Budapest University of Technology and Economics
Faculty of Electrical Engineering and Informatics
Department of Telecommunications and Artificial Intelligence

Sadi Mahmud Shurid

Decoding Brainwaves into Words using Neural EEG-to-Text Generation with LLMs

SUPERVISOR

Dr. Al-Radhi Mohammed Salah

BUDAPEST, 2025

Contents

STUDENT DECLARATION

I, **Sadi Mahmud Shurid**, the undersigned, hereby declare that the present BSc thesis work has been prepared by myself and without any unauthorized help or assistance. Only the specified sources (references, tools, etc.) were used. All parts taken from other sources word by word, or after rephrasing but with identical meaning, were unambiguously identified with explicit reference to the sources utilized.

I authorize the Faculty of Electrical Engineering and Informatics of the Budapest University of Technology and Economics to publish the principal data of the thesis work (author's name, title, abstracts in English and in a second language, year of preparation, supervisor's name, etc.) in a searchable, public, electronic and online database and to publish the full text of the thesis work on the internal network of the university (this may include access by authenticated outside users). I declare that the submitted hardcopy of the thesis work and its electronic version are identical.

Full text of thesis works classified upon the decision of the Dean will be published after a period of three years.

Budapest, 12 December 2025

Sadi Mahmud Shurid

Abstract

Translating brain activity into natural language has become an emerging direction in brain-computer interfaces, offering new possibilities for assistive communication. Yet, working with EEG signals remains challenging, as they are often noisy and capture only an indirect view of the underlying neural processes. These limitations make it difficult for existing systems to learn both fine-grained patterns and the broader temporal structure present in brain responses.

This thesis explores an approach that aims to improve EEG-to-Text generation by introducing an encoder that blends convolutional layers with Transformer based modeling. The design allows the network to learn short-term features while also capturing relationships that unfold across the full EEG sequence. The complete framework follows a three-stage training process, and the system was tested with two different large language models. Both the quantitative and qualitative evaluations indicate that the enhanced encoder produces more coherent descriptions than a purely CNN-based baseline. Overall, the results suggest that incorporating global context modeling can meaningfully strengthen EEG-to-Text generation.

1 Introduction

Electroencephalography to text (EEG-to-Text) generation is one of the most recent approaches of brain-computer interfaces (BCIs). Its aim is to translate non-invasive (recorded without penetrating the human body) brain signals into natural language.

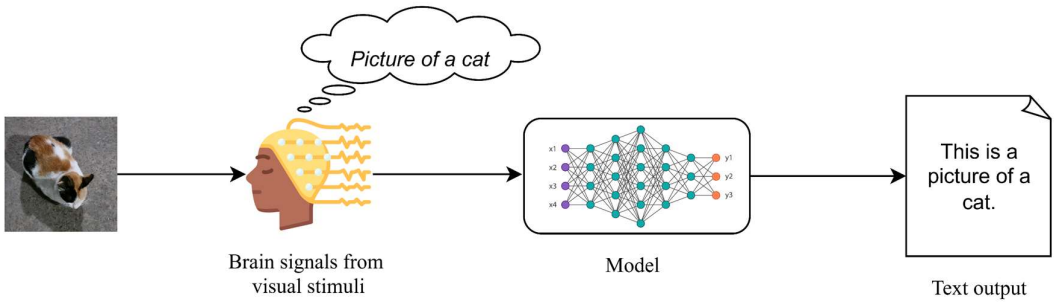

Figure 1: The introductory idea of decoding brainwave (in form of EEG) to text.

As illustrated in **Figure 1**, the process starts from a person observing a visual (an image). This induces activity in the brain to be measured and recorded via EEG setup. Then it is to be passed via a "model" (symbolic at this point in **Figure 1**). In the end the output is expected in form of a text containing what the person would have probably thought while looking at the visuals.

1.1 Background and Motivation

BCI research has evolved from early demonstrations of "mental prosthesis" communication to current efforts at decoding complex thoughts. Early BCIs predominantly relied on event related potentials [1] and other explicit neural responses to allow basic communication. Text generation from EEG is a logical next step. It aims to produce descriptions or sentences reflecting a person's "thoughts" or "perceptions." Early attempts toward this goal have been modest, for instance, classifying a small set of imagined words from EEG signals [2]. However, with the surge in natural language processing capabilities, especially with large language models (LLMs), there is a strong motivation to enable open vocabulary EEG-to-Text generation. Instead of limiting the user to a fixed dictionary or spelling interface, an LLM based BCI could, in principle, express any idea the user may have in mind in natural language. This would greatly

enhance the communicative bandwidth of BCIs, benefitting users with speech or motor impairments by allowing more fluid and semantically rich communication.

However, EEG signals are kind of an indirect reflection of underlying neural activity [3]. Therefore, it is extremely important to ensure proper mapping for EEG-to-Text gneration, also, structure the language models (e.g. deepseek-llm-7b-chat [4] in our case) in a proper way so that the outputs are semantically controlled.

1.2 Thesis Objectives and Structure

This thesis investigates the generation of natural language text (in English) from EEG signals by proposing an enhanced neural architecture for EEG-to-Text translation integrated with LLMs. The main aim is to strengthen representation learning by modeling both short term and long range patterns in brain activity.

Research Objectives: The core objective is to design and test an enhanced encoder that combines convolutional neural networks (CNN) with Transformer layers. This architecture is intended to capture complex spatiotemporal patterns in EEG data more effectively. The work also aims to show that the performance stability of this encoder architecture integrated with newly released LLM which has not been tested in this kind of frameworks yet.

Thesis Contributions: This thesis offers three main contributions. Firstly, it proposes a CNN-Transformer encoder that handles EEG signals more effectively than earlier CNN-based models. The CNN extracts local features, while the Transformer layers capture temporal relationships across the full recording window. Then, it evaluates this encoder with two different LLMs. The results show that the encoder improvement benefits both models, meaning the gains come from better EEG representations rather than the choice of language model. Lastly, it provides quantitative and qualitative evidence of improvements compared to existing baseline; achieving a fair amount of relative increase in object classification accuracy and improves most text generation metrics.

Thesis Organization: The rest of the thesis is structured as follows. Chapter 2 reviews related work on EEG based BCIs, neural models for EEG processing, and approaches that link neural data with LLMs. Chapter 3 describes the dataset, baseline system, proposed encoder, training method, and evaluation setup. Chapter 4 reports the evaluation in details

by discussing the quantitative results and qualitative examples. Chapter 5 concludes with the main findings, limitations, and potential future work.

2 Related Work

EEG refers to a non-invasive technique for recording brain activity using sensors placed on the human scalp [5]. By capturing tiny electrical signals produced by neurons, EEG allows us to peer into a person's brain activity. Using EEG to generate written text (essentially translating "thoughts" into words) is an exciting goal with potential applications in assistive communication (helping paralyzed or non-verbal patients convey thoughts) and human-computer interaction.

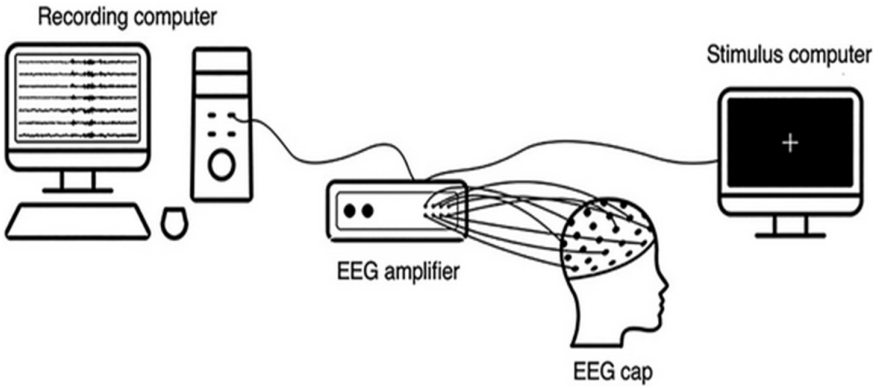

Figure 2: Schematic of a typical EEG recording setup [5].

2.1 EEG-to-Text Generation Techniques

Researchers in BCIs have been aiming to translate brain activity directly into natural language for a long time. The ability to translate a human's brain signals into text, or, in other words, reading "thoughts" promises new communication pathways for individuals with paralysis, disabilities or speech impairments. Early BCI systems, however, achieved communication in more constrained ways. For example, P300 speller interfaces allowed users to select letters by focusing attention on flashing choices, relying on event related EEG responses to output words by one character at a time [6]. Such systems demonstrated that EEG could be used for communication, but they were slow and limited to controlled selection tasks rather than language generation from brain activity.

Being non-invasive, EEG measures the brain's electrical activity from the scalp and is considered safe, also portable. However, it presents significant challenges for text

decoding. EEG signals have low spatial resolution and a poor signal-to-noise ratio. They vary greatly across individuals. These properties make it difficult to map EEG patterns reliably to the complex space of words and sentences. In contrast, invasive recording methods like electrocorticography (ECoG) offer cleaner signals and have shown remarkable success in neural speech decoding. For instance, researchers have translated ECoG signals into text by decoding imagined handwriting or synthesizing spoken sentences from cortical activity [1]. This work demonstrated one of the first encoder and decoder frameworks that converted ECoG signals directly into text [1]. Similarly, in another work researchers have enabled a paralyzed patient to communicate at 90 character/minute by decoding neural signals for handwriting into text [7]. These invasive approaches highlight the potential for brain-to-text communication, but their surgical requirements make them impractical and inconvenient for widespread use.

With EEG, early efforts toward language output focused on classification tasks rather than thoughtful sentence generation. Researchers have trained EEG classifiers to recognize a limited set of words or phonemes that a user is thinking of, for example, by classifying a small set of vocabulary of imagined words from EEG signals [2]. Other studies targeted even more elementary units, like identifying phonological categories or syllable rhythms from EEG. Beyond linguistic content, BCI based on EEG research has also included tasks like recognizing visual stimuli that a person sees or detects when a certain keyword is heard. These classification approaches treat brain to text as a recognition problem (selecting from predefined categories) rather than truly generating novel sentences based on raw thoughts.

Despite some success in classification, the open-ended generation of natural language from EEG remained largely uncharted until recent years. The difficulty lies in EEG signal's ambiguity. There is no simple or direct mapping from an EEG waveform to a specific word or sentence. One cannot reliably pinpoint, for instance, an "Aha, the person just thought of the word *aeroplane*" moment in a raw EEG signal. This is why most of the earlier EEG based communication systems either used spelling interfaces or limited vocabulary classification [2]. In the mid-2010s, Herff and Schultz [3] reviewed the state of brain-to-text research and highlighted the challenge that even cutting-edge attempts at EEG-based speech recognition were achieving only modest accuracies on small word sets [3]. The general consensus was that EEG-to-Text translation required more advanced

machine learning and likely multimodal assistance to succeed, which set the stage for the deep learning approaches.

A key development reigniting interest in EEG-to-Text was the rise of deep learning and language models. By the early 2020s, LLMs had demonstrated an ability to generate fluent text given appropriate inputs. Researchers began to ask whether those generative abilities could be exploited to decode brain signals. However, applying LLMs to neural data is not a straightforward task. An LLM expects inputs in the form of text or similarly rich representations, whereas EEG signals are noisy time-series data. Bridging this gap became a focus of this field of the BCI research.

Recent work has shown promising progress by using multimodal alignment techniques and deep neural networks to translate EEG into a form that an LLM can understand. Rather than attempting a direct translation from raw EEG-to-Text (which would require an impractically large training corpus of brain data paired with text), these new approaches break the problem into different parts. They often involve an intermediate step of mapping EEG data into a semantic representation, leveraging another modality such as images or known concepts (We will explore these strategies in Section 2.3 and will be discussed further in details in the "Methodology" chapter). Notably, in 2023 a team at Meta AI demonstrated that with proper machine learning models, even non-invasive recordings can recover surprising amounts of linguistic information [8]. They decoded heard speech from brain recordings with performance far above chance [8]. While their work used data from magnetoencephalography (MEG) [8] recording and focused on speech perception (identifying what a person was hearing) rather than spontaneous speech generation, it provided a promising pathway that modern deep learning can extract semantic content from non-invasive brain signals. This and similar advances in neural decoding have set the stage for true EEG-to-Text translation systems.

In summary, earlier EEG-to-Text techniques were constrained to classification and control paradigms, highlighting the difficulty of the task. Only in the last couple of years have researchers started to combine powerful neural network encoders with language-generation models to move beyond classification toward fluent text generation. The following sections discuss the neural network architectures that make learning useful EEG representations possible (Section 2.2) and then the multimodal alignment and prompting approaches that link those representations to language models (Section 2.3).

2.2 Neural Architectures for EEG Representation

The central aspect to any EEG-to-Text system is an EEG encoder. An encoder transforms raw EEG signals into a meaningful internal representation that can be used for downstream tasks. Designing an effective encoder is challenging because EEG data are high-dimensional time series with complex spatiotemporal patterns. Modern approaches use deep neural networks to automatically learn feature representations from the data. Two classes of neural architecture have proven especially useful for EEG representation: convolutional neural networks (CNNs) [9] and Transformers [10].

CNNs are a natural fit for EEG because they excel at extracting local patterns from multivariate signals [11, 12]. In EEG, relevant information is often encoded in temporal windows (e.g. an oscillation or waveform occurring over a few hundred milliseconds) and across particular subsets of electrodes. CNNs can capture these patterns through learned filters that slide over time and across channels. A prime example is EEGNet [11], a compact CNN architecture introduced for EEG-based BCIs [11]. EEGNet [11] uses convolutional filters to first capture frequency-specific temporal features (by convolving along the time dimension) and then spatial filters to capture relationships across EEG channels [11]. Despite having only a few thousand parameters, EEGNet [11] achieved competitive performance on a variety of BCI tasks, including event related potential detection and motor imagery classification. Its success demonstrated that CNNs can learn efficient representations from raw EEG, often outperforming earlier approaches that required manual feature extraction.

In the context of EEG-to-Text translation, CNN based encoders have been widely used in recent frameworks [12]. While CNNs have become a standard tool for EEG feature extraction, the Transformer [10] architecture offers an alternative that has attracted increasing attention. Transformers were first developed for language processing, but their core mechanism “self-attention” is general and can be applied to any sequence data [10]. However, applying Transformers to EEG is non-trivial. Transformers typically have many more parameters than CNNs, which raises the risk of overfitting, especially given that EEG datasets tend to be small (often only a few of hours of data or even minutes). To address this, researchers have explored hybrid models that combine Transformers with CNN-like elements to get the best of both worlds, for instance, a framework called EEGEncoder [13] was introduced for motor imagery classification that fuses Temporal Convolutional Networks with modified Transformer blocks [13]. In their design,

convolutional layers first extract low-level features and downsample the signal, and then Transformer style attention layers operate on these features to capture long range temporal relations [13]. By using a dual-stream architecture (one focusing on temporal features, one on spatial features) and then merging them, they achieved state-of-the-art accuracy on a standard motor imagery EEG dataset [13]. This demonstrates a possible trend of hybrid CNN-Transformer encoders for EEG.

In summary, CNNs remain the workhorse for EEG feature extraction due to their efficiency and strong track record in BCI tasks, whereas Transformer based models are an emerging frontier showing potential for capturing more complex patterns in EEG data [14]. In practice, current EEG-to-Text systems often start with a CNN encoder. Mostly because successful prior frameworks did so. But researchers are actively exploring the possibility of combined models to replace or augment the CNN. As we move to multimodal EEG-to-Text pipelines, the encoder's job is to produce a representation rich enough to be mapped to semantic concepts, which both CNNs and Transformers can achieve. The next section will discuss how such EEG representations are aligned with other modalities and fed into language models to generate text.

2.3 Multimodal alignments and LLM based approaches

A crucial challenge in EEG-to-Text translation is linking neural signal representations to text outputs. Recent approaches address this by introducing a multimodal alignment phase that trains an EEG encoder to map brain signals into a shared semantic space compatible with language models. This enables subsequent use of LLMs to generate text descriptions based on aligned neural embeddings. The Thought2Text [12] system exemplifies this strategy, adopting a three stage pipeline in which EEG representations are first aligned with visual features, then used to fine-tune multimodal captioning models, and finally adapted so that the LLM can generate text directly from EEG-derived embeddings [12].

In this pipeline, an EEG encoder is trained to produce embeddings that approximate corresponding image features, facilitating alignment with pretrained vision-language spaces. Next, an instruction-tuned LLM such as Mistral-7B [15] is fine-tuned on paired visual and textual data to generate descriptive captions. Finally, the language model is

further tuned to accept EEG aligned representations as inputs, enabling direct EEG-to-Text translation during inference.

An important observation from this line of work is that using instruction-tuned LLMs, such Mistral-7B [15], influences the nature of generated outputs. These models tend to produce verbose and detailed descriptions, which can inflate overlap-based evaluation scores like BLEU [16] even if some details are not strictly followed by the observed image visual.

3 Methodology

3.1 System Overview

This thesis extends the state-of-the-art Thought2Text [12] framework by adding architectural improvements to the EEG encoder and testing the system with multiple language models. We have a three-stage training approach with a proposed encoder to capture both local patterns (through CNNs) and global context (through Transformers). The complete pipeline transforms raw EEG recordings into text descriptions through three main components: (1) a proposed CNN-Transformer encoder that processes brain signals, (2) a projection layer that connects the encoder to the language model, and (3) an instruction-tuned LLM that generates the actual text. Unlike simple classification systems [2, 7], our approach can generate open-ended descriptions directly from neural activity while keeping up with the global context of brain activity unlike only CNN dependent encoders such as in the baseline [12].

3.2 Dataset and Preprocessing

3.2.1 Data Source

We use a publicly available EEG dataset[1] where six people viewed images from 40 different object categories (selected from 1000 different ImageNet[2] classes). During data collection, participants saw visual stimuli while their brain activity was recorded using a 128-channel EEG system. Each trial shows a fixation cross, then the image for 0.5 seconds, followed by a brief rest period. The raw signals were sampled at 1000 Hz and preprocessed with a 5-95 Hz bandpass filter to keep relevant frequencies while removing noise.

[1] https://drive.google.com/drive/folders/1XqV6MMl28iYXkQBMEFHfEXllGmCbqpOu

[2] https://deeplearning.cms.waikato.ac.nz/user-guide/class-maps/IMAGENET/

3.2.2 Data Preprocessing Pipeline

The preprocessing pipeline implements several transformations to prepare raw EEG recordings for neural network processing. First, temporal windowing extracts relevant signal segments corresponding to the visual stimulus presentation. The analysis focuses on the time interval from 20 to 460 milliseconds post-stimulus onset, capturing the critical period of visual information processing while excluding early sensory transients and late cognitive components [12]. The extracted temporal segments undergo standardization across channels to account for inter-electrode variance in signal amplitude. Each EEG trial is represented as a tensor of shape (1, 128, 440), where the first dimension represents the single-channel depth, the second corresponds to the 128 spatial electrodes, and the third captures the 440 temporal samples (equivalent to 440 milliseconds at the original sampling rate). This representation preserves both the spatial arrangement of electrodes and the temporal dynamics of neural activity. The final curated dataset maintains sufficient sample diversity for robust model training while adhering to strict quality standards.

3.3 Baseline Architecture

This section describes the baseline Thought2Text framework [12] that our work extends. We provide this overview to establish context for our architectural modifications described in Section 3.3. **Figure 3** illustrates the framework which serves as the baseline for this thesis. The system works in three stages: first, it trains an EEG encoder to understand brain signals; second, it aligns these signals with visual concepts using CLIP; and third, it connects everything to an instruction tuned language model, such as Mistral-7B-Instruct [17], that generates text descriptions.

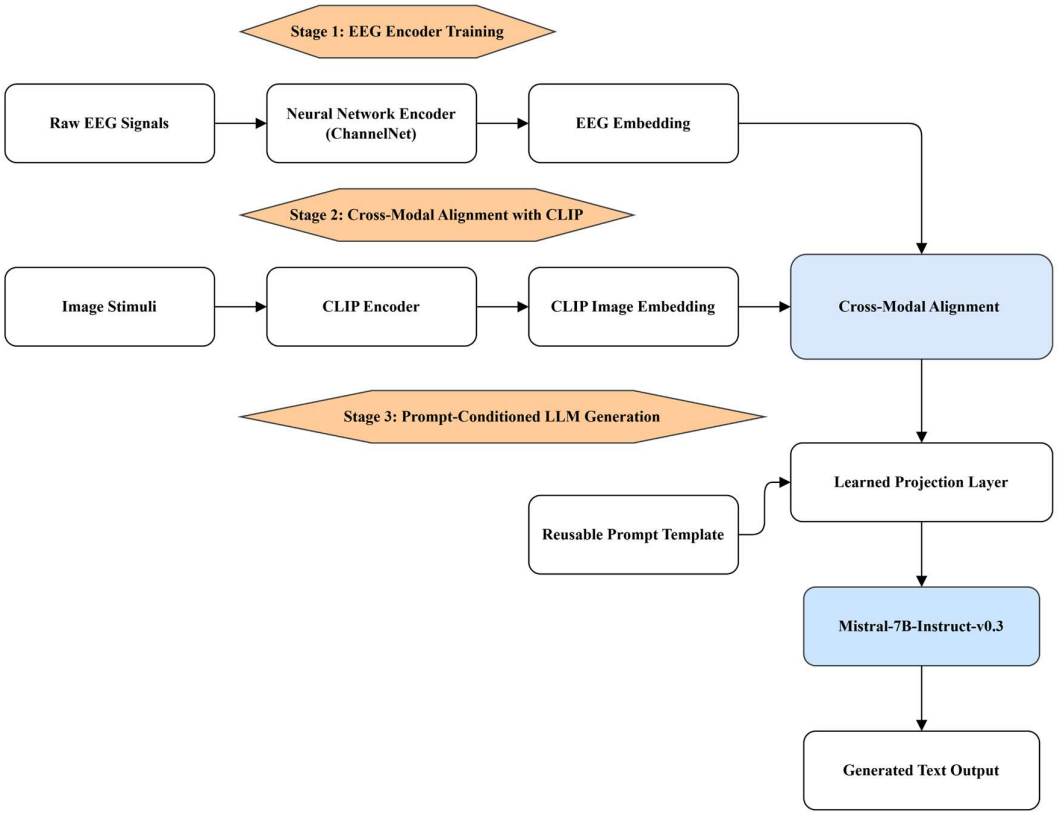

Figure 3: Overview of the baseline [12] framework. The pipeline consists of three training stages.

3.3.1 Stage 1: EEG Encoder Training

The first stage trains a neural network to process raw EEG recordings and convert them into meaningful representations. The encoder used here is ChannelNet [18], a CNN architecture designed specifically for visual classification [19] from multi channel time-series data like EEG. The ChannelNet encoder architectureprocesses EEG signals through three main components:

Temporal Block: This part handles the time dimension of EEG signals. It uses five parallel convolutional layers with different dilation rates (1, 2, 4, 8, 16), which lets the network capture both quick changes and longer patterns in brain activity. Each layer uses 1×33 kernels that slide along the time axis. The output from all four layers gets concatenated together, giving the network features at multiple time scales.

Spatial Block: After temporal processing, the spatial block looks at relationships between different electrode positions on the scalp. It uses four parallel branches with different kernel sizes to capture both local activity (from nearby electrodes) and broader patterns

(across distant electrodes). This multi-scale approach helps the network understand spatial patterns in brain activity.

Residual Blocks: Finally, four residual blocks refine the extracted features. Each block has two 3×3 convolutional layers with skip connections that help with training stability. Between blocks, downsampling layers gradually reduce the spatial and temporal dimensions. At the end, a 1×1 convolution compresses everything down to 50 channels.

Creating the EEG Embedding: The final CNN output gets flattened and passed through a linear layer to create a 512-dimensional embedding. This compact representation captures the essential information about what the person saw, encoded in their brain activity.

3.3.2 Stage 2: Alignment with CLIP

Stage 2 connects EEG representations to visual concepts using Contrastive Language Image Pretraining (CLIP), a model pretrained on millions of image-text pairs. CLIP stays frozen during training - we don't change its weights at all. Instead, we train the EEG encoder to produce embeddings that match CLIP's image embeddings.

How CLIP Works: CLIP processes images using a Vision Transformer that splits each image into small patches and analyzes them with self-attention. For our 224×224 images, CLIP produces 512-dimensional embeddings that capture high-level visual concepts. These embeddings serve as targets for the EEG encoder to match.

The encoder learns from two losses simultaneously:

MSE Loss: Mean squared error (MSE) loss measures how close the EEG embedding is to CLIP's image embedding. By minimizing this distance, the encoder learns to map brain signals to the same semantic space that CLIP uses for images, and that is how it encourages semantic alignment.

Classification Loss: A separate linear classifier tries to predict which of the 40 object categories the person was viewing. This cross-entropy loss ensures the encoder learns discriminative features that can distinguish between different objects.

The total loss is simply:

$$L_{\text{total}} = L_{\text{MSE}} + L_{\text{CE}} \quad (1)$$

where, L_{total} is the combined loss function; L_{MSE} is the mean squared error loss measuring the distance between EEG and CLIP image embeddings (encouraging semantic alignment), and L_{CE} is the cross-entropy classification loss for predicting the viewed object category from 40 different classes.

3.3.3 Stage 3: Text Generation with LLM

The final stage connects the trained EEG encoder to a large language model. This involves two new components: (i) a projection layer that bridges the dimensional gap, and the LLM itself that generates text.

Projection Layer: The EEG encoder outputs 512-dimensional embeddings, but Mistral-7B expects 4096-dimensional inputs. A learned linear projection handles this transformation:

$$h_{\text{LLM}} = W_{\text{proj}} \times e_{\text{EEG}} + b_{\text{proj}} \quad (2)$$

where, h_{LLM} is the projected vector fed into the language model; W_{proj} is the learned linear projection matrix (mapping the encoder's embedding space to the LLM input space), e_{EEG} is the 512 dimensional embedding produced by the EEG encoder, and b_{proj} is the learned bias term.

This projection layer is the key interface between brain signals and language generation since it translates EEG representations into something the LLM understands.

3.3.4 Problem Formulation and Why We Need a Better Encoder

While the state-of-the-art Thought2Text [12] framework demonstrates effective EEG-to-Text translation; the ChannelNet encoder has an inherent limitation: CNNs process signals through local receptive fields, making it difficult to capture long-range temporal dependencies in EEG data. Brain activity patterns often span extended time periods (for example, sustained attention or gradual changes in neural state), which may be missed by purely convolutional approaches. This limitation motivates our proposed enhanced encoder architecture (Section 3.4), which adds Transformer layers to explicitly model global dependencies through self-attention [10] mechanisms while maintaining ChannelNet's strength in local feature extraction.

3.4 Proposed Methodology

Although the baseline encoder [12] provides strong performance in local feature extraction, it remains limited in its abilitiy to capture long-range patterns in EEG signals. CNNs only look at nearby values in each layer, so they tend to miss important global context. To address this, we propose a an enhanced encoder architecture that keeps ChannelNet's strong local feature extraction but adds Transformer layers to model global dependencies across the entire temporal sequence. As illustrated in **Figure 4**, the model first uses ChannelNet to extract local spatiotemporal EEG features, which CNNs handle effectively but only within limited receptive fields. This means long range temporal structure may be overlooked. To overcome this limitation, the architecture incorporates Transformer layers after the CNN feature extractor, enabling the model to capture global dependencies across the entire sequence while preserving ChannelNet's strong local feature extraction.

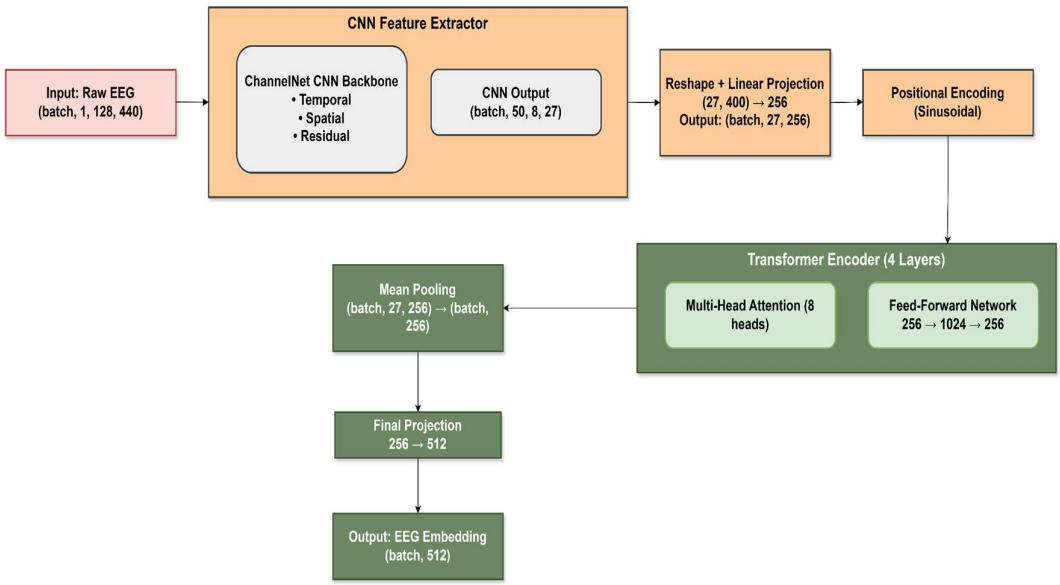

Figure 4: Proposed CNN–Transformer encoder architecture. ChannelNet extracts local EEG features within limited receptive fields, after that a sequence projection and positional encoding enable Transformer layers to model global temporal relationships across the entire EEG sequence.

3.4.1 Architectural Motivation

EEG signals have structure at multiple time scales. Quick events like sudden attention shifts need local processing, which CNNs handle well. But other patterns such as sustained focus or gradual changes in mental state span longer time periods, and require

looking at the whole sequence at once. Transformers are perfect for this because they use self-attention to compute relationships between all positions in a sequence simultaneously. However, applying Transformers directly to raw EEG (with 440 time points) would be computationally expensive. Our proposed approach solves this by using the CNN to first compress the signal down to a shorter sequence (~27 time steps), then applying Transformers to this compact representation. This gives us the best of both worlds: i. CNN layers extract local temporal and spatial features efficiently, ii. Transformer layers model global context and long-range dependencies, iii. The two components work together in a single architecture.

3.4.2 CNN-to-Transformer Transition

Our architecture starts with ChannelNet that produces a feature map with shape (batch, 50, 8, 27), where 50 is the number of channels, 8 is the reduced spatial dimension, and 27 is the compressed temporal length. To feed this into Transformer layers, we need to convert it into a sequence format. We do this by treating the width dimension (27) as the sequence length and flattening the channel and height dimensions ($50 \times 8 = 400$) into the feature dimension at each time step. This gives us a sequence of shape (batch, 27, 400). Next, a learned linear projection maps each 400-dimensional time step to 256 dimensions, which is the internal dimension our Transformer uses. This projection serves two purposes: it reduces the feature size for efficiency, and it provides a learnable adaptation layer between the CNN and Transformer components.

Transformers process all sequence positions in parallel, so they don't inherently know the order of time steps. To give the model temporal awareness, we add positional encodings to the sequence before the Transformer layers. We use sinusoidal positional encodings. This works by assigning each position a unique pattern of sine and cosine waves. For position t and dimension i, and model dimension d, the encoding is:

$$\sin\left(\frac{t}{10000^{\frac{2i}{d}}}\right) \tag{3}$$

$$\cos\left(\frac{t}{10000^{\frac{2i}{d}}}\right) \tag{4}$$

These patterns help the model understand temporal distance - nearby positions have similar encodings, while distant positions have more different encodings. The sinusoidal

approach has a nice property: it can handle sequence lengths the model hasn't seen during training, and it doesn't require any learnable parameters.

3.4.3 Transformaer Encoder Layers

After adding positional encodings, the sequence passes through four Transformer encoder layers. Each layer has two main components:

Multi-Head Self-Attention: This is where the model looks at relationships between all time steps. We use 8 attention heads, each working in a 32-dimensional subspace (256 / 8 = 32). The attention mechanism works like this:

1. For each position, compute how similar it is to every other position.
2. Normalize these similarities into weights that sum to 1.
3. Use the weights to create a new representation as a weighted average.

The multi-head setup lets different heads focus on different types of patterns. For example, one head might focus on nearby time steps while another looks at distant relationships.

Feed-Forward Network: After attention, each position passes independently through a small neural network. This network has two linear layers with a GELU activation in between. The inner dimension is 1024 (4× the model dimension), which gives the network enough capacity to transform features in complex ways. Both sub-layers use a few important tricks for stable training:

1. Layer normalization before each sub-layer (pre-norm architecture)
2. Residual connections that add the input to the output
3. Dropout (rate 0.1) for regularization

The four stacked layers let the model build increasingly abstract representations. Early layers might focus on local patterns, while later layers integrate information across the entire sequence. After all four Transformer layers, we have a sequence of shape (batch, 27, 256). To get a single embedding per EEG trial, we use mean pooling. It just takes the average across all 27 time steps. This gives us one 256-dimensional vector that summarizes the entire sequence.

Finally, we project this vector up to 512 dimensions to match the expected embedding size for the rest of the pipeline. This projection includes layer normalization to keep the output distribution stable. The final 512-dimensional embedding represents the EEG signal with both local features (from the CNN) and global context (from the Transformer).

3.5 Training Pipeline

We train and implement the enhancded encoder architecture in three stage approach. **Figure 5** illustrates the complete training pipeline, showing how the stages work together: from training the encoder to generating text from the eeg embeddings; each stage has a specific learning objective, and training them sequentially prevents interference between different goals.

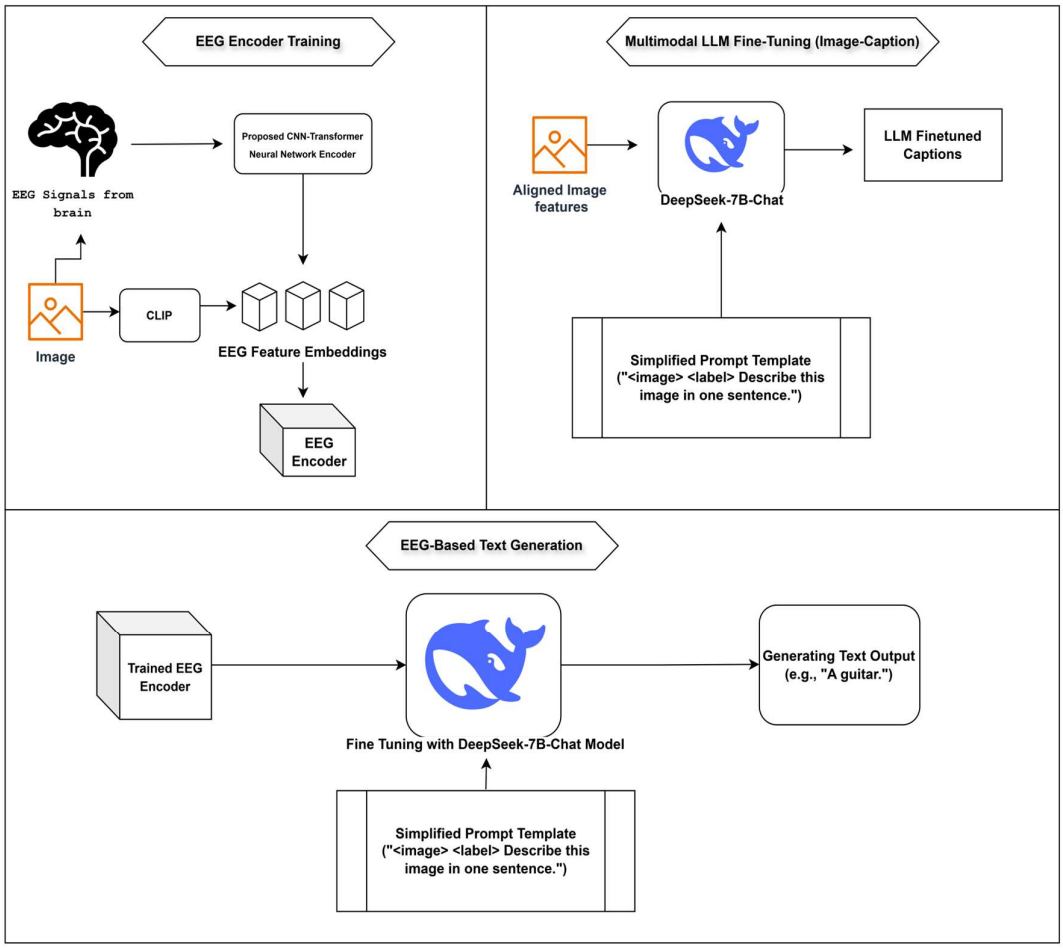

Figure 5: Complete overview of training pipeline for the proposed EEG-to-Text generation system.

3.5.1 Training the Encoder

In the first stage, we train our proposed CNN-Transformer encoder from scratch so that the encoder learns to produce 512-dimensional embeddings that both align with CLIP's visual representations and contain discriminative information for object classification. The training uses EEG signals that have been filtered to the 5-95 Hz frequency range, with data splits organized by unique images rather than by subject. This ensures the model learns to generalize across different visual stimuli rather than memorizing specific examples. We train with a batch size of 8 samples for 100 epochs, using the learning rate of 5×10^{-5}.

The encoder learns from two loss functions simultaneously. The MSE loss measures the squared Euclidean distance between EEG embeddings and CLIP image embeddings, encouraging the encoder to map brain signals into the same semantic space that CLIP uses for visual concepts. The classification loss uses cross entropy to train a linear classifier that predicts which of the 40 object categories the person was viewing. We combine these objectives with equal weight, so the total loss is simply the sum of MSE and classification losses. During each training iteration, we extract EEG embeddings using the encoder and obtain CLIP image embeddings from the corresponding visual stimuli. CLIP remains frozen throughout this stage, serving only as a source of target embeddings. We compute both the MSE loss between embeddings and the classification loss from the encoder's classifier head, then backpropagate the combined loss to update only the encoder's parameters. This includes all CNN layers, Transformer layers, and the classification head. By the end of this stage, the encoder produces 512 dimensional EEG embeddings capturing both CLIP aligned semantic structure and object discriminative features. The following script was used for training:

```
python train_eeg_classifier.py \
  --eeg_dataset data/block/eeg_55_95_std.pth \
  --splits_path data/block/block_splits_by_image_all.pth \
  --output ./hybrid_eeg_encoder \
  --image_dir data/images/ \
  --batch_size 8 \
  --num_epochs 100 \
  --learning_rate 5e-5
```

3.5.2 LLM Fine-Tuning

After training the EEG encoder, we connect it to a large language model for caption generation. The next two stages are executed in one script but represent two separate fine-tuning phases: first with image embeddings, then with EEG embeddings. Both stages follow the same training setup to ensure consistency. The following script(s) (It was slightly modified based on whether we are using DeepSeek or Mistral) was prepared to complete the finetuning phase:

```
python finetune_llm.py \
  --eeg_dataset data/block/eeg_55_95_std.pth \
  --splits_path data/block/block_splits_by_image_all.pth \
  --eeg_encoder_path ./hybrid_eeg_encoder \
  --image_dir data/images/ \
  --output deepseek_chat_hybrid_eeg_model \
  --llm_backbone_name_or_path deepseek-ai/deepseek-llm-7b-chat \
  --load_in_8bit \
  --bf16 \
  --batch_size 2 \
  --gradient_accumulation_steps 32
```

We train with a batch size of 2 and accumulate gradients over 32 steps (effective batch size 64). Each stage runs for 5 epochs using 1×10^{-5} learning rate.

Image based fine-tuning stage: This stage trains the projection layer to condition the LLM on visual information using image-caption pairs. CLIP produces a 512-dimensional image embedding, which is transformed by the projection layer before entering the frozen LLM. Training examples following the Mistral instruction format:

"[INST] You are a helpful assistant. <image> <object_label> Describe this image in one sentence: [/INST]" followed by the caption target.

The <image> token is replaced by the processed embedding, and <object_label> is filled with the category label (e.g., cat, airplane). Loss is computed only over the caption text. This stage teaches the projection layer how to express visual concepts in the LLM's language space. That learned mapping becomes the foundation for EEG-based generation in the next stage.

EEG Based Fine-Tuning Stage: At this next stage, training continues automatically with EEG embeddings. Before starting, we filter the training data by using the encoder to classify all examples and keeping only those where the predicted label matches the ground truth. This keeps about 60–70% of samples but ensures that the input embeddings truly reflect the viewed object. The projection layer now learns to map EEG derived embeddings into the same semantic space it learned for images. The category label included in the prompt still provides useful guidance to the model.

By the end of this stage, the system can generate coherent descriptions directly from recorded EEG responses to visual stimuli.

3.5.3 Implementation Details

We implement the training pipeline using the HuggingFace Transformers library.[3] It provides standardized interfaces for model loading, tokenization, and training loops. The Trainer API automatically handles gradient accumulation, mixed precision training, and checkpoint saving, which simplifies our implementation and ensures best practices. The small batch size of 2 combined with large gradient accumulation of 32 steps gives us an effective batch size of 64. This approach lets us train with limited GPU memory while still benefiting from the stability that larger batches provide. The 8-bit quantization applied to the LLM is crucial for feasibility. All of the implementation was done on the lightning.ai platform leveraging the computation capability of the NVIDIA L4 GPU. With our configuration, Stage 1 takes approximately 7 hours to run for 100 epochs. The finetuning stages take approximately 5 and 3 hours respectively. The actual training time depends on the specific GPU model and parameter configuration.

3.6 Evaluation Setup

After training, we evaluate the complete pipeline on held out test data that wasn't seen during any training stage. The evaluation assesses how well the system can generate

[3] The complete implementation steps, including training scripts and model configurations, is available at: https://github.com/Sadi-Mahmud-Shurid/DecodingBrainwaves

text descriptions from EEG signals and how effectively the encoder captures object information from brain activity.

3.6.1 Inference Protocol

At test time, the system receives a raw EEG recording and generates a text description without any access to ground-truth information. The inference process follows a straightforward pipeline. First, we preprocess the EEG signal by extracting the 20 to 460 millisecond time window from stimulus onset and normalizing each of the 128 channels independently. This preprocessing matches exactly what was done during training. Next, the preprocessed EEG passes through the frozen combined encoder to obtain a 512-dimensional embedding. This embedding captures the neural representation of what the person saw, encoded in a semantic space aligned with visual concepts through the training process. The embedding then passes through the learned projection layer, which transforms it from 512 dimensions to 4096 dimensions to match the LLM's expected input space.

We insert the projected embedding into the prompt template along with the object label, following the same format used during training. The prompt structure is: "[INST] You are a helpful assistant. <image> <object_label> Describe this image in one sentence: [/INST]" where <image> is replaced with the projected EEG embedding and <object_label> contains the category name. The frozen LLM then generates a text description with a maximum generation length of 64 tokens.

As a side output, the encoder also produces classification logits for all 40 object categories. We take the argmax over these logits to obtain the predicted object category, which allows us to compute classification accuracy as an additional measure of how well the encoder captures stimulus information. In **Figure 6**, we can observe the inference: the word after suffix is the the classified object, followed by an instruction. Then below the warning we can we can notice our generated output based on what the person saw while he was looking at the image; and also the expected output string. After completing all the 1987 iterations, the outputs gets saved in a csv file (including the classified object and outputs generated against the expected output) which is later used for our metrics based evaluations. The used metrics will be explained in the next section, and the results will be discussed in the next chapter.

Figure 6: Terminal screenshot showing inference outputs for samples from 296 to 300 out of 1987, including classified object, expected output and generated output.

3.6.2 Evaluation Metrics

To quantitatively evaluate the output performance of our EEG-to-Text translation, we have used several standard evaluation metrics. These include BLEU [16], METEOR [20], ROUGE-L [21], and BERTScore [22]. Each of these measure a different aspect of similarity between the generated text and the reference caption. We compare the results of our proposed model within its frameowrk and also against the existing models. The following subsections will clarify the chosen metrics, how these metrics work and the context in detail.

BLEU: BLEU [16] (bilinguan evaluation understudy) is one of the common metrics used to judge the quality of machine translation. However, it can be considered a bit outdated now, or maybe useful depending on usecases. It works by counting lexical overlap, meaning counting overlapping sequence of words (n grams) between the generated output and the textual reference. For example, BLEU-1 counts matching single words (also called unigrams), while BLEU-4 counts matching sequences of four words. BLEU

heavily emphasises exact lexical overlap, therefore, a high BLEU score indicates the model reproduced many of the same words or phrases like the reference. However, this also means BLEU is relatively insensitive to paraphrasing. Because using different words with the same meaning won't score high unless there is exact word overlap. Notably, BLEU's stringency grows with the n-gram order. In fact, BLEU-4 can drop to zero if the generated sentence and reference share fewer than four consecutive words. This is a pertinent issue in our task. Suppose if a model's output is very short or lacks longer phrases in common with the reference, the BLEU-4 score will be closer and closer to zero. In practice, because DeepSeek-7B-Chat [4] (one of the LLMs that we tested in our model) often outputs very brief descriptions in our experiment (sometimes only a couple of words), its BLEU-4 score is frequently minimal, underscoring this limitation of BLEU.

METEOR: METEOR (Metric for Evaluation of Translation with Explicit Ordering) [20] was introduced to address some of the weknessses of BLEU. To be specific, this was to be done by considering synonymy and morphological variations [20]. METEOR [20] aligns the generated text with the reference in a more flexible manner. This allows matches not only on exact words but also on stemmed forms and synonyms (for example, "run" vs. "running"). It then computes a harmonic mean of precision and recall of unigrams [20], with a higher weight for recall and a penalty for inadequate alignment. This means METEOR rewards a system for finding many of the reference words (recall), while also ensuring the generated words are mostly accurate (precision). Because of these features, METEOR [20] tends to correlate better with human judgments in many cases than raw BLEU [16]. Specially when the model uses different yet acceptable wording. Nevertheless, METEOR still fundamentally relies on lexical overlap. It improves over BLEU by catching some synonyms, but it will still penalize the omission of content words present in the reference. If the output is much shorter than the reference (missing many expected words), METEOR will drop significantly due to recall penalties.

ROUGE-L: ROUGE-L or Recall-Oriented Understudy for Gisting Evaluation [21] is a recalling focused metric. It was originally developed for the evaluation of automatic summarization [21]. ROUGE-L measures the length of the Longest Common Subsequence (LCS) [21] and compares them between the generated text and the sentences or captions. Rather than requiring contiguous words, it finds the longest word sequence that occurs in both texts in the same order (not necessarily consecutively). This effectively captures how much of the reference's content is covered by the output in order, which is

called focusing on recall of information [21]. A higher ROUGE-L score means that a larger portion of the reference's information (in terms of ordered words or phrases) is present in the generated sentence. Like BLEU and METEOR, ROUGE-L is also grounded in surface level overlap or lexical overlap of to some extent. It does not account for synonyms or rephrasings beyond exact 25 word matches. A shorter output will naturally have a shorter common subsequence (meaning shorter LCS) with the reference, and thus a lower ROUGE-L. In my case, if the model only produces a few words, whereas the reference is a longer sentence, the longest common subsequence might be just that one or two word, which ultimately will lead to a relatively limited ROUGE-L score. In my observation, among my two tested models (Mistral and DeepSeek), Mistral generates more expressive and longer outputs compared to DeepSeek, therefore, Mistral naturally has an edge over DeepSeek on this LCS based measure.

BERTScore: While BLEU [16], METEOR [20], and ROUGE-L [21] largely assess lexical overlap (exact or slightly flexible word matching), BERTScore [22] provides a complementary semantic evaluation. BERTScore [22] uses contextualised embeddings from a pretrained language model (BERT or similar) to compare the generated text and reference at the level of meaning [22]. In this approach each sentence is represented as a set of vector embeddings for its words (or tokens) and the metric computes a soft alignment between these sets to determine how much competent the model's outputs are in terms of meaning compared to the reference. Crucially BERTScore can detect semantic equivalence even when different words are used. For instance, "a man riding a bicycle" vs "a person on a bike" could score high on BERTScore despite having few exact words in common. This makes BERTScore [22] more tolerant to paraphrasing and variation in wording than the previous three metrics. It addresses cases where the model output captures the correct idea but with different vocabulary or phrasing. A high BERTScore indicates that the generated caption embeds to a similar vector space position as the reference caption, signifying strong semantic similarity.

Overall, by combining these four metrics we evaluate both the surface level accuracy *(did the output use the same words as the reference?)* and the semantic accuracy *(did the output convey the same meaning as the reference?).*

4 Results and Discussion

This chapter presents the quantitative and qualitative evaluation of our proposed architecture for EEG-to-Text generation. We compare our approach against the Thought2Text [12] framework across multiple evaluation metrics, demonstrating the effectiveness of incorporating Transformer layers for capturing global temporal dependencies in brain signals. We also evaluate the performance of multiple LLMs within our proposed framework. All evaluation metrics are presented as percentages for clarity.

4.1 Quantitative Analysis

We evaluated text generation quality using four established metrics. Each metric captures different aspects of similarity between generated and reference captions. We also measured the encoder's object classification accuracy as a direct assessment of its representational capacity. We integrated and tested our proposed encoder with two language models: Mistral-7B-Instruct [17] and DeepSeek-7B-Chat [4], evaluating whether architectural improvements generalize across different model backbones. **Figure 7** presents a comprehensive comparison across all metrics.

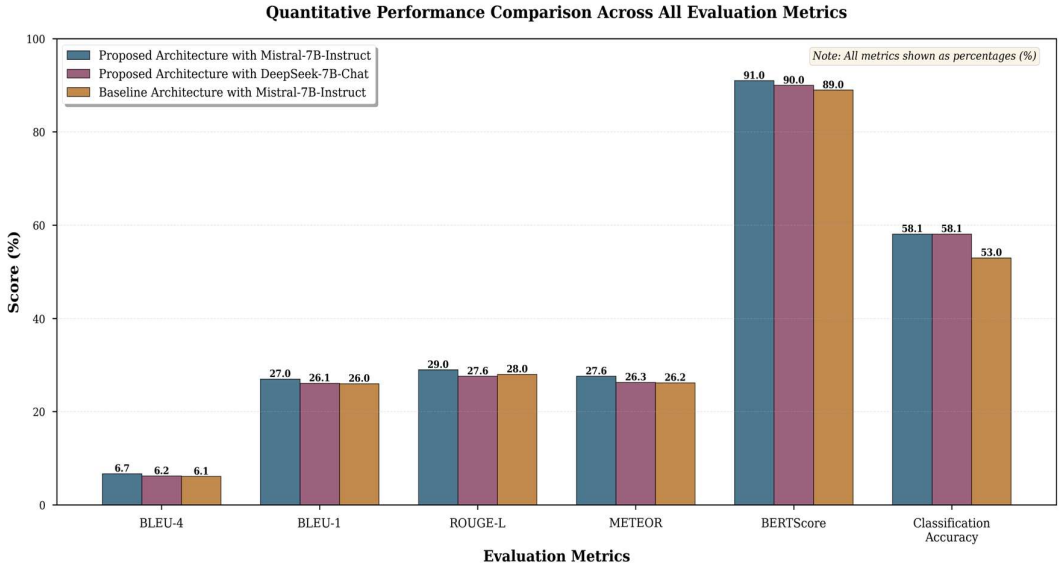

Figure 7: Quantitative performance comparison across all evaluation metrics (shown as percentages). The proposed enhanced CNN-Transformer architecture paired with both Mistral-7B-Instruct and DeepSeek-7B-Chat models is compared against the baseline [12] system. Note that the baseline scores are paired with Mistral-7B-Instruct.

Figure 7 shows that our architecture consistently outperforms the baseline across all metrics. The proposed architecutre configured with Mistral achieves the best overall performance with improvements in almost every metric, suggesting that Transformer layers genuinely enhance the encoder's ability to capture meaningful information from EEG signals.

4.1.1 BLEU-1 and BLEU-4 Scores

BLEU [16] scores measure lexical overlap between generated and reference texts. BLEU-1 counts matching individual words while BLEU-4 requires matching four word sequences. Our proposed model integrated with Mistral-7B-Instruct [17] achieves BLEU-1 of 27.0%, compared to 26.0% for the baseline. A relative improvement of 3.8%. While seemingly modest, this is meaningful given the inherent noise in EEG signals. Our mode with DeepSeek scores 26.1%, still above baseline despite its tendency toward shorter outputs. For BLEU-4, the proposed model integrated with Mistral scores 6.7% versus the baseline's 6.1%, representing a 9.8% relative improvement. Our architecture with DeepSeek achieves 6.2%, slightly exceeding the baseline. The BLEU-4 improvement is particularly significant. This metric easily drops to near zero for brief outputs or mismatched word sequences, even with correct meaning. The fact that our architecture improves BLEU-4, it suggests better phrase alignment.

4.1.2 ROUGE-L Scores

ROUGE-L [21] measures the longest common subsequence between texts, focusing on recall without requiring consecutive words. This flexibility better captures content coverage. The introduced with Mistral achieves 29.0% versus the baseline's 28.0%, a 3.6% relative improvement indicating better content coverage. The model includes more key information from reference captions, suggesting the enhanced encoder captures more complete stimulus information. The designed architecure with DeepSeek model scores 27.6%, slightly below baseline, explained by DeepSeek's very brief outputs that naturally limit subsequence length.

The ROUGE-L [21] improvement validates our architectural choice to add global context modeling. Local CNN features may miss sustained neural responses spanning the entire temporal window. Transformers ensure these extended patterns contribute to the final embedding, producing more comprehensive descriptions.

4.1.3 METEOR Scores

METEOR [20] considers synonyms and word stems while balancing precision and recall, making it more sophisticated than BLEU. The presented architecure with Mistral achieves 27.6% compared to approximately 26.2% for the baseline causing a 5.3% relative improvement. It is the largest gain among lexical overlap metrics. The adopted model with DeepSeek scores 26.3%, somewhat matching baseline performance.

The METEOR [20] score improvement is particularly significant because it captures both exact matches and semantic variations. Our model showing the best improvement on this metric (among the lexical overlap based metrics) suggests the enhanced encoder helps generate appropriate paraphrasing, not just memorized word patterns. This indicates that global context from Transformer layers helps the system understand the semantic concept behind visual stimuli.

4.1.4 BERTScore

BERTScore uses contextual embeddings from BERT to measure semantic similarity rather than counting word overlaps [22]. This makes it valuable for assessing whether descriptions convey the same meaning with different wording.

Our architecture shows strongest performance on BERTScore. Our designed model with Mistral achieves 91.0%, while integrated with DeepSeek, it reaches 90.0%. The baseline scores 89.0% on the same metric. These high scores indicate generally good semantic understanding across all systems, but the improvements remain meaningful, that is 2.2% and 1.1% relative improvement respectively.

These BERTScore [22] improvements provide compelling evidence that our architectural modifications enhance semantic understanding for decoding brainwaves to text. The metric's tolerance to paraphrasing means it specifically measures conceptual accuracy, not just word matching. Consistent improvement across both language models in our expirement, suggests the enhanced encoder provides better semantic representations regardless of decoder characteristics. **Figure 8** provides a focused view of architectural impact.

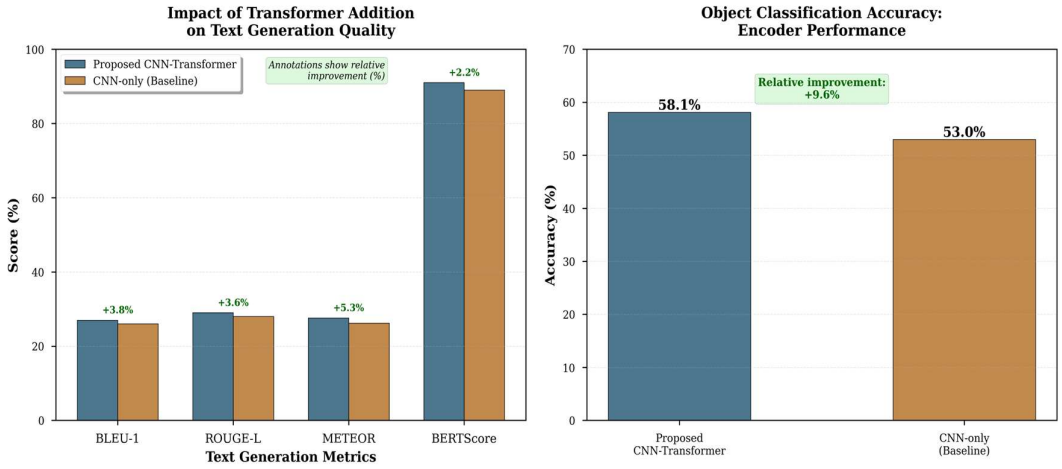

Figure 8: Architectural impact of adding Transformer layers in the encoder. Left panel shows text generation metrics with annotations indicating relative improvement percentages over baseline. Right panel compares classification accuracy.

4.1.5 Object Classification Accuracy

Beyond text generation, we measured encoder classification accuracy as a direct assessment of how well it extracts stimulus information from brain signals before language model involvement. Both of our architectures achieve identical 58.1% (**Figure 7**) overall classification accuracy versus 53.0% overall for baseline [12] (**Figure 7** and **Figure 8**). It is a substantial 9.6% relative improvement. This is highly significant in EEG classification where even small gains require considerable architectural advances. This improvement directly validates our hypothesis that Transformer layers enhance representational capacity. The ability to model global dependencies allows better discrimination between object categories based on subtle brain activity patterns. Brain responses vary in both local features (quick recognition) and global patterns (sustained attention, semantic processing). The CNN-only baseline captures local features well but may miss global patterns.

The identical performance of both our experimented model configurations (58.1%) confirms that encoder quality is independent of the subsequent language model, which makes architectural sense since classification occurs entirely within the encoder. The 9.6% relative improvement also explains text generation gains. Meaning better classification means more discriminative embeddings that provide the language model

with clearer input about visual concepts, thus it ultimately improves in terms of the quatitative metrics as well.

4.2 Qualitative Analysis

Table 1: Representative samples comparing generated descriptions across models. The proposed architecture captures more specific object attributes and demonstrates improved semantic understanding regardless of chosen language model compared to the baseline. Reference captions are shown in black.

Images	Reference Captions	Baseline + Mistral	Proposed + Mistral	Proposed + DeepSeek
	A black and gold grand piano with the Boston Piano Company logo.	A black and white piano with a microphone in front of it.	A black grand piano with a black and white keyboard.	A grand piano with a black and white finish.
	A large yellow mushroom with a brown stem and a brown cap, surrounded by green foliage.	A group of mushrooms growing on a log.	A colorful, large mushroom with a brown cap and white spots.	A small brown mushroom growing on a log.
	A pair of handmade, knitted gloves with a mix of brown, orange, and black yarn.	A pair of knitted gloves with a white background.	A pair of black leather gloves with a white stripe on the back.	Black, knitted gloves with white trim.
	A hand holding a mug with a blue background and a handprint design.	A person holding a coffee mug with the words “World’s Best Dad” written on it.	A white coffee mug with a white handle and the words “I’m not a morning person” written on it.	A white coffee mug and the words “World’s Best Dad” written on it.

While quantitative metrics provide objective performance measures, examining actual generated descriptions reveals how the models differ in practice. **Table 1** presents representative examples from the test set. It compares outputs from the baseline [12] system and our proposed architectures.

4.2.1 Descriptive Detail and Attribute Capture

Table 1 shows a clear pattern: the architecture introduced in this work gives more detailed and accurate descriptions than the baseline. For example, in the *"piano"* image, the baseline says, *"A black and white piano with a microphone in front of it,"* even though no *"microphone"* is actually present. In contrast, out model integrated with Mistral-7B-Instruct [17] more accuretly describes *"A black grand piano with a black and white keyboard,"* avoiding false details. This suggests that the improved encoder extracts visual features from brain signals more reliably.

The *"mushroom"* example shows the same trend. The baseline gives a very general caption, *"A group of mushrooms growing on a log."* While technically correct, it ignores key visual features. The proposed Mistral version adds specific traits such as *"colorful," "large,"* and *"white spots,"* indicating that the encoder is capturing more fine grained information from the EEG. This aligns with our 9.6% relative boost in classification accuracy. Better object discrimination naturally leads to more detailed descriptions.

The introduced model integrated with DeepSeek-7B-Chat [4] behaves slightly differently. Its outputs are much shorter, such as *"A small brown mushroom growing on a log."* Although concise, it still includes the essential attributes. This shorter style explains why DeepSeek performs worse on lexical overlap metrics like BLEU-4 [16] and ROUGE-L [21], even though it uses the same improved encoder. This difference comes from the language model's generation style, not the encoder itself.

4.2.2 Semantic Accuracy and Contextual Understanding

The examples also show that our architecture improved in semantic understanding, not just descriptive detail. In the *"coffee mug"* case, the reference caption mentions a hand holding a mug with a handprint design. The baseline identifies the main elements but mistakes the handprint for text. The adopted model's Mistral output, *"A white coffee mug with a white handle and the words "I'm not a morning person" written on it.,"* is still not a perfect match, but it shows the model is trying to interpret both visual text and the overall scene.

The *"glove"* example shows a similar pattern. The reference describes handmade gloves with mixed colors. The baseline reduces this to *"knitted gloves with a white background,"*

ignoring the color details. The proposed framework with Mistral model instead gives *"black, leather gloves with a white stripe,"* which gets the color pattern right and identifies a specific feature, even though the material is incorrect. This suggests the encoder is capturing meaningful structural information from the EEG signals, even if the language model occasionally misinterprets it.

These observations match our 2.2% improvement in BERTScore [22]. Since, BERTScore measures semantic similarity [22], the slightly higher score reflects that our model captures the main meaning of objects, even when using different words.

4.2.3 Connecting Qualitative Observations to Proposed Methodology

The qualitative results directly reflect how our architecture processes EEG signals. When someone looks at an image, the brain responds over time; early activity represents simple visual features like shapes and colors, while later activity represents object meaning. The baseline CNN encoder can capture these fast, local patterns. But its limited receptive field makes it difficult to combine information across the full 440 ms window [12, 18, 19]. The examples in **Table 1** show this limitation. Generic outputs such as *"a group of mushrooms"* or *"knitted gloves with a white background"* suggest that the baseline mostly captured object category but missed richer details. Our introduced enhanced encoder fixes this by adding Transformer layers that apply "self-attention" across the entire sequence. This lets the model link early visual responses with later semantic signals, resulting in embeddings that represent both the object type and its specific attributes.

The *"piano"* example highlights this effect clearly. Identifying a *"grand piano"* instead of just *"a piano"* requires combining information spread over time, which the Transformer's global attention [10] supports. This is why the proposed model produces the more precise term while the baseline does not in this case.

Both our model versions (integrated Mistral [17] and DeepSeek [4]) show similar improvements in detail and accuracy, even though DeepSeek-7B-Chat [4] tends to generate shorter outputs. This supports the quantitative findings as well.

These qualitative observations also explain the metric gains. A 3.8% relative improvement in BLEU-1 [16] may look small, but the examples show that it leads to

clearer object attributes, fewer hallucinations, and more coherent descriptions. Likewise, the 9.6% relative increase in classification accuracy appears in outputs that correctly identify object types and their defining features. Even modest numerical gains can produce meaningful improvements in how well the generated text reflects the information encoded in brain signals.

5 Conclusion and Future Work

5.1 Summary of Contributions

This thesis presents a comprehensive exploration of decoding brainwaves into words by integrating large language models with neural networks. It introducdes a framework that connects low level brain signals to latest language generation techniques. The core of the work is a three stage translation task that begins with EEG encoding, followed by cross modal semantic alignment, and ends with prompt based language generation.

A key innovation of this work is the enhanced encoder architecture that combines the local feature extraction of the ChannelNet CNN encoder [18] with the global modeling ability of Transformers [10]. While the CNN captures short, rapid patterns, its limited receptive field can miss information spread across the 440 ms window. Adding Transformer layers after the CNN allows the model to learn global relationships through "self-attention" [10]. It connects early visual signals with later semantic ones and creating richer embeddings that better represent object categories and attributes.

This architectural change leads to clear performance gains. Classification accuracy rises relatively by 9.6% (from 53.0% to 58.1%), showing that the encoder learns more discriminative EEG features. These improvements carry over to text generation, with consistent metric gains. The benefits apply to both language models that we tested, Mistral-7B-Instruct [17] and DeepSeek-7B-Chat [4]. It goes to validate that the encoder upgrade works well regardless of the language model used.

To evaluate the system, the thesis uses standard NLP metrics (BLEU, METEOR, ROUGE-L, and BERTScore) [16, 20, 21, 22] along with qualitative analysis. The metrics improve across the board indicating better semantic alignment. The qualitative results further show that the designed model generates more detailed and accurate descriptions with fewer hallucinated elements. Comparing Mistral-7B-Instruct [17] and DeepSeek-7B-Chat [4] also highlights the difference between detailed vs. concise generation styles while confirming that both benefit from the stronger encoder.

5.2 Limitations and Future Directions

This thesis came across some limitations that also point to promising directions for future work. One of the biggest challenges comes from the nature of non-invasive EEG recordings. EEG has a low signal-to-noise ratio and only provides an indirect view of brain activity. This makes it difficult to extract precise semantic information. In our experiments, the system could reliably capture broad features, main objects or general colors; but it often failed to pick up finer details. Improving this may require combining EEG with other techniques like MEG or using higher density electrode setups that can record richer neural signals.

Differences between participants also remain a major issue. EEG patterns vary widely from person to person and can even shift across sessions for the same individual. As a result, a model trained on one subject may not transfer well to another without extra calibration. Future studies should look into more generalizable, subject independent models that perform consistently across users. This will be especially important if such systems are to be used outside research settings.

Data availability is perhaps the most impactful constraint. Very few public EEG-to-Text datasets exist, and the dataset used in this thesis includes a limited number of images and participants. Larger and more diverse datasets which are ideally covering multiple languages and more subjects, would benefit both this work and the broader research community. They would allow for training more robust models that can better capture the complexity of translating brain signals into text.

Despite these challenges, this thesis shows that decoding brainwaves into words EEG is possible. The architectural choices and evaluation methods developed here form a solid base for future progress. By improving signal quality, model generalization, dataset scale, and practical usability, decoding brainwaves and EEG-to-Text generation can move closer to real world applications in communication support and brain-computer interfaces.

6 Publication

[1] Mohammed Salah Al-Radhi, Sadi Mahmud Shurid, Géza Németh, “Prompting the Mind: EEG-to-Text Translation with Multimodal LLMs and Semantic Control,” *27th International Conference on Speech and Computer (SPECOM). Lecture Notes in Computer Science*, Szeged, Hungary, pp. 52–66, 2025.doi: https://doi.org/10.1007/978-3-032-07956-5_4.

[2] Sadi Mahmud Shurid, “Semantically Controlled EEG-to-Text Translation Using Deep Learning and LLMs,” *TDK Conference, 2025.* **(3rd Prize)**

References

[1] J. G. Makin, D. A. Moses, and E. F. Chang, "Machine translation of cortical activity to text with an encoder–decoder framework," *Nature Neuroscience*, vol. 23, no. 4, pp. 575–582, Mar. 2020, doi: 10.1038/s41593-020-0608-8.

[2] A. Kamble, P. H. Ghare, and V. Kumar, "Classifying Phonological Categories and Imagined Words from EEG Signal," *Biomedical Signal Processing for Healthcare Applications*, pp. 93–121, May 2021, doi: https://doi.org/10.1201/9781003147817-5.

[3] C. Herff and T. Schultz, "Automatic Speech Recognition from Neural Signals: A Focused Review," *Frontiers in Neuroscience*, vol. 10, Sep. 2016, doi: https://doi.org/10.3389/fnins.2016.00429.

[4] "deepseek-ai/deepseek-llm-7b-chat · Hugging Face," *Huggingface.co*, Aug. 16, 2024. https://huggingface.co/deepseek-ai/deepseek-llm-7b-chat.

[5] M. Marino and D. Mantini, "Human brain imaging with high-density electroencephalography: Techniques and applications," *The Journal of Physiology*, Aug. 2024, doi: 10.1113/jp286639.

[6] L. A. Farwell and E. Donchin, "Talking off the top of your head: toward a mental prosthesis utilizing event-related brain potentials," *Electroencephalography and Clinical Neurophysiology*, vol. 70, no. 6, pp. 510–523, Dec. 1988, doi: 10.1016/0013-4694(88)90149-6.

[7] F. R. Willett, D. T. Avansino, L. R. Hochberg, J. M. Henderson, and K. V. Shenoy, "High-performance brain-to-text communication via handwriting," *Nature*, vol. 593, no. 7858, pp. 249–254, May 2021, doi: 10.1038/s41586-021-03506-2.

[8] A. Défossez, C. Caucheteux, J. Rapin, O. Kabeli, and J.-R. King, "Decoding speech perception from non-invasive brain recordings," *Nature Machine Intelligence*, vol. 5, no. 10, pp. 1097–1107, Oct. 2023, doi: 10.1038/s42256-023-00714-5.

[9] Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition," in *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278-2324, Nov. 1998, doi: 10.1109/5.726791.

[10] A. Vaswani et al., "Attention Is All You Need," arXiv.org, 2017. https://arxiv.org/abs/1706.03762

[11] V. J. Lawhern, A. J. Solon, N. R. Waytowich, S. M. Gordon, C. P. Hung, and B. J. Lance, "EEGNet: a compact convolutional neural network for EEG-based brain computer interfaces," *Journal of Neural Engineering,* vol. 15, no. 5, p. 056013, Jul. 2018, doi: https://doi.org/10.1088/1741-2552/aace8c.

[12] A. Mishra, S. Shukla, J. Torres, J. Gwizdka, and S. Roychowdhury, "Thought2Text: Text Generation from EEG Signal using Large Language Models (LLMs)," in *Findings of the Association for Computational Linguistics: NAACL 2025,* Albuquerque, New Mexico, Apr. 2025, pp. 3747–3759. doi: 10.18653/v1/2025.findings-naacl.207.

[13] W. Liao, H. Liu, and W. Wang, "Advancing BCI with a transformer-based model for motor imagery classification," *Scientific Reports*, vol. 15, no. 1, Jul. 2025, doi: https://doi.org/10.1038/s41598-025-06364-4.

[14] M. A. Pfeffer, S. Sai, and J. Kwok, "Exploring the frontier: Transformer-based models in EEG signal analysis for brain-computer interfaces," *Computers in Biology and Medicine*, vol. 178, pp. 108705–108705, Aug. 2024, doi: https://doi.org/10.1016/j.compbiomed.2024.108705.

[15] A. Q. Jiang et al., "Mistral 7B," arXiv.org, Oct. 10, 2023. https://arxiv.org/abs/2310.06825

[16] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "BLEU: a Method for Automatic Evaluation of Machine Translation," *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics* - ACL '02, 2002, doi: https://doi.org/10.3115/1073083.1073135.

[17] "mistralai/Mistral-7B-Instruct-v0.3 · Hugging Face," huggingface.co. https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.3

[18] S. Palazzo, C. Spampinato, I. Kavasidis, D. Giordano, J. Schmidt and M. Shah, "Decoding Brain Representations by Multimodal Learning of Neural Activity and Visual Features," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 11, pp. 3833-3849, 1 Nov. 2021, doi: 10.1109/TPAMI.2020.2995909.

[19] C. Spampinato, S. Palazzo, I. Kavasidis, D. Giordano, M. Shah, and N. Souly, "Deep Learning Human Mind for Automated Visual Classification," arXiv (Cornell University), Jan. 2016, doi: https://doi.org/10.48550/arxiv.1609.00344.

[20] S. Banerjee and A. Lavie, "METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments," ACLWeb, Jun. 01, 2005. https://aclanthology.org/W05-0909/

[21] C.-Y. Lin, "ROUGE: A Package for Automatic Evaluation of Summaries," aclanthology.org, Jul. 01, 2004. https://aclanthology.org/W04-1013/

[22] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi, "BERTScore: Evaluating Text Generation with BERT," arXiv:1904.09675 [cs], Feb. 2020, Available: https://arxiv.org/abs/1904.09675

Annex

I. Declaration on the Use of Generative Artificial Intelligence

☐ **I have not used** any generative AI tools.

■ **I have used generative AI tools.** I have verified the content generated by AI, ensured the accuracy of the outputs, and properly indicated each instance of use in the table below.

Usage type	**Name of Generative AI Tool(s)**	**Affected Sections** (chapter, page number, reference)	**Estimated Proportion of Use** (per usage type)
Literature Review	ChatGPT	Sections 2.1, 2.2, pages 10-14	8-10%
Brief Summary of the Prompt	Used the deep research feature to find out and study about the recent developments in neural approaches for EEG-to-Text translation.		
Program Code Generation			
Brief Summary of the Prompt			
Generating New Ideas or Solution Proposals			
Brief Summary of the Prompt			
Creating an Outline (text structure, bullet points)			
Brief Summary of the Prompt			
Creating Text Blocks			
Brief Summary of the Prompt			
Generating Images for Illustrative Purposes	Claude	Figure 4, page 21	12-15%
Brief Summary of the Prompt	Took help to draft the Mermaid code for draw.io from my rough drawing.		

Data Visualization, Generating Charts Based on Data Points	Claude	Figure 7, Figure 8; page 32, 35	12-15%
Brief Summary of the Prompt	Generate Python matplotlib code for creating grouped bar charts comparing multiple models across evaluation metrics with proper labeling and annotations.		
Preparing a Presentation			
Brief Summary of the Prompt			
Other (please specify) Grammar refinement and clarity improvement of own expressions	Google Gemini, ChatGPT	Various sections for grammar refinement and clarity improvement throughout Chapters 3-4.	5-7%
Brief Summary of the Prompt	Refine technical explanation for clarity while maintaining academic tone for the following section; check grammar and improve sentence structure for better understaning.		
Aggregated Percentage Value (for the core part of the task)			**9-12%**

Brief Textual Justification of the Aggregated Value:

I have used generative AI tools to help with small, technical tasks, not to create any of the main research content. For the literature review, AI helped me pull together and organize ideas from existing papers, but I personally checked everything against the original sources. When creating data visualizations (Figure 7 and 8), I used AI to draft some basic Python templates for the comparison charts. However, I had to adapt, populate using my own results and refine the style as per my requirements and liking. For the diagram in Figure 4, I first drew a draft of it, then I used generative AI to prepare a mermaid outline based on the draft, after that I had to heavily edit it so it would correctly reflect the architecture I developed. I also used AI for minor grammar and sentence structure cleanup. All of the technical experiments, implementing the methods, analyzing the results, and drawing conclusions are my own. I reviewed and fact checked every AI generated piece to make sure it was accurate and aligned with what I was trying to achieve.