Boosting Transformer Performance for Multi-Label Sentiment Analysis in Short Texts

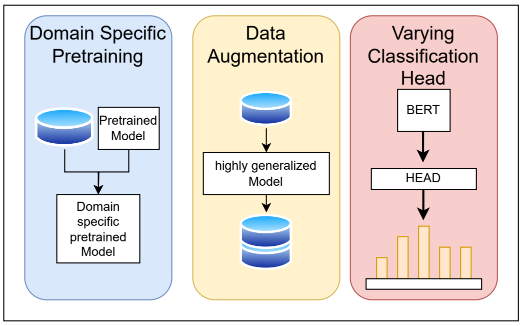
Julius Neumann
Matriculation number
Module
julius.neumann@mailbox.tu-dresden.de

Konstantin Wrede
Matriculation number
Module
email@domain

Robert Lange
Matriculation number
Module
robert.lange4@mailbox.tu-dresden.de

Abstract

Sentiment classification in short-text datasets presents unique challenges, including class imbalance, limited training samples, and the subjective nature of sentiment labels. In this study, we evaluate the performance of transformer-based models, such as BERT and RoBERTa, on a multi-label sentiment classification task. Our approach investigates three key factors: (1) further pre-training on domain-specific data, (2) augmentation of the training set with generated examples, and (3) variations in classification head architectures.



Domain Specific Pretraining
Pretrained Model
Domain specific pretrained Model
Data Augmentation
highly generalized Model
Varying Classification Head
BERT
HEAD

Figure 1: Abstract overview of the proposed mechanisms to increase the performance of transformer based short text classifiers

1 Introduction

Instructions:

1. Clone/copy this file and fill it with your content.

2. Write no more than 8 pages (references do not count towards this).

3. In "Introduction", briefly lay out the problem you address, and your contribution. Include an overview/teaser image.

4. In "Related Work", reference both (1) a selection of previous works on the same/similar problems (and try to differentiate your approach from those), (2) a set of foundational literature relevant to the problem, and your methodology.

5. "Methodology" lays out your technical approach, high-level, as well in technical detail. Subsections are highly recommended here (and also elsewhere).

6. "Evaluation" should contain both a quantitative and a qualitative evaluation of your results. If in doubt about metrics and evaluation methodologies, talk to us.

7. "Discussion" on the one hand builds upon the evaluation, and should critically discuss strengths and weaknesses of your solution, and possible ways to improve it further. On the other hand, it should discuss relevant ethical questions related to the problem and/or your solution at hand.

The increasing demand for effective Natural Language Processing (NLP) solutions has driven significant advancements in leveraging pre-trained transformer models such as BERT and RoBERTa for downstream tasks. In this work, we focus on the problem of multi-label sentiment classification in short text data, a challenging domain due to the inherent subjectivity and class imbalances present in the dataset.

Our contribution lies in evaluating the impact of three factors on classification performance: (1) further domain-specific pre-training of models, (2) augmentation of training data with generated examples, and (3) variations in the architecture of the classification head. By systematically exploring these elements, we aim to identify strategies that maximize performance while addressing challenges such as limited training data and class imbalance.

We demonstrate our methodology using a dataset comprising 2,768 examples with five sentiment labels: Anger, Fear, Joy, Sadness, and Surprise. Through quantitative evaluation metrics such as F1 Score and Macro F1 Score, as well as qualitative methods like SHAP analysis, we provide insights into model behavior and classification reasoning. The findings highlight both the potential and limitations of current approaches, offering valuable guidance for future research in NLP model optimization.

2 Related Work

To solve this NLP task we use a transformer model which is pre-trained on a large text corpus and then fine-tune this model for the specific task. Utilizing the broad language understanding of a pre-trained model to solve downstream tasks is a common approach that often yields good results for reasonable computational expense [19] [14] [20].

For the specific task of sentence classification the BERT model or one of its variants are often used [16] [8] [4] [8] because of their capability to capture contextualized representations of words and sentences.

Besides the used model the size of the training data set plays a significant role in performance. It shows that a larger amount of trainings samples can lead to better performance in NLP tasks [7] [11].

2.1 BERT Model

The BERT model was originally introduced by Devlin et al.. It has a bidirectional transformer encoder architecture with 12 Transformer blocks, 12 self-attention heads and a hidden size of 768 and was pre-trained on a large text corpus comprising the Toronto Book Corpus and Wikipedia. For the pre-training a combination of the masked language objective and next sentence prediction was used.

2.2 RoBERTa Model

Based on the BERT model Liu et al. introduced RoBERTa which is a robustly improved BERT pre-training approach. The key modifications of the training procedure are the inclusion of dynamic masking patterns, the removal of the next-sentence pre-training objective, training on longer sequences, larger mini-batches and improved learning rates. These changes lead to the RoBERTa model performing better than the standard BERT model in many cases [6] [9].

2.3 Text Classification

To utilize a pre-trained BERT model for a text classification task the hidden state of the first token of the output sequence [CLS] is used. The token is passed through a classifier to obtain the probabilities for all the possible labels. This classifier is often referred to as classification head. While fine-tuning the parameters of the classification head as well as the parameters of the BERT model are adjusted to maximize the log-probability of the correct label. According to Wolfe and Lundgaard it is possible to optimize the classification performance by varying the embedding size in the classification head as well as the classification head type. So we decided follow Stickland and Murray approach on implementing so called "Projected Attention".

2.4 Data Augmentation

The process of "selecting important samples, adding more useful data samples or adding extra information to adapt the model" to a task specific domain is in literature referred as data augmentation [5].

While Guo and Yu and Qu et al. discuss an data augmentation approach using adversarial training relying on heavily complex methods to implement and where not to promising to improve the models capability by adding examples that are more promising than other, existing ones.

Balkus and Yan themselves propose an "improved method of data augmentation", "expanding the coverage of the training data and help better capture unique edge cases within classes". Them, utilizing the at their time new GPT-3 model and its completion endpoint to generate similar text to a given text from the trainingsset and further filtering the created examples with an generic algorithm only allowing the most promising candidates to be chosen in the augmented few shot examples, achieved a big increase in accuracy of there classification task. To set this into perspective, the classification task of Balkus and Yan only had a very small training dataset of 26 examples and only 2 labels to choose from.

TODO (find the curve to our task)

To leverage an improved model performance and its significantly better contextual understanding of state of the art models like gpt4o, we where curious on how a rather big, generated unfiltered trainingset to train our Bert classification model on would perform.

3 Methodology

Furthermore different approaches to enhance the performance of the models are investigated.

Firstly the pre-trained models are further pre-trained on the domain specific data, as suggested in this paper [16]. This Theory is also backed by the study [12], where the pre-training of a classification model on a given text domain was significantly more promising than the increasing of the trainingsset beyond a certain threshold in its size.

Secondly the training is performed with different trainingset sizes. We decide to split the trainingsset and 1. only use half of the given training examples for fine-tuning, 2. use the whole set for training and 3. augment the trainingsset with unfiltered generated examples.

Lastly different architectures of the classification head are evaluated. As reported in [17] the structure of the feed-forward classification head can influence the performance of the model.

Furthermore a evaluation metric is proposed to compare the models´ performance against human evaluation.

3.1 Dataset Description

The dataset, provided by Codabench, comprises 2,768 examples, each containing a short English text labeled with one or more sentiment classes: Anger, Fear, Joy, Sadness, and Surprise. The labels are represented using one-hot encoding, allowing for multi-label classification. On average, each text consists of 78.4 characters, corresponding to approximately 15 words, which categorizes this task as a "short text classification" problem.

Label	**Frequency**	**Probability (%)**
Anger	333	12.0
Fear	1611	58.2
Joy	674	24.3
Sadness	878	31.7
Surprise	839	30.3

Table 1: Label Frequencies and Probabilities

The distribution of labeled classes in the training set is presented in Table 1. Notably, the dataset exhibits a significant class imbalance, with the "Fear" class being over-represented (58.2% of the total instances) and the "Anger" class being under-represented (12.0% of the total instances). This imbalance may introduce bias in the classification model, potentially leading to a higher propensity for predicting the "Fear" class and a reduced likelihood of correctly identifying instances of the "Anger" class.

3.2 Further Pre-training

To further pre-train the models as suggested by Sun et al. only the 'text' field of the training data set is used. The text samples are tokenized with a pre-trained tokenizer. For every model the corresponding tokenizer from Hugginface is used. The models are than trained on these text samples with the masked language modeling objective. Tokens are randomly masked with a probability of 15%. The Pre-training is done for 30 epochs with a learning rate of $2 * 10^{-5}$ and a weight decay of 0.01.

3.3 Increased Trainings-Set

In order to augment the given training examples with additional generated trainings examples, we are using the Open AI completion endpoint with an finetuned model of gpt4o-mini. The model was finetuned by the given openai endpoint for 3 Epochs on the splitted Trainingsset with its given Sentiment Labels and a System Prompt stating: "You create short texts with their corresponding sentiment labels. The sentiment labels are Anger, Fear, Joy, Sadness, and Surprise. The texts are in English. The texts are short, with a maximum length of 256 characters." To generate an additional training example we first provide the finetuned gpt4o-mini Endpoint with the same System prompt as used in finetuning as well with an intended output template using a json schema including an text field and an array of the assigned sentiments. This process is repeated until a desired number of additional examples is generated to augment the trainingsset. We decided on a reasonable number of 6000 additional trainings examples, to add about twice the amount of original trainings examples. This would leave us with roughly 9000 trainings examples. This represents about the threshold of 1000 to 10000 examples mentioned by Nguyen et al..

3.4 Variable Size of Classifier Head

We expand each pre-trained model with a custom classification head which consists of a linear layer, a dropout layer and a second linear layer. The first linear layer has as input size the hidden size h of the pre-trained model and as output size the internal classifier size c. The second linear layer has as input size the internal classifier size c and as output size the number of possible labels. To get each labels probability for the evaluation metric the Sigmoid function is used.

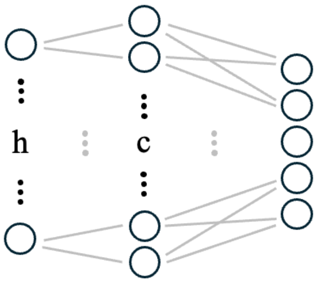


Figure 2: Visualization of the custom classifier head

3.5 Trainings-Setup

To see the impact of the different approaches two trails are run. Both times all four models are fine-tuned once without further pre-training and once with further pre-training on the domain specific data. But in the first trail only the original trainings-data is used for pre-training and fine-tuning. In the second trail the pre-training and fine-tuning is performed on the artificial expanded trainings-set. This way the influence of the model size and the trainings-set size are compared. To see if a bigger classification head enhances the models performance the best performing approach from the to trails is repeated with a step wise increase classification head size. As the starting point the classification head size is set equal to the hidden size of the pre-trained BERT or RoBERTa model. For every further trainings-run the classification head size is doubled. In all the runs the fine-tuning is done for 10 epochs with a learning rate of $2 * 10^{-5}$, a weight decay of 0.01 and a training and evaluation batch size of 6. Furthermore 500 warm up steps are used.

3.6 Evaluation Metrics

To evaluate the model's performance and compare it with other models and the Codabench challenge results [2], we used Accuracy, F1 Score, and Macro F1 Score. These metrics were computed on the development set provided by the Codabench challenge. The inclusion of Macro F1 Score ensures a fair evaluation across all sentiment classes, addressing class imbalance.

- **Accuracy**: A text is considered correctly classified only if all its labels are predicted correctly. Accuracy is calculated as:

  $$\text{Accuracy} = \frac{N_{correct}}{N_{total}}$$

  where $N_{correct}$ is the number of correctly classified texts and N_{total} is the total number of classified texts.

- **F1 Score**: Harmonic mean of precision and recall.

  $$\text{F1} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

  where:

  $$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \quad \text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

- **Macro F1 Score**: Average F1 Score across all classes

  $$\text{Macro F1 Score} = \frac{1}{N}\sum_{i=1}^{N} \text{F1}_i$$

  where N is the number of classes.

These metrics provide a robust evaluation of the model's performance, enabling meaningful comparisons with other models and the Codabench challenge results [2].

3.7 Explainability

To further understand why a model makes specific predictions, we created confusion matrices and further evaluated the reasoning behind the classification utilizing the python library SHAP (SHapley Additive exPlanations) introduced by Lundberg and Lee.

3.7.1 Confusion Matrix Analysis

To better gain insights into the models performance, we analyse the label class confusion matrices. This allows us to identify common misclassifications and potential biases, including valuable insights into the limitations of the models decision making.

3.7.2 SHAP Analysis

SHAP is using SHAP values to analyse the contribution scores of input features, we can determine the impact of specific words on the sentiment classification.

3.8 Implementation

4 Evaluation

4.1 Classification Results

4.2 Reasoning Results

4.3 Reviewer Agreement / Human Evaluation

While F1 scores below 0.9 and accuracies under 80% may appear suboptimal, it is important to recognize that the classification of multi-class labeled data, particularly in sentiment analysis, is inherently subjective. This subjectivity, combined with the high-dimensional complexity of the data, often prevents the achievement of very high accuracies.

To better understand the subjective nature of the labeling process, we engaged three persons to classify a small set of texts, taken from the validation set. This approach allowed us to measure the variability and agreement in their labeling, providing deeper insight into the inherent subjectivity of the task.

The Results... TODO

5 Discussion

6 Contribution statement

If you work in a team, describe here briefly who did what.

References

[1] Salvador V. Balkus and Donghui Yan. 2024. Improving short text classification with augmented data using gpt-3. *Natural Language Engineering*, 30(5):943–972.

[2] Codabench. 2024-25. Bridging the gap in text-based emotion detection - semeval 2025 task 11 - track a. Codabench. Accessed: 2025-01-13.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *Preprint*, arXiv:1810.04805.

[4] Andr'es Garc'ia-Silva, Cristian Berr'io, and Jos'e Manuel G'omez-P'erez. 2024. Space-ideas: A dataset for salient information detection in space innovation. In *International Conference on Language Resources and Evaluation*.

[5] Xu Guo and Han Yu. 2022. On the domain adaptation and generalization of pretrained language models: A survey. *Preprint*, arXiv:2211.03154.

[6] Dan Hirlea, Christopher Bryant, and Marek Rei. 2021. Contextual sentence classification: Detecting sustainability initiatives in company reports. *ArXiv*, abs/2110.03727.

[7] Yunjie Ji, Yong Deng, Yan Gong, Yiping Peng, Qiang Niu, L. Zhang, Baochang Ma, and Xiangang Li. 2023. Exploring the impact of instruction data scaling on large language models: An empirical study on real-world use cases. *ArXiv*, abs/2303.14742.

[8] Ning Liu and Jianhua Zhao. 2022. A bert-based aspect-level sentiment analysis algorithm for cross-domain text. *Computational Intelligence and Neuroscience*, 2022.

[9] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *Preprint*, arXiv:1907.11692.

[10] Scott M Lundberg and Su-In Lee. 2017. Shap. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

[11] Houman Mehrafarin, S. Rajaee, and Mohammad Taher Pilehvar. 2022. On the importance of data size in probing fine-tuned models. In *Findings*.

[12] Thi Huyen Nguyen, Hoang H. Nguyen, Zahra Ahmadi, Tuan-Anh Hoang, and Thanh-Nam Doan. 2022. On the impact of dataset size:a twitter classification case study. In *IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, WI-IAT '21, page 210–217, New York, NY, USA. Association for Computing Machinery.

[13] Yanru Qu, Dinghan Shen, Yelong Shen, Sandra Sajeev, Jiawei Han, and Weizhu Chen. 2020. Coda: Contrast-enhanced and diversity-promoting data augmentation for natural language understanding. *Preprint*, arXiv:2010.08670.

[14] Mark Sandler, Andrey Zhmoginov, Max Vladymyrov, and Andrew Jackson. 2022. Fine-tuning image transformers using learnable memory. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12145–12154.

[15] Asa Cooper Stickland and Iain Murray. 2019. Bert and pals: Projected attention layers for efficient adaptation in multi-task learning. *Preprint*, arXiv:1902.02671.

[16] Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. 2020. How to fine-tune bert for text classification? *Preprint*, arXiv:1905.05583.

[17] Prateek Verma. 2021. Attention is all you need? good embeddings with statistics are enough:large scale audio understanding without transformers/ convolutions/ berts/ mixers/ attention/ rnns or ....

[18] Cameron R. Wolfe and Keld T. Lundgaard. 2021. Exceeding the limits of visual-linguistic multi-task learning. *ArXiv*, abs/2107.13054.

[19] Lingling Xu, Haoran Xie, S. Joe Qin, Xiaohui Tao, and Fu Lee Wang. 2023. Parameter-efficient fine-tuning methods for pretrained language models: A critical review and assessment. *ArXiv*, abs/2312.12148.

[20] Shuo Yang and Gjergji Kasneci. 2024. Is crowdsourcing breaking your bank? cost-effective fine-tuning of pre-trained language models with proximal policy optimization. In *International Conference on Language Resources and Evaluation*.