Twitter Sentiment Classification on Ice: Fine-tuning BERTweet with Layer-Freezing

Zhenjie Jiang, Sebastian Müksch, Gabriel Fringeli, and Nathalie Bäbler

Group: Oops, Department of Computer Science, ETH Zürich
{zhejiang, smueksch, fgabriel, baeblern}@student.ethz.ch

***Abstract*—Sentiment analysis is an important branch of natural language processing. The sentiment analysis of Twitter data can give valuable insights into public perception of current events. In this paper, we perform sentiment classification on a data set of 2.5 million tweets. We compare various models from Logistic Regression to BERTweet, gaining a final test set accuracy of 91.88% using BERTweet in the Kaggle competition. We show that classical preprocessing steps do not conclusively improve the performance of BERTweet. Further, we use weight freezing for the initial fine-tuning epochs to improve BERTweet's performance. We finally demonstrate that we can gain small improvements by translating non-English tweets prior to classification.**

I. Introduction

Understanding natural language is an important branch of machine learning. An interesting and widely applicable part of this branch is sentiment analysis. One of the example application is to gauge the public perception on a current topic, such as the COVID-19 outbreak [1]. Our paper studies sentiment classification applied to Twitter data. Twitter is an increasingly popular platform, where a vast number of people give their opinion through the use of tweets. As such, tweets provide a fast and contemporaneous overview of the perception of a current situation by the public.

Although there is a substantial amount of research done on sentiment analysis, many questions remain open. We concentrate on large transformer models which have become a ubiquitous tool in natural language processing (NLP). Despite their popularity, there is not yet a commonly agreed-upon methodology for fine-tuning such models for downstream tasks such as sentiment classification. Furthermore, we do not know the effects of classical text preprocessing methods on such models. Another issue with our Twitter data set is that some tweets contain non-English words, which makes the task more complicated. In this report, we demonstrate that freezing the pre-trained weights in the initial phase of fine-tuning improves the performance for Tweet sentiment classification. We also show that translating non-English tweets to English using the Google Translate API can improve the classification accuracy on these tweets with a model trained on English tweets only.

II. Related Work

Sentiment classification is a task that used to be tackled with very different approaches. For an overview of these approaches, including lexicon approaches and SVM approaches, we recommend Medhat et al.'s survey [2]. For a general overview of deep learning models, such as convolutional neural networks and transfer learning methods, we refer to [3].

Transformer models such as BERT [4], which build on the attention mechanism [5], have achieved state-of-the-art performance in many aspects of NLP. This is reflected by the current GLUE [6] and SuperGLUE [7] leaderboards. With BERTweet [8] there also exists a model that is particularly suitable for Twitter data.

Excluding layers from the backward pass of the back-propagation algorithm, hereafter referred to as freezing, has been used in several situations. Brock et al. apply progressive layer freezing to accelerate training [9]. In the context of fine-tuning, updating only the final layers and fixing the initial layers to the pre-trained values can improve performance as compared to fine-tuning the network as a whole [10]. In contrast, our work is tailored to fine-tuning where the pre-trained model is extended with additional layers specific to a downstream task. We initially restrict training to the additional layers and switch to fine-tuning all weights after convergence of the additional layers.

III. Data

The given data set contains 1.25 million positive tweets and 1.25 million negative tweets for training, with 10k unlabeled tweets for testing. The tweets were labelled through distant supervision, where the tweets got classified as positive if they originally contained a ':)', and classified as negative if they originally contained a ':)'. In the given data set, ':)' and ':(' were removed, user mentions and links were replaced with '<user>' and '<url>' respectively.

In addition, we prepared the data set for experimentation by removing the duplicated entries, which reduced the training set to around 1.12 million positive and 1.14 million negative tweets. We then mixed the positive and the negative tweets and randomly selected 20% of the data for

the validation set, with the remaining 80% of the data kept as the training set.

From analysing the data, we discovered that the data set contains a small portion of non-English tweets. This is discussed in more detail in Section IV-D.

IV. Methods

A. Problem Formulation

We are given a set $\mathcal{D}_{\text{train}} = \{(d_i, y_i)\}_{i=1}^{N}$ of N training pairs each one consisting of a tweet text d_i and a sentiment label $y_i \in \{\text{Pos}, \text{Neg}\}$. A tweet $d_i = (w_1, \ldots, w_{n_i})$ is a sequence of n_i tokens which are part of a vocabulary $\mathcal{V}$. Using $\mathcal{D}_{\text{train}}$ we aimed to find a function $f : \mathcal{V}^* \rightarrow \{\text{Pos}, \text{Neg}\}$ that is able to correctly predict the sentiments of tweets not observed during training. More concretely, we wanted f to maximise the expected prediction accuracy:

$$\mathbb{E}_{(d,y)} \left[\mathbf{1}_{[f(d)=y]} \right]$$

where the expectation is over the real-world distribution of tweet-sentiment pairs (d, y). For practical reasons, we evaluated the accuracy of f on a finite set $\mathcal{D}_{\text{test}} = \{(d'_i, y'_i)\}_{i=1}^{N'}$ of N' test tweets $\mathcal{D}_{\text{test}}$ which are not part of $\mathcal{D}_{\text{train}}$

$$\text{Accuracy}(\mathcal{D}_{\text{test}}) = \frac{1}{|\mathcal{D}_{\text{test}}|} \sum_{(d,y) \in \mathcal{D}_{\text{test}}} \mathbf{1}_{[y=f(d)]}.$$

B. Transformer-based Models

For the actual sentiment classification task, we explore a transformer-based neural network architecture, namely the pre-trained base models of BERT and BERTweet. We applied dropout with rate 0.3 to their output and append a fully-connected layer with two nodes that outputs the class probabilities.

For fine-tuning, we use the AdamW optimizer [11] to minimise the negative log-likelihood. AdamW is a variant of the Adam optimizer [12] that allows for weight decay with the use of a scheduler and has been shown to generalize better compared to Adam [11]. We used a linear scheduler, which decreases the learning rate linearly to the number of steps.

C. Preprocessing

We explored various preprocessing methods found in the literature such as stop word removal and dictionary normalization [13]. When experimenting with transformer models, we replaced '<user>' and '<url>' with '@USER' and 'HTTPURL', respectively, following [8]. The replacement prevents tokenizers from treating tags as multiple tokens.

1) Punctuation Removal: To reduce the noise in the data set, we removed any non-alphanumeric character [13][14][15], except the pound symbol ('#'), to keep hashtags intact. We condensed multiple white spaces, which in part occur due to the removal of non-alphanumerical characters.

2) Stop Word Removal: Stop word removal aims to reduce noise in the data set by removing common words [13][16][17]. Libraries such as the NLTK [18] provide predefined lists of stop words. We found that treating negations such as "isn't" as stop words and removing them could change the sentiment of a tweet. We thus experimentally determined a list of stop words for our data set. This list is composed of words without a bias, i.e. occur with equal weight in all classes, explained in Section V-B2.

3) Dictionary Normalization: Dictionary normalization includes among others corrections, the replacement of slang and abbreviations with formal English or the replacement of special tokens such as '<user>' [13][19][20].

Existing work in dictionary normalization uses either fixed dictionaries [21], or learnt normalizations [22][23][24]. Here, we used the former approach. Through investigating the misclassified tweets of our BERTweet baseline model, we found that common Twitter abbreviations [25] generally have a higher misclassification rate than the model itself. We thus constructed a normalization dictionary based on common Twitter abbreviations, described in Section V-B3.

4) Subword Units: Subword units have been proposed to solve the open dictionary problem in neural machine translation [26]. This aids the recognition of rare words in a corpus. As this technique is used in BERTweet [8], we also adopted this approach. Section V-B4 describes our experiments relating to this.

D. Translation

As Section III notes, we found that our data set contains non-English and mixed-language tweets. To our knowledge, most of the literature removes such tweets in the initial cleaning of the data. As a novel contribution, we sought to tackle this problem in a proof-of-concept attempt by using the Google Translate API [27] to translate a subset of the validation tweets which are misclassified by our baseline BERTweet model as well as the entire test set. We then reclassified the translated tweets without retraining the model to measure the viability of this approach. Section V-C discusses our findings.

V. Experiments & Results

A. Baselines

1) Logistic Regression: We trained a Logistic Regression model using a one-hot encoding with a maximum of 10^4 features. We set the inverse regularisation strength to 10^5 and used the L-BFGS solver [28]. We performed no hyperparameter tuning to obtain a clean comparison of preprocessing methods as described in Section V-B.

2) GloVe: We conducted an experiment based on GloVe embeddings [29] that have been trained on 2B tweets[1]. For each tweet, we calculated the average embedding over its

[1] https://nlp.stanford.edu/data/glove.twitter.27B.zip

METHOD	VALIDATION ACCURACY
LOG. REGRESSION	80.44%
GLOVE	82.94%
BERT W/O FREEZING	88.98%
BERTWEET W/O FREEZING	90.58%
BERTWEET W/ FREEZING	**91.19%**

Table I
COMPARISON OF CLASSIFICATION METHODS

tokens leading to a feature vector in $\mathbb{R}^{200}$, which is used to train a neural network with two hidden layers both consisting of 512 units. The network is trained for 10 epochs with batch size 32 and the AdamW optimizer [11] with an initial learning rate of 10^{-4}, linearly reduced towards zero.

3) BERT & BERTweet: Both models are trained following the methodology in Section IV-B but without weight freezing. We also trained BERTweet with weight freezing.

All baseline results can be found in Table I.

B. Preprocessing

1) Punctuation Removal: We followed the methodology described in Section IV-C1. However, for technical reasons, before we applied the aforementioned methodology, we replace '<user>' and '<url>' with 'HTTPURL' and '@USER', respectively. This allowed us to easily remove the symbols '<' and '>', which, for instance, occur due to emoticons such as '<3' being split into '<' and '3'.

2) Stop Word Removal: As described in Section IV-C2, we constructed a custom list of stop words. For this, we computed the number of occurrences in positively and negatively classified tweets for every token in the data set. Then, we selected tokens that occur one half of a standard deviation above the mean overall frequency, i.e. positive and negative combined. This was to filter out the relatively rare token that may not be discriminatory, i.e. occur in each class with effectively equal frequency, but occur only in few tweets. Finally, we only kept those tokens where the difference of frequency between the classes is below 10^{-1}, a hyperparameter we have chosen experimentally. This process determined a list of 38 stop words we removed from the data set before training.

3) Dictionary Normalization: Section IV-C3 describes that we have found through investigation of tweets misclassified by BERTweet, that common Twitter abbreviations [25] tend to have a higher misclassification rate than the overall model. As such, we constructed our normalization dictionary by referring to the common Twitter abbreviations [25] and filtering out candidates in three steps.

Firstly, we only considered abbreviations where the abbreviation or the full-text version have a higher misclassification rate than the model. Secondly, we only considered abbreviations less common in the data set than the full-text versions. This eliminates abbreviations such as 'yolo' for which the full text 'you only live once' is rarely used. Lastly, we selected only abbreviations with a higher misclassification rate than their full-text versions, so we only replace where we expect an improvement. This left us with 21 replacement pairs which we applied before training our models.

METHOD \ MODEL	LOG. REGRESSION	BERTWEET
NO PREPROCESSING	80.44%	91.19%
PUNCTUATION REMOVAL	**80.64%**	89.73%
STOPWORD REMOVAL	80.41%	90.97%
DICTIONARY NORM.	80.42%	**91.21%**
SUBWORD UNITS	80.30%	91.10%

Table II
PREPROCESSING VALIDATION ACCURACY COMPARISON

4) Subword Units: We followed suit with the application within BERTweet [8] and trained `fastBPE` on our training set with the operations hyperparameter set to $64\,000$. We then applied the learnt subword types to our entire data set.

All preprocessing experiments were conducted with our baseline Logistic Regression and BERTweet with layer-freezing. The results can be found in Table II.

C. Translation

As described in Section IV-D, we used the Google Translate API [27] for our translation experiments. For this, we picked a random sample of around $29\,000$ validation tweets misclassified by our BERTweet baseline model to keep the workload small enough. We then classified the language of every tweet within this random sample and the entire test set. Afterwards, we translated every tweet classified as non-English into English and used the same BERTweet baseline model to reclassify the sentiment of the translated tweets.

Within this limited set of around $39\,000$ tweets or approx. 1.73% of all tweets, we already classified 63 different languages. Of the around $29\,000$ misclassified validation tweets, 559 or approx. 1.9% have not been classified as English and of those only 155 or approx. 0.53% have a changed sentiment prediction after reclassification. This is partly because a subset of the non-English tweets could not be translated properly. Despite the small number of actually translated and differently classified tweets, we found that we were able to improve upon the baseline performance by approx. 0.03% on the validation data. Unfortunately, we cannot report an improvement on the test data. Here we found that 122 tweets or 1.22% of the test data have been classified as non-English, but only 23 tweets or 0.23% have a changed sentiment prediction and again, a subset of tweets could not properly be translated.

D. Extended Model Architecture

We also consider an extension of the network architecture presented in Section IV-B which is shown in Figure 1. We added two fully-connected layers with 256 and 128 nodes between the transformer model and the classification layer.

Both layers use ReLU activation functions and their inputs have dropout with rate 0.3. BERTweet is chosen as the transformer model and the whole network is trained with weight freezing as in Section IV-B. This architecture achieves a validation accuracy of 91.21%. This slight improvement over the previous architecture may be explained by statistical noise and further experimentation would be required to get a more conclusive result. Nevertheless, it performs better on the test set, hence we selected the extended version for the Kaggle competition, achieving an accuracy of 91.88%.

VI. Discussion

From Table I we see that the performance of the transformer models BERT and BERTweet are superior to both logistic regression with one-hot encodings and a neural network with average GloVe embeddings. Moving from BERT to BERTweet leads to an improved performance, which is expected, as the latter was specifically developed for and trained on Twitter data. A similar improvement is achieved by weight freezing, leading to a validation accuracy of 91.19%. With the additional architecture extension we achieved a validation accuracy of 91.21%.

As Table II shows, classical preprocessing techniques do not increase and often even decrease the performance of our BERTweet models. Dictionary normalization is the exception, but the test set performance of the BERTweet model without it is higher, so there is likely unaccounted statistical noise. The results in Table II hint at the pre-trained BERTweet embedding's ability to handle the noise present in Twitter data. To our knowledge, there is a lack of insight in the literature on the exact effect of classical preprocessing techniques on BERTweet or other transformer embedding models, as review papers used simpler models [13][30][31][32].

While for us a significant unexplored domain, we concede

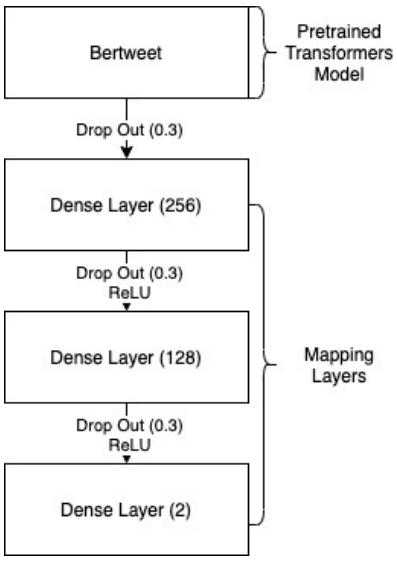


Figure 1. Final model architecture

that our experiments regarding non-English tweets remain inconclusive. We demonstrated that a performance gain is possible, but not consistently so. We attribute this to the very limited sample size of non-English tweets as shown in Section V-C. Pragmatic limiting factors in our research into this novel area were also the cost of even language detecting the entire data set and the abundance of foreign languages, also observed in Section V-C. This made the training of bespoke translation models far beyond the scope of this paper. Furthermore, we hypothesise that the in Section V-C observed issues with non-English tweets not being translated are due to spelling errors. There is serious evidence for poor spelling amongst the English tweets. Hence, expecting flawless spelling in the non-English ones appears unreasonable. While our BERTweet model seems to handle misspellings well due to being trained on Twitter data initially, translation models may struggle with these.

VII. Conclusions & Future Work

We saw that our weight freezing method for fine-tuning improves the sentiment classification accuracy of BERTweet. By looking into multiple standard preprocessing techniques, we saw a deterioration of performance, similarly observed by Nguyen et al. [8]. This could hint at the importance of punctuation marks and abbreviations for determining the sentiment of tweets. Additionally, we identified non-English tweets in our data set, which consists mostly of English tweets, as a source of misclassification. We tackled this problem by translating those tweets to English using the Google Translate API. To our knowledge, this topic is not as well researched as other topics in sentiment analysis. However, it seems to be a significant issue considering the many multilingual individuals that could mix multiple languages in one tweet. For such tweets, a model solely trained on English tweets is unlikely to reach the maximal possible accuracy.

For future work, we suggest to explore more involved weight freezing schedules in which the pre-trained weights are included in the updates gradually as opposed to our all-or-nothing approach. Examining the effect of our freezing method in more detail may give insights into its superiority over the regular fine-tuning approach. Moreover, it would be worth looking into classical preprocessing steps and their effects on BERTweet and other transformer models more closely, in order to definitively decide if they can lead to any improvements. Another possible future approach is to work more on the non-English and mixed language tweets. One could explicitly train a translation model so that we can do sentiment analysis on the translated tweets with a model that is trained for English tweets. This approach may be improved by spell correcting the non-English tweets to achieve better results from the translation model. Another approach would be to train a BERTweet embedding specifically on the mixed language tweets.

REFERENCES

[1] S. Boon-Itt and Y. Skunkan, "Public perception of the covid-19 pandemic on twitter: Sentiment analysis and topic modeling study," *JMIR Public Health and Surveillance*, vol. 6, no. 4, p. e21978, 2020.

[2] W. Medhat, A. Hassan, and H. Korashy, "Sentiment analysis algorithms and applications: A survey," *Ain Shams Engineering Journal*, vol. 5, no. 4, pp. 1093–1113, 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2090447914000550

[3] L. Zhang, S. Wang, and B. Liu, "Deep learning for sentiment analysis: A survey," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 8, no. 4, p. e1253, 2018.

[4] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[5] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need." [Online]. Available: http://arxiv.org/abs/1706.03762

[6] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman, "Glue: A multi-task benchmark and analysis platform for natural language understanding," 2018, cite arxiv:1804.07461Comment: https://gluebenchmark.com/. [Online]. Available: http://arxiv.org/abs/1804.07461

[7] A. Wang, Y. Pruksachatkun, N. Nangia, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman, "Superglue: A stickier benchmark for general-purpose language understanding systems," 2019, cite arxiv:1905.00537Comment: NeurIPS 2019, super.gluebenchmark.com updating acknowledegments. [Online]. Available: http://arxiv.org/abs/1905.00537

[8] D. Q. Nguyen, T. Vu, and A. T. Nguyen, "Bertweet: A pre-trained language model for english tweets," *arXiv preprint arXiv:2005.10200*, 2020.

[9] A. Brock, T. Lim, J. M. Ritchie, and N. Weston, "FreezeOut: Accelerate training by progressively freezing layers." [Online]. Available: http://arxiv.org/abs/1706.04983

[10] J. Lee, R. Tang, and J. Lin, "What would elsa do? freezing layers during transformer fine-tuning." [Online]. Available: http://arxiv.org/abs/1911.03090

[11] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," 2019.

[12] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization." [Online]. Available: http://arxiv.org/abs/1412.6980

[13] G. Angiani, L. Ferrari, T. Fontanini, P. Fornacciari, E. Iotti, F. Magliani, and S. Manicardi, "A comparison between preprocessing techniques for sentiment analysis in twitter." in *KDWeb*, 2016.

[14] R. Roychoudhury and S. K. Naskar, "Fine-tuning bert to classify covid19 tweets containing symptoms," in *Proceedings of the Sixth Social Media Mining for Health (# SMM4H) Workshop and Shared Task*, 2021, pp. 138–140.

[15] E. C. Ates, E. Bostanci, and M. S. Guzel, "Comparative performance of machine learning algorithms in cyberbullying detection: Using turkish language preprocessing techniques," *arXiv preprint arXiv:2101.12718*, 2021.

[16] A. Krouska, C. Troussas, and M. Virvou, "The effect of preprocessing techniques on twitter sentiment analysis," in *2016 7th International Conference on Information, Intelligence, Systems Applications (IISA)*, 2016, pp. 1–5.

[17] S. Hiremath, S. Manjula, and K. Venugopal, "Predictive analytics for sentiment classification of social media data using deep neural network," *Annals of the Romanian Society for Cell Biology*, pp. 5769–5778, 2021.

[18] S. Bird, E. Klein, and E. Loper, *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.", 2009.

[19] J. Eisenstein, "What to do about bad language on the internet," in *Proceedings of the 2013 conference of the North American Chapter of the association for computational linguistics: Human language technologies*, 2013, pp. 359–369.

[20] B. Han, P. Cook, and T. Baldwin, "Lexical normalization for social media text," *ACM Trans. Intell. Syst. Technol.*, vol. 4, no. 1, Feb. 2013. [Online]. Available: https://doi.org/10.1145/2414425.2414430

[21] E. Aramaki, "Typo corpus," 2010, Accessed: 2021-07-08. [Online]. Available: http://luululu.com/tweet/

[22] B. Han, P. Cook, and T. Baldwin, "Automatically constructing a normalisation dictionary for microblogs," in *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*, 2012, pp. 421–432.

[23] F. Liu, F. Weng, and X. Jiang, "A broad-coverage normalization system for social media language," in *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2012, pp. 1035–1044.

[24] D. Pennell and Y. Liu, "Toward text message normalization: Modeling abbreviation generation," in *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2011, pp. 5364–5367.

[25] V. Beal, "Twitter Dictionary: A Guide To Understanding Twitter Lingo," 2010, Accessed: 2021-07-08. [Online]. Available: https://www.webopedia.com/reference/twitter-dictionary-guide/

[26] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword units," *arXiv preprint arXiv:1508.07909*, 2015.

[27] "Google Cloud Translation," Accessed: 2021-07-22. [Online]. Available: https://cloud.google.com/translate/docs

[28] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu, "A limited memory algorithm for bound constrained optimization," *SIAM Journal on scientific computing*, vol. 16, no. 5, pp. 1190–1208, 1995.

[29] 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)*. Association for Computational Linguistics, pp. 1532–1543. [Online]. Available: http://aclweb.org/anthology/D14-1162

[30] A. Kadhim, "An evaluation of preprocessing techniques for text classification," *International Journal of Computer Science and Information Security,*, vol. 16, pp. 22–32, 06 2018.

[31] A. Go, R. Bhayani, and L. Huang, "Twitter sentiment classification using distant supervision," *CS224N project report, Stanford*, vol. 1, no. 12, p. 2009, 2009.

[32] A. Bifet and E. Frank, "Sentiment knowledge discovery in twitter streaming data," in *Discovery Science*, B. Pfahringer, G. Holmes, and A. Hoffmann, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 1–15.

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):

Twitter Sentiment Classification on Ice: Fine-tuning BERTweet with Layer-Freezing

Authored by (in block letters):

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

Name(s):	**First name(s):**
Jiang	Zhenjie
Müksch	Sebastian
Fringeli	Gabriel
Bäbler	Nathalie

With my signature I confirm that

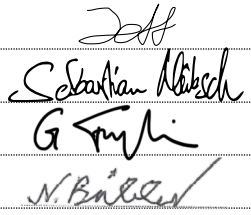
- I have committed none of the forms of plagiarism described in the 'Citation etiquette' information sheet.
- I have documented all methods, data and processes truthfully.
- I have not manipulated any data.
- I have mentioned all persons who were significant facilitators of the work.

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

Place, date

Zürich, 26/07/2021

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.