SemEval-2019 Task 3 Sub-Task A: Emotion Classification with RNN and BERT

Iris Ma and Ines Pancorbo
Georgetown University
jm3309@georgetown.edu, ip221@georgetown.edu

Abstract

We present our results and findings of SemEval-2019 task 6 sub-task A. Task 6 was based on the Offensive Language Identification Dataset, which contains more than 14,000 English tweets and was itself divided into 3 sub-tasks (A, B, C). We experimented with GRU, LSTM, and BERT based neural networks to classify tweets, and propose BERT as our recommended approach, given its higher macro F1-score of 81.73 percent and accuracy of 86.18 percent. Given this F1-score, our approach is ranked between the 1st and 2nd place (out of 104) of the scoreboard of the competition.

1.Introduction

Twitter is usually treated as a platform for online debates, where individuals express their opinions and are therefore, often attacked for doing so. Twitter, as well as other platform providers, usually aim to remove or prevent these attacking posts. Doing so manually can be costly and time-consuming, so automatic detection is a nice alternative.

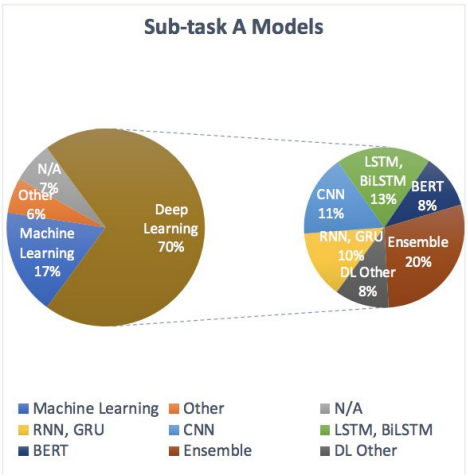
In this paper, we present our results for the *SemEval 2019 Task: Identifying and Categorizing Offensive Language in Social Media* on the Offensive Language Identification Dataset. This task was divided into 3 subtasks (A, B, C) and we focused on sub-task A, in which the goal was to classify tweets as offensive (OFF) or non-offensive (NOT) posts. Offensive posts include insults, threats, and posts containing any form of untargeted profanity. Each instance is assigned one of the following two labels.

- Not Offensive (NOT): Posts that do not contain offense or profanity;
- Offensive (OFF): Posts that contain any form of profanity or a targeted offense (insults, threats, and posts containing profane language or swear words).

The corpus provided by the organizers consists of 14,100 tweets in English. The data collection methods used to compile the dataset used in *OffensEval* is described in Zampieri et al. The 14,100 English tweets were divided into a training set of 13,249 tweets and a testing set of 860 tweets. See the table below for more details.

	Train	Test	Total
OFF	4,400	240	4,640
NOT	8,840	620	9,460
Total	13,240	860	14,100

The official evaluation measure for task 6 sub-task A was the macro F1 score. Teams in this competition used models that ranged from traditional machine learning, such as SVM or logistic regression, to deep learning, such as CNN, RNN, Bi-LSTM, attention-based models such as ELMo and BERT. Below is a pie chart summarizing models used.

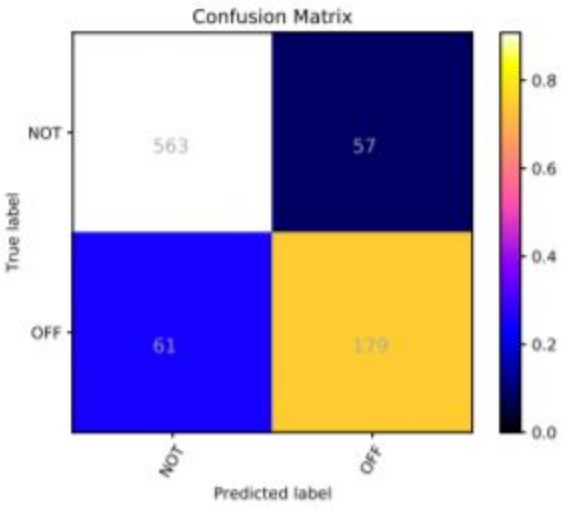


Source: Zampieri et al.

104 teams participated in Sub-Task A: Among the top ten teams, seven used BERT but with variations in hyperparameters and approaches to preprocessing. The top team used BERT-base-uncased with default-parameters, trained for 2 epochs, and used a maximum sentence length of 64. The top team achieved an F1 score of 82.9 percent. BERT seemed to perform well on this Sub-Task, as the top nonBERT model was ranked 6th and consisted of an ensemble of CNN, Bi-LSTM and Bi-GRU with Twitter word2wec embeddings. Find below, the scoreboard for Sub-Task A. Further, find below the confusion matrix for the top team (F1 score of 82.9 percent).

Sub-task A	
Team Ranks	**F1 Range**
1	0.829
2	0.815
3	0.814
4	0.808
5	0.807
6	0.806
7	0.804
8	0.803
9	0.802
CNN	**0.800**
10	0.798
11-12	.793-.794
13-23	.782-.789
24-27	.772-.779
28-31	.765-.768
32-40	.750-.759
BiLSTM	**0.750**
41-45	.740-.749
46-57	.730-.739
58-63	.721-.729
64-71	.713-.719
72-74	.704-.709
SVM	**0.690**
75-89	.619-.699
90-96	.500-.590
97-103	.422-.492
All NOT	**0.420**
All OFF	**0.220**
104	0.171

Source: Zampieri et al. 2019



Source: Liu et al. 2019

We mainly experimented with two different types of classifiers. (1) RNN models, making use of bidirectional GRU and LSTM units, due to their capability of sequential processing and ability to retain past information through past hidden states, and (2) a fine-tuned Bidirectional Encoder Representation from Transformer (BERT) (Devlin et al., 2018), therefore avoiding recursion and making use of an encoder/decoder. We ended up relying on BERT given that it reached the best macro F1 score of 81.73 percent when testing our respective models on the test dataset.

2.Approach Description

Sub-task A is a binary classification task. A tweet can be either offensive (OFF) or non-offensive (NOT). The model takes a tweet as input and predicts the corresponding label of that tweet. We split the given data into 80 percent for training purposes and 20 percent for validation purposes. For this task, we experimented with the following:

Preprocessing

We transformed the labels from "OFF" / "NOT " to "1" / " 0". Then we used a github-based emoji project (https://github.com/carpedm20/emoji), which mapped emoji unicode to an English phrase. Lastly, we removed URL and twitter references, as well as duplicate punctuation and spaces.

Pre-trained word embeddings

We experimented with pre-trained word embeddings. Instead of randomly initializing the word embeddings we initialized them with pre-trained vectors. We used "glove.6B.100d", i.e., used GloVe as the algorithm to calculate the vectors, an embedding size of 100 pre-trained on 6 billion tokens, as it resulted in the best macro F-1 score on the validation dataset.

2.1 RNN

2.1.1 Model Details

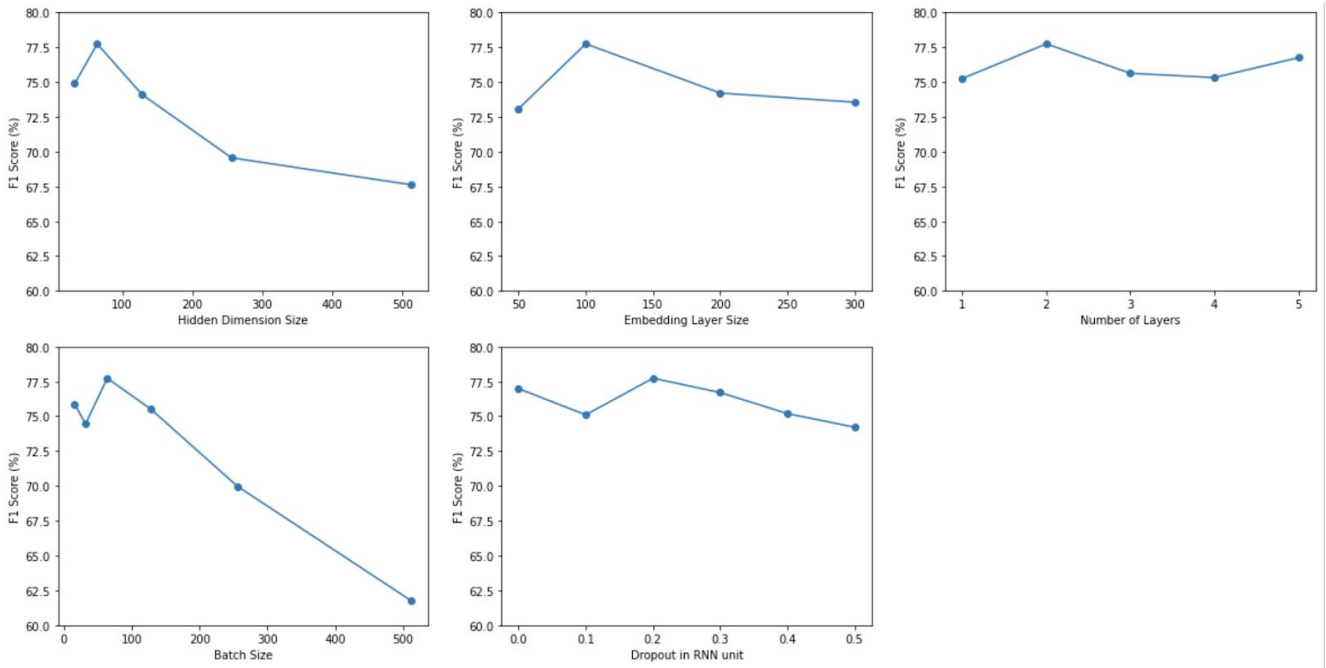
We used a GRU model as our second powerful baseline model to compare and report our results. The first layer of the GRU model is an embedding layer, initialized with the GloVe 100-dimensional embeddings. We used "rnn.pack_padded_sequence" on the result of the embedding layer so that our RNN model only processes the non-padded elements of our sequence. The next layer is a bi-directional GRU (hidden size equal to 64, number of layers in RNN unit equal to 2, and dropout in RNN unit equal to 0.2). The output of the GRU layer is a concatenation of the last hidden state from the last word of the rpost and the hidden state from the first word of the post. We then add a linear layer, which is then followed by a sigmoid layer (since we have a binary classification problem), which produces the final prediction.

We trained the GRU model using 3 epochs, batch size of 64, Binary Cross Entropy Loss as our loss function, and Adam as our optimizer.

We note that LSTM units were also considered, but they performed slightly worse than GRU units, when hyperparameter tuning on the validation set.

2.1.2 Sensitivity Analysis

We conducted sensitivity analyses on the GRU based model's hyperparameters: hidden size, embedding size, number of layers in Bi-GRU units, batch size, and dropout in Bi-GRU units. The results are depicted below. Our model is relatively robust to hyperparameters except, one could argue, to batch size, as we see the greatest fluctuations in F1 score on the test dataset.



2.2 BERT

2.2.1 Model Details

We trained a classifier by fine-tuning pre-trained BERT (huggingface:https://huggingface.co/transformers/index.html) with a linear layer for text sequence classification on top.

Every tweet is used as a text "sentence" of arbitrary length in the BERT model. We added tokens to the start and end of each sentence. The maximum length for our dataset is 115, thus we chose the maximum length to equal 128, which means the shorter sequences were padded. We tokenized with the BERT model to lower casing and performed punctuation splitting.

For this task, we fine-tuned the pre-trained BERT model and we chose the BertForSequence Classification for training. The documentation for the pre-trained bert model can be found here. (https://huggingface.co/transformers/v2.2.0/model_doc/bert.html).

We selected the "bert-base-uncased" which is a smaller version compared to the "bert-large-uncased". Also, the "bert-large-uncased" doesn't perform better compared to the result of "bert-base-uncased" on our test dataset. The "bert-base-uncased" contains 12 Transformer blocks, 12 self-attention heads, and a hidden dimension of 768, totaling 110 million parameters. The corpus of this pretrained model is the BookCorpus (800M words) and the

English Wikipedia (2,500M words). We used AdamW as the optimizer and Cross-Entropy Loss as the loss function.

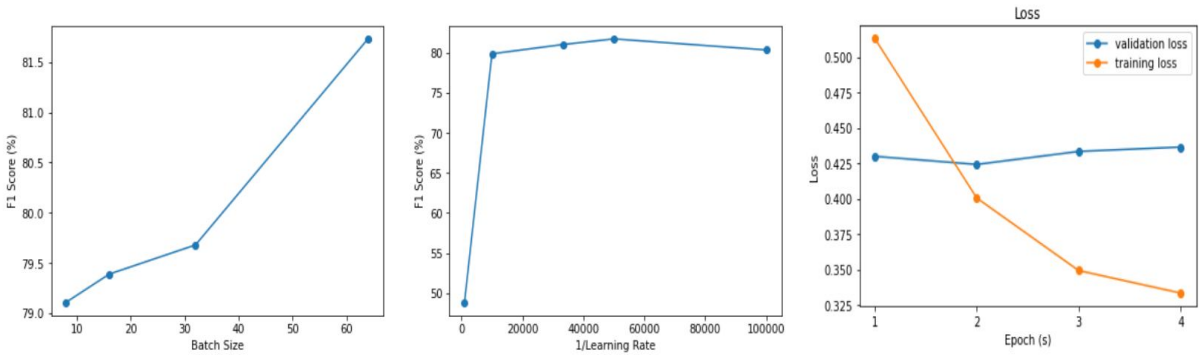
2.2.2 Implementation Details

The model was built in PyTorch and a GPU was used to train it. We experimented with different batch sizes including 8, 16, 32, 64 and 128, with different epochs including 0.9, 1, 2, 3 and 4, and with different learning rates for our AdamW optimizer including 3e-5, 2e-5, 1e-5, 1e-4, 1e-3, …, etc. Due to a better F1-score on the validation set, we relied on a batch-size of 64, 2 epochs, and a learning rate of 2e-5. We also experimented with the scheduler: we experimented with *get_constant_schedule_with_warmup*, *get_linear_schedule_with_warmup, get_cosine_schedule_with_warmup* and *get_cosine_with_hard_restarts_schedule_with_warmup.* Again, due to a better performance on the validation set, we relied on the *get_cosine_schedule_with_warmup*. Training in each epoch took approximately 6 minutes.

We also tried to add weight to the BERT loss function, however probably because the distribution of training dataset and testing dataset are different or our test dataset is not very imbalanced, the result doesn;t performs well. So we decided to not add the weight.

2.2.3 Sensitivity Analysis

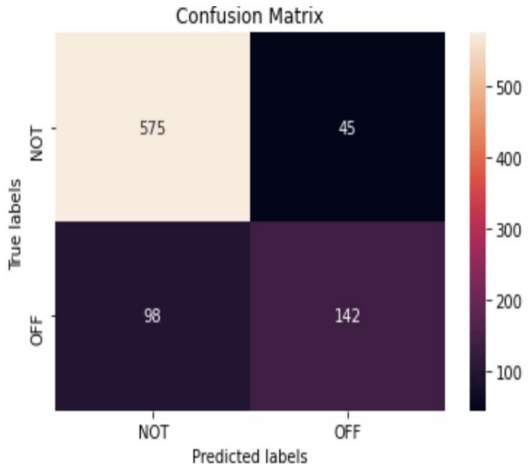
We conducted sensitivity analyses on our model hyper-parameters to understand the robustness of our model. The results are shown in the graphs below.



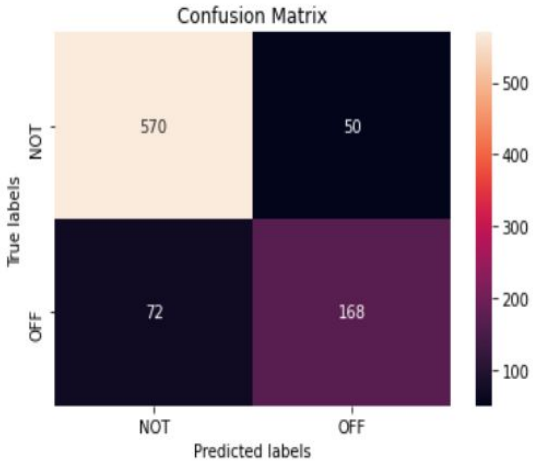
3. Result Analysis

The results of our respective models on the test set is shown in the table below. As can be seen, BERT outperformed the Bidirectional GRU in terms of F1 and accuracy, with an F1 score of 81.85percent and an accuracy of 85.81percent. To analyze the correct label of a tweet, we also show the confusion matrix which shows correct class predictions.

System	F1 (macro)	Accuracy
Bi-GRU	77.73%	83.37%
BERT	**81.85%**	**85.81%**



Confusion Matrix for Bi-GRU



Confusion Matrix for BERT

Other than the results displayed above, we also present the results per class in the table below. The results in the table below show how well our models performed on each class label (OFF and NOT).

	OFF			NOT		
	Precision	Recall	F1	Precision	Recall	F1
Bi-GRU	75.94%	59.17%	66.51%	85.44%	92.74%	88.94%
BERT	77.06%	70.00%	73.36%	88.79%	91.94%	90.33%

From the above table we can see that BERT performed better on both classes with F1-score of X and Y respectively for OFF and NOT tweets. From the table one can also observe that offensive tweets are relatively harder to classify.

4. Conclusion

In this paper we addressed the challenge of being able to identify offensive tweets. We presented our approach on categorizing offensive tweets: We experimented with RNN and BERT, showing how applying BERT leads to a better classification of the text. Overall, our approach provides an efficient way of text classification in social media, in particular, Twitter.

References

Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805 (2018).

Liu, Ping, Wen Li, and Liang Zou. "NULI at SemEval-2019 Task 6: transfer learning for offensive language detection using bidirectional transformers." *Proceedings of the 13th International Workshop on Semantic Evaluation*. 2019.

Zampieri, Marcos, et al. “Semeval-2019 task 6: Identifying and categorizing offensive language in social media (offenseval).” arXiv preprint arXiv:1903.08983 (2019).