Large Language Models Performance Comparison of Emotion and Sentiment Classification

William Stigall*
Kennesaw State University
Marietta, Georgia, USA
wstigall@students.kennesaw.edu

Md Abdullah Al Hafiz Khan†
Kennesaw State University
Marietta, Georgia, USA
mkhan74@kennesaw.edu

Dinesh Attota
Kennesaw State University
Marietta, Georgia, USA
dattota@students.kennesaw.edu

Francis Nweke
Kennesaw State University
Marietta, Georgia, USA
fnweke@students.kennesaw.edu

Yong Pei
Kennesaw State University
Marietta, Georgia, USA
ypei@kennesaw.edu

ABSTRACT

The increasing application of artificial intelligence in daily life necessitates precise emotion classification for improved user interactions in areas like healthcare, marketing, and customer service. This work explores the development of emotion classification algorithms, focusing on text classification and providing low latency inferencing for seamless application into persistent-state systems. We use a parallel multi-task learning approach, to learn multiple tasks with a single loss function, allowing it to learn representations in both Emotion Classification and Sentiment Analysis simultaneously. We present a detailed analysis of a fine-tuned $\text{BERT}_{\text{Tiny}}$ model $\text{EmoBERT}_{\text{Tiny}}$ for emotion and sentiment classification tasks, comparing its performance against baseline models and state-of-the-art 7B parameter models. $\text{EmoBERT}_{\text{Tiny}}$ is benchmarked against Llama-2-7B-chat and Mistral-7B-Instruct across Accuracy, F1-score, precision-recall curves and inference speed. $\text{EmoBERT}_{\text{Tiny}}$ outperforms pre-trained and state-of-the-art models across all metrics and computational efficiency, achieving 93.14% accuracy in sentiment analysis and 85.48% accuracy on emotion classification. $\text{EmoBERT}_{\text{Tiny}}$ processes a 256 token context window in 8.04ms on average post-tokenization, and 154.23ms on average in total processing speed.

CCS CONCEPTS

• **Machine learning model that learns underlying patterns to generate new data** → **Generative Models**; • **Time taken to perform an inference** → *Latency*; • **Predefined Instructions that guides model response** → System Prompt; • **Instruction given to model** → Prompt; • **Providing the model with a number of examples to assess performance** → Shot Testing.

*Lead Author
†Secondary contributor

This work is licensed under a Creative Commons Attribution International 4.0 License.
ACMSE 2024, April 18–20, 2024, Marietta, GA, USA
© 2024 Copyright held by the owner/author(s).
ACM ISBN 979-8-4007-0237-2/24/04.
https://doi.org/10.1145/3603287.3651183

KEYWORDS

Emotion Classification, Sentiment Analysis, Large Language Models, BERT, Llama, Mistral, NLP, Multitask Models

ACM Reference Format:
William Stigall, Md Abdullah Al Hafiz Khan, Dinesh Attota, Francis Nweke, and Yong Pei. 2024. Large Language Models Performance Comparison of Emotion and Sentiment Classification. In *2024 ACM Southeast Conference (ACMSE 2024), April 18–20, 2024, Marietta, GA, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3603287.3651183

1 INTRODUCTION

Human Emotions are complex and multi-faceted. Understanding human emotions requires nuanced understanding and interpretations of multiple aspects of human behavior. As artificially intelligent machines are further integrated into human livelihood, accurately classifying human emotions has profound implications in psychology, healthcare, marketing, customer service, and personal assistants [2, 8, 25]. In recent years, automated systems have been developed to recognize human emotions through various modalities, including speech analysis, facial recognition, and text-based methods [3, 30]. In the context of emotion and sentiment classification, 'sentiment' often refers to the overall polarity of text categorized as positive, neutral, negative, and occasionally as mixed. Emotion classification delves deeper. It categorizes emotions into more granular contexts, using labels such as 'joy', 'sadness', 'anger', and 'fear'. This finer distinction requires more sophisticated analysis and advanced computational methods, given the nuanced nature of human emotions. Persistent-state systems that operate in real-time rather than when prompted (Robotics, Interactive AI) necessitate paradigms that can process a continuous stream of information at low latency. The ability to accurately process and respond to human emotions becomes critical in systems that interact with human beings.

In the context of real-time applications, understanding human communication speeds is crucial. The American Society of Administrative Professionals reports the average typing speed is around 40 words per minute [22]. The fastest typer in history Barbra Blackburn achieved 212 words per minute using a simplified keyboard. [20] The average spoken word per minute is 150 words per minute [10] and a fast rate of speech is 200 words per minute [10]. To ensure seamless interaction across a range of real-time scenarios, including those without additional computational delays, a processing speed threshold of 0.25ms is considered optimal. This threshold

is designed to accommodate high-end users in terms of speech and typing speeds. For the average user in a text-only environment, 1.5 seconds is sufficient, however .6ms can support most advanced users.

Progress in machine learning and deep learning has been instrumental in tackling the challenges posed by the high-dimensional, sequential, and often imbalanced nature of natural language data. Our research focuses on the application of a Transformer-based model, specifically a fine-tuned **BERT$_{\text{Tiny}}$** model, for real-time emotion and sentiment classification. The **EmoBERT$_{\text{Tiny}}$** model, tailored for computational efficiency in accuracy demonstrates promising capabilities in high accuracy and low-latency inferences. We summarize our comparative study findings as follows.

- Our fine-tuned model **EmoBERT$_{\text{Tiny}}$** achieves 93.14% accuracy on sentiment analysis, and 85.46% accuracy on emotion classification; significantly outperforming baseline models like prajjwal1's pre-trained **BERT$_{\text{Tiny}}$** and pre-trained BERT-Base Cased, which scored 38.09% and 30.41% respectively on sentiment analysis and 15% and 8% on emotion classification respectively.
- **EmoBERT$_{\text{Tiny}}$** also outperforms SOTA models in both speed and accuracy. **EmoBERT$_{\text{Tiny}}$** outperforms Llama-2-7B by over 1000% on post-tokenization inferencing at an input length of 256.**EmoBERT$_{\text{Tiny}}$** performs inferencing on the entire 256 context window in 8.04ms on average.**EmoBERT$_{\text{Tiny}}$** can process and predict the sentiment and emotion classifications of untokenized text in 146ms on average, faster than our theoretical ideal processing speed for real-time applications.

2 RELATED WORK

In the evolution of emotion and sentiment analysis in AI, early linear classifiers while simple, struggled with the complex nonlinear data of language and emotion [18]. The early 2000s saw a shift towards Support Vector Machines for handling high-dimensional data [31], and then to Convolutional Neural Networks (CNNs) for image-based emotional classification through hierarchical feature learning [17]. The emergence of Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks enabled learning of long-term dependencies in sequential data like text and time-series [13, 23]. Recently, Transformer models [32] such as BERT [6] and GPT-3 [4] have set new benchmarks in NLP tasks with their parallel processing capabilities. Furthermore, Multi-Modal Emotion Recognition (MER) systems have significantly advanced affective computing and human-computer interaction by integrating visual auditory and textual inputs for a more nuanced understanding of human emotions, as shown in recent studies [1, 7, 19].

3 METHODOLOGY

This section discusses the details of our proposed large language model (LLM) based on emotion and sentiment recognition models. Our LLM-based recognition model comprises text preprocessing, LLM-enabled classification models, and a performance analysis model. Our overall methodology is shown in Figure 1.

3.1 Text Preprocessing

Our preprocessing module parses texts and cleans and tokenizes words to feed into the LLM model for feature representations. Data cleaning and label standardization during the preprocessing phase, we undertook a systematic approach to standardize emotion and sentiment labels. This involved consolidating labels that were either synonymous or exhibited similar emotional polarity. For instance, labels representing similar emotional states of varying intensities of the same emotion were grouped under a unified category. This process was guided by linguistic principles and existing psychological frameworks to ensure consistency and relevance to the emotional spectrum being analyzed.

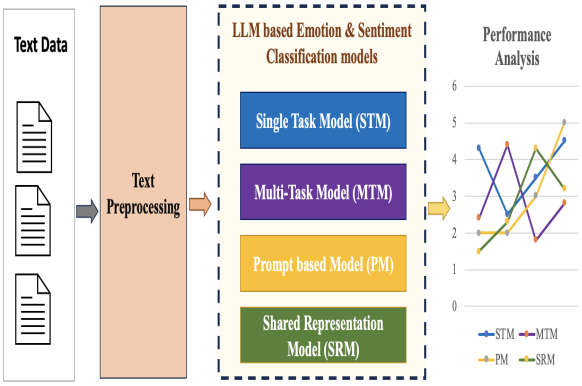

Figure 1: Our Large Language Model (LLM) based Sentiment and Emotion Classification Performance Comparison Framework

3.2 LLM Based Emotion and Sentiment Classification Models

To understand LLM-Based Sentiment and Emotion classification model, we developed four different types of LLM-enabled Recognition models - i) Single Task Model (STM), ii) Multi-Task Model (MTM), iii) Prompt based Model (PM) and iv) Shared Representation Model (SRM). We discuss each of these models below in subsequent sections.

3.2.1 Multi-Task Model (MTM). With our focus on computational efficiency and classification accuracy in this multi-task setting, we opted for a Transformer-based model [32]. Despite the significant evolutions in the field that have seen some state-of-the-art Transformer models encompass billions or trillions of parameters, the architecture's core advantage remains its ability to achieve superior performance with fewer parameters compared to LSTM and RNN models [32]. BERT stands for Bidirectional Encoder Representations from Transformers [6]; bi-directionality allows for richer context understanding during the training phase and boasts a high level of accuracy. For real-time applications, the need to process the text in both directions would lead to a higher computational cost than a unidirectional model. However, we believe the benefits of higher training accuracy during fine-tuning are worth the increase in overhead [6]. Why did we select BERTTiny specifically

as the model architecture? TinyBERT is 7.5 times smaller and 9.4x faster on inference than BERTBASE [16]. There are both computational power limitations on the machine used for fine-tuning and computational limitations on client machines that host the model to run inferences. Therefore, we opt for a smaller version of the BERT architecture that retains much of the performance (96.8% of BERTBASE on GLUE) [16]. TinyBERT is built on top of BERTTiny, making BERTTiny (I. Turc et al., 2019) an optimal choice to perform fine-tuning [29].

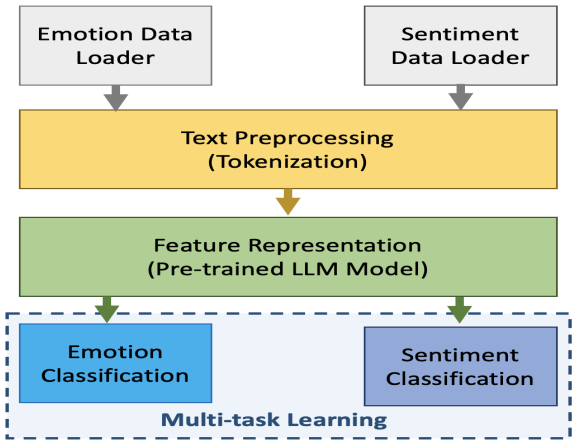

Figure 2: Our Overall Multitask (Sentiment, Emotion) Recognition Model

Figure 2 shows our overall multi-task models for the classification. Our model loads sentiment and emotion text data and tokenizes them to feed into the LLM (e.g., BERTtiny) model. We introduce two dense layers as task heads for Emotion Classification and Sentiment Analysis with (128, number of task labels) as the dimension for each dense layer at the end of the LLM model representation. We fine-tune this classification model using a custom loss function is used to facilitate balanced learning between the two tasks. To achieve this, we employed a custom loss function that combines the losses from the two tasks and applies specific weighting factors based on task complexity, and the function is defined as follows.

$$L_{\text{custom}} = \alpha * L_{\text{sentiment}} + \beta * L_{\text{emotion}} \quad (1)$$

where $L_{Sentiment}$ is the categorical cross-entropy loss computed for the sentiment classification task, and $L_{emotion}$ is the categorical cross-entropy for the emotion classification task. For this work, the weights are set to 0.2 and 0.8, respectively, empirically based on the conjecture that emotion classification is a more complex task to learn features than sentiment analysis based on the number of labels, among other characteristics of the two tasks. During the model fine-tuning process, we use two separate dataloaders, one for each task, of equal length. We step through batches of each dataloader (as shown in Figure 2, performing forward passes for each task's batch and computing the logits. In this multitask classification, EmoBERT-tiny was trained with a batch size of 256 over 25 epochs, and further training decreases the precision-recall curve of the model, so further training was not necessary. A leaning rate of 1x10-5 and a weight decay of 1x10-6 was used as the parameters for the AdamW Optimizer.

3.2.2 Single Task Model (STM). In the single task model: In this setting, we use a pre-trained LLM model to classify sentiment and emotion individually. In this setting, we keep the pre-trained model weight intact. As a trained model, we use the BERT-base and BERT-tiny models as the feature representation model for the given text inputs. We utilize both BERT preprocessing models to preprocess text and convert it to tokens before feeding it into the BERT model.

3.2.3 Prompt-based Model (PM). Figure 3 shows prompt based model diagram. We utilized prompt model in different ways to classify sentiments and emotions. **System Prompt:** We use a system prompt containing the instructions given to the model, constraints on model outputs, and each label in the class given. **Prompt:** Contains a reiteration of the objective, followed by the text sample the model is generating the output in the evaluation of. **Shot Prompting:** For each model, we do 0-shot when the prompt contains no examples and perform one and 3-shot prompting. For each Shot, one example is provided for each label in the dataset, which is from a separate set from the subset of the dataset used for the sampling. **Evaluation:** The model output is stored, and then a function is used that parses through the output, and if the output contains one of the labels in the System Prompt, then that is used as the prediction of the model. We compare this to the true value to calculate metrics. While inferencing is being performed, the environment keeps track of the time taken for each sample. In this setting, specific prompt

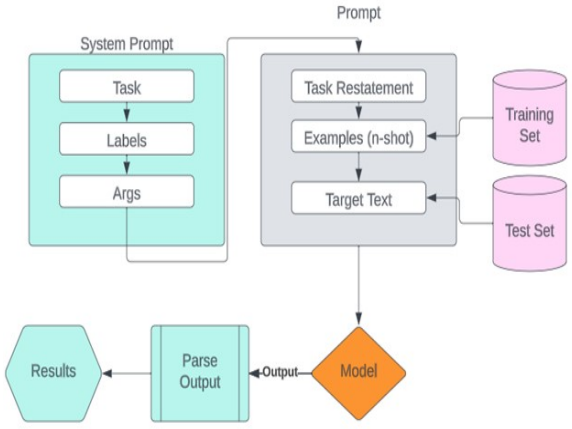

Figure 3: Comparative Model Inference Details for Prompt-Based Models

designs were used for the "mistralai/Mistral-7B-Instruct-v0.1" [15] and "meta-llama/Llama-2-7B-chat-hf" [27] models, reflecting their unique requirements and functionalities. Due to computational power limits, a representative sample of the original test set (100 samples) was used for testing. For shot testing, examples were pulled randomly for each label from the training set.

3.2.4 Shared Representation Model (SRM). This settings examines the extent to which models trained for sentiment analysis and emotion classification can adapt its learned features to effectively

perform on the other task. The test is to provide insights on the generalizability and transferability of two similar but distinct tasks. In this scenario, we employed a cross-task evaluation approach. Two separate BERT-tiny models were fine-tuned on the training sets from section 3A. Each model after being trained on either emotion classification or sentiment analysis, is then used to evaluate the test set of the opposite task. The metrics of the two models are then compared to a pre-trained BERT-tiny model, to determine if there is evidence of shared representations.

3.3 EmoBERT latency Performance Monitoring

Testing the real-time capabilities of the model requires creating an environment that can simulate real-time communication with the model. In this work, we simulate a real-time environment through text input into a GUI interface. The model performs an inference on the entire context window each time the space bar is pressed. The time it takes for the input text to be tokenized "Tokenization Time", The time it takes for the model to perform the inference post-tokenization "Inference Time" and the total time it takes to perform an Inference on the context window including Tokenization time "Processing Speed" are measured. We pool these figures throughout the runtime of the environment, in which we terminate when text exceeds the length of the context window. The metrics for each inference are then averaged out to give us a comprehensive view of the model's performance in a simulated real-time scenario.

4 EXPERIMENTAL SETUP

In this section, we will discuss the experimental setup, including the dataset description, implementation details $\text{EmoBERT}_{\text{Tiny}}$, and comparisons against other models, including an un-fine-tuned version of BERT*Tiny* and state of the art models Llama-2-7B and Mistral-7B. BERT configurations are trained and deployed in Sequence Classification while Generative models are deployed for CausalLM This section also includes the metrics and methods used for the evaluation of the models.

4.1 Dataset Description

In this work we use a collection of publicly available datasets hosted on Kaggle and Huggingface [5, 9, 11, 12, 14, 21, 24, 28].

The majority of datasets were compiled using web scraping from social media sources. After collecting the datasets, we constructed an in-depth mapping of the labels to combine semantically similar words. The emotion classification datasets and the sentiment analysis datasets are aggregated to form two separate DataLoaders of 417423 and 1176509 samples respectively. Label mappings for sentiments are categorized as {'negative': 0, 'neutral': 1, 'positive': 2}, and for emotions as {'anger': 0, 'fear': 1, 'joy': 2, 'love': 3, 'neutral': 4, 'sad': 5, 'sadness': 6, 'surprise': 7, 'worry': 8}. The class distributions are strategically curated to reflect a wide range of emotional and sentiment expressions, facilitating a robust training environment for the LLM model. Fig. 1 shows the dataset distribution for both sentiment and emotion classess.

4.2 Evaluation Metrics

Evaluation metrics are used to assess the performance of our proposed model, as well as the models we compare it to. In this work, we used a collection of standard metrics, which were measured for performance on Sentiment Analysis, Emotion Classification, and the averaged metrics between the two tasks. Accuracy, Precision, F1 score, and Matthew's Correlation Coefficient, are measured according to the standard definitions of the formula.

Table 1: Final Class Distributions

Label	Type	Map	Sample Size
Negative	Sentiment	0	320538
Neutral	Sentiment	1	498067
Positive	Sentiment	2	357904
Anger	Emotion	0	71306
Fear	Emotion	1	60399
Joy	Emotion	2	175262
Love	Emotion	3	41678
Neutral	Emotion	4	22819
Sad	Emotion	5	146814
Surprise	Emotion	6	22819
Worry	Emotion	7	8459

In addition to these standard metrics, we also evaluated each model on different metrics to test the computational efficiency.

Tokenization Time: the duration required to convert raw text into a structured format suitable for processing.

Memory Usage: the amount of system resources the model requires to perform inferencing.

Total Process Time: the duration required to perform a prediction on raw text including tokenization.

Inference Time: the duration required for the model to generate predictions post-tokenization.

4.3 Implementation Details

In this work, we developed $\text{EmoBERT}_{\text{Tiny}}$ using the PyTorch 2 library. To train EmoBERT we alternated through batches of the sentiment and emotion Dataloaders, we extracted the input IDs attention mask and labels from both Dataloader batches and performed a forward pass while computing the loss using the weighted loss function. The model was trained using a learning rate of 1e-5 a weight decay of 1e-6 and the AdamW Optimizer. The model was trained at a batch size of 256 over 26 epochs, Early Stopping was not triggered but Tensorboard was monitored for best convergence. We used a machine equipped with a Nvidia RTX 2080 Super with 8GB VRAM (1-2 GB VRAM dedicated to system tasks). Additional libraries used include transformers and our own library FallingPlanet-Toolkit [26]. The BERT configuration we use from this library calls "prajjwal1/bert-tiny" with support for multitask classification by creating a separate Linear layer of (128,labels) for each class with no limit.

4.4 Single Task Models

4.4.1 Implementation. The single-task models were implemented by calling the model with only one task, label pair. This gives us a single Linear layer. We train separate emotion classification and sentiment models using a learning rate of 1e-5 a weight decay of

1e-6 and the AdamW Optimizer. The model was trained at a batch size of 256 over 26 epochs. Categorical Cross entropy loss was used.

4.4.2 Single task Comparison. In this work, we sought to determine whether there were shared representations between emotion and sentiment tasks. Both single-task models are trained on either the sentiment or emotion dataset. Post-training, the model is then evaluated on the dataset of the opposite task. We use a pretrained $\text{BERT}_{\text{Tiny}}$ model as the baseline of all model performance. We use Accuracy, Precision, Recall, F1 score and MCC to evaluate the models' performance.

4.5 Comparative Analysis

4.5.1 Non-Generative Models. We tested $\text{EmoBERT}_{\text{Tiny}}$'s performance against pre-trained BERT-base and BERT-tiny models by 0-shot predicting the test set used to evaluate $\text{EmoBERT}_{\text{Tiny}}$. This determines the degree of improvement finetuning makes over baseline.

4.5.2 Generative Models. In this work, we use very simple prompting to compare our proposed model's performance against State-of-the-Art 7 billion parameter models. In this study, we use the Instruct version of Mistral AI's Mistral-7B model: "mistralai/Mistral-7B-Instruct-v0.1" as well as Meta's Llama-2-7B model: "meta-llama/Llama-2-7B-chat-hf".

System Instructions: Pre-prompting of system-prompting was done by first providing the task "Classify the sentiment of the text" or "Identify the emotion from". This informs the model of what task it should be performing. The next section of the system-prompt is the labels, this informs the system of the available labels that it should include as part of the model output. Then we provide arguments: "Always provide a label". Optional arguments can be added here to apply CoT or other techniques.

Prompt Instructions: Restate the task, and then provide the text, if there are examples append them defined as "example (n):". An example is provided for each label for shot prompting.

Pipeline Hyperparameters: float32 max_length 256 (gets overwritten by max_new_tokens).

Do_sample = False
Top_p = None
Repetition_penalty = 1.1
max_new_tokens = 7

Notes: Mistral does not respond well to 'max_new_token' restrictions and will attempt to output sequences longer than the 'max_new_token' length leading to the truncation of some outputs.

4.5.3 Evaluation of Generative Models. In this work, due to the computational limitations of our system, the generative models could only be deployed using a 4.6 GHz CPU and 32 GB of RAM, therefore, we only evaluate the model on a 200 sample subset of the test set used to evaluate $\text{EmoBERT}_{\text{Tiny}}$ The models are evaluated for their accuracy, precision, recall, f1 score, and Matthews correlation coefficient, in addition, the processing time of each sample was measured using Python's 'time' library. A parser was used to parse the model output for the class labels stated in the prompt. These extracted labels are compared to the true labels using String comparison. We use the boolean result of these String comparisons to compute the metrics. Each model is evaluated 3 times on both tasks using shot prompting. We use 0-shot 1-shot and 3-shot in this work. We define a shot as a complete class,example pair for all classes in the task and append them to the prompt for in-context learning.

4.6 Latency Testing

In this section we will discuss the deployment of $\text{EmoBERT}_{\text{Tiny}}$ in a simulated real-time environment, to test the performance capabilities of the model.

Latency testing on the model was performed using Python. The $\text{EmoBERT}_{\text{Tiny}}$ model was implemented using the PyTorch framework. Additional libraries included psutil for system monitoring and tkinter for the graphical user interface. The model is loaded using the state dictionary and set to eval mode.

4.6.1 Data Collection and Analysis. Each inference triggers a separate thread that calculates and logs the tokenization time, inference time, and total process time. Another separate thread continuously monitors and updates the memory usage. Upon closing of the application, average times for tokenization, inference, and total process are printed to the console for analysis.

Interactive GUI Setup: To test the $\text{EmoBERT}_{\text{Tiny}}$ model's performance in real-time, we developed an interactive graphical user interface (GUI). This GUI, built using Python and the Tkinter library, allowed users to input text and receive immediate sentiment and emotion classification results. The GUI was designed to be user-friendly, enabling direct interaction with the model for latency assessment. For inferencing upon recieving text input, the GUI facilitated the inference process by tokenizing the text and feeding it to the $\text{EmoBERT}_{\text{Tiny}}$ model. We utilized our custom tokenization function, tokenize_unlabeled_text, which aligns with the model's input requirements. The model then performed sentiment and emotion classification on the input text. For monitoring the performance of the model The GUI provided live feedback on the model's performance. it displays the inference time and memory usage for each input, offering insights into the model's efficiency and resource demands. This provides a quality evaluation of whether the model is suitable for potential real-time applications, and the hardware requirements to inference the model.

To maintain fairness in assessment, each test was performed from a cold boot with no extraneous background tasks running.

5 RESULTS

In this section we will discuss the outcomes of our performed experiments. First, we evaluate the performance of $\text{EmoBERT}_{\text{Tiny}}$ on emotion and sentiment classification as well at its performance in the real-time simulation. Then, we discuss the results of comparative testing against pretrained models. We follow this up with an evaluation of Cross-Task Testing, not only to compare single-task trained models with our multi-task model, but to discover if there are shared representations between sentiment and emotion classification.

5.1 $\text{EmoBERT}_{\text{Tiny}}$ Results

5.1.1 Sentiment and Emotion Classification. We conducted experiments with our finetuned model, which was trained through alternating batches of Sentiment and Emotion DataLoaders and using

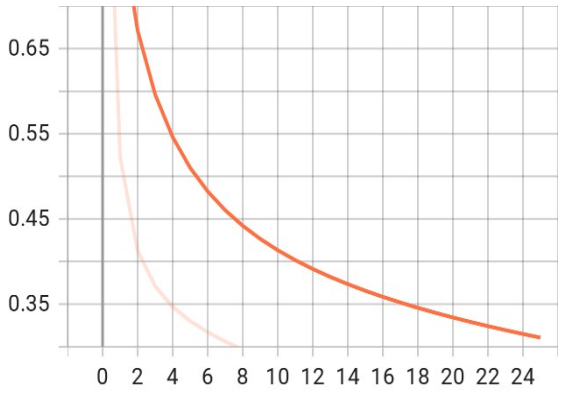

Figure 4: EmoBERT$_{\text{Tiny}}$ Training Loss over 26 Epochs

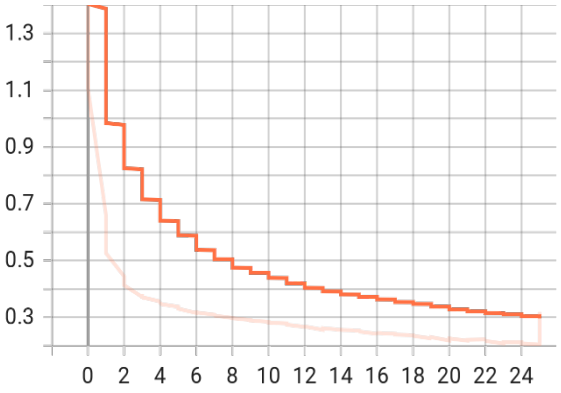

Figure 5: EmoBERT$_{\text{Tiny}}$ Validation Loss over 26 Epochs

a weighted loss function. EmoBERT$_{\text{Tiny}}$ demonstrates exceptional performance on both sentiment and emotion classification tasks as shown in Figure 2, With sentiment accuracy reaching 93.14% and emotion accuracy at 85.46% the model performs well on capturing emotional nuance. (MCC) values are significantly positive, indicating strong predictive quality, with 0.8949 in sentiment analysis and 0.815 in emotion classification.

5.1.2 Low Latency Testing Results. The real time latency test was simulated using a GUI environment and user input, metrics were calculated on a separate thread from inferencing to maintain both consistency and measurement integrity. The EmoBERT$_{\text{Tiny}}$ model exhibited an average tokenization time of 146.20ms and an average inference time of 8.04ms per sample. The combined total processing time for un-tokenized text input by a user in a GUI environment is 154.23ms. These results surpass the predicted upper limit of speed required for human interaction based on both typed word [20] and spoken word [10] For average conversational speed the model performs 432% and 259% faster than the estimated required inference speed for typed and spoken word respectively [10, 20]. The simulated real-time environment results make it applicable to many different use-cases reliant on real-time performance capabilities, both sufficient for direct inferencing on user facing programs, or informing other larger processes of the emotion of the text.

5.2 Comparative Single-Task Results on Pre-trained Models

In this section we will discuss the Single-task results of the pre-trained BERT$_a$s well as Bert-Base Cased to measure the finetuned EmoBERT$_{\text{Tiny}}$'s gain in performance from finetuning on Sequence Classification.

5.2.1 Prajjwal1 BERT$_{Tiny}$ -PreTrained. As shown in Table 3 This baseline model exhibits substantially lower performance, with sentiment accuracy at 38.09% and emotion accuracy at 14.70%. The low precision, recall and f1 scores across both tasks indicate a limited ability to distinguish between different sentiments and emotions effectively. The near-zero MCC in emotion classification suggests that the predictive power of the model is no better than random guessing.

5.2.2 Bert-Base Cased. Similarly, Bert-Base Cased underperforms in both tasks, with sentiment accuracy at 30.41% and emotion accuracy at 8.17% The negligible MCC scores reinforce the model's limited predictive capability in this context.

5.3 Comparative Results on SOTA Generative Models

In this section, we will discuss the performance of Mistral-7B and Llama-2-7B. We will see both the metrics and inference speed of the model during testing on the same hardware and a balanced subset of the dataset.

(Llama-2-7B, Mistral-7B-Instruct) Generative Models underperformed compared to the fine-tuned BERT models, a complete evaluation can be found in Table 4

5.3.1 Llama-2-7B. at the base float32, both models are very slow to inference on our hardware. Llama-2-7B's performance varies across different shot scenarios (0-shot, 1-shot, and 3-shot). The model shows moderate accuracy in sentiment classification but struggles significantly in emotion classification, especially in the 0-shot scenario with an accuracy of only 3%. The average inference speed of 12 seconds per sample indicates a considerable delay, which could be a limiting factor for real-time applications. Shot learning did not improve the generalization capabilities of the model, and the prompting methods resulted in simplified reasoning capabilities at the cost of increased inference speed.

5.3.2 Mistral-7B-Instruct. Mistral-7B-Instruct also displays varying performance with different shot scenarios. In sentiment classification, its accuracy peaks at 43%, while in emotion classification, it reaches a maximum of 27% in the 1-shot scenario. The high average inference speed of 800 seconds per sample suggests significant latency, making it un-suitable for applications requiring quick feedback using the setup in this work. Inference speed was un-altered by prompt engineering.

5.4 Label Comparison Test (Cross-Task Testing)

In this section, we will discuss the test results of finetuning BERT$_{\text{Models}}$ on the single Dataloader for emotion or sentiment classification. The test results for models trained exclusively on one task show interesting patterns illustrated in Table 5. The BERT$_{\text{Tiny}}$ model trained only on emotion achieves an accuracy of 85.34%, while

Table 2: EmoBERT$_{Tiny}$ Results

Metric	Value
Total Sentiment Accuracy	0.931
Total Emotion Accuracy	0.855
Total Sentiment Precision	0.931
Total Emotion Precision	0.855
Total Sentiment Recall	0.931
Total Emotion Recall	0.855
Total Sentiment F1	0.931
Total Emotion F1	0.855
Total Sentiment MCC	0.895
Total Emotion MCC	0.815

the sentiment-only model reaches an impressive 95.09% accuracy. When evaluated against the results of the multi-task finetuned model, we can determine that performance can be preserved to a considerable measure even when training on another additional task. When the sentiment-trained model is used for emotion classification, accuracy plummets to 2.06%, and similarly, the emotion-trained model yields a sentiment accuracy of 25.60%. This significant drop in performance indicates the specialized nature of the training and the challenges in cross-task applicability without specific training. Evaluating that post-finetuning the cross-task accuracy is lower than it was for the pretrained model, meaning that it is likely that the training had overwrote some of the weights that were predicting that task.

5.5 Summary of Key Findings

EmoBERT$_{Tiny}$ achieves an accuracy of 93.1% on sentiment and 85.5% accuracy on our test set. Our model performs at least twice as well as Bert-Base and BERT$_{Tiny}$ pretrained models on sentiment and over 5 times better than those models on emotion classification. These results highlight the effectiveness of fine-tuning strategies on compact models like BERT$_{Tiny}$. Furthermore, the model achieves low latency in inference times, with an average tokenization time of 146.20ms and inference time of 8.04ms, showcasing its suitability for real-time applications. The model is fast enough for usage in both systems that receive text input and systems that receive voice input. The model can be run locally at the speed tested in this work as long as 2GB of ram is allocated to the environment the model is running in.

6 CONCLUSION

In this work we presented the EmoBERT$_{Tiny}$ model, a fine-tuned BERT$_{Tiny}$ model for sentiment analysis and emotion classification in text. Using multi-task learning we are able to fine-tune the model to accurately predict the classes of text examples of both tasks.

The EmoBERT$_{Tiny}$ model demonstrates notable accuracy in sentiment (93.14%) and emotion (85.46%) classification, substantially outperforming baseline models such as Bert-Base Cased and Prakwal1's pre-trained BERT$_{Tiny}$. The model also greatly outperforms Llama-2-7B and Mistral-7B-Instruct in both accuracy and performance when prompting and parameters are optimized for low latency.

The EmoBERT$_{Tiny}$ model achieves top of the line performance in input processing speed, both sufficient for nearly all real-time requirements according to our calculations.

In this work we also studied whether emotion and sentiment had shared representations between the two tasks, in which we discovered that at least for sequence classification models, emotion and sentiment do not have shared representations and training on one will not increase the performance of models on the other task.

We are excited to expand on this work, fast and accurate calculation of human emotions are particularly valuable for interactive AI systems in customer service, mental health monitoring, and social media analytics. These systems require nuanced understanding of human emotions to enhance user engagement, offer personalized support, and generate insights into human perception of model outputs.

Model outputs could be used to with nearly no additional overhead for BERT models inform generative models of the sentiment and emotion of the text received by the model while being able to inform models of different architectures in less than half of a second..

In addition to the areas already highlighted, there exists significant potential for integrating EmoBERT$_{Tiny}$' s outputs with models analyzing other modalities, such as facial expressions and voice tone, for a more comprehensive emotion analysis system. This multimodal approach can provide a more holistic understanding of human emotions, especially in scenarios where text alone may not fully capture the emotional context.

However, it is crucial to ensure that labels across these different modalities are harmonized. Aligning the classification labels and criteria across different modalities will be the key to developing a seamless and effective multimodal emotion recognition system. This approach could further enhance application in virtual assistants, empathetic robotics, and other emotion-aware systems. As companies replace customer service teams with chatbots, it could be beneficial to use EmoBERT$_{Tiny}$ to provide a very quick prediction of the emotion of the client, and then append some information based on the state to the system prompt of the model. This allows the chatbot to better interact with the client while providing almost no additional overhead.

6.1 Limitations and Threats to Validity

6.1.1 Label Imbalance. Joy and Sadness are over-represented, while Neutral and Worry are heavily underrepresented in comparison. Neutral being underrepresented is particularly worrisome because for humans it would be the most common emotion for people to self-identify.

6.1.2 Model Evaluation. Class-wise testing was not conducted as part of our evaluations of model performance. This could potentially be important information that informs us in more detail about which specific labels the model performed well in, and which labels we could improve model performance by collecting more data.

6.1.3 Dual Task Training: The method in which we used to train the model, makes the length of the data the number of batches in the smaller of the DataLoaders, as a result the larger dataloader

Table 3: Single-task Model Comparisons

Model Name	Parameters	Sen Acc	Emo Acc	Sen F1	Emo F1	Avg Inference Speed (s/sample)
$\text{EmoBERT}_{\text{Tiny}}$	4.4M	0.9314	0.8546	0.9314	0.8546	8.04ms
$\text{BERT}_{\text{Tiny}}$ -Pretrained	4.4M	0.3809	0.1470	0.3809	0.1470	N/A
Bert-Base Cased	24M	0.3041	0.0817	0.3041	0.0817	N/A

Table 4: Generative Model Results

Model Name	Parameters	Test Type	Sen Acc	Emo Acc	Sen F1	Emo F1	Avg Inference Speed (s/sample)
Llama-2-7B	7B	0-Shot	0.43	0.03	0.2004	0.0072	12.15s
Llama-2-7B	7B	1-Shot	0.43	0.02	0.2004	0.0049	21s
Llama-2-7B	7B	3-Shot	0.43	0.11	0.2004	0.0247	25s
Mistral-7B-Instruct	7B	0-shot	0.30	0.13	0.1538	0.0287	200s
Mistral-7B-Instruct	7B	1-shot	0.30	0.27	0.154	0.05	300s
Mistral-7B-Instruct	7B	3-shot	0.43	0.08	0.2	0.002	300s

Table 5: Cross-Task Testing Results

Test Type	Accuracy	Precision	Recall	F1-Score	MCC
$\text{BERT}_{\text{Tiny}}$ (Emotion Only)	0.8534419	0.8534419	0.8534419	0.8534419	0.8137156
$\text{BERT}_{\text{Tiny}}$ (Sentiment Only)	0.9509892	0.9509892	0.9509892	0.9509892	0.9249869
Emotion from Sentiment Model	0.0205631	0.0205631	0.0205631	0.0205631	-0.0123171
Sentiment from Emotion Model	0.2560721	0.2560721	0.2560721	0.2560721	0.0034028

loses a small bit of information, if more samples are gained this gap would increase.

In models that use argmax classification, where the output is the category with the highest probability, the representation of 'neutral' is tricky. Argmax inherently selects the most prominent category, but in cases where emotions are subtle or balanced ('ie., closer to a 'neutral state), this method might not accurately reflect the true emotional state. This is because argmax will still pick the most prominent of the low-intensity emotions, potentially overlooking the overall balance or neutrality of the emotional expression.

Therefore, it may be appropriate to remove all 'neutral' samples from the dataset, or it may become necessary to gather more 'neutral' samples in order to prevent bias in generalization of the model and to improve accuracy when applied to real-life settings.

6.1.4 Preprocessing. Data Cleaning and Label Standardization During the preprocessing phase, we undertook a systematic approach to standardize emotion and sentiment labels. This involved consolidating labels that were either synonymous or exhibited similar emotional polarity. For instance, labels representing similar emotional states of varying intensities of the same emotion were grouped under a unified category. This process was guided by linguistic principles and existing psychological frameworks to ensure consistency and relevance to the emotional spectrum being analyzed.

> The limitation of this process is while the standardization of labels, while being necessary for analytical consistency, introduces a potential limitation. It inherently assumes a degree of uniformity in emotional expression and perception that may not capture nuanced differences in how individuals experience and express similar emotions. For example, what is categorized as 'sad' in one instance might be experienced with different intensities or shaded by different individuals, which the model may not distinctly recognize due to the amalgamation of similar emotional states. This limitation highlights the challenge in emotion classification tasks of balancing the need for manageable and coherent label categories with the inherent diversity and complexity of human emotional experiences.

6.1.5 Limited Subset for Comparative Analysis. With the computational resources defined in this study, inferencing on Llama-2-7B and Mistral-7B cannot be performed on the GPU, on the CPU to inference all sentiment samples is 103 days. Using a smaller subset with the same label distribution gives us correct representations, however it is possible that the best samples are not the ones that are selected.

6.1.6 Prompting Limitations in Generative Models. Computational limitations prevent the use of more complex prompting techniques, like Chain of Thought (CoT), in which would need to have the 'max_tokens' by the output increased to allow for the model to exhibit better reasoning techniques. More advanced prompt techniques would increase the inference time for each sample.

6.1.7 Generative Model Hyperparameter Tuning. It is likely that performance was left on the table by utilizing the default hyperparameters for Mistral and Llama models, it could also be considered that quantization of the model would have likely led to less latency than we experienced in our testing.

6.1.8 Harmony with other methods: For one of our proposed use-cases of $\text{EmoBERT}_{\text{Tiny}}$ we suggested using it to inform other models or methods of the emotional state of the speaker/typer. The threat to this approach is for automated systems, the model that we try to inform would have to be built around receiving this input and managing the variance in response based on the emotional state received. With this approach, to researchers and designers, it could be more beneficial to build the entire system from the ground up.

REFERENCES

[1] Sumya Akter, Rumman Ahmed Prodhan, Tanmoy Sarkar Pias, David Eisenberg, and Jorge Fresneda Fernandez. 2022. M1M2: Deep-learning-based Real-time Emotion Recognition from Neural Activity. *Sensors* 22, 21 (2022), 8467.

[2] Nazish Azam, Tauqir Ahmad, and Nazeef Ul Haq. 2021. Automatic Emotion Recognition in Healthcare Data using Supervised Machine Learning. *PeerJ Computer Science* 7 (2021), e751.

[3] Santosh Kumar Bharti, S Varadhaganapathy, Rajeev Kumar Gupta, Prashant Kumar Shukla, Mohamed Bouye, Simon Karanja Hingaa, and Amena Mahmoud. 2022. Text-Based Emotion Recognition Using Deep Learning Approach. *Computational Intelligence and Neuroscience* 2022 (2022).

[4] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language Models are Few-shot Learners. *Advances in neural information processing systems* 33 (2020), 1877–1901.

[5] Dair-ai. [n. d.]. Emotion. https://huggingface.co/datasets/dair-ai/emotion/tree/main/data

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv preprint arXiv:1810.04805* (2018).

[7] Zhongyu Fang, Aoyun He, Qihui Yu, Baopeng Gao, Weiping Ding, Tong Zhang, and Lei Ma. 2022. FAF: A Novel Multimodal Emotion Recognition approach Integrating Face, Body and Text. *arXiv preprint arXiv:2211.15425* (2022).

[8] Nicolas Fragopanagos, John G. Taylor, Roddie Cowie, Winfried Fellenz, Stefaos Kollias, George Votsis, and Ellen Douglas-Cowie. 2005. Emotion Recognition in Human–Computer Interaction. *Neural Networks* 18, 4 (2005), 389–405. https://doi.org/10.1016/j.neunet.2005.03.006 Emotion and Brain.

[9] Praveen Govi. [n. d.]. Emotion Dataset for NLP. https://www.kaggle.com/datasets/praveengovi/emotions-dataset-for-nlp

[10] Roger Griffiths. 1990. Facilitating Listening Comprehension Through Rate-control. *RELC Journal* 21, 1 (1990), 55–65.

[11] Pashupati Gupta. [n. d.]. Emotion Detection From Text. https://www.kaggle.com/datasets/pashupatigupta/emotion-detection-from-text

[12] M Yasser H. [n. d.]. Twitter Tweets Sentiment Analysis Dataset. https://www.kaggle.com/datasets/yasserh/twitter-tweets-sentiment-dataset

[13] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-term Memory. *Neural computation* 9, 8 (1997), 1735–1780.

[14] Ishant. [n. d.]. Emotions in Text. https://www.kaggle.com/datasets/ishantjuyal/emotions-in-text

[15] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7B. *arXiv preprint arXiv:2310.06825* (2023).

[16] Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2019. Tinybert: Distilling Bert for Natural Language Understanding. *arXiv preprint arXiv:1909.10351* (2019).

[17] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet Classification with Deep Convolutional Neural Networks. *Advances in neural information processing systems* 25 (2012).

[18] Ting-Mei Li, Han-Chieh Chao, and Jianming Zhang. 2019. Emotion Classification based on Brain Wave: a Survey. *Human-centric Computing and Information Sciences* 9, 1 (2019), 1–17.

[19] Zheng Lian, Haiyang Sun, Licai Sun, Kang Chen, Mngyu Xu, Kexin Wang, Ke Xu, Yu He, Ying Li, Jinming Zhao, et al. 2023. Mer 2023: Multi-label Learning, Modality Robustness, and Semi-supervised Learning. In *Proceedings of the 31st ACM International Conference on Multimedia*. 9610–9614.

[20] Academy of Learning Career College. [n. d.]. The Fastest Typists In the World-Past and Present. https://www.academyoflearning.com/blog/the-fastest-typists-in-the-world-past-and-present/

[21] Passionate-NLP. [n. d.]. Twitter Sentiment Analysis. https://www.kaggle.com/datasets/jp797498e/twitter-entity-sentiment-analysis?select=twitter_training.csv

[22] Svetlana Pinet, Christelle Zielinski, F.-Xavier Alario, and Marieke Longcamp. 2022. Typing Expertise in a Large Student Population. *National Library of Medicine* (2022). https://www.ncbi.nlm.nih.gov/pmc/articles/PMC9356123/

[23] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. 1986. Learning Representations by Back-propagating Errors. *nature* 323, 6088 (1986), 533–536.

[24] Saurbabh Shahane. [n. d.]. Twitter Sentiment Dataset. https://www.kaggle.com/datasets/saurabhshahane/twitter-sentiment-dataset

[25] Matteo Spezialetti, Giuseppe Placidi, and Silvia Rossi. 2020. Emotion Recognition for Human-Robot Interaction: Recent Advances and Future Perspectives. *Frontiers in Robotics and AI* (2020), 145.

[26] William Stigall. [n. d.]. Falling Planet Repository. https://github.com/FallingPlanet/FallingPlanet-Toolkit/

[27] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open Foundation and Fine-tuned Chat Models. *arXiv preprint arXiv:2307.09288* (2023).

[28] Achintya Tripathi. [n. d.]. Emotion Classification NLP. https://www.kaggle.com/datasets/anjaneyatripathi/emotion-classification-nlp

[29] Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Well-read Students Learn Better: On the Importance of Pre-training Compact Models. *arXiv preprint arXiv:1908.08962* (2019).

[30] Farzan Vahedifard, Atieh Sadeghniiat Haghighi, Tirth Dave, Mohammad Tolouei, and Fateme Hoshyar Zare. 2023. Practical Use of ChatGPT in Psychiatry for Treatment Plan and Psychoeducation. *arXiv preprint arXiv:2311.09131* (2023).

[31] Vladimir Vapnik. 1995. Support-Vector Networks. *Machine learning* 20 (1995), 273–297.

[32] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is All You Need. *Advances in neural information processing systems* 30 (2017).