Temporal Ensembles of Fine-Tuned BERT Models for Offensive Language Identification

Jennifer Kadowaki (jkadowaki@email.arizona.edu)

Task 6a: Identifying Offensive Language

With the advent of major social media platforms, growing concerns surround online user safety and experience. We participated in the SemEval-2019's OffensEval shared task, which uses the Offensive Language Identification Dataset (OLID; Zampieri et al., 2019) to identify offensive and abusive language in Tweets.

Approach: Temporal Ensembles of Fine-Tuned BERT Models

Due to the speed of fine-tuning and the nature of the classification task, we used the BERT-based model (Devlin et al., 2018) for a single sentence classification task (Figure 1). We opted to use the cased version due to the presence of capital letters in our dataset. We did not perform any data preprocessing or cleaning with the exception of making the data compatible with BERT. Table 1 lists the hyper-parameters for our setup. We performed 10-fold cross-validation to check the consistency of our results across repeated experiments. Due to the memory-intensiveness of fine-tuning, we fine-tuned BERT on a GPU node on Ocelote using a Singularity container with Tensorflow and CUDA. For each fold, we fine-tuned the model for 100 epochs, saving the model and dev set predictions after each training epoch.

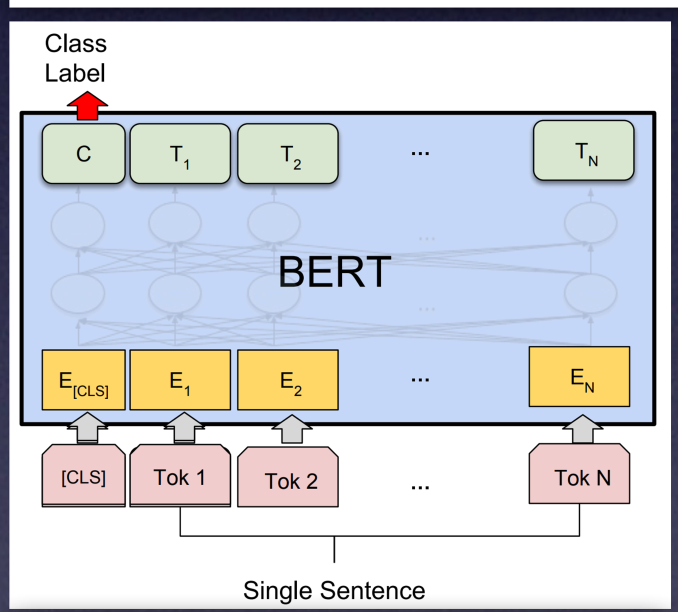


Figure 1: Single sentence classification task with BERT (Image Credit: Devlin et al., 2018)

Hyper-Parameters	
Task	CoLA
Bert Model	BERT-Base (Cased)
Max. Sequence Length	128
Training Batch Size	32
Learning Rate	2×10^{-5}
Training Epochs	100

Table 1: Hyper-parameters of BERT.

Ensembling is a common technique used to improve the accuracy of prediction. An ensemble of models will separately make predictions on the test set. In its simplest form, the overall prediction is based on the majority vote for classification tasks and on weighted averages for regression tasks. In our experiments, we consider each model saved after every training epoch during the fine-tuning. We use both majority voting for binary class labels and averaging for BERT's predicted probability for each label. To convert between the predicted probability to a class label, we find an optimal threshold.

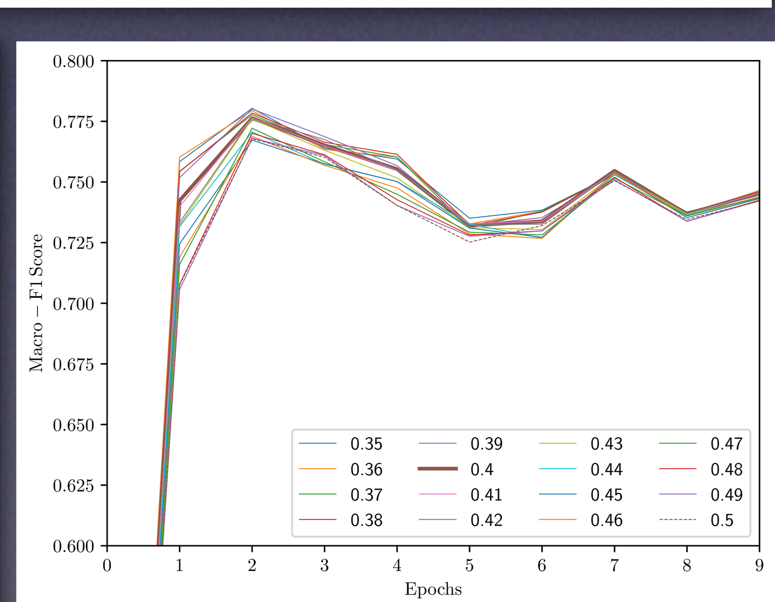


Figure 2: We analyze how different thresholds affect our Macro-F1 score across all folds. We selected a global threshold value of 0.4.

Ensemble #1: Best Performers

Experiment 1: *Do certain epochs yield consistently better predictions? If so, how much does the ensemble prediction improve the macro-F1 score?*
We computed the macro-F1 score for every prediction (Figure 3). We counted the number of times each epoch yielded a prediction rated within the top 10% macro-F1 scores in its fold (Figure 4). Epochs 2, 3, and 4 consistently yielded the best predictions. Their ensemble over BERT probabilities yielded an average macro-F1 score of 0.7692 +/- 0.00014 across all folds. Figure 5c quantifies the small F1 score improvement.

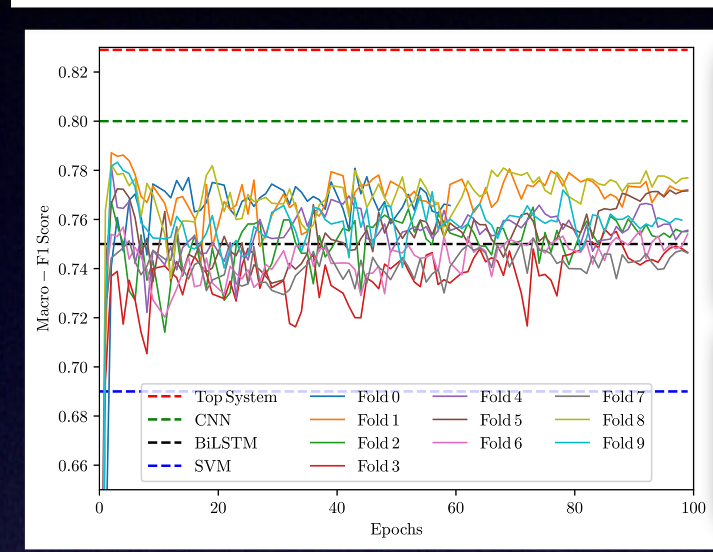
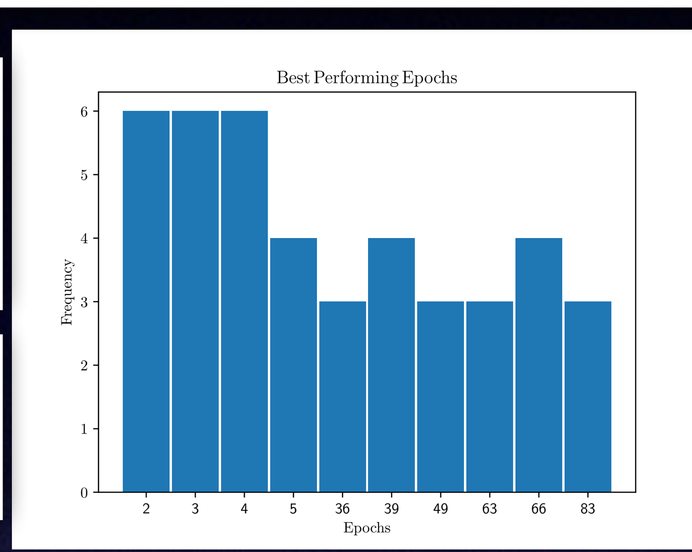


Figure 3: Macro-F1 Scores for all epochs for each fold. Dashed lines denote top performing system, CNN, BiLSTM, and SVM baselines in Task 6a.

Figure 4: The number of folds each epoch yielded a prediction in the top 10% F1-scores.



Ensemble #2: Consecutive Epochs

Experiment #2: *Is there a good basis for ensembling the last n epochs? Is early-stopping a good metric for selecting n consecutive epochs?*
In both cases (with early-stopping and without), we are interested in ensembling the predictions generated by consecutive epochs. We perform 1D-convolution over the results of continuous epochs with a uniform kernel of varying sizes (n=2,3,5,9 epochs). We observe the best results using a window size of n=3 over epochs 1, 2, and 3. However, increasing the window size appears to decrease the performance gain at epochs 1-3.

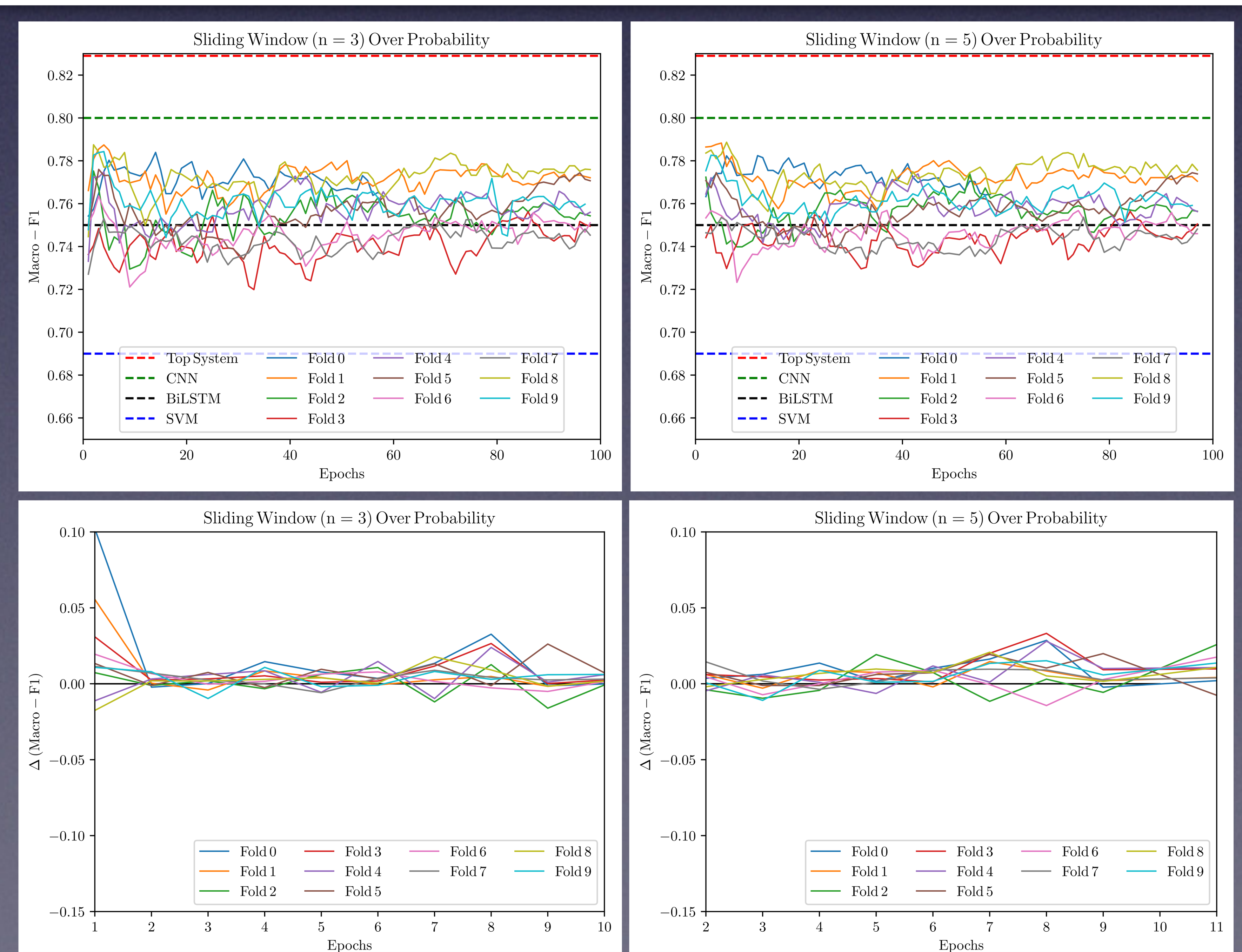


Figure 5: We convolved the predictions of consecutive epochs with epoch window sizes of 2,3,5,9. Ensemble of 3 models yields the highest performance gain.

Ensemble #3: Minimum Redundancy

Experiment 3: *Do ensembles comprised of models with minimal prediction redundancy perform better?*
We quantify redundancy with the average pairwise overlap metric (Yadav et al., 2019). We computed the average pairwise overlap between every pair and triplet of epochs in each fold. In both the pairs and triplets, no correlation was observed between the macro-F1 score and the overlap metric. We conclude that the overlap metric alone is not a sufficient indicator for model selection.

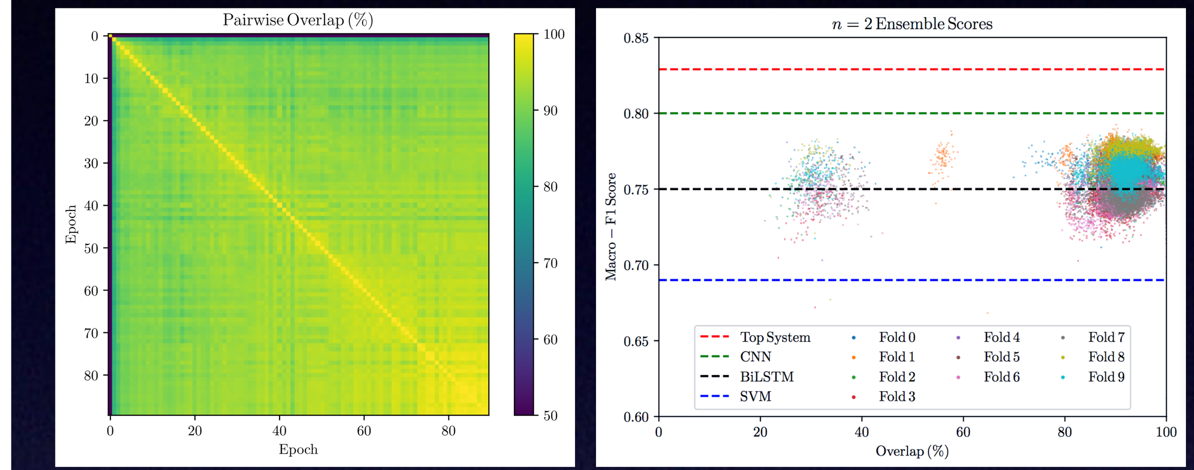


Figure 6: We computed the pairwise overlap for ever pair of epochs and compared it to the macro-F1 scores of the ensemble prediction. We do not observe a correlation between the overlap and performance. While only the the n=2 ensemble results are shown, the results for n=3 appear nearly identical.

Results

Ensemble	
Epochs	2,3,4
Macro-F1 Score	0.800512
Accuracy	0.843023
Weighted-F1 Score	0.841203

Table 1: Final Results of Task 6a.

We fine-tuned the final model on all the annotated data. We used an ensemble of 3 epochs as guided by the first 2 experiments. Due to Epoch 1's performance volatility between each fold and sensitivity to the selected threshold (as seen in Figure 2), we opted to use Epochs 2, 3, and 4 (Experiment #1). While we note the drastic improvement in macro-F1 performance, we caution readers that some of the performance gain is attributed to training on the full dataset. Additionally, other ensembles may result in higher performance gains.

Acknowledgements

The author would like to thank Vikas Yadav and Professor Steven Bethard for productive discussions regarding model architecture and metrics for selecting epochs. Additionally, the author would like to Manoj Gopale for his instrumental help in modifying BERT's default Tensorflow classifier.

Future Work

(1) We will implement text preprocessing prior to fine-tuning BERT models. The top performing group used the uncased BERT-base model with highly cleaned data. (2) We can apply a unique threshold to each epoch (in contrast to applying a global threshold). (3) Ensemble #1: While we've obtained a recommendation for epoch selection, more rigorous statistics are needed to quantify the quality of the recommendation. We can vary the number of epochs in the ensemble and alter the percentage of top performing epochs to combat the heuristics nature of the algorithm. (4) Ensemble #2: We can experiment with a non-uniformly weighted kernel. (5) Ensemble #3: Alone, the overlap metric cannot distinguish good ensembles from bad. However we can test for performance gains when the overlap metric is coupled with other ensembling metrics (e.g., best performing epochs).

References

Devlin et al., 2018
Yadav et al., *in prep.*
Zampieri et al., 2019a
Zampieri et al., 2019b