Two-Stage Curriculum Ensemble Learning for Text Sentiment Classification

Andreas Psaroudakis   Dimitrios Dekas   Georgios Tsolakis   Konstantinos Fanaras
Group name: Machinists
Department of Computer Science, ETH Zurich, Switzerland

I. Abstract

***Abstract*— Sentiment analysis in human-generated text has gained increased popularity over the past few years. The abundance of online corpora and the rapid advances in models capable of handling large-scale data has played a crucial role towards this direction. In this work, we perform an extensive experimental study for tweet sentiment classification. We also propose a novel two-stage curriculum ensemble learning approach that combines the benefits of curriculum learning with the utilization of erroneous data. Our method surpasses all other state-of-the-art (SOTA) models, achieving a classification accuracy of 91.48% on the public test set.**

II. Introduction

Sentiment Analysis refers to a prominent field within Natural Language Processing which is concerned with identifying emotions in human-written text. The ability to automatically extract the opinions of a large number of individuals for a given subject can prove to be an extremely powerful tool. Therefore, the identification of sentiment finds extensive use in numerous domains, such as market research, social media monitoring, and campaign reach evaluation.

A crucial factor contributing to the immense popularity and growth of sentiment analysis is the universal adoption of daily social media usage. Platforms like Facebook and Twitter have witnessed an unprecedented amount of text data generation, providing researchers with the opportunity to develop and evaluate novel techniques in a diverse landscape of ever-growing data collections.

Furthermore, recent advancements in deep learning paved the way for powerful frameworks that have given rise to even more efficient approaches for sentiment analysis. However, current state-of-the-art methods showcase drawbacks that impair their large-scale usability. They require large corpora of labeled samples, drastically increasing the resources necessary for their usage. In addition, samples obtained from open sources display several problematic features, hindering their quality.

The rest of this report is organized as follows: In Section III we present numerous methods that are commonly utilized to tackle the sentiment analysis task. In Section IV we discuss the details surrounding the data used in our study and propose various transformations for data pre-processing. In Section V the employed models and their technical details are introduced. In Section VI the experimental results are analyzed and useful conclusions are being drawn. Finally, in Section VII we summarize our work and provide guidelines for possible future extensions.

III. Related Work

Many methods attempting to tackle the sentiment analysis task have been proposed over the years. Lexicon-based analysis has mostly relied on hard-coded dictionaries, attributing words with a certain level of sentiment, aggregating the individual scores to obtain a global one [1] [2] [3]. Feature extracting procedures, namely Bag-of-Words, and TF-IDF, have been deployed to transform the initial word representations into numerical vectors and create an easier classification task, solved by using classical machine learning algorithms [4] [5].

Building on the idea of feature extraction, later works proposed alternative representations, producing more compact and expressive vector spaces than their previous counterparts, commonly referred to as word embeddings [6] [7] [8] [9]. In addition, advancements in deep learning architectures presented researchers with a powerful toolkit of methods. Recurrent Neural Networks (RNNs) [10] [11] [12], Convolutional Neural Networks (CNNs) [13] [14] [15], and hybrid models [16] [17] [18], have all been extensively used in classifying the sentiment of individual samples, demonstrating promising results.

In recent years, there has been a significant shift in the Natural Language Processing literature toward the widespread utilization of attention-based transformer architectures. From language translation to sentiment analysis, Bidirectional Encoder Representations from Transformers (BERT) [19] and its variants [20] [21] [22] [23], have been constantly monopolizing the state-of-the-art architectures, achieving high-performance level and surpassing previously developed strategies.

IV. Dataset

A. Exploratory Data Analysis

The initial dataset consists of 1.25M positive and 1.25M negative tweets. The labeling process was conducted by deploying a "noisy" scheme, where a given tweet is labeled as positive if it contains a smiley emoji ':)'. On the other hand, for negative labeling, the presence of a frowning emoji ':(' is necessary. Also, duplicate tweets can be found in the dataset, on both the training and the test set. Moreover, user mentions have been replaced with the unique <user> tag and the hyperlinks with a <url> tag in the entirety of the dataset.

Apart from the typical problems that are expected from any Twitter data collection, such as spelling mistakes and slang usage, there exist issues that are a result of the data acquisition and labeling scheme. A considerable amount of tweets are characterized by having the '... <url>' sequence as their trailing characters. The technique utilized to create the dataset opts to use shortened hyperlinks[1] for lengthy tweets. Therefore, such hyperlink instances are converted to the '... <url>' sequence.

In addition, the usage of smiling and frowning emojis for labeling is problematic in different ways. First, there is no certainty of the correlation between sentiment and the usage of the emoji character sequences. This fact leads to the usage of samples without an evident sentiment load. Also, there are instances where emoji dropping leaves parentheses in a tweet with no particular meaning. This peculiarity can result in biased models that are "inferring" the sentiment of a given tweet by detecting the ')' and '(' characters.

B. Data Pre-processing

An integral part of NLP is the pre-processing transformations applied to the raw data provided for a specific task. Therefore, we opt to utilize various pre-processing pipelines, basing our decisions on the complexity and internal mechanism of our models.

1) General Steps: At first, we implement some general pre-processing steps for all model architectures:

- Dropping duplicate tweets with a factor of 0.8: The factor is used to determine the label selected for samples that appear with both labeling options. In such cases, the occurrences of the majority label to be used are required to constitute at least 80% of the total sample instances for the tweet not to be dropped.
- Using ekphrasis [24] we apply the following:
  - Normalization of emails, percentages, money tokens, phones, users, time tokens, urls, and dates.
  - Hashtag segmentation into individual words.
  - Normalization of 3 or more consequtive characters to a single one (e.g. 'thankssss' transformed to 'thanks')
  - Spell correction of words based on Twitter corpus.
  - Transformation of emoticons into words.
- Normalization of laughter to 'haha'. A typical regular expression is being used in order to identify laughing patterns in tweets and replace them with the 'haha' substring. For instance, tokens such as 'aahaahhha', 'haaahaahhhahah', and 'bwahaahahhaha' are all replaced with 'haha'.
- Removal of angle brackets surrounding words and tags.
- Removal of retweet tags ('rt').

[1]https://help.twitter.com/en/using-twitter/url-shortener

2) Non-Transformer-based steps: Now, we focus on the baseline models, which possess lower complexity when compared to transformer-based approaches, while also lucking the powerful attention mechanism offered by the latter. For the data used to train and evaluate these models, we propose the following pre-processing steps, in addition to the general steps, taking into account their need for simple, sentiment-rich tokens:

- Usage of ekphrasis is done with its Whitespace tokenizer.
- Removal of user tags, url tags, stopwords, punctuations, single characters, numeric tokens, and trailing urls.
- Word lemmatization.
- Manual hardcoded spell correction.

The resulting dataset will be referred to as '**n-TrDs**' for subsequent sections.

3) Transformer-based steps: For the transformer-based architectures, after applying the general steps, we make use of a much simpler, tailored pre-processing pipeline:

- Usage of ekphrasis is done with its Social tokenizer.
- Substitution of user tags with '@USER'.
- Substitution of url tags with 'HTTPURL'.

We will refer to this dataset as '**TrDs**' from now on.

C. Extra Dataset

Due to the various limitations in the provided data, as described in section IV-A, we opt to use an additional dataset[2] to try to further boost the classification performance of our best model. This dataset consists of 1.6M tweets extracted using the Twitter API, 50% positive and 50% negative, and has been automatically annotated.

V. Models and Methods

In this section, we present the models used for the sentiment classification task, describing their architecture and associated experimental details. The models are divided into two categories: a) Non-Transformer-based: trained on n-TrDs and b) Transformer-based: fine-tuned on TrDs.

A. Non-Transformer-based architectures

First, we experiment with simple, non-transformer-based models, to set some baselines for our task.

- **TF-IDF** [25]: An approach that combines "Term Frequency" and "Inverse Document Frequency" to express word importance within a corpus. A TF-IDF Vectorizer is utilized to transform tweets into embeddings, which are then fed into a Linear Support Vector Machine and a Random Forest Classifier.

[2]https://www.kaggle.com/datasets/kazanova/sentiment140

- **GloVe** [26]: An unsupervised algorithm that constructs word-level representations based on co-occurrence statistics in a corpus. We aggregate 200-dimensional world-level GloVe embeddings[3] to construct Tweet encodings, which are then fed into an XGBoost Classifier.
- **Word2Vec** [27]: As the next step, we opt to create our own embeddings based on the provided Twitter dataset. Thus, we train a Word2Vec Neural Network for 25 epochs using the CBOW technique [28]. To evaluate the resulting 250-dimensional word embeddings, we query for the most correlated words with "fun". Similar to before, we construct our tweet embeddings by aggregating the Word2Vec word representations and then feed them into an XGBoost Classifier to perform sentiment categorization.
- **FastText** [29]: This method utilizes a skip-gram model to compute word embeddings, representing each word as a "bag of character n-grams". For text classification, it efficiently implements a classifier using a technique similar to Bag-of-Words, known as Bag-of-Tricks [30]. In a supervised fashion, we fine-tune FastText for 20 epochs using 150-dimensional embeddings.
- **Bidirectional LSTM** [31]: A model that is commonly used in various NLP tasks. We utilize an architecture that consists of an Embedding layer, a Bidirectional LSTM, a Dropout, and a Dense layer. We tokenize and pad the tweets before feeding them into the network and train for 25 epochs.

B. *Transformer-based architectures*

Now we move on to some state-of-the-art transformer-based models:

- **BERT** [19]: A recent self-attention mechanism that applies the bidirectional training of Transformer to language modeling. Due to computational limitations, we utilize the base version of BERT[4]. Two different approaches are tested. First, we directly use the pre-trained BERT base model to transform our tweets into 758-dimensional embeddings, which are utilized to train an XGBoost Classifier. In the second approach, we fine-tune the pre-trained BERT base model, incorporating a classification head comprising a Pooling layer, a Dropout, a Dense layer, and a softmax activation. The tweets are encoded using the corresponding tokenizer with a maximum length of 128, and the model is fine-tuned for 3 epochs.
- **BERTweet** [23]: It constitutes the first public large-scale pre-trained language model for English Tweets, has the same architecture as BERT and is trained using the RoBERTa [20] pre-training procedure. We utilize the base version[5] which is pre-trained on a corpus of 850M Tweets. Similar to BERT's fine-tuning, we add a classification head on top of the rest architecture, set a maximum length of 128 and train for 3 epochs.
- **Twitter RoBERTa Sentiment:** This RoBERTa based model[6] is pre-trained on 124 million tweets and is already finetuned on the TweetEval benchmark [32] for sentiment classification. Similar to previous models, we train the existing fine-tuned classification head for 3 epochs.

At this point we want to test the following hypothesis: Can multilingual transformers exploit knowledge from different languages to enhance the accuracy of the English tweet classification task? To provide an answer to this question, we experiment with two popular multilingual BERT-based models, that are pre-trained on billion tweets.

- **TwHIN-BERT** [33]: This model is pre-trained on a corpus of 7 billion tweets from 100 languages. During the pertaining phase, a new "contrastive social loss" is employed in addition to the classic Masked Language Modelling loss. Adding a classification head on top of it and using the respective tokenizer[7], we train the model for 3 epochs.
- **Bernice** [34]: Another multilingual BERT-encoder model pre-trained on 2.5 billion tweets from 66 languages. Similar as before, we utilize its custom tokenizer[8] and fine-tune the model for 3 epochs.

C. *Two-Stage Curriculum Training Ensemble*

To further boost the classification performance of our best transformer-based model (i.e. BERTweet), we implement a novel two-stage curriculum learning approach with voting ensemble, inspired by [35].

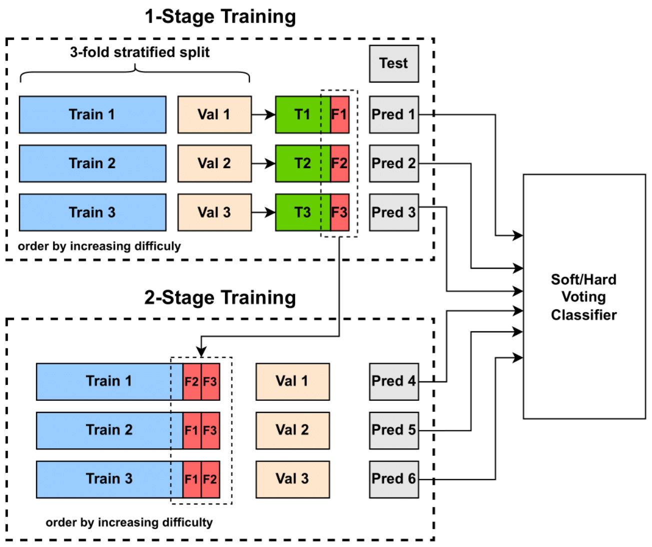

Figure 1. Two-stage curriculum learning approach with voting ensemble

As illustrated in Figure 1, in the 1-Stage training, we first perform a 3-fold-stratified split to construct three train-

[3]https://nlp.stanford.edu/projects/glove/
[4]https://huggingface.co/bert-base-uncased
[5]https://huggingface.co/vinai/bertweet-base
[6]https://huggingface.co/cardiffnlp/twitter-roberta-base-sentiment-latest
[7]https://huggingface.co/Twitter/twhin-bert-base
[8]https://huggingface.co/jhu-clsp/bernice

val datasets. Then we apply a curriculum learning strategy [36], reordering each train fold by increasing difficulty, i.e. from the easy samples to the hard ones. To achieve this, we use a proxy pre-trained BERTweet model, fine-tuned on our dataset, to assign a hardness score to each tweet of the initial dataset. This score is calculated as the absolute deviation between the positive logit probability and the true class label (supposing the negative class corresponds to the 0 label and the positive class to 1). Last but not least, to avoid accumulating a large number of samples from the same category (i.e. many negative samples at the start or many positive samples at the end) we slightly redistribute the train folds by maintaining the original class distribution ratio alongside (i.e. alternating between positive and negative samples of similar difficulty).

The goal of 2-Stage is to supplement the 1-Stage training with erroneous data. For that purpose, we cross-supplement the misclassified data from the validation sets to the training sets, creating a number of duplicate samples in the train sets. To be more precise, in the first stage of training, we divide the validation predictions of each fold into two groups: Correctly predicted/True (T_i) and Wrongly predicted/False (F_i), for $i \in \{1, 2, 3\}$. Each false set is now added to the other two train folds (for instance, $F1$ is added to $Train2$ and $Train3$), resulting in three new train sets, depicted in Figure 1. Similar to 1-Stage training, this augmentation process is first followed by a reordering of the updated train data, by increasing difficulty, and then by a final rearrangement to prevent sentiment clusters along the edges.

The predictions on the test set, produced by each one of the six separate experiments, are subsequently integrated using either a soft or a hard voting ensemble algorithm to produce the final predictions. To the best of our knowledge, this is the first time the benefits of curriculum learning are combined with the utilization of erroneous data in a unified framework to increase the performance of text classification.

VI. Evaluation

To train and evaluate our models we split our dataset into a training and a validation set. For all experiments, except the last ones involving 2-Stage training approaches, we define the validation set as a 25% partition of the total dataset.

Table I summarizes the results of our experiments both on validation and test set. All non-transformer-based architectures are not evaluated in the public test set, since they perform relatively low on the validation set. However, we also include them in the table for our analysis to be complete.

As shown, the Bidirection LSTM architecture outperforms all other baseline approaches by scoring 80.41%. The TF-IDF technique performs the worst, however, when combined with a non-linear model (i.e. RandomForest) we notice a performance boost. Our own trained Word2Vec embeddings lead to noticeably higher accuracy, i.e. 79.32%, surpassing even the pre-trained Twitter GloVe embeddings. In addition, the FastText method scores slightly above 80%, whereas, surprisingly, the embeddings extracted from a BERT transformer result in worse performance.

Model Architecture	Val Acc %	Test Acc %
TD-IDF + LinearSVM	67.80	-
TD-IDF + RandomForest	73.44	-
Glove + XGBClassifier	79.05	-
Word2Vec + XGBClassifier	79.32	-
FastText	80.18	-
Bidirectional LSTM	**80.41**	-
BERT embeddings + XGBClassifier	77.71	-
BERT	89.44	89.72
BERTweet	**90.52**	**90.88**
BERTweet + Extra Dataset	89.53	90.86
Twitter RoBERTa Sentiment	89.96	89.88
TwHIN-BERT	89.52	89.48
Bernice	89.75	89.88
BERTweet with curriculum	90.71	91.14
BERTweet two-stage curric. train soft voting	**90.96**	**91.48**
BERTweet two-stage curric. train hard voting	90.77	91.32

Table I
Experimental results on validation and test set

Regarding the transformer-based fine-tuning experiments, we observe that BERTweet is the only model achieving classification accuracy over 90%, both on validation and test set. However, it is worth observing that the augmentation with additional data does not further improve its performance. Finally, the two multilingual models, TwHIN-BERT and Bernice, appear to perform worse than BERTweet.

Applying curriculum learning on BERTweet shows a small accuracy increase compared to training on randomly ordered data. Last but not least, it is evident that our novel two-stage curriculum ensemble learning approach surpasses all previous models by achieving the best classification score on validation and public test set.

VII. Summary

In this work, we conduct an extensive experimental analysis for tweet sentiment classification. At first, we focus on a thorough data exploration and cleaning, by applying various pre-processing steps on the raw data. Next, we conduct a large experimental study that includes: performance comparison between simple, non-transformer-based models, transformer-based architectures, and ensemble approaches with and without curriculum learning techniques.

Our study demonstrates the superiority of transformer-based fine-tuning methods, highlighting BERTweet as the best-performing model. It also depicts that our proposed novel two-stage curriculum ensemble learning approach leads to additional performance gains, by incorporating the benefits of curriculum learning and utilizing erroneous data.

Future work should focus on experimenting with different numbers of stratified splits (i.e. hyperparameter k) for the dataset partitioning as well as on including additional SOTA transformer-based architectures in the training pipeline.

References

[1] M. Taboada, J. Brooke, M. Tofiloski, K. Voll, and M. Stede, "Lexicon-based methods for sentiment analysis," *Computational linguistics*, vol. 37, no. 2, pp. 267–307, 2011.

[2] L. Zhang, R. Ghosh, M. Dekhil, M. Hsu, and B. Liu, "Combining lexicon-based and learning-based methods for twitter sentiment analysis," *HP Laboratories, Technical Report HPL-2011*, vol. 89, pp. 1–8, 2011.

[3] F. M. Kundi, A. Khan, S. Ahmad, and M. Z. Asghar, "Lexicon-based sentiment analysis in the social web," *Journal of Basic and Applied Scientific Research*, vol. 4, no. 6, pp. 238–48, 2014.

[4] M. Wang, D. Cao, L. Li, S. Li, and R. Ji, "Microblog sentiment analysis based on cross-media bag-of-words model," in *Proceedings of International Conference on Internet Multimedia Computing and Service*, ser. ICIMCS '14. New York, NY, USA: Association for Computing Machinery, 2014, p. 76–80. [Online]. Available: https://doi.org/10.1145/2632856.2632912

[5] M. S. Basarslan, F. Kayaalp *et al.*, "Sentiment analysis with machine learning methods on social media," 2020.

[6] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," *CoRR*, vol. abs/1310.4546, 2013. [Online]. Available: http://arxiv.org/abs/1310.4546

[7] J. Pennington, R. Socher, and C. Manning, "GloVe: Global vectors for word representation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1532–1543. [Online]. Available: https://aclanthology.org/D14-1162

[8] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *Transactions of the Association for Computational Linguistics*, vol. 5, pp. 135–146, 2017. [Online]. Available: https://aclanthology.org/Q17-1010

[9] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," 2018.

[10] S. Wen, H. Wei, Y. Yang, Z. Guo, Z. Zeng, T. Huang, and Y. Chen, "Memristive lstm network for sentiment analysis," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 51, no. 3, pp. 1794–1804, 2019.

[11] Y. Ma, H. Peng, T. Khan, E. Cambria, and A. Hussain, "Sentic lstm: a hybrid network for targeted aspect-based sentiment analysis," *Cognitive Computation*, vol. 10, pp. 639–650, 2018.

[12] Y. Wang, M. Huang, X. Zhu, and L. Zhao, "Attention-based lstm for aspect-level sentiment classification," in *Proceedings of the 2016 conference on empirical methods in natural language processing*, 2016, pp. 606–615.

[13] R. Johnson and T. Zhang, "Effective use of word order for text categorization with convolutional neural networks," *arXiv preprint arXiv:1412.1058*, 2014.

[14] S. Liao, J. Wang, R. Yu, K. Sato, and Z. Cheng, "Cnn for situations understanding based on sentiment analysis of twitter data," *Procedia Computer Science*, vol. 111, pp. 376–381, 2017, the 8th International Conference on Advances in Information Technology. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050917312103

[15] ——, "Cnn for situations understanding based on sentiment analysis of twitter data," *Procedia computer science*, vol. 111, pp. 376–381, 2017.

[16] J. Wang, L.-C. Yu, K. R. Lai, and X. Zhang, "Dimensional sentiment analysis using a regional cnn-lstm model," in *Proceedings of the 54th annual meeting of the association for computational linguistics (volume 2: Short papers)*, 2016, pp. 225–230.

[17] T. Chen, R. Xu, Y. He, and X. Wang, "Improving sentiment analysis via sentence type classification using bilstm-crf and cnn," *Expert Systems with Applications*, vol. 72, pp. 221–230, 2017.

[18] M. E. Basiri, S. Nemati, M. Abdar, E. Cambria, and U. R. Acharya, "Abcdm: An attention-based bidirectional cnn-rnn deep model for sentiment analysis," *Future Generation Computer Systems*, vol. 115, pp. 279–294, 2021.

[19] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2019.

[20] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized bert pretraining approach," *arXiv preprint arXiv:1907.11692*, 2019.

[21] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "Albert: A lite bert for self-supervised learning of language representations," *arXiv preprint arXiv:1909.11942*, 2019.

[22] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter," *arXiv preprint arXiv:1910.01108*, 2019.

[23] D. Q. Nguyen, T. Vu, and A. 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*. Online: Association for Computational Linguistics, Oct. 2020, pp. 9–14. [Online]. Available: https://aclanthology.org/2020.emnlp-demos.2

[24] C. Baziotis, N. Pelekis, and C. Doulkeridis, "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)*. Vancouver, Canada: Association for Computational Linguistics, August 2017, pp. 747–754.

[25] J. Ramos *et al.*, "Using tf-idf to determine word relevance in document queries," in *Proceedings of the first instructional conference on machine learning*, vol. 242, no. 1. Citeseer, 2003, pp. 29–48.

[26] J. Pennington, R. Socher, and C. D. Manning, “Glove: Global vectors for word representation,” in *Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1532–1543. [Online]. Available: http://www.aclweb.org/anthology/D14-1162

[27] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, “Distributed representations of words and phrases and their compositionality,” *Advances in neural information processing systems*, vol. 26, pp. 3111–3119, 2013.

[28] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” *arXiv preprint arXiv:1301.3781*, 2013.

[29] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, “Enriching word vectors with subword information,” *CoRR*, vol. abs/1607.04606, 2016. [Online]. Available: http://arxiv.org/abs/1607.04606

[30] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov, “Bag of tricks for efficient text classification,” *CoRR*, vol. abs/1607.01759, 2016. [Online]. Available: http://arxiv.org/abs/1607.01759

[31] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[32] F. Barbieri, J. Camacho-Collados, L. Espinosa Anke, and L. Neves, “TweetEval: Unified benchmark and comparative evaluation for tweet classification,” in *Findings of the Association for Computational Linguistics: EMNLP 2020*. Online: Association for Computational Linguistics, Nov. 2020, pp. 1644–1650. [Online]. Available: https://aclanthology.org/2020.findings-emnlp.148

[33] X. Zhang, Y. Malkov, O. Florez, S. Park, B. McWilliams, J. Han, and A. El-Kishky, “Twhin-bert: A socially-enriched pre-trained language model for multilingual tweet representations,” 2022.

[34] A. DeLucia, S. Wu, A. Mueller, C. Aguirre, P. Resnik, and M. Dredze, “Bernice: A multilingual pre-trained encoder for Twitter,” in *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 6191–6205. [Online]. Available: https://aclanthology.org/2022.emnlp-main.415

[35] S. Cui, Y. Han, Y. Duan, Y. Li, S. Zhu, and C. Song, “A two-stage voting-boosting technique for ensemble learning in social network sentiment classification,” *Entropy*, vol. 25, no. 4, p. 555, 2023.

[36] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, “Curriculum learning,” in *Proceedings of the 26th annual international conference on machine learning*, 2009, pp. 41–48.

[37] J. Lamy-Poirier, “Layered gradient accumulation and modular pipeline parallelism: fast and efficient training of large language models,” *CoRR*, vol. abs/2106.02679, 2021. [Online]. Available: https://arxiv.org/abs/2106.02679

VIII. Appendix

All the experiments are carried out on Google Colab T4 GPUs. In Table II we present the values for the basic training hyperparameters used in our optimal configuration setting. To speed up the training process, we use mixed precision fp16 training and gradient accumulation [37].

Hyperparameter	Value
Number of epochs	3
Batch size	64
Learning rate	1e-4
Weight decay	0.005
Warm up steps	500
Gradient accumulation steps	4

Table II
Optimal training hyper-parameters used to train our transformer-based models

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Declaration of originality

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):

TWO-STAGE CURRICULUM ENSEMBLE LEARNING FOR TEXT SENTIMENT CLASSIFICATION

Authored by (in block letters):

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

Name(s):	**First name(s):**
PSAROUDAKIS	ANDREAS
DEKAS	DIMITRIOS
TSOLAKIS	GEORGIOS
FANARAS	KONSTANTINOS

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	**Signature(s)**
ZURICH, 28/07/2023	[illegible]
	[illegible]
	[illegible]
	[illegible]

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