Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, pages 82–85
with a Shared Task on Offensive Language Detection.
Language Resources and Evaluation Conference (LREC 2020), Marseille, 11–16 May 2020

Offensive Language Detection in Arabic using ULMFiT

Mohamed Abdellatif, Ahmed Elgammal
Rutgers University - Computer Science
Piscataway, NJ, USA
{mma215, elgammal}@cs.rutgers.edu

Abstract

In this paper, we approach the shared task OffenseEval 2020 by Mubarak et al. (2020) using ULMFiT Howard and Ruder (2018) pre-trained on Arabic Wikipedia Khooli (2019) which we use as a starting point and use the target data-set to fine-tune it. The data set of the task is highly imbalanced. We train forward and backward models and ensemble the results. We report confusion matrix, accuracy, precision, recall and F_1 of the development set and report summarized results of the test set. Transfer learning method using ULMFiT shows potential for Arabic text classification.

Keywords: language models, text classification, transfer learning, opinion mining

1. Introduction

Imbalanced data set is a data set that has at least one (minority) class with *significantly* smaller population than others (majority). If the minority class is a label of interest (to study and predict), imbalanced data represents a challenge since during the training there is relatively no sufficient representation of the minority class(es) to stand out in the trained model. Examples of applications include: finance (e.g. fraud transaction detection), security (e.g. intrusion detection), networking (e.g. anomaly traffic detection), systems (e.g. irregular resource usage detection), medical (e.g. disease [e.g. cancer] detection), nature (e.g. volcano eruption, earthquake, tsunami predictions) and text processing (e.g. opinion mining and spotting hate speech).

Opinion mining and spotting hate speech in the context of social networking using deep learning attracted researchers' attention recently. For example, Park & Fung combined results from CNN (convolutional neural network) and LR (logistic regression) in Park and Fung (2017). They applied their method on the data set by Waseem and Hovy (2016). The same data-set was subject for experimenting a combination of both convolutional and recurrent units by Zhang et al. in Zhang et al. (2018).

State of the art text classification has been recently pushed forward by the advancements of the Transfer Learning (e.g. Devlin et al. (2018), Howard and Ruder (2018) and Radford et al. (2018))

From the work by Mahendran and Vedaldi (2016), inspecting neural network of more than one layer that was trained on a certain data-set of images (say a cats vs dogs binary classification task), the earlier layers tend to capture high level features (e.g. edges, contours .. etc) while the later layers tend to capture low level features (e.g. dogs faces, cats faces .. etc). Even though both types of features are extracted from the same data-set, the high level one is more general so it can be made use of in training the same network for a different task (since almost any kind of image classification will benefit from capturing edges and contours [and similarly general image features] in the weights of the model as concluded by Sharif Razavian et al. (2014)). Observing that, Howard & Ruder (Howard and Ruder (2018)) applied gradual unfreezing associated with discriminative fine-tuning and slanted triangular learning rates (as concluded by Smith (2017)) and successfully apply it on text classification.

Our goal is to investigate applying ULMFiT on the imbalanced Arabic data-sets OffenseEval 2020. Khooli pretrained ULMFiT on Arabic Wikipedia in Khooli (2019). We use their model as a starting point and use the Arabic data-set of interest to fine tune it.

The rest of the paper is organized as follows: we illustrate the data-sets properties in section 2.. In section 4. we describe the model, training parameters and experiments. We show results in section 5. and finally conclude the work in section 6..

2. Data sets

For this work, we use data provided by the organizers of OSACT4. The target of the shared task is to achieve as high macro F_1 score as possible. 10k Arabic tweets were collected. They are splitted to train (7k), development (1k) and test (2k) subsets. The train and development are released along with labels while the test set is released without them. The task has two sub tasks, sub task A is classifying the tweet as 'offensive' vs 'not offensive' while sub task B is about classifying the tweet as 'hateful' vs 'not hateful'. So each tweet is labeled twice. The labeled data sets in both cases are imbalanced with sub task B more so than A.

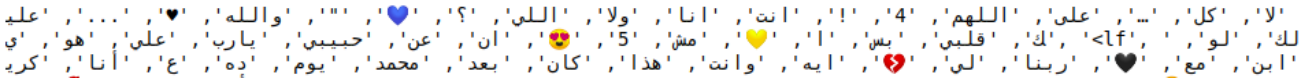
2.1. Sub task A

A tweet is considered *offensive* if it has any level of profanity. Table 1 shows instances count of different classes of sub task A. As the table shows the distribution of both training and development data sets show imbalance between the two existing classes.

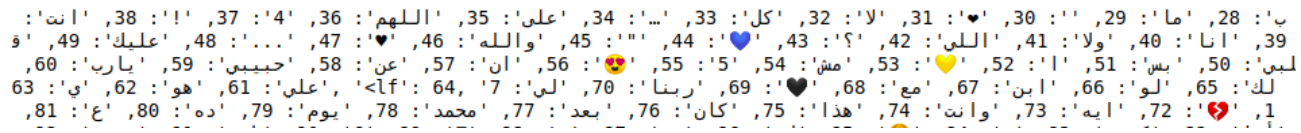
2.2. Sub task B

A tweet is considered *hateful* if it has an attack against one or more person based on their nationality, ethnicity, gender, political affiliation, sport affiliation or religious belief. Table 2 shows instances count of different classes of sub task B. As the table shows the distribution of both training and development data sets show imbalance between the two ex-

Figure 1: Part of vocabulary words



(a) Tokenized



(b) Numericalized

Class	Train	Development	Test
Not offensive (regular)	5.6k	821	-
Offensive	1.4k	179	-
Total	7k	1k	2k

Table 1: Classes distribution of sub task A

isting classes that is more significant than in case of sub task A.

3. Approach

3.1. Pre-processing

We do simple tokenization based on white-spaces and keep words that appeared more frequently than a certain threshold (replaced by 'xxunk'). Since pre-processing is not specific to Arabic, we kept all the non-Arabic words as long as they exist above the threshold (e.g. mentions). Among the special tokens: 'xxpad' is a padding token, 'xxeos' is an end of scentence token, 'xxup' is used to indicate the next word is capitalized (for English parts), 'xxrep' and 'xxwrep' are used to indicate repetition. After segmentation/tokenization, we convert the set of tokens to unique ids. Figure 1 shows part of the resulting vocabulary.

3.2. Method

Language modeling is a problem that deals with learning the joint probability function of sequences of words in this language. Such that given a sequence of a certain number of words, it can assigns a probability for it (as defined in Bengio et al. (2003)).

Inductive transfer learning is to make use of the knowledge learned by training a model (model A) on a source problem to be used towards building another model (model B) that handles a target (different) problem (as defined in Ruder et al. (2019)). In the case of ULMFiT, the source problem is unlabeled (language modeling) and the target problem is (text classification).

ULMFiT transfer learning method (by Howard and Ruder (2018)) can be summarized as three steps applied on two neural networks. The first neural network is a Language Model (LM) the second one is a text classifier. The three steps are 1- pre-training the LM on a general corpus (we used the model by Khooli (2019) for this step), 2- ~~training~~ fine-tuning the LM on the target data-set and then saving a part off the LM (the encoder) and 3- Loading the saved part of the LM (result of step 2) and attaching it to the classifier then ~~train~~ fine-tuning the classifier with the target data-set. Following Howard and Ruder (2018) For both the language model and classifier networks, we used LSTM AWD (by Merity et al. (2017)) which uses a 3 layers LSTM.

4. Experimental Setup

4.1. Experiments

Following the original work by Howard and Ruder (2018), we fine-tune two separate (forward and backward) models, classify twice and average results for each sub-task. That was shown to be always better on all the six of the English data-sets experimented on by Howard and Ruder (2018). We report three different sets of results for each sub task as well to study whether the same conclusion can be made on Arabic imbalanced data-set in question.

Since the source task of the transfer learning (language modeling) needs unlabeled data, we use all the available unlabeled Arabic text (both train and validation) to fine-tune and save (forward and backward) language models and use their encoders for two separate classifiers (two [forward and backward] for each sub task).

4.2. Settings and training

We use fastai library [1] and adjust the hyper-parameters based on the observed performance of training on the development set. The forward language model was trained for 2 epochs while the backward one was trained for 3. After applying a 3-steps of gradual unfreezing, both the forward and the backward classifiers of sub task A were unfrozen and fine-tuned for 3 epochs. Similar steps were followed for sub task B, except we ended up with 30 epochs for fine tuning the forward classifier and only 3 to fine tune the backward one. We use an Nvidia Titan X with 12 GB of memory that allowed us to use a batch size of 64.

[1] https://github.com/fastai/fastai

Class	Train	Development	Test
Not hateful (regular)	6.6k	956	-
Hateful	0.4k	44	-
Total	7k	1k	2k

Table 2: Classes distribution of sub task B

Model	Accuracy	Weighted			Macro		
		precision	recall	F_1	precision	recall	F_1
Forward	86	85	86	85	77	71	74
Backward	87	87	87	87	78	78	78
Averaged	89	88	89	89	82	78	80

Table 3: Validation results (%) of sub-task A

5. Results

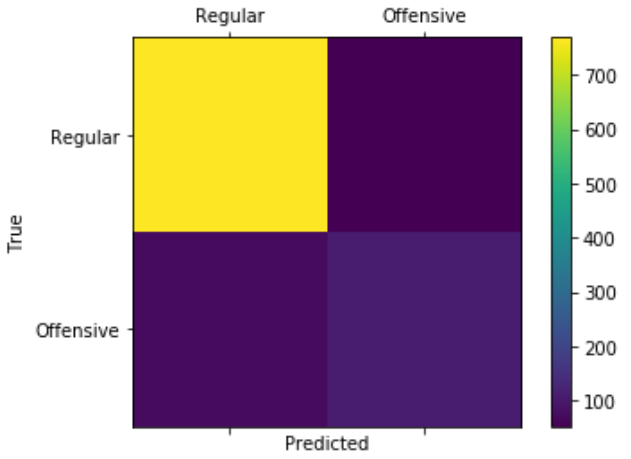
We report accuracy, weighted and macro F_1 as evaluation metrics for the validation set while we report accuracy and only the macro F_1 for the test set. F_1 is the harmonic mean of Precision (the ratio between the true positives and all the positive) and Recall (the ratio between the true positives and all the true). The macro version adds the metrics values of separate classes with equaly weights while the weighted version weights them by the ratio of class population. Recall that Both the language model and the classifier networks use AWD LSTM (Merity et al. (2017)).

5.1. Validation results

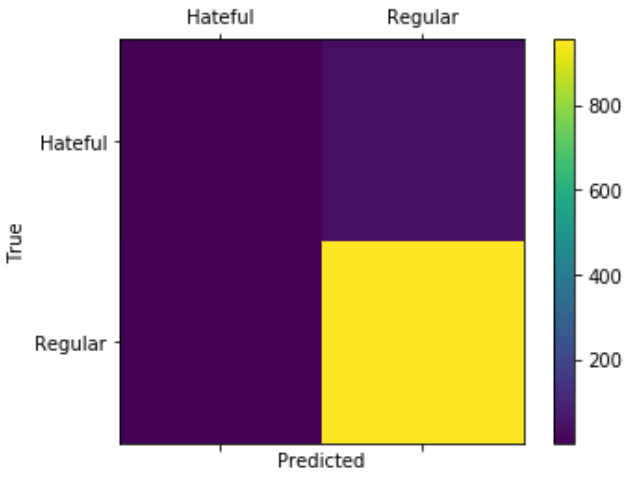
Table 3 presents validation results of sub task A while table 4 present task B. Since we have access to validation labels, we show the results of the forward, backward and averaged models. Since weighted measures favor majority classes (they aggregate using a weighted average), they are not very descriptive of the performance in case of imbalanced datasets where the minority class is important (like in our case). This can be seen from the tables. In terms of validation results, training two models instead of one and averaging results boosts the results in terms of macro F_1 in both sub tasks. The confusion matrix of the validation set is illustrated in figure 2.

5.2. Test results

Table 5 shows the test results of both sub tasks. Inspecting this table, the imbalance of the data-sets under question renders accuracy metric not descriptive of the performance. The very low population minor classes (offensive and hateful tweets in tasks A and B respectively) receive little attention from the trained classifier (relative to the majority class) since they are not as well represented in the training either. This is reflected in the low recall which drags F_1 down.



(a) Sub task A



(b) Sub task B

Figure 2: Confusion matrix

6. Conclusion and future work

We applied ULMFiT pre-trained on Arabic Wikipedia to approach the problem of classifying imbalanced Arabic data sets. Experiments on imbalanced data-sets of OffenseEval 2020 show that using two models (forward and backward) helps the final result in terms of macro F_1. Arabic-specific tokenization (e.g. based on Arabic morphological rules) may help building a better representation of Arabic text and hence improve performance, we leave this for future work. Another avenue for future work would be

Model	Accuracy	Weighted			Macro		
		Precision	Recall	F_1	Precision	Recall	F_1
Forward	96	94	96	94	75	57	60
Backward	96	95	96	95	77	58	61
Averaged	96	95	96	95	86	57	61

Table 4: Validation results (%) of sub-task B

Sub task	Accuracy	Macro		
		precision	recall	F_1
A	86	79	76	77
B	95	75	56	58

Table 5: Test results (%)

using generative models (e.g. language modelling) as a way of over-sampling the minor classes in imbalanced data sets. It can be experimented with by its own or associated with other (existing) techniques (e.g. random Ghazikhani et al. (2012) and SMOTE Chawla et al. (2002)).

7. Bibliographical References

References

Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003.

N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16: 321–357, 2002.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

A. Ghazikhani, H. S. Yazdi, and R. Monsefi. Class imbalance handling using wrapper-based random oversampling. In *20th Iranian Conference on Electrical Engineering (ICEE2012)*, pages 611–616. IEEE, 2012.

J. Howard and S. Ruder. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*, 2018.

A. Khooli. Applied data science. `https://github.com/abedkhooli/ds2`, 2019.

A. Mahendran and A. Vedaldi. Visualizing deep convolutional neural networks using natural pre-images. *International Journal of Computer Vision*, 120(3):233–255, 2016.

S. Merity, N. S. Keskar, and R. Socher. Regularizing and optimizing lstm language models. *arXiv preprint arXiv:1708.02182*, 2017.

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

J. H. Park and P. Fung. One-step and two-step classification for abusive language detection on twitter. *arXiv preprint arXiv:1706.01206*, 2017.

A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever. Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/researchcovers/languageunsupervised/language understanding paper. pdf*, 2018.

S. Ruder, M. E. Peters, S. Swayamdipta, and T. Wolf. Transfer learning in natural language processing. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Tutorials*, pages 15–18, 2019.

A. Sharif Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 806–813, 2014.

L. N. Smith. Cyclical learning rates for training neural networks. In *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 464–472. IEEE, 2017.

Z. Waseem and D. Hovy. Hateful symbols or hateful people? predictive features for hate speech detection on twitter. In *Proceedings of the NAACL student research workshop*, pages 88–93, 2016.

Z. Zhang, D. Robinson, and J. Tepper. Detecting hate speech on twitter using a convolution-gru based deep neural network. In *European semantic web conference*, pages 745–760. Springer, 2018.