu Farha and Magdy, 2020)	University of Edinburgh, UK	1,2
mabdellatif (Abdellatif and Elgammal, 2020)	Rutgers University, US	1,2
Marc_Djandji (Djandji et al., 2020)	American University of Beirut. Lebanon	1,2
premjithb	Center for Comp. Engineering and Networking, India	1,2
SAJA	Jordan University of Science and Technology, Jordan	1

Table 2: List of participating teams in Subtasks A and B

Alharbi, A. and Lee, M. (2020). Combining character and word embeddings for the detection of offensive language in arabic. *OSACT*, 4.

Djandji, M., Baly, F., antoun, w., and Hajj, H. (2020). Multi-task learning using arabert for offensive language detection. *OSACT*, 4.

Elmadany, A., Zhang, C., Abdul-Mageed, M., and Hashemi, A. (2020). Leveraging affective bidirectional transformers for offensive language detection. *OSACT*, 4.

Haddad, B., Orabe, Z., Al-Abood, A., and Ghneim, N. (2020). Arabic offensive language detection with attention-based deep neural networks. *OSACT*, 4.

Hassan, S., Samih, Y., Mubarak, H., Abdelali, A., Rashed, A., and Absar Chowdhury, S. (2020). Alt submission for osact shared task on offensive language detection. *OSACT*, 4.

Husain, F. (2020). Osact4 shared task on offensive language detection: Intensive preprocessing based approach. *OSACT*, 4.

Keleg, A., El-Beltagy, S. R., and Khalil, M. (2020). Asu_opto at osact4 - offensive language detection for arabic text. *OSACT*, 4.

Saeed, H. H., Calders, T., and Kamiran, F. (2020). Ocast4 shared tasks: Ensembled stacked classification for offensive and hate speech in arabic tweets. *OSACT*, 4.

Team	Preprocessing	Methods
(Abdellatif and Elgammal, 2020)	simple tokenization, replace words below certain threshold with a special token	DNN: ULMFiT – a fine tuned language model based on a 3 layer RNN (LSTM)
(Abu Farha and Magdy, 2020)	character normalization and diacritic, kashida, repeated letter, and non-Arabic character removal	Multitask CNN+BiLSTM with pertained embedding (Mazajak)
(Abuzayed and Elsayed, 2020)	character normalization and diacritic, kashida, repeated letter, and non-Arabic character removal	SVM, Random Forest, XGBoost, Extra Trees, Decision Trees, Gradient Boosting, and LR; DNN: CNN, RNN, CNN+RNN and two different word representations (tf-idf and pre-trained word embeddings (AraVec)
(Alharbi and Lee, 2020)	character normalization and diacritic, kashida, repeated letter, and non-Arabic character removal. Also split يا ("yA")	LR and XGBoost; DNN: RNN using Mazajak, Aravec, and subword FastText embeddings
(Djandji et al., 2020)	removal of non-Arabic characters, segmentation of words using Farasa segmenter, and splitting of hashtags	fine tuning of contextual embeddings (AraBERT)
(Elmadany et al., 2020)	numbers, usernames, hashtags, and hyperlinks replacement with NUM, USER, HASH, and URL respectively; character normalization; and diacritic removal	fine tuned multilingual BERT-based affective models
(Haddad et al., 2020)	removed non-Arabic words, diacritization, punctuation, emoticons, stopwords, and repeated characters	convolutional neural network (CNN) and bidirectional recurrent neural network with GRU units (Bi-GRU) models with and without attention
(Saeed et al., 2020)	letter normalization, repeated letter removal, and word splitting	DNN: CNN and RNN using contextual embeddings (multilingual BERT) and non-contextual embeddings (Aravec, FastText, word2vec) with an ensemble classifier that combines all outputs using SVM, RF, NB, etc.
(Hassan et al., 2020)	diacritic, kashida, repeated letter, and non-Arabic character removal	ensemble of SVM (character n-grams) and pre-trained embeddings (Mazajak) and DNN: FastText (subword), CNN+RNN, and contextual embeddings (multilingual BERT)
(Husain, 2020)	Intensive preprocessing: normalizing emoticons, dialectal to MSA conversion, word category identification (ex. animals), letter normalization, stopword removal, and hashtag segmentation	SVM (character n-grams)
(Keleg et al., 2020)	word segmentation	LR; DNN: CNN (with Aravec), RNN, and contextual embeddings (multilingual BERT and AraBert)

Table 3: Different methods used by different teams. LR: Logistic Regression; SVM: Support Vector Machines; NB: Naive Bayes; DNN: Deep Neural Network; CNN: Convolutional Neural Network; RNN: Recurrent Neural Network

Team	**First Submission**				Second Submission			
	$\mathcal{F}$	$\mathcal{A}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$	$\mathcal{A}$	$\mathcal{P}$	$\mathcal{R}$
(Hassan et al., 2020)	**90.5**	**93.9**	90.2	**90.8**	89.4	93.4	90.5	88.3
(Djandji et al., 2020)	90.0	93.7	**90.7**	89.4	88.5	93.1	91.9	85.9
(Husain, 2020)	89.8	90.2	89.9	90.2	88.6	89.1	88.6	89.1
(Keleg et al., 2020)	89.6	93.5	90.5	88.7	85.6	90.9	86.2	85.0
(Abu Farha and Magdy, 2020)	87.8	92.4	88.8	86.8	87.8	92.3	88.5	87.1
(Saeed et al., 2020)	87.4	92.4	90.3	85.1	87.8	92.8	91.5	85.1
(Alharbi and Lee, 2020)	86.8	92.1	89.6	84.7	85.7	91.2	87.7	84.1
(Haddad et al., 2020)	85.9	91.5	88.6	83.8	84.6	90.0	84.1	85.2
alisafaya	84.2	90.8	88.4	81.4	81.9	89.5	86.1	79.1
(Abuzayed and Elsayed, 2020)	83.3	89.7	84.7	82.1	82.6	89.9	86.8	79.8
(Elmadany et al., 2020)	82.9	89.4	84.1	81.8	79.3	87.8	82.5	77.1
(Abdellatif and Elgammal, 2020)	77.4	86.2	78.9	76.2	77.4	86.2	78.9	76.2
SAJA	76.2	86.4	80.5	73.6				
premjithb	72.6	81.8	71.9	73.3	72.6	81.8	71.9	73.3

Table 4: Subtask A results

Team	**First Submission**				Second Submission			
	$\mathcal{F}$	$\mathcal{A}$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}$	$\mathcal{A}$	$\mathcal{P}$	$\mathcal{R}$
(Husain, 2020)	**95.2**	95.9	**95.2**	**95.9**	94.4	95.4	94.3	95.4
(Djandji et al., 2020)	82.3	**96.7**	82.8	81.8	80.7	96.5	82.6	78.9
(Keleg et al., 2020)	80.7	96.5	82.1	79.4				
(Hassan et al., 2020)	80.6	96.6	83.8	78.1	75.5	96.4	86.4	70.0
(Saeed et al., 2020)	79.9	96.5	83.4	77.1	77.7	96.6	86.9	72.4
(Abu Farha and Magdy, 2020)	76.1	96.0	80.2	73.0	74.3	95.8	79.1	71.1
(Haddad et al., 2020)	75.0	95.3	75.5	74.6	74.8	95.2	74.7	74.9
(Alharbi and Lee, 2020)	74.2	96.3	86.4	68.5	48.7	95.0	47.5	50.0
(Elmadany et al., 2020)	70.5	95.2	75.2	67.5	67.7	95.6	80.3	63.0
alisafaya	70.4	95.8	81.6	65.4	69.9	94.9	72.7	67.8
(Abuzayed and Elsayed, 2020)	69.4	95.1	74.0	66.5	64.9	94.8	71.6	61.6
premjithb	63.2	92.3	62.2	64.5				
(Abdellatif and Elgammal, 2020)	58.5	95.1	75.0	55.7	58.5	95.1	75.0	55.7

Table 5: Subtask B results