Emotion Detection for Enhanced Psychiatric Diagnosis

Julien GADONNEIX

***Abstract*—Emotional states play a crucial role in psychological health, influencing conditions like depression and presenting challenges due to their subjective nature. Recent advances in emotion detection systems offered promising glimpses for improving psychiatric assessments. Specifically, Brain-Computer Interface technologies provide a direct interface between brain activity and computational systems. This enables a possibly very accurate classification of emotional states using EEG signals. This report reviews methodologies in EEG-based emotion recognition and details the work achieved during this internship and the work in progress. It explores forward-looking deep learning techniques such as the deptwise separable convolutional layer, the Capsule network or the transformer-based models. They are discussed for their effectiveness in extracting temporal, spatial, and frequency features from EEG data. Moreover, the report focuses on self-supervised learning and contrastive learning for emotion classification across diverse patient populations. The potential of these frameworks to provide personalized insights is highlighted making a connection between technologies and mental health diagnostics.**

I. Introduction

Psychological states, troubles or illnesses are often challenging due to the subjective nature of emotions and the diverse ways individuals express them and emotional states greatly impact psychological health and can contribute to conditions like depression. However, research over the past few years has proven that advanced emotion detection systems can assist psychologists in the diagnostic process [1]. By utilizing signal processing algorithms and deep learning models, an automated process would analyze brain waves measured by a Brain-Computer Interface (BCI) equipment to accurately classify and understand patients' emotions. Integrating this technology into psychiatric assessments could offer a more objective and comprehensive view of an individual's emotional state, helping clinicians create more targeted treatment plans or simply promoting relaxation.

Emotion detection methods can be categorized into those based on physiological signals and those based on non-physiological signals. Physiological signal-based methods use indicators such as electrocardiograms (ECG), electrooculography (EOG), electromyograms (EMG), blood pressure and electroencephalograms (EEG) to detect emotional states. EEG signals have been shown to provide valuable features for emotion recognition, as they respond more quickly to emotional changes compared to other peripheral neural signals due to their direct measurement of brain activity [3]. Furthermore, since emotional states are closely linked to the central nervous system, EEG is particularly effective in accurately reflecting emotional states. Non-physiological signal-based methods rely on features like text, speech, body posture, and facial expressions. Physiological signals provide more detailed and complex information about emotions and it is harder to influence them by conscious control.

BCI technologies enable direct connections between human brains and peripheral devices, significantly impacting daily life, particularly in the realm of emotion recognition. Recently, this field has garnered increasing attention from both academic and industry communities thanks to its various applications [1].

Emotion recognition using EEG has been extensively studied in the literature. Researchers have primarily focused on classifying emotional states within the same subject, achieving very high performance. However, there is still significant work needed in modeling emotions across different subjects and datasets [2].

EEG emotion recognition methods generally involve two steps: feature extraction and feature classification. EEG features can be categorized into three domains: time domain, frequency domain, and time–frequency domain. Time-domain features describe the temporal characteristics of EEG signals whereas frequency-domain features capture the frequency characteristics of EEG signals. To obtain these features, EEG signals must first be transformed from the time domain to the frequency domain using the fast Fourier transform (FFT). Through the short-time Fourier transform (STFT) and wavelet transform, raw EEG signals can be converted into the time–frequency domain, allowing time–frequency features to describe both the temporal and frequency characteristics of EEG signals [5]. In the EEG feature classification step, the extracted features are classified to determine the specific emotional state. Popular classifiers for this task include k-nearest neighbor (KNN) or support vector machine (SVM) [2]. For example, Ramoser et al. proposed a strategy for EEG utilizing common spatial pattern matrices for optimal filterning to enhance emotion classification performance with a linear classifier [6]. Traditionally, the features were then input into the selected classifier but this methodology was not very efficient on emotion detection [2]. However, deep learning methods can both use the raw and processed EEG directly as input and a combination of deep learning models can be utilized as well.

In recent years, deep learning-based approaches have gained significant importance in EEG-based emotion recognition. Numerous studies have tested their algorithms on publicly available datasets using subject-dependent methods. Let us detail the state-of-the-art deep learning methodologies for emotion classification. For instance, one study utilized long short-term memory (LSTM) networks to extract temporal features from EEG signals [11]. Yang et al. proposed an

approach based on a parallel convolutional recurrent neural network (CRNN) to simultaneously learn spatial and temporal features from EEG data [12]. Gong et al. used a convolutional module to extract local features which are then concatenated and processed by a transformer-based temporal encoding layer for global feature awareness [13]. Another study employs an adaptive multi-head self-attention mechanism to encompass spatial, frequency and temporal aspects [7]. Song et al. presented the graph embedded CNN (GECNN), which extracts local CNN features along with global functional features to provide complementary emotion information [14]. Li et al. developed an efficient CNN-based EEG emotion recognition approach using contrastive learning to make full use of emotion labels [15]. A method leverages contrastive learning to extract meaningful representations from EEG data by recombining channels from multi-channel recordings to focus on the subject-independent scenario [16]. Shen et al. used contrastive learning to align EEG signal representations across subjects by enhancing similarity for the same emotional stimuli and reducing it for different stimuli [17]. An adversarial discriminative-temporal convolutional network (AD-TCN) ensures the invariance of feature graph representations across domains and handle the temporal attributes of EEG [19]. Another study focuses on adversarial network but for the subject-dependent scenario with a dual encoder: variational autoencoder-generative adversarial network [18]. Some studies wished to overcome the limitations of locality of a CNN by utilizing a capsule network [30] or [29]. Additionally, a different study used a transformer Capsule Network. This proposed algorithm featured an EEG Transformer module for extracting EEG features and an Emotion Capsule module for refining the features and classifying the emotional states [31].

In this project, various solutions or attempts to the classification of emotions using EEG and deep learning both within and across subjects are suggested.

- Starting from a famous model in global EEG features know for its small size and its flexibility, the project aims at turning it into a model for emotion classification. The goal is to adapt it to the current task while keeping the fundamental and original advantages of the model. The small number of parameters combined with the efficiency on the subject-dependent scenario could provide an alternative for therapists. Instead of loading a cross-patients model already trained, they could easily train a new one for each new patient.
- Self-supervised learning is a great approach to extracting meaningful features across patients in EEG data. In this project two benchmark models using a pre-training phase, that can be personalized in one of the model, will be reworked to fit the problem. These models involve a huge number of parameters but their innovative architecture or process could have many prolific consequences.
- Finally, a part of the project centers around a state-of-the art model that uses the latest deep learning architectures. However, both in the preprocessing phase and in the choice of some hyperparameres, it seems like some improvements could be brought.

II. Background

A. Emotions in the brain

1) Physiology: The brain is the most complex organ in the human body, consisting of hundreds of billions of neurons interconnected in various ways. It controls task-evoked responses, movement, senses, emotions, language, communication, thinking, and memory. Additionally, the brain is protected by the skull and three covering layers called the meninges. The brain can be divided into four main parts: the cerebellum, diencephalon, cerebrum, and brain stem. The cerebrum is considered the center of intellect, enabling activities such as reading, communication, memory, and emotions. It is divided into four lobes, each associated with specific functions:

- The frontal lobe is responsible for awareness, rational thinking, memory, emotion and its regulation, reasoning, decision-making, and speech.
- The temporal lobe contains the Wernicke area, which is crucial for understanding spoken and written language. It is also involved in smell, learning, and memory formation.
- The occipital lobe is primarily responsible for vision.
- The parietal lobe is involved in taste, sensation, vision, sensory integration, spatial awareness, and speech recognition.

Regarding the origin of emotions, the amygdala is the most crucial structure in emotional physiology. It is located deep within the temporal lobes of the brain, near the hippocampus, and is part of the limbic system, which is involved in processing emotions and memory. The amygdala modulates behavioral responses, fear, the intensity of emotion, and emotional memory. Emotions also arise from the prefrontal cortex, which, together with the amygdala, regulates the reward circuit. The right prefrontal cortex is associated with behavioral inhibition, while the left prefrontal cortex is linked to positive reinforcement. The insular cortex, situated between the frontal and temporal lobes, processes feelings of disgust and discomfort. Lastly, the hypothalamus acts as an emotional transducer, converting emotional information from the amygdala, insula, and prefrontal cortex into autonomic and endocrine responses [20]. Figure 1 shows these different brain areas.

2) Emotion model: Emotion is a psycho-physiological process resulting from the conscious or unconscious perception of an object or situation. It is often associated with mood, temperament, personality, disposition, and motivation. Emotions are triggered by a stimulus that generates a subjective experience. This subjective experience manifests in physical behaviors and physiological responses, such as facial expressions, changes in breathing, or alterations in heart rate [2].
Up to now, two main frameworks have been proposed to describe emotions: the discrete emotion space and the dimensional emotion space. In the discrete emotion space, emotions are categorized into a limited number of basic emotions. For example, Ekman [23] identified six basic emotions: sadness, happiness, surprise, fear, anger, and disgust. However, there is no consensus on the exact number of basic emotional states.

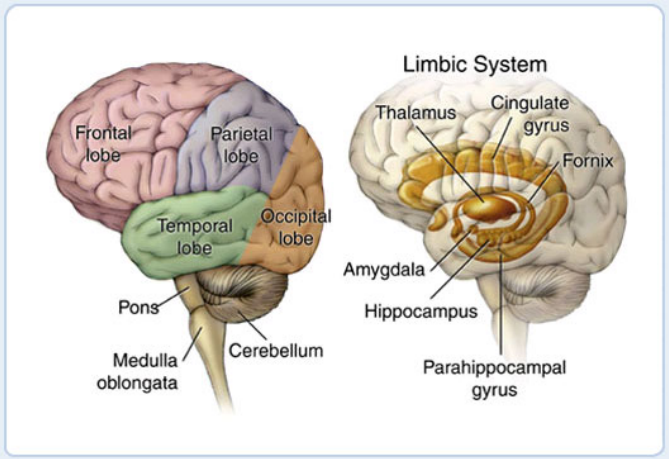


Fig. 1. Left panel shows the major lobes of the outer layer of the brain, and right panel shows some of the major brain areas internal to the brain. [22].

In contrast, the dimensional emotion space describes emotions along several dimensions, such as valence and arousal. This second framework is more flexible and extensive and it includes the discrete emotions. Figure 2 illustrates the popular valence–arousal–dominance dimensional emotion space, where valence refers to the degree of pleasure, arousal to the degree of intensity, and dominance to the degree of subjective control [4]. In this space, each emotion is represented by a set of coordinates; for example, the coordinates for ”fun” are 6.8571, 5.8571, and 6. The distance between points in this space can measure the similarity between the corresponding emotions. Compared to the discrete emotion model, which defines a limited range of emotions, the dimensional model offers a more continuous and accurate representation, making it better suited for capturing complex emotions. Consequently, dimensional models have become increasingly popular in the study of emotion recognition.

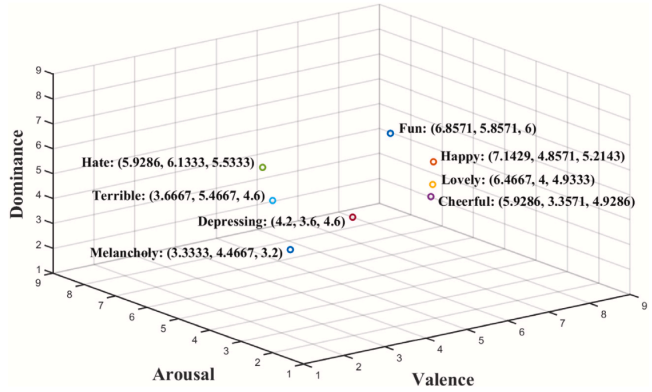


Fig. 2. Emotion distribution in Valence-Arousal-Dominance space [4].

B. Electroencephalography

The electroencephalogram enables the recording of the brain’s bioelectric activity by placing various electrodes on the scalp to detect the sum of positive and negative charges in their vicinity. Pyramidal neurons, primarily located in the outer layers of the cerebral cortex, generate large extracellular currents through synchronized synaptic activity, which can be detected by surface EEG.

The standardized physical placement (represented on Figure 3) and designations of EEG electrodes on the scalp are known as the 10-20 electrode placement system. In this system, the distance between adjacent electrodes is either 10% or 20% of the total front-back or right-left distance of the skull. Electrodes are labeled with letters that correspond to the brain areas they cover: F for frontal, C for central, T for temporal, P for parietal, and O for occipital. These letters are accompanied by odd numbers on the left side of the head and even numbers on the right side, based on the convention from the subject’s perspective [24].

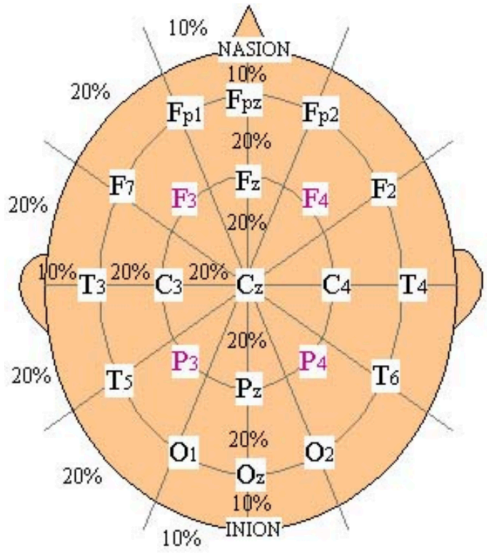


Fig. 3. Labels for points according to 10-20 electrode placement system [24].

EEG signals are traditionally divided into frequency bands associated with various states of consciousness and activity:

- Delta (0.1-3.5 Hz): Linked to deep sleep.
- Theta (4-7.5 Hz): Associated with drowsiness.
- Alpha (8-13 Hz): Present when awake with eyes closed.
- Beta (14-30 Hz): Increases during cognitive efforts.
- Gamma (30 Hz): Involved in sensory processing and intense concentration.

EEG provides a non-invasive technique with high temporal resolution (in milliseconds) and is generally more cost-effective than techniques like positron emission tomography (PET) or functional magnetic resonance imaging (fMRI) when studying brain activity compared to other methods. Additionally, unlike fMRI, some EEG devices are portable and do not exacerbate claustrophobia in patients.

C. Neural network architectures

1) Depthwise convolution: Depthwise convolution is a well-known technique in neural network architectures that

enables the reduction of computational complexity. It is often used in depthwise separable convolution, which is described in the next section (see Section II-C3). In depthwise convolution, each input channel has its own set of filters, whereas in a standard convolution, each filter is applied to all input channels [25]. Therefore, the number of output channels has to be a multiple of the number of input channels (where the multiplier will be the number of filters for each input channel). As a result, the total number of multiplications is reduced as follows:

- for standard convolutions:
  $F_1 \times F_2 \times C_{in} \times C_{out} \times H \times W$;
- for depthwise convolutions:
  $F_1 \times F_2 \times C_{in} \times \frac{C_{out}}{C_{in}} \times H \times W$
  $= F_1 \times F_2 \times C_{out} \times H \times W$;

where C_{in} and C_{out} are the number of input and output channels, H and W are the height and width of the input feature map and F_1 and F_2 are the height and width of the kernel. Figure 4 describes the functioning of this type of layer when the multiplier is set to 1.

Fig. 4. Scheme of a depthwise convolution layer [25].

2) Pointwise convolution: Pointwise convolution is the second part of the depthwise and separable convolution famous technique (see Section II-C3). It is a standard convolution with a kernel size of $(1, 1)$ that operates across the depth of the input feature map [25]. The number of multiplications is:
$1 \times 1 \times C_{in} \times C_{out} \times H \times W = C_{in} \times C_{out} \times H \times W$;
with the same notations as before. Figure 5 describes the functioning of this type of layer.

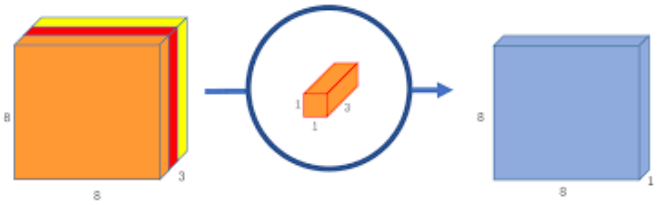


Fig. 5. Scheme of a pointwise convolution layer [25].

3) Depthwise separable convolution: Depthwise separable convolution is an efficient convolutional layer operation used to reduce the computational complexity and the number of parameters without significantly compromising the performance. It breaks down the standard convolution into two separate steps: depthwise convolution and pointwise convolution (see Sections II-C1 and II-C2). The most crucial benefit of depthwise separable convolution is its computational efficiency. Let us compare the number of multiplications required for a standard convolution versus a depthwise separable convolution:

- for standard convolutions:
  $F_1 \times F_2 \times C_{in} \times C_{out} \times H \times W$;
- for depthwise separable convolutions:
  $(F_1 \times F_2 \times C_{out} \times H \times W) + (C_{in} \times C_{out} \times H \times W)$
  $= ((F_1 \times F_2) + C_{in}) \times (C_{out} \times H \times W)$.

with the same notations as before.

4) Capsule network: Neural networks based on CNN have excelled in computer vision tasks such as classification, object detection, and semantic segmentation. However, CNNs have a notable limitation due to the pooling operation. Pooling, which is the downsampling operation of the feature maps obtained by convolution kernels, reduces computational complexity. Unfortunately, this downsampling erases important spatial relationships between high-level parts, which are essential for some recognition tasks.

To address this limitation, the capsule network (CapsNet) was proposed [26]. It was originally designed for and tested on the MNIST dataset [28], but quickly gained renown and has been applied to various fields and datasets such as emotion recognition with EEG data [29], [30], and [31]. CapsNet can represent the spatial relationships between local parts and the entire object. Its core units, called "capsules" [27], are groups of neurons that recognize visual entities and encode their properties into vectors. The length of the vector (ranging from 0 to 1) indicates the presence of the entity, while the orientation of the vector represents the instantiation parameters.

Capsules in different layers are connected through an iterative "routing-by-agreement" mechanism, where a lower-level capsule prefers to send its output to higher-level capsules whose vectors have a large scalar product with the prediction from the lower-level capsules. This mechanism together with transformation matrices in CapsNet encode the spatial relationship between a part and the whole object. Thus, CapsNet is able to overcome the limitation of CNNs caused by pooling.

Let us delve deeper into its implementation that is represented on Figure 6.

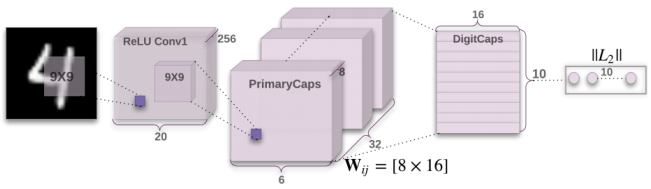


Fig. 6. Scheme of the CapsNet neural network [26].

After extracting a feature map from the input data thanks to convolution kernels, the latent representation of higher-level in groups of neurons is reshaped. These capsules are vectors of neurons, which means that the reshaping consists in adding a new dimension to the batch of latent representation of the

input considered. The length of this dimension is the length of the vectors considered that are now the "primary capsules". In Figure 6, the capsules are 8D and there are $k_1 = 32 \times h \times w$ of them. It is also necessary to define the number of "digit capsules" or output capsules and their dimension. The number of capsules corresponds to the number of classes for the classification task of our problem. In Figure 6 there are $k_2 = 10$ capsules because the authors worked with the MNIST dataset and the digit capsule dimension is 16.

For each primary capsule $\mathbf{u}_i (i = 1, ..., k_1)$, a transformation matrix $\mathbf{W}_{ij} (j = 1, ..., k_2)$ is defined to get the "prediction vector" $\hat{\mathbf{u}}_{j|i}$ from primary capsule $\mathbf{u}_i$ of the j^{th} output capsule. The calculation is:

$$\hat{\mathbf{u}}_{j|i} = \mathbf{W}_{ij} \mathbf{u}_i \tag{1}$$

Then, for each output capsule, a weighted sum of its inputs is computed:

$$\mathbf{s}_j = \sum_{i=1}^{k_1} c_{ij} \hat{\mathbf{u}}_{j|i} \tag{2}$$

The c_{ij} are the "coupling coefficients" that determine the relationship between the primary capsules and the output capsules. They define the relationship between a local part of an object and the entire object. The coupling coefficients between the i^{th} primary capsule and all the output capsules sum to 1. They are determined by a "routing softmax" where the initial logits b_{ij} represent the log prior probabilities that capsule i should be coupled to capsule j.

$$c_{ij} = \frac{\exp(b_{ij})}{\sum_{j=1}^{k_2} \exp(b_{ik})} \tag{3}$$

Finally, the probability that the entity represented by the capsule is present in the current input has to be represented by the norm of an output capsule. To achieve this, the authors used the "squashing", a non-linear function defined as follows:

$$\mathbf{v}_j = \frac{\|\mathbf{s}_j\|^2}{1 + \|\mathbf{s}_j\|^2} \frac{\mathbf{s}_j}{\|\mathbf{s}_j\|} \tag{4}$$

The coupling coefficients are initialized at 0 and then iteratively refined by measuring the agreement between the current output $\mathbf{v}_j$ and the prediction vector $\hat{\mathbf{u}}_{j|i}$ using their scalar product called "agreement". Then the coupling coefficients and the output vectors are computed again. The number of iterations is preset.

$$\begin{aligned} a_{ij} &= \hat{\mathbf{u}}_{j|i} \cdot \mathbf{v}_j \\ b_{ij} &= b_{ij} + a_{ij} \end{aligned} \tag{5}$$

5) The Transformer: The self-attention mechanism and the transformer model [32] are often referred to as state-of-the-art neural architectures in various deep learning problems. It is based on an encoder-decoder model containing the forward-looking attention mechanism:

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_K}})V \tag{6}$$

The matrices Q (queries), K (keys) and V (values) are obtained from the input via a linear layer. This calculation enables to catch the interactions within the sequence and to transmit information across data.

It is more efficient to linearly project the queries, keys, and values h times using different learned linear projections. The attention function in parallel on each of these projected versions of queries, keys, and values is then applied. These outputs are concatenated and projected again, resulting in the final values, as illustrated in the following equation. This method is called the multi-head self-attention (MSA).

$$\begin{aligned} &\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, ..., \text{head}_h) \\ &\text{where head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V) \end{aligned} \tag{7}$$

This mechanism is used as a part of the encoder and the decoder of the Transformer model. Both parts combine the self-attention mechanism and a multi-layer perceptron (MLP) with non-linear activation functions called feed-forward network. The multi-head self-attention and the feed-forward network are separated by residual operations and layer normalizations (LN). Layer normalizations enable a faster convergence and a more stable training and it is computed as follows:

$$\text{LN}(X) = \frac{X - \mathbb{E}(X)}{\sqrt{\mathbb{V}(X)}} \odot \gamma + \beta \tag{8}$$

where X is an input $\odot$ is an element-wise multiplication and γ and β are the gain and bias parameters that are learnable.

The encoder and the decoder can be made of several layers by stacking them through a repetition of this scheme.

The output of the encoder is sent to the decoder that also receives the output, but shifted "to the right". This means that for the first element of the sequence, there is no other input to the decoder than the output of the encoder. For the other elements, the decoder first uses the self-attention mechanism on the outputs of the preceding elements of the sequence and then on both these "self-attentioned" outputs and the encoder output.

For the model to utilize the sequence order, information about the relative or absolute positions of the tokens in the sequence must be present. To achieve this, the authors added "positional encodings" to the input embeddings of the encoder and decoder stacks. These positional encodings have the same dimension as the embeddings, allowing them to be summed. There are various options for positional encodings, both learned and fixed. The complete architecture of the Transformer model can be observed in Figure 7.

The encoder-decoder model is trained to predict the following element of a sequence. It can be used for many types of different sequential data such as EEG.

6) Swin transformer: The Swin Transformer [33] is a type of vision transformer that was originally designed to handle image data more efficiently and effectively than transformers. As for the Transformer model, it quickly gained renown and was applied to various research areas, including EEG! As mentioned earlier, the Transformer model is referred as the state-of-the-art architecture in various fields, however its computational complexity prevents it from being ubiquitous. Moreover, the Transformer model does not make the most

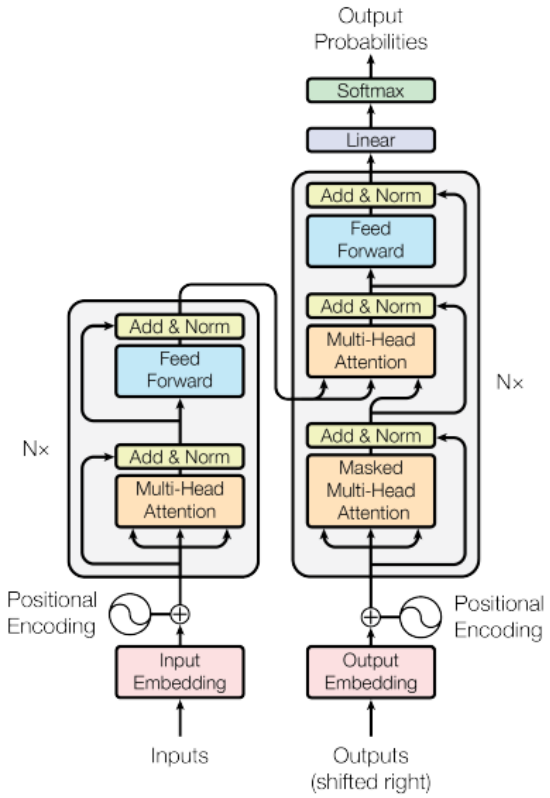


Fig. 7. The Transformer model [32].

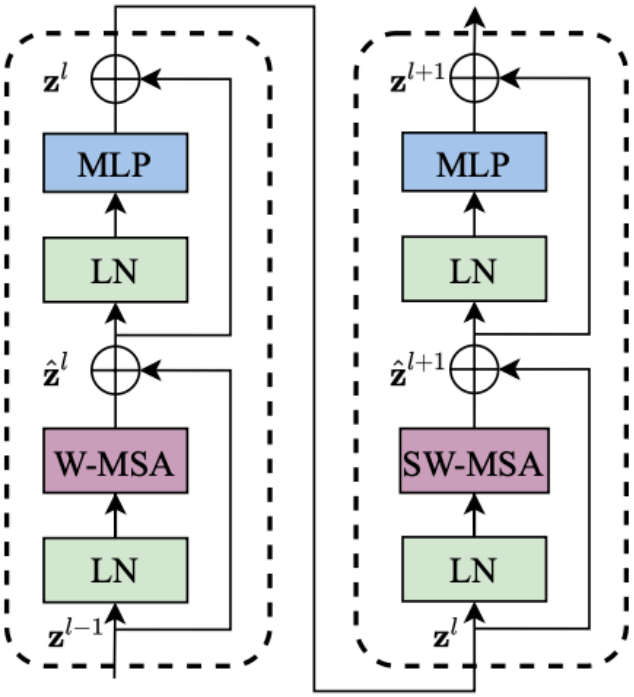


Fig. 8. The architecture of a Swin encoder [33].

of a famous principle in deep learning which is the hierarchical representation of features in higher levels. The Swin Transformer was designed to provide a solution to these two problems making it more scalable and efficient.

Firstly, this model partitions the input data into patches that stand as tokens in a sequential input data.

The Swin Transformer constructs hierarchical feature maps in a similar way as a CNN does. This allows the model to handle large input data by gradually reducing the spatial dimensions while increasing the number of feature channels.

As part of its process, the Swin Transformer divides the input into non-overlapping windows and processes each window separately, thereby reducing the computational cost. The "shifted window" mechanism shifts the window partitioning between layers. It enables the model to establish relationships between windows for the long-range dependencies across the entire image and to enhance its ability to capture the global context.

Within each window, the model applies a standard self-attention mechanism, reducing the computational complexity because of the smaller windows. Finally, the Swin Transformer uses layer normalization (LN) and a Feed-Forward network between the self-attention layers. This part is the Swin encoder which is schemed on Figure 8 and it is made of two successice Swin transformer block where the window is shifted in the second.

The hierarchical aspect of this model comes from the successive repetition of a merging of tokens using CNN and the Swin encoder with its shited self-attention mechanism described earlier. The full architecure of the Swin Transformer is represented on Figure 9.

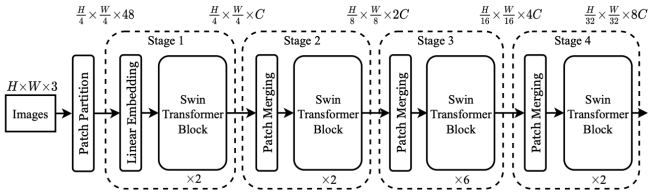


Fig. 9. The architecture of a Swin Transformer [33].

Additionally, the self-attention mechanism incorporates a relative positional bias:

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_K}} + B)V \qquad (9)$$

where $Q, K, V \in \mathbb{R}^{M^2 \times d}$ and $B \in \mathbb{R}^{M^2 \times M^2}$. M^2 is the total number of patches in a window. Since the relative position along each axis ranges from $-M + 1$ to $M - 1$, a smaller bias matrix $\hat{B} \in \mathbb{R}^{(2M-1)\times(2M-1)}$ is parameterized, and the values in B are derived from $\hat{B}$.

D. Signal processing techniques

1) Filtering: A digital filter is a system that processes digital signals using finite precision arithmetic and exhibits consistent behavior over time. Designing a digital filter involves three main steps:

- Defining the desired properties by specifying the goal of the filter, such as removing noise or isolating certain frequencies;

- Creating a causal discrete-time system approximation of these desired properties;
- Implementing the filter using precise numerical values.

There are two main categories of digital filters:

- Finite impulse response (FIR) filters: used when you need the output signal to have a linear phase, meaning all frequency components are delayed by the same amount, preserving the wave shape.
- Infinite impulse response (IIR) filters: used when linear phase is not critical. They are generally more efficient because they can achieve similar performance with fewer parameters, require less memory, and have lower computational complexity.

Thus, IIR filters are mainly used in EEG data preprocessing. The most common filters for this type of data are notch filters (also called band-stop, they remove a specific frequency range) and low pass, high pass and band-pass filters (they keep a specific frequency range of the signal and remove the rest) [34].

2) Continuous wavelet transform: Continuous wavelet transform (CWT) is a mathematical tool used to analyze localized variations of power within a signal. Unlike the Fourier transform, which provides frequency information but loses time information, the CWT provides a time-frequency representation of the signal.

It uses functions called "wavelets" which are small waves in a limited time span. These wavelets are scaled and shifted versions of a mother wavelet. The CWT works by scaling and translating the wavelet across the signal to cover it entirely and at various scales.

The CWT provides information about both the frequency content and the location in time where these frequencies occur. This is particularly useful for analyzing non-stationary signals where frequency components change over time like EEG. It analyzes the signal continuously, offering a highly detailed representation. Its computation is the following:

$$\mathrm{CWT}(a,b) = \int_{-\infty}^{\infty} x(t) \frac{1}{\sqrt{(|a|)}} \psi\left(\frac{t-b}{a}\right) dt \qquad (10)$$

where a and b are the scaling and translation parameter and ψ is the mother wavelet [35].

3) Empirical mode decomposition: Empirical mode decomposition (EMD) is a data-driven method for analyzing non-linear and non-stationary signals. It decomposes a signal into a set of intrinsic mode functions (IMF) which represent simple oscillatory modes embedded in the signal. They are components of the signal that each capture a single mode of oscillation. It involves two conditions:

- The number of extrema and the number of zero crossings must either be equal or differ at most by one;
- At any point, the mean value of the envelope defined by the local maxima and the envelope defined by the local minima is zero.

Unlike traditional methods that require predefined basis functions, EMD adaptively finds the basis functions directly from the data. The process involves iteratively extracting IMF from the signal. Each iteration sifts out the most oscillatory component, leaving behind a residue that represents slower oscillations. Here is the step-by-step sifting algorithm:

- Detect all the local maxima and minima in the signal;
- Create upper and lower envelopes by interpolating between the local maxima and between the local minima;
- Compute the mean of the upper and lower envelopes;
- Subtract the mean from the original signal to obtain an IMF (this process is repeated until the resulting function meets the criteria for an IMF);
- Subtract the extracted IMF from the original signal, consider the residue as the signal and repeat the steps to find the next IMF until the standard deviation of the new signal reaches a threshold.

In a dataset, all the input data need to have the same shape. Hence, if the sifting algorithm stops earlier for some data, they are padded with 0, which is inefficient if it occurs too frequently. This is avoided as much as possible by fixing the maximal number of extracted intrinsic mode functions close to the minimum of the number of possible IMF extractions across the dataset considered.

EMD is a powerful tool for decomposing complex signals into simpler components, making it easier to analyze and interpret their underlying structures for deep learning networks for instance [36].

E. Self-supervised learning

Self-Supervised Learning (SSL) is a machine learning approach that employs unsupervised algorithms like a masking approach or next step prediction tasks to produce initial parameters for a supervised model that are better than random. It consists in trying to predict a part of the input that is masked thanks to the other parts of the input. Self-supervised learning learns the inherent relationships and latent patterns within unlabeled data through auxiliary objectives. Furthermore, this approach prevents from the costly labeling process and fosters models with high adaptability, robustness and scalability.

Initially, it trains on unlabeled data to learn useful representations, which are then used to improve performance on subsequent classification tasks. It consists of two phases:

- The pretext phase that involves training on unlabeled data to generate parameters for the downstream model;
- The downstream task that involves fine-tuning the algorithm using labeled data.

There are three main types of pretext tasks in SSL: contrastive, generative, and hybrid.

- Contrastive models: the goal is to distinguish between different data augmentations, which can be done in either a supervised or unsupervised manner. In the supervised approach, the original and augmented data (this data is modified with a definite process) are pseudo-labeled as negative and positive, respectively, to identify common patterns. In this approach of contrastive learning, the goal is to differentiate or recognize dissimilar or similar data and then, to map far away the dissimilar ones and

close from each other the similar ones. In the unsupervised approach, models like autoencoders are used for reconstructing the original data from the augmented data. Although SSL has not been extensively studied for EEG, some augmentations for EEG include additive Gaussian noise, amplitude scale zero masking, time shift, direct current drift, and band-stop filtering, as suggested by neurologists to retain physiological meaning [37].

- Generative models: these models focus on reconstructing or generating input data points in an unsupervised manner. Common algorithms include autoencoders for reconstruction and Generative Adversarial Networks (GAN) for generating synthetic data. However, these models tend to be computationally expensive.
- Hybrid models: these models combine elements of both generative and contrastive approaches, leveraging the benefits of both supervised and unsupervised pretext tasks.

Overall, the feature representations obtained from the pretext task capture the general characteristics of the data and can be derived in either a supervised or unsupervised manner. For example, this can be interesting for catching the features of EEG across subjects, sessions or recording devices. Unlike traditional machine learning methods that are task-specific, features learned via SSL can potentially be applied across multiple tasks. This makes SSL particularly useful in low-data scenarios, such as in medicine where obtaining labeled data is costly.

III. Data collection and preprocessing

A. Datasets used

The most common technique for conducting emotion recognition studies is event-induced emotion stimulation. In this approach, participants passively experience emotional stimuli, such as images, music, and videos, to elicit specific emotions [1]. Visual or audio-visual stimuli are particularly favored in research due to their superior ability to evoke specific emotions and the accessibility of presentation tools. Additionally, there are freely accessible EEG databases for emotion recognition, such as DREAMER [8] and SEED [9], [10] that are used for the training and validation of the presented models.

- The SEED dataset comprises EEG signals and eye movements from 15 participants (7 males and 8 females). Each participant took part in the experiment over three different sessions on separate days, watching 15 Chinese film clips that had predefined emotional labels: positive, neutral and negative. Thus, this dataset agrees with the discrete model of emotions. EEG signals were recorded using the 62-channel ESI NeuroScan System at a sampling rate of 1000 Hz. The signals were then preprocessed with a bandpass filter from 0 to 75 Hz and downsampled to 200 Hz.
- The DREAMER dataset consists of EEG and ECG recordings from 23 subjects (14 males and 9 females) while they watched 18 movie clips. The recordings were made using the Emotive EPOC system, a 14-channel EEG device, with a sampling rate of 128 Hz. During preprocessing, eye artifacts are removed using linear-phase FIR filters. After viewing each clip, participants rated their emotions on three dimensions (valence, arousal, and dominance) using the Self-assessment Manikin (SAM) on a scale from 1 to 5 as in Figure 10 which is in alignment with the continuous model of emotions. Additionally, in this dataset, the authors recorded a 60 seconds record baseline before each patient exposure to a stimulus which is supposed to allow the subject to return to a neutral state of mind.

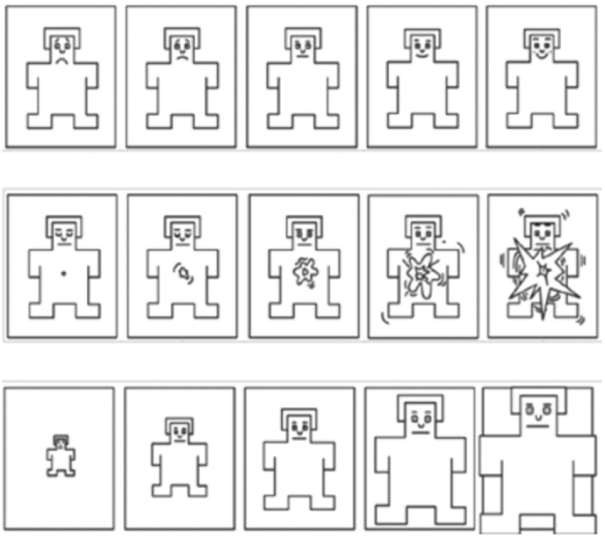


Fig. 10. Graphical representation of the SAM questionnaire. The emotion dimension are represented from top to bottom as follows: valence, arousal and dominance [21].

Table 1 summarizes the main characteristics of these two datasets for emotion recognition. Utilizing these databases facilitates the comparison of results across different studies.

B. Data preprocessing

Although some preprocessing steps were already applied on the EEG signals from the datasets, depending on the model used, additional techniques were implemented. The preprocessing of the dataset as all coding work for this project was done using Python.

1) The DREAMER dataset: For this dataset, additional filtering was found to improve the efficiency of the models. The parameters of the filter employed are optimized hyperparameters (see Section IV-E). It is a high pass Butterworth of order 3 with a frequency cut at 0.5 Hz that was obtained using the MNE python library [38]. The frequency response of the filter can be seen in Figure 11.

Let us consider $\mathbf{S} \in \mathbb{R}^{C \times T}$ the recorded EEG signal of one subject during one experiment where $C = 14$ is the number of channels and T is the sampling rate (128 Hz) multiplied by the length (in seconds) of the stimulus. In order to reduce the size of the input data to the models and to increase its efficiency,

Dataset	Number of subjects	Number of stimuli	Total recording duration	Type of stimuli	Emotion model	Assessment
DREAMER	23	18	80 minutes/subject	Film clips	Continuous	SAM
SEED	15	15 (×3 sessions)	45 minutes/subject	Film clips	Discrete (positive, neutral, negative)	Predefined

Table 1. Description of two available emotion recognition EEG datasets.

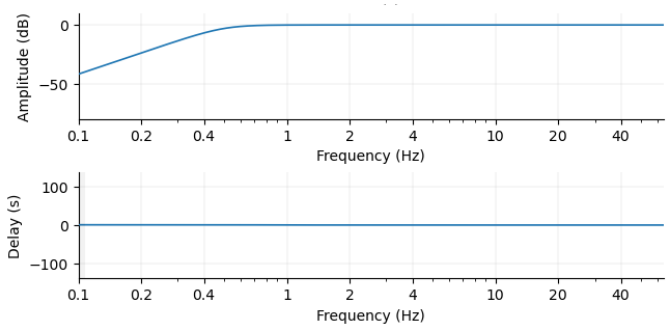


Fig. 11. 3^{rd} order Butterworth filter frequency response.

it is better to segment this matrix in the temporal axis. The choice of the length of the slices is a hyperparameter that was optimized as well as the others (see Section IV-E). The best duration for a sample as input for the models was found to be 1 second or $L = 128$ sampled points. Thus, for one experiment with one subject, there are $N = \lfloor \frac{T}{L} \rfloor$ generated inputs. This number is different for each stimulus as the video clips do not have the same length. The matrix is thus decomposed as follows:

$$\mathbf{S} \rightarrow \{\mathbf{S}^{(1)}, ..., \mathbf{S}^{(N)}\} \quad \text{where } \mathbf{S}^{(i)} \in \mathbb{R}^{C \times L} \tag{11}$$

The same is done for the matrices of the baseline. Let us consider the one corresponding to the baseline associated to the same couple $(subject, stimulus)$: $\mathbf{S}_b \in \mathbb{R}^{C \times T_b}$ where $C = 14$ and $T_b = 60 \times 128$. Therefore, $N_b = \lfloor \frac{T_b}{L} \rfloor = 60$. The result is:

$$\mathbf{S}_b \rightarrow \{\mathbf{S}_b^{(1)}, ..., \mathbf{S}_b^{(N_b)}\} \quad \text{where } \mathbf{S}^{(i)} \in \mathbb{R}^{C \times L} \tag{12}$$

The idea of this baseline segmentation is to then, remove the baseline features from the inputs because they are not related to the subjects' elicited emotion after their exposure to the stimulus. For this purpose, a common way to proceed is the following [29]:

- The average baseline sample is computed according to the following:

$$\overline{\mathbf{S}_b} = \frac{\sum_{k=1}^{N_b} \mathbf{S}_b^{(k)}}{N_b} \quad \text{where } \overline{\mathbf{S}_b} \in \mathbb{R}^{C \times L} \tag{13}$$

- The standard deviation baseline sample is computed according to the following:

$$\sigma_{\mathbf{S}_b} = \sqrt{\frac{\sum_{k=1}^{N_b} (\mathbf{S}_b^{(k)} - \overline{\mathbf{S}_b})^2}{N_b}} \quad \text{where } \sigma_{\mathbf{S}_b} \in \mathbb{R}^{C \times L} \tag{14}$$

- These average and standard deviation samples are removed from the experimental samples:

$$\mathbf{S'}^{(i)} = \frac{\mathbf{S}^{(i)} - \overline{\mathbf{S}_b}}{\sigma_{\mathbf{S}_b}} \quad \text{where } \mathbf{S'}^{(i)} \in \mathbb{R}^{C \times L} \tag{15}$$

An example with all the samples of a recording reconcatenated can be observed with one sample in Figure 12. It shows the inputs and the output of this preprocessing phase.

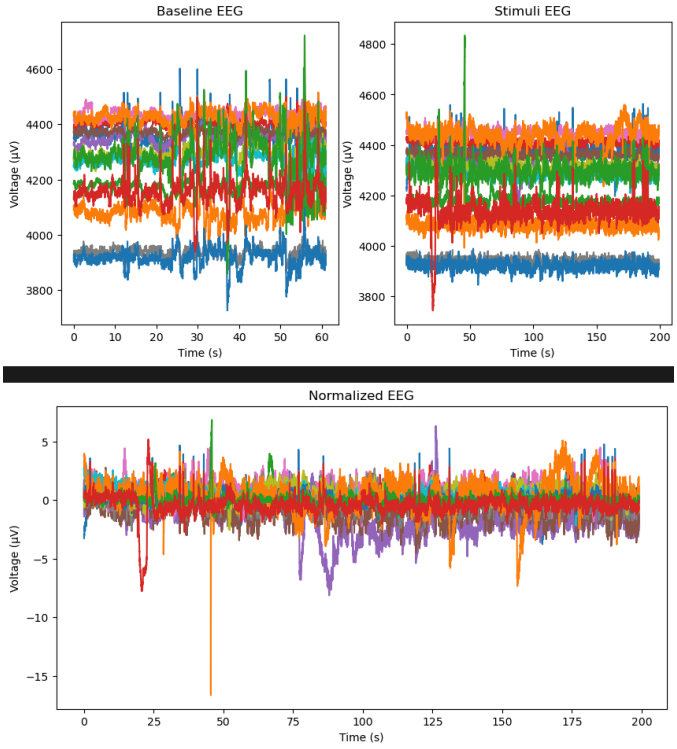


Fig. 12. Top: baseline and stimulus EEG recordings for all the electrodes. Bottom: Reconcatenated preprocessed samples.

After this preprocessing step a predefined number of the first samples $\mathbf{S'}^{(i)}$ is not considered. It seems logical because in the beginning of the exposure to a stimulus, emotions are not already elicited. Moreover, it yielded better results. The number of samples to skip is an optimized hyperparameter (see Section IV-E).

Finally, in some of the models developed, the data was also transformed in a time-frequency representation using the continuous wavelet transform or the empirical mode decomposition methodologies (see Sections II-D2 and II-D3).

As for the labels of this dataset, they are also preprocessed in many papers ([29], [30] or [31]). After each exposure to a stimulus, the patients had to self-assess their emotions on the valence, arousal and dominance dimensions using the SAM with a scale from 1 to 5. However, it is not an easy task to self-assess the emotions and errors could intervene.

Furthermore, moving from a 5-class classification task to a binary one could improve the model efficiency without losing its primary goal. The emotions of patients could still be analysed, with more accuracy, but less degrees if analysis. Thus a threshold is fixed at 3 and the labels 1, 2 and 3 are considered as "low" valence, arousal or dominance whereas the labels 4 and 5 are considered as "high".

2) The SEED dataset: For this dataset, no additional filtering were found to improve the efficiency of the models.

Furthermore, this dataset does not provide us with a baseline record before the patients exposure to a stimulus. Hence, the exact same process as that of the DREAMER dataset is applied to this one except that $\mathbf{S}_b$ is replaced by $\mathbf{S}$. This means that an "artificial" baseline was created based on the entire experiment. It makes the assumption that during an entire experiment which lasts around 200 seconds for the SEED dataset, the average elicited emotion is the neutral one. This is not very realistic for the stimulus that elicits positive and negative emotions, but this process happened to improve the result as a sort of data normalization.

The input data does not have the same shape in this dataset because there are $C = 62$ channels or electrodes and $L = 200$ sampled points. This also corresponds to 1 second of recording due to the 200 Hz sampling rate in the SEED dataset.

IV. Methodology

A. EEGNet

1) The original model: The first architecture established to approach this problem was based on a famous deep learning models specifically designed for EEG data [39]. It is a specialized convolutional neural network designed for EEG data classification, offering a lightweight architecture suitable for real-time applications and hardware with limited resources. It was originally designed to extract the features directly from raw EEG data. EEGNet is widely used in BCIs, neurological disorder diagnosis, and cognitive state monitoring due to these advantages.

This model implements brilliant ideas in neural network architecture such as depthwise and separable convolutions (see Section II-C3). On the one hand, it enables to considerably lower the number of parameters of the model without significantly decreasing its efficiency as explained previously. On the other hand, these layers enable to increase the explanability of the different layers of the model. Some convolutional layers are similar to spatial filters concerning the mapping of the electrodes while others are similar to time filters concerning the time series of individual electrodes. Figure 13 shows the architecture of the model as designed by its inventors. Another benefit of EEGNet is its flexibility in architecture customization because the number of layers, filter sizes, and other hyperparameters can easily be adapted to optimize the model for a particular dataset.

Initially, the model used $F_1 = 4$ or 8, $D = 1$ or 2, $F_2 = F_1 \times F_2$ and for the temporal kernels: $kernLength = \frac{sample\_rate}{2}$ to compute over a minimal frequency of 2 Hz.

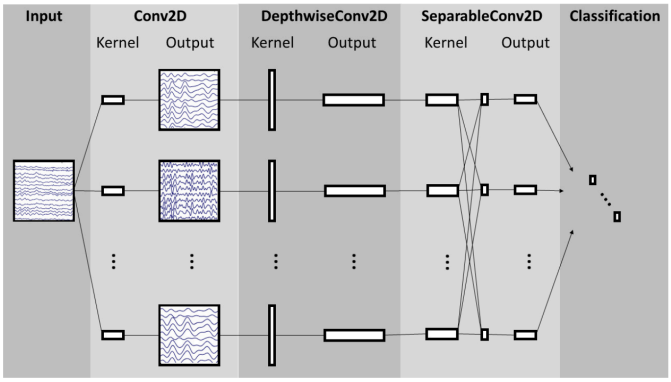


Fig. 13. Overall visualization of the EEGNet architecture. Lines denote the convolutional kernel connectivity between inputs and outputs. The network starts with a temporal convolution (second column) to learn frequency filters, then uses a depthwise convolution (middle column), connected to each feature map individually, to learn frequency-specific spatial filters. The separable convolution (fourth column) is a combination of a depthwise convolution, which learns a temporal summary for each feature map individually, followed by a pointwise convolution, which learns how to optimally mix the feature maps together. Full details about the network architecture can be found in Figure 14 [39].

Block	Layer	# filters	size	# params	Output	Activation	Options
1	Input				(C, T)		
	Reshape				(1, C, T)		
	Conv2D	F_1	(1, 64)	$64 * F_1$	(F_1, C, T)	Linear	mode = same
	BatchNorm			$2 * F_1$	(F_1, C, T)		
	DepthwiseConv2D	D * F_1	(C, 1)	$C * D * F_1$	(D * F_1, 1, T)	Linear	mode = valid, depth = D, max norm = 1
	BatchNorm			$2 * D * F_1$	(D * F_1, 1, T)		
	Activation				(D * F_1, 1, T)	ELU	
	AveragePool2D		(1, 4)		(D * F_1, 1, T // 4)		
	Dropout*				(D * F_1, 1, T // 4)		$p = 0.25$ or $p = 0.5$
2	SeparableConv2D	F_2	(1, 16)	$16 * D * F_1 + F_2 * (D * F_1)$	(F_2, 1, T // 4)	Linear	mode = same
	BatchNorm			$2 * F_2$	(F_2, 1, T // 4)		
	Activation				(F_2, 1, T // 4)	ELU	
	AveragePool2D		(1, 8)		(F_2, 1, T // 32)		
	Dropout*				(F_2, 1, T // 32)		$p = 0.25$ or $p = 0.5$
	Flatten				(F_2 * (T // 32))		
Classifier	Dense	N * (F_2 * T // 32)			N	Softmax	max norm = 0.25

Fig. 14. EEGNet architecture, where C = number of channels, T = number of time points, F_1 = number of temporal filters, D = depth multiplier (number of spatial filters), F_2 = number of pointwise filters, and N = number of classes, respectively. [39])

The main idea behind this model was to be small, easy to understand and efficient on various EEG data. The authors of this model proved its efficiency on various paradigms in Neuroscience such as P300 visual-evoked potentials, error-related negativity responses (ERN), movement-related cortical potentials (MRCP) and sensory motor rhythms (SMR). EEGNet outperforms traditional machine learning methods in all these EEG classification tasks. When it was published, this model quickly became a major work. Many have adapted it to their own experiments, as have researchers who try to extract patients' emotions from their electroencephalograms.

Regarding emotion recognition, this model has not been directly used. However, some papers tried to use some principles of this model mixed with features coming from other deep neural networks for EEG such as [29] that mixed it with a CapsNet (see Section II-C4).

2) The improvements brought: Nevertheless, as explained in the introduction, the objective of this project is to find a solution to both subject-dependent and subject-independent emotion classification that could be used by psychologists. The literature review showed that the subject-independent problem is more intricate and even the state-of-the-art solutions are not

good enough to be employed in the medical area. However, this is not the case for subject-dependent emotion classification where the state-of-the-art results are very satisfying.

An idea could be to indicate to therapists to train a new model for each patient instead of possessing a general pre-trained model that can perform on all patients. Unfortunately, the most efficient models are too complex and have too many parameters. This makes them not applicable for psychologists because they do not possess powerful hardwares.

Henceforth, a very light and efficient model on the subject-dependent emotion classification task seems a great solution to this current gridlock. Let us detail the progressive improvements brought to the model, with the evolution of the total number of parameters and accuracy on the datasets.

- Firstly, the original hyperparameters were optimized. In Section IV-E, the optimization process is detailed. Table 2 shows all the optimized hyperparameters mentioned in the following.
  - Among them, some hyperparameters are the ones present in every deep learning model which are the learning rate, the batch size, the number of training epochs, the loss function and the optimizer.
  - Others are specific to the dataset preprocessing like the sample size, the number of samples to skip. There also are hyperparameters concerning the high pass filter in the case of the DREAMER dataset with the order, the lowcut frequency and the type.
  - Finally, the hyperparameters specific to the EEGNet model are the number of temporal filters, the number of spatial filters or depth multiplier, the number of pointwise filters, the length of the temporal kernel, the dropout rate and the norm constraints of the second convolutional layer and of the final fully connected layer.

Hyperparameter	DREAMER value	SEED value
`lr`	0.01	0.01
`batch_size`	128	128
`epochs`	500	300
`loss_fn`	`CrossEntropyLoss`	`CrossEntropyLoss`
`optimizer`	`Adam`	`Adam` [40]
`sample_size`	128	200
`start`	1	1
`order`	3	`None`
`lowcut`	0.5	`None`
`type`	`Butter`	`None`
`F1`	64	64
`D`	8	8
`F2`	64	64
`kernLength`	A*:20, {D*,V*}:12	15
`dropout`	0.1	0.1
`norm_conv`	1.0	1.0
`norm_fc`	0.25	0.25

Table 2. Hyperparameters of the customized EEGNet model for the two datasets.
*: A, D, V respectively stands for arousal, valence and dominance

- Secondly, as the number of channels was much larger in the SEED dataset than in the DREAMER dataset and the optimizations resulted in similar hyperparameters, the total number of parameters was much larger for the model trained with the SEED dataset. Many methods in signal processing enable to increase or decrease the number of channels thanks to interpolation or filter. However, this would add a consequent step in the data preprocessing step and add imprecision to the data. The idea is thus to dynamically train a neural network layer that reduce this number of channels. Its goal would then be to find the best spatial filter to apply on the electrodes to interpolate them in a smaller number of channels. On the one hand, it reduces the total number of parameters of the model with the SEED dataset. On the other hand, it can increase the efficiency of the model because it adds an extra layer which adds complexity the model and also because with many electrodes that are close to each other on a patient's scalp, there is a lot of redundancy that might be useless for a deep learning model. The intermediate number of channels was also optimized.
The extra layer added in the beginning of the EEGNet model is a convolutional layer with no padding, the number of intermediate desired channels as the number of output channels and a kernel size with a temporal size of 1 and an electrode size depending on the dataset. A linear layer was also tested but it did not improve the performance of the model and it further increased the total number of parameters of the model. After the operation of these kernels on the data, the data dimensions have to be permuted to keep the shape $[B, N, C, L]$ where B is the batch size $N = 1$ is the number of features (which is 1 in the beginning of the deep learning model), C is the number of intermediate channels and L is the temporal length. An attempt to gather the label classes as explained in Section III-B1 was implemented as a hyperparameter in the data preprocessing step. In the DREAMER dataset, as the patients label the recordings themselves, even if the classes are grouped, the classes are imbalanced as shown in Figure 15 where the labels of attributed by all the subjects are shown.

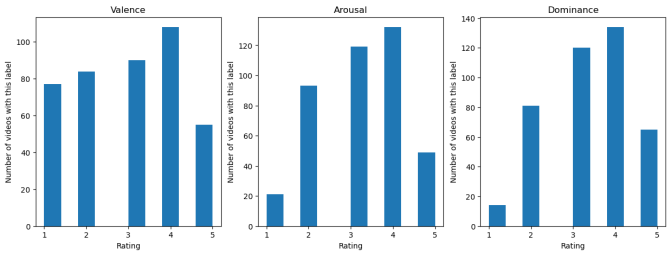


Fig. 15. Imbalanced classes for the DREAMER dataset.

This factor can be taken into account in the loss function by providing it with the various class weights and this parameter was also optimized.
This is also the case in the SEED dataset even though the labels of the stimulus were given by the designers of the experiments and they chose 5 video clips of approximately the same length for each categorical emotion. However, the video clips do not last exactly the same time which makes the classes slightly imbalanced as shown in

Figure 16.

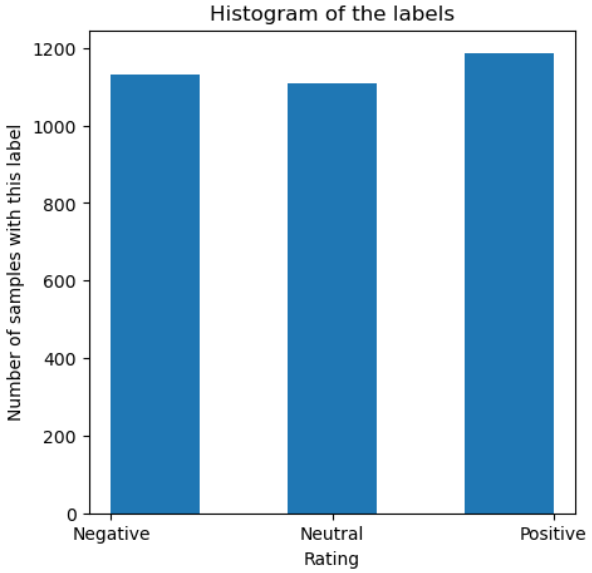


Fig. 16. Imbalanced classes for the SEED dataset.

All these parameters are in Table 3.

Hyperparameter	DREAMER value	SEED value
`innerChans`	18	24
`kernel_size`	$(14, 1)$	$(62, 1)$
`group_classes`	`False`	`None`
`class_weights`	`False`	`False`

Table 3. Hyperparameters in the first customized EEGNet model for the two datasets.

The added part of the EEGNet with intemediate channels architecture is represented on Figure 17.

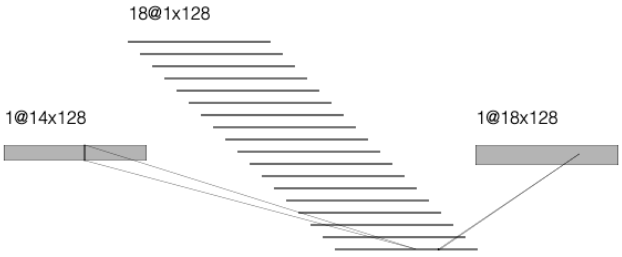


Fig. 17. Visualization of the added part in the beginning of EEGNet for one sample of the DREAMER dataset. Firstly, a convolutional layer is applied on the input and then, it is reshaped.

- Lastly, a time-frequency transformation of the data enables to add complexity to the input data and the model which can improve its efficiency. Moreover, as this preprocessing step keep a temporal dimension, the data can keep being used and understood by the EEGNet model. In the preceding cases, the input shape was $[B, N, C, L]$ where B is the batch size $N = 1$ is the number of features (which is 1 in the beginning of the deep learning model), C is the number of intermediate channels and L is the temporal length. From now on, N is the number of frequency decomposition of the input signal. In the first EEGNet model, it only changes the parameter about the number of input channels in the first convolutional layer. However, the idea is to combine this idea of time-frequency input data with the one about the intermediate channels. For this purpose, the idea is to reuse the fundamental principle of depthwise convolutional layer (see Section II-C1) in EEGNet to avoid introducing too much additional parameters. In this way:
  - The first original convolutional layer of the EEGNet architecture possesses the number of frequency decomposition as the number of input channels;
  - The added convolutional layer for the changes in the number of channels is turned into a depthwise convolutional layer where:
    - The number of input channels is the number of frequency decomposition;
    - The number of output channels is the number of frequency decomposition multiplied by the number of intermediate channels
    - The kernel size keeps being with a temporal length of 1 and an electrode length of the number of electrodes in the dataset considered.

  The number of frequency decomposition is also optimized depending on the dataset and on the time-frequency transformation used: continuous wavelet transform (see Section II-D2) or empirical mode decomposition (see Section II-D3). The current status of the optimization of these hyperparameters is shown in Table 4.

Hyperparameter	DREAMER value	SEED value
`innerChans`	18	24
`kernel_size`	$(14, 1)$	$(62, 1)$
`group_classes`	`False`	`None`
`class_weights`	`False`	`False`
`nb_freqs_cwt`	48	48
`nb_freqs_emd`	3	3

Table 4. Hyperparameters in the second customized EEGNet model for the two datasets.

The added part to the architecture of the EEGNet with intemediate channels and adaptation to the time-frequency transformation of the input data is represented on Figure 18.

B. The BENDR model

To classify raw EEG using deep learning models, these models must develop useful features from EEG signals and subsequently classify those features. This dual task frames both the possibility and challenge of using deep learning for supervised EEG classification. On the one hand, it promises to almost entirely prevent from the need for manual feature engineering. On the other hand, it requires both feature discovery and classification to be learned from a limited supply of high-dimensional data. Smaller neural networks

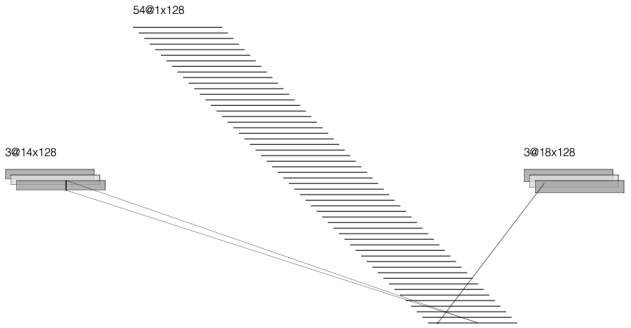


Fig. 18. Visualization of the added part in the beginning of EEGNet for one sample of the DREAMER dataset with the EMD technique. Firstly, a depthwise convolutional layer is applied on the input and then, it is reshaped.

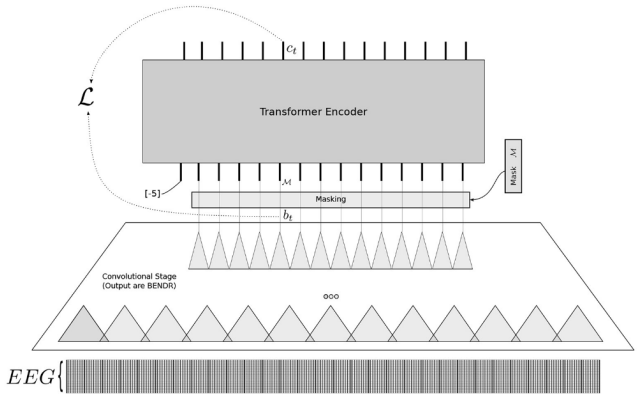


Fig. 19. The overall architecture used to construct BENDR. Loss L is calculated for a masked BErt-inspired Neural Data Representations (BENDR) bt (after masking, it is replaced by the learned mask M), itself produced from the original raw EEG (bottom) via a progression of convolution stages. The transformer encoder attempts to produce ct to be more similar to bt (despite that it is masked) than it is to a random sampling of over BENDR. [41]

like the previous ones (see Section IV-A) are very effective classifiers, especially when trained independently for each subject. However, these smaller networks are limited in the range of features they can learn, relying on a small number of constrained linear operations with few non-linear activation functions which are crucial for the complexity of the model.

1) The original model: The BErt-inspired Neural Data Representations (BENDR) [41] model is a cutting-edge approach designed to make the most of the power of transformers for EEG signal processing. It adapts the principles of BERT (Bidirectional Encoder Representations from Transformers) [43] from natural language processing to the domain of EEG. The BENDR model aims to create robust and compressed representations of raw EEG data by utilizing massive EEG datasets of unlabeled EEG recordings for pre-training.

In EEG, differences in domain are a critical challenge, as subjects and sessions can vary significantly, affecting classifier performance. Different tasks also require different features, further complicating the problem. The pre-training phase, contrastive using self-supervised learning (see Section II-E), helps the model learn general features and patterns in the EEG signals. These features can enhance the classification accuracy and offer broader insights about the data across subjects or even datasets. Then, the model can then be fine-tuned on specific tasks such as emotion recognition, sleep stage classification, or motor imagery. It makes the model robust to different devices, sessions and tasks. This transfer learning offers a framework to collect knowledge from one context which will benefit to another one. By incorporating a bidirectional attention mechanism, BENDR captures the complex temporal dependencies in EEG data, offering an improved performance over traditional methods.

BENDR draws inspiration from self-supervised end-to-end speech recognition. It adapts a self-supervised speech recognition framework called wav2vec 2.0 [44] to EEG, encoding arbitrary EEG segments into a sequence of learned vectors, called BErt-inspired Neural Data Representations. Figure 19 represents the architecture of the model.

In the pretraining process, the BENDR model predicts portions of EEG recordings from unlabelled audio using surrounding information. The model learns to map similar contexts closer in the embedding space while pushing apart dissimilar contexts. This approach with the self-attentioon mechanism enables the model to discern contextual information. Let us detail the architecture of this model.

The model receives as input the raw EEG data which are time series. In speech data, there only is 1 channel so the authors decided to implement a first 1D convolutional layer to turn the several electrodes channel into a unique one. Then, the model needs to reduce the size of the data to be able to process the entire signal and for this purpose, it applies convolutional layers (Convolutional Stage in Figure 19) to the input. It yields latent EEG representations of different continuous parts of the input in a lower dimensional space called BENDR. An input token is prepended to this sequence and a masking step with predefined masking probabilities on both the tokens and the features dimension occurs. The masking token is learnable.

In a second step, a transformer is applied to the latent EEG representations (Transformer Encoder in Figure 19). This allows the model to efficiently model long-range interrelations within the data, but it is not used directly on the raw input as it would be too costly computationally. The term "long-range" stands for the number of sampled points in the time series, but temporally speaking, they are quite close because the samples last few seconds. The model ends up with final latent EEG representations in a lower dimensional space of different part of the EEG input and it contains the context of the entire sample.

Let us center around the learning task which makes use of the quantization and the contrastive loss. The key step for the pre-training is the computation of the loss, which is computed as follows:

$$\mathcal{L}_m = -\log \frac{\exp(sim(c_t, b_t)/\kappa)}{\sum_{b_i \in B_D} \exp(sim(\mathbf{c}_t, b_i)/\kappa)} \quad \text{where } sim(a,b) = \frac{a^T \times b}{||a|| \times ||b||} \tag{16}$$

where c_t is the output of the transformer at position t, b_i is the

un-masked BENDR vector at some position i and B_D is a set of 20 uniformly selected distractors or negatives from the same sequence, including b_t. The κ factor adjusts the sensitivity of the cosine similarity.

The goal is to identify the true latent EEG representation for a masked output vector within a set of distractors. For each output vector c_t, the similarity with its BENDR b_t is maximized in terms of the cosine similarity. The similarity of the masked time step with the other latent EEG representations b_i is minimized. This requires the transformer to learn a sufficiently general model of BENDR so that the entire sequence of BENDR can effectively characterize position t. Additionally, to prevent the activations from becoming excessively large, the mean squared activation of the BENDR are added the loss function.

The pretraining dataset consists encompasses numerous subjects, each recorded over multiple sessions. These sessions span large time scales and involve a variety of tasks, providing a representative sample of EEG data. Additionally, the dataset include data from different recording hardware and configurations. It is the Temple University Hospital EEG Corpus (TUEG) [42]. This dataset comprises clinical recordings from over $10,000$ individuals, using mostly conventional recording configurations (monopolar electrodes in a 10–20 configuration), with some recording sessions separated by up to 8 months. The subjects are 51% female, and ages range from under 1 year old to over 90 years old. It amounts to approximately 1.5 TB of European-data-format (EDF) EEG recordings before preprocessing.

The authors of the BENDR model created 6 different configurations for the classification head in the fine-tuning of the model for downstream task. They are all represented in Figure 20 and there are:

- Fine-tuning with a new linear layer (Figure 20.1): A linear layer with softmax activation is added to the first output token of the pre-trained transformer. The entire model is fine-tuned, including the new classification layer;
- Using only the pre-trained convolutional stage (Figure 20.2): The pre-trained convolutional stage (BENDR) is used to create a consistent-length representation by averaging four contiguous sub-sequences. A new linear layer with softmax activation is added for classification;
- Training a randomly initialized DNN (Figure 20.3): Similar to Figure 20.1, but starting with a randomly initialized DNN instead of using pre-trained weights;
- Training only the transformer and classification layer (Figure 20.4): The BENDR weights are frozen, and only the transformer and new classification layer are trained;
- Using a randomly initialized convolution stage (Figure 20.5): Similar to Figure 20.2), but starting with a randomly initialized convolution stage;
- Training only the new classification layer (Figure 20.6): The weights of the first stage are kept fixed, and only the new classification layer is trained.

2) The improvements brought: Unlike the EEGNet model, the architecture of this model is not flexible at all, which does not leave a lot of space for additional work and potential

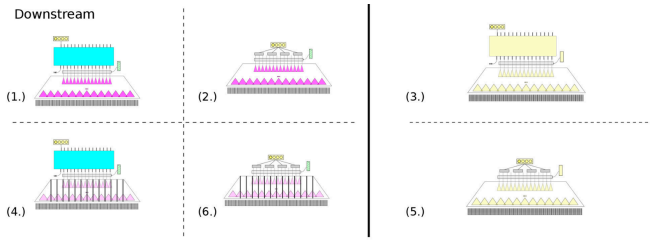


Fig. 20. Indicated here is the portion of the overall architectures used (see Figure 19), and how pre-training model weights were leveraged for a four-way classification task (rectangle with four circles in it). Four tasks (left half) leveraged model weights that were first developed through pre-training. All yellow modules here indicate randomly initialized weights. Color that progresses in intensity (from pre-training to downstream) indicates further training, while added bars indicate weights that were kept unchanged during that training stage. [41]

improvements. Firstly, it is programmed in a precoded library which is hard to access and to manipulate. Secondly, the model is a pretrained one which means that if the architecture is modified, the available to download pretrained model can not be loaded anymore. Pre-training with another dataset was not a conceivable emotion since the dataset used by the authors for the pre-training phase is few TB and the combination of the DREAMER and the SEED dataset is few GB. This means that the model can not be correctly pre-trained with these datasets and the interest of the efficient model across datasets and tasks is lost.

Nevertheless, the architecture was designed to receive EEG data with 19 electrodes from the UI 10-20 electrode placement system (see Figure 3). If fewer electrodes are provided, the input data is padded with 0 which is very inefficient. The same happens if the electrodes used in the fine-tuning dataset differs in position from the ones in the pre-training dataset. If more electrodes are provided, the extra electrodes are not considered whihc is also very unefficient. The solution brought consisted in the following:

- A spatial filtering with a Laplacian filter designed with the MNE python library enables to turn the electrodes that does not correspond to the ones of the pre-training dataset into ”correct” ones;
- Similarly to what was done for the EEGNet model (see Section IV-A2): the idea is to dynamically train a neural network layer that increase the number of channels. The extra layer added in the beginning of the EEGNet model is a convolutional layer with no padding, the number of intermediate desired channels as the number of output channels and a kernel size with a temporal size of 1 and an electrode size depending on the dataset. A linear layer was also tested but it did not improve the performance of the model. After the operation of these kernels on the data, the data dimensions have to be permuted to keep the shape $[B, N, C, L]$ where B is the batch size $N = 1$ is the number of features (which is 1 in the beginning of the deep learning model), $C = 19$ is the number of intermediate channels and L is the temporal length.

The pre-training of the BENDR model was done with raw EEG which implies that no signal processing transformation

could be applied on the data. For the data preprocessing the model applied its own methods including electrode channels rescaling to $[-1, 1]$ and a resampling of the data to 256Hz.

Moreover, as for the other models some hyperparameters were optimized (see Section IV-E) for the DREAMER and the SEED dataset and Table 5 shows their final values. Similar to the hyperparemeters of the EEGNet model, some refer to the training like the learning rate, the batch size and the number of epochs and some refer to the data like the size of the input data. Finally, the option for the fine-tuning of the BENDR model is also a hyperparameter.

Hyperparameter	DREAMER value	SEED value
`lr`	0.00001	0.0001
`batch_size`	64	64
`epochs`	50	30
`sample_size`	128	200
`ft_option`	2	2

Table 5. Hyperparameters of the customized BENDR model for the two datasets.

This model being very huge and designed for testing across subjects and satasets, the emotion classification using the BENDR model was only done across subjects.

C. Group Meiosis

1) The original model: The self-supervised group meiosis contrastive learning (SGMC) [45] framework aims at improving EEG-based emotion recognition, particularly in scenarios with limited labeled data. SGMC addresses the challenge by incorporating a contrastive learning task based on aligning EEG responses to stimuli across different subjects, thus reducing reliance on emotion labels and on subject-specific features.

The algorithm for the pretraining consists of five components: a group sampler, the Meiosis data augmentation (inspired from genetics), a base encoder, a group projector, and a contrastive loss function. Let us describe them.

- The group sampler: It creates mini-batches using the following process:
  - First, P videos or stimuli are chosen from the dataset to include in the mini-batch;
  - For each chosen video or stimulus, $2Q$ subjects are selected $(s_i, i \in [1, 2Q])$;
  - For each chosen video or stimulus, a group is created with $2Q$ different subjects.

  More specifically, if $X_{v,s}$ represents a 1-second signal recorded when a subject s watched a 1-second video clip v, a group G_k for a specific video v_k would be defined as follows:

$$G_k = \{X_{v_k,s_1}, ..., X_{v_k,s_{2Q}}\} \tag{17}$$

  Overall, a mini-batch consists of $\{G_1, ..., G_P\}$. The functioning of the group sampler is shown in Figure 21.
- The meiosis data augmentation: Positive pairs are created from each group obtained through the group sampler. A positive pair consists of two groups that maintain the same stimulus-related features. The augmentation process involves a crossover of two signals within the group, following this process:
  - Individual signals within the same group are paired to form Q pairs;
  - For each pair, both signals are split at the same random point in time. The splits are then combined such that the first split of one signal is combined with the second split of the other signal, and the other way around for the other splits. For example, with a split at time step c, with signals $A = \{a_1, ..., a_M\}$ and $B = \{b_1, ..., b_M\}$, the crossover will create two new signals: $A' = \{b_1, ..., b_c, a_{c+1}, ..., a_M\}$ and $B' = \{a_1, ..., a_c, b_{c+1}, ..., b_M\}$.
  - The recombined signals of a group are randomly divided into two subgroups, ensuring that the signals of each pair are placed in separate subgroups.

  As a result, the output of the meiosis data augmentation for a group G_k will be two subgroups, G_k^A and G_k^B. Therefore, a batch will consist of $2P$ groups: $\{G_1^A, G_1^B, ..., G_P^A, G_P^B\}$ where each group contains Q signals. The functioning of the meiosis data augmentation is shown in Figure 22.
- The base encoder: Each sample is then processed by an encoder based on the ResNet18-1D architecture, which maps each signal to a 512-dimensional feature space. The encoder begins with a convolutional layer followed by batch normalization and a ReLU activation. This is followed by a max pooling layer and then eight convolutional residual blocks. Each residual block consists of:
  - A convolutional layer;
  - Batch normalization and ReLU activation;
  - Another convolutional layer;
  - Batch normalization.

  Both convolutional layers within each residual block have the same number and length of kernels. The complete architecture of the base encoder is represented on Figure 23.
- The group projector: It is designed to extract group-level features from multiple samples. It includes a base

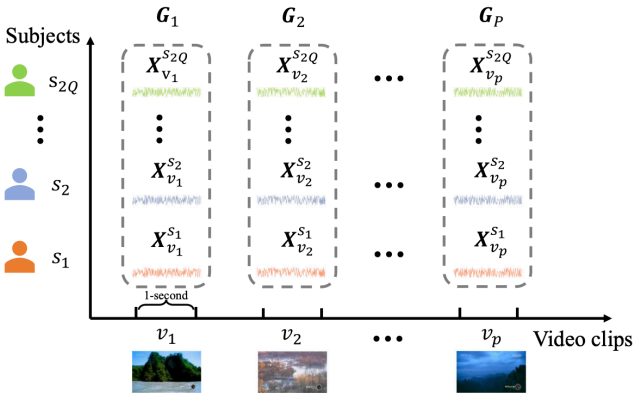


Fig. 21. The illustration of sampling for a batch. The sampler first samples P video clips and $2Q$ subjects. For each sampled video clip, the sampler next samples a group of EEG signals recorded when $2Q$ subjects watching it. Then P groups of EEG samples are obtained for a batch. [45]

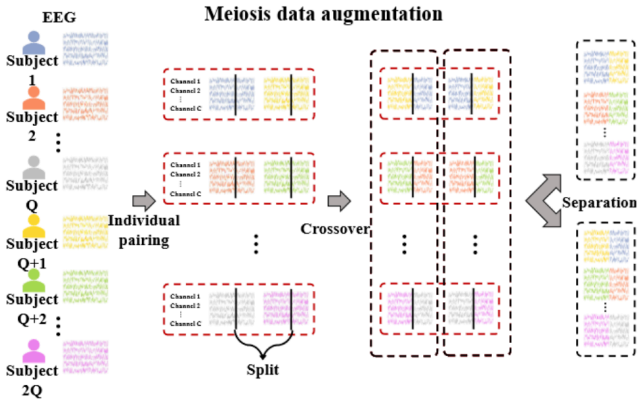


Fig. 22. The illustration of the meiosis data augmentation. A group of EEG samples sharing the same clip are randomly paired and cross exchanged a part of the signal in a pair, and then separated into two parts. [45]

projector which is an MLP that projects each individual representation into a 4096-dimensional space. The projector consists of three fully connected layers with ReLU activations after the first two layers. Additionally, batch normalization and dropout are applied before each layer to enhance performance.

After the base projector processes the individual representations, the resulting features are aggregated using 1-dimensional max-pooling (MaxPool1D). This pooling operation extracts the maximum values of the considered Q encoded representations for the 4096 features. As a result, a group of feature representations, which comprises the maximum values of each of the 4096 points across all the Q feature representations within a group, is obtained. This process integrates the individual signals within that particular group and computes a comprehensive group-level feature representation, mitigating subject-specific differences and random variations in EEG signals. The architecture of the group projector is shown in Figure 23.

- The loss function: The contrastive loss function is applied with the objective of maximizing the similarity between z_k^A and z_k^B, where z_k^A is the group feature representation of G_k^A and z_k^B is the feature representation of G_k^B obtained after the encoder and group projector. The algorithm focuses on maximizing the similarity of features corresponding to the same 1-second-long video clip. The formula is similar to Equation 16:

$$\mathcal{L} = \frac{1}{2P}\sum_{i=1}^{P}(l_i^A + l_i^B)$$

where:

$$l_i^A = -\log \frac{\exp(\frac{s(z_i^A, z_i^B)}{\tau})}{\sum_{j=1}^{P} \mathbb{1}_{[j\neq i]} \exp(\frac{s(z_i^A, z_j^A)}{\tau}) + \sum_{j=1}^{P} \exp(\frac{s(z_i^A, z_j^B)}{\tau})}$$

$$s(a,b) = \frac{a^T \times b}{||a|| \times ||b||} \tag{18}$$

where the τ factor adjusts the sensitivity of the cosine similarity.

Similar to the previous algorithm, the encoder from the pretraining phase is transferred to the downstream phase with the learned weights. This encoder extracts the feature representation of each 1-second long signal, which is then sent to the classifier.

The classifier consists of several fully connected layers. The first two layers are followed by batch normalization and ReLU activation functions. Before the final layer, a dropout of 0.5 is applied to reduce overfitting. The architecture of the classification head is represented in Figure 23. The algorithm is optimized using the cross-entropy loss function, which measures the difference between the predicted output and the actual label during training.

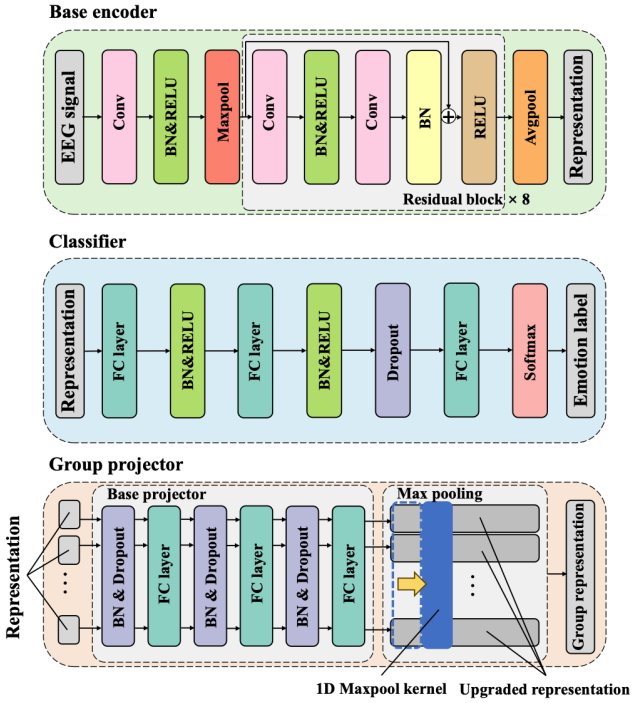


Fig. 23. The architectures of the base encoder, classifier and group projector. [45]

2) The work in progress: SGMC achieves competitive performance on subject-dependent emotion classification. Some feature visualizations show that the model is able to capture video-level emotion-related representations, validating the robustness of its architecture. The authors already used the SEED dataset for subject-dependent classification in their testing, but not the DREAMER dataset.

However, the purpose of working on this model in this project was not to modify the architecture or the model or the preprocessing of the data. The model used is very huge and the meiosis data augmentation method together with the group projector enables this methodology to catch group-level features while omitting subject-specific specificities. This aspect can be very useful for subject-independent emotion classifications. The work done regarding this SGMC model

consisted in preprocessing the data in order to pretrain, train and test across the subjects and optimizing a few hyperparameters.

Here, hyperparameters like the learning rate, the batch size and the number of epochs exist for both the training and the pretraining. There also are hyperparameters specific to the model that can be changed like the number of videos and subjects in the group sampler and the sample size (the value of the authors of 1 second was reused). Table 6 shows the current status of this hyperparameters optimization.

Phase	Hyperparameter	DREAMER value	SEED value
Pretraining	`lr`	0.0001	0.001
	`batch_size`	32	64
	`epochs`	2500	3200
	`tau`	0.1	0.1
	`P`	8	16
	`Q`	2	2
Training	`lr`	0.0001	0.0001
	`batch_size`	64	64
	`epochs`	80	100
	`sample_size`	128	200

Table 6. Hyperparameters used in the SGMC methodology for the two datasets.

D. The TC-Net model

1) The original model: To address the limitation of the CNN, the Transformer Capsule Network (TC-Net) [31] which features an EEG Transformer module for extracting EEG features and an emotion capsule module for refining these features and classifying emotions. Let us review the operational process of this model.

- The signal preprocessing: This model uses as input for the deep learning model, time-frequency transformed data, this is what inspired the current last version of the customized EEGNet model (see Section IV-A2). Firstly, it decomposes and normalizes the signal as detailed in Section III-B1 but without the division by the standard deviation of the baseline. Then, it applied the continuous wavelet transform signal processing technique detailed in Section II-D2. The pre-processed signals have the shape $[C, F, L]$ where C represents the number of EEG electrodes, $F = 48$ indicates the frequency resolution, and L denotes the temporal resolution. This preprocessing phase is represented in Figure 24. Moreover, this model gather the labels of the classes in the DREAMER dataset into 2 classes as explained before (see Section III-B1)
- The patch partition: The EEG data are divided into small patches using a convolution operation. The patch size is set to $(3, 4)$ based on the frequency and temporal resolution. The kernel size and stride in the convolution operation are both set to match the patch size, ensuring that the EEG signals for each channel are split into non-overlapping neighboring patches, while preserving their frequency and temporal characteristics. The number of convolutional kernels is set to 32. Each patch, treated as a token, captures the primary features of the EEG signals.

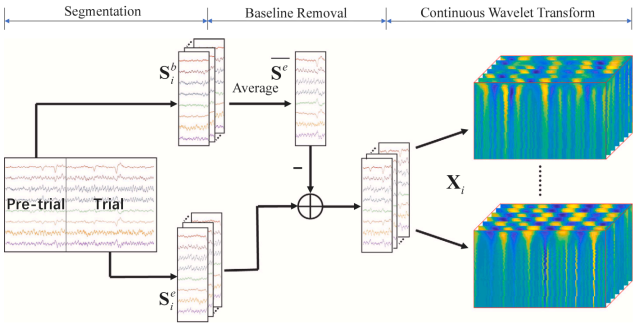


Fig. 24. The flowchart of the signal pre-processing approach, which consists of three steps including segmentation, baseline removal and continuous wavelet transform. [31]

- The EEG Transformer: This module comprises four main components: the window partition block, the Transformer block, the window reverse block, and the patch merging block. Here are their description.
  - In the window partition block, tokens are divided into a sequence of feature windows in a non-overlapping manner to ensure efficient computation. Each feature window is sized $(2, 2)$, containing four neighboring tokens.
  - The Transformer block captures relationships between different feature windows using the multi-head self-attention mechanism, MLP layers and layer normalization (see Section II-C5). It draws inspiration from the Swin Transformer model (see Section II-C6) with a position embedding to incorporate positional information where $B \in \mathbb{R}^{M^2 \times M^2}$ is the relative position bias and M is the number of tokens in each window.
  - The window reverse block reverses the process of the window partition block, converting the sequence of feature windows back into tokens.
  - The patch merging block is used to create a hierarchical representation of EEG signals as the Swin Transformer does (see Section II-C6). To preserve local, EEG Patch Merging (EEG-PM) is proposed. EEG-PM has a large receptive field and better captures local EEG features. It is implemented as a convolutional operation with a kernel size of $(4, 4)$, stride of $(2, 2)$, and padding of $(1, 1)$. The number of output channels is set to twice the number of input channels, similar to the Swin Transformer. Consequently, the number of feature maps is doubled, while the resolution is halved.

  The EEG Transformer module is cycled four times to extract high-level emotion features, with the final feature map for classification outputted before the last patch merging block.
- The Emotion Capsule module: This module is an implementation of CapsNet (see Section II-C4). To effectively capture the relationships between different channels of the feature map, the number of capsules is set to match the number of channels. The EEG features are first

reshaped into capsules, each containing 8 neurons. The dynamic routing-by-agreement mechanism is then used to identify the intrinsic relationships between the feature map channels. Finally, the L2-Norm of each capsule is computed to produce the classification results. The margin loss function is applied to guide the optimization process.

The complete data processing after the preprocessing phase of this TC-Net framework can be observed in Figure 25.

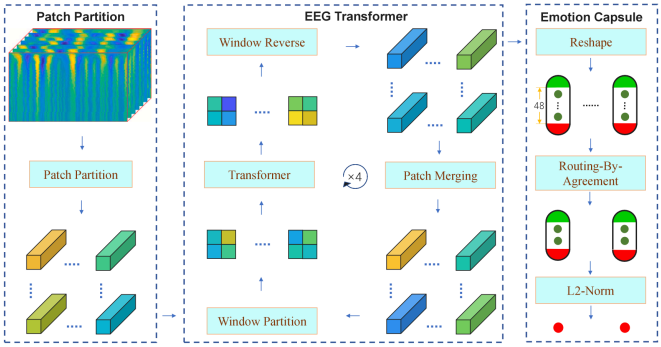


Fig. 25. The architecture of our Transformer capsule network (TC-Net) for EEG emotion recognition. Our TC-Net contains three main modules: the patch partition module for data preparation, the EEG Transformer module for feature extraction, and the emotion capsule module for emotion classification. [31]

2) The work in progress: Experiments showed that TC-Net achieves state-of-the-art performance in subject-dependent emotion classification tasks, with impressive average accuracies for valence, arousal, and dominance dimensions on the DREAMER dataset.

The suggested work around this model comes mainly from the fact that this model is currently the state-of-the-art and that it seems to be possible to optimize some part of it. Firstly, it is possible to replace the continuous wavelet transform technique (see Section II-D2) in the preprocessing phase with the empirical mode decomposition (see Section II-D3) one. Indeed, while keeping a time-frequency transformation of the data, the latter adds less computing imprecision to the data and provides a slightly different to the model. However, the frequency resolution is smaller when using EMD because of the threshold in the sifting algorithm. In order to keep the efficiency of the following patch partition, EEG transformer and patch merging modules, a convolutional layer was added in the beginning of the model to increase the frequency resolution. A reshaping of the data was also necessary to introduce as it is represented in Figure 26.

There is also a possibility to replace this convolutional layer by a depthwise one even though in this case, the extra number of parameters added via this layer is insignificant compared to the total number of parameters of TC-Net.

Moreover, it seems redundant and computationally expensive to have both the hierarchical transformer and the CapsNet. A suggestion is to replace the CapsNet by a simpler and lighter classification head like an MLP for example. Finally, the authors did not mention any dropout layer to avoid over-

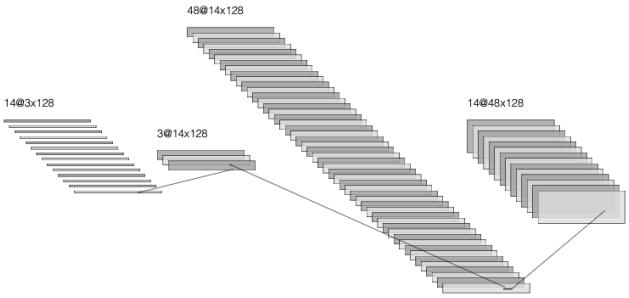


Fig. 26. Visualization of the added part in the beginning of TC-Net for one sample of the DREAMER dataset with the EMD technique. Firstly, the input is reshaped, then a convolutional layer is applied on it and finally, it is reshaped.

fitting so it was implemented in this project between the EEG transformer module and the emotion capsule.

There also are a few hyperparameters currently being optimized regarding the training with the learning rate, the batch size, the loss function and the number of epochs. There are those regarding the architecture of the model like the dropout ratio or the type of transformer module because it is not mentioned in the article. Finally, those regarding the data preprocessing with the sample size and the frequency resolution. Table 7 shows the current status of this hyperparameters optimization.

Hyperparameter	DREAMER value	SEED value
`lr`	0.0001	Coming soon
`batch_size`	32	Coming soon
`epochs`	5000	Coming soon
`loss_fn`	`margin_loss`	Coming soon
`transformer_type`	`swin`	Coming soon
`dropout`	0.1	Coming soon
`sample_size`	128	Coming soon
`frequency_resolution`	48	Coming soon

Table 7. Hyperparameters used in the TC-Net methodology for the two datasets.

E. Hyperparameter optimization

This part focuses on hyperparameter optimization which is a crucial aspect of machine learning that involves seeking the best set of hyperparameters to improve the performance of a model. Unlike model parameters, which are learned during the training process, hyperparameters are set before training and can significantly influence the training dynamics and final performance of the model. In this project, this optimisation phase was done using the python library Ray Tune [46].

The first step in hyperparameter optimization is to define the search space which includes the range of values or discrete choices for each hyperparameter. Various strategies can be employed to explore the search space and this project implemented the following.

- Searching strategy: Grid search which exhaustively searches through the search space was used. This ensured to find the best hyperparameters but can be computationally expensive. Sometimes the search space was

subdivided into multiple smaller (in terms of the number of hyperparameters considered) search space. Random Search which randomly samples hyperparameters from the defined search space was also employed. It often finds good hyperparameters faster than grid search.
- Scheduler: Asynchronous successive halving algorithm which efficiently allocates resources to configurations, rapidly eliminates poor performing configurations and allocates more resources to promising configurations was used.

The objective function evaluates the performance of the model with a given set of hyperparameters. While training the model, the evaluation of the performance was done on a validation set with the mean accuracy metric as it is the one used in the literature for emotion classification. Cross-validation was implemented to ensure the reliance on the selection of the hyperparameters. Thanks to this process, good hyperparameters were selected and improved the performance of the various models.

There is a python script that iteratively trains models with different hyperparameter settings, reports the performance, and Ray Tune tracks these to identify the best-performing configurations. At the end of the tuning process, the best model's state is saved for further use and its configuration is reported.

This systematic approach ensures that a configuration close to the best possible model is found, improving the performance and robustness of the model on the task of emotion classification using EEG data.

In Appendix VI, there are figures representing the accuracy on the validation set as a function of the number of epochs for different configurations of the same search space. The configurations that do not perform well are stopped earlier.

V. Results and Discussions

All the results are not present as this report is written before the end of the internship. Moreover, the more work that was achieved, the longer the computations took.

A. About the number of parameters vs performance

As explained in Section IV-A2, an alternative to the subject-independent classification is the following. On the one hand, trained models on subject-independent scenarios whose weights can, in theory, easily be loaded by a therapist do not yield sufficient performance for medical applications. On the other hand, state-of-the-art models demonstrate excellent results for subject-dependent scenarios, but their high computational requirements make them impossible to train by psychologists who would face an issue with new patients. Therefore, the solution involves creating a lightweight model that maintains high accuracy for individual patients while being computationally feasible for use on standard devices.

In Tables 8 and 9, some models are present together with their relevant aspects regarding the current section: their accuracy and their number of parameters. The results and comparison are different for the two datasets. The comparisons are made with state-of-the-art models. To obtain the following results and to be able to compare them with the results from the literature, a 10-fold cross-validation was implemented. The data is divided in 10 groups, and for each group, the model is trained with the others and then, tested with this group. The average of these 10 configurations provides the average accuracy per subject, which is then averaged to yield the final results.

It is crucial to note that the variants of the EEGNet model developed during this project had approximately the same performance on both the binary and the 5-class classification tasks of the DREAMER dataset. If the fact that there could be errors in the self-assessment of emotions by the subjects is omitted, this means that these models can generate many degrees of analysis for patients' emotions. Figure 27 represents the confusion matrices for all the subjects and their validations of this 5-class classification task on arousal, dominance and valence for the DREAMER dataset.

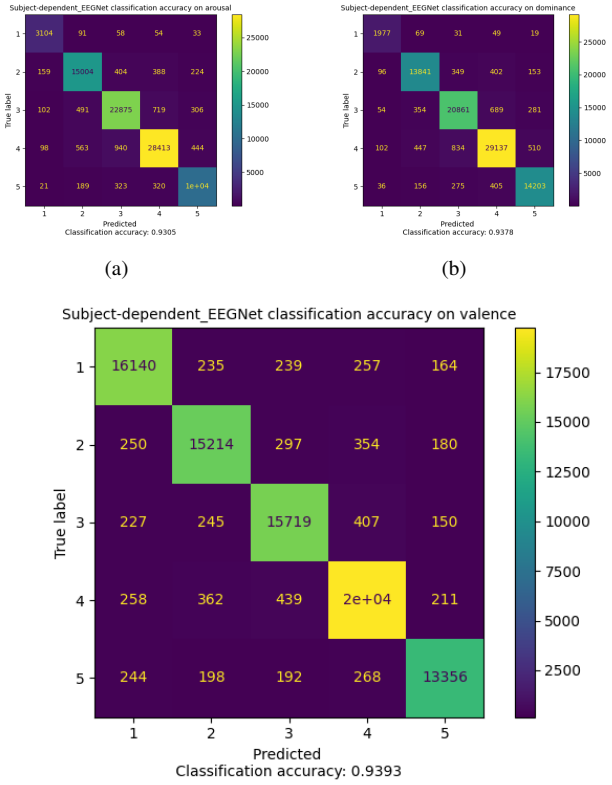


Fig. 27. The summed confusion matrices for all subjects and validations on arousal (a), dominance (b) and valence (c) on the DREAMER dataset.

It is also necessary to point out the fact that even if the results are promising for the EEGNet with EMD or CWT and their number of parameters are low, the time-frequency transformation of the data in the preprocessing phase adds some computational time.

Finally, the models used for comparison are the state-of-the-art with the current best results. Hence, it can be observed that the models developed reach great accuracies on these two datasets. They compete with the performance of state-of-the-art models in terms of accuracy and surpass them in terms

Model	Number of parameters	Accuracy on arousal	Accuracy on dominance	Accuracy on valence
EEGNet	$1.5k$	$69.98 \pm 2.96\%$	$71.23 \pm 2.80\%$	$70.12 \pm 2.78\%$
EEGNet with optimization*	$51k$	$91.89 \pm 2.55\%$	$91.90 \pm 2.61\%$	$93.01 \pm 2.57\%$
EEGNet with inner channels*	$53k$	$93.34 \pm 2.28\%$	$93.81 \pm 2.09\%$	$94.03 \pm 2.0\%$
EEGNet with EMD*	$56k$	No final	results	yet
EEGNet with CWT*	$114k$	No final	results	yet
Caps-EEGNet [29]	$1.1M$	$92.60 \pm 5.10\%$	$93.74 \pm 5.64\%$	$91.12 \pm 3.82\%$
TC-Net	$6.0M$	$\mathbf{98.61 \pm 1.34}\%$	$\mathbf{98.67 \pm 1.57}\%$	$\mathbf{98.59 \pm 1.38}\%$
MLF-CapsNet [30]	$3.5M$	$95.26 \pm 3.63\%$	$95.13 \pm 3.81\%$	$94.59 \pm 3.77\%$
JDAT [7]	$\approx M$	$98.61 \pm 1.34\%$	-	$98.01 \pm 2.13\%$

Table 8. Number of parameters and performance of various models on the DREAMER dataset on the subject-independent scenario.
*: models that were designed in this project

Model	Number of parameters	Accuracy on {Negative, Neutral, Positive}
Original EEGNet	$2.6k$	$85.34 \pm 2.79\%$
EEGNet with optimization*	$76k$	$96.90 \pm 0.88\%$
EEGNet with inner channels*	$59k$	$\mathbf{98.56 \pm 0.70}\%$
EEGNet with EMD*	$65k$	No final results yet
EEGNet with CWT*	$205k$	No final results yet
SGMC	$27.6M$	94.96%
ACTNN [13]	$\approx M$	$98.47 \pm 1.72\%$
JDAT [7]	$\approx M$	$97.30 \pm 1.74\%$

Table 9. Number of parameters and performance of various models on the SEED dataset on the subject-dependent scenario.
*: models that were designed in this project

of the number of parameters. For the SEED, even if it is far smaller than the state-of-the-art model, the EEGNet model with the reduction of channels and the optimized hyperparameters is the new state-of-the-art. The summed confusion matrix of all validations and all subjects is obtained and shown in Figure 28.

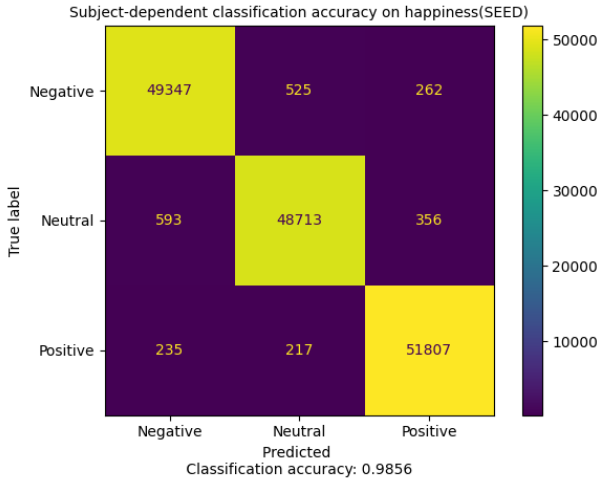


Fig. 28. The summed confusion matrix for all subjects and validations on the SEED dataset.

A possible consequence of this result is that unlike what is said in the literature, the emotion classification problem is not very complex. Henceforth, a well-adapted light model can well perform on this task without being too outperformed by some huge models.

In Appendix VI, figures represent the accuracy for the different subjects.

B. The subject-independent classification task

In this part, no more attention is drawn on the number of parameters. This project aimed at working on both the subject-dependent and the subject-independent classification problem applied to the medical field for psychologists. An alternative to the subject-independent scenario was suggested in the preceding section, but the real scenario will now be targeted.

On the one hand, the variants of the EEGNet model are poorly performing on this classification task. On the other hand, throughout this project, a lot of work was achieved on model using self-supervised learning, but the results obtained are not very satisfying and do not compete with the state-of-the-art as it can be seen in Tables 10 and 11. To obtain the following results and to be able to compare them with the results from the literature, a leave-one-subject-out method was implemented. For each subject, the model is trained with the data of the other subjects and tested with its data. The accuracy of all the subjects are averaged to give the following results.

The chance level of the DREAMER scenario is at 50% if the 5 classes are gathered into 2 which is what is done in the literature and in this project for comparison. The chance level is at 20% in the other case. For the SEED dataset which has 3 classes, the chance level is at 33%. It can be observed that the subject-independent task seems easier on the SEED dataset than on the DREAMER dataset. This can be due to a too large difference in the EEG recording set-up between subjects in the case of the DREAMER dataset for example.

Model	Accuracy on arousal	Accuracy on dominance	Accuracy on valence
EEGNet with inner channels*	$53.4 \pm 5.56\%$	$52.2 \pm 7.98\%$	$50.5 \pm 5.49\%$
Adapted SGMC*	No final	results	yet
Customized BENDR*	$59.75 \pm 9.78\%$	$59.01 \pm 9.45\%$	$51.94 \pm 5.12\%$
AD-TCN [19]	$63.69 \pm 6.57\%$	-	$66.56 \pm 10.04\%$

Table 10. Performance of various models on the DREAMER dataset on the subject-independent scenario.
*: models that were designed or reworked in this project

Model	Accuracy on {Negative, Neutral, Positive}
EEGNet with inner channels*	$56.61 \pm 5.57\%$
SGMC	$58.37 \pm 5.18\%$
BENDR	$56.21 \pm 4.00\%$
CLISA [17]	$77.4 \pm 13.4\%$

Table 11. Performance of various models on the SEED dataset on the subject-independent scenario.
*: models that were designed in this project

The models used for comparison are the state-of-the-art with the current best results.

A consequence of these poor results is that despite the accessibility of the subject-dependent emotion classification problem and the feasibility of its solution, the subject-independent one is completely different. It seems necessary to extract emotion-specific features because the BENDR model was not efficient, although it is supposed to be a highly generalizable model across tasks and subjects. Furthermore, even with an innovative data augmentation like the group meiosis one which help to focus on group-level features and a sizeable model, it is intricate to extract the inherent feature of an emotion regardless of the specificity of a subject. Indeed, this framework was only performing well on the subject-dependent scenario.

Figures representing the accuracy for the different subjects were not created due to the disappointing quality of the results obtained.

The state-of-the-art results on the emotion classification using EEG task across subject are good but perhaps not convincing enough to assert that emotions follow a generalized pattern across humans and that EEG data is sufficient to interpret them using signal processing and machine learning.

C. Looking for a new state-of-the-art

In this part, the result about the modifications brought to the TC-Net framework will be detailed. Unfortunately, the work on this model and the testing computations are not finished yet, so they can not appear on this report. Nevertheless, they seem promising and could potentially lead to a publication in a scientific journal according to the supervisor.

The current results oriented the research towards the Swin Transformer (see Section II-C6).

VI. Conclusions

The integration of advanced deep learning models in EEG-based emotion recognition offered promising solutions for enhancing psychological assessments and treatments. Throughout this project, attention was focused on various fields, which provided a comprehensive overview of the problem. By employing innovative approaches such as self-supervised learning and the latest neural network architectures, this project aims to overcome current limitations in both subject-dependent and subject-independent emotion classification using EEG data.

Through the development and adaptation of a flexible and relevant model, a practical tool that can be easily trained for individual patients was proposed. This can provide therapists with personalized and precise insights into emotional states. Furthermore, exploring state-of-the-art techniques with potential improvements in preprocessing and hyperparameter tuning is both promising and motivating.

Ultimately, this project aims to connect technological advancements with practical applications in psychology to enhance the effectiveness and objectivity of mental health care. However, it is neither asserted nor thought that such tools could replace humans in tasks related to psychology. Objective assistance is enabled, but the depth and complexity of the human mind still necessitate the empathy of a therapist.

REFERENCES

[1] Shu, Lin and Xie, Jinyan and Yang, Mingyue and Li, Ziyi and Li, Zhenqi and Liao, Dan and Xu, Xiangmin and Yang, Xinyi. "A review of emotion recognition using physiological signals", *Sensors*, vol. 18, no. 7, 2018.

[2] Li, Xiang and Zhang, Yazhou and Tiwari, Prayag and Song, Dawei and Hu, Bin and Yang, Meihong and Zhao, Zhigang and Kumar, Neeraj and Marttinen, Pekka. "EEG based emotion recognition: A tutorial and review", *ACM Computing Surveys*, vol. 55, no. 4, 2022.

[3] Zhang, Jianhua and Yin, Zhong and Chen, Peng and Nichele, Stefano. "Emotion recognition using multi-modal data and machine learning techniques: A tutorial and review", *Information Fusion*, vol. 59, no. 4, 2020.

[4] Verma, Gyanendra K and Tiwary, Uma Shanker. "Affect representation and recognition in 3D continuous valence–arousal–dominance space", *Multimedia Tools and Applications*, vol. 76, 2017.

[5] Kıymık, M Kemal and Güler, İnan and Dizibüyük, Alper and Akın, Mehmet. "Comparison of STFT and wavelet transform methods in determining epileptic seizure activity in EEG signals for real-time application", *Computers in biology and medicine*, vol. 35, no. 7, 2005.

[6] Ramoser, Herbert and Muller-Gerking, Johannes and Pfurtscheller, Gert. "Optimal spatial filtering of single trial EEG during imagined hand movement", *IEEE transactions on rehabilitation engineering*, vol. 8, no. 4, 2000.

[7] Wang, Zeyu and Zhou, Ziqun and Shen, Haibin and Xu, Qi and Huang, Kejie. "JDAT: Joint-dimension-aware transformer with strong flexibility for EEG emotion recognition", *Authorea*, 2023.

[8] Katsigiannis, Stamos and Ramzan, Naeem. "DREAMER: A database for emotion recognition through EEG and ECG signals from wireless low-cost off-the-shelf devices", *IEEE journal of biomedical and health informatics*, vol. 22, no. 1, 2017.

[9] Zheng, Wei-Long and Lu, Bao-Liang. "Investigating critical frequency bands and channels for EEG-based emotion recognition with deep neural networks", *IEEE Transactions on autonomous mental development*, vol. 7, no. 3, 2015.

[10] Duan, Ruo-Nan and Zhu, Jia-Yi and Lu, Bao-Liang. "Differential entropy feature for EEG-based emotion classification", *2013 6th international IEEE/EMBS conference on neural engineering (NER)*, 2013.

[11] Alhagry, Salma and Fahmy, Aly Aly and El-Khoribi, Reda A. "Emotion recognition based on EEG using LSTM recurrent neural network", *International Journal of Advanced Computer Science and Applications*, vol. 8, no. 10, 2017.

[12] Yang, Yilong and Wu, Qingfeng and Qiu, Ming and Wang, Yingdong and Chen, Xiaowei. "Emotion recognition from multi-channel EEG through parallel convolutional recurrent neural network", *2018 international joint conference on neural networks (IJCNN)*, 2018.

[13] Gong, Linlin and Li, Mingyang and Zhang, Tao and Chen, Wanzhong. "EEG emotion recognition using attention-based convolutional transformer neural network", *Biomedical Signal Processing and Control*, vol. 84, no. 1, 2023.

[14] Song, Tengfei and Zheng, Wenming and Liu, Suyuan and Zong, Yuan and Cui, Zhen and Li, Yang. "Graph-embedded convolutional neural network for image-based EEG emotion recognition", *IEEE Transactions on Emerging Topics in Computing*, vol. 10, no. 3, 2021.

[15] Li, Chang and Lin, Xuejuan and Liu, Yu and Song, Rencheng and Cheng, Juan and Chen, Xun. "EEG-based emotion recognition via efficient convolutional neural network and contrastive learning", *IEEE Sensors Journal*, vol. 22, no. 20, 2022.

[16] Mohsenvand, Mostafa Neo and Izadi, Mohammad Rasool and Maes, Pattie. "Contrastive representation learning for electroencephalogram classification", *Machine Learning for Health*, 2020.

[17] Shen, Xinke and Liu, Xianggen and Hu, Xin and Zhang, Dan and Song, Sen. "Contrastive learning of subject-invariant EEG representations for cross-subject emotion recognition", *IEEE Transactions on Affective Computing*, vol. 14, no. 3, 2022.

[18] Tian, Chenxi and Ma, Yuliang and Cammon, Jared and Fang, Feng and Zhang, Yingchun and Meng, Ming. "Dual-encoder VAE-GAN with spatiotemporal features for emotional EEG data augmentation", *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 31, 2023.

[19] He, Zhipeng and Zhong, Yongshi and Pan, Jiahui. "An adversarial discriminative temporal convolutional network for EEG-based cross-domain emotion recognition", *Computers in biology and medicine*, vol. 141, 2022.

[20] Barooah, Rituparna. "Physiology of Emotion", *Application of Biomedical Engineering in Neuroscience*, 2019.

[21] Morris, Jon D. "Observations: SAM: the Self-Assessment Manikin; an efficient cross-cultural measurement of emotional response", *Journal of advertising research*, vol. 35, no. 6, 1995.

[22] O'Reilly, Munakata, Hazy and Frank. "Navigating the Functional Anatomy of the Brain", *LibreTexts Medicine*.

[23] Ekman, Paul. "An argument for basic emotions", *Cognition & emotion*, vol. 6, no. 3-4, 1992.

[24] Teplan, Michal and others. "Fundamentals of EEG measurement", *Measurement science review*, vol. 2, no. 2, 2002.

[25] Chi-Feng Wang. "A Basic Introduction to Separable Convolutions", *Towards Data Science*, 2018.

[26] Sabour, Sara and Frosst, Nicholas and Hinton, Geoffrey E. "Dynamic routing between capsules", *Advances in neural information processing systems*, vol. 30, 2017.

[27] Hinton, Geoffrey E and Krizhevsky, Alex and Wang, Sida D. "Transforming auto-encoders", *Artificial Neural Networks and Machine Learning–ICANN 2011*, 2011.

[28] Yann LeCun, Corinna Cortes, Christopher J.C. Burges. "THE MNIST DATABASE of handwritten digits".

[29] Chen, Kun and Jing, Huchuan and Liu, Quan and Ai, Qingsong and Ma, Li. "A novel caps-EEGNet combined with channel selection for EEG-based emotion recognition", *Biomedical Signal Processing and Control*, vol. 86, 2023.

[30] Liu, Yu and Ding, Yufeng and Li, Chang and Cheng, Juan and Song, Rencheng and Wan, Feng and Chen, Xun. "Multi-channel EEG-based emotion recognition via a multi-level features guided capsule network", *Computers in Biology and Medicine*, vol. 123, 2020.

[31] Wei, Yi and Liu, Yu and Li, Chang and Cheng, Juan and Song, Rencheng and Chen, Xun. "TC-Net: A Transformer Capsule Network for EEG-based emotion recognition", *Computers in biology and medicine*, vol. 152, 2023.

[32] Vaswani, Ashish and Shazeer, Noam and Parmar, Niki and Uszkoreit, Jakob and Jones, Llion and Gomez, Aidan N and Kaiser, Łukasz and Polosukhin, Illia. "Attention is all you need", *Advances in neural information processing systems*, vol. 30, 2017.

[33] Liu, Ze and Lin, Yutong and Cao, Yue and Hu, Han and Wei, Yixuan and Zhang, Zheng and Lin, Stephen and Guo, Baining. "Swin transformer: Hierarchical vision transformer using shifted windows", *Proceedings of the IEEE/CVF international conference on computer vision*, 2021.

[34] Puthusserypady, Sadasivan. "Applied signal processing", *Now Publishers*, 2021.

[35] "Continuous wavelet transform", *Wikipedia*.

[36] Quinn, Andrew J. and Lopes-dos-Santos, Vitor and Dupret, David and Nobre, Anna C. and Woolrich, Mark W.. "EMD: Empirical Mode Decomposition and Hilbert-Huang Spectral Analyses in Python", *Journal of Open Source Software*, vol. 6, no. 59, 2021.

[37] Julia Cámara Aracil. "Emotion Classification using Self Supervised learning with a brain computer interface device", *DTU Health Tech*, 2023.

[38] Gramfort, Alexandre and Luessi, Martin and Larson, Eric and Engemann, Denis A. and Strohmeier, Daniel and Brodbeck, Christian and Goj, Roman and Jas, Mainak and Brooks, Teon and Parkkonen, Lauri and Hämäläinen, Matti S.. "MEG and EEG Data Analysis with MNE-Python", *Frontiers in Neuroscience*, vol. 7, no. 267, 2013.

[39] Lawhern, Vernon J and Solon, Amelia J and Waytowich, Nicholas R and Gordon, Stephen M and Hung, Chou P and Lance, Brent J. "EEGNet: a compact convolutional neural network for EEG-based brain–computer interfaces", *Journal of neural engineering*, vol. 15, no. 5, 2018.

[40] Kinga, D and Adam, Jimmy Ba and others. "A method for stochastic optimization", *International conference on learning representations (ICLR)*, vol. 5, 2015.

[41] Kostas, Demetres and Aroca-Ouellette, Stephane and Rudzicz, Frank. "BENDR: Using transformers and a contrastive self-supervised learning task to learn from massive amounts of EEG data", *Frontiers in Human Neuroscience*, vol. 15, 2021.

[42] Obeid, Iyad and Picone, Joseph. "The temple university hospital EEG data corpus", *Frontiers in neuroscience*, vol. 10, 2016.

[43] Devlin, Jacob and Chang, Ming-Wei and Lee, Kenton and Toutanova, Kristina. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding", *Proceedings of the 2019 Conference of the North American*, vol. 1, 2019.

[44] Baevski, Alexei and Zhou, Yuhao and Mohamed, Abdelrahman and Auli, Michael. "wav2vec 2.0: A framework for self-supervised learning of speech representations", *Advances in neural information processing systems*, vol. 33, 2020.

[45] Kan, Haoning and Yu, Jiale and Huang, Jiajin and Liu, Zihe and Wang, Heqian and Zhou, Haiyan. "Self-supervised group meiosis contrastive

learning for eeg-based emotion recognition", *Applied Intelligence*, no. 22, vol. 53, 2023.

[46] Liaw, Richard and Liang, Eric and Nishihara, Robert and Moritz, Philipp and Gonzalez, Joseph E and Stoica, Ion. "Tune: A Research Platform for Distributed Model Selection and Training", *ICML 2018 AutoML Workshop*, 2018.

APPENDIX

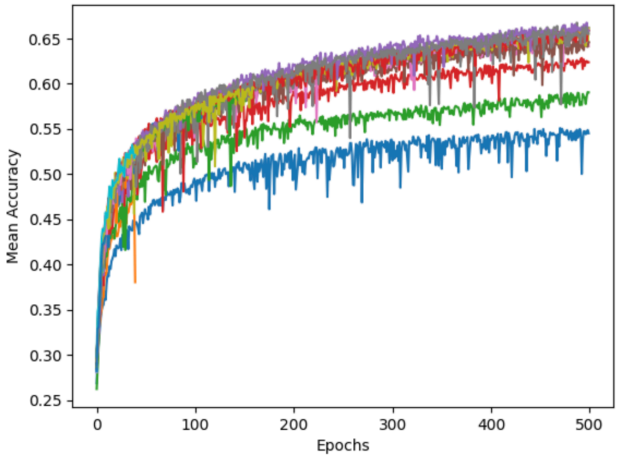


Fig. 29. Example of a hyperparamter optimization. In this scenario, only the number of inner channels for the DREAMER dataset was optimized. It was a grid search in a 1D space using the ASHA scheduler to allocate more computing resources to the best performing configurations.

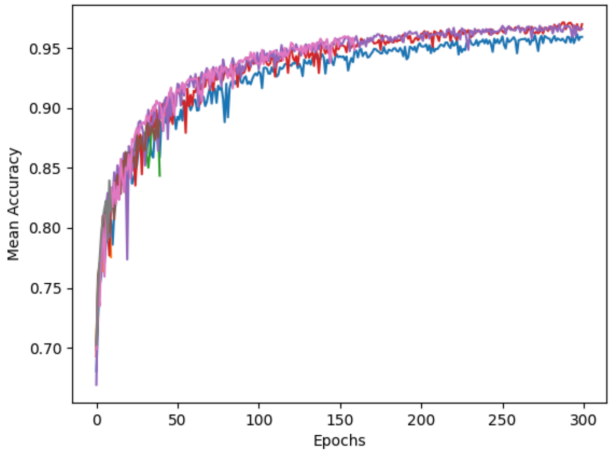
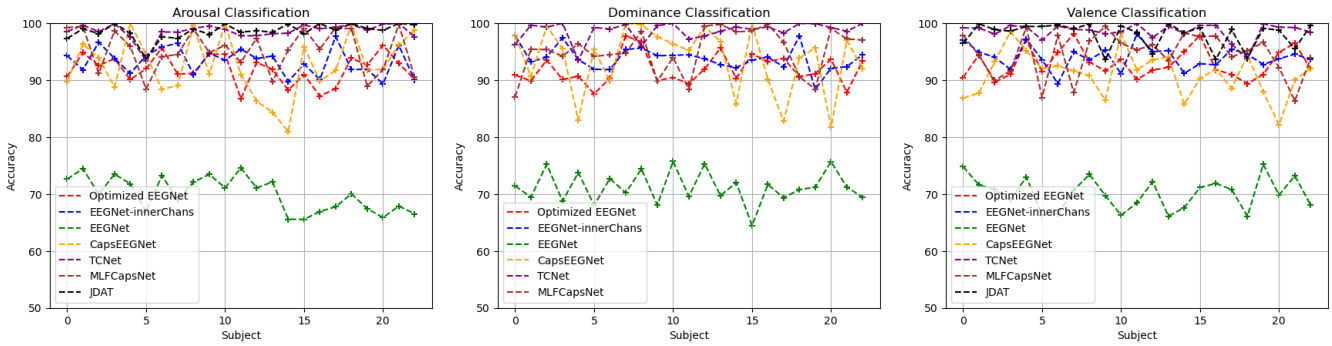


Fig. 30. Example of a hyperparamter optimization. It is a similar scenario as in Figure 29 except that this optimization concerns the SEED dataset. Moreover, the search space is 2D because both the number of inner channels and the length of the temporal kernel in the first convolutional layer of EEGNet were optimized at the same time.



(a) (b) (c)

Fig. 31. Performance comparison of each subject using different methods for arousal (a), dominance(b) and valence (c) on the DREAMER dataset.

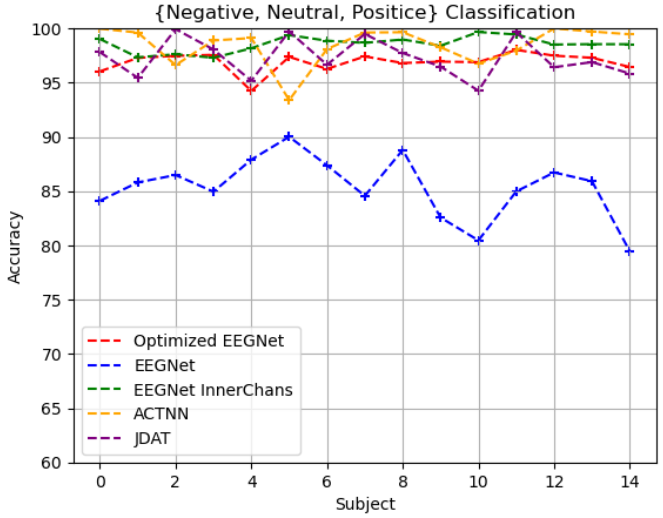


Fig. 32. Performance comparison of each subject using different methods on the SEEED dataset.