Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

A Novel Temporal Variability-Aware Architecture for Ictal-Interictal Continuum Patterns Classification

zaib unnisa

zaib.unnisa82@yahoo.com

Superior University https://orcid.org/0000-0002-2533-2576

Sohail Masood Bhatti

Superior University

Arfan Jaffar

Superior University

Umair Muneer Butt

University of Management and Technology

Anees Baqir

Northeastern University London

Research Article

Keywords: Harmful Brain Activities, EfficientNet, Deep Learning, Brain signals

Posted Date: November 20th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11788323/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: The authors declare no competing interests.

A Novel Temporal Variability-Aware Architecture for EEG and MRI Analysis, EfficientNetZEEG

Authors: Zaib Unnisa, Sohail Masood Bhatti, Arfan Jaffar,

Abstract

Harmful brain activity may compromise cerebral function and can culminate in mortality if not addressed appropriately. The global deficit of professionals adept in electroencephalogram (EEG) data analysis exacerbates the difficulty in resolving this issue. There is a pressing need for a reliable and automated system, particularly designed to effectively classify harmful brain activity using both signals and images. In this study, we proposed a novel model that can be used for signal as well as image classification. This study employs EEG recordings obtained from the HMS dataset and MRI scans from the Brain Tumor MRI dataset. The EEG dataset comprises 106,800 unique EEG identification numbers. Our proposed model, EfficientNetZEEGv1, with instability-aware gated TSA, demonstrated exceptional performance by achieving a validation accuracy of 97% and a loss of 0.20. We made some enhancements in the gating mechanism of the model, and our second model EfficientNetZEEGv2, achieved a validation accuracy of 98.74%. We have achieved the highest accuracy, precision, and recall values amongst all studies conducted in the domain of harmful brain activities.

1. Introduction

Cerebral activities are utilized to diagnose a range of medical conditions. These cerebral activities may be identified through the acquisition of brain signals. The methodology employed to capture brain signals designates the nomenclature of the signals; specifically, electroencephalography (EEG) signals represent the cerebral signals obtained by positioning electrodes on the scalp [1]. Harmful cerebral activities can be categorized into two extremes: ictal, which refers to seizures, and non-ictal, indicating the absence of seizures. The intermediary region between ictal and non-ictal states encompasses ambiguous EEG patterns resulting from rhythmic and periodic discharges. It is called ictal-interictal continuum (IIC), which is closely associated with seizures [2]. IIC have patterns of ongoing seizures but cannot be categorized as seizures [3]. During the ongoing monitoring of EEGs in critically ill patients, rhythmic and periodic discharges are observed in over one-third of the cases of patients [4].

This situation worsens in the case of patients suffering from coma. There is a dire need to classify these signals. only a few studies are found in the literature, and those studies didn't achieve any remarkable results, i.e., 69% [5], and 82% [6] accuracies; therefore, they cannot be used in clinical settings. Dual modality of the data is never discussed for seizures. It is needed to work on the dual modalities. So, AI models need not to switch between the models, while making predictions. While saving a person every second counts. An AI model that can work on the dual modality can be a big plus in the research domain of seizure and seizure like patterns. In order to show the novelty of the proposed model, we have compared the models' architectural aspects with the literature in Table 4.1. The table shows that the proposed pipeline, MFCC, and deltas with a modified architecture of Efficient Net, i.e., Efficient Net with TSA block, in never been proposed in the literature earlier for seizure classification tasks, and it is an original and novel approach.

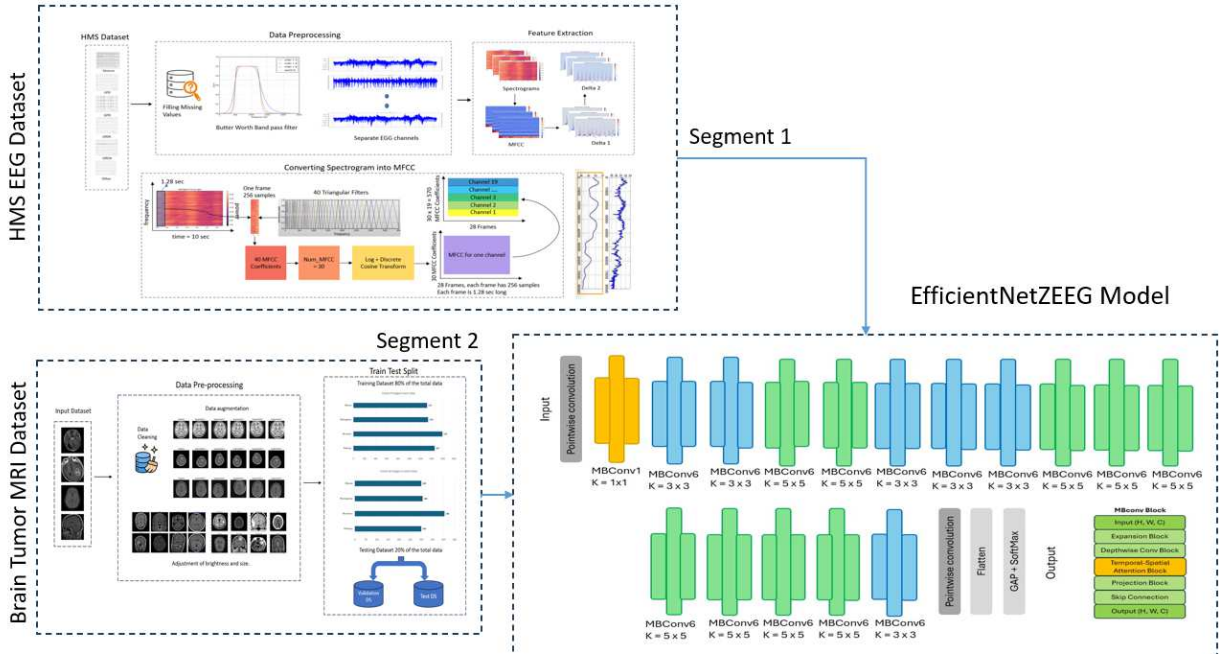


Figure 1: Workflow of current study

Table 1. Comparison of proposed model's (for RO3) pipeline with the literature for novelty.

Authors	Efficient Net base	SE blocks modified	Temporal attention used	Spatial attention used	EEG task	Used MFCC and deltas	Feature extraction and input	Ref
Sadoon et al.	EfficinetNetb0	No	No	No	Seizure Prediction	No	STFT-based spectral feature + channel selection	[7]
Sun et al.	Efficient Net (2D CNN)	SE with spatial attention layer	No	Yes	Seizure detection	No	Time frequency feature maps (2D) via decomposition + handcrafter features.	[8]
Hu et al.	Used custom models	N/A	Yes	Yes	Emotion recognition	No	Derived spatio-temporal representation (graph features, raw waveforms)	[9]
Proposed model, EfficientNetZEEG	EfficientNetB3	Yes	Yes	Yes	Harmful Brain activity classification/ seizure and seizure like patterns	Yes	MFCC and deltas	N/A

Novel Contributions

- EfficientNet inspired 1D architecture for MFCC-based EEG representation learning.
- Hybrid Temporal Self-Attention combining local convolutional attention and global multi-head attention.
- Temporal Instability-Aware Adaptive Gating that dynamically modulates local and global attention pathways based on feature-space variability.
- Integration of the above components into a unified framework for harmful brain activity classification.

2. Proposed Methodology

In this section we proposed a novel model for Harmful brain activity classification by adding a novel temporal and spatial attention block in the Efficient Net architecture, for EEG signals' processing. Multimodal evaluation is often considered a strong indicator of generalizability and robustness. Used two modalities of datasets, EEG signals and an MRI image dataset, to show the robustness of the proposed model. Research has shown that most of the critically ill patients suffer from harmful brain activities, and in that situation, every second counts. Switching between the AI models and loading data can be time-consuming and can cause loss of lives. Our proposed model can be used for seizure detection, cancer detection, and tumor prediction, and thus can be very beneficial in clinical settings. Our proposed model achieved the best accuracy, precision, and recall values in the literature in the domain of Harmful brain activity classification (HBAC) using the HMS dataset. According to the research, a better loss value indicates a better certainty score [10].

In this section proposed methodology of the current study will be presented, which includes dataset description, pre-processing, feature extraction, feature selection, and classification method. EEG signals are the brain signals that are captured by placing electrodes on the skull. This is a non-invasive procedure by which brain signals can be captured, and it has been very popular in research for a long time [11]. Different portions of the brain are capable of different activities. By placing electrodes on those portions, physicians can find the abnormalities or various disorders. For example, Parkinson's disease can be found by placing electrodes on the frontal, central, and occipital lobes. Depression can be detected by placing electrodes on the frontal and central lobes. EEG signals are also used for cancer detection. In this study, we are focusing on using EEG signals to classify harmful brain activities. We conducted this study in two segments, so we can use the datasets of the two modalities regarding harmful brain activities. Workflow of this study is shown in Figure 1.

2.1. Datasets

Two datasets are used in this research study. First one is the HMS dataset that contains EEG signals and the second one is Brain tumor dataset, that contains MRI images of brain tumor.

2.1.1. HMS Dataset

In this research study we have used the best features that are extracted from the HMS dataset in the study [12]. HMS dataset consists of six classes, namely lateralized Periodic Discharge (LPD), generalized Periodic discharge (GPD), lateralized Rhythmic Delta Activity (LRDA), generalized Rhythmic delta activity (GRDA), seizure (SZ), and "other". Current study is an extension of the research study [12]. Therefore, we have used the same data cleaning, preprocessing, feature extraction methods as used in study [12]. However, in this study instead of using the best features, we have used second best features i.e., MFCC to reduce time complexity of the model.

2.1.2. Brain Tumor Dataset

For the MRI images, we have employed brain tumor (BT) dataset from Kaggle, that is a public data repository. BT dataset consists of four classes, i.e., glioma, meningioma, no tumor, and pituitary. There are in total 7023 MRI images. There are 5712 images in the training and 656 images in both validation and test datasets. In the preprocessing step, data augmentation is done using image flipping, rotation, shearing, resizing. These augmentations reduce overfitting and enhance model robustness.

2.2. EfficientNetZEEGv1

In this model, we used a modified version of EfficientNetB3, as shown in Figure 2. The detail about the modified MBConvBlock is presented here:

Expansion Block: This block expands the number of channels. In the first MBConv block, the expansion ratio = 1. In that case, the expansion block will be skipped. In the later MBblocks, i.e., MBBlock6, the expansion ratio is six. In that case, the expansion block expands the number of channels by multiplying the expansion ratio by the number of filters. In the expansion block, a filter of size one is used, thus point-wise convolution is achieved. There will be no change in the dimensions of H and W. Batch normalization helps in reducing overfitting, even though it is not a typical method to reduce overfitting (like dropout). In the last layer swish activation function is used. after batch normalization, the output is centered at 0. By using swish, non-linearity is introduced. Non-linearity is the main component of the excellent performance of deep learning networks. A linear function means that the output is proportional to the input. The functions follow a straight line. When a function follows a straight line, it cannot learn the complex patterns that need bent function lines (having many curves) for learning. Swish introduces non-linearity and thus helps the function to learn complex patterns. The Conv function is linear; swish introduces the nonlinearity into it. If there is no non-linear function, even the deep learning model behaves like a linear model.

Depth-wise Conv block: It is used in the EfficientNetB3 and other variants of Efficient Net to reduce computational cost. The conventional Conv network works on all the input channels and then sums up the output. In the depth wise Conv, each filter is applied to only one channel. Therefore, there will be less number of parameters, as well as less number of weights to work with. It dramatically reduces computational cost.

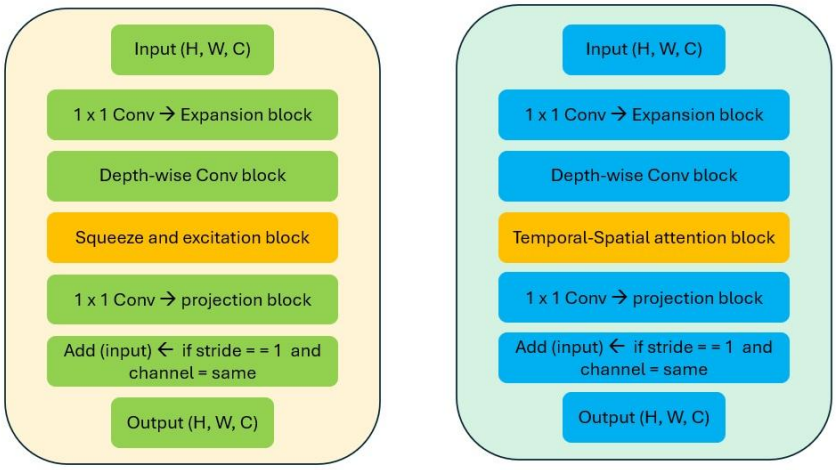


Figure 2: Difference between the proposed model, EfficientNetZEEGv1, and the original model of EfficientNetB3

SE vs TSA block: The layers in the SE block are shown in Figure 4.4. The goal of the SE block is to learn channel-wise weights. It tells the network which weights are important and which are not. It amplifies the important weights. At first, it reduces the number of channels from C to C * se ratio. Say if C = 32, and se ratio is 0.25 then C*se ratio will be equal to 8. The output will have 8 channels. It forces the model to learn channel-wise descriptors or information. If there are 32 channels, then there will be 32 neurons in the global average pooling vector. Each neuron's value will show the significance of the channel. By using it, a network can learn the importance of the channels, so it can enhance the importance of important channels and suppress the importance of less important ones. After compressing the channel descriptors, they are expanded back to the original number of channels. In the last step, the learned weights are broadcast across the entire map x, so the important channels are given more importance than the unimportant ones.

Primary Difference: In the case of EEG signals, we have only one channel, and we don't want to suppress the information, as we have already learned the important information by extracting delta features. we don't need to find the important channels. Therefore, we have replaced the SE block with TSA block in our proposed model. All the other blocks are used in the same way as they are in the original EfficientNetB3 model.

Projection Block: In the projection block, pointwise convolution is being performed. In a pointwise convolution, the kernel has a size of 1 x 1. With this size, it spans all input channels. In the first experiment, we have only one input channel. So, it won't mix the information across all the channels.

Skip Connection: It will be applied since both conditions are met, strides =1, and input filters = filter out. Some important layers of the EfficientNetZEEG model are described below.

Batch Normalization: It is a method for normalizing the input of each layer; the training speed, stability, and performance of the network can be improved. It computes the mean and variance of the activations and normalizes them so they have zero mean and a variance of 1. It promotes faster training by using higher learning rates, without instability. It acts as a regularizer and adds noise to the data during training. It is also used for preventing vanishing gradient by keeping activations in range. It keeps the activations in range by using scale and shift parameters. Due to noise, the model doesn't rely on the initialization of neurons. Sometimes, poor initialization may lead to poor results. Batch normalization fixes this issue.

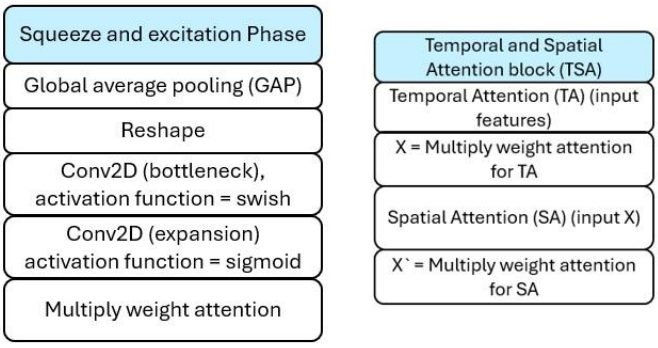


Figure 4. Layers in the SE block of the original EfficientNetB3 vs layers in the simple TSA block in EfficientNetZEEGv1.

The overall model structure is shown in Figure 5.

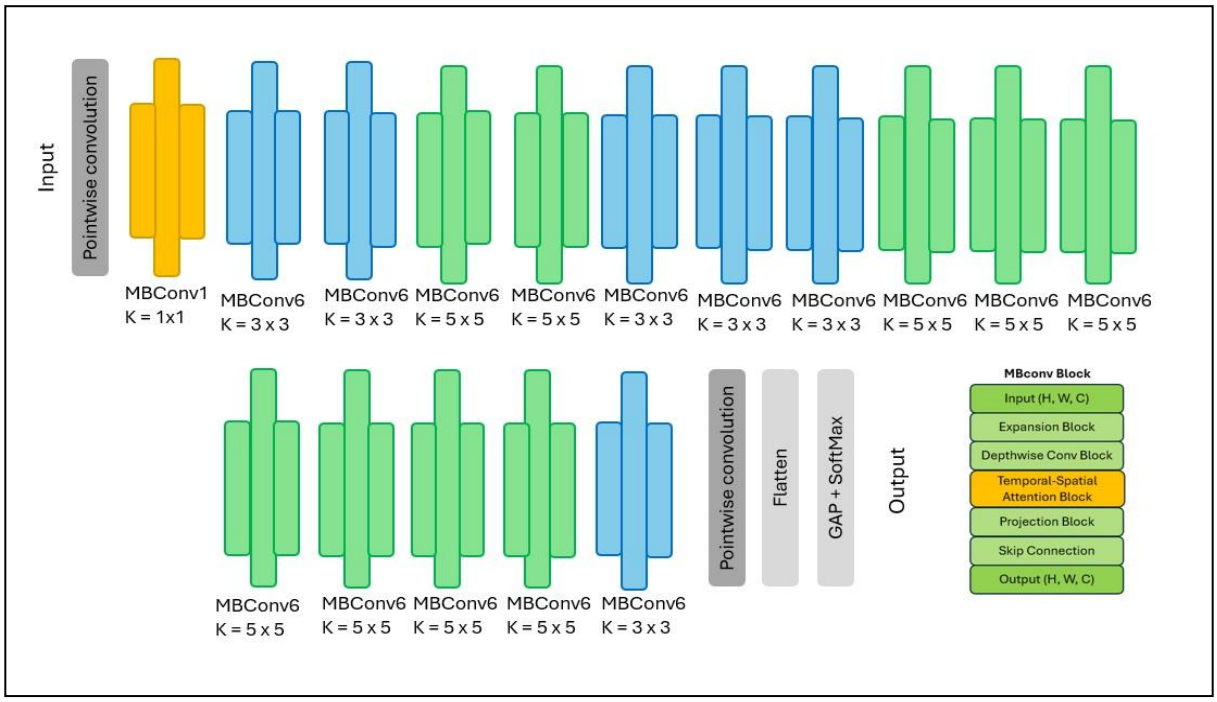


Figure 5: Architectural diagram of the proposed model, EfficientNetZEEG

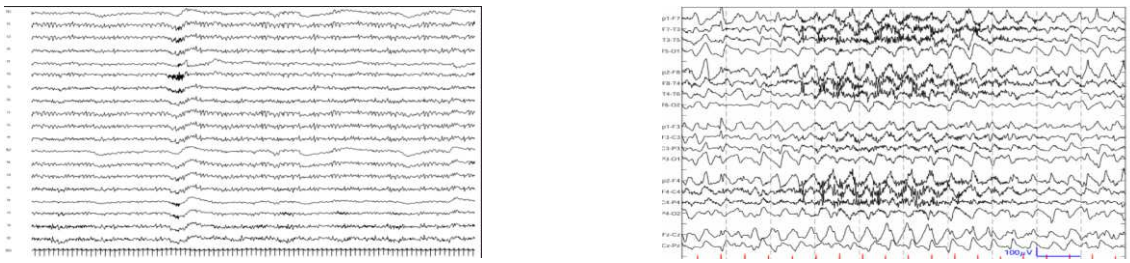


Figure 6: Figure in the left is EEG signals of GRDA, having low energy signals. Figure on the right is EEG signals of seizures, having very high energy contents.

Some signals have low energy and some have high, as shown in Figure 6. If we used the same TSA for all types of signals. It caused misclassification of the signals. So, we introduced a 3-tier architecture (three TSAs). TSA is chosen based on the energy content of the signal.

TSA is divided into 3 tiers. Simple, local, and the third one is both local and global. The choice of the TSA is based in value of "g".

Temporal Instability-Aware Adaptive Gating (TIAG): To the best of my knowledge, existing MFCC-based EEG classification methods primarily focus on feature extraction and classification architectures. The proposed work introduces an adaptive gating mechanism that estimates temporal instability from learned MFCC feature representations and dynamically adjusts the balance between local and global temporal attention. "This combination has not been widely explored in MFCC-based EEG classification." From the EEG signals MFCC features are extracted. These features are used in the EfficientNetZEEG architecture. From the feature data, temporal instability estimation is calculated. That is further used for adaptive gating.

A batch of size 64 is used, it means about 64 windows are processed in one training step. For each sample in the batch, the temporal instability score "g" is computed. So it can be found if local TSA branch is needed or the global one. In order to compute "g" at first squared energy differences between the two consecutive temporal frames is computed. The average squared difference gives the temporal instability energy for each sample, as shown in Eq. (1).

$$e_i = 1/(T-1)C \quad t=2\sum T \quad c=1\sum C \, (xi,t,c - xi,t-1,c)^2 \quad (1$$

The instability energy is then normalized across the batch using the batch mean μ_e and standard deviation σ_e, as shown in Eq. (2).

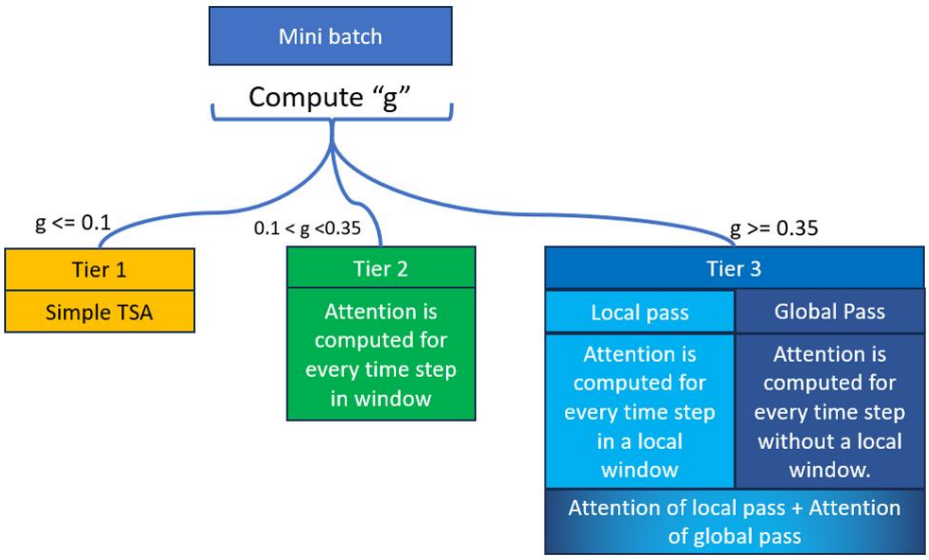
$$z_i = (e_i - \mu_e)/ (\sigma_e+\epsilon) \qquad (2$$

Finally, the normalized instability measure is passed through a sigmoid activation function to obtain the gating score g_i, as shown in Eq. (3).

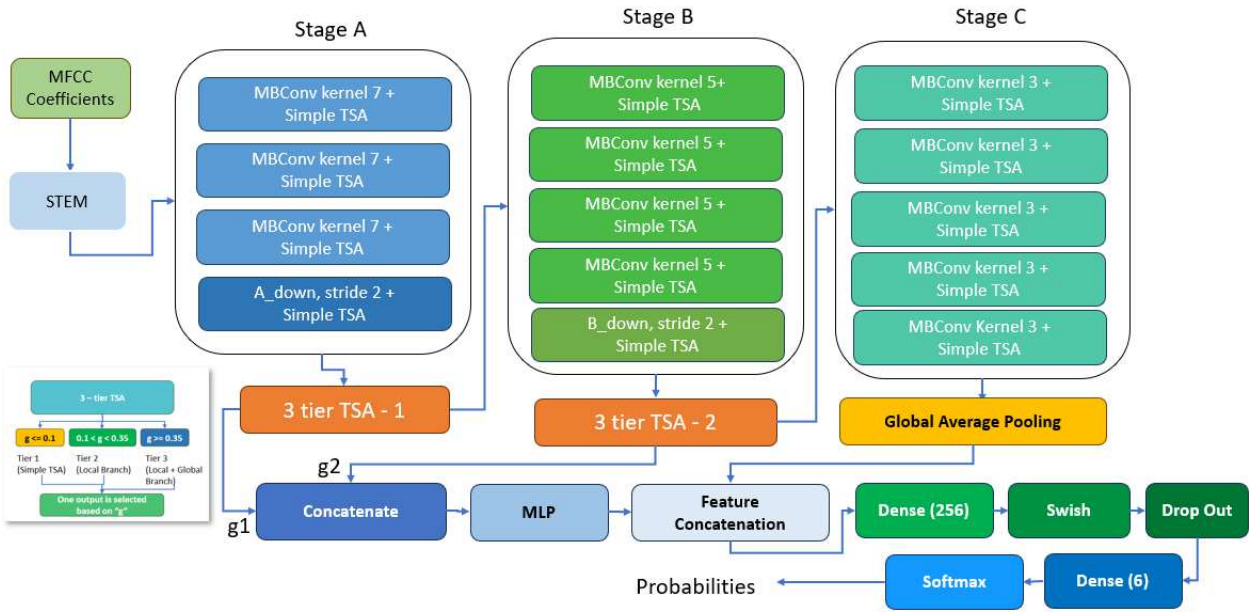
$$g_i = 1/(1+e^{-zi}) \qquad (3$$

T are the total number of time steps in the EEG window. Small t is the time index. In case of normal brain activity $x_t - x_{t-1}$ is smooth, and in case of variability there will be a sudden jump. We cannot compute $x_1 - x_0$, that is why the summation starts from x =2. Low g means smooth signal, and high g means there is a possible high transition. TSA is used in place of SE block, in each MBConvBlock, as shown in Figure 2.

There are three TSA tiers. If g is less than 0.10 then we will use simple TSA, that is also shown in Figure 3. If g is between 0.10 and less than 0.35 then we use TSA tier 2. In tier 2, only local window is used. Finally, if the g score is greater than or equal to 0.35 then we used tier 3. g measures how jumpy the signal from one time step to the next. More sudden jump means more energy. In the tier 2, attention is computed for every time step in the sequence. But each time step attends only the neighbors with in the local window, i.e. before and after 32 frames. In the tier 3, 2 passes happen for each time step. A local pass, same as the tier 2 and a global pass, with no window. In the global pass all the time passes are included. Tier 3 adds attentions weights of local and global, as shown in Figure 7.



a) The 3 Tier TSA: Temporal Spatial Attention block's internal computation



b) The work flow diagram of EfficientNETEEGv1

Figure 7: The architecture of EfficnetnetZEEGv1

The output of the three TSAs are mixed using equation 2.

mix = 0.3x + (m1 x y1) + (m2 x y2) + (m3 x y3). (4

Where, y1, y2 and y3 are outputs of the three TSAs branches respectively and m1, m2 and m3 are the selection gates, computed using equation 5, 6, and 7.

m1 = (g < = T_LIGHT) (5

m3 = (g > = T_GLOBAL) (6

m2 = 1 – m1 – m3 (7

By using these gates only one branch is selected along with small residual. If the g score is less than 0.1 then only y1 is selected. If g score is greater than 0.35 then only y3 is selected. This function is inspired by real time traffic light system.

This model helps to learn all types of signals based on their energy levels. Three TSA are defined here:

TSA1 is the first tier, that uses Simple TSA, that is used in the base model.

TSA2 is second tier, it uses local window with SeparableConvolutional1D, followed by batch normalization and swish. SeparableConvolutional1D() used depthwise-convolution() and Pointwise-convolion(). While SeparableConv1D is frequently characterized by channel-wise convolution, in our MFCC-based form, the feature dimension functions as the channel axis. In our proposed model, depthwise convolution independently captures **local temporal dynamics** for each **MFCC-derived feature**. It finds which feature dimension has meaningful temporal structure. It learns one temporal filter per feature dimension, i.e., does MFCC feature show useful short-term temporal behavior? Does it show rhythmic, bursty, or stable patterns? If the features are of no use then the learned weights become very small. It is followed by pointwise convolution to model inter-feature relationships. Point wise convolution finds how to mix the learned filter weights. Output features of point wise convolution are new set of features combined with learned weights from the depth-wise convolution. This process is done at each time step.

TSA3 is the third tier that uses, global window as well as local window. For global window it uses muti-head-attention global across the time. It learns like, this burst at 3 sec, is related to burst at 8 sec. It ignores the local noise and look for the long term dependencies. Local branch looks for short term patterns and global branch looks for long term dependencies.

Limitation: The TSA block is used inside MBConv block. Each time MBconv block is executed the model computes three TSAs and then according to the g value, one TSA is selected. It increased the computational and time complexity of the model to many folds. Almost 2 months are spent in model training. Although this gating approach improved the models' performance but we needed a light weight model that takes less time and provide better results.

There is one more limitation in the current model. "Local branch looks for short term patterns and global branch looks for long term dependencies". It might happen that local branch is chosen based on low energy score, but it has some long-term dependencies. Or global branch is chosen based on high g score, but it contains only short-term dependencies. To deal with this issue we need to make a mechanism that combines both short term and long-term dependencies, and do not cut off one branch completely based on g score.

2.3. EfficientNetZEEGv2

Again, we introduce an more effective and light Efficient Net-style 1D architecture where a per-window instability score *g* dynamically blends a local separable-conv branch and a global multi-head self-attention branch via smooth, graded gates. Our gates are explicitly temporal instability-aware. The workflow of this model is presented in Figure 8. Moreover, 3 MBConv blocks are used in stage A, 4 are used in stage B, and 5 are used in stage C. To reduce the computational cost of the model, we distributed the MBConv blocks in different stages.

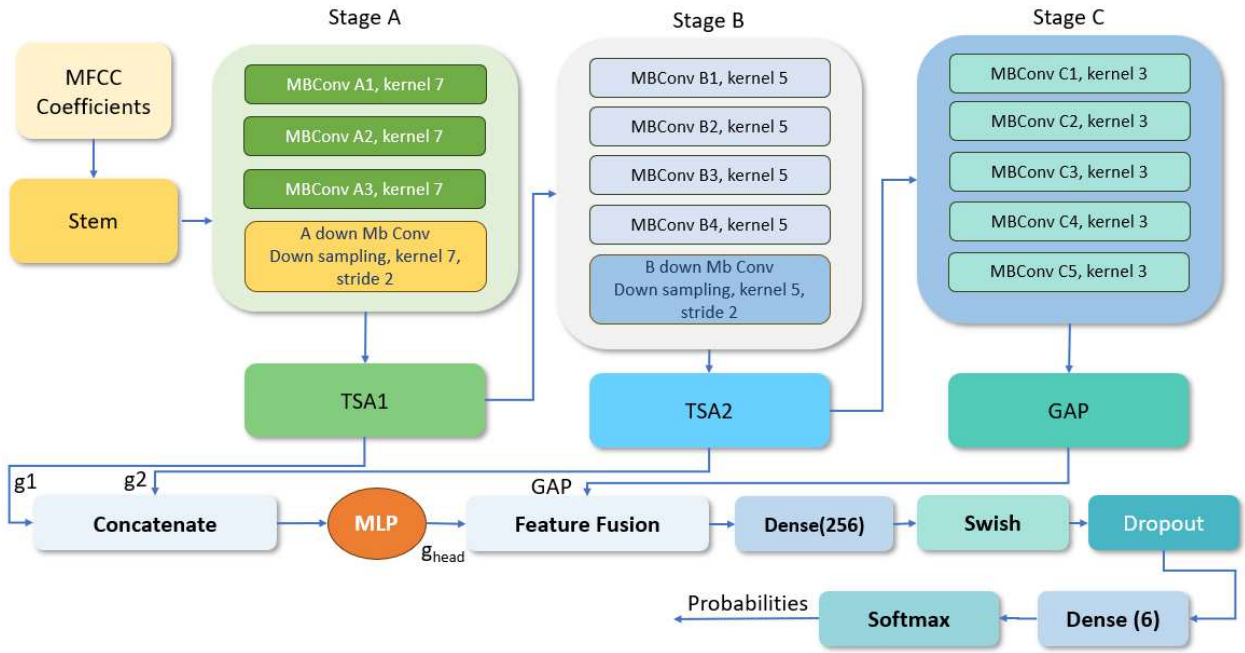


Figure 8. Work flow of EfficientNetZEEGv2 (the final proposed model)

Algorithm of EfficinetNetZEEGv2

Input (B,T,D)

Stem

Conv1D(3) -> BN -> SiLU -> SpatialDropOut1D

Stage A

MBConv1D x nA(stride =1)

A_down: MBConv1D (Stride = 2) <- downsample

TSA1 (local + global TSA + gating)

local_out -> SeperableConv1D(C, k = local_kernel) ->BN->SiLU

global_out -> MultiHeadAttention(time) -> LayerNorm

g1 -> instability_score_g(x) = $W_{L1} = \sigma((g1-T_LIGHT)/SIGMA)$,

$W_{G1} = \sigma((g1-T_GLOBAL)/SIGMA)$

mixed1 -> $0.3 \cdot x + (1-WG1) \cdot WL1 \cdot local_out + WG1 \cdot global_out$

Stage B (k =5)

MBConv1D x nB (Stride = 1)

B_down: MBConv1D (Stride =2) <- downsample

TSA2 (local + global TSA + gating)

local_out -> SeperableConv1D(C, k = local_kernel) -> BN -> SiLU

global_out -> MultiHeadAttention(time) -> LayerNorm

g2 -> instability_score_g(x) = W_{L2} = σ((g2 −T_LIGHT)/SIGMA),

W_{G2} = σ((g2 −T_GLOBAL)/SIGMA)

mixed2 -> 0.3·x + (1−WG2)·WL2·local_out + WG2·global_out

Stage C (k = 3)

MBConv1D x nC (Stride =1)

GlobalAveragePooling1D -> gap_feat

g-MLP: [g1, g2] -> Dense(16,swish) -> Dense(8, swish) -> g_feat

Concatenate: [gap_feat, g_feat]

Head: Dense(256, swish) -> Dropout -> Dense(NUM_CLASSES, softmax)

It can be seen from the Figure 5 that we have made some changes in the main architecture of the model. Instead of replacing SE block with TSA blocks, we added them immediately after MbConv blocks. Using TSA within the MBConv blocks increases computational time to manifolds. Now in the current scenario, we down sample the input before passing it to TSA blocks. Moreover, we used a dynamic gating mechanism instead of strict if conditions. The gating function is computed using equation 8.

$y = \alpha x + a(x)L(x) + b(x)G(x).$ (8

Where α is the coefficient value for allowing identity path to add into the equation. In the current scenarios its value is 0.3.

Gating function = Identity path + local branch + Global branch. (9

Gating function composed of three parts, as shown in equation 9, the identity path, local branch and the global branch. Identity path is inspired from the ResNet model, so the gradient vanishing/exploding problem can be managed in case local and global branch gives us a very less weight.

$y = 0.3\ x + (1\text{-}W_G).W_L$. local_out + W_G. global_out (10

W_G and W_L in equation 8, are local and global weights that are computed using sigmoid functions. Sigmoid function is widely used for calculating weights in the literature [13]. Local_out is the function that works on the local window, and use SeparableConvolutional1D(), and Global_out is the function that uses global window and Multi-head Attention () function.

Let g € (0,1) be the per sample x, instability score, then $W_L(g)$, and $W_G(g)$ can be computed using equations 9 and 10.

$W_L(g) = \sigma(g - T_{LIGHT} / SIGMA)$, (11

And

$W_G(g) = \sigma(g - T_{GLOBAL} / SIGMA)$ (12

Where, sigmoid is computed using equation 13.

$\sigma(u) = (1/ 1 + e^{-u})$ (13

$W_L(g)$, and $W_G(g)$ are the weights for the local and global branches, that will determine how much weight should be given to each branch for a given sample x. According to the values of these weights the local and global branch will contribute to the output of TSA. Here are some values for the constants we used in the above equations.

$T_{LIGHT} = 0.10$, shows where the local gate begins to open.

$T_{GLOBAL} = 0.35$, shows where the global gate begins to open.

SIGMA = 0.08, in controls softness. Larger value shows a gentler slope.

The intuition behind using this logic is same as we have used in the temporal instability-aware gating. But we have changed the position of TSA along with a dymanic gating mechanism that allows both branches to contribute.

If $g < 0.10$; W_L and W_G will be approximately equal to zero.

If g is near 0.10, W_L rises from zero to one, and W_G will remain zero.

If g is near 0.35, then both W_L and W_G will have non-zero values.

If g is greater than 0.35, W_G will approach to 1 i.e., global branch will dominate.

Two branches global and local are used, because local branch is inexpensive, robust to noise, excels at short range features, and best for calm signals. Whereas global branch is expensive, and binds distant features by detecting long range dependencies that would need many layers of convolution neural network to determine.

We compute both local and global outputs on the same x, and subsequently, our gating function fuses them based on the per-sample instability score g. In the prior case, we did not identify weights for determining which branch should be utilized. In EfficientNetZEEGv2, weights were utilized to ascertain the contribution of each branch to the final output. It gives us adaptive compute and receptive field per sample. Moreover, by using identity path gradient also flows smoothly.

Logic behind using low g values for short range features and high g values for long range time features comes from the facts given below:

- Low g means that there is little temporal changes, EEG is smooth, calm and less instable signals. So, for a low g score, we used local branch.
- High g means rapid fluctuations or spikes, it can be due to strong transient events (i.e., seizure bursts, or muscle noise etc.)
- If for low g signals we use global attention, it would waste compute and risk over smoothing tiny oscillations. Similarly, if for a high g we use local branch, it will not be able to grasp long range features. Global attention helps bind distant frames and detect long range correlations of EEG signals.

Table 2 summarize the relations of g score with local and global branches of TSA(s).

Table 2. Relation of g-score with global and local branch

g-range	EEG condition	Preferred branch	Reason
g < 0.10	Calm, low activity	Local	Short spikes, stable signal.
0.10 <= g < 0.35	Moderate activity	Both	Signal have both short-range and long-range features and there is medium level of activity.
g >= 0.35	Bursts	Global	High energy in the signals and long range dependencies.

Local Branch

branch consists of these functions. SeperableConv1D, Batchnormalization, and Activation("swish"). SeperableConv1D is further divided into Depthwise-temporal-conv and Pointwise(1x1). SeperableConv1D is lighter than CNN.

In our proposed model, local branch uses convolutional network for extracting local temporal dependencies of the frame. Say if the model is processing frame 5, then local branch will consider only the neighboring frames according to the kernel size. It won't consider all the 20 frames in the window in one go. Thus, local branch learns the temporal dependencies at a smaller level.

Global branch

It has the function MultiheadAttention, also called MHA. In MHA we build Queries, Keys, and Values from the same input x, also called self-attention. By attention we mean, given a query "q", we want to know which key "k" the query should pay "attention" to, with respect to the associated value "v".

MHA: say if there are sequence of EEG signals (or words of NLP). $x = [x1,x2,x3,\ldots.x_T]$ each x is a feature vector at time t. Self-attention allows model to decide "how much each time step attend to other time steps?". Instead of fixing the number of kernels, MHA lets the model decide and connect distant dependencies. It allows long range patterns to be recognized easily. Say for frame 5, MHA will find the relation of frame 5 with all the other frames in the 10-sec window, which have 20 frames. It finds the long-term dependencies.

Query, Keys and Values can be defined by $Q = xW_Q$, $K = xW_K$, and $V = xW_V$. each of them is divided into "h" heads. i.e. Q_h, K_h, and V_h. It is like viewing on sample from different perspectives simultaneously.

For each head h, attention weights compare every time step with every other.

$$Ah = \text{Softmax}(Q_h . K_h^T / \sqrt{d_k}) \quad (14$$

$Q_h . K_h^T$ in equation 12 gives similarity score between all pairs of time steps. Dividing by $\sqrt{d_k}$ prevents overly large values

Then we use weights to aggregate the values using equation 15.

$$\text{Head } h = A_h . V_h \quad (15$$

Each time step now contains a weighted combination of all others.

At the end all heads are concatenated to form one output sequence.

The difference between two models is summarized in Table 3.

Table 3: Key differences between the two models.

EfficientNetZEEGv1	**EfficientNetZEEGv2**
Used strict gates, m1,m2 and m3.	Used soft gating function by introducing weights, W_L and W_G.
Only used one tier information other two tiers are totally skipped	Used information from the three tiers based on the weighting function. None of the tiers are completely skipped.
Used TSA inside MB conv block	Used TSA outside the MBConv block, between two stages.
Used simple TSA for first tier	Didn't use Simple TSA in any tier.
If $g < 0.1$ the use simple TSA, tier 1 If $0.1 < g < 0.35$, use only local window, tier 2 If $g > 0.35$ use local and global windows, tier 3	If $g < 0.1$, use only local window, tier 1 If $0.1 < g < 0.35$ use both local and global branches according to the weights. If $g > 0.35$ only use global branch
Local branch: sparable Conv1D Global branch: Multi-head attention	Local branch: sparable Conv1D (same) Global branch: Multi-head attention (same)
Didn't use SE block	Used SE block in MBconv, instead of Channels we used feature dimensions.
Used 17 MBConv blocks in total	Used 12 MBConv blocks in total

3. Results and Discussion

This section discusses the results of the proposed method, comparing them with state-of-the-art methods and models presented in the literature. The code implementation utilizes a 13th-generation Intel Core i7, equipped with 32 GB of internal RAM and a GPU. The system operates on a 64-bit architecture. The Visual Studio framework, Jupyter notebook environment, and Python 3.10 programming language were utilized. The code implementation is divided into two segments as previously discussed. This section describes the performance of the proposed model on datasets from two modalities: signals and images.

3.1. Performance Metrics

Performance metrics used in this research study are accuracy, recall, precision, F1-score, and loss. The values of these performance metrics can be found using equations 14, 15, and 16.

Accuracy = (TP +TN) / (TP +TN+FP +FN) (14

Precision = TP / (TP +FP) (15

Recall = TP / (TP +FN) (16

TP denotes true positive, FP indicates false positive, TN represents true negative, and FN signifies false negative. TN indicates that the sample is accurately classified as negative, while FN denotes that the sample is positive but incorrectly classified as negative. In the same way, TP indicates that the data sample is accurately classified as positive, whereas FP denotes that the sample is negative but is incorrectly classified as positive [14]. This section presents results for both datasets separately: segment 1 addresses the HMS dataset, while segment 2 focuses on the brain tumor dataset.

We are using two datasets for testing the models' performance. One if EEG signals' dataset, i.e., HMS and the second on is brain tumor dataset consists of MRI images. The results are divided into two segments, as shown in Figure 9.

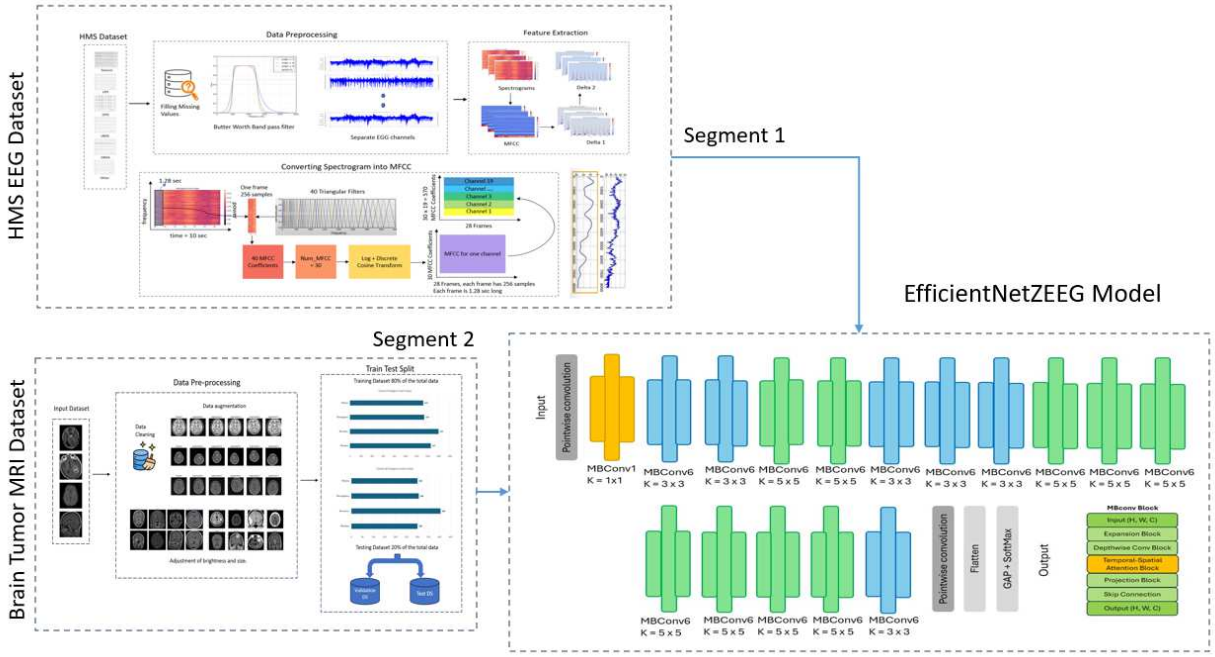


Figure 9: Dual modal data can be used with the proposed model, EfficientNetZEEG. This is done by dividing the experiment in two segments. Segment 1 for EEG signals and Segment 2 for MRI images.

3.2. Segment 1, EEG Signals

3.2.1. EfficientNetZEEGv1

This section presents results and discussion pertaining to the HMS dataset, using the proposed model. We used 10-fold cross validation for **instability aware gating TSA**. The results are shown in the Figure 10.

```
======================================================================
10-fold CV Summary (mean ± std)
======================================================================
Accuracy      : 0.9707 ± 0.0073
Macro Precision: 0.9718 ± 0.0079
Macro Recall   : 0.9708 ± 0.0096
Macro F1       : 0.9711 ± 0.0088
```

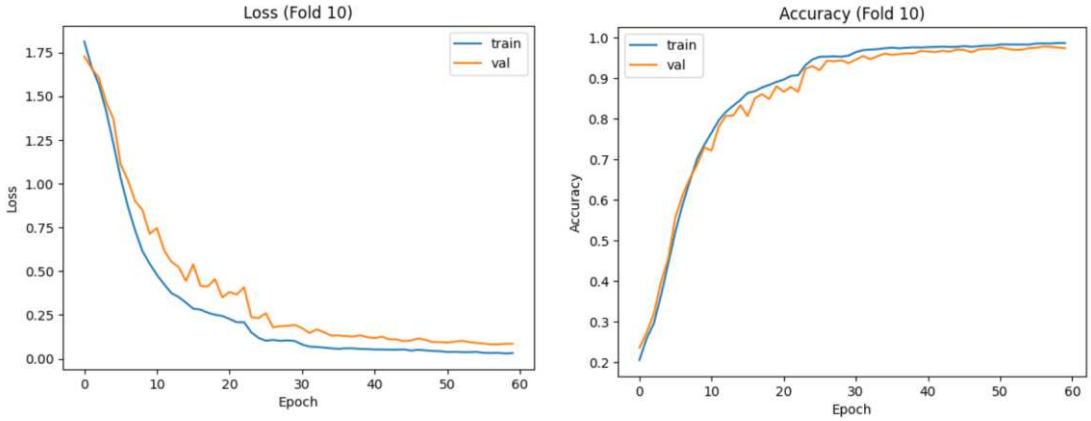
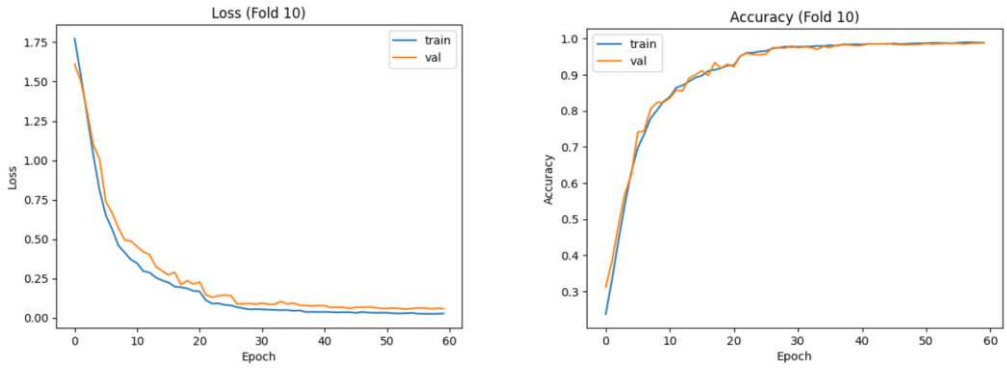


Figure 10: Results of instability aware gating for 10-fold cross validation, model achieved 97% validation accuracy, and 0.20 loss value.

Mean tells about expected performance **Std** tells about reliability. Low std means performance is consistent across folds (good generalization). High std means performance fluctuates a lot (model unstable). In this case, mean (average) is the central performance across all folds (or runs). Std (standard deviation) shows how much the performance varies between folds. So, for result: Accuracy $= 0.9702 \pm 0.0056$ means: Average accuracy across the 10 folds is **97.02%**. Standard deviation is **0.56%**, showing small variation (stable performance).

3.2.2. EfficientNetZEEGv2 (Soft-gating)

The results of EffiicentNetZEEGv2 are shown in the Figure 11 blow. The results clearly surpass all the other studies in the literature, with best accuracy, precision and recall.



a) Accuracy and loss plots on 10-fold cross validation.

```
======================================================================
10-fold CV Summary (mean ± std)
======================================================================
Accuracy      : 0.9871 ± 0.0021
Macro Precision: 0.9877 ± 0.0017
Macro Recall   : 0.9886 ± 0.0021
Macro F1       : 0.9881 ± 0.0018
```

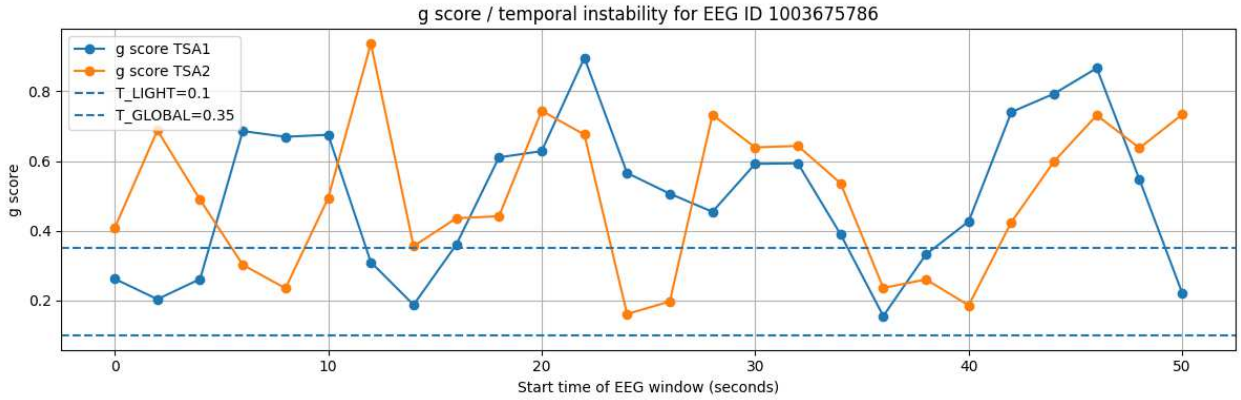
b) Results of 10-fold cross validation.

	precision	recall	f1-score	support
seizure	0.9817	0.9978	0.9897	912
lpd	0.9841	0.9920	0.9880	624
gpd	0.9921	0.9843	0.9881	508
lrda	0.9931	0.9931	0.9931	289
grda	0.9897	0.9836	0.9867	489
other	0.9891	0.9798	0.9844	1389
accuracy			0.9874	4211
macro avg	0.9883	0.9884	0.9883	4211
weighted avg	0.9874	0.9874	0.9874	4211

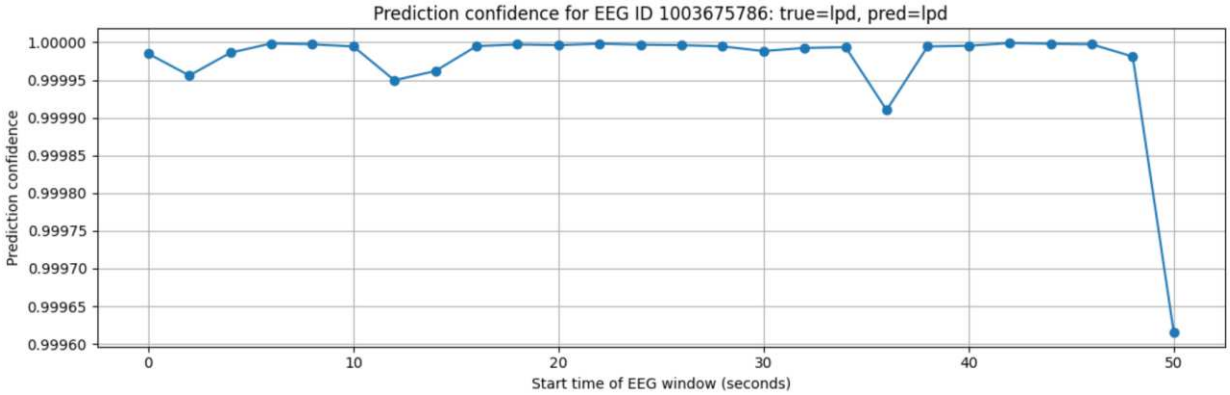
c) Classification report of test dataset.

Figure 11: Classification results of the EfficientNetZEEGv2

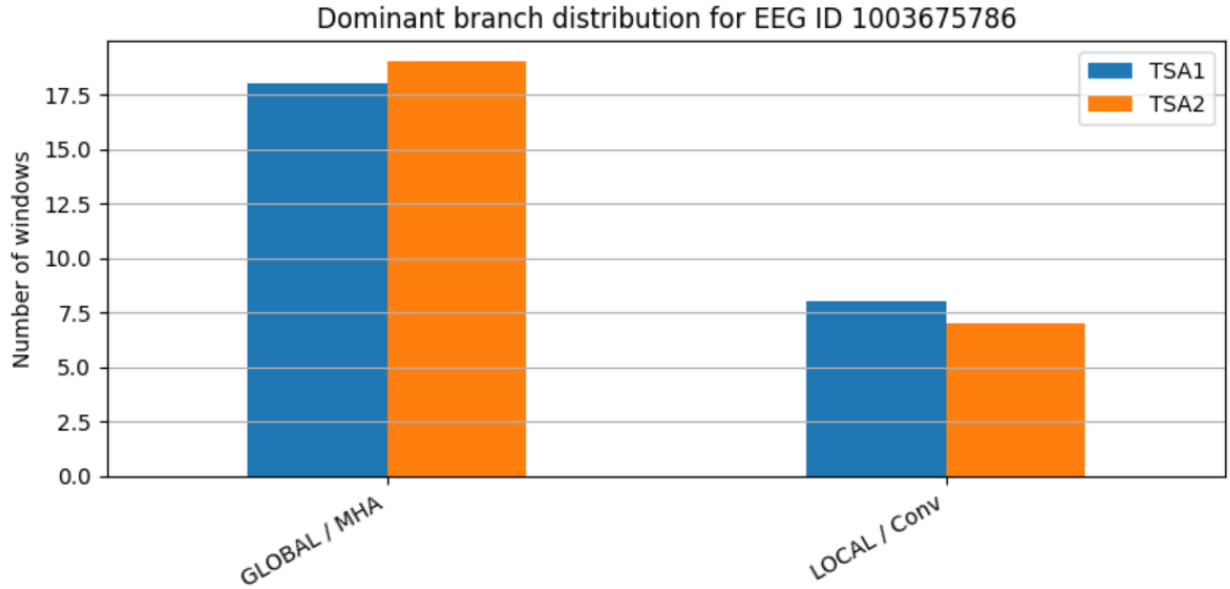
G-scores for two branches



For EEG_ID:1003675786



Prediction certainty for the different windows for EEG_ID: 1003675786



Number of windows used by local and global branches for EEG_ID 1003675786

Table: G_scores and TSA branch selection for EEG_ID: 1003675786

window_id	start_frame	start_sec	true_label_id	true_label	true_class	pred_class	confidence	g_TSA1	wLocal_TSA1	wGlobal_TSA1	branch_TSA1	g_TSA2	wLocal_TSA2	wGlobal_TSA2	branch_TSA2
0	0	0	1	lpd	lpd	lpd	0.999985	0.262044	0.883451	0.249846	LOCAL / Conv	0.407888	0.979134	0.673398	GLOBAL / MHA
1	4	2	1	lpd	lpd	lpd	0.999956	0.203541	0.784865	0.138153	LOCAL / Conv	0.688256	0.99936	0.98563	GLOBAL / MHA
2	8	4	1	lpd	lpd	lpd	0.999986	0.261373	0.882585	0.248277	LOCAL / Conv	0.488975	0.992325	0.850327	GLOBAL / MHA
3	12	6	1	lpd	lpd	lpd	0.999999	0.686097	0.999342	0.985243	GLOBAL / MHA	0.302212	0.926055	0.354951	LOCAL / Conv
4	16	8	1	lpd	lpd	lpd	0.999997	0.669812	0.999194	0.981971	GLOBAL / MHA	0.23481	0.843579	0.191567	LOCAL / Conv
5	20	10	1	lpd	lpd	lpd	0.999995	0.675635	0.99925	0.983216	GLOBAL / MHA	0.494141	0.992802	0.85836	GLOBAL / MHA
6	24	12	1	lpd	lpd	lpd	0.99995	0.309717	0.932228	0.376712	LOCAL / Conv	0.93713	0.999972	0.999351	GLOBAL / MHA
7	28	14	1	lpd	lpd	lpd	0.999962	0.187349	0.748729	0.115769	LOCAL / Conv	0.355828	0.960752	0.518203	GLOBAL / MHA
8	32	16	1	lpd	lpd	lpd	0.999995	0.360119	0.962725	0.531579	GLOBAL / MHA	0.436031	0.985231	0.745619	GLOBAL / MHA
9	36	18	1	lpd	lpd	lpd	0.999997	0.61073	0.998315	0.962998	GLOBAL / MHA	0.441665	0.986221	0.758743	GLOBAL / MHA
10	40	20	1	lpd	lpd	lpd	0.999996	0.628627	0.998652	0.970194	GLOBAL / MHA	0.744505	0.999683	0.992834	GLOBAL / MHA
11	44	22	1	lpd	lpd	lpd	0.999998	0.896458	0.999953	0.998921	GLOBAL / MHA	0.676033	0.999254	0.983298	GLOBAL / MHA
12	48	24	1	lpd	lpd	lpd	0.999997	0.565771	0.997047	0.936855	GLOBAL / MHA	0.161036	0.681992	0.086115	LOCAL / Conv
13	52	26	1	lpd	lpd	lpd	0.999996	0.506466	0.993823	0.876077	GLOBAL / MHA	0.197014	0.770769	0.128723	LOCAL / Conv
14	56	28	1	lpd	lpd	lpd	0.999995	0.454423	0.988229	0.786722	GLOBAL / MHA	0.732658	0.999632	0.9917	GLOBAL / MHA
15	60	30	1	lpd	lpd	lpd	0.999988	0.592161	0.997875	0.953778	GLOBAL / MHA	0.639261	0.99882	0.973807	GLOBAL / MHA
16	64	32	1	lpd	lpd	lpd	0.999993	0.593468	0.997909	0.954493	GLOBAL / MHA	0.643664	0.998883	0.975175	GLOBAL / MHA
17	68	34	1	lpd	lpd	lpd	0.999994	0.389794	0.973976	0.621851	GLOBAL / MHA	0.536529	0.99575	0.911459	GLOBAL / MHA
18	72	36	1	lpd	lpd	lpd	0.99991	0.15518	0.66591	0.080526	LOCAL / Conv	0.235799	0.845204	0.19349	LOCAL / Conv
19	76	38	1	lpd	lpd	lpd	0.999995	0.332148	0.947936	0.444443	LOCAL / Conv	0.260351	0.881254	0.245899	LOCAL / Conv
20	80	40	1	lpd	lpd	lpd	0.999995	0.426825	0.983459	0.723181	GLOBAL / MHA	0.185843	0.745172	0.113857	LOCAL / Conv
21	84	42	1	lpd	lpd	lpd	0.999999	0.740685	0.999668	0.992486	GLOBAL / MHA	0.424777	0.983038	0.718027	GLOBAL / MHA
22	88	44	1	lpd	lpd	lpd	0.999998	0.79353	0.999828	0.996105	GLOBAL / MHA	0.599153	0.998053	0.957482	GLOBAL / MHA

23	92	46	1	lpd	lpd	lpd	0.999997	0.866806	0.999931	0.998438	GLOBAL / MHA	0.73164	0.999628	0.991595	GLOBAL / MHA
24	96	48	1	lpd	lpd	lpd	0.999981	0.547174	0.996277	0.921626	GLOBAL / MHA	0.63756	0.998794	0.973259	GLOBAL / MHA
25	100	50	1	lpd	lpd	lpd	0.999616	0.221472	0.8203	0.167064	LOCAL / Conv	0.733629	0.999637	0.991799	GLOBAL / MHA

This table shows the inside calculation, and showing the weights that are calculated for including global and local branch.

Comparative Analysis

Initially, we utilized raw EEG signals alongside pretrained algorithms to identify the differences in outcomes between the proposed model and the pretrained algorithms. This study proposes a modified version of a pretrained algorithm. For result comparison, we utilized raw EEG signals alongside pretrained models to evaluate classification accuracy, precision, and recall. The models comprise VGG19, VGG16, EfficientNetB0, EfficientNetB3, and ResNet50. Table 4 presents the classification results of the transfer learning algorithms. The HMS dataset from Kaggle was utilized for the evaluation of these models, is the same dataset that we use for proposed model. Table 4.5 indicates that the highest accuracy achieved through the Transfer learning approach is 66.78%. In transfer learning methodologies, the classification layers are typically frozen, and customized layers are incorporated atop the pre-trained models. This method reproduces previous research utilizing the same dataset, allowing for a comparison of the performance of our proposed model.

Table 4: Performance of the Transfer Learning Algorithm on the HMS Dataset.

Algorithm	Training Accuracy	Validation Accuracy	Validation Precision	Validation Recall	Loss
VGG19	0.97	0.48	0.49	0.46	4.90
VGG16	0.91	0.29	0.30	0.28	4.32
EfficientNetB0	0.89	0.62	0.63	0.61	1.08
EfficientNetB3	0.96	0.64	0.65	0.64	1.65
Resnet50	0.92	0.66	0.67	0.65	1.30
EfficientNetZEEGv1 (Proposed model)	0.98	0.970	0.971	0.970	0.20
EfficientNetZEEGv2 (Proposed model)	0.99	0.987	0.987	0.988	0.056

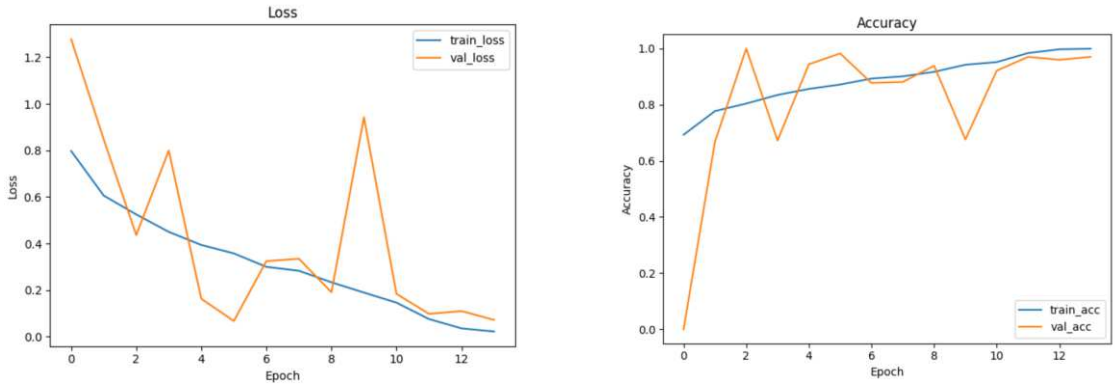
Table 5: Comparison of accuracy and loss values on the HMS dataset with the existing studies.

Reference, Year	Dataset	Accuracy	Loss
[5], 2024	HMS dataset	0.69	0.96
[6], 2024	HMS dataset	0.82	Not Mentioned
[15], 2024	HMS dataset	0.86	0.38
Proposed Model, Simple TSA	HMS dataset	0.92	0.31

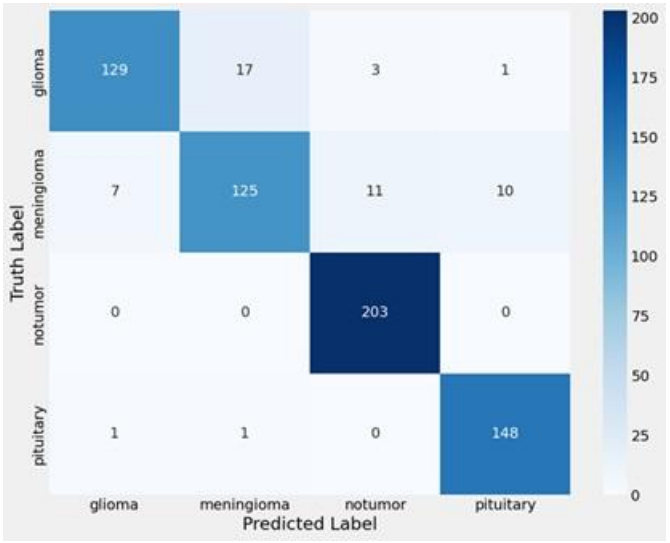
Proposed model, instability aware gating TSA		0.97	0.23
Proposed model, EfficientNetZEEGv2	HMS	0.99	0.056

3.3. Segment 2, MRI Images

The proposed model was evaluated using an MRI dataset focused on an HBA problem related to brain tumors to illustrate its robustness. The model achieved a training accuracy of 99.79% and a validation accuracy of 93%, with a loss of 0.27, precision of 0.93, and recall of 0.93. Figure 12 illustrates the confusion matrix. The accuracy is inadequate for evaluating the model's performance. It is essential to identify models that exhibit certainty in their predictions. A correlation was observed between the model's loss and its prediction certainty. A greater loss in the model signifies diminished confidence in its predictions regarding unseen data. A reduction in loss signifies enhanced model efficiency and increased confidence in the prediction value. A reduced value of the loss function signifies the model's predictive accuracy and the degree of divergence from actual values. An optimal condition is characterized by a lower loss value. A study investigates the loss values of benchmark deep learning models alongside various transfer learning models to identify which model exhibits superior predictive performance. The reliability of the models is inversely related to the loss value [43].



a) Accuracy and Loss graphs of EfficientNetZEEG on MRI train and validation dataset. Model achieved 99 % training accuracy and 97.02% validation accuracy.



b) Confusion matrix for the EfficientNetZEEG model for the MRI test dataset.

Figure 12: Performance of EfficientNetZEEG on MRI dataset.

Table 6 indicates that, aside from the loss value, all other performance metrics are nearly identical for both models. The model's prediction results indicate a different narrative. EfficientNetZEEG demonstrates superior prediction accuracy and a higher certainty in the prediction values compared to the CNN model, despite yielding identical accuracy, precision, and recall.

Future Work: In the future, we will use the EEG signals and the MRI dataset of the same patients for the same events. We will also use MFCCs with varying window sizes.

Table 6: Performance metric values on training, testing, and validation datasets for CNN and EfficientNetZEEG.

Model	Dataset	Accuracy	Loss	Precision	Recall
CNN	Training	95%	0.12	95%	95%
	Validation	90%	0.37	90%	90%
	Testing	91%	0.30	91%	91%
EfficientNetZEEG	Training	99%	0.014	99%	99%
	Validation	97%	0.07	97%	95%
	Testing	93%	0.27	93%	93%

4. Conclusion

Harmful brain activity may impair cerebral function and cause death if not treated. The worldwide shortage of EEG data analysis experts makes this problem harder to solve. A reliable and automated technique is needed to classify harmful brain activities, including seizures and seizure-like patterns (LPD, GPD, LRDA, and GRDA). We not only classified EEG signals, but also the MRI images. Kaggle contributed EEG records from 106,800 critically ill patients by obtaining their seizure and seizure-like patterns. In this research study, EEG signals were converted into spectrograms, and then MFCCs and their first and second derivatives, Delta1 and Delta2, were obtained. We used extracted features, labels, and EEG IDs to create a new dataset. Expert consensus classifies the dataset as "LPD," "GPD," "LRDA," "GRDA," "Seizure," and "Other." Accuracy, precision, recall, and are used to evaluate the model. EfficientNetZEEGv1, our proposed model, achieved validation accuracy of 97% and a loss of 0.20. We made some enhancements in the gating mechanism of the model, and our second model EfficientNetZEEGv2, achieved a validation accuracy of 99%. Two sets of data were used to test this model: EEG delta features from people who have seizures and MRI images of brain tumors. To test the proposed model, both types of data were used. The model can handle many different types of input, even though they are from distinct scenarios and use different types of data. On the MRI dataset, our proposed model achieved an accuracy of 93% and a loss value of 0.27. This work provides a robust AI-based classification approach. This study will help doctors and researchers diagnose brain damage sooner and more accurately, enabling faster and more precise seizure therapy. The light architecture of the EfficientNetZEEGv2 model used in this study makes it ideal for real-time scenario.

Reference

[1] Q. Liu, L. Yang, Z. Zhang, H. Yang, Y. Zhang, and J. Wu, "The Feature, Performance, and Prospect of Advanced Electrodes for Electroencephalogram," Jan. 01, 2023, *MDPI*. doi: 10.3390/bios13010101.

[2] E. L. Johnson and P. W. Kaplan, "Population of the ictal-interictal zone: The significance of periodic and rhythmic activity," Jan. 01, 2017, *Elsevier B.V.* doi: 10.1016/j.cnp.2017.05.001.

[3] L. J. Hirsch *et al.*, "American Clinical Neurophysiology Society's Standardized Critical Care EEG Terminology: 2021 Version," *Journal of Clinical Neurophysiology*, vol. 38, no. 1, pp. 1–29, Jan. 2021, doi: 10.1097/WNP.0000000000000806.

[4] J. W. Lee *et al.*, “Development and Feasibility Testing of a Critical Care EEG Monitoring Database for Standardized Clinical Reporting and Multicenter Collaborative Research,” *Journal of Clinical Neurophysiology*, vol. 33, no. 2, pp. 133–140, Apr. 2016, doi: 10.1097/WNP.0000000000000230.

[5] V. R. A’izzah, A. Rahmatulloh, and V. Purwayoga, “Improving Harmful Brain Activity Classification Model with Electroencephalography Spectrogram Using Transfer Learning,” *SSRN*, 2024, [Online]. Available: https://ssrn.com/abstract=4928067

[6] S. S. Bhatti, A. Yadav, M. Monga, and N. Kumar, “Comparative Analysis of Deep Learning Approaches for Harmful Brain Activity Detection Using EEG,” Dec. 2024, [Online]. Available: http://arxiv.org/abs/2412.07878

[7] Y. A. Saadoon, M. Khalil, and D. Battikh, “Predicting Epileptic Seizures Using EfficientNet-B0 and SVMs: A Deep Learning Methodology for EEG Analysis,” *Bioengineering 2025, Vol. 12, Page 109*, vol. 12, no. 2, p. 109, Jan. 2025, doi: 10.3390/BIOENGINEERING12020109.

[8] Z. Xiong *et al.*, “A Study on Seizure Detection of EEG Signals Represented in 2D,” *Sensors 2021, Vol. 21, Page 5145*, vol. 21, no. 15, p. 5145, Jul. 2021, doi: 10.3390/S21155145.

[9] F. Hu *et al.*, “HASTF: a hybrid attention spatio-temporal feature fusion network for EEG emotion recognition,” *Front. Neurosci.*, vol. 18, p. 1479570, Oct. 2024, doi: 10.3389/FNINS.2024.1479570/BIBTEX.

[10] Z. U. Nisa *et al.*, “Beyond Accuracy: Evaluating certainty of AI models for brain tumour detection,” *Comput. Biol. Med.*, vol. 193, p. 110375, Jul. 2025, doi: 10.1016/J.COMPBIOMED.2025.110375.

[11] S. Alzahrani, H. Banjar, and R. Mirza, “Systematic Review of EEG-Based Imagined Speech Classification Methods,” Dec. 01, 2024, *Multidisciplinary Digital Publishing Institute (MDPI)*. doi: 10.3390/s24248168.

[12] Z. Unnisa, A. Jaffar, S. Akram, I. Din, K. Khan, and S. Masood Bhatti, “EEG-based harmful brain activity classification using deep learning and feature fusion,” *Sci. Rep.*, May 2026, doi: 10.1038/s41598-026-48670-5.

[13] J. Hu, L. Shen, and G. Sun, “Squeeze-and-Excitation Networks,” in *IEEE conference on computer vision and pattern recognition*, IEEE, 2018, pp. 7132–7141. [Online]. Available: http://image-net.org/challenges/LSVRC/2017/results

[14] W. Pei, Y. Li, P. Wen, F. Yang, and X. Ji, “An automatic method using MFCC features for sleep stage classification,” *Brain Inform.*, vol. 11, no. 1, Dec. 2024, doi: 10.1186/s40708-024-00219-w.

[15] D. Manolescu, N. Buckley, and E. L. Secco, “Machine Learning Models for Probability Classification in Spectrographic EEG Seizures Dataset,” *WSEAS Transactions on Biology and Biomedicine*, vol. 21, pp. 260–271, 2024, doi: 10.37394/23208.2024.21.27.