Robust language models for binary classification on English tweets using active meta learning and various pre-processing techniques

Till Arnold[1], Lucas Cosier[1], Georg Teufelberger[1], Shih-Chi Yang[2]
group: barely-passing
[1]Department of Computer Science, ETH Zurich, Switzerland
[2]Department of Information Technology and Electrical Engineering, ETH Zurich, Switzerland

July 31, 2022

Abstract

In this report we use transformer based language models like BERT and its derivatives as a basis to perform sentiment analysis on Twitter data. We enhance BERTweet with various pre-processing techniques to deal with noisy labels produced by distant supervision. We leverage meta and active learning to further improve performance. Our best model achieves 91.36% test accuracy on the public leaderboard.

1 Introduction

In the past few years, there has been a huge growth in the use of microblogging platforms such as Twitter [5]. The age of interconnectedness has provoked an increased interest from companies seeking to extract and/or analyze information embedded in data which is published on such platforms. This phenomenon led to the emergence and proliferation of the sentiment analysis task in the domain of natural language processing. The description of the goal is simple: to classify a corpora of text as belonging to some sentiment class. In this case, the scope of our project was to classify tweets as positive or negative.

This task is made interesting due to the informal nature of the language used in these corpora, as well as the peculiarities of the language itself, such as sarcasm. Informal language often includes informal intensifiers, spelling mistakes, abbreviations, emojis, and sometimes it might not even be intelligible. The dataset provided consists of $2.5M$ tweets labeled using distant supervision, as opposed to human annotators. The tweets are deemed positive if they originally contained a `:)`, and negative if they contained a `:(`. Therefore, the dataset contains *noisy* labels, with possible false positives and negatives. We discuss possible solutions to this problem in section 2.1.

2 Approach

In the class of language models the current state-of-the-art on many NLP tasks is achieved by the multi-layer bidirectional Transformer encoder which uses the self-attention mechanism [11]. In this family of models, BERT [3] has been successfully applied to the tasks of language modelling and sequence classification [3]. There are two steps in the BERT framework: *pre-training* and *fine-tuning*. For the first stage, the authors there use BERT pre-trained on two self-supervised tasks (masked language modeling and next sentence prediction) using a large-scale unlabeled general domain corpus ($3.3B$ words from BooksCorpus and English Wikipedia). For the latter stage, BERT is initialized with the pre-trained parameter weights, and all of the parameters are fine-tuned using labeled data from downstream tasks, in this case sequence classification.

BERTweet (BERT pre-trained on English Tweets), is a domain-specific variant of RoBERTa which itself is a variation of BERT [7, 6]. Here, the large-scale unlabeled general domain corpus is replaced with $\approx 880M$ informal, English tweets, on which the language representation model is pre-trained. Therefore, BERTweet shares almost the same architecture across tasks with BERT, but outperforms it on domain-specific language modeling tasks. We use BERTweet due to its domain specificity as we believe its enriched vocabulary (which can include slang and other informal styles of speech) would be better suited for sentiment analysis.

2.1 Pre-processing

Our data pre-processing stage consists of multiple steps, organized in a pipeline. The aim of our pre-processing is that the multiple steps will lead us to a token that is understood by BERTweet while being as close as possible to the original token. We follow [7] and normalize tweets by formatting input strings, like user and URL tags, with special tokens. Our hope is that this normalization improves the performance of our chosen language model during inference.

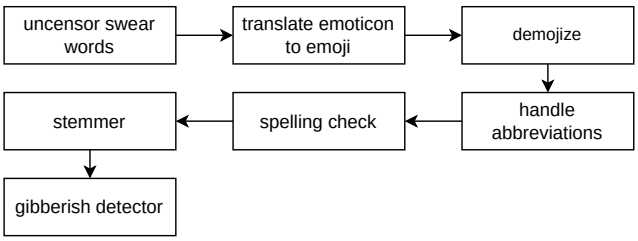

Figure 1: Various steps in pre-processing pipeline

Curse word uncensoring Curse words are generally more emotional, as such they can give a greater hint towards the sentiment of a sentence. Investigating the dataset we found tweets that had user-censored curse words, e.g. "`d*ck`", which are not detected correctly by BERTweet. To uncensor them we used a static mapping that was generated using the censored words and publicly available datasets of swearwords[1]. With this mapping we can then uncensor words again, i.e. "`d*ck`" → "`dick`", where only the latter of which is detected by BERTweet, which in turn should allow for more precise sentiment detection.

Emoticons & Demojizing We follow [5] and treat emoticons and abbreviations as individual tokens, since they contribute to the tokenization process.
BERTweet recommends converting emoji into a textual representation, as such converting ☺️ to `:smiling_face_with_smiling_eyes:`. We call this process *demojizing*. We go one step further and also convert emoticons such as `:)` or `=)` or `:-]` to the same token. For this we first convert emoticons to emoji and then demojize that emoji. For example `:'(` or `; '(` would be converted to the 😢 emoji and then to `:crying_face:`.

Abbreviations and stemming Abbreviations are replaced by their actual meaning (e.g., `idk` → `i don't know`). We also try to identify words with informal intensifiers such as character repetitions (e.g., `awwww`), and replace them with their *stem*.

Spelling corrections If after these steps the result is still not in the BERTweet dictionary we take the original token and correct the spelling of it.[2] Tweets simply may occasionally contain typos and correcting theses should allow BERTweet to correctly tokenize the words without changing the meaning of the sentence.
During experimentation however, we found out that spelling correction had negative impact on our performance so we decided not to us it in the end.

Gibberish detection Tweets are pre-processed in the order as described above. If a resulting token is contained in the BERTweet dictionary it is left as-is. But due to the informal language exhibited, it is possible that even after applying all of the steps the token is still not understood by BERTweet. This can, for example, be the case because the token is not actually a human understandable word or because we are dealing with a compound word such as "backache": The BERTweet dictionary contains both "back" and "ache" but not "backache". To distinguish these cases we employ a simple pre-existing Markov Chain-based gibberish detector.

Weak labels In their report from 2020 [2] students working with a similar dataset found that there are "numerous unmatched parentheses in the tweets" and claim to have found "strong evidence [...] that unmatched parentheses are remainders of labels within our tweets" which "can lead to models that rely on these patterns which originated as an artifact of the data collection process". In their analysis they rely on unmatched parenthesis.
As our goal is not to simply rely on unmatched parenthesis for a high score, but to instead train the model on the language of the tweets, we decided to incorporate the findings of the other report and to consequently:

1) Remove all parenthesis before passing the tweets to BERTweet

2) Drop all tweets with unmatched parenthesis after converting emoticons to their demojified form from our training set.

3) Remove negative tweets containing positive emoticons and positive tweets containing negative emoticons from our training set.

By removing tweets with unmatched parenthesis only after replacing emoticons with their demojified form we avoid removing tweets for containing emoticons. Additionally by dropping tweets with non matching emoticons we deal with sarcasm. Our aim it that with these steps we reduce the impact of the weak labeling caused by the unmatched parenthesis that was described in the previous mentioned project.

2.2 Active learning

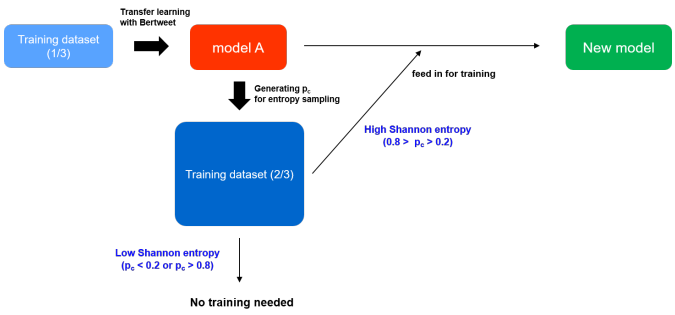

Figure 2: Active learning in tweet classification task.

The main idea for active learning (AL) is to achieve better performance by using fewer data for training, which can be selected based on the confidence from the classifier. It can not only reduce the training time but also significantly reduce the annotation effort in the future when new data comes.

Herein, we performed uncertainty sampling, which utilizes probabilistic classifiers to select the most uncertain samples. The key is that the samples with high uncertainty define the decision boundary. Clarifying this

[1] The two lists used are:

- https://www.freewebheaders.com/download/files/facebook-bad-words-list_comma-separated-text-file_2022_05_01.zip
- https://www.freewebheaders.com/download/files/youtube-blacklist-words_comma-separated-text-file.zip

[2] The spelling correction is done using symspellpy using `frequency_dictionary_en_82_765.txt`.

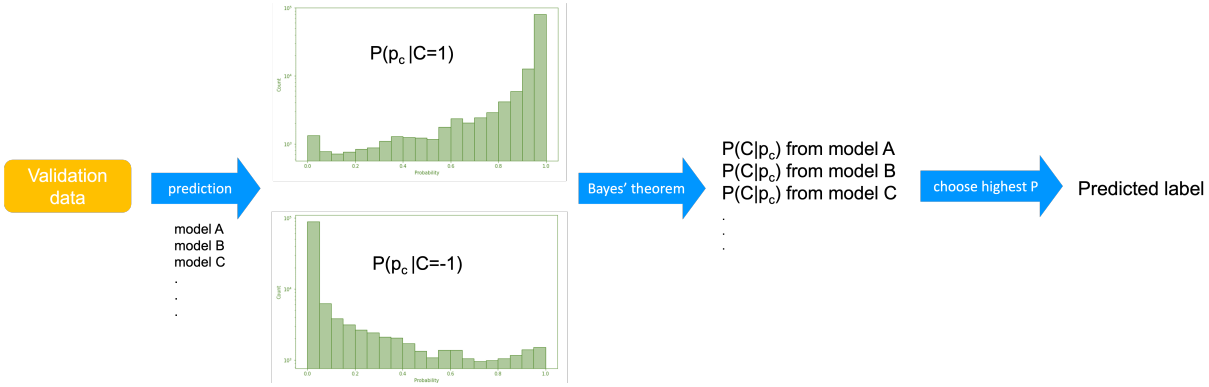

Figure 3: Applying Bayes' theorem to predict the labels with different models

boundary is more efficient to learn an even better classifier. One of the most common way to perform uncertainty sampling is applying entropy sampling, which selects the samples with the highest Shannon entropy, as shown in **Equation 1**, where p_c is the probability of the sample belonging to class c. In our tweet binary classification, entropy sampling selects the samples with p_c around 0.5. For example, if the classifier predicts a p_c of 0.99, the classifier is rather confident in the predicted result. Therefore, the model will not learn too much about the decision boundary from this tweet, so training with it can be skipped. The process of active learning in this task is illustrated in **Figure 2**. Moreover, if this tweet is not yet labeled, the effort for annotating on it might not be worthy.

$$H = -\sum_{c} p_c \log p_c \tag{1}$$

2.3 Meta-learning

Finally, we further improved the accuracy by meta-learning (ML), which refers to machine learning algorithms that learn from the outputs of other machine learning algorithms.

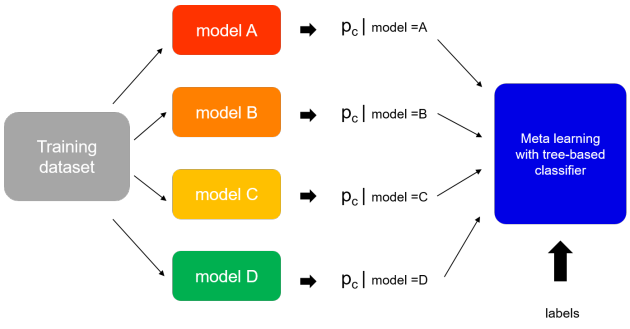

Figure 4: Meta-learning with tree-based classifiers

2.3.1 Meta-learning with Bayes' theorem

Figure 3 shows how the predictions were generated among different models. First of all, we generated the p_c distribution with different models given the ground truth, i.e. $P(p_c|C)$. By applying Bayes' theorem, $P(C{=}1|p_c)$ and $P(C{=}{-}1|p_c)$ for all the models can be derived. The highest $P(C|p_c)$ among all the models determine the final predictions.

2.3.2 Meta-learning with tree based classifiers

The method with Bayes's theorem is based on the assumption that only the highest $P(C|p_c)$ is relevant for the prediction. However, it might lose some information given by other models. To take all $P(C|p_c)$ into account, we trained tree-based classifiers for meta-learning, such as XGBoost and Randomforest, on the whole training dataset with p_c generated from different models as predictors to predict the labels, as shown in **Figure 4**.

3 Experimental results

3.1 Results

To compare our ideas, we measure the model's performance on the public leaderboard. Models A to E in Table 1 represent our baseline approach, and F to I refer to applying AL and ML to our best running models. All our models were run with a batch size of 40, learning rate of $2e-5$, and sequence length = 70.
Our first attempt at this challenge was using DistilBERT [9], which is a compressed variant of BERT, with about 30% less parameters. The model was fine-tuned over 2 epochs using Adam [4], achieving test accuracy on the raw dataset of 88.04% on the public leaderboard (Model A). Additionally, as seen in Table 1, using data normalization as defined in the previous section, yields a further increase to 88.42% test accuracy (Model B). A major improvement in our baseline constituted using BERTweet instead, with an improvement of 1.23% on raw data alone (Model C). For BERTweet with pre-processed data we did not include the score in the table as it proved to be suboptimal due to a bug. Using AL, we gain 0.26% (Model F), and further adding ML with an ensemble of three BERTweet models together with a random forest classifier yields our best submission with an overall improvement of 0.56% over model C, with only raw data.

3.2 Discussion & Further Work

For all runs, we ran our models on $\approx 1/3^{rd}$ of the dataset, since we achieve the best performance on this subset, as

	Method	Description	Leaderboard Score
Model A	DistilBERT	using 0.8M raw data	0.8804
Model B		using 0.8M preprocessed data	0.8842
Model C	BERTweet	using 0.8M raw data	0.9072
Model D		using raw 1.6M data	0.9042
Model E		using raw 2.25M data	0.9018
Model F	BERTweet + AL	model C + AL ($0.2 < p_c < 0.8$)	0.9098
Model G	BERTweet + AL + ML	Model F + Bayes' theorem	0.9126
Model H		3 models with XGBoost classifier	0.9132
Model I		3 models with Randomforest classifier	0.9136

Table 1: Model performance with and without active learning

reflected in Table 1. We believe this is due to the noisiness given by distant supervision and choosing a lucky seed. Additionally, our initial analysis revealed that, although the tweet length was variable, with some going over 160, setting the sequence length to something bigger, i.e. 140, did not yield improvements.

We expected that by combining data pre-processing with BERTweet we would be able to gain a noticeable increase in accuracy, as we did with DistilBERT. However, much to our dismay, we discovered a bug in our code that *demojified* the smileys incorrectly. While for DistilBERT this did not matter too much, since it can't "understand" emojis anyway, performance with BERTweet was sub-optimal. We note that this bug has been fixed since, however due to extensive training times we were not able to re-run our experiments to check the effect of this fix, similarly with weak labeling. We detected $\approx 10k$ tweets which are wrongly labeled according to the procedure outlined in section 2.1, and produced a "curated" dataset, which was never tested. We leave training a model on this dataset as further work.

Therefore, we note three areas of further work: removing weakly labeled tweets, and model compression and pipeline optimization.

Training such a model for one epoch on just $1/3^{rd}$ of the training set, which is *small* compared to datasets of today, consumes 2 hours of computational time. Distillation is the process of transferring the knowledge from a large and accurate model (the teacher) to a smaller model with less representational power (the student). [10] were able to train a single-layer biLSTM by distilling knowledge from BERT, to achieve results comparable to ELMo [8] across multiple datasets and tasks, while using roughly 100× fewer parameters and 15× less inference time. More recently, [12] were able to use weight quantization (the process of reducing the number of bits used to represent a single scalar parameter) in conjunction with distillation to achieve comparable performance as the full-precision model while being $\sim 15\times$ smaller. We note these are viable approaches that could be very easily applied to distill BERTweet for computation-efficient inference.

Additionally, the quality, as well as the performance of the pre-processing pipeline, could be improved. Our initial testing revealed that the spelling correction was detrimental to our results, and in the end we opted to not use it. This might be due to the usage of `symspell`, which was used as all other tested alternatives were too slow. Employing a context-sensitive spelling correction tool might lead to improvements of the results and might be worth investigating.

Secondly, the pre-processing is time intensive which is probably due to implementation issues: A lot of string and regular expression based searches and replacements are naively performed. It is to be presumed that a significant speedup could be achieved, for example using something like a finite-state machine based string-searching algorithm similar to Aho–Corasick [1].

4 Conclusion

Our experiments show that pre-processing the raw tweets by identifying emoticons and correcting spelling leads to slight improvement of the results. Although this is beneficial for DistilBERT, our pipeline proved to be detrimental for BERTweet due to one now caught bug.

We showed that using BERTweet as a basis and building on top of it with our own pre-processing as well as using active and meta learning generally gives >90% accuracy. This was despite the low performance improvement achieved with pre-processing.

Active learning helps us to select samples with high-entropy and reduce the training time while reaching even higher accuracy. It also demonstrates the potential to save efforts for labeling when more unlabeled tweets are available.

Last but not least, we performed meta-learning with different tree-based classifiers to extract additional information from the output probabilities of different models. At the time of writing we reached 7th place on the scoreboard out of 25.

References

[1] Alfred V. Aho and Margaret J. Corasick. Efficient string matching: An aid to bibliographic search. *Commun. ACM*, 18(6):333–340, jun 1975.

[2] Sinan Demirci, Claudio Ferrari, Jérémy Scheurer, and Vanessa Tschichold. @bertweet and friends analyzing sentiments #investigation, 2020.

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

[4] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014.

[5] Efthymios Kouloumpis, Theresa Wilson, and Johanna Moore. Twitter sentiment analysis: The good the bad and the omg! *Proceedings of the International AAAI Conference on Web and Social Media*, 5(1):538–541, Aug. 2021.

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

[7] Dat Quoc Nguyen, Thanh Vu, and Anh Tuan Nguyen. BERTweet: A pre-trained language model for English Tweets. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 9–14, 2020.

[8] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations, 2018.

[9] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter, 2019.

[10] Raphael Tang, Yao Lu, Linqing Liu, Lili Mou, Olga Vechtomova, and Jimmy Lin. Distilling task-specific knowledge from bert into simple neural networks, 2019.

[11] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[12] Wei Zhang, Lu Hou, Yichun Yin, Lifeng Shang, Xiao Chen, Xin Jiang Xin, and Qun Liu. Ternarybert: Distillation-aware ultra-low bit bert. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020.

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Declaration of originality

The signed declaration of originality is a component of every semester paper, Bachelor's thesis, Master's thesis and any other degree paper undertaken during the course of studies, including the respective electronic versions.

Lecturers may also require a declaration of originality for other written papers compiled for their courses.

I hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by the supervisor.

Title of work (in block letters):

ROBUST LANGUAGE MODELS FOR BINARY CLASSIFICATION ON ENGLISH TWEETS USING ACTIVE META LEARNING AND VARIOUS PRE-PROCESSING TECHNIQUES

Authored by (in block letters):

For papers written by groups the names of all authors are required.

Name(s):	**First name(s):**
TEUFELBERGER	GEORG
YANG	SHIH-CHI
COSIER	LUCAS
ARNOLD	TILL

With my signature I confirm that

- I have committed none of the forms of plagiarism described in the 'Citation etiquette' information sheet.
- I have documented all methods, data and processes truthfully.
- I have not manipulated any data.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for plagiarism.

Place, date

ZÜRICH, 31. JUL 2022

Signature(s)

For papers written by groups the names of all authors are required. Their signatures collectively guarantee the entire content of the written paper.