CoLi at UdS at SemEval-2020 Task 12: Offensive Tweet Detection with Ensembling

Kathryn Chapman, Johannes Bernhard, Dietrich Klakow

Department of Computational Linguistics & Phonetics, Saarland University, Germany

{kathrync, johannes}@coli.uni-saarland.de, dietrich.klakow@lsv.uni-saarland.de

Abstract

With today's proliferation of maliciously intended communication across all social media platforms, finding ways of effectively combating these messages grows increasingly important. We present our submission and results for SemEval-2020 Task 12: Multilingual Offensive Language Identification in Social Media (OffensEval 2020) where we participated in offensive tweet classification tasks in English, Arabic, Greek, Turkish and Danish. Our approach included classical machine learning architectures such as support vector machines and logistic regression combined in an ensemble with a multilingual transformer-based model (XLM-R). The transformer model is trained on all languages combined in order to create a fully multilingual model which can leverage knowledge between languages. The machine learning model hyperparameters are fine-tuned and the statistically best performing ones included in the final ensemble. We further discuss the results of our model and see that our broad approach provides competitive but not task-winning performance. We also include an error analysis and potential improvements for future work.

1 Introduction

In the past years, offensive language and other forms of maliciously intended online communication (trolling, cyber-bullying, fake news) have grown drastically on social media platforms such as Facebook, Twitter or YouTube. It is in the interest of everyone to inhibit this form of communication as quickly as possible once it happens. A very efficient and effective way to achieve this is via machine learning and deep learning architectures that automatically identify a message, comment or tweet as maliciously intended. These architectures have been steadily improving at this task.

The purpose of the competition in SemEval-2020 Task 12: Multilingual Offensive Language Identification in Social Media (OffensEval 2020) (Zampieri et al., 2020) is to encourage further progress in the above mentioned challenges. The organizers presented participants with tweets in English, Arabic, Greek, Turkish and Danish. For all languages, task A consisted of identifying whether a tweet was labeled as offensive ("OFF") or non-offensive ("NOT"). Tasks B and C were only available for the English data. They consisted of identifying whether a tweet was targeted ("TIN") or untargeted ("UNT") for task B and in task C whether a targeted tweet was aimed at an individual ("IND"), group ("GRP") or other ("OTH"). We participated in task A for all languages and task B for English.

Data set sizes and label balance varied greatly between languages which provided additional challenges when deciding on how to approach the task. We chose to construct an ensemble of more traditional machine learning architectures (SVM, Logistic Regression, etc.) together with a multilingual transformer-based model (XLM-R) (Conneau et al., 2019) with the thought that the strengths of the different models would help the ensemble generate the most well-rounded results. The ensemble generated promising results during experiments and was submitted for all languages in task A except for Danish. For Danish task A and task B (English), we submitted only the XLM-R model predictions.

Proceedings of the 14th International Workshop on Semantic Evaluation, pages 1916–1924
Barcelona, Spain (Online), December 12, 2020.

2 Related Work

SemEval 2019 Task 6: Identifying and Categorizing Offensive Language in Social Media (OffensEval 2019) (Zampieri et al., 2019b) saw many competitors make use of the recently released pre-trained transformer model, BERT. Liu et al. (2019a), the top performing model in OffensEval 2019, combined a BERT model with several pre-processing steps, such as hashtag segmentation, converting emojis to textual descriptions, and lowercasing the text. Compared to runner-up BERT submissions for the 2019 subtask A such as the rank 2 submission by Nikolov and Radivchev (2019), the rank 3 team UM@LING (Zhu et al., 2019) or the rank 7 submission "SentiBERT" by Yin et al. (2020), this illustrates that proper pre-processing is often neglected over algorithmic advances. Therefore, judging the impact of algorithmic advances proves difficult. Nikolov and Radivchev (2019) compared the performance of a BERT-Large model to those of more traditional machine learning models, such as a logistic regression model and soft voting classifier (SVC). While the BERT-Large model outperformed the other on the test data, the logistic regression model and the SVC outperformed BERT on the dev data.

Another approach from last year's competition, presented by Han et al. (2019), used a bi-directional recurrent architecture with Gated Recurrent Units (GRU) (Chung et al., 2014) and rule-based sentence offensiveness calculations to categorize tweets. This worked especially well for subtask B, where the submission achieved the highest results. It was less successful for subtasks A and C, where the rule-based nature proved to be detrimental to the classifier's performance.

3 Data

3.1 Overview

The data provided by the task organizers consists of social media posts in English, Arabic, Danish, Turkish, and Greek, the non-English data human-annotated using the hierarchical tagset proposed in Zampieri et al. (2019a). The positive/negative classes in subtask A were OFF (offensive)/NOT (not offensive) and in subtask B UNT (untargeted)/TIN (targeted).

English The English data for subtasks A and B contained 9 million+ and 190k+ tweets respectively (Rosenthal et al., 2020). All English data for the OffensEval 2020 task were machine-annotated. Task organizers provided the confidence scores (between 0 and 1) per-tweet obtained by averaging the predictions of multiple supervised classifiers, along with the standard deviations.

Arabic The Arabic social media data were collected using the Twitter API, the full process is described in Mubarak et al. (2020). The authors took care to ensure that the Arabic tweets collected were not biased toward any particular dialect, genre, or topic, and used linguistic features (rather than a keyword list) to identify tweets most likely to contain offensive language or intent. All URLs, numbers, and tweet-specific tokens (mentions, retweets, and hashtags) were removed from the data. This resulted in a training data set of 8000 tweets, 19% of which represent the positive class "OFF."

Danish The Danish social media posts, described in Sigurbergsson and Derczynski (2020), were taken largely from the r/Denmark and r/DANMAG subreddit posts and comments. Additional data were collected from user-generated comments on the Facebook page for the Danish media company, Ekstra Bladet. As in Zampieri et al. (2019a), a pre-defined list of keywords was used to identify tweets which could contain offensive language or intent. Any mention of a specific person was replaced with '@USER', and the resulting data set contained 2961 training tweets, 13% of which were labeled as "OFF."

Turkish The Turkish data set, collected from Twitter and described in Çöltekin (2020), was the largest data set after English. All tweets containing a URL were discarded, as were any tweets made by a verified Twitter account (i.e. one belonging to a public figure, celebrity, public office, etc) and retweets. This resulted in a training data set of 31,756 tweets, 19% of which were labeled as "OFF."

Greek The Greek data, described in detail in Pitenis et al. (2020), followed a similar data collection approach to Zampieri et al. (2019a), using a pre-defined offensive keyword list to elicit tweets which were more likely to contain offensive language. This resulted in a training set of 8743 tweets, 28% of which were of the "OFF" class. All URLs, emojis, and emoticons were removed from the tweets, and twitter handle mentions were replaced with '@USER'.

3.2 Additional Data-Processing

We decided on a threshold of 0.5 to map the English data's confidence scores to binary labels. Any confidence score below or equal to the threshold was labeled with the negative class, while any above was labeled with the positive for the training.

Additionally, we shuffled the English data for subtask A and selected the first 1 million tweets for the machine learning models' training data, and the first 200k tweets for the deep learning model. Additional to these 200k machine-annotated tweets, we added 12k human-annotated tweets from Zampieri et al. (2019a) as we expected our model predictions to be evaluated on human-annotated data. This resulted in roughly 15% of the tweets for subtask A and 21% of those for subtask B belonging to the positive classes ("OFF"/"UNT"), respectively.

Finally, we created balanced dev data sets for each language. For English, we used two separate 1000-tweet dev data sets, one human-annotated and one machine-annotated. For the remaining languages, we created a 95/5 train/dev split.

3.3 Preprocessing

The following preprocessing steps were the same for all languages, but were only applied to data for the classical machine learning models.

Emojis In order to discern meaningful content from the emojis present in a tweet, we used a project that converts the emojis into their respective textual representation[1]. An example might be ":face_with_tears_of_joy:" for the crying with laughter emoji. This approach was inspired by the winner of last year's OffensEval competition (Liu et al., 2019a). This substitution was done for all languages, meaning that, as an example, a model trained on Arabic tweets will have had English text representations of the emojis as an additional source of information.

Tokenization We tokenized the emoji substituted tweets using a specific twitter tokenizer [2]. This was done after processing the emojis in order to incorporate their textual information directly into the tweets as if the user had written them that way. Other steps during tokenization included proper handling of special text emoticons such as "o.O". Words were not lower-cased.

Vectorization Hyper-parameters for the vectorization of the tweets were fine-tuned and inspired by Nobata et al. (2016). For the machine learning models, we included various n-gram feature vectors with their TF-IDF scores. On the word level, we took uni-, bi- and trigrams with no lower-casing, while on the character level, we took tri-, four- and fivegrams with lower-casing. We also included two additional features that had a significant impact on performance. One was tweet length and the other was average word length per tweet.

4 Methodology[3]

4.1 Non-Neural Machine Learning Models

The non-neural machine learning models we used came from the scikit-learn library (Pedregosa et al., 2011) and included:

- Multinomial Naive Bayes: baseline for performance.
- Linear Support Vector Classifier: fast and effective for binary classification.
- Support Vector Machines with Stochastic Gradient Descent: same as linear SVC with more fine-tuning options.

[1] https://github.com/carpedm20/emoji

[2] https://github.com/myleott/ark-twokenize-py

[3] Our implementation avilable at: https://github.com/jb-1811/NLP_with_NN_ws1920_OffensEval2020

- Logistic Regression: similarly effective and cheap to train
- ML Stacked Ensemble: a stacked ensemble of the best performing model settings from above with a linear SVC as the final meta-classifier.

The hyper-parameters of all models were fine-tuned and the best five were statistically compared on a balanced Arabic evaluation data set. The final best performing model from each architecture type was included in the ML stacked ensemble. The ensemble ultimately consisted of 1x Linear SVC, 1x Logistic Regression and 2x SVM with SGD. Both top performing SVM with SGD models achieved similar results. The multinomial model was dropped due to mediocre performance.

4.2 XLM-RoBERTa

Overview Bidirectional Encoder Representations from Transformers, known as 'BERT' and described in Devlin et al. (2018), is a pre-trained language model released by Google. The original BERT model is trained using next sentence prediction and masked language modeling (MLM) objectives. BERT has inspired a variety of similar language models, such as Mutli-lingual BERT (mBERT) and a Robustly Optimized BERT Pretraining Approach (RoBERTa) by Liu et al. (2019b), the latter outperforming BERT on several benchmark tasks and trained using only the MLM objective. Extending RoBERTa to a multi-lingual setting, Conneau et al. (2019) propose XLM-RoBERTa (XLM-R), a cross-lingual RoBERTa language model.

These models take as input a sequence of m words and apply sub-word tokenization according to a pre-defined vocabulary (e.g. 'cats' becomes '_cat', 's'). The model-specific tokenizer pads the sequence with a start- and end-sequence token (in XLM-R, '<s>' and '</s>') which produces a sequence of n length. The vectorized representations for these n tokens are then passed to the encoder, which outputs n contextualized embeddings. As these language models are very computationally expensive to train, their typical use involves initializing a model with their pre-trained parameters and adding the necessary task-specific layers. All parameters, including those in the encoder, are updated during this training, known commonly as 'fine-tuning'. For classification tasks, the encoder output representation for the start-sequence token is taken to represent the entire sequence, and is passed to the classification layer.

The XLM-R model is pre-trained on 100 languages, and all languages share the same word-piece vocabulary and parameters, thereby allowing the model to accept input from any language without explicitly specifying the language. As XLM-R supports all OffensEval 2020 languages and outperforms mBERT, we chose this model as our deep learning component of the final ensemble.

Experiments Using the XLM-R implementation available in the HuggingFace Transformer Library[4] (Wolf et al., 2019), we further pre-trained an XLM-R-base model on a concatenation of all subtask A train, dev, and test tweets using the MLM objective with a maximum sequence length of 150 (default parameters otherwise) while maintaining word casing. Then, we initialized a new model with these further pre-trained parameters, added a classification-layer after the encoder, and fine-tuned the whole model for subtask A classification using a cross entropy loss function with class weights. The training data was a concatenation of all languages' subtask A training data. We trained the model for 3 epochs, a maximum sequence length of 150, a batch size of 32 (8 per GPU), an initial learning rate of 5e-5 for the AdamW optimizer, and 2000 warmup steps for the scheduler, with the remaining parameters being default. We compared this model to one which receives no further pre-training on tweets but is fine-tuned on the same data with the same parameters to evaluate the performance boost this further pre-training yields. Additionally, we fine-tuned the further pre-trained model on each language individually to assess whether using a single multi-lingual model yields a performance drop. Finally, we balanced the precision and recall on the dev data sets, setting a different threshold for each language. These per-language thresholds are the only language-specific component of the XLM-R models. Note that we did not find it necessary to balance precision and recall for English subtask A.

[4] `https://github.com/huggingface/transformers` (version 2.4.1)

4.3 Ensembling

Our final ensembling approach used a simple 'majority vote' to decide on a final prediction. Votes came from the best performing individual non-neural models (LinearSVC, SVC with SGD, LogReg) and the non-neural ML stacked ensemble in addition to the neural model (XLM-RoBERTa). This amounted to a final ensemble with four non-neural and one neural votes from which the ensembler chose the most-predicted class. We only made use of this final ensembler for subtask A languages English, Arabic, Turkish, and Greek. The submissions for Danish and subtask B are predictions made solely by the XLM-R model.

5 Results

5.1 Experiments

Here, we present our individual model results on the language evaluation data sets. These results later informed our decision as to which models to include in the final ensemble. We use an F1 macro metric, as this is the chosen metric for OffensEval 2020. F1 macro weighs each class equally, meaning errors for the minority class are penalized more harshly on an individual basis than for the majority class.

Machine Learning Models Table 1 shows the results of the different machine learning models on the different language dev sets, averaged over ten runs. We can see that for all languages except Turkish, the ML stacked ensemble performed the best. For English, the average macro-F1 score was 0.8380, for Arabic 0.8558 and for Greek 0.7966. The Turkish ensemble with a score of 0.7897 lost out to the Linear SVC model with a score of 0.7908. Overall however, the margins between the best scores were quite small and as such each individual model and the ML stacked ensemble will be used in the final ensemble.

Model	English	Arabic	Greek	Turkish
LinearSVC	0.8294	0.8395	0.7913	**0.7908**
SVM with SGD	0.8301	0.8343	0.7537	0.7576
Log. Regression	0.8286	0.8342	0.7771	0.7662
ML Stacked Ensemble	**0.8380**	**0.8558**	**0.7966**	0.7897

Table 1: Results of machine learning models on different language eval. datasets (n=10).
score: F1-macro

XLM-R Table 2 shows the performances of the XLM-R model subtask A experiments described in 4.2. We observed an overall increase in performance when we further pre-trained the XLM-R encoder on the train, dev, and test tweets, as expected. Additionally, with the exception of Danish we did not observe a performance increase when we fine-tuned the model on each language individually. This motivated our decision to use the single further pre-trained, multi-lingual model in our final ensemble. For subtask B, we used the same further pre-trained XLM-R model, but we fine-tuned it separately using the same architecture and parameters as the other model for subtask A. It is worth noting that we did experiment with converting emojis to their textual descriptions as with the machine learning models, but this yielded a performance decrease and we therefore left them as-is.

5.2 Test Data

Table 3 shows the F1 macro score for each model and language, as well as our final submission ('Final Ensemble').

While our final ensemble performs above average across all languages, we observed an unexpected performance drop with respect to the XLM-R model. In order to probe this drop, we again ran the test data sets for both subtasks through the XLM-R model, but removed the step of balancing precision and recall. Table 4 shows the performance increase observed across the subtask A languages both in the individual XLM-R model AND in the Final Ensemble, but a massive performance drop for subtask B. When looking into the label distributions in the test data sets, this makes sense as the test data for subtask B contained

Language	Basic All Lang	Tweet Specific All Lang	Tweet Specific Single Lang
English Human	0.78	**0.80**	0.77
English Machine	**0.91**	0.90	0.90
Arabic	0.83	**0.87**	**0.87**
Danish	0.74	0.73	**0.77**
Greek	0.77	**0.80**	0.79
Turkish	0.77	**0.80**	**0.80**

Table 2: F1-macro scores on dev sets for subtask A. 'Basic' is an as-is xlm-r model fine-tuned for classification, and 'Tweet Specific' is an xlm-r language model further trained on tweets, then fine-tuned for classification. 'All/Single Lang' refers to the fine-tuning language(s).

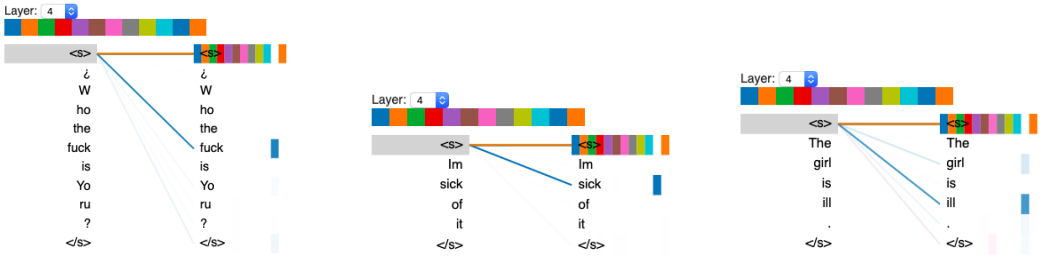
Model	English task A	Arabic	Greek	Turkish	Danish	English task B
LinearSVC	0.9097	0.8117	0.7827	0.7511	-	-
SVM with SGD	0.9095	0.8381	0.7994	0.76252	-	-
Log. Regression	0.9077	0.8513	0.7711	0.7419	-	-
ML Stacked Ensemble	0.9091	0.8095	0.7545	0.7298	-	-
xlm-r	0.9086	0.8267	0.7432	0.7512	0.7086	0.6445
Final Ensemble	0.9091	0.8176	0.774	0.7461	0.7086	0.6445
Mean F1	0.8709	0.7924	0.769	0.7136	0.7067	0.5556
Top submission	0.9222	0.9017	0.852	0.8257	0.812	0.74618

Table 3: F1-macro scores of different models on test datasets. 'Mean F1' refers to all submissions from all teams for a given language in OffensEval 2020.

40% positive labels, which is closer to the distribution in the dev data set on which we balanced the precision and recall. This illustrates that our Final Ensemble can outperform all individual models for Arabic, Greek, and Turkish, which is what we hoped to observe in our final submission.

Error Analysis In order to probe what the models learned, we analyzed some of the errors made by the models on the test data. All models labeled the English subtask A test tweets containing the word 'sick' as false positives. Recall that for the XLM-R model, the contextualized vector representation of the start-sequence token '<s>' is passed to the classification layer. Adapting the BertViz[5] tool to work with our XLM-R model, we recognized that an attention head with respect to '<s>' identifies profanity. Figure 1 shows all attention heads at layer 4 with respect to '<s>' for three input sequences: a true positive, a false positive, and a made-up sequence we would consider a false positive. We interpret these visualizations to mean that 'sick' and its synonyms are recognized by the model as profane words, thereby triggering 'OFF' predictions. Interestingly, the Danish word for 'sick' ('syg') occurs more often with 'NOT' in the training data, yet the model labels all sequences containing this word in the test data as 'OFF'. In fact, when given an input sequence containing 'syg' from the training data, the model predicts an 'OFF' label for the sequence *it saw labeled as 'NOT' in training*. We feel that this illustrates that model is capable to some degree of leveraging its knowledge from one language and applying it to another. However, it also highlights the importance of high-quality data, as incorrectly learned patterns from one language can be propagated to others. To address this, we could have resampled our English training data to include more neutral uses of the word 'sick' to prevent the English misclassifications as well as the propagation to other languages. It is worth noting that we see the same 'OFF' prediction and attention pattern with the Spanish translation of Figure 1c, "La niña esta enferma," a language seen in the initial pre-training of the model but in neither the further pre-training data nor the fine-tuning. Exploring the

[5] https://github.com/jessevig/bertviz



(a) ¿Who the fuck is Yoru?
True: OFF; Pred: OFF

(b) Im sick of it
True: NOT, Pred: OFF

(c) The girl is ill.
Pred: OFF

Figure 1: Attention visualization at layer 4, head 10 for the '<s>' token. The different color squares represent the 12 attention heads, and the darkness of the lines represent the strength of the attention score between a selected token on the left and each token it is attending to on the right.

model's ability in a zero-shot setting like so could be valuable future work.

Language	xlm-r		Final Ensemble	
Arabic	0.86	(+0.034)	0.86	(+0.042)
Greek	0.764	(+0.02)	0.823	(+0.049)
Turkish	0.791	(+0.04)	0.786	(+0.04)
Danish	0.761	(+0.052)	0.761	(+0.052)
English task B	0.568	(-0.077)	0.568	(-0.077)

Table 4: F1 macro and differences on test data when we remove custom thresholds for balancing precision and recall on dev data.

6 Conclusion

Our final ensemble performs above average across all languages and sub-tasks. The ensemble worked well in some cases (Greek, English) and less so in other cases (Turkish, Arabic). This seems to not stem from the characteristics of the languages themselves but more so the differences in label balanced between the respective training and test sets and how we balanced precision and recall during training. However, error and performance analysis on gold labels showed that removing our custom thresholds for balanced precision and recall significantly boosted performance for the final ensemble on the competition test sets (see Table 4). Future work could involve better sampling of the data, or a more sophisticated ensemble approach which utilizes the models' probability outputs rather than their binary predictions. This might also include vote weighting based on specific label precision or recall scores or including meta-learners. Additionally, we were unable to run the XLM-R experiments with an XLM-RoBERTa-Large architecture as planned due to time constraints, and we feel investigating that could yield a performance gain due to the model's increased capacity while is more appropriate for a multi-lingual model. Lastly, a noteworthy challenge in this multi-lingual setting was that the data sets provided were processed to varying degrees, some even containing no emojis at all while others retained them. Perhaps more uniformly processed data could encourage more focus being placed on algorithmic advances rather than pre-processing.

References

Çağrı Çöltekin. 2020. A Corpus of Turkish Offensive Language on Social Media. In *Proceedings of the 12th International Conference on Language Resources and Evaluation*. ELRA.

Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, F. Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *ArXiv*, abs/1911.02116.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Jiahui Han, Shengtan Wu, and Xinyu Liu. 2019. jhan014 at SemEval-2019 task 6: Identifying and categorizing offensive language in social media. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 652–656, Minneapolis, Minnesota, USA, June. Association for Computational Linguistics.

Ping Liu, Wen Li, and Liang Zou. 2019a. NULI at SemEval-2019 task 6: Transfer learning for offensive language detection using bidirectional transformers. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 87–91, Minneapolis, Minnesota, USA, June. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019b. Roberta: A robustly optimized bert pretraining approach.

Hamdy Mubarak, Ammar Rashed, Kareem Darwish, Younes Samih, and Ahmed Abdelali. 2020. Arabic offensive language on twitter: Analysis and experiments. *arXiv preprint arXiv:2004.02192*.

Alex Nikolov and Victor Radivchev. 2019. Nikolov-radivchev at SemEval-2019 task 6: Offensive tweet classification with BERT and ensembles. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 691–695, Minneapolis, Minnesota, USA, June. Association for Computational Linguistics.

Chikashi Nobata, Joel Tetreault, Achint Thomas, Yashar Mehdad, and Yi Chang. 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.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Zeses Pitenis, Marcos Zampieri, and Tharindu Ranasinghe. 2020. Offensive Language Identification in Greek. In *Proceedings of the 12th Language Resources and Evaluation Conference*. ELRA.

Sara Rosenthal, Pepa Atanasova, Georgi Karadzhov, Marcos Zampieri, and Preslav Nakov. 2020. A Large-Scale Semi-Supervised Dataset for Offensive Language Identification. In *arxiv*.

Gudbjartur Ingi Sigurbergsson and Leon Derczynski. 2020. Offensive Language and Hate Speech Detection for Danish. In *Proceedings of the 12th Language Resources and Evaluation Conference*. ELRA.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Da Yin, Tao Meng, and Kai-Wei Chang. 2020. Sentibert: A transferable transformer-based architecture for compositional sentiment semantics. 05.

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 (SemEval)*.

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*.

Jian Zhu, Zuoyu Tian, and Sandra Kübler. 2019. Um-iu@ling at semeval-2019 task 6: Identifying offensive tweets using BERT and svms. *CoRR*, abs/1904.03450.

Appendix

Machine learning model hyperparameters:

1	Linear SVC	C = 1.0 | Penalty: L1-norm
2	SVM + SGD	Constant @ 0.8 | Penalty: L2-norm | Loss: hinge
3	SVM + SGD	Optimal | Penalty: L2-norm | Loss: Squared hinge
4	Logistic Regression	C = 10 | Penalty: L1-norm
Meta-learner	Linear SVC	C = 1.0 | Penalty: L1-norm

Table 5: ML stacked ensemble composition as well as all architecture hyperparameters.
Top performing model from each architectur on balanced Arabic dev. data was chosen. An exception was the SVM with SGD category where both models 1 and 5 performed equally.