The Quest for Winning Tickets

Norica Băcuieți
Department of Computer Science, EPFL, Switzerland

1 Introduction

Over the last decade, an increasing number of people started expressing their feelings and views through tweets, in particular, (Weller et al., 2013). As the trend seems to show no signs of slowing down, the interest in extracting information from this data has grown as well, thus making sentiment analysis one of the fastest-growing research fields (Mäntylä et al., 2016). In general, sentiment analysis tackles the problem of classifying a piece of text into some predefined sentiment classes. For this, deep neural networks such as BERT (Devlin et al., 2019) and RoBERTa (Liu et al., 2019) have been proposed, but despite them achieving a remarkable performance, they are considerably slow to train due to their large network size. To counter this, a BERT neural network reduced by 40% called DistilBERT (Sanh et al., 2020) was recently introduced. However, despite it being faster, we wondered whether this is the smallest one can go before the performance starts to significantly decrease. This research question is the main focus of this paper, in addition to the study of result explainability and improving the classification performance. Concretely, focusing on classifying tweets into positive or negative tweets, our contributions are:

- We evaluate the Lottery Ticket Hypothesis on the BERT neural network, finding that up to 90% of this network's weights can be pruned, while the accuracy remains within 1 percentage point.
- We study the feature importances using the state-of-the-art Local Interpretable Model-agnostic Explanations (LIME) (Ribeiro et al., 2016a) explainer, finding that BERT struggles with negative, sarcastic, and half-positive-half-negative sounding tweets and that a stronger explainer is needed.
- We construct an ensemble using the above-mentioned 3 neural networks aiming to improve the true negative rate, discovering an interesting caveat that opens up further research avenues.

2 Related work

In 2019, Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2019) was for the first time introduced. Being designed to pre-train deep bidirectional representations from an unlabeled text by jointly conditioning on both left and right context in all layers, the pre-trained BERT model could be fine-tuned with just one additional output layer, thus obtaining state-of-the-art results on various datasets (Devlin et al., 2019). However, researchers quickly realized that the hyperparameter choices of the BERT model have a significant impact on the final classification results and that BERT was significantly undertrained, giving rise to a Robustly Optimized BERT pre-training Approach (RoBERTa) (Liu et al., 2019) that addressed these problems. To give a bit more insight, RoBERTa was trained with dynamic masking (instead of static masking like BERT) to enhance accuracy on larger datasets, full sentences without Next Sentence Prediction (NSP) loss to improve downstream task performance, a larger byte-level Byte-Pair Encoding (BPE) which makes it possible to learn a subword vocabulary of modest size, and large mini-batches to increase speed and end-task accuracy (Liu et al., 2019). Nevertheless, as both BERT and RoBERTa are large models, and, therefore, have a very time-consuming training process, researchers started looking into reducing BERT's size. They concluded their research with the DistilBERT (Sanh et al., 2020) model, which is a smaller, faster, cheaper, and lighter version of BERT. Concretely, they leveraged knowledge distillation dur-

ing the pre-training phase and showed that it is possible to reduce the size of the BERT model by 40%, while retaining 97% of its language understanding capabilities on top of being 60% faster. Thus, knowing that the BERT network can be pruned without significantly impacting its performance, since Sanh et al. (2020) did not go beyond the 40% pruning mark, in this work we will explore that, presenting our findings.

3 Structure of the paper

We start by describing the original dataset and the preprocessing done before training the baseline models in Section 4. Then, in Section 5, we present the baselines' performance on the dataset, we evaluate the Lottery Ticket Hypothesis, we perform feature visualization using LIME, and we try to improve the performance using an ensemble. For each of these, we will present the results, draw conclusions, and offer intuitions when needed. Finally, we conclude with an overview of the results and findings presented in this paper and offer suggestions for further research.

For details regarding the setup, as well as all the notebooks we used to obtain these results, please refer to the report's repository[1].

4 Data preprocessing

The original Kaggle dataset consists of 2.5 million lower-cased English tweets with their associated sentiment labeling. The tweets were provided as scraped from the platform, with usernames and URLs replaced by special tokens. However, before we can train our models on this dataset, we performed the following preprocessing:

Text cleaning: We begin by restricting character repetitions to 2 (saaaaad → saad) and replacing slang words (2mrow → tomorrow) and emoticons with their respective meaning. Then, we use the social tokenizer ekphrasis (Baziotis et al., 2017) to unfold hashtags and remove other unstructured expressions left in the text, such as emphasis, repetitions, and contractions. Moreover, using a corpus of common misspellings (Mitton) and the grammar library JamSpell (jam), we perform spelling corrections. Lastly, stop-words are removed as they add no semantic information, and the tweets are lemmatized to attain better performance.

[1] https://github.com/NoricaBacuieti/TheQuestForWinningTickets

Removing duplicates: Repeated tweets are removed from the original dataset, as well as the empty tweets that resulted after applying the preprocessing step mentioned above. This left us with 1123904 positive and 1138848 negative tweets.

Negation handling: Attention-based models such as BERT struggle in practice with handling negations and double negations (Carranza Tejada et al., 2021), in part because of the way word embedding is performed. Optimizing for this is an open problem with high practical applicability as most BERT misclassifications are caused by mishandling negations (Carranza Tejada et al., 2021). To date, the best approach to handling negations is by explicitly training on each sentence and its negation at the same time. This approach is not feasible within the scope of this project, since it would at least double the dataset and hence, the training time. Thus, as an experiment, we explicitly glued the negation to the word it precedes (I don't like movies → I do not_like movies). The intuition behind this step is that our model will be forced to consider negated words as specifically distinct tokens, thus, hopefully achieving a better classification accuracy.

5 Models and methods

5.1 Baselines

The baseline models were trained on 90% of the preprocessed dataset, validated on 5%, and evaluated on the remaining 5% of the preprocessed dataset. This process was repeated 3 times, the results were averaged, and they can be seen in Table 1.

Model	Accuracy
BERT	$0.843 \pm 1.23 \times 10^{-3}$
RoBERTa	$0.829 \pm 2.43 \times 10^{-3}$
DistilBERT	$0.840 \pm 1.15 \times 10^{-3}$

Table 1: Accuracies of the three baseline models.

As expected, BERT and DistilBERT have a similar performance, with DistilBERT being slightly worse than BERT; however, RoBERTa did not surpass BERT. This might be because RoBERTa has had more training and a better parameter tuning on the corpus than BERT, a corpus that is different from our training data. Perhaps the specific things it learned to look for that increased its performance are not present in our training data, which has many

sentences that are not written most understandably - even for a human.

We also noticed that the True Negative Rate (TNR) is consistently lower than the True Positive Rate (TPR) - TPR being around 86/87% and TNR being around 80/81%. This is again consistent with previous findings by Carranza Tejada et al. (2021), who also noted that BERT does not manage to correctly deal with negations. As mentioned in Section 4, we performed a cheaper form of negation handling in the form of gluing the negation to the word it precedes to see if this would improve the accuracy of our models, but the improvement was minor - below 1%.

Nevertheless, confirming that DistilBERT - a BERT model reduced by 40% - performs almost as good as BERT we started our first experiment where we wanted to see how much one could prune BERT before its accuracy would degrade significantly.

5.2 The Lottery Ticket Hypothesis

The *Lottery Ticket Hypothesis* (LTH) was first proposed by Frankle and Carbin (2018) as a result of finding that appropriately initialized *pruned* networks are capable of training effectively while achieving a comparable accuracy to the original network in a similar number of training epochs. It reads as follows:

> *A randomly initialized dense neural network contains a subnetwork that is initialized such that - when trained in isolation - it can match the test accuracy of the original network after training for at most the same number of iterations.*

Now, as mentioned, these subnetworks are obtained by pruning, and in the following subsections, two pruning strategies will be put under test for the LTH: *one-shot pruning* and *iterative pruning*.

The winning tickets: As defined in (Frankle and Carbin, 2018), after having pruned the trained baseline network of the smallest-magnitude weights, a *winning ticket* is a subnetwork that, when trained in isolation after having had the remaining weights reinitialized with the weights of the baseline network *prior to training*, will provide classification accuracy equivalent or superior to the baseline network.

One-shot pruning: In one-shot pruning, the baseline network is trained once, $p\%$ of the weights are pruned, and then the remaining weights are reinitialized to the weights of the baseline network prior to training. Algorithm 1 describes one-shot pruning.

Algorithm 1 Definition of one-shot pruning

Input: Baseline neural network f, Dataset x, Random initial weights θ_0, Pruning ratio $p\%$
Output: Pruned trained network $f_{p\%}$
$\theta_j \leftarrow$ Train model $f(x, \theta_0)$ for j epochs
$m \leftarrow$ Prune $p\%$ of the smallest weights from θ_j
$f_{p\%} \leftarrow$ Train model $f(x, \theta_0 \wedge m)$ for j epochs
return $f_{p\%}$

Having this pruning method, the process will be repeated for several values of $p\%$ to see the possible changes in the performance of the classifier.

Iterative pruning: In iterative pruning, we again start from a baseline network that is trained once, but unlike in one-shot pruning where for each time we repeat the process with a different value of $p\%$ we start from the same pre-trained weights θ_0; now, at each pruning trial $i \in \{1, t\}$, $p\%$ of the *remaining* weights are pruned. Algorithm 2 describes iterative pruning.

Algorithm 2 Definition of iterative pruning

Input: Baseline neural network f, Dataset x, Random initial weights θ_0, Pruning ratio $p\%$ per trial, Pruning trials t
Output: Pruned trained network f_i
$\theta_j \leftarrow$ Train model $f(x, \theta_0)$ for j epochs
$m \leftarrow 0$
for i= 1 to t **do**
  $m' \leftarrow$ Prune $p\%$ of the smallest weights from θ_j
  $\theta_j \leftarrow \theta_j \wedge m'$
  $m \leftarrow m \vee m'$
  $f_i \leftarrow$ Train model $f(x, \theta_0 \wedge m)$ for j epochs
end for
return f_i with highest accuracy on the test set

Results: The results that were obtained by evaluating the LTH for the BERT network can be seen in Figure 1. This time, due to time constraints, the model was trained on 10% of the preprocessed dataset, validated on 5%, and evaluated on the remaining 5% of the preprocessed dataset. Experiments with both one-shot pruning (depicted in ***red***) as well as iterative pruning (depicted in ***blue***) were conducted and compared to the results obtained for

the (unpruned) baseline model (depicted in ***black***). The experiment was run 3 times per pruning ratio for each pruning method, the results were averaged, and the minimum and maximum at each pruning trial were indicated.

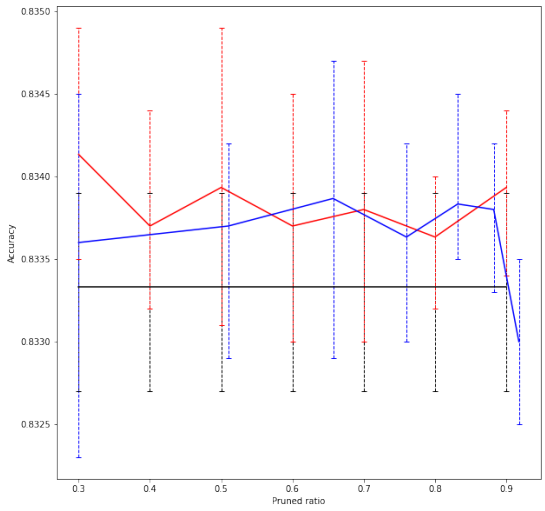

Figure 1: The accuracy of the BERT network after pruning $p\%$ of the network's weights.

For both pruning methods, there were 7 pruning trials, where for one-shot pruning 30%, 40%, 50%, 60%, 70%, 80%, and 90%, respectively, of the network was pruned, and for iterative pruning 30% of the remaining network's weights were pruned per trial.

Discussion: The reason behind applying the LTH was to see whether:

1. Some subnetworks perform similar to the baseline network, and how much the performance decreases as the network becomes more sparse to get an idea of the trade-off between the network's size and its performance.

2. There are winning tickets and whether their performance is significantly better than the baseline network's.

3. Similar conclusions to the ones in (Frankle and Carbin, 2018) can be drawn.

Looking at the results, two things can be observed right away: that up to 90% of BERT's weights can be pruned without losing (on average) performance, and that there are winning tickets that even slightly outperform (on average) the baseline network. Therefore, it can be anew empirically confirmed that the LTH does indeed find subnetworks that, when reinitialized with the weights of the baseline network *prior to training*, will provide classification accuracy equivalent or superior to the baseline network.

Then, looking at the findings of Frankle and Carbin (2018), they have found that iterative pruning finds winning tickets that match the accuracy of the baseline network at smaller network sizes than does one-shot pruning, and that the winning tickets are 10% to 20% (or less) of the baseline network's size. These findings could not be entirely confirmed by the results above. Regarding their first finding, iterative pruning does not seem to be superior to one-shot regarding finding winning tickets that match the accuracy of the baseline network at smaller network sizes. Looking at the results, iterative pruning is either outperformed by one-shot pruning or is just barely outperforming one-shot pruning. Perhaps more trials per pruning ratio are needed to be able to be firm in this sense. However, since this process is costly, the conclusion is left at both pruning methods performing similarly with respect to finding winning tickets at smaller network sizes. Then, regarding their second finding, the results presented above show that winning tickets can be found, on average, at lower, as well as at higher pruning ratios, and, in general, at every pruning ratio by both pruning methods.

Answering our research question in the affirmative, we have shown that one can indeed go significantly (weight-wise) smaller than DistilBERT without compromising the network's performance, and with this, we move on to the next section.

5.3 Feature visualization using LIME

In this subsection, we will look closer at the feature importances and classification explanations using one of the state-of-the-art explanation techniques called Local Interpretable Model-agnostic Explanations (LIME) (Ribeiro et al., 2016b). In short, according to Ribeiro et al. (2016b), LIME explains the predictions of any classifier in an interpretable and faithful manner by learning an interpretable model locally around the prediction. In other words, it means that an explanation corresponds to the model's behavior in the vicinity of the predicted instance. Moreover, LIME is model-agnostic, meaning that it treats the classifier as a black-box, which removes model-specific biases that might be introduced, in addition to making it possible to be deployed for every classifier.

Knowing this, the feature importances were com-

puted using LIME, for each computing the explanations for the 2 best and worst predictions belonging to class 1 (positive tweet) and class 0 (negative tweet) among 20000 tweets. These can be seen in Appendix A.

From the obtained figures, we see that the importance of each feature is insignificant, even suggesting that removing any/all of the input words (of a tweet) will insignificantly affect the performance of the classification. Since this happened for a fairly good classifier, this suggests that the *linear* explanation model LIME uses will not be able explain BERT's behavior as there might be no linear boundary to begin with.

Therefore, due to LIME's struggle to correctly assign the feature's importances, it was not strange to see that certain positive parts of the tweets are assigned as contributing towards a negative sentiment and vice versa. In addition, from Figures 4 and 5, BERT seems to be struggling with sarcastic tweets and tweets that contain one positive and one negative sounding half, confirming prior findings (Carranza Tejada et al., 2021).

Nevertheless, from all the figures, an overall observation we could draw is that BERT deals better with positive tweets than negative ones, which we also saw previously from the true positive and true negative rates. Looking closer at the feature importances, we conclude that our approach needs to go in the direction of Carranza Tejada et al. (2021). Gluing the negations to next word seems indeed to not be sufficient on its own; however, if there were more training data for each glued negation so that the model could see various contexts in which it can occur, we theorize that a better performance could be achieved, even better than the one in (Carranza Tejada et al., 2021). We suspect that gluing the negation to the word it precedes, in addition to training on both the original tweet and its negation, would eliminate the burden of the model learning how to integrate the negation on its own, thus allowing the model to concentrate more on the context of the glued negation, which is indeed the main focus.

Given the results, we conclude that a stronger non-linear explainer is needed to gain more insight into the classification result, as well as further experiments need to be conducted to test whether our approach combined with that in (Carranza Tejada et al., 2021) will indeed attain a better performance.

5.4 Ensemble learning

Ensemble learning (Sollich and Krogh, 1995) is a technique of combining the output of a finite set of predictors - independently trained on the same task - in a weighted or unweighted voting (Dietterich, 2000) manner. In practice, it has been shown that model ensembles often perform much better than their constituent classifiers. The theoretical reasoning for such a result is that ensembling multiple models with different hypotheses about a task can help minimize the statistical, computational and representational biases of a single, accurate classifier (Dietterich, 2000).

The most common approaches (Maclin and Opitz, 1999) to perform ensembles are bagging ("Bootstrap AGGregatING") (Breiman, 1996), boosting, and stacking. Bagging seeks to find a set of good candidate algorithms by fitting the same model on varied, distinct subsets of the training data and merging the resulting decision via simple statistics (Breiman, 1996), e.g., majority voting. Boosting is an iterative process used to debias a simple learning algorithm (Maclin and Opitz, 1999): at each step, a new model instance is trained with adapted input data (gradient boosting (Friedman, 1999)) or weights (adaptive boosting (Freund and Schapire, 1997)) to emphasize the training instances that previous models misclassified. Finally, stacking is a procedure similar to bagging, where the statistical choice is replaced with a meta-model that learns to optimize its decision based on the individual predictions of the aggregated models (Pavlyshenko, 2018).

Using stacking - since we have homogeneous models, trained on 90% of the data -, we attempted to combine the outputs of BERT, DistilBERT, and RoBERTa with a logistic regression meta-classifier. Each of the three models was initially trained as described in Subsection 5.1. Then, the meta-model was trained based on the predictions of these models on the validation set (5%), and its performance was evaluated on the test set (5%). The results can be seen in Table 2.

Model	Accuracy	TPR	TNR
BERT-Ensamble	80.03	71.846	88.706

Table 2: Performance of the ensemble model.

We observe that the resulting ensemble obtains a worse accuracy than the baseline models. According to Dietterich (2000), this suggests a lack

of diversity in the classification errors of the base models. Since their errors are correlated due to training data and architecture overlap, the meta-learner reports erroneous results when any of its base models produces a mistake (Dietterich, 2000). With this, the switch in the TPR to TNR ratio can be explained as well. We theorize that, since the TNRs were low for all three baselines, the resulting ensemble achieved initially a significantly worse TNR compared to the baselines. Similarly, we theorize that the TPR of the ensemble has decreased initially even more compared to the TNR since one of the baseline TPRs was significantly worse than the other ones. However, as both of these values might have been below 50%, the ensemble might have then switched the classification by assigning the opposite class to the class it initially established, thus inverting the labels, which resulted in a better TNR.

6 Conclusion and future work

In this paper, we studied whether the BERT network can be (weight-wise) pruned beyond the 40% mark of the DistilBERT. To this end, we evaluated the Lottery Ticket Hypothesis for the BERT model, finding that one could prune even up to 90% of the model's weights without significantly compromising the performance, thus empirically confirming the hypothesis anew. We then moved on to studying the feature importances and classification explainations using one of the state-of-the-art explanation techniques called LIME, and concluded with an ensemble of the three baseline models aimed at improving the classification accuracy. These last two experiments opened up avenues for further research such as deploying a stronger non-linear explainer, combining our glued negations approach with the training strategy mentioned in (Carranza Tejada et al., 2021) as we suspect that it would lead to an even higher classification accuracy, and constructing an ensemble with more diverse models with respect to their classification errors (Dietterich, 2000).

References

Jamspell: Modern spell checking library.

Christos Baziotis, Nikos Pelekis, and Christos Doulkeridis. 2017. Datastories at semeval-2017 task 4: Deep lstm with attention for message-level and topic-based sentiment analysis. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 747–754, Vancouver, Canada. Association for Computational Linguistics.

Leo Breiman. 1996. Bagging predictors. *Machine Learning*, 24(2):123–140.

Giorgia Nidia Carranza Tejada, Jan Scholtes, and Gerasimos Spanakis. 2021. A study of bert's processing of negations to determine sentiment. In *Proceedings of BNAIC/BeneLearn 2021*, pages 47–59. 33rd Benelux Conference on Artificial Intelligence and 30th Belgian-Dutch Conference on Machine Learning, BNAIC/BeneLearn 2021 ; Conference date: 10-11-2021 Through 12-11-2021.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Thomas G. Dietterich. 2000. Ensemble methods in machine learning. In *Multiple Classifier Systems*, pages 1–15, Berlin, Heidelberg. Springer Berlin Heidelberg.

Jonathan Frankle and Michael Carbin. 2018. The lottery ticket hypothesis: Training pruned neural networks. *CoRR*, abs/1803.03635.

Yoav Freund and Robert E Schapire. 1997. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139.

Jerome Friedman. 1999. Stochastic Gradient Boosting. Technical report, Stanford University.

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

R. Maclin and D. Opitz. 1999. Popular ensemble methods: An empirical study. *Journal Of Artificial Intelligence Research Volume 11*, pages 169–198.

Mika Viking Mäntylä, Daniel Graziotin, and Miikka Kuutila. 2016. The evolution of sentiment analysis - A review of research topics, venues, and top cited papers. *CoRR*, abs/1612.01556.

Roget Mitton. [link].

Bohdan Pavlyshenko. 2018. Using stacking approaches for machine learning models. pages 255–258.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016a. "why should i trust you?": Explaining the predictions of any classifier.

Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016b. "why should I trust you?": Explaining the predictions of any classifier. *CoRR*, abs/1602.04938.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2020. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter.

Peter Sollich and Anders Krogh. 1995. Learning with ensembles: How overfitting can be useful. In *Advances in Neural Information Processing Systems*, volume 8. MIT Press.

Katrin Weller, Axel Bruns, Jean Burgess, Merja Mahrt, and Cornelius Puschmann. 2013. *Twitter and society*. Peter Lang New York.

A Feature visualization using LIME

A.1 The best predictions for Class 1 - positive tweets

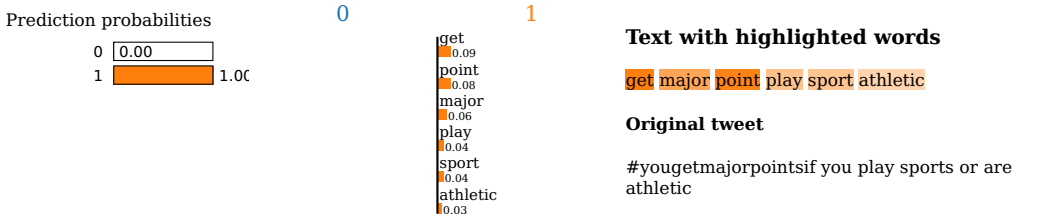

Figure 2: The best (most accurate) prediction for a positive tweet among 20000 tweets.

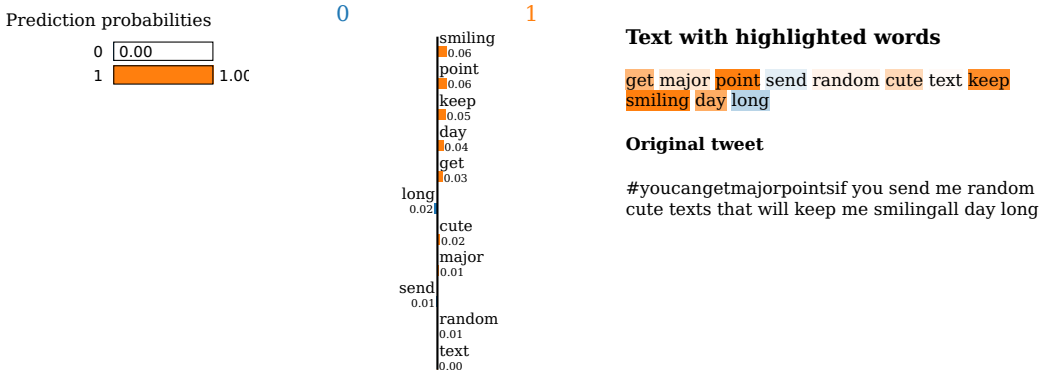

Figure 3: The second-best (most accurate) prediction for a positive tweet among 20000 tweets.

A.2 The best predictions for Class 0 - negative tweets

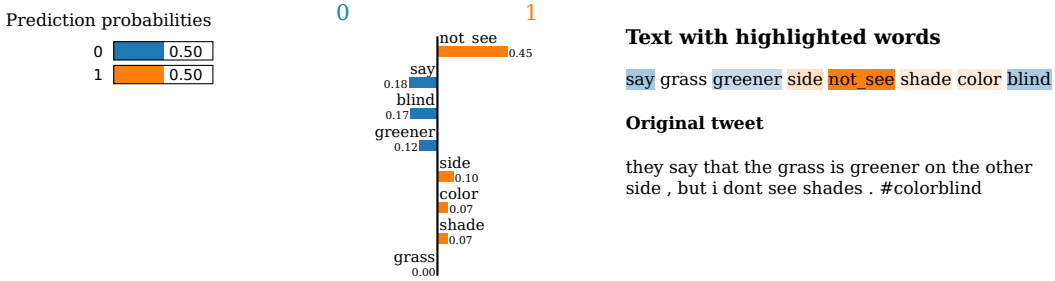

Figure 4: The best (most accurate) prediction for a negative tweet among 20000 tweets.

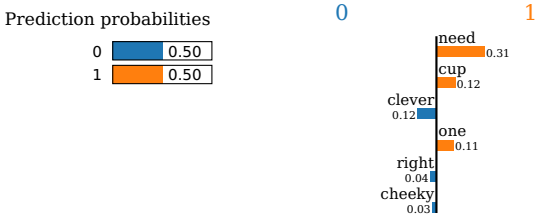

Figure 5: The second-best (most accurate) prediction for a negative tweet among 20000 tweets.

A.3 The worst predictions for Class 1 - positive tweets

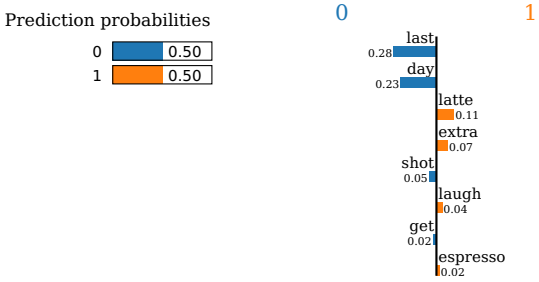

Figure 6: The worst (most inaccurate) prediction for a positive tweet among 20000 tweets.

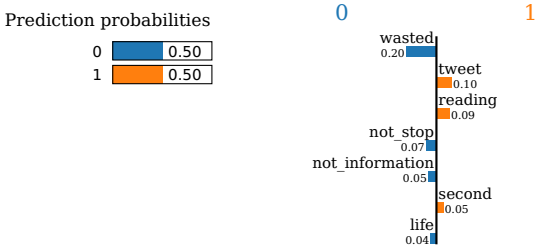

Figure 7: The second-worst (most inaccurate) prediction for a positive tweet among 20000 tweets.

A.4 The worst predictions for Class 0 - negative tweets

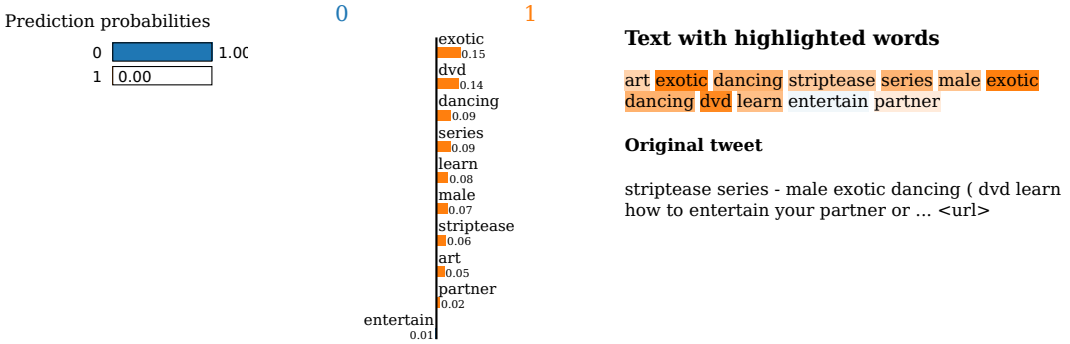

Figure 8: The worst (most inaccurate) prediction for a negative tweet among 20000 tweets.

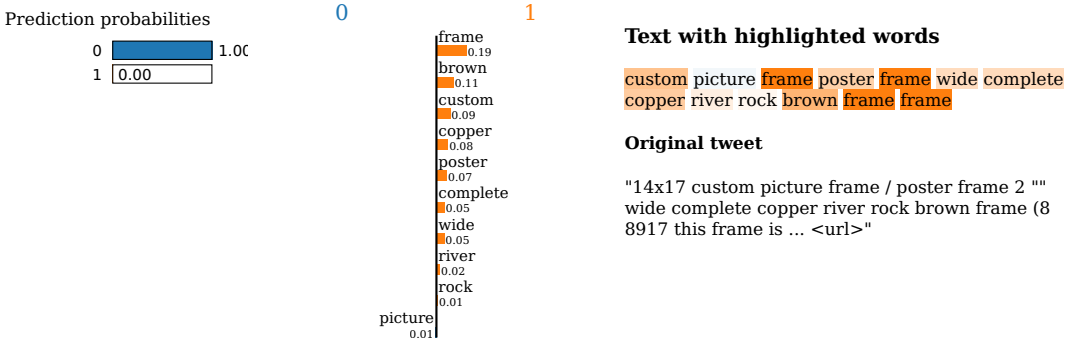

Figure 9: The second-worst (most inaccurate) prediction for a negative tweet among 20000 tweets.