EEG and Eye Tracking Feature Contextualization based Affective Computing through LightGBM enabled Stacked Ensembling of LLMs

by

Maheer Helal
21101154
Siam Rahman Karip
21101155
Naveya Novely Datta
21101321

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science

Department of Computer Science and Engineering
Brac University
February 2025

Declaration

It is hereby declared that

1. The thesis submitted is our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

Maheer Helal

21101154

Siam Rahman Karip

21101155

Naveya Novely Datta

21101321

Approval

The thesis titled **"EEG and Eye Tracking Feature Contextualization based Affective Computing through LightGBM enabled Stacked Ensembling of LLMs"** submitted by

1. Maheer Helal (21101154)
2. Siam Rahman Karip (21101155)
3. Naveya Novely Datta (21101321)

of Fall, 2024 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on January 31, 2025.

Examining Committee:

Supervisor:

Rafeed Rahman

Senior Lecturer
Department of Computer Science and Engineering
BRAC University

Co-Supervisor:
(Member)

Md. Golam Rabiul Alam

Professor
Department of Computer Science and Engineering
BRAC University

Head of Department:
(Chair)

Dr. Sadia Hamid Kazi

Chairperson and Associate Professor
Department of Computer Science and Engineering
BRAC University

Abstract

Affective Computing plays a crucial role in analyzing and interpreting human emotional states. Electroencephalography (EEG) signals is widely regarded as a reliable modality due to their direct measurement of brain activity. However, traditional deep learning models used in EEG-based affective computing face vital challenges, such as high inter-subject variability, reliance on extensive feature engineering, and difficulties in integrating multimodal information such as eye-tracking data. To address these limitations, a novel preprocessing pipeline was developed in this study which encodes high-dimensional EEG features along with the eye-tracking features into structured contextualized sequences, making them compatible with Large Language Model (LLM) architectures. Additionally, an ensemble learning framework was incorporated to effectively integrate these heterogeneous modalities. This research explores the potential of LLMs in affective computing which have demonstrated their ability to process structured text representations utilizing their pretraining on diverse and extensive corpora. This adaptability was harnessed by transforming EEG and eye-tracking data into structured and contextualized string representations. The experimental results demonstrate the effectiveness of our approach, in which FLAN-T5, GPT-2, and BERT achieved classification accuracies of 86%, 92%, and 90%, respectively on the EEG dataset. Furthermore, the integration of EEG and eye-tracking data using a Stacked Ensemble method with LightGBM as the meta-model led to a significant improvement, achieving a final classification accuracy of 96% which indicates the benefits of combining different types of data for affective computing.

Keywords: Affective Computing; Emotion Recognition; Electroencephalography; Large Language Models; Ensemble Learning;

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Human affections are intrinsic psychological behaviors that shape how individuals perceive and respond to the surrounding world. These behaviors form the basis of human interactions and the overall well-being. Affection can be expressed in many ways, such as through facial expression, eye movements, gestures, tone of voice, etc, providing observable cues to an individual's emotional state, intentions or level of engagement. By using different types of technologies, devices and sensors like Electroencephalogram (EEG) sensors, eye-tracking devices, microphones, and cameras we can capture these affections or emotions. With the advancements in Brain Computer interface (BIC), many researchers have a growing interest in affection analysis techniques [35]. Affective computing can bridge the gap between human emotional understanding and machine processing by logical processing of complex emotions through signals and data. Among other modalities such as speech or gesture, EEG-based affecting computing has been a preferred choice in this field of research because of its capabilities to directly capture the brain's electrical activity as well as provide an objective reflection of the emotional state, which has proven to be more accurate over other methods [35]. For this reason, EEG is considered to be the most suitable modality for emotion analysis applications in various fields including healthcare, entertainment, and e-learning.

In EEG-based emotion analysis tasks, traditional machine learning techniques heavily rely on intensive feature extraction methods that require human expertise [14]. In contrast, Deep Learning (DL) models, such as Convolutional Neural Networks (CNNs), Long Short-Term Memory networks (LSTMs), and hybrid architectures, have demonstrated superior performance by learning hierarchical representations directly from raw EEG data [30]. For instance, in the study from paper [25] we find a combined convolutional neural network (CNN) with a bidirectional long short-term memory network (Bi-LSTM) with an output accuracy of 99.55% with 3 classification tasks (positive, neutral, negative) and 99.79% with four classification tasks (happiness, sadness, fear, neutral). Despite their success, these DL models depend on carefully crafted preprocessing pipelines to ensure optimal performance and avoid overfitting [32].

Building on these traditional methods, researchers are now exploring advanced approaches that integrate Large Language Models (LLMs) to further enhance Affection computing systems. Although LLMs were initially built to process text data, their architecture can be modified and extended to handle non-textual numerical data such as EEG signals. Their ability to understand complex patterns and process vast amounts of information enables them to achieve highly accurate results. LLMs also have the potential for strong generalization capabilities due to their massive scale and ability to learn generalizable representations. Recent studies, such as [38], discuss the incorporation of traditional signal processing techniques within the architecture of LLMs. LLMs, particularly those based on Transformer architectures like GPT, have the ability to process complex data in the form of trainable time-frequency representations [38]. These representations are dynamically optimized for specific tasks, allowing LLMs to handle diverse data types, including physiological signals like EEG.

However, relying solely on a single modality, such as EEG, is often insufficient for capturing the complexity for affection analysis. Different modalities like eye tracking and facial expressions, provide complementary information that can enhance the accuracy and robustness of the systems. For example, the researchers of paper [12] got improved classification results by using a multimodal fusion method which enabled them to capture nonlinear emotional correlation between the EEG signals and blood volume pulse (BVP). However, processing and merging data from such multimodalities present significant challenges due to the diverse structures and characteristics of the data. Transformer-based LLMs can address these challenges as they are capable of processing structured data. With appropriate adaptations, they can effectively handle multimodal inputs by utilizing their attention mechanisms and contextual learning capabilities [42].

Although LLMs have the potential to process EEG and eye-tracking data separately, the addition of ensemble learning methods can integrate multiple models and enhance the accuracy and robustness of affection analysis models. The stacked ensemble method is capable of combining the predictions of multiple models allowing more robust predictions and reducing the shortcomings of a single model [22]. Likewise, applying this approach to a multimodal system can be more effective for affection computing as these approaches aggregate predictions from multiple sources which help to reduce the risk of overfitting and ensures better performance.

1.1 Research Problem

The field of affective computing has made significant progress in emotion recognition using different physiological signals. However, despite rapid advancements in deep learning-based approaches, there are several key limitations which hinder the generalizability, efficiency, and scalability of current models. LLMs have primarily been developed for natural language processing (NLP) tasks, demonstrating exceptional contextual learning capabilities. However, their potential for handling structured physiological signals, specially in the field of affection computation remains largely unexplored. Current research focuses heavily on traditional deep learning models (CNNs, RNNs, Bi-LSTMs) for EEG-based affection analysis, but these methods are constrained by predefined feature sets and require custom architectures to process multimodal data. This creates a significant gap, as LLMs may offer a novel alternative by leveraging their transformer-based attention mechanisms to process complex sequential and multimodal data more effectively.

EEG-based emotion recognition models face a well-documented issue of high inter-subject variability. Individual differences in brain activity result in models overfitting to specific subjects or datasets, leading to poor generalization on unseen participants. Existing deep learning methods often require transfer learning or domain adaptation techniques to address these variations, yet they still struggle to maintain robust cross-subject performance. This presents a major challenge, as a practical affective computing system should generalize well across diverse populations without requiring excessive retraining.

Traditional machine learning (ML) approaches to EEG-based emotion recognition rely heavily on feature extraction and transformation techniques to convert raw EEG signals into structured representations. Commonly used features include power spectral density (PSD), differential entropy (DE), functional connectivity measures, and wavelet-based features, all of which require domain expertise and manual tuning to optimize model performance. Large Language Models (LLMs) have demonstrated their ability to learn complex feature representations in natural language processing tasks without the need for explicit feature extraction [33]. This raises the question of whether LLMs can be adapted to autonomously learn feature representations from EEG signals, reducing or even eliminating the reliance on handcrafted features in affective computing.

Affection computation is inherently multimodal, as different physiological signals provide complementary information. However, the fusion of these heterogeneous data sources presents a major technical challenge due to differences in data structure, statistical properties, and temporal resolution. These variations create difficulties in aligning, synchronizing, and effectively integrating multimodal inputs, often requiring manual feature engineering or complex fusion strategies. Existing deep learning models struggle to handle such diverse modalities efficiently, limiting their ability to fully capture the nonlinear relationships between different physiological signals and affecting the overall robustness of affection computation like emotion recognition systems. Current multimodal deep learning approaches employ early fusion, late fusion, and hybrid models, but these methods require custom architectures and extensive hyperparameter tuning. Given that transformer-based LLMs are inherently

designed to process structured data, their potential for seamless multimodal fusion remains an open research question.

Ensemble learning has been widely used to improve model performance by combining multiple classifiers, thereby reducing individual model weaknesses. However, most ensemble methods in deep learning-based emotion recognition:

- Rely on homogeneous datasets, typically using a single modality.
- Combine models with similar architectures, rather than exploring heterogeneous model ensembles.

There is currently no significant research on how LLMs can be incorporated into ensemble learning frameworks for affective computing. Understanding whether stacked ensemble methods can enhance the robustness of LLM-based multimodal emotion recognition systems remains an unexplored area.

1.2 Research Objective

The primary objective of this research is to dive into the application of Large Language Models (LLMs) in the field of affective computing, integrating LLMs with numeric physiological data aiming to improve generalization, reduce feature engineering and enhance multimodal fusion through an ensemble learning framework.

- This study investigates the potential of Large Language Models (LLMs) in affective computing, specifically in EEG-based emotion recognition. By adapting LLMs to process structured physiological data, this research aims to determine their effectiveness in capturing contextual dependencies and reducing reliance on handcrafted feature engineering.
- This study introduces a Generalizable Data Preprocessing Pipeline that transforms high-dimensional numerical data, such as, EEG, into a contextualized representation, making it compatible with LLM architectures. The pipeline is designed to be scalable and adaptable to various physiological signal types, enabling broader applications beyond affective computing.
- This study also implements an Ensemble Learning Framework for Multimodal Fusion to enhance the reliability and the robustness of the system by utilizing the strengths of different LLMs and reducing the errors. An ablation study was conducted to assess the contribution of individual models (FLAN-T5, BERT, GPT-2) within the stacked ensemble, highlighting how model fusion improves overall classification accuracy and enhances generalization.

Chapter 2

Literature Review

This comprehensive literature review explores the latest advancements in affection computing, specially in the field of emotion recognition, with a particular focus on multimodal approaches and the integration of large language models (LLMs). It begins by examining studies on emotion recognition using EEG signals highlighting the role of EEG in capturing the brain's electrical activity for accurate emotion recognition and discusses the utilization of different types of deep learning methods as well as feature extraction for this task. Subsequently, the review then moves to the part of research related to multimodal data which combines EEG with complementary models, such as, eye tracking, for the enhancement of affection analysis performance. Moreover, the review explores Large Language Models in analyzing their capability to handle complex numerical data and applications like in multimodal sentiment analysis. A discussion of works related to the use of ensemble methods, especially stacked Ensembling as a means of enhancing the reliability of the multimodal system has been done. In a word, the studies including the insights, potential and limitations in their exploration of the field of affection analysis are summarized in the review.

The researchers in [10] proposed a method that uses convolutional neural networks (CNNs) to classify emotions based on differential entropy (DE) features extracted from EEG signals. By generating topology-preserving EEG images through interpolation methods, The research shows the preservation of spatial relationships between electrodes. The proposed method achieved state-of-the-art performance for EEG-based emotion recognition, with improved accuracy and robustness by preserving EEG signal topology. The approach requires complex preprocessing steps, such as artifact removal and data smoothing, which may limit scalability and lack publicly available large-scale EEG datasets for broader validation. The researchers in [15] developed a deep learning framework using a convolutional neural network (CNN) with ResNet50 architecture to classify emotions (positive, neutral, and negative) based on EEG signals. The study demonstrated the effectiveness of CNN with ResNet50 for EEG-based emotion recognition, achieving superior performance compared to previous approaches. The drawback of the study was that the method relies solely on the SEED dataset, limiting its generalizability to other datasets, and might face potential challenges in real-world applications due to subject-specific variability in EEG signals. Paper [17] proposed a model, VAE-D2GAN, that combines Various Autoencoders (VAE) and Generative Generative Adversarial Networks (GAN). The goal of this research was to tackle the issue of data scarcity in EEG-based emo-

tion recognition by producing diverse high-quality data. The model was able to enhance the classification of the seed-IV dataset. One limitation of the research was that the diversity of the generated sample could cause latent space complexity and increase computational cost. The study of paper [21] proposed the ATDD-LSTM (Attention-based LSTM with Domain Discriminator) model for EEG-based emotion recognition. This model introduces a domain discriminator to address distribution shifts between training and test datasets and employs an attention mechanism to focus on emotion-related EEG channels dynamically. It also leverages LSTM to capture nonlinear relationships among EEG channels. However, a significant drawback of the study is its reliance on pre-defined EEG channel sequences, which may not generalize to entirely new configurations or other datasets without similar preprocessing. Furthermore, while the domain discriminator reduces distribution shifts, its performance heavily depends on sufficient labeled data for robust training. The authors of [23], discussed a Regularized Graph Neural Network (RGNN) model for EEG-based emotion recognition. The model's graph neural networks achieved to capture local and global inter-channel relations inspired by the brain's topology. They developed two regularizers: Node-wise Domain Adversarial Training (NodeDAT) to enhance subject-independent classification and Emotion-Aware Distribution Learning (EmotionDL) to address noisy labels. The model showed improved accuracy but involved significant computational effort, which makes it less practical or feasible for real-time applications where quick processing is required. On the other hand, the authors of [13] introduced a dynamic graph convolutional neural network (DGCNN) that learns the adjacency matrix dynamically to capture the relationships between EEG channels. Although the model was able to report accuracies of 90.4% (subject-dependent) and 79.95% (subject-independent) on SEED, the High computational complexity and difficulty in ensuring robust generalization across subjects. On the other hand, transfer learning frameworks like the one proposed by [11] showcase a multisource transfer learning framework to address the challenge of large individual differences in EEG data, which hinder cross-subject emotion recognition. Their method involves selecting appropriate source subjects, performing style transfer mapping (STM) to align the target subject's data with the sources, and combining classifiers from the sources using a weighted ensemble approach. The model in this study achieved a 12.72% improvement in accuracy by leveraging multi-source transfer learning, however, a limitation of the research was its focus on scenarios with a small number of emotion categories (three), which may not generalize well to datasets with more complex emotional states. Another research [19] which included transfer learning, combined Tunable Q Wavelet Transform (TQWT) with Rotation Forest demonstrated that lightweight frameworks could achieve high accuracy (93%). The primary outcome of this research demonstrated the effectiveness of combining TQWT and RFE for EEG-based emotion recognition, providing a computationally lightweight and accurate model. However, a limitation of the study is the focus on only three emotion categories (positive, negative, neutral), and the method has not been validated on larger, more diverse datasets or additional emotion categories like surprise or anger.

The study [18] aimed to develop a robust multi-modal emotion recognition framework by combining EEG signals and facial expressions. The proposed system integrated a 3D-Convolutional Neural Network (3D-CNN) with ensemble learning techniques, including stacking and bagging, to improve accuracy for valence and arousal

emotion classifications. The stacking-based ensemble approach achieved superior recognition accuracies of 96.13% for valence and 96.79% for arousal classes, outperforming existing methods. The proposed fusion system demonstrated significant improvement compared to single-modality models and other fusion techniques in the literature. The study faced challenges in real-time application due to the computational complexity and offline nature of the models. The DEAP dataset's use of multiple electrodes around the face posed issues for face segmentation, affecting emotion recognition accuracy from facial data. Future work needs to address these limitations by developing online systems and refining fusion techniques for better performance.

To improve the accuracy of emotion recognition through shared feature representations and explore unimodal, multimodal, and cross-modal learning approaches. The researchers of the paper [1] conducted their research by fusing EG and eye movement data using Bimodal Deep Autoencoders (BDAE). The BDAE model was able to improve accuracy to 91.01% but ross-modal learning (training with one modality and testing with another) showed that shared representations extracted using BDAE achieved 66.45% accuracy. The limitations to these studies were high computability and are still limited.

In the paper [38] the researchers aimed to evaluate the efficiency of LLMs compared to traditional multilayer feedforward networks (MLPs) in predicting numerical patterns. Specifically, it explored whether LLMs could not only predict response variables from predictor variables but also identify the inherent numerical functions governing the data. The outcome of the rescuers showed LLMs demonstrated an ability to approximate the numerical functions governing the data, as shown by gradient evaluations closely matching the theoretical derivatives. A similar study [26] where LLMs used numerical inputs introduced signal processing concepts into LLMs by learning time-frequency representations for intermediate activations during training. The primary goal of the study was to understand how LLMs would work with numerical data and it was successful in showcasing a framework that would enable GPT-like models to understand numerical features. The study had a major limitation with its dataset where the dataset was created by the researchers and the pipeline would need to be verified over a larger, more renowned dataset.

The study paper [37] introduces a framework, that integrates Brain-Computer Interface (BCI) devices with Large Language Models (LLMs) to provide emotional support through conversational interactions. Challenges in integrating BCI-based emotion recognition with LLM-powered conversational agents for emotional support tasks. The framework combines EEG-based emotion recognition and improves emotional well-being through supportive interactions. The study included LLMs like LLaMA 2.0 to facilitate real-time monitoring of users' emotional states and provide tailored dialogues aimed at alleviating mental distress. The framework was able to dynamically detect emotions using EEG signals and adapt conversations to guide users toward positive emotional states. The drawback of this study is that the framework has not been tested in real-world scenarios and there's indication of lack of robustness. The paper [34] introduces the Thought2Text framework, which leverages fine-tuned LLMs with multimodal data (EEG and image embeddings) to generate descriptive text from EEG signals. The model outperformed baseline models,

demonstrating the importance of multimodal alignment and fine-tuning. In addition, the framework showcases the potential for portable and affordable "thought-to-text" systems, with promising applications in assistive technologies, mental health, and beyond. Drawback of this study is that the model suffers from misclassification with noisy EEG signals, limited generalizability due to a small participant dataset, and struggles with hallucinations and text generation not grounded in EEG inputs. Paper [27] talks about EEG Emotion Copilot, a lightweight large language model system that addresses the challenges in EEG-based emotion recognition and assisted medical record generation. The model processes EEG signals to identify emotional states, generates personalized diagnostic and treatment suggestions, and automates the creation of electronic medical records (EMRs). By leveraging a novel prompt data structure, model pruning, and fine-tuning techniques, the copilot achieves real-time performance and computational efficiency, which makes it suitable for deployment on resource-constrained devices. The challenges found in this study include the difficulty of building multimodal datasets for LLMs and processing EEG as long-sequence data which leads to significant computational overhead. Additionally, pruning lightweight LLMs can compromise output quality compared to larger models. The study of paper [28] focused on developing an automated system that integrates EEG data and differential large language models (LLMs) to enhance language teaching. Here the research introduced a system that combined an EEG fusion module with differential LLMs to optimize language teaching by dynamically monitoring, analyzing, and personalizing the learning process. The differential LLM adapts learning content based on individual needs, such as language proficiency and cognitive abilities, while the EEG fusion module analyzes brain activity in real time to adjust teaching strategies. Their model out outperformed baseline models and traditional teaching methods. Although models like LLama and GLM performed well, the proposed system consistently delivered better results due to its integration of real-time EEG analysis and personalized teaching adjustments. The researchers of the paper [36] investigate the use of large language models (LLMs) to estimate attention states, sleep stages, and sleep quality while providing personalized feedback for sleep improvement. This work integrates EEG data and physical activity data with LLMs, aiming to enhance health monitoring and adaptive interventions. The main objectives of this paper were to provide personalized sleep improvement suggestions and guided imagery scripts using EEG and user profile data. Additionally, The study also compared the performance of fine-tuned LLMs, in-context learning, and traditional machine learning models like XGBoost across these tasks. The drawback of this study was that the LLMs were not performing as well as the traditional ML models. Also the LLMs struggled with high-dimensional data and failed to provide reliable predictions for EEG-based attention states. The researchers in [41] developed a semi-supervised learning framework named PAWS to enhance EEG-based emotion recognition, particularly in scenarios with limited labeled data. PAWS integrates "Intermediate Mixup" to improve domain adaptation by generating robust features. The outcome showed PAWS significantly improved accuracy and robustness in EEG-based emotion recognition, particularly in scenarios where there's scarcity of data, setting a new benchmark for semi-supervised learning methods. The method showed some sensitivity to noise, as higher noise levels led to slight decreases in accuracy.

The study of paper [29] reviewed the potential of large language models (LLMs) in enhancing text-centric multimodal sentiment analysis, with a focus on their adaptability, advantages, and limitations in integrating modalities like images, videos, and audio. The outcome of the study showed LLMs gave better results in zero-shot and few-shot settings for sentiment analysis tasks. For instance, it achieved an F1 score of 0.9036 on MER2023-SEMI and the highest WAR (59.37%) in zero-shot evaluations on the DFEW dataset. The reliance on annotated datasets like MERR limits scalability, and there was no integration of other modalities like EEG. Additionally, while the model outperformed in multimodal integration, audio analysis remains underexplored. Another study [40] used large language models (LLMs) in text-centric multimodal sentiment analysis, focusing on their adaptability, strengths, and limitations in integrating text, images, and audio for sentiment tasks. The results showed effectiveness in zero-shot and few-shot settings for sentiment analysis, with their reasoning and contextual knowledge adding value. However, They lack support for non-text modalities like physiological signals, and high training costs make them less accessible. To develop an open-vocabulary framework for multimodal emotion recognition, the researchers of [39] integrated audio, video, and textual data using LLMs without requiring fine-tuning. The proposed system outperformed Video-ChatGPT and generated a more diverse range of emotion words highlighting the complexity of multimodal reasoning. However, its performance lagged behind GPT-4V. Additionally, The approach relied on well-crafted prompts, which can be error-prone. Few-shot learning failed to enhance performance, and the model showed limited adaptability without fine-tuning. The computational overhead of integrating multiple feature extraction models posed challenges for scalability.

The paper [20] talks about the enhancement of EEG-based emotion recognition by developing a stacking ensemble model that classifies emotions into three categories: positive, negative, and neutral. The study aimed to improve classification accuracy and address the limitations of existing models. The proposed model achieved a high classification accuracy of 99.55%, outperforming standalone models and existing methods. The dataset used was built with data collected from only two individuals, which limits generalizability. Additionally, the study relied on a limited number of EEG channels, potentially affecting the model's ability to capture complex brain activity patterns. Another paper [43] to classify motions into three categories (positive, negative, neutral) using EEG signals, combined Multi-Class Common Spatial Pattern (MCCSP) for feature extraction with an ensemble of deep learning models. The proposed method achieved a classification accuracy of 99.44%, outperforming traditional methods. The model proved its strength in dealing with noise and effectiveness across various frequency bands, with the highest accuracy observed in the alpha band (99.96%). The limitation factors of this data were that the dataset size was limited, and the study did not consider gender-based differences in emotional responses. The use of culturally specific musical stimuli limits the generalizability of the findings across different populations.

Chapter 3

Dataset and Analytics

This study utilized the SEED-IV dataset, a standard benchmark for emotion recognition using EEG signals. To ensure comprehensive data collection, 72 film clips that were likely to evoke feelings of happiness, sadness, fear, or neutrality were selected. The experiment involved 15 subjects in total. Three sessions, each with 24 trials, were conducted for each participant on various days. In one trial, the participant watched a movie clip while the 62-channel ESI NeuroScan System and SMI eye-tracking glasses recorded his or her eye movements and EEG signals [7].

3.1 EEG Data

During the experiment, EEG signals were recorded which captured the brain activity while participants watched specific emotion-evoking film clips. The captured signals were first downsampled to 200 Hz sampling rate, after which a band-pass filter of 1-75 Hz was applied to minimize noise and artifacts. This ensured that only the necessary frequency were retained in the acquired signal. Finally, each session was then segmented into non-overlapping 4-second windows, representing each sample of the data [7].

After data collection, some smoothing and feature extraction techniques were applied to extract the spatial information from the signal. This resulted in a dataset consisting of EEG features derived after applying Differential Entropy (DE) and Power Spectral Density (PSD); two commonly used feature extraction techniques in EEG signal analysis. These features were computed across five distinct frequency bands:

- Delta (1-4 Hz)
- Theta (4-8 Hz)
- Alpha (8-14 Hz)
- Beta (14-31 Hz)
- Gamma (30-50 Hz)

3.1.1 Class Distribution (Pre-String Conversion)

To gain insights into the representation of emotion labels within the dataset, the class distribution of the EEG data was analyzed in this study.

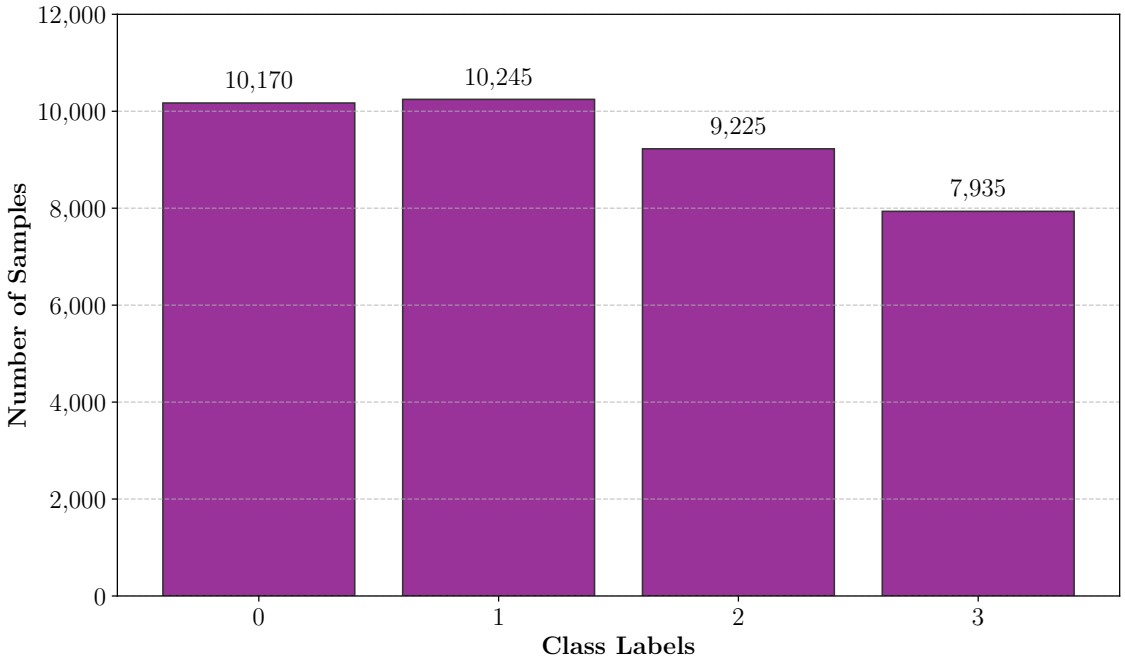


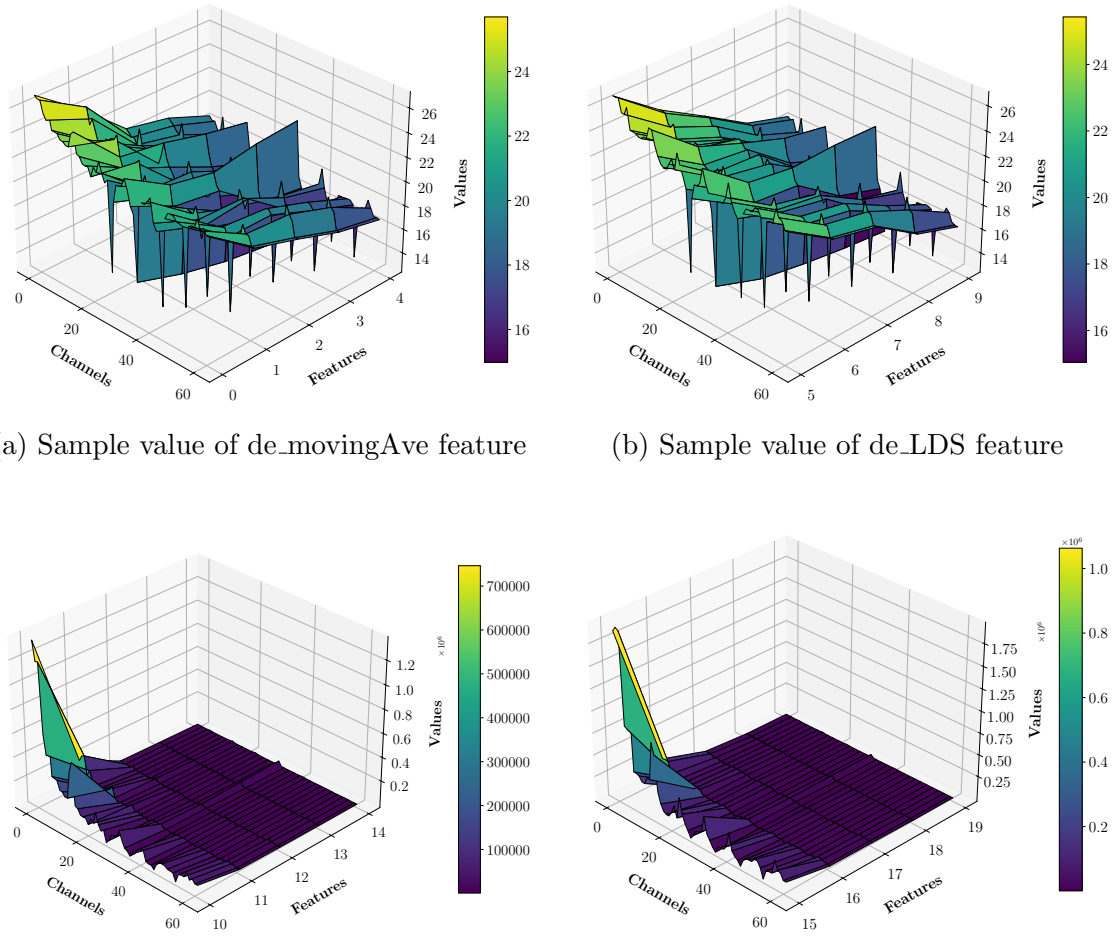
Figure 3.1: Class distribution of EEG data

Number of samples for each emotion category (**0: neutral**, **1: happy**, **2: sad**, **3: fear**). The distribution is reasonably balanced across the four classes, ensuring that no significant bias exists toward any single emotion category, making the dataset suitable for classification tasks.

3.1.2 Feature Analysis

Sample Value

To better understand the characteristics of the EEG data after preprocessing, an analysis was conducted on the feature values across all channels and extracted features. The focus of this analysis was on the sample, mean, and standard deviation (SD) values of the features, visualized through 3D surface plots for each feature group.



(a) Sample value of de_movingAve feature

(b) Sample value of de_LDS feature

(c) Sample value of psd_movingAve feature

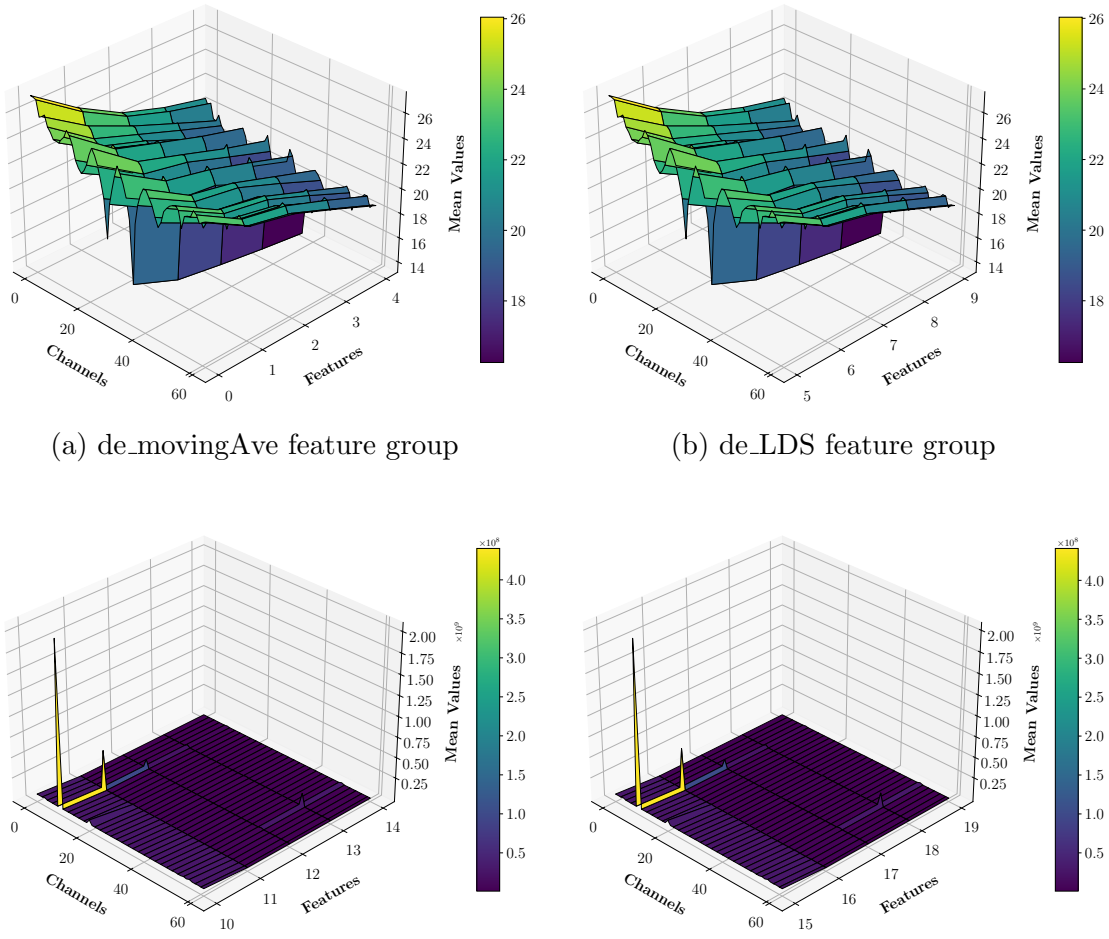
(d) Sample value of psd_LDS feature

Figure 3.2: Comparison of an EEG data sample across different feature ranges.

Figure 3.2 illustrates the spatial distribution of feature values across EEG channels and frequency bands, providing a detailed snapshot of the raw feature space.

Mean

To analyze the general trends in the dataset, the mean of the features were computed across all tuples.



(a) de_movingAve feature group

(b) de_LDS feature group

(c) psd_movingAve feature group

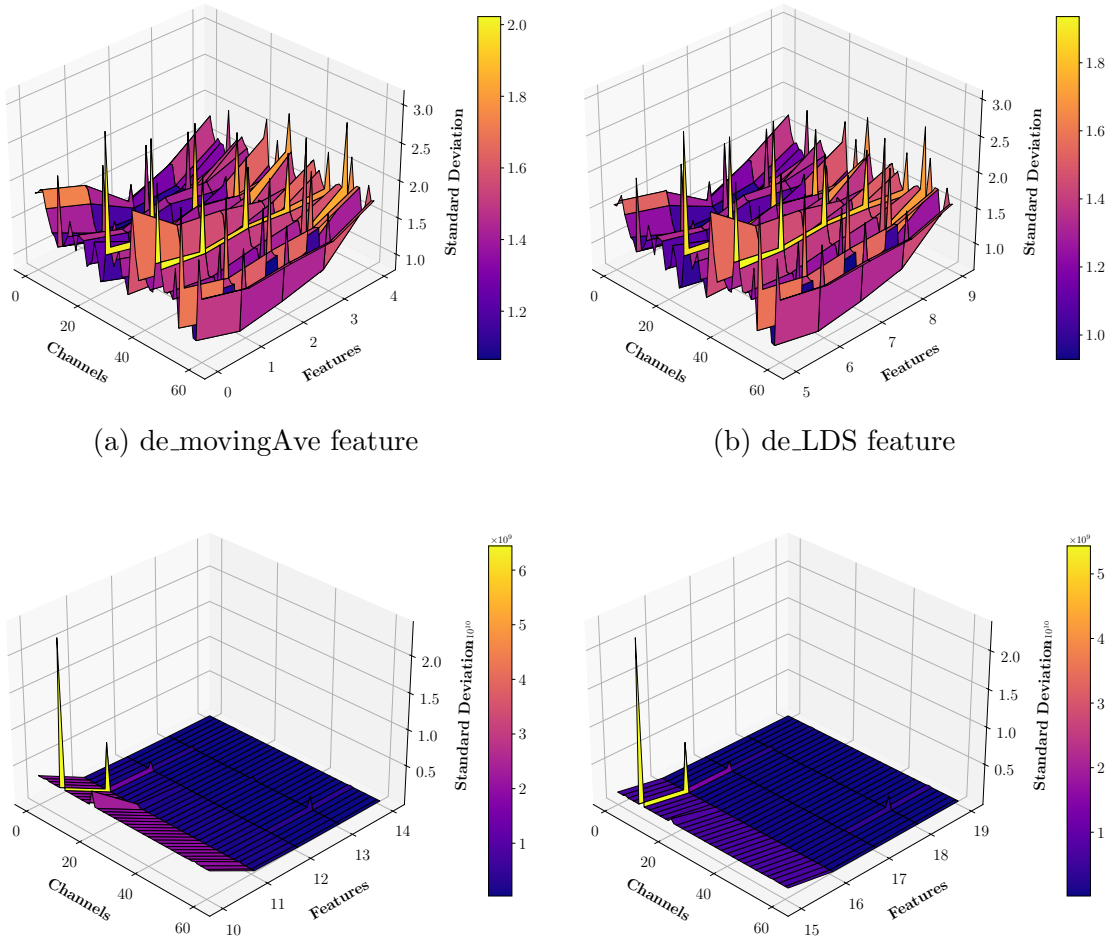
(d) psd_LDS feature group

Figure 3.3: Comparison of mean values across different feature ranges.

The figure (3.3) above shows the mean of the each feature group. It illustrates how the values of each feature group are distributed on average across all channels and frequency bands.

Standard Deviation

To understand the variation in the each feature group of the dataset, the standard deviation were calculated and illustrated in the following figure(3.4):



(a) de_movingAve feature

(b) de_LDS feature

(c) psd_movingAve feature

(d) psd_LDS feature

Figure 3.4: Comparison of SD values across different feature ranges.

3.1.3 Class Distribution (Post-String Conversion)

After the contextualization and string conversion process, the class distribution of the dataset was re-evaluated to ensure coherence to the original data. The total number of samples in the dataset increased significantly and proportionally, as each sample of the unprocessed data was split channel-wise.

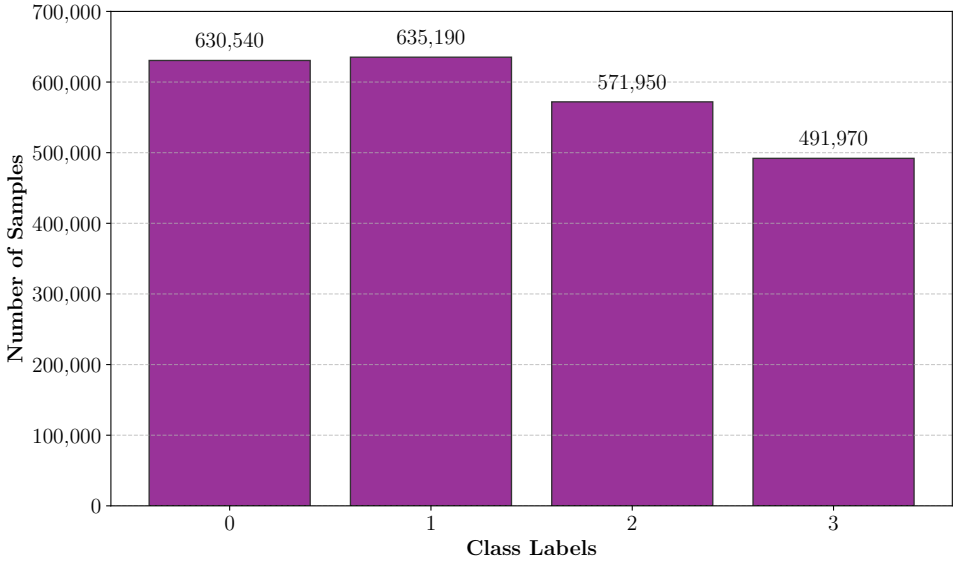


Figure 3.5: Class Distribution of Single Channel String Dataset

3.1.4 String Length Analysis

String lengths of the single-channel dataset were analyzed to verify the consistency of the generated textual representations. The graphical visualization, shown in Figure 3.6, utilizes a logarithmic scale to highlight the bin-wise distribution of string lengths across samples. The lengths range from approximately 356 to 369 characters, and the log scale provides an enhanced perspective of the distribution. Such uniformity is critical for efficient tokenization and processing by large language models.

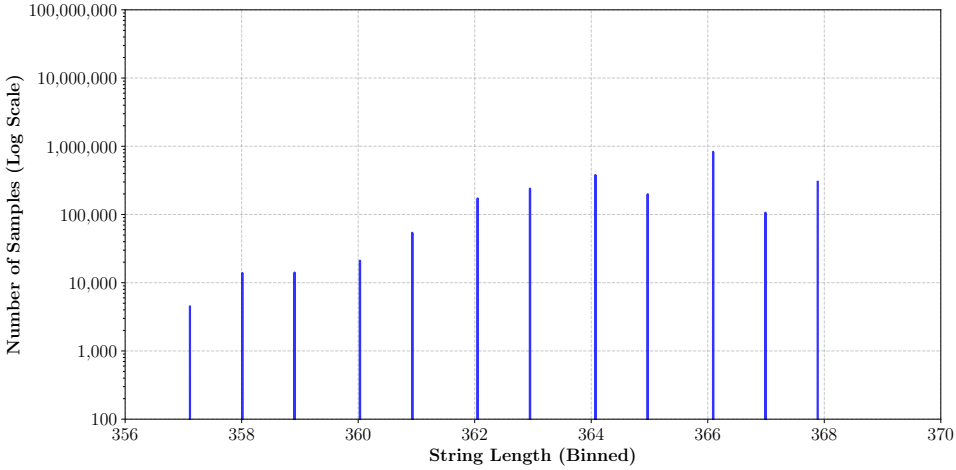


Figure 3.6: String Lengths of Single Channel String Dataset

3.1.5 Tokenization Analysis

The compatibility of the EEG dataset's string representation with the FLAN-T5, GPT-2, and BERT models was assessed through a tokenization analysis. For each model tokenizer, a token count distribution graph was generated to check the distribution of tokens without any truncation or padding in the string dataset. After checking the distribution, an appropriate token limit wass decided for each tokenizer.

FLAN-T5 Tokenization

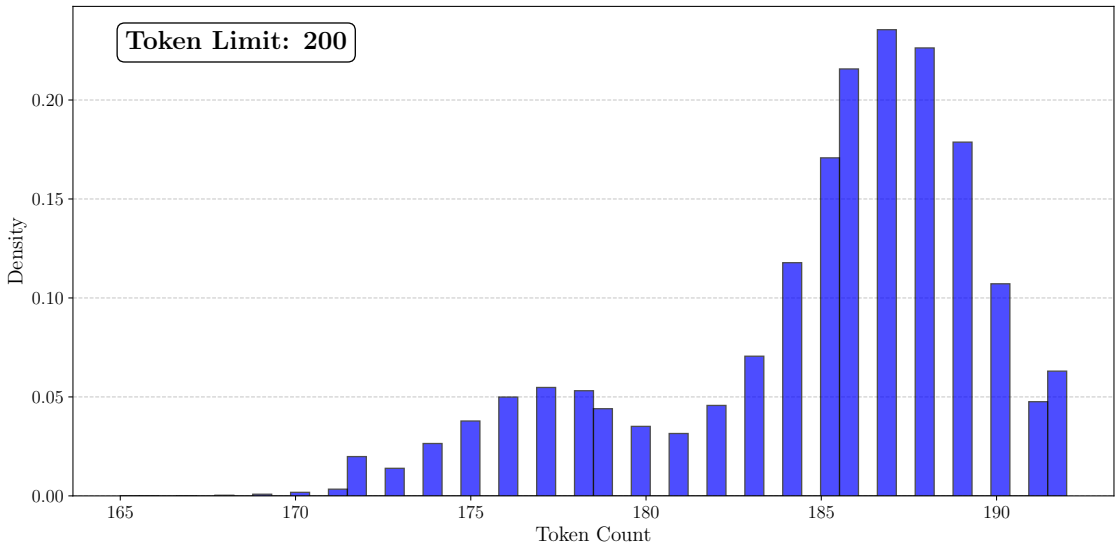


Figure 3.7: Token Count using T5 Tokenizer

Figure 3.7 shows the token count distribution for FLAN-T5, with the majority of samples falling between 185 and 195 tokens. Based on this analysis, the maximum token length was set to 200 tokens, ensuring that all samples fit within the model's constraints.

GPT-2 Tokenization

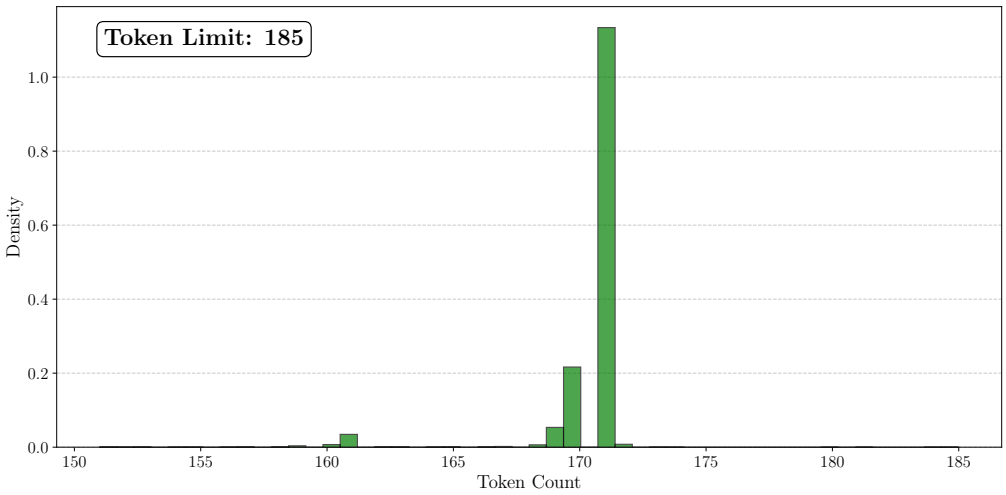


Figure 3.8: Token Count using GPT2 Tokenizer

As shown in Figure 3.8, the token counts for GPT-2 were highly consistent, primarily concentrated around 170 tokens. To accommodate all samples, the token limit was set to 185 tokens, ensuring that the tokens remains consistent with the GPT-2 model.

BERT Tokenization

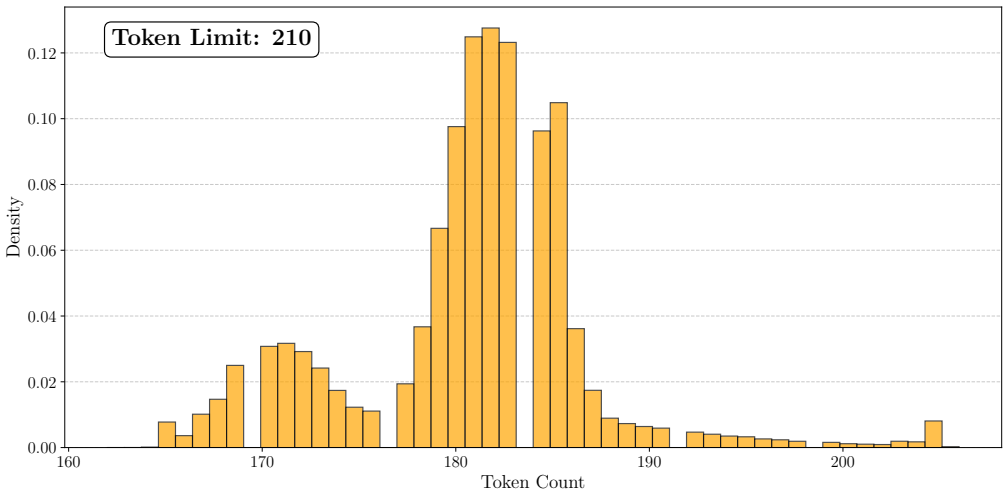


Figure 3.9: Token Count using BERT Tokenizer

For BERT, Figure 3.9 demonstrates that most token counts ranged between 180 and 190 tokens, with a small fraction exceeding this range. To ensure compatibility, the token limit was set to 210 tokens, allowing for minor variability without truncation. The analyses across all three models demonstrate that the string representations are well-optimized for tokenization, with all samples fitting comfortably within the respective token limits. This consistency ensures that the datasets are effectively prepared for processing by each model during training and evaluation.

3.2 Eye-Tracking Data

The eye movement data in the SEED-IV dataset was collected using SMI Eye Tracking Glasses, a device designed to record detailed eye movement patterns during experiments. The participants' eye movements were tracked while they watched the emotion-inducing film clips, ensuring that the recorded data captured responses corresponding to the target emotions.

For each trial, a range of features was extracted from the raw eye movement recordings. These features represent various aspects of eye behavior that have been shown to correlate with emotional states. The extracted features include:

- **Pupil Diameter (1-12)**: Captures the size of the pupil in two dimensions.
- **Dispersion (13-16)**: The range of spread of eye movements.
- **Fixation Duration (17-18)**: The duration for which the gaze remained fixed on a specific location.
- **Saccades (19-22)**: Includes metrics such as saccade duration and amplitude, referring to rapid eye movements.

3.2.1 Feature Analysis

To understand the characteristics of the preprocessed eye movement data, a sample value, mean, and standard deviation of the 22 features were analyzed.

Sample Value

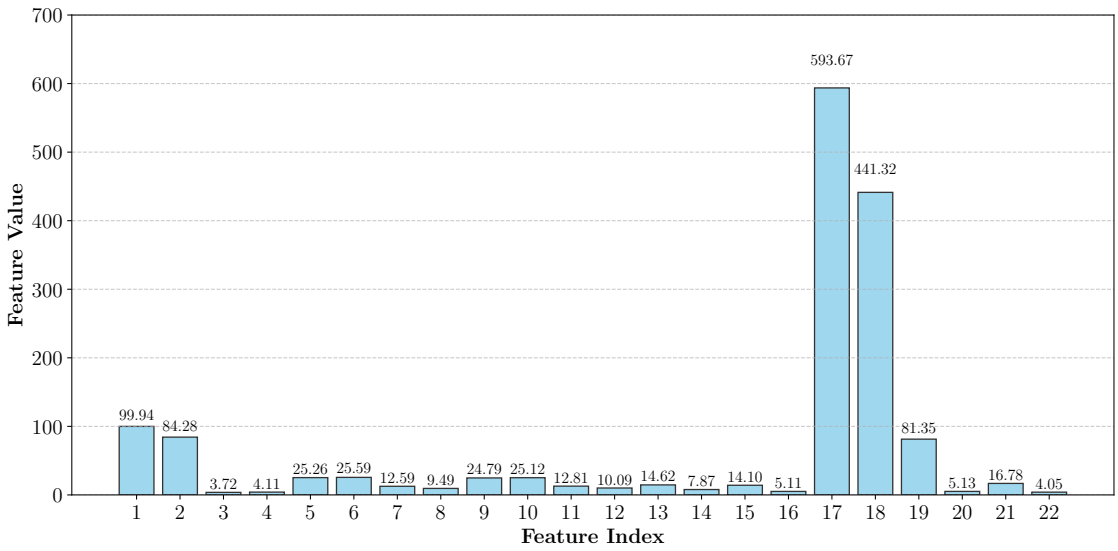


Figure 3.10: Bar Chart of the features of a Sample Eye Data

Figure 3.10 shows the feature values for a single randomly selected sample from the dataset. It highlights the range of values for each feature of a sample of the eye data.

Mean

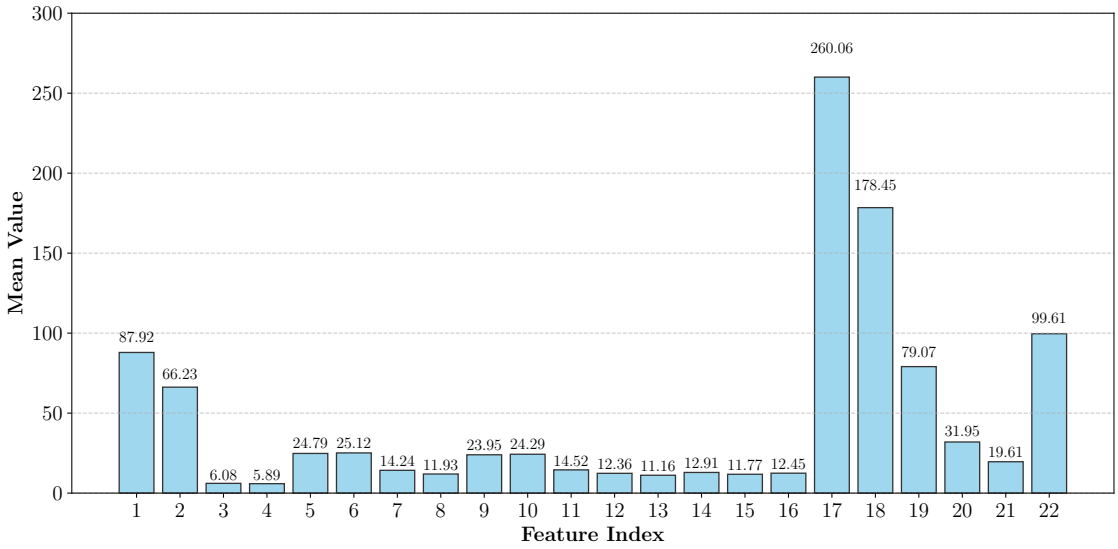


Figure 3.11: Bar Chart of the mean of all the features of Eye Movement Data

Figure 3.11 shows the mean of each feature of the dataset. It highlights the importance of certain features in capturing variations in eye movement data associated with emotional states.

Standard Deviation

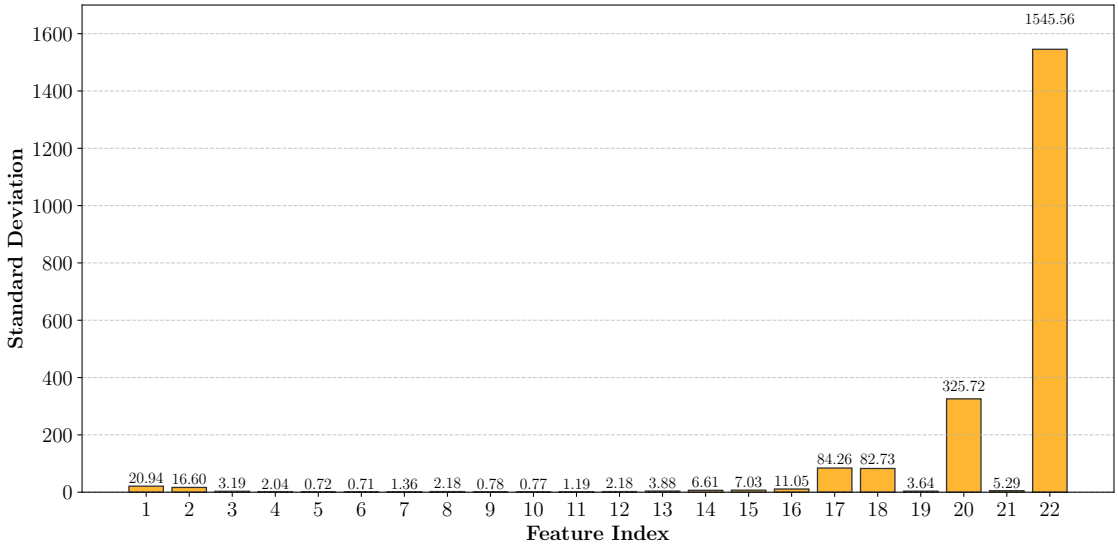


Figure 3.12: Bar Chart of the Standard Deviation of all the features of Eye Movement Data

Figure 3.12 shows the standard deviation of each feature of the dataset. Features with higher variations denotes significant fluctuations across samples.

3.2.2 String Length

After converting to string format, the string lengths of the eye movement dataset were analyzed to assess the consistency of the textual representations. String lengths need to be uniform as the efficiency of the tokenization and downstream process depend on it. The analysis shows that the conversion process is compatible with all tokenization requirements and will be effective when processed by LLMs. Figure 3.13 illustrates the distribution of string lengths across all samples.

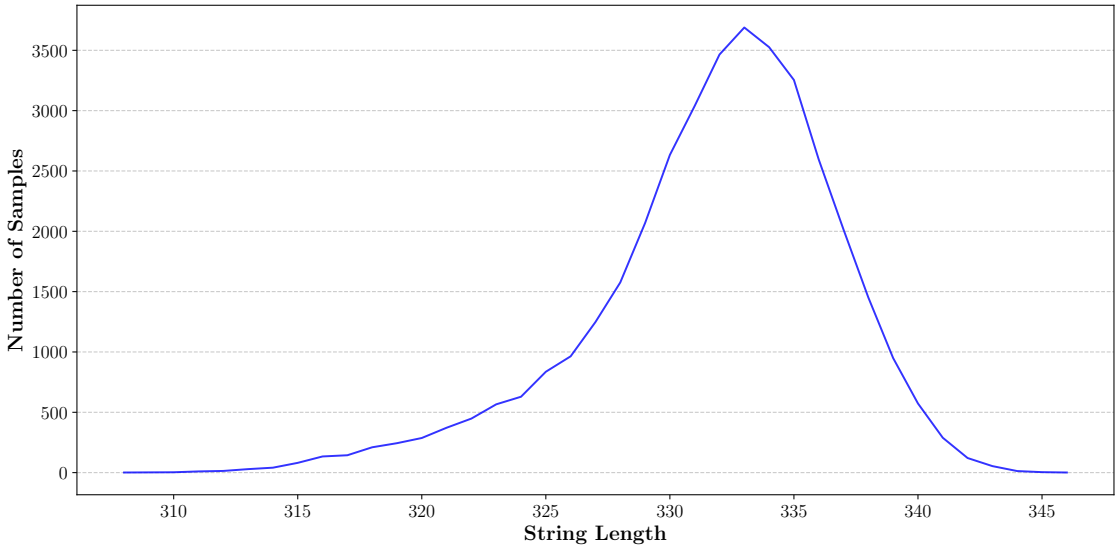


Figure 3.13: String lengths of Eye String Data

The plot reveals that string lengths range from approximately 310 to 345 characters, with a peak of around 335 characters. This indicates a high degree of consistency in the string conversion process as the lengths of all samples are within a small range. Such uniformity is critical for efficient tokenization and downstream tasks, as it reduces the need for extensive preprocessing adjustments.

3.2.3 Tokenization Analysis

The tokenization analysis for the eye movement dataset was conducted to evaluate its compatibility with three large language models: FLAN-T5, GPT-2, and BERT. Token count distributions were examined to ensure that the string representations adhered to the token limits of each model without requiring truncation.

FLAN-T5 Tokenization

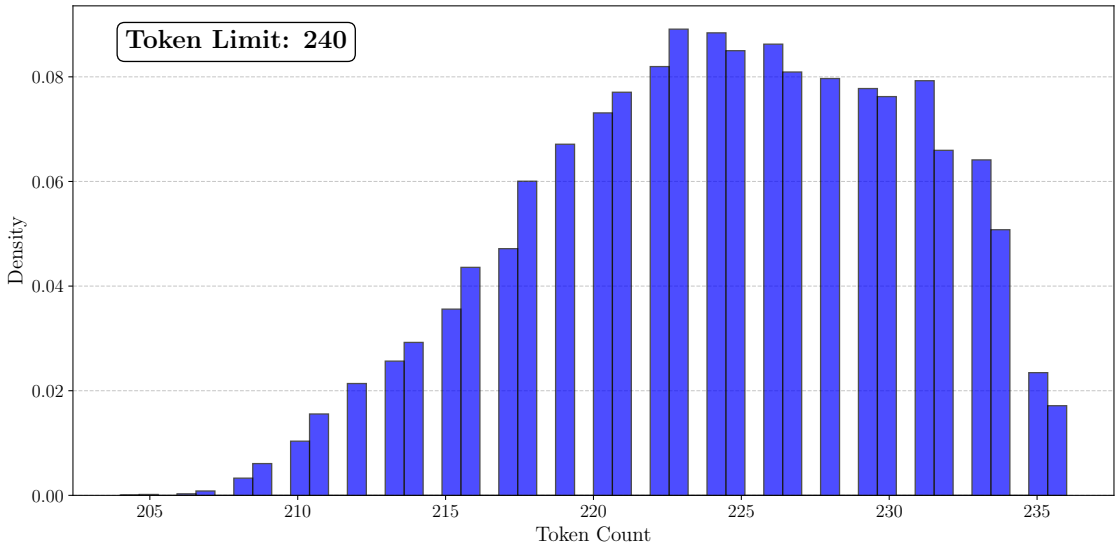


Figure 3.14: Token Count using T5 Tokenizer

Figure 3.14 presents the token count distribution for FLAN-T5, where most samples had token counts ranging between 220 and 230 tokens. Based on this distribution, the token limit was set to 240 tokens, ensuring no truncation occurred, while consistent padding was applied to maintain uniform input lengths during processing.

GPT-2 Tokenization

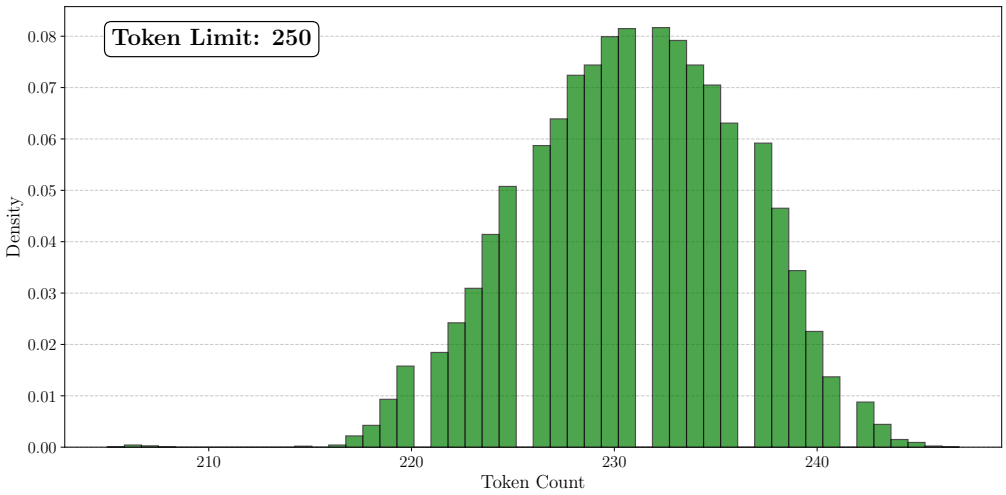


Figure 3.15: Token Count using GPT2 Tokenizer

The token count distribution for GPT-2, as shown in Figure 3.15, was observed to concentrate primarily between 220 and 240 tokens, with minimal variability. A token limit of 250 tokens was chosen to ensure that all samples were included, accompanied by consistent padding for streamlined model processing.

BERT Tokenization

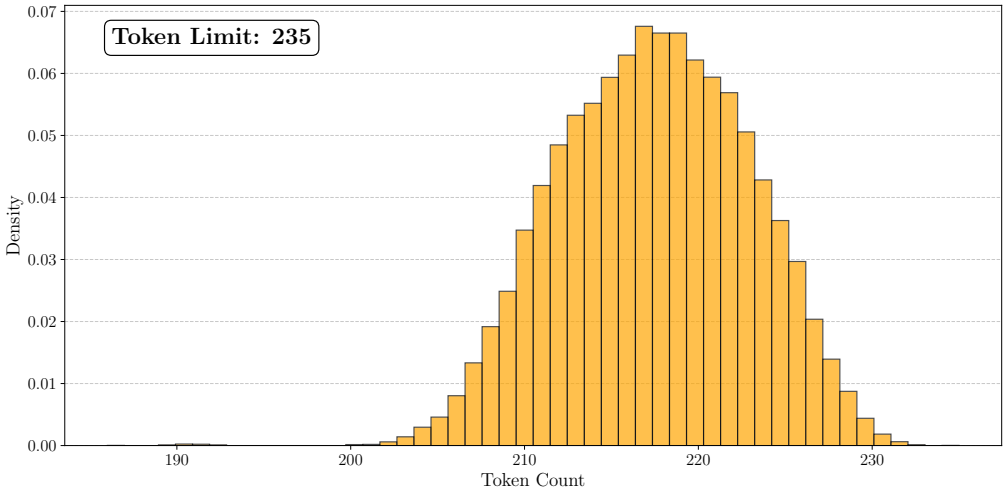


Figure 3.16: Token Count using BERT Tokenizer

As illustrated in Figure 3.16, the token counts for BERT ranged predominantly between 210 and 230 tokens. To ensure compatibility with the dataset, a token limit of 235 tokens was selected. Padding was applied consistently to align all samples to the specified length. The analysis confirmed that the eye movement dataset was well-suited for processing by all three models. The selected token limits ensured that no samples were truncated, and consistent padding facilitated uniformity across the dataset, enabling efficient model training and evaluation.

Chapter 4

Methodology

This section gives a theoretical and in-depth description of each component of the proposed approach. The approach consists of three key phases: (1) Data transformation, where EEG and eye-tracking data are converted into structured text format. (2) Model fine-tuning, where pre-trained LLMs are adapted to classify emotion states based on the transformed data. (3) Ensemble learning approach, where predictions from multiple LLMs are combined using a stacked ensemble approach. Fig 4.1 shows a top-level view of our novel approach. In the data transformation phase, high-dimensional EEG and eye-tracking features are structured into contextualized string formats to enable compatibility with LLM tokenization processes. This involves converting raw numerical sequences into meaningful tokenized representations, ensuring that the contextual relationships between numerical features are preserved in the transformed dataset. The model fine-tuning phase involves training three pre-trained LLMs—Flan-T5, GPT-2, and BERT—on the structured dataset. These models represent three major LLM architectures: encoder-only (BERT), decoder-only (GPT-2), and encoder-decoder (Flan-T5). Each model is fine-tuned for emotion classification, where the input consists of the string representation of the original EEG and eye-tracking datasets, and the output corresponds to emotion labels. In the ensemble learning phase, predictions from individual LLMs are aggregated using a Stacked Ensemble method, with LightGBM as the meta-learner. This step addresses structural differences between EEG and eye-tracking data by resampling predictions, aligning labels, and leveraging ensemble learning to improve classification performance.

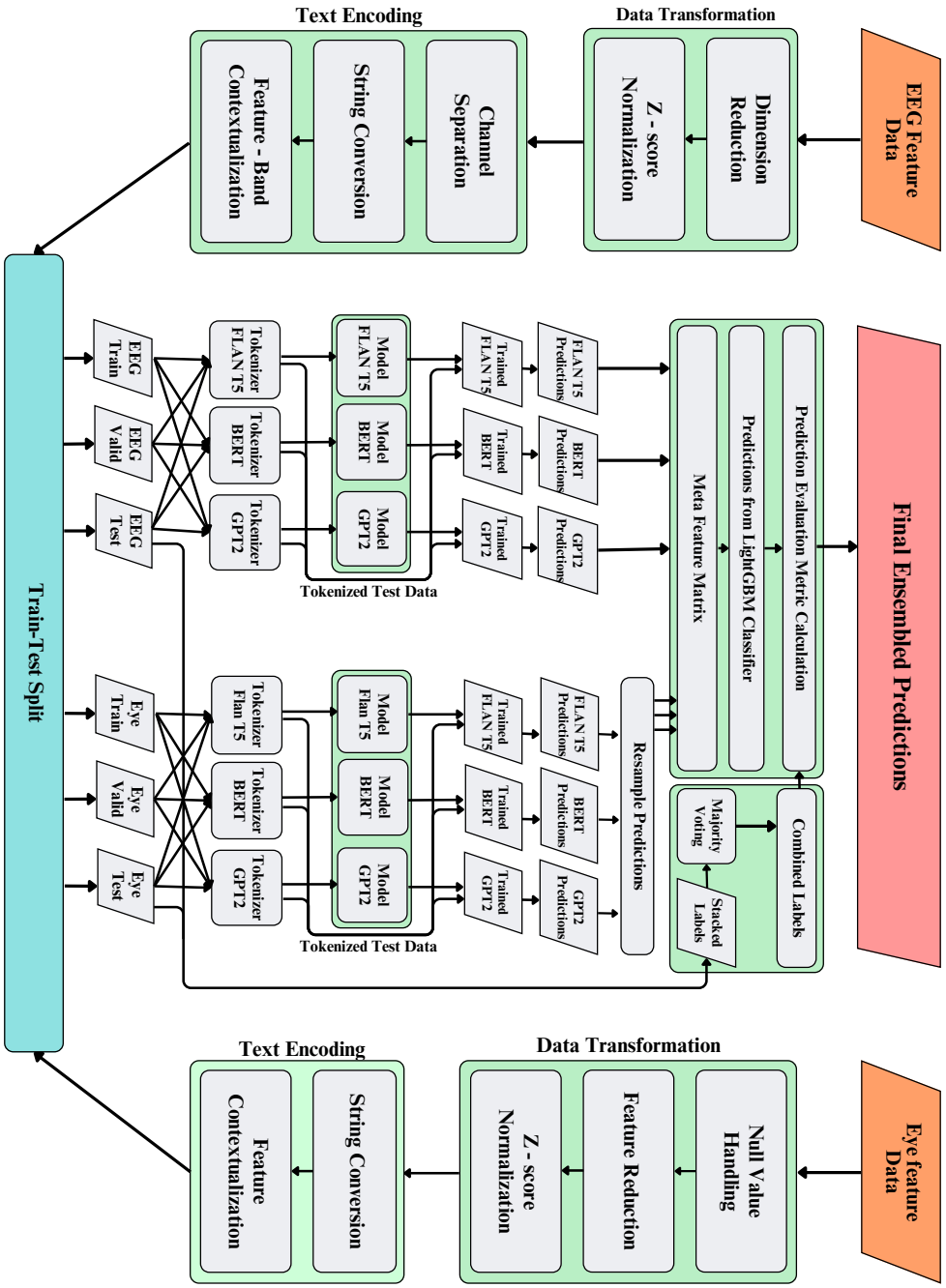


Figure 4.1: Top Level Layout of the proposed technique

4.1 FLAN-T5 (Encoder-Decoder Model)

Flan-T5 is a fine-tuned version of the Text-to-Text Transfer Transformer (T5) model, initially introduced by Raffel et al. [9]. It is an encoder-decoder transformer model that processes input sequences in a sequence-to-sequence manner, making it highly versatile for various natural language processing (NLP) tasks. Flan-T5 is available in several configurations, ranging from Flan-T5-Small (80 million parameters) to Flan-T5-XXL (11 billion parameters). Fig 4.2 shows the general architecture of the Flan T5 model

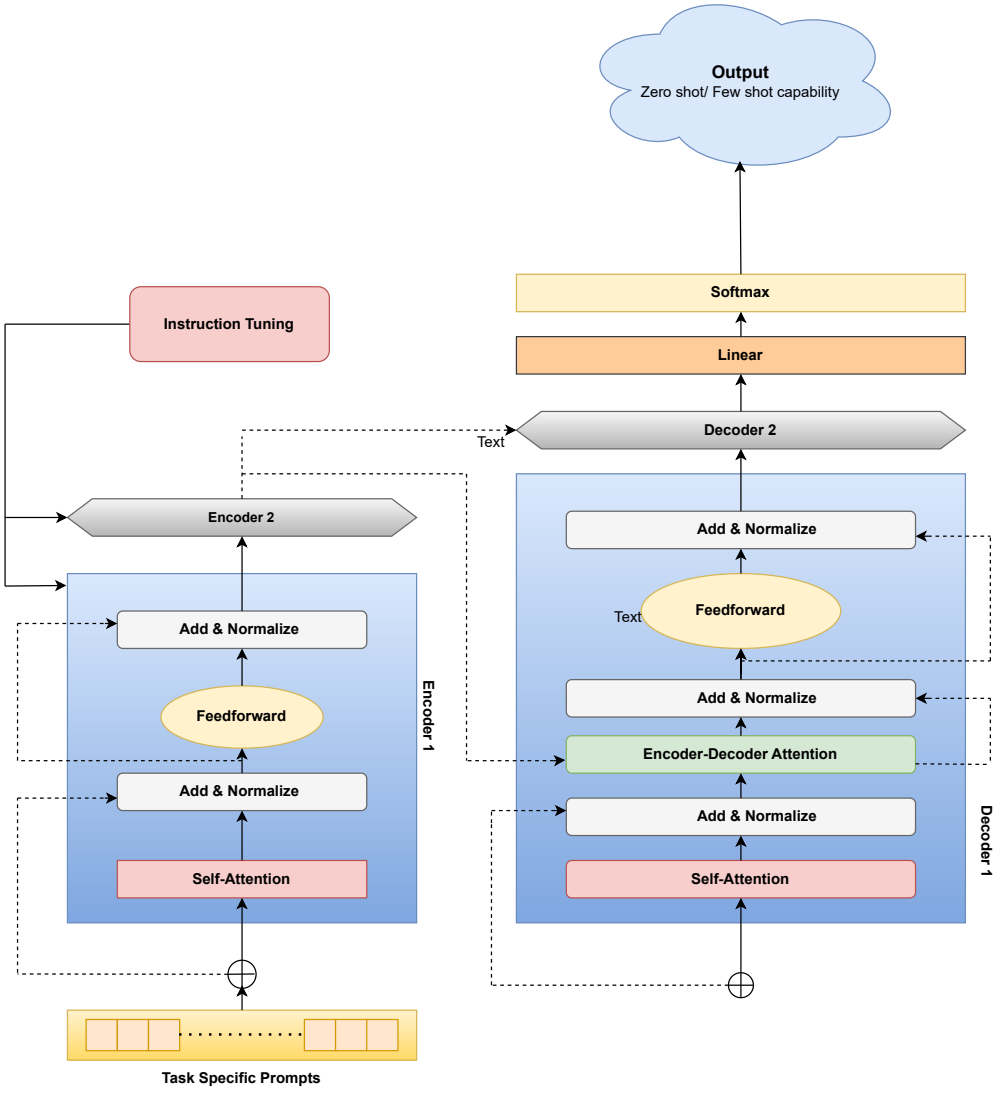


Figure 4.2: FLAN-T5 Model Architecture

4.1.1 Tokenization

Flan-T5 uses SentencePiece tokenization [6], which segments input sequences into subwords or tokens. Each token is then converted into a dense vector representation using pre-trained embeddings from Flan-T5's vocabulary.

4.1.2 Encoder Framework

- **Multi-Head Attention:** Captures contextual relationships between tokens in the input sequence using self-attention [4].

$$H = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{4.1}$$

where:

  - H is the output of the self-attention mechanism.
  - Q (Query) is the matrix containing query vectors that determine how much focus should be given to other words in the sequence.
  - K (Key) is the matrix containing key vectors that help in measuring the relevance of each word to the query.
  - V (Value) is the matrix containing value vectors that store the actual information to be attended to.
  - d_k is the dimensionality of the key vectors.
  - K^T represents the transpose of the Key matrix.
  - softmax is a function which normalizes attention scores to ensure that they sum to 1.

  This equation represents the scaled dot-product attention mechanism, a fundamental component of the Transformer architecture used in large language models (LLMs).

- **Feed Forward Layer:** Transforms intermediate representations through a two-layer feed-forward neural network.

$$\text{FFN}(x) = \max(0, xW_1 + b_1)W_2 + b_2 \tag{4.2}$$

where:

  - $\text{FFN}(x)$ is the output of the feed-forward network.
  - x is the input vector.
  - W_1 and W_2 are weight matrices.
  - b_1 and b_2 are bias vectors.
  - $\max(0, \cdot)$ represents the ReLU activation function, which introduces non-linearity by zeroing out negative values.

- **Other Layers:** Includes layer normalization and residual connections to stabilize training.

4.1.3 Decoder Framework

- **Masked Multi-Head Attention:** Uses masked self-attention to ensure autoregressive generation of tokens.

- **Cross-Attention:** Aligns decoder outputs with encoder-generated representations.

4.1.4 Handling Numeric Data

Flan-T5 utilizes SentencePiece tokenization, a language-independent subword tokenizer and detokenizer designed for neural text processing. SentencePiece segments text into subword units based on the frequency of character sequences in the training data, effectively handling both standard text and numerical data [31]. Unlike traditional tokenizers that rely on whitespace or predefined vocabularies, SentencePiece treats the entire input as a sequence of characters, allowing it to model languages without explicit word boundaries and handle diverse data types, including numerical values [6].

For example, the numerical value `"-0.2245"` might be tokenized into subword units such as (`'-'`, `'0'`, `'.'`, `'22'`, `'45'`), depending on the tokenization model's vocabulary. Each of these subword units is assigned an embedding vector, similar to how words in a regular text sentence are processed. This subword segmentation allows the model to handle numerical data effectively by leveraging its existing architecture for text processing.

4.2 GPT-2 (Decoder-Only Model)

The GPT-2 model, developed by OpenAI, is a decoder-only transformer-based large language model [8]. Unlike Flan-T5, GPT-2 follows an autoregressive framework, where it predicts each token sequentially based on preceding tokens. GPT-2 improves upon its predecessor, GPT, by increasing model size, training efficiency, and scalability. GPT-2 is available in various configurations, with 12 to 48 decoder layers and a maximum context length of 1024 tokens.

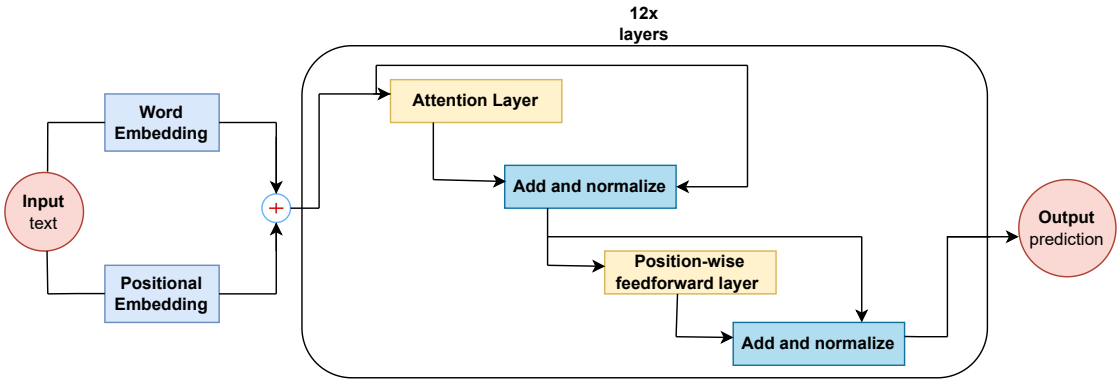


Figure 4.3: GPT-2 Model Architecture

4.2.1 Tokenization

GPT-2 employs Byte Pair Encoding (BPE) [2], segmenting text into subword units. Each token is embedded into a high-dimensional vector before being passed to the model.

4.2.2 Decoder Framework

- **Masked Multi-Head Self-Attention:** Prevents each token from attending to future tokens, ensuring autoregressive text generation.
- **Feed Forward Layer:** A two-layer fully connected network with ReLU activation transforms intermediate representations.
- **Layer Normalization:** Applied before attention and feed-forward layers, stabilizing training. Each token is transformed using formula (4.3)

$$z = \text{LayerNorm}(x + \text{MLP}(\text{SelfAttention}(x))) \tag{4.3}$$

  where:

  - x is the input to the layer.
  - SelfAttention(x) represents the self-attention mechanism, which computes contextualized representations of tokens based on their relationships.
  - MLP($\cdot$) denotes a multi-layer perceptron, typically composed of feed-forward neural network layers that refine the output from self-attention.
  - The residual connection x + MLP(SelfAttention(x)) helps to mitigate vanishing gradient problems and improves gradient flow during training.
  - LayerNorm($\cdot$) applies layer normalization to stabilize activations and improve convergence.

4.2.3 Handling Numeric Data

GPT-2 uses Byte Pair Encoding (BPE) [2], a subword tokenization technique that differs from the SentencePiece tokenization used in Flan-T5. While both methods break down text into smaller units, BPE learns frequently occurring character sequences without an explicit vocabulary size constraint, making it more adaptive to unknown words or numerical sequences. For instance, in GPT-2, the numerical value `"-0.2245"` might be tokenized as (`'-'`, `'0'`, `'.'`, `'22'`, `'45'`). GPT-2 does not view `"-0.2245"` as a distinct numerical entity. Instead, it sees the sequence of tokens leading up to it and attempts to generate a plausible continuation, rather than comprehending its numerical properties. Once tokenized, each subword unit is converted into an embedding vector before being passed into the model's transformer layers.

GPT-2 employs a learned word embedding matrix E where each token x_i is mapped to an embedding vector e_i:

$$e_i = E(x_i) \tag{4.4}$$

where:

- e_i represents the embedding vector corresponding to the input token x_i.
- $E(\cdot)$ is a pre-trained embedding lookup table that maps tokens to dense vector representations.

Numerical tokens such as "22" and "45" are treated similarly to regular text tokens, meaning they inherit contextual dependencies from the training data rather than possessing explicit numerical reasoning.

4.3 BERT (Encoder-Only Model)

BERT (Bidirectional Encoder Representations from Transformers) is a pre-trained encoder-only model designed for deep bidirectional contextual learning [5]. Unlike GPT-2, which is unidirectional, BERT captures context from both the left and right sides of a token, enabling improved language understanding. BERT is available in two main configurations: BERT-Base (12 encoder layers, 110M parameters) and BERT-Large (24 encoder layers, 340M parameters).

4.3.1 Tokenization

BERT employs WordPiece tokenization [3], which breaks text into subword units, improving generalization across rare words and domain-specific terms.

4.3.2 Encoder Framework

- **Multi-Head Self-Attention:** Enables bidirectional contextualization of all tokens in a sequence.
- **Feed Forward Layer:** Processes token representations after self-attention.
- **Embedding Layers:** Includes Token Embeddings, Segment Embeddings, and Position Embeddings.

4.3.3 Handling Numeric Data

BERT uses WordPiece tokenization [3], a subword segmentation technique that differs from Byte Pair Encoding (BPE) used in GPT-2 and SentencePiece tokenization in Flan-T5. WordPiece tokenization is designed to balance vocabulary efficiency and representational flexibility by breaking words into the most frequently occurring subword units, enabling BERT to handle rare words, domain-specific terms, and numerical data more effectively.

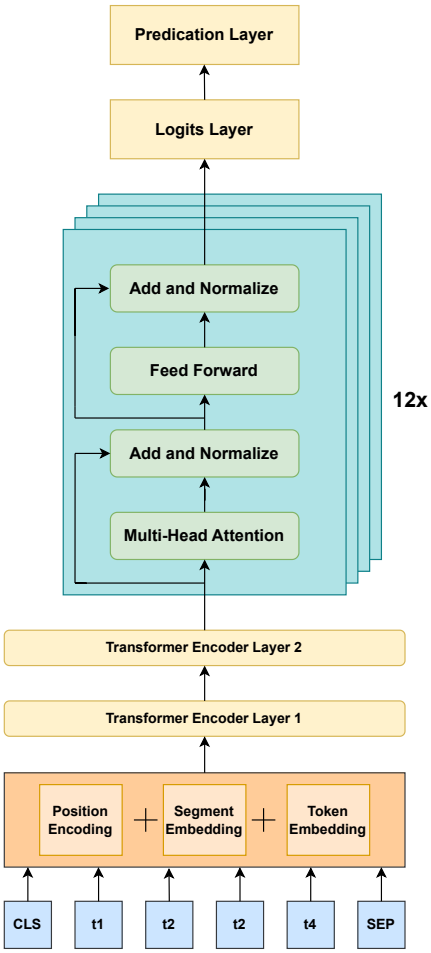


Figure 4.4: BERT Model Architecture

Using the previous example, the BERT model might tokenize the number as follows: `'-', '0', '.', '22', '##45'`, where the `##` symbol denotes subword continuation, indicating that "45" is part of a larger numerical unit rather than a standalone token. Unlike GPT-2's BPE, which learns subword segmentation dynamically based on training corpus frequency, WordPiece uses a predefined vocabulary of tokenized subword units, making it slightly more structured and deterministic when handling numerical values. The embedding vector representation follows the same approach as GPT-2.

Unlike GPT-2, which predicts text token by token in an autoregressive manner, BERT processes entire sequences bidirectionally, meaning each token is influenced by both preceding and succeeding tokens during training. This bidirectional attention mechanism allows BERT to better understand the context in which a number appears, even if it lacks explicit mathematical reasoning. BERT can learn associations between different numerical features based on their surrounding tokens, rather than just predicting the next token in a sequence like GPT-2. However, similar to GPT-2, BERT does not perform numerical operations directly; instead, it infers numerical relationships through token co-occurrence patterns seen during training.

4.4 Majority Voting

Majority voting is one of the simplest yet effective ensemble classification techniques widely used in predictive modeling. It is particularly useful when multiple models provide different predictions, as it aggregates their outputs to derive a consensus decision. In this research, majority voting was applied to integrate predictions from different models trained on EEG and eye-tracking data, ensuring a robust final classification.

Each model individually classifies a given input sample, generating a prediction. These predictions are then collected, and the final output is determined based on the majority label among them. In cases where an equal number of models predict different classes (a tie), strategies such as weighted majority voting or predefined rules can be used to resolve conflicts. The primary advantage of majority voting is its ability to leverage diversity among models, reducing individual model bias and improving generalization [24].

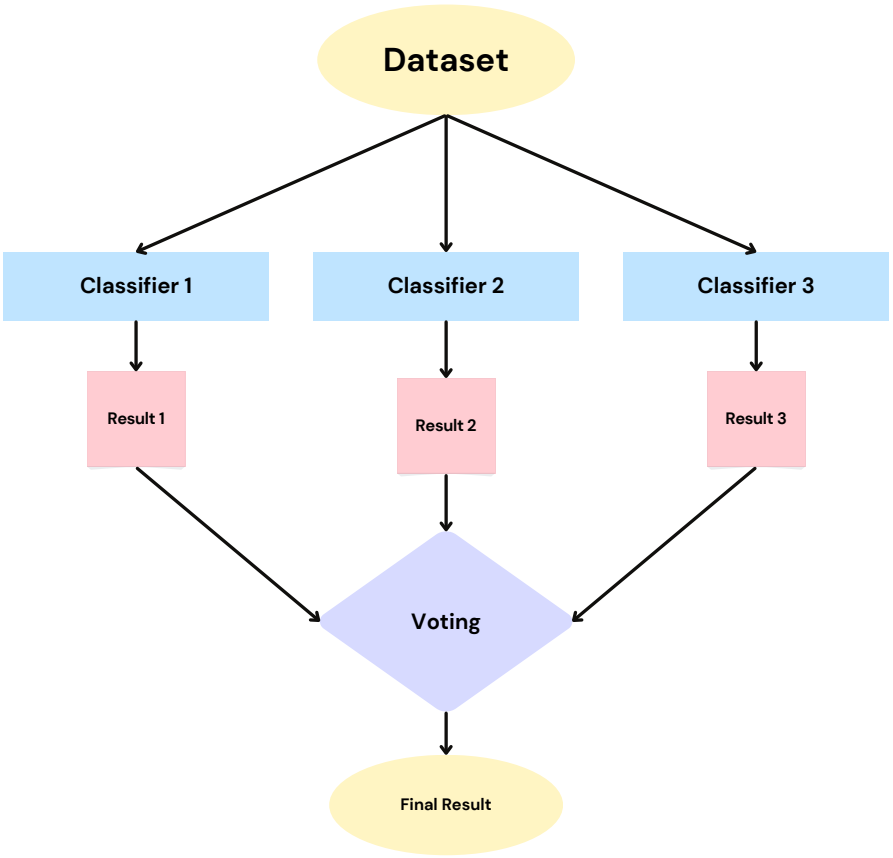


Figure 4.5: Majority Voting

4.5 Stacked Ensemble Learning

Stacked ensemble learning is an advanced technique that combines multiple base models and a meta-model to improve classification performance. Unlike majority voting, which selects the most common prediction, stacking uses a meta-learner to find the optimal combination of base model outputs for better decision-making.

The stacked ensemble consists of different components. First, there are the base models from which initial predictions are determined. Second, the outputs of the base models were used as features to train a secondary classifier, LightGBM, which learned the best way to combine predictions [16]. Third, These predictions were structured into a meta-feature matrix which was used to train the LightGBM Classifier.

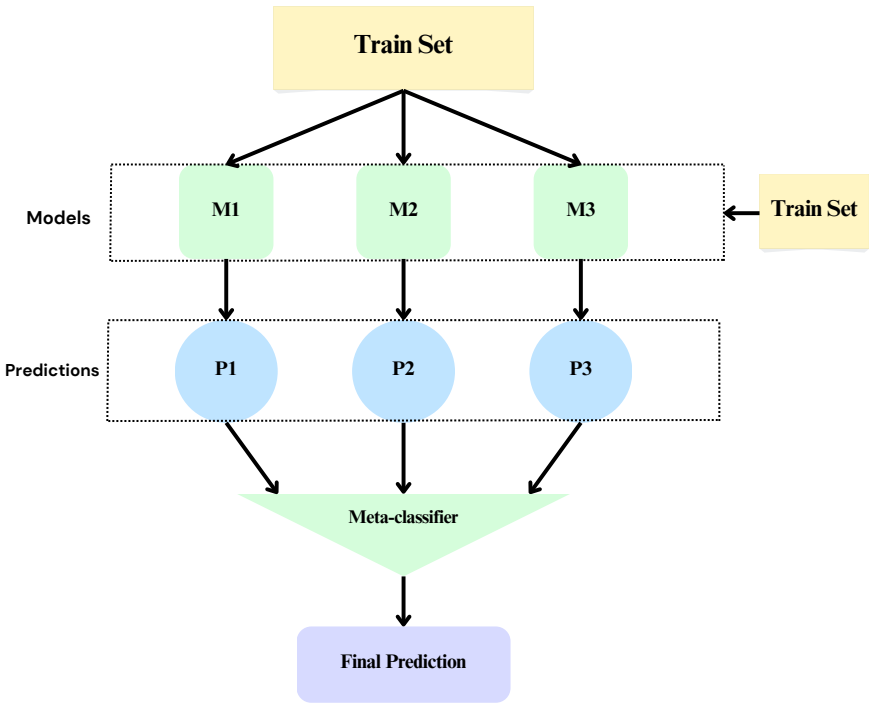


Figure 4.6: Stacked Ensemble Learning

Chapter 5

Implementation

For this experiment, pre-trained Large Language Models (LLMs) were selected and fine-tuned separately for each dataset type. The process of fine-tuning the LLMs were quite similar due to the models being pre-trained. Some adjustments were made based on their respective architecture. The implementation details for each model are described in separate sections.

5.1 Data Preprocessing

5.1.1 EEG Data

Preprocessing Using TorchEEG

The initial data from the dataset was highly complex and multi-dimensional, consisting of multiple features across numerous channels and trials. To reduce the complexity of the data, a specialized library called TorchEEG was used. Based on PyTorch, the TorchEEG library was created in order to handle EEG data and make it compatible for deep learning models. In addition, TorchEEG is able to provide a robust framework for preprocessing which allows us to transform the high-dimensional feature data into a more simplified structure.

TorchEEG allows easy EEG data conversion by specifying a standardized input-output format and implementing commonly used EEG datasets [44], including SEED-IV. This library includes various preprocessing techniques suited for datasets with high-dimentionality. Techniques like dimensionality reduction and normalization are often employed in EEG signal processing. TorchEEG provides a specialized class, SEEDIVFeatureDataset, built explicitly for handling the SEED-IV dataset. This class allows seamless integration of the dataset into the preprocessing pipeline by defining a standardized structure.

The key parameters that the SEEDIVFeatureDataset class accept to configure the dataset are as follows:

- **root_path**: Specifies the path of the extracted EEG data (`eeg_feature_smooth`).
- **feature**: List of specific features to be extracted from the dataset. For this study, `de_movingAve`, `de_lds`, `psd_movingAve` and `psd_lds` were selected.
- **online_transform**: Allows the application of preprocessing transformations directly to the EEG signals during data loading. In this study, this was used for normalizing the data using z-score normalization via `MeanStdNormalize()`.
  - `MeanStdNormalize()`: Used to standardize the EEG features by centering them around a mean of zero (0) and scaling them to have a standard deviation of one (1). The formula used for this normalization is:

    $$z = \frac{x - \mu}{\sigma} \tag{5.1}$$

    where x represents the feature value, μ is the mean of the feature values, and σ is the standard deviation.
- **label_transform**: Transforms the label information for each trial. In our case, this was used to map the labels to specific emotion categories (happiness, sadness, fear, neutral).

Structure of the Processed Data

Following the preprocessing steps, the final dataset consisted of 37,575 tuples, each containing a 2D array of shape 62×20 and an integer label representing the emotion category.

- The 62 rows in the 2D array correspond to the EEG channels, capturing data from all electrodes used during the recording process.
- The 20 columns represent features extracted from the EEG signals, specifically *de_movingAve*, *de_lds*, *psd_movingAve*, and *psd_lds* across five frequency bands: *delta*, *theta*, *alpha*, *beta*, and *gamma*.
- The integer label indicates the emotion category, with values ranging from 0 to 3 (**0: neutral**, **1: happy**, **2: sad**, **3: fear**).

This structured representation ensures that the dataset is compressed, easier to handle, and ready for further preprocessing.

String Conversion of EEG Data

To make the numerical EEG data compatible with large language models (LLMs), the data was represented in a contextualized string representation. This transformation adds contextual labels to the numerical features, enabling the LLMs to effectively interpret the data. The data from each channel was converted into a descriptive string representation. Features extracted across all five frequency bands were included in each sample.

The conversion procedure is explained below:

- **Input Data Structure:** The preprocessed EEG dataset consisted of 37,575 tuples. Each tuple contains 2 elements; a 2D array of shape 62×20, indicating 62 channels and 20 features per channel, and an integer label, indicating the emotion class.
- **Channel-wise Conversion:** For each tuple, the features corresponding to all 62 channels were iteratively processed. The 20 features for each channel were divided into four groups across five frequency bands:
  - **de_movingAve:** Differential entropy (DE) of moving average values.
  - **de_lds:** DE of Linear Dynamic Systems (LDS) values.
  - **psd_movingAve:** Power spectral density (PSD) of moving average values.
  - **psd_lds:** PSD of Linear Dynamic Systems (LDS) values.
- **String Representation:** Each channel's features were formatted into a structured string with contextual labels for each frequency band. The structure includes:
  - **Channel Context:** Each string begins with the channel identifier in the format [ChannelX], where X is the channel number.
  - **Feature Groups:** The features were divided into four groups: `de_movingAve`, `de_LDS`, `psd_movingAve`, and `psd_LDS`.
  - **Frequency Bands:** Each feature group includes values for five frequency bands: Delta, Theta, Alpha, Beta, and Gamma.
  - **Formatting:** Features within each group were contextualized with their frequency band name and corresponding values.

The string representation for a single channel was structured as:

```
[ChannelX] de_movingAve:  Delta=value1 | Theta=value2 |
Alpha=value3 | Beta=value4 | Gamma=value5; de_LDS: Delta=value6
| Theta=value7 | Alpha=value8 | Beta=value9 | Gamma=value10;
psd_movingAve:  Delta=value11 | Theta=value12 | Alpha=value13
| Beta=value14 | Gamma=value15; psd_LDS: Delta=value16 |
Theta=value17 | Alpha=value18 | Beta=value19 | Gamma=value20
```

Here, `value1` through `value20` represents the numerical feature values extracted for the corresponding frequency bands and feature groups. The generated string, along with the associated emotion label, was stored as a sample in the new dataset. By the end of this process, a new dataset was created with 2 columns, samples and labels. This format provides explicit context for each numerical feature, enabling LLMs to effectively learn from the data structure.

Train Test Split

To prepare the dataset for model training and evaluation, the data was split into three subsets: training, validation, and testing, with proportions of 80%, 10%, and 10%, respectively. To ensure a diverse distribution of samples across all sets, the splitting process was performed randomly. A random state of 42 was used which ensured the splits were identical. This approach ensures that the training set is sufficiently large for the models to learn effectively, while the validation and test sets provide reliable metrics for evaluating and testing the model performance.

Tokenization

The EEG dataset's contextualized string representations were tokenized using model-specific settings, determined through the tokenization analysis conducted in the dataset section. Each pre-trained model has a different approach for processing tokens. Hence, the tokenization process ensures that the textual data maintains consistency across all samples when feeding into the LLMs. Truncation was enabled

(truncation=True) for handling varying token lengths, ensuring that any sequence exceeding the predefined maximum token length was truncated to fit within the model constraints. Padding was also applied (`padding=True`), aligning all sequences to a uniform length by adding padding tokens where necessary. This step was essential for maintaining a fixed input size across the dataset and optimizing batch processing efficiency during training. The maximum token length was set based on

model-specific constraints:

- **FLAN-T5**: 200 tokens
- **GPT-2**: 185 tokens
- **BERT**: 210 tokens

The tokenization process generates PyTorch tensors as output. This is to ensure compatibility with PyTorch-based deep learning frameworks, allowing for seamless integration into the training pipeline.

5.1.2 Eye Tracking Data

Preprocessing for Structural Alignment with EEG Data

Compared to the EEG data, the eye-tracking data had a simpler representation having fewer dimensions. It was preprocessed to structurally align with the format of EEG data, ensuring consistency across both modalities and streamlining integration and analysis. During preprocessing, the features in the eye data were structured into a tuple format. Each tuple consisted of a 1D array of length 22 and an integer label representing an emotion class. Other steps of preprocessing were as follows:

1. **Handling Missing Values:** A total of 26 samples contained missing values (NaN) in certain features. For these features, the mean and standard deviation of the non-NaN values were calculated. To preserve variability, random values were imputed based on the mean and standard deviation of the corresponding feature.

2. **Dropping Redundant Features:** The Event Statistics feature had duplicate values of already existing data. The 8 columns that represented this feature were removed to ensure variability and consistency in the dataset while avoiding redundancy.

3. **Feature Arrangement:** The remaining features were arranged into a 1D array of size 22, representing the following:

    - 1–12: Pupil diameter (X and Y coordinates).
    - 13–16: Dispersion (X and Y coordinates).
    - 17–18: Fixation duration (in milliseconds).
    - 19–22: Saccades.

4. **Normalization:**

    - Z-score normalization was applied to standardize the feature values and make them comparable and consistent across all samples.
    - This transformation ensured that each feature had a mean of 0 and a standard deviation of 1.

String Conversion of Eye Movement Data

Similar to the EEG data, eye data was transformed into a structured text format to be processed by LLMs. This transformation introduced contextual labels to the features, enabling LLMs to better interpret and utilize the data. Each sample's features were converted into a descriptive string format, incorporating key information about pupil diameter, dispersion, fixation duration, and saccades.

Input Data Structure: The normalized dataset comprised 37,575 tuples, with each tuple consisting of a 1D array of 22 features and an integer label indicating the emotion category.

String Representation: The features were converted into a structured string format that added contextual information for each metric. Each feature was labeled with its corresponding group and index (e.g., `Pupil_1`, `Dispersion_1`, etc.), followed by its normalized value. Since the eye-tracking data has a much simpler representation than the EEG data, each sample could fit in a single row while keeping within the limits of the tokenizers. One of the samples of the eye-tracking dataset after string conversion looks like following:

An example of a row from the eye-tracking dataset after string conversion is as follows:

```
Pupil_1: value1, Pupil_2: value2, Pupil_3: value3, Pupil_4: value4,
Pupil_5: value5, Pupil_6: value6, Pupil_7: value7, Pupil_8: value8,
Pupil_9: value9, Pupil_10: value10, Pupil_11: value11, Pupil_12: value12,
Dispersion_1: value13, Dispersion_2: value14, Dispersion_3: value15,
Dispersion_4: value16, Fixation_1: value17, Fixation_2: value18,
Saccade_1: value19, Saccade_2: value20, Saccade_3: value21, Saccade_4: value22
```

Here value 1 through value 22 represents the numerical values of each feature.

Final Data Structure: The final dataset was expanded to include a descriptive string representation for the features in each sample, paired with its respective emotion label. Unlike the converted EEG dataset, the size of the eye-tracking dataset remained unchanged after the string conversion. By adding meaningful labels and organizing the features, this transformation allowed LLMs to understand the structure and context of the data, facilitating downstream tasks like emotion classification.

Train Test Split

The eye movement data was split into training, validation, and testing subsets in the same manner as the EEG data, with proportions of 80%, 10%, and 10%, respectively. The random splitting ensured class balance across subsets, and a fixed random state of 42 was used to maintain reproducibility.

Tokenization

The tokenization process for the eye movement data followed the same approach as the EEG data, ensuring consistency in model input formatting. Truncation (`truncation=True`) was applied to remove excess tokens beyond the model-specific limits, while padding (`padding=True`) ensured all sequences had a uniform length. The maximum token length for each model was determined based on tokenization analysis:

- **FLAN-T5**: 240 tokens
- **GPT-2**: 250 tokens
- **BERT**: 235 tokens

Similar to the EEG data, the tokenized outputs were converted into PyTorch tensors, allowing seamless integration with deep learning frameworks. These settings ensured that the tokenized eye movement data remained structured and compatible with all models.

5.2 Fine-Tuning Setup for LLMs

For this experiment, pre-trained models were chosen and each model was trained two times separately for each type of dataset. Fine-tuning setups for each model were mostly similar with some minor modifications.

5.2.1 FLAN-T5

In our implementation, the input strings and the corresponding labels were tokenized and passed as input tokens to the Flan-T5 model. The models prebuilt tokenizer was utilized, adapting the data into its expected format:

- **Input Sequences:** Numerical data from our dataset was converted into text format and tokenized to align with the model's input requirements.
- **Attention Masks:** Attention masks were computed to differentiate meaningful tokens from padding tokens, ensuring efficient processing of sequences with variable lengths.
- **Labels:** As this is an encoder-decoder model the labels were also tokenized to ensure compatibility with the model's text-to-text framework.

- **Dataloader:** The training dataset was loaded using a DataLoader with a batch size of 16, enabling mini-batch gradient descent. Data shuffling was applied to ensure randomness in each epoch. The validation dataset was loaded into a separate DataLoader with the same batch size.
- **Optimizer:** The model was optimized using the AdamW optimizer, which is known for handling sparse gradients and working well with transformer models. A constant learning rate of 1e-4 was used.
- **Epochs:** The model was trained for 10 epochs after which the train and validation curves converge.
- **Checkpointing:** Model checkpoints were saved at the end of each epoch, which enables the restart of training and fine-tuning in the event that the process is interrupted.

5.2.2 GPT-2

GPT-2 follows a **decoder-only** transformer architecture, where input sequences are processed autoregressively.

- **Input Sequences**: The input text was tokenized using GPT-2's prebuilt tokenizer and passed into the model.
- **Attention Masks**: Applied to manage variable-length sequences, ensuring proper token alignment and processing.
- **Logits and Prediction**: The model outputs logits, representing unnormalized probabilities for each token in the vocabulary. In our classification task, logits were mapped to the four possible emotion categories, with the final predicted label being the class with the highest probability.
- **Dataloader**: The training dataset was used for optimizing model parameters, while the validation dataset monitored generalization across epochs. Both datasets were loaded using PyTorch `DataLoader` with a batch size of 16.
- **Optimizer**: Fine-tuning was performed using the AdamW optimizer, with a constant learning rate of $2e^{-5}$, incorporating weight decay for stability.
- **Early Stopping**: Training was terminated if the validation loss did not improve for five consecutive epochs, ensuring optimal model performance.
- **Checkpointing**: Model checkpoints were saved at the end of each epoch using `torch.save()`, preserving the model state, optimizer state, and training/validation losses. This allowed training to resume seamlessly in case of interruptions.

5.2.3 BERT

BERT employs an **encoder-only** transformer architecture, designed for bidirectional contextual understanding.

- **Input Sequences**: A special `[CLS]` token was prepended to each input sequence, serving as a global representation for classification.
- **Tokenization**: The input text was tokenized using BERT's tokenizer, ensuring compatibility with the model's structure.
- **Fine-Tuning**: The pre-trained **BERT-base-uncased** model was fine-tuned on labeled datasets by adding a classification head. The `[CLS]` token embedding was used as input to the classification layer, which outputted logits corresponding to the emotion classes.
- **Dataloader**: Similar to GPT-2 and Flan-T5, the training and validation datasets were loaded using PyTorch `DataLoader` with a batch size of 16.
- **Optimizer**: Fine-tuning was performed using the AdamW optimizer with a constant learning rate of $2e^{-5}$.
- **Loss Function**: The model internally computed cross-entropy loss for classification tasks.
- **Epochs**: A total of 9 epochs were run, after which the training and validation loss curves showed convergence.
- **Checkpointing**: Model checkpoints were saved at the end of each epoch, preserving model states for continuity in training.

5.3 Ensemble Learning

Two different ensemble learning approaches were explored in this paper for a better understanding of how we can integrate multi-modal data predicted by LLMs. The implementation of both techniques was different.

5.3.1 Majority Voting

The Majority Voting ensemble method was employed to predict emotions by combining both EEG and eye-tracking data, integrating predictions from all three LLMs.

In the first stage, the training and validation sets of both EEG and eye-tracking data were used to fine-tune each of the three pre-trained models separately. After training, the fine-tuned models were saved individually for each dataset type, resulting in six separate models: Flan-T5 EEG, Flan-T5 Eye, GPT-2 EEG, GPT-2 Eye, BERT EEG, and BERT Eye.

In the second stage, the test sets of both datasets (EEG and eye-tracking) were used to generate predictions using their respective fine-tuned models. Each model produced independent predictions based on its assigned modality.

A key challenge in this process was that the labels of the EEG and eye-tracking test sets were not aligned, meaning the first sample of the EEG test dataset did not correspond directly to the first sample of the eye-tracking test dataset. To resolve this and to ensure accurate evaluation, the labels from both test sets were combined using Majority Voting. The labels of both datasets were stacked using NumPy's np.vstack() function, and the mode function was applied to derive a final combined label.

Additionally, the EEG dataset was structured channel-wise, whereas the eye-tracking dataset was directly converted into a text format. Since each EEG sample contained 62 channels, the eye-tracking dataset had 62 times fewer samples than the EEG dataset. To align these datasets for ensemble prediction, the eye-tracking predictions were resampled using a granularity of 62, ensuring proper correspondence between the two modalities.

Finally, the predictions from all three models (Flan-T5, GPT-2, and BERT) on both EEG and eye-tracking datasets were aggregated, and the final decision was determined using Majority Voting. The accuracy of this approach was measured using the combined labels.

5.3.2 Stacked Ensemble

The implementation of the Stacked Ensemble technique is initiated by getting the predictions from the fine tuned models and then resampling of the eye-tracking data predictions. Both of these steps are similar to the Majority Voting approach explained previously.

The next step for this technique was to create **meta-features** from the model outputs. This was achieved by combining predictions from multiple LLMs into a single structured matrix. These arrays were stacked together using NumPy's `np.column_stack()` function, ensuring that all models contributed equally to the meta-model's decision-making process.

Once the meta-feature matrix was constructed, a LightGBM classifier was trained as the meta-model, responsible for making the final classification decision.
Additionally, the meta-feature matrix was split into training and testing sets, ensuring that the meta-model learned from a diverse set of examples. LightGBM was chosen due to its ability to handle heterogeneous features efficiently, making it well-suited for combining predictions from multiple models.

Chapter 6

Results and Analysis

6.1 Performance Evaluation Metrics

To evaluate the performance of each model on different data types, a set of metrics was used across all tests. These metrics are as follows:

Accuracy

Accuracy measures the proportion of correctly classified samples to the total number of samples. It provides a general overview of the model's performance.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

Where:

- **TP:** True Positives
- **TN:** True Negatives
- **FP:** False Positives
- **FN:** False Negatives

Precision

Precision evaluates the accuracy of positive predictions by measuring the proportion of true positive predictions among all predicted positives. It is particularly useful in scenarios with imbalanced datasets.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Recall

Recall (or Sensitivity) measures the model's ability to identify all relevant positive instances. It is critical when minimizing false negatives is important.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

F1 Score

The F1 Score is the harmonic mean of Precision and Recall. It balances these two metrics and is useful when there is an uneven class distribution.

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

Cohen's Kappa

Cohen's Kappa is a statistical measure that evaluates the agreement between two raters (or a model and the ground truth) for categorical classification tasks. Unlike simple accuracy, Cohen's Kappa accounts for the possibility of agreement occurring by chance. This makes it a more robust metric, especially when dealing with imbalanced datasets where high accuracy can be misleading. The formula for Cohen's Kappa is:

$$\kappa = \frac{p_o - p_e}{1 - p_e}$$

Where:

- p_o (Observed Agreement): The proportion of instances where the raters agree (i.e., the proportion of correct predictions in the context of a model).
- p_e (Expected Agreement): The proportion of instances where agreement is expected by chance, calculated based on the marginal probabilities of each class.

Loss Function

The loss function evaluates how well the model predicts the target labels. For this task, categorical cross-entropy was used, which is defined as:

$$\text{Cross-Entropy Loss} = -\frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{C} y_{ij} \cdot \log(\hat{y}_{ij})$$

Where:

- N: Total number of samples
- C: Total number of classes
- y_{ij}: True label indicator (1 if class j is the correct class for sample i, otherwise 0)
- $\hat{y}_{ij}$: Predicted probability for class j of sample i

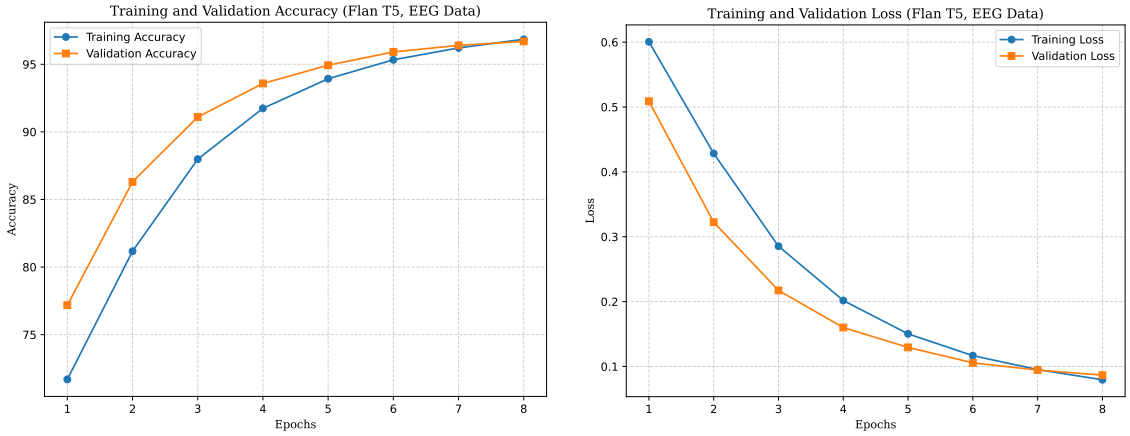
6.2 Experimental Result Analysis

To evaluate the performance of each model on each data type a test set was created before training with 10% of the data. This test set was kept completely separate during training and validation to ensure an unbiased assessment of the models' generalization capabilities. The models were trained on 80% of the dataset, with 10% reserved for validation to fine-tune the hyperparameters and prevent overfitting. All three models were fine-tuned and tested on both EEG and eye-tracking data.

6.2.1 Model performance on EEG data

FLAN T5

During the fine-tuning stage, Flan T5 showed the best results in terms or training and validation accuracy. Figure 6.1(a) illustrates the accuracy and loss curves during the fine-tuning process, highlighting the model's consistent improvement across epochs. The training accuracy gradually increased, converging with the validation accuracy at approximately 96% by the final epoch, as shown in the first graph. Similarly, the loss curves in Figure 6.1(b) demonstrate a steady decline for both training and validation, indicating effective learning and minimal overfitting.



(a) Training and Validation Accuracy (b) Training and Validation Loss

Figure 6.1: Training and Validation Performance of Flan T5 on EEG Data

The performance of Flan-T5 was further evaluated on the test set using the metrics, shown in Figure 6.2. Flan-T5 maintained high scores across all evaluation metrics:

Table 6.1: Performance Metrics for Flan-T5

Metric	**Value**
Accuracy	0.8681
Precision (Weighted)	0.8682
Recall (Weighted)	0.8681
F1 Score (Weighted)	0.8679
Matthews Correlation Coefficient	0.8235
Cohen's Kappa	0.8234

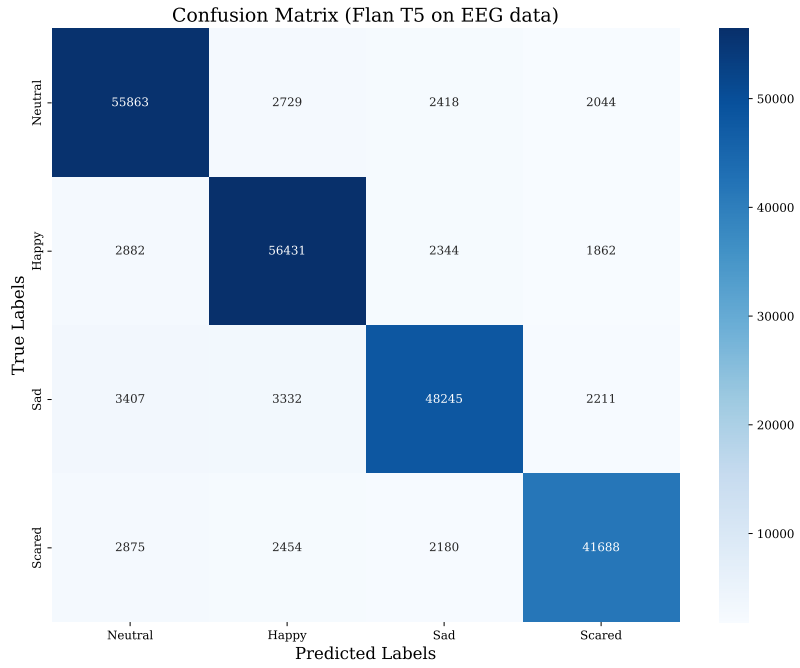
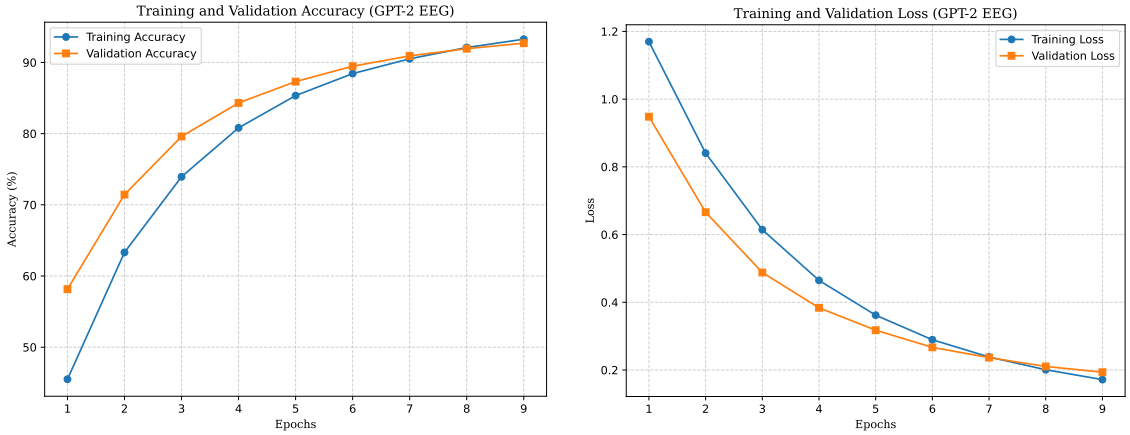


Figure 6.2: Confusion Matrix of FLAN T5 on EEG Data

These results validate Flan-T5's ability to effectively process the string-based EEG data, capturing meaningful contextual patterns for emotion recognition.

GPT-2

For fine-tuning the GPT2 model, the training and validation accuracy curves in Figure 6.3 indicate rapid convergence, with validation accuracy surpassing 92% by the final epoch. The loss curves also show a smooth and consistent reduction for both training and validation, reinforcing the model's stability.



(a) Training and Validation Accuracy (b) Training and Validation Loss

Figure 6.3: Training and Validation Performance of GPT-2 on EEG Data

GPT-2 demonstrated superior performance compared to Flan-T5, particularly in terms of test accuracy. Upon evaluating its performance on the test set we can see an improvement across all the performance metrics highlighting GPT-2's strong capacity for generalizing to unseen data, making it the best-performing standalone LLM on the EEG dataset.

Table 6.2: Performance Metrics for GPT-2

Metric	**Value**
Accuracy	0.9269
Precision (Weighted)	0.9269
Recall (Weighted)	0.9269
F1 Score (Weighted)	0.9269
Matthews Correlation Coefficient	0.8998
Cohen's Kappa	0.8997

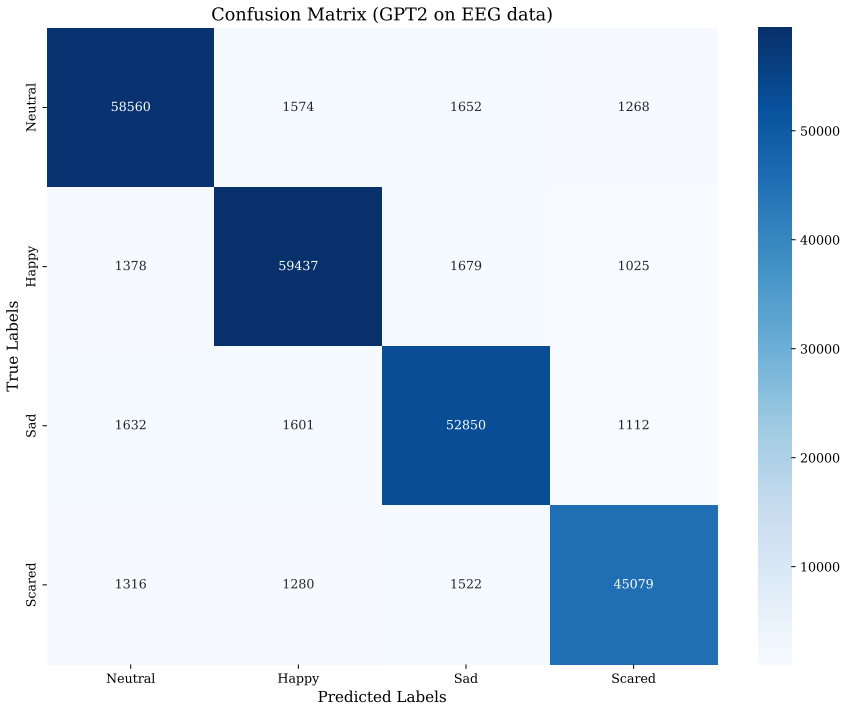
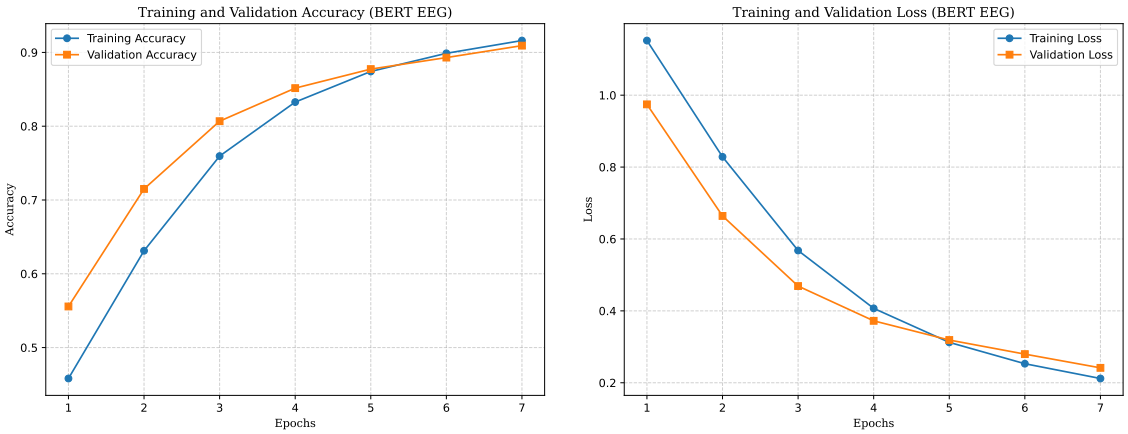


Figure 6.4: Confusion Matrix of GPT-2 on EEG Data

BERT

BERT achieved competitive results, closely aligning with GPT-2 in terms of accuracy. As seen in Figure 6.4, BERT reached a training and validation accuracy of approximately 91% and 90% respectively by the final epoch. The corresponding loss curves also show consistent improvement, indicating effective learning.



(a) Training and Validation Accuracy (b) Training and Validation Loss

Figure 6.5: Training and Validation Performance of BERT on EEG Data

The BERT model was also evaluated further using the test set and the following table shows the results of each metric.

Table 6.3: Performance Metrics for BERT

Metric	**Value**
Accuracy	0.9098
Precision (Weighted)	0.9100
Recall (Weighted)	0.9098
F1 Score (Weighted)	0.9098
Matthews Correlation Coefficient	0.8824
Cohen's Kappa	0.8823

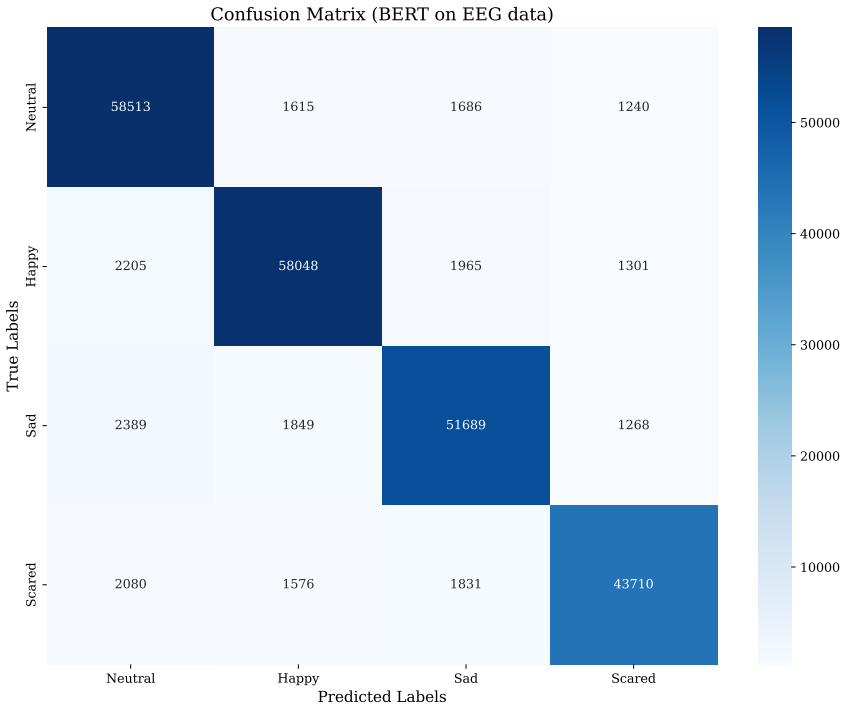
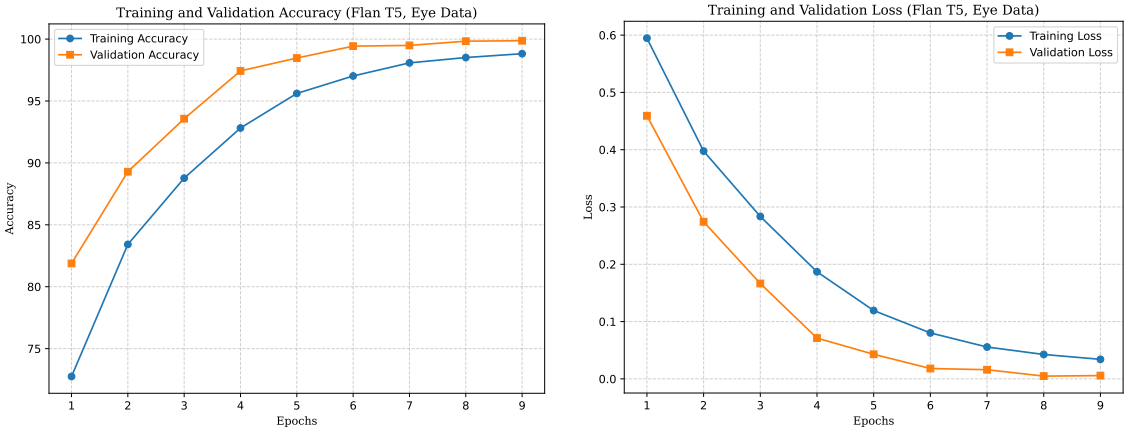


Figure 6.6: Confusion Matrix of BERT on EEG Data

6.2.2 Model performance on Eye-Tracking Data

FLAN T5

Flan-T5 demonstrated exceptional performance on the eye-tracking dataset during both fine-tuning and testing. As shown in Figures 6.7(a) and 6.7(b), the model achieved nearly perfect convergence in both accuracy and loss curves. The training and validation accuracy approached 99.6%, while the validation loss was reduced to near-zero levels, highlighting the effectiveness of the model in learning from the structured string-based eye data.



(a) Training and Validation Accuracy (b) Training and Validation Loss

Figure 6.7: Training and Validation Performance of FLAN T5 on Eye Tracking Data

The following table shows the results of the performance metrics on eye-tracking test dataset.

Table 6.4: Performance Metrics of FLAN T5 on Eye-Tracking Test Dataset

Metric	**Value (%)**
Accuracy	99.63
Precision	99.63
Recall	99.63
F1-Score	99.63
Matthews Correlation Coefficient	99.50
Cohen's Kappa	99.50

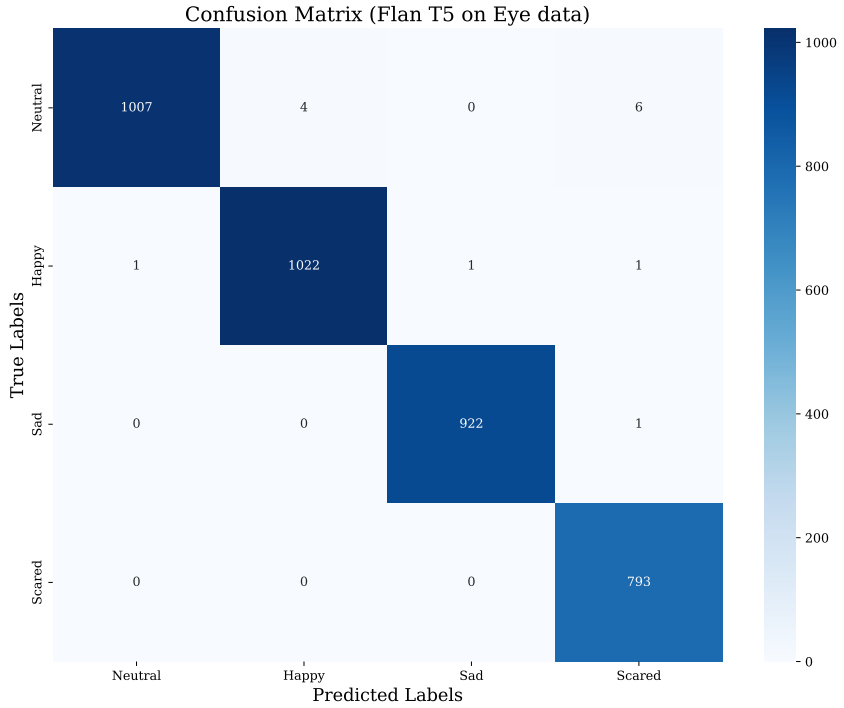
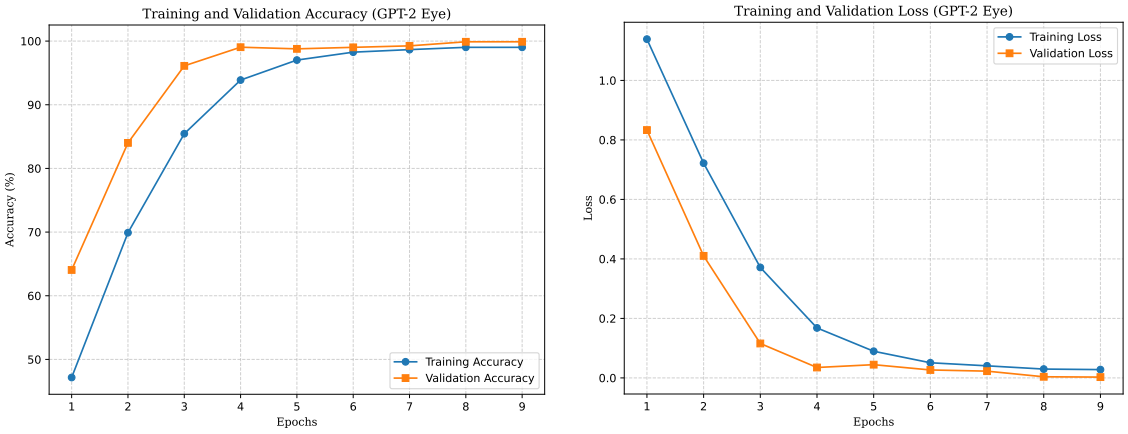


Figure 6.8: Confusion Matrix of FLAN T5 on Eye Tracking Data

These results demonstrate that Flan-T5 can effectively capture the fine-grained details in eye-tracking data, making it the best-performing model on this dataset.

GPT-2

GPT-2 also showed strong performance on the eye-tracking dataset, achieving a high level of accuracy and stable convergence during training. As seen in Figures 6.9(a) and 6.9(b), the training and validation accuracy reached 97.6%, with validation loss steadily decreasing to a low value, indicating effective learning with minimal overfitting.



(a) Training and Validation Accuracy (b) Training and Validation Loss

Figure 6.9: Training and Validation Performance of GPT-2 on Eye Tracking Data

The following table shows the results of the performance metrics on eye-tracking test dataset.

Table 6.5: Performance Metrics of GPT-2 on Eye-Tracking Test Dataset

Metric	**Value (%)**
Accuracy	97.63
Precision	97.63
Recall	97.63
F1-Score	97.63
Balanced Accuracy	97.58
Matthews Correlation Coefficient	96.83
Cohen's Kappa	96.83

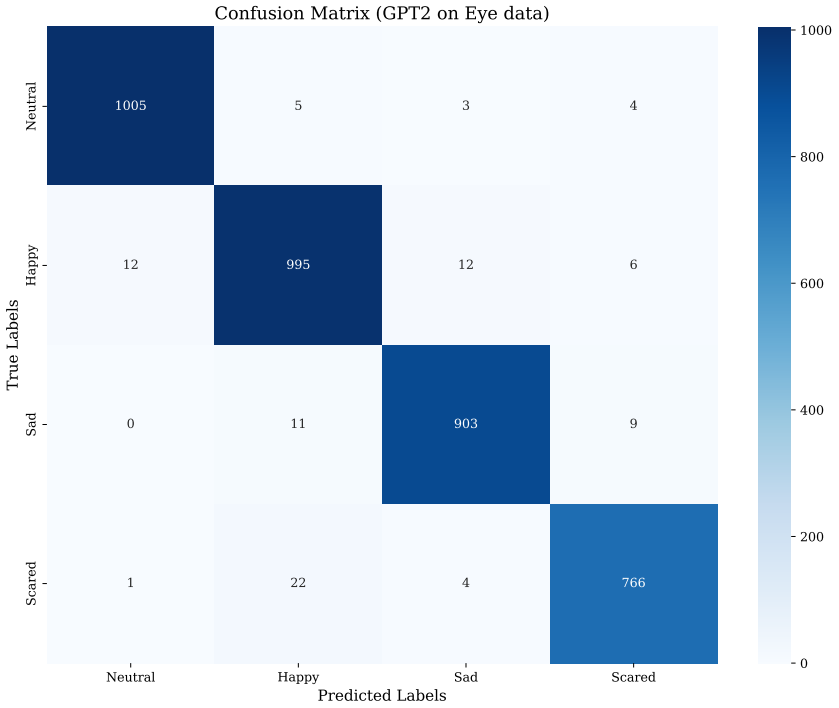
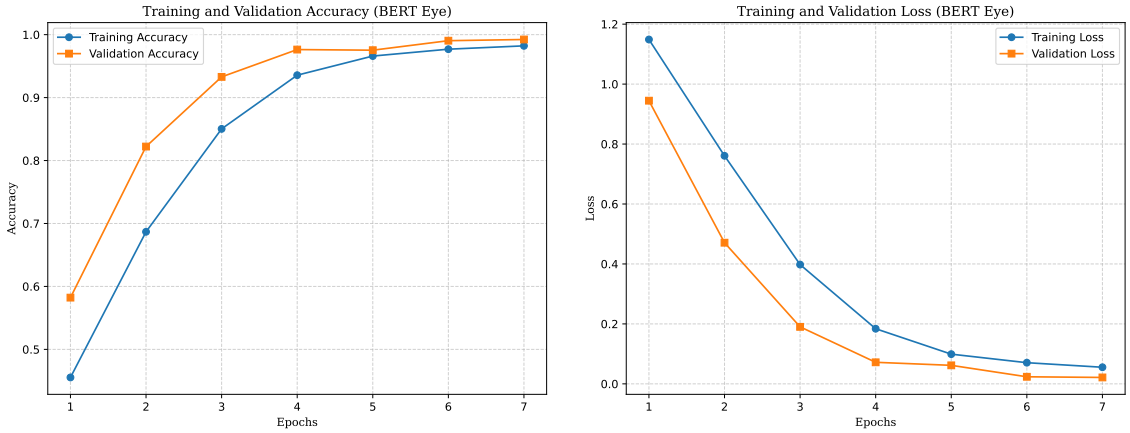


Figure 6.10: Confusion Matrix of GPT-2 on Eye Tracking Data

While slightly behind Flan-T5, GPT-2 demonstrated robust and reliable performance, confirming its utility for processing eye-tracking data in emotion recognition tasks.

BERT

BERT achieved highly competitive results, surpassing GPT-2 but slightly trailing Flan-T5 in terms of accuracy. Figures 6.11(a) and 6.11(b) show consistent improvements in training and validation accuracy, with final values reaching 99.04%. The corresponding loss curves indicate smooth and effective learning with minimal overfitting.



(a) Training and Validation Accuracy (b) Training and Validation Loss

Figure 6.11: Training and Validation Performance of BERT on Eye Tracking Data

The following table shows the results of the performance metrics on eye-tracking test dataset.

Table 6.6: Performance Metrics of BERT on Eye-Tracking Test Dataset

Metric	**Value (%)**
Accuracy	99.04
Precision	99.05
Recall	99.04
F1-Score	99.04
Balanced Accuracy	99.02
Matthews Correlation Coefficient	98.72
Cohen's Kappa	98.72

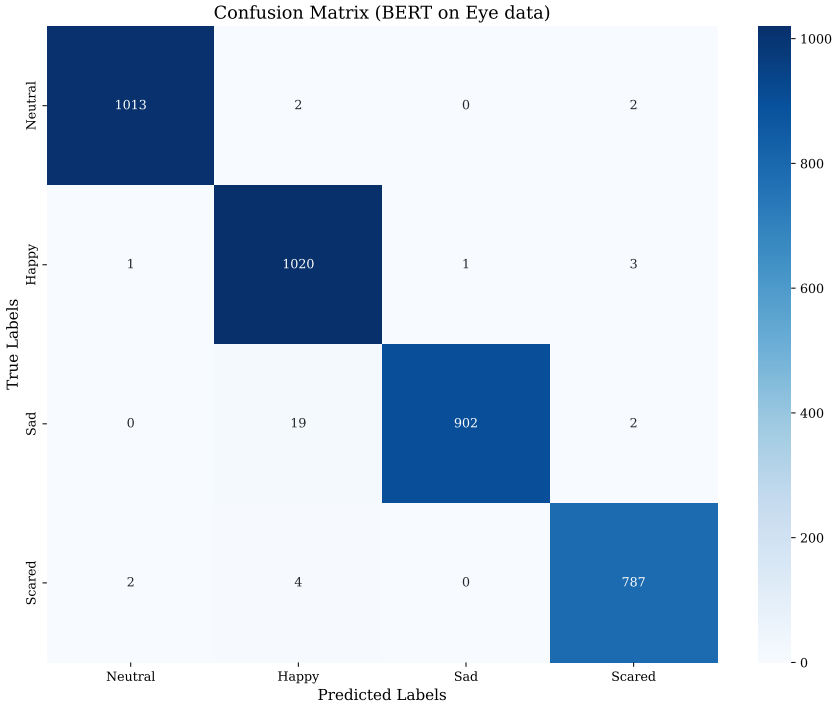


Figure 6.12: Confusion Matrix of BERT on Eye Tracking Data

The performance comparison between EEG data and eye data across all models highlights the strengths of both modalities and the effectiveness of the LLMs. For EEG data, GPT-2 achieved the highest accuracy at 92%, followed by BERT at 90%, and Flan-T5 at 86%. These results indicate that GPT-2 is best suited for processing the string-based EEG dataset, leveraging its capability to handle contextual information effectively. For eye data, Flan-T5 demonstrated exceptional

performance with an accuracy of 99.63%, outperforming both BERT (99.04%) and GPT-2 (97.63%). The near-perfect accuracy of Flan-T5 suggests its superior ability to process fine-grained information in eye-tracking data, making it the most reliable model for this modality. In general, the models performed better on eye data than

EEG data across all metrics, likely due to the structural simplicity and richness of eye-tracking features compared to the complex and high-dimensional EEG signals. This comparison sets the foundation for exploring multi-modal ensemble methods to combine the strengths of both data types.

6.3 Results of Ensembling Learning Technique

6.3.1 Majority Voting

The first ensemble technique involved majority voting based on all three model predictios on both data types. This approach combined predictions across the dataset and achieved the following results on the validation set:

Table 6.7: Performance Metrics of Majority Voting Using All 3 models

Metric	Value
Accuracy	91.93
Precision (Weighted)	93.46
Recall (Weighted)	91.93
F1 Score (Weighted)	92.36
Cohen's Kappa	0.8784

The confusion matrix (Figure 6.13) reveals that the majority voting ensemble performed well across all classes, with a notable number of correct predictions in each class. However, some misclassifications persisted, particularly in classifying borderline cases between adjacent classes. While effective, the results indicate that with ensemble voting, there are limitations in maximizing overall performance.

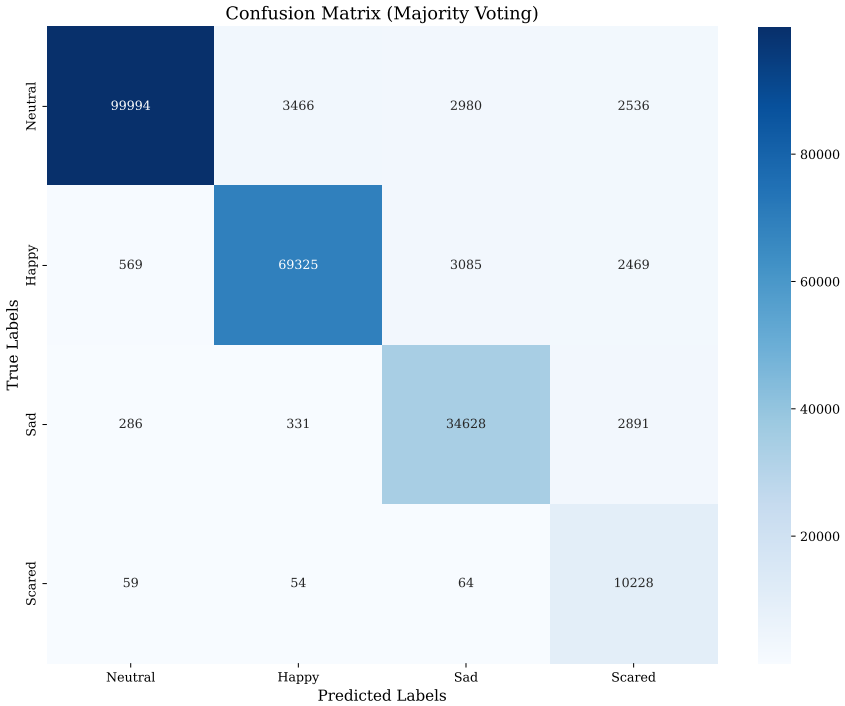


Figure 6.13: Confusion Matrix of Majority Voting

6.3.2 Stacked Ensembling Using All Predictions

The second approach, stacked ensembling, utilized predictions from all three LLMs; Flan-T5, GPT-2, and BERT, as input features for a LightGBM meta-model. This method demonstrated a significant improvement over majority voting, achieving the following metrics:

Table 6.8: Performance Metrics of Stacked Ensembling Using All Predictions

Metric	Value
Accuracy	96.29%
Precision	96.29%
Recall	96.29%
F1 Score	96.29%
Cohen's Kappa	94.27%

The confusion matrix (Figure 6.14) highlights a reduction in misclassifications across all classes, confirming the effectiveness of combining predictions from multiple models. The stacked ensemble leveraged the complementary strengths of the three LLMs to create a more robust decision space, significantly enhancing classification accuracy.

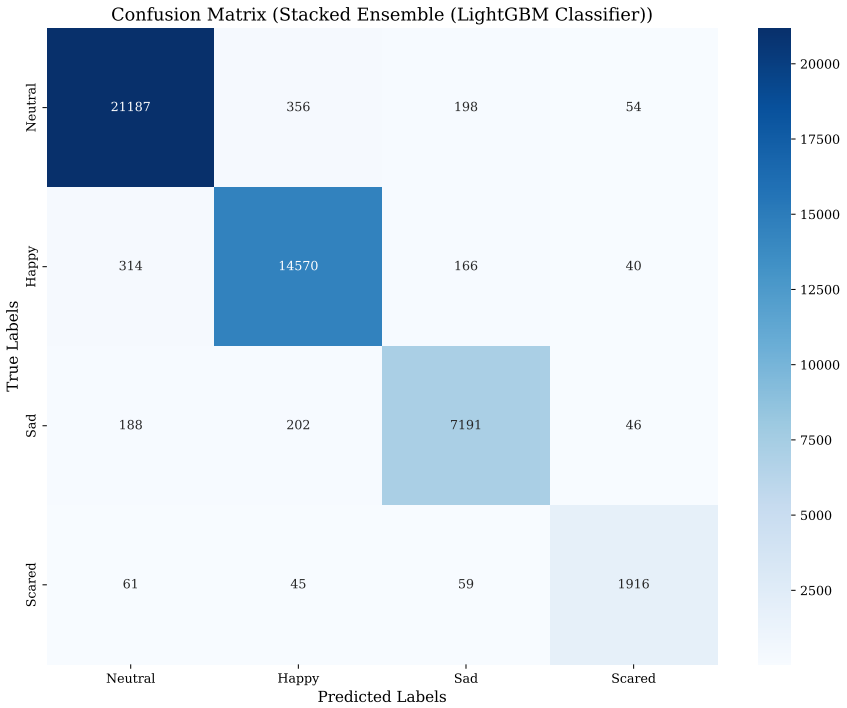


Figure 6.14: Confusion Matrix of Stacked Ensembling Using All Predictions

The predicted probability distribution for each class, as shown in Figure 6.15 provides further insights into the ensemble methods' performance. The stacked ensemble demonstrates well-separated probability densities for all classes, indicating high confidence in the predictions. The baseline probability (red dashed line) is far exceeded, especially for Class 0 and Class 1, which show distinct peaks at higher predicted probabilities. This clear separation reflects the model's ability to make precise and reliable predictions across all classes.

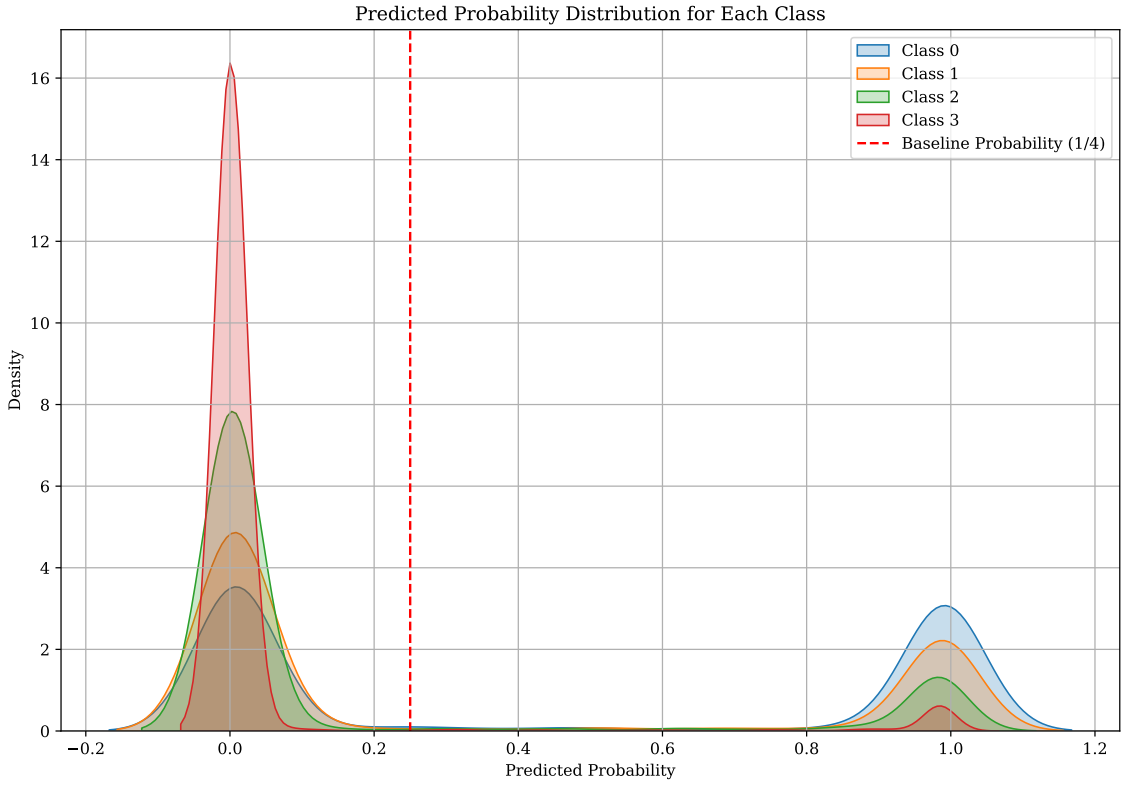


Figure 6.15: Prediction Probability Distribution of Stacked Ensembling Method

While majority voting improved prediction consistency, it was limited to a single model's perspective. In contrast, stacked ensembling with LightGBM capitalized on the diverse outputs of all three LLMs, achieving significantly better performance across all metrics. The probability distribution further highlights the robustness of the stacked ensemble approach, demonstrating its capacity to produce highly confident and accurate predictions.

6.4 Ablation Study on Ensemble Methods

An **ablation study** was conducted to evaluate the contribution of different models within the stacked ensemble approach. This analysis aimed to determine the impact of individual models—Flan-T5, BERT, and GPT-2—on the overall classification performance when used independently in the stacked ensemble framework.
Table 6.9 shows the performance metrics of using single model predicitons.

Model	Accuracy	F1 Score	Cohen's Kappa
Flan-T5	0.9238	0.9237	0.8820
BERT	0.9467	0.9466	0.9175
GPT-2	0.9489	0.9489	0.9211

Table 6.9: Performance Metrics of Individual Models in the Stacked Ensemble

From these results, it is evident that the stacked ensemble benefits significantly from the inclusion of multiple models. The BERT and GPT-2 models outperform FLAN-T5 individually, suggesting that FLAN-T5 may not contribute as effectively in isolation. However, the full stacked ensemble setup with the prediction of all models, yield even better performance which highlight the complementary strengths of different transformer architectures. Figure 6.16 shows a comparison between the accuracies of single model peformance against all model performance.

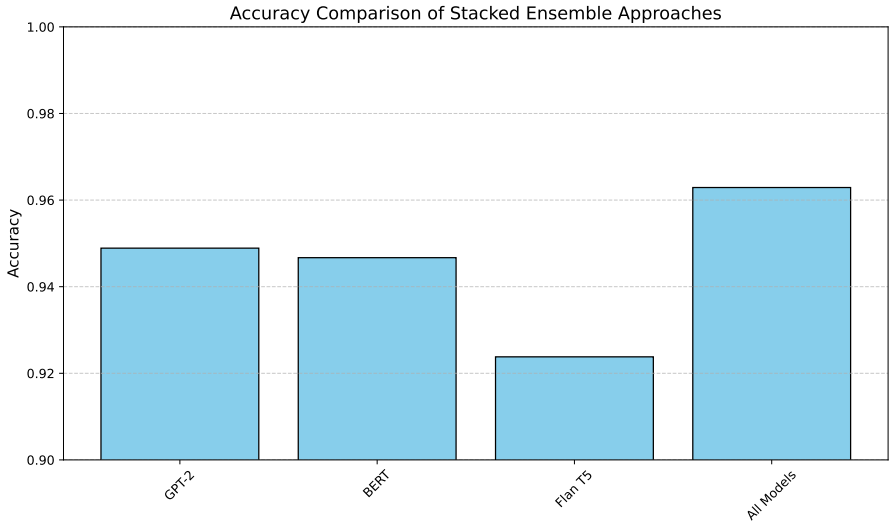


Figure 6.16: Accuracy Comparison of Stacked Ensembling Approaches

6.5 Discussion

This study demonstrated the effectiveness of Large Language Models (LLMs) for emotion recognition using both EEG and eye-tracking data. By transforming high-dimensional physiological signals into contextual string representations, we enabled LLMs to process numerical data in a way that has not been widely explored in affective computing. The experimental results highlight the strength of fine-tuned LLMs in classifying emotional states, with GPT-2 achieving the highest accuracy for EEG data (92%) and Flan-T5 excelling in eye-tracking data (99.63%). Furthermore, the implementation of a stacked ensemble method with LightGBM led to a significant improvement, achieving an overall classification accuracy of 96.29% which proves the potential of integrating multi-modal data through ensemble learning.

6.5.1 Performance of LLMs on EEG and Eye-Tracking Data

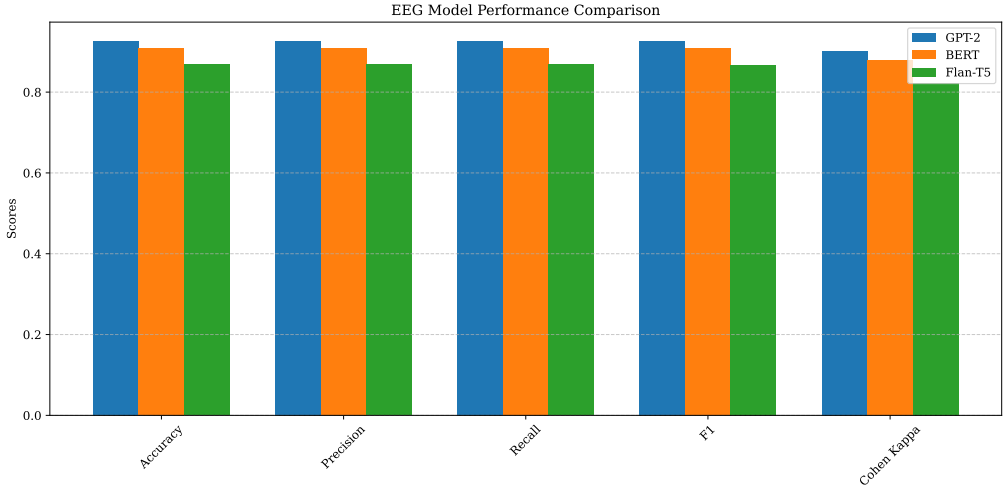


Figure 6.17: Performance Evaluation on EEG Data

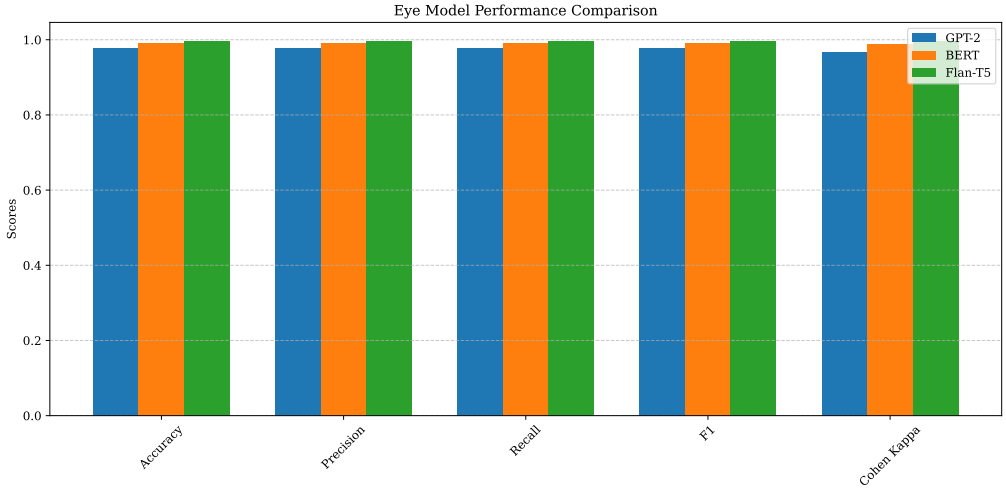


Figure 6.18: Accuracy Comparison on Eye Data

The final results for each LLM across EEG and eye-tracking data indicate a key difference in the classification performance of LLMs. GPT-2 and BERT achieved 92% and 90% accuracy on EEG data demonstrating strong generalization capabilities. On the other hand, their performance on eye-tracking data was slightly lower than that of Flan-T5. This suggests that decoder-based architectures like GPT-2 are better suited for handling the complex contextual dependencies present in EEG data, whereas encoder-decoder models like Flan-T5 may be more effective at capturing fine-grained variations in eye movement patterns. Graph 6.17 and 6.18 show a comparison of performance metrics for each model on EEG and eye-tracking data respectfully. The performance gap between EEG and eye data models also highlights the inherent structural differences between the two modalities, with EEG signals being more complex and multidimensional compared to the relatively structured and sequential nature of eye-tracking features.

6.5.2 Advantages of Data Contextualization for LLMs

One of the primary challenges in using LLMs for physiological data is the requirement for an appropriate formatting. Unlike CNNs or RNNs, which operate on raw numerical data, LLMs require sequential tokenized input. Our study addressed this by converting EEG and eye-tracking data into structured text, making it compatible with transformer architectures. The results suggest that this approach successfully retained essential signal characteristics, allowing LLMs to learn meaningful patterns from the data. Prior research in multi-modal learning has explored textual representations for structured data, but the application of such techniques to physiological signals remains underexplored. The success of our approach suggests that text-based encoding can serve as a viable method for leveraging LLMs in non-traditional domains such as affective computing.

6.5.3 Effectiveness of Multi-Modal Integration

The integration of EEG and eye-tracking data using ensemble learning significantly improved classification performance, achieving a final accuracy of 96.29%. The majority voting approach resulted in an accuracy of 92.32%, demonstrating that combining predictions from multiple samples improved consistency. The stacked ensemble method, further enhanced performance by leveraging the strengths of different architectures. The meta-learning approach with LightGBM effectively captured complementary features from both EEG and eye-tracking modalities, reducing misclassifications and improving robustness. The ablation study confirmed that the stacked ensemble benefited significantly from multiple models, with BERT and GPT-2 making substantial contributions to overall accuracy.

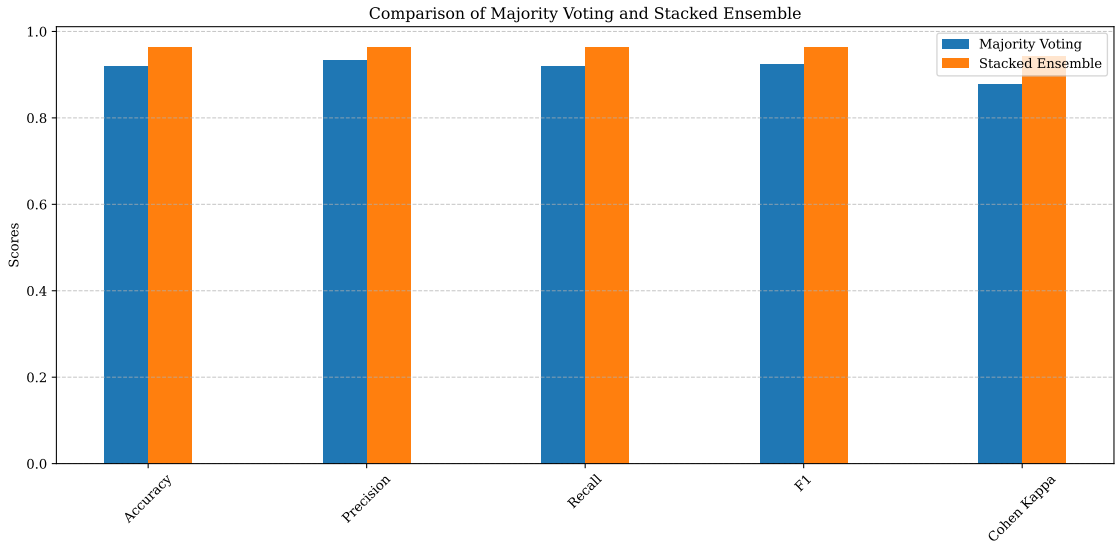


Figure 6.19: Comparison between Majority Voting and Stacked Ensemble Approach

Chapter 7

Conclusion

In conclusion, our research has made significant strides in emotion recognition by utilizing numerical EEG and eye-tracking data. Our primary objective was to explore the potential of large language models (LLMs) in classifying emotions from signal data by fine-tuning them on preprocessed datasets. This approach has provided novel insights into how LLMs, traditionally used for natural language tasks, can be repurposed for signal processing through innovative data transformation techniques. The results of our study demonstrate that fine-tuned LLMs, such as GPT-2 and FLAN-T5, exhibit better performance in emotion classification when numerical data is converted into text format. Moreover, our methodology is a valuable tool for emotion researchers by automating the classification process. By transforming signal data into textual representations, we have simplified preprocessing and unlocked new possibilities for integrating LLMs into signal-processing workflows. This approach opens up ways for future discoveries of complex patterns in EEG and eye-tracking data, potentially leading to a deeper understanding of the underlying mechanisms of emotion. The experimental results showed that GPT-2 performed best on EEG data (92%), while Flan-T5 excelled in eye-tracking classification (99.63%). Additionally, stacked ensemble learning further improved performance, achieving a final classification accuracy of 96.29% This study enhances emotion recognition capabilities and broadens the application of LLMs, advancing knowledge in machine learning and emotion research on a broader scientific scale.

7.1 Challenges

Throughout our research, we encountered several challenges while conducting our research. One of the main challenges was to convert and format the numerical EEG and eye-tracking data into text format to ensure compatibility with the LLMs while preserving the integrity of the original data. While adapting to LLMs, it was quite challenging for us to handle signal data posed challenges in terms of tokenization strategies, loss functions, and sequence modeling. Another challenge that we faced was fine-tuning large language models like GPT-2 and FLAN-T5 on large datasets demanded significant computational resources, particularly in terms of GPU memory and training time. Moreover, handling imbalanced emotion labels in the dataset likely required techniques like data augmentation or class-weighted loss functions to ensure fair model performance across all categories. Moreover, ensuring that the transformed text-based signal data fit within the token length limits of LLMs, without losing critical information, was an ongoing challenge. Furthermore, Combining EEG and eye-tracking data in a meaningful way required careful design and experimentation.

7.2 Future Work

Future research will focus on Knowledge Distillation (KD) to develop a specialized EEG-processing LLM, where a student model learns from multiple fine-tuned LLMs. Additionally, we plan to apply this processing pipeline to other datasets like DEAP to assess generalizability. Another direction involves combining BERT and GPT-2 into a custom encoder-decoder LLM, leveraging BERT's bidirectional encoding with GPT-2's generative strengths. Lastly, we aim to design a custom attention layer optimized for numerical token processing, improving how LLMs interpret EEG and physiological data. These advancements will enhance LLMs' effectiveness in biomedical signal processing.

Bibliography

[1] W. Liu, W.-L. Zheng, and B.-L. Lu, "Multimodal emotion recognition using multimodal deep learning," 2016. eprint: 1602.08225 (cs.HC).

[2] R. Sennrich, B. Haddow, and A. Birch, *Neural machine translation of rare words with subword units*, 2016. arXiv: 1508.07909 [cs.CL]. [Online]. Available: https://arxiv.org/abs/1508.07909.

[3] Y. Wu et al., "Google's neural machine translation system: Bridging the gap between human and machine translation," *CoRR*, vol. abs/1609.08144, 2016. arXiv: 1609.08144. [Online]. Available: http://arxiv.org/abs/1609.08144.

[4] A. Vaswani et al., "Attention is all you need," *CoRR*, vol. abs/1706.03762, 2017. arXiv: 1706.03762. [Online]. Available: http://arxiv.org/abs/1706.03762.

[5] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," *CoRR*, vol. abs/1810.04805, 2018. arXiv: 1810.04805. [Online]. Available: http://arxiv.org/abs/1810.04805.

[6] T. Kudo and J. Richardson, *Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing*, 2018. arXiv: 1808.06226 [cs.CL]. [Online]. Available: https://arxiv.org/abs/1808.06226.

[7] W. Zheng, W. Liu, Y. Lu, B. Lu, and A. Cichocki, "Emotionmeter: A multimodal framework for recognizing human emotions," *IEEE Transactions on Cybernetics*, pp. 1–13, 2018, ISSN: 2168-2267. DOI: 10.1109/TCYB.2018.2797176.

[8] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," *OpenAI*, 2019, Accessed: 2024-11-15. [Online]. Available: https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf.

[9] C. Raffel et al., "Exploring the limits of transfer learning with a unified text-to-text transformer," *CoRR*, vol. abs/1910.10683, 2019. arXiv: 1910.10683. [Online]. Available: http://arxiv.org/abs/1910.10683.

[10] S. Hwang, K. Hong, G. Son, and H. Byun, "Learning CNN features from DE features for EEG-based emotion recognition," en, *Pattern Anal. Appl.*, vol. 23, no. 3, pp. 1323–1335, Aug. 2020.

[11] J. Li, S. Qiu, Y.-Y. Shen, C.-L. Liu, and H. He, "Multisource transfer learning for cross-subject EEG emotion recognition," en, *IEEE Trans. Cybern.*, vol. 50, no. 7, pp. 3281–3293, Jul. 2020.

[12] B. Nakisa, M. N. Rastgoo, A. Rakotonirainy, F. Maire, and V. Chandran, "Automatic emotion recognition using temporal multimodal deep learning," *IEEE Access*, vol. 8, pp. 225 463–225 474, 2020. DOI: 10.1109/ACCESS.2020.3027026.

[13] T. Song, W. Zheng, P. Song, and Z. Cui, "EEG emotion recognition using dynamical graph convolutional neural networks," *IEEE Trans. Affect. Comput.*, vol. 11, no. 3, pp. 532–541, Jul. 2020.

[14] Y. Zhang et al., "An investigation of deep learning models for EEG-based emotion recognition," en, *Front. Neurosci.*, vol. 14, p. 622 759, Dec. 2020.

[15] I. S. Ahmad et al., "Deep learning based on CNN for emotion recognition using EEG signal," en, *WSEAS Trans. Signal Process.*, vol. 17, pp. 28–40, Apr. 2021.

[16] R. Ahuja and S. C. Sharma, "Stacking and voting ensemble methods fusion to evaluate instructor performance in higher education," *International Journal of Information Technology*, vol. 13, no. 5, pp. 1721–1731, Jul. 2021, ISSN: 2511-2112. DOI: 10.1007/s41870-021-00729-4. [Online]. Available: http://dx.doi.org/10.1007/s41870-021-00729-4.

[17] G. Bao et al., "Data augmentation for EEG-based emotion recognition using generative adversarial networks," en, *Front. Comput. Neurosci.*, vol. 15, p. 723 843, Dec. 2021.

[18] E. S. Salama, R. A. El-Khoribi, M. E. Shoman, and M. A. Wahby Shalaby, "A 3d-convolutional neural network framework with ensemble learning techniques for multi-modal emotion recognition," en, *Egypt. Inform. J.*, vol. 22, no. 2, pp. 167–176, Jul. 2021.

[19] A. Subasi, T. Tuncer, S. Dogan, D. Tanko, and U. Sakoglu, "EEG-based emotion recognition using tunable Q wavelet transform and rotation forest ensemble classifier," en, *Biomed. Signal Process. Control*, vol. 68, no. 102648, p. 102 648, Jul. 2021.

[20] S. Chatterjee and Y.-C. Byun, "EEG-based emotion classification using stacking ensemble approach," en, *Sensors (Basel)*, vol. 22, no. 21, p. 8550, Nov. 2022.

[21] X. Du et al., "An efficient LSTM network for emotion recognition from multichannel EEG signals," *IEEE Trans. Affect. Comput.*, vol. 13, no. 3, pp. 1528–1540, Jul. 2022.

[22] E. M. G. Younis, S. M. Zaki, E. Kanjo, and E. H. Houssein, "Evaluating ensemble learning methods for multi-modal emotion recognition using sensor data fusion," *Sensors*, vol. 22, no. 15, p. 5611, Jul. 2022, ISSN: 1424-8220. DOI: 10.3390/s22155611. [Online]. Available: http://dx.doi.org/10.3390/s22155611.

[23] P. Zhong, D. Wang, and C. Miao, "EEG-based emotion recognition using regularized graph neural networks," *IEEE Trans. Affect. Comput.*, vol. 13, no. 3, pp. 1290–1301, Jul. 2022.

[24] S. Cui, Y. Han, Y. Duan, Y. Li, S. Zhu, and C. Song, "A two-stage voting-boosting technique for ensemble learning in social network sentiment classification," *Entropy*, vol. 25, no. 4, 2023, ISSN: 1099-4300. DOI: 10.3390/e25040555. [Online]. Available: https://www.mdpi.com/1099-4300/25/4/555.

[25] Z. Huang, Y. Ma, R. Wang, W. Li, and Y. Dai, "A model for eeg-based emotion recognition: Cnn-bi-lstm with attention mechanism," *Electronics*, vol. 12, no. 14, p. 3188, Jul. 2023, ISSN: 2079-9292. DOI: 10.3390/electronics12143188. [Online]. Available: http://dx.doi.org/10.3390/electronics12143188.

[26] S. Chakraborty, "How efficient are large language models(LLMs) in finding numerical patterns? -an empirical study," 2024.

[27] H. Chen et al., "EEG emotion copilot: Optimizing lightweight LLMs for emotional EEG interpretation with assisted medical record generation," 2024. eprint: 2410.00166 (cs.CV).

[28] Y. Chen, W. Wang, S. Yan, Y. Wang, X. Zheng, and C. Lv, "Application of electroencephalography sensors and artificial intelligence in automated language teaching," en, *Sensors (Basel)*, vol. 24, no. 21, p. 6969, Oct. 2024.

[29] Z. Cheng et al., "Emotion-LLaMA: Multimodal emotion recognition and reasoning with instruction tuning," 2024. eprint: 2406.11161 (cs.AI).

[30] Y. Geng, S. Shi, and X. Hao, "Deep learning-based EEG emotion recognition: A comprehensive review," en, *Neural Comput. Appl.*, Dec. 2024.

[31] S. Hellsten, *Incremental re-tokenization in bpe-trained sentencepiece models*, 2024.

[32] B. Kim and Y. Kwon, *Searching for effective preprocessing method and cnn-based architecture with efficient channel attention on speech emotion recognition*, 2024. DOI: 10.48550/ARXIV.2409.04007. [Online]. Available: https://arxiv.org/abs/2409.04007.

[33] P. Kumar, "Large language models (llms): Survey, technical frameworks, and future challenges," *Artificial Intelligence Review*, vol. 57, no. 10, Aug. 2024, ISSN: 1573-7462. DOI: 10.1007/s10462-024-10888-y. [Online]. Available: http://dx.doi.org/10.1007/s10462-024-10888-y.

[34] A. Mishra, S. Shukla, J. Torres, J. Gwizdka, and S. Roychowdhury, "Thought2Text: Text generation from EEG signal using large language models (LLMs)," 2024. eprint: 2410.07507 (cs.CL).

[35] P. Samal and M. F. Hashmi, "Role of machine learning and deep learning techniques in EEG-based BCI emotion recognition system: A review," en, *Artif. Intell. Rev.*, vol. 57, no. 3, Feb. 2024.

[36] A. Sano, J. Amores, and M. Czerwinski, "Exploration of LLMs, EEG, and behavioral data to measure and support attention and sleep," 2024. eprint: 2408.07822 (eess.SP).

[37] P. Sorino et al., "ARIEL: Brain-computer interfaces meet large language models for emotional support conversation," in *Adjunct Proceedings of the 32nd ACM Conference on User Modeling, Adaptation and Personalization*, Cagliari Italy: ACM, Jun. 2024.

[38] P. Verma and M. Pilanci, "Towards signal processing in large language models," 2024. eprint: 2406.10254 (cs.CL).

[39] Y. Xu, Y. Zhou, Y. Cai, J. Xie, R. Ye, and Z. Wu, "Multimodal emotion captioning using large language model with prompt engineering," in *Proceedings of the 2nd International Workshop on Multimodal and Responsible Affective Computing*, Melbourne VIC Australia: ACM, Oct. 2024, pp. 104–109.

[40] H. Yang et al., "Large language models meet text-centric multimodal sentiment analysis: A survey," 2024. eprint: 2406.08068 (cs.CL).

[41] S. Yao, L. Liu, J. Lu, D. Wu, and Y. Li, "Advancing semi-supervised EEG emotion recognition through feature extraction with mixup and large language models," in *2024 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, Lisbon, Portugal: IEEE, Dec. 2024, pp. 2772–2779.

[42] S. Yin et al., "A survey on multimodal large language models," *National Science Review*, vol. 11, no. 12, Nov. 2024, ISSN: 2053-714X. DOI: 10.1093/nsr/nwae403. [Online]. Available: http://dx.doi.org/10.1093/nsr/nwae403.

[43] B. Yousefipour, V. Rajabpour, H. Abdoljabbari, S. Sheykhivand, and S. Danishvar, "An ensemble deep learning approach for EEG-based emotion recognition using multi-class CSP," en, *Biomimetics (Basel)*, vol. 9, no. 12, Dec. 2024.

[44] Z. Zhang, S.-h. Zhong, and Y. Liu, "Torcheegemo: A deep learning toolbox towards eeg-based emotion recognition," *Expert Systems with Applications*, vol. 249, p. 123 550, 2024, ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2024.123550. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0957417424004159.