An Ensamble for Twitter sentiment analysis

Filippo Ficarra {fficarra@ethz.ch}[1]
Gaia Di Lorenzo {gdilorenzo@ethz.ch}[1]
Marco Calzavara {mcalzavara@ethz.ch}[1]
Paul Doucet {pdoucet@ethz.ch}[1]

[1]*Departement of Computer Science, ETH Zürich*

***Abstract*—With the widespread use of social media platforms, Twitter has emerged as a significant platform for users to express their opinions, emotions, and sentiments on various topics. Analyzing these sentiments in real-time can provide valuable insights for businesses, policymakers, and researchers. This paper presents an ensemble-based approach for sentiment analysis of tweets, harnessing the power of multiple models to capture diverse linguistic and contextual features of the text.**

I Introduction

The transformer architecture has revolutionized natural language processing, with large language models leading the way in achieving state-of-the-art performance on many tasks. Sentiment analysis has garnered significant attention, along with GPT-like chatbots, for its various applications in industry and politics.

Sentiment analysis encounters various obstacles, such as dealing with typos, slang, and sarcasm. Tweets, in particular, are significantly impacted by these factors due to their colloquial nature. Unlike books or articles, tweets often contain informal language, including slangs, hashtags, and errors, which further complicate the analysis process. These elements present unique challenges when attempting to determine sentiment from Twitter content.

The initial breakthrough in NLP involved the utilization of pre-trained models like GloVe [1] and Word2Vec [2] to generate word representations capable of encoding semantic relationships between words. However, both Word2Vec and GloVe produce static word embeddings, wherein each word's representation remains detached from the overall context of the sentence or document. This limitation poses challenges when a word has multiple meanings depending on its context.

A groundbreaking leap in natural language processing came with the introduction of the Transformer architecture [3]. The model from Vaswani et al. is based on the self-attention mechanism, which allows for contextualized word embeddings. The introduction of attention mechanisms increased the model's ability to take into account the connections between words within a sentence, leading to remarkable performance and surpassing prior constraints.

With our model we aim at improving reliability of the model when fine-tuned. We do so by combining BERT-like models and the CLIP text encoder, with the goal of incorporating different representations into the final prediction.

II Related work

The Transformer architecture [3] served as a cornerstone in the development of the BERT [4] (Bidirectional Encoder Representations from Transformers) model. This approach revolutionized the field by enabling bidirectional language modeling and contextual understanding, which allowed BERT to achieve state of the art performance on different tasks.

Considerable efforts have been devoted to enhancing reliability and performance of fine-tuned models on downstream tasks. Notably, research has focused on improving robustness during training, as seen in SMART [5], and enhancing the self-attention mechanism, as demonstrated in ”Fine-tune BERT with Sparse Self-Attention Mechanism” [6]. The primary objective of the latter endeavors has been to optimize the self-attention mechanism, thereby reducing irrelevant or less relevant word connections. In their work, Cui et al. adapted the encoder self-attention using the sparsegen-lin transformation [7], with the goal of making the attention-scores matrix sparse.

III Our Model

Our model is a text-driven ensamble of three different models: BERTweet, BERTweet with sparsemax and CLIP. Section III-A, III-B and III-C describe the three models, while section III-E describes the ensambler which combines the output of each model to produce the final prediction.

A. BERTweet

BERTweet [8] is a model based on BERT, but whose pre-training procedure is that used to train RoBERTa [9] (A Robustly Optimized BERT Pretraining Approach), a variant of the BERT model. BERTweet is trained on a corpus of 850M tweets, hence it is geared towards contextual understanding of tweet-like text.

B. BERTweet with SparseMax

Following the footsteps of Cui et al. [6], BERTweet with SparseMax aims at improving the self-attention mechanism of the BERTweet [8] transformer architecture by modifying specific layers. The architecture is designed to prioritize crucial word connections, thereby enhancing the model's performance. The self-attention mechanism calculates connection probabilities using the softmax function, as depicted in Equation (1):

$$\text{softmax}_i(z) = \frac{\exp(z_i)}{\sum j \exp(z_j)} \tag{1}$$

Equation (1) ensures that the generated probabilities are strictly positive, thereby maintaining the availability of all connections for the model. Sparsemax removes some connections by setting attention scores to 0.

The concept employed in BERTweet with SparseMax is inspired by Fine-tune BERT with sparse self-attention mechanism" [6], which modifies the encoder self-attention using the sparsegen-lin transformation [7].
Furthermore, BERTweet with SparseMax leverages the Sparsemax transformation [10] to achieve sparsity.
Let $\Delta^{K-1} := \{\mathbf{p} \in \mathbb{R}^K | \mathbf{1}^T\mathbf{p} = 1, \mathbf{p} \geq \mathbf{0}\}$ be the (K-1) dimensional simplex. The function proposed by Martins and Astudillo is the following:

$$\text{sparsemax}(z) = \underset{\mathbf{p}\in\Delta^{K-1}}{\arg\min} \|\mathbf{p} - \mathbf{z}\|^2 \tag{2}$$

It has been shown that Equation (2) possesses most of softmax properties while being simultaneously sparse.
Three different version of the model has been trained: Sparsemax in the first and the last layer, Sparsemax in the first 2 layers, Sparsemax in the last 2 layers.

The adjustments have been implemented to take advantage of distinct semantic levels in our sentences. Changes to the initial layers aim to enhance word-level characteristics, while changes to the last layers have the goal of grasping more advanced, contextual information.

C. CLIP

The CLIP (Contrastive Language-Image Pretraining) model is a groundbreaking neural network architecture that has garnered significant attention in the field of computer vision. CLIP is a novel approach to multimodal learning, where it leverages a vast dataset containing images and their corresponding natural language descriptions to learn joint representations for both modalities. Unlike traditional models that require extensive domain-specific annotations, CLIP is trained in a contrastive manner, enabling it to learn to associate images and texts without the need for paired data. This unique capability empowers CLIP to generalize effectively across a wide range of tasks. The versatility and performance of the CLIP model have sparked numerous exciting applications in various domains, making it a prominent candidate for multimodal research and deployment in real-world scenarios.

In our model, we undertake fine-tuning of CLIP text embeddings to harness cross-modal knowledge and potentially capture previously unseen features. By refining the text embeddings through this process, we aim to enhance the model's ability to bridge the gap between different modalities and extract valuable insights from the data, ultimately boosting its performance on various tasks. The fine-tuning of CLIP text embeddings allows us to leverage the rich semantic relationships encoded in the joint image-text representation, enabling the model to discover and leverage latent associations between visual and textual elements. As a result, we expect our approach to unlock new perspectives and foster a deeper understanding of the data, leading to improved performance and more robust generalization across diverse domains.

D. Classifier

The prediction are given by a linear layer of size $\mathbb{R}^{hidden\_size\times 2}$ that outputs the probabilities of the corresponding labels.
For BERTweet-based models, the classifier is a linear layer with size $\mathbb{R}^{768\times 2}$, where 768 is the length of the CLS token in BERT, while for CLIP the classifier is a linear layer of size $\mathbb{R}^{512\times 2}$, where 512 is the length of CLIP text embeddings.

E. Ensambler

Ensembles are popular in machine learning to increase reliability and robustness, it's especially useful when the classifiers used in the ensemble focus on disjoint ideas and architectures, i.e. the sparsemax model focus on strong pairs of words whereas Bertweet, and CLIP focus on each part of the tweet in a more "dense" manner, i.e. even less important words will have some weight. All these differences make each of these models unique, and allow for a model to correct for the others mistakes.

The ensambler is made of three layers: an embedding layer, which generates the input embeddings, a transformer encoder layer, which is a self-attention layer followed by a fully-connected layer, and a linear layer, which produces three weights. The weights are used to generate a linear combination of the outputs of the models in the ensamble. This model learsn to assign weights to the models in the ensamble based on text, i.e., it learns which model perform best based on text.

IV Our task

A. Data

In our task, we were assigned 2.5 million tweets to classify either as positive (1) or negative (-1) in equal proportions. The positive tweets were labeled based on the presence of ":)", while negative tweets were identified by the occurrence of ":(". However, this labeling approach presents challenges, as sarcasm can lead to misinterpretations. For instance, a tweet like "I can't believe I failed the exam and now my GPA is ruined :)" could be sarcastic, expressing a negative sentiment despite the positive emoticon. Due to the noise introduced by the labeling procedure, our efforts have been focused on reducing noise and increase robustness. We tried to achieve both through the architecture and during training.

The complete dataset contains duplicated sentences with different labels, and we removed those as a preprocessing step.

B. Preprocessing

We employ several preprocessing steps. First, we replace emoticons with special tokens, the same used by BERTweet tokenizer. Then, we remove some special characters, and we split the hashtags. This step aims at enhancing the understanding of hashtags. Finally, we replace <user> with "@USER" and <url> with "HTTPURL" to maintain compatibility with BERTweet tokenizer, and we remove words which have less than 10 occurrences and contain both characters and numbers, as we observed that these words are mainly ids and serial numbers, to reduce noise.

C. Baselines

Nowadays, most state-of-the-art models for sentiment analysis are deep neural networks containing millions (if not billions) of parameters. Before experimenting with such models, we tried different simpler approaches in order to analyse drawbacks and advantages. When experimenting with the baseline models, it's particularly necessary to visualise wrongly predicted tweets in order to make an educated guess on which part of the model to improve. We then try to modify our model based on the previous finding during a profound analysis of our predictions.

1) GloVe with Logistic Regression (first baseline)

We tried to keep our first baseline model as simple as possible in order to build on top of it later. In the first baseline we tokenized tweets and assigned an embedding to each word based on a pre-trained GloVe [1] look up table. Then, a simple Logistic Regression model was applied to these embeddings. Despite being straightforward, this method quickly showed its limitations with the accuracy being bounded to 0.716. It's pretty clear that the sentiment conveyed by a tweet isn't always determined by a linear relationship between words and that the capacity of our first baseline model isn't large enough to account for even the most simple sentences. This is shown by 1, which plots the accuracy in function of the tweet length. We observe that the first baseline struggles on longer tweets, as the number of non linearities increases. This further proves the importance of a model that takes context into account as the accuracy will be immediately improved.

2) GloVe with bidirectional LSTMs (second baseline)

GloVe makes a very optimistic assumption, as it is a context indifferent method, so we already know that this would be a direction to explore for future improvement. When retrieving the GloVe embeddings from a pre-trained association between words and embeddings, the context is not taken into account, i.e. a word will always have the same embeddings even if the context is different. When performing sentiment analysis this can be problematic as, for example, the word "strike" will always have the same embedding in any context even though its meaning and positivity/negativity can vary substantially ("missile strike" vs "baseball strike").

Our GloVe baseline classifies the following as negative instead of positive:
"<user> can't wait for the day malcolm cooks for me lol" (positive sample 10992)

This is a perfect example of why context matters: "can't" and "wait" taken separately convey a negative sentiment but "can't wait" expresses a positive feeling. Therefore, by embedding with GloVe, even multiple layers of bidirectional LSTMs will miss some of the context and will be less accurate and extremely limited compared to an embedding method that fully exploits context. Even our fine-tuned LSTM model obtained a maximum of 0.777 accuracy, which is nowhere near what a context-aware embedding is capable of.

V Results

In our study, we conducted a comprehensive comparison between the baselines, individual models, and our final model. As anticipated, among the individual models, Bertweet exhibited the best performance. While Bertweet and Bertweet with sparse-max achieved similar results, CLIP lagged behind after an equal number of epochs. We attribute this outcome, with a high degree of certainty, to the distinct nature of the training corpus, as tweets possess peculiarities like slang, mistakes, and hashtags, which are uncharted territory for CLIP. Nevertheless, we observed that CLIP effectively mitigated common mistakes across the models, resulting in an improved overall score on the validation set. This demonstrates CLIP's ability to contribute valuable insights despite facing challenges posed by the idiosyncrasies of tweet data.

Reported in the table are the accuracies on the evaluation set on 10 thousand samples.

The BERTweet model has been trained for three epochs, with batch size of 256 and learning rate 1e-5. The best models have been chosen using the lowest validation loss among the epochs to avoid overfitting. We trained the CLIP model for three epochs, with also learning rate of 1e-5 and batch size 264.

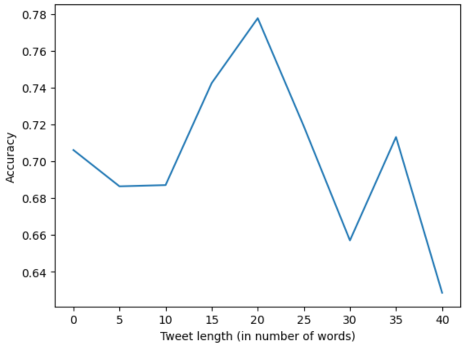

Figure 1. Baseline 1 accuracy in function of the length of tweets (in words) for the evaluation set

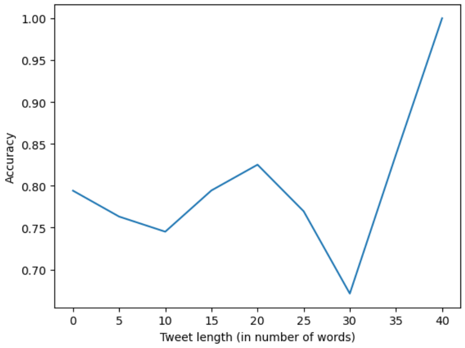

Figure 2. Baseline 2 accuracy in function of the length of tweets (in words) for the evaluation set

The results have been obtained using a balanced evaluation set with 10k samples and no duplicates.

Model	Accuracy
GloVe with Logistic Regression (Baseline 1)	71.6
GloVe with bi-directional LSTMs (Baseline 2)	77.7
Bertweet	92.50
Bertweet-with-sparsemax-last-2-layers	92.45
Bertweet-with-sparsemax-first-2-layers	91.95
Bertweet-with-sparsemax-first-last-layer	91.79
Clip	87.75
Ensemble	**92.62**

Table I
ACCURACY ACROSS OUR DIFFERENT MODELS.

In future research, the insights gained from CLIP's application to sentiment analysis could be further explored on different types of corpora that align more effectively with the model's capabilities. Additionally, a promising avenue for investigation involves implementing the ensemble attention layer at the embedding level, as opposed to the logit level. This approach has the potential to yield enhanced performance and uncover novel perspectives in sentiment analysis tasks.

References

[1] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation." vol. 14, pp. 1532–1543, 2014.

[2] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," vol. 26, 2013. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2013/file/9aa42b31882ec039965f3c4923ce901b-Paper.pdf

[3] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017.

[4] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2018, cite arxiv:1810.04805Comment: 13 pages. [Online]. Available: http://arxiv.org/abs/1810.04805

[5] H. Jiang, P. He, W. Chen, X. Liu, J. Gao, and T. Zhao, "SMART: Robust and efficient fine-tuning for pre-trained natural language models through principled regularized optimization," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2020. [Online]. Available: https://doi.org/10.18653%2Fv1%2F2020.acl-main.197

[6] B. Cui, Y. Li, M. Chen, and Z. Zhang, "Fine-tune BERT with sparse self-attention mechanism," pp. 3548–3553, Nov. 2019. [Online]. Available: https://aclanthology.org/D19-1361

[7] A. Laha, S. A. Chemmengath, P. Agrawal, M. M. Khapra, K. Sankaranarayanan, and H. G. Ramaswamy, "On controllable sparse alternatives to softmax," 2018.

[8] D. Q. Nguyen, T. Vu, and A. Tuan Nguyen, "BERTweet: A pre-trained language model for English tweets," pp. 9–14, Oct. 2020. [Online]. Available: https://aclanthology.org/2020.emnlp-demos.2

[9] 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," 2019, cite arxiv:1907.11692. [Online]. Available: http://arxiv.org/abs/1907.11692

[10] A. F. T. Martins and R. F. Astudillo, "From softmax to sparsemax: A sparse model of attention and multi-label classification," 2016.