Tackling brain signal inter-subject variability with adaptive neural architectures

Sébastien Velut
CNE/TAU - LISN, INRIA
Fédération ENAC ISAE-SUPAERO ONERA
Toulouse, France
sebastien.velut@isae-supaero.fr

Stella Douka
TAU - LISN, INRIA
Université Paris-Saclay
Gif-sur-Yvette, France
styliani.douka@inria.fr

Theo Rudkiewicz
TAU - LISN, INRIA
Université Paris-Saclay
Gif-sur-Yvette, France
theo.rudkiewicz@inria.fr

Alex Davey
TAU - LISN
INRIA
Gif-sur-Yvette, France
alex.davey@inria.fr

Stephane Rivaud
TAU - LISN
INRIA
Gif-sur-Yvette, France
stephane.a.rivaud@inria.fr

Francois Landes
TAU - LISN, INRIA
Université Paris-Saclay
Gif-sur-Yvette, France
francois.landes@inria.fr

Julien Mille
TAU - LISN, INRIA / LIFAT
INSA Centre-Val de Loire
Blois, France
julien.mille@insa-cvl.fr

Guillaume Charpiat
TAU - LISN
INRIA
Gif-sur-Yvette, France
guillaume.charpiat@inria.fr

Sylvain Chevallier
TAU - LISN, INRIA
Université Paris-Saclay
Gif-sur-Yvette, France
sylvain.a.chevallier@inria.fr

Marie-Constance Corsi
NERV, Paris Brain Institute
Paris, France
marie-constance.corsi@inria.fr

Frederic Dehais
CNE, Fédération ENAC ISAE-SUPAERO
ONERA, Université de Toulouse
Toulouse, France
frederic.dehais@isae-supaero.fr

***Abstract*—Decoding electroencephalography (EEG) signals with deep learning remains challenging due to strong inter- and intra-individual variability, low signal-to-noise ratio, and the absence of an optimal architecture. While deep neural networks have demonstrated promising performance for EEG decoding, selecting an appropriate model architecture remains a costly and subject-specific process. Neural architecture search approaches partially address this issue but are computationally prohibitive in practice. In this work, we investigate a growing neural network strategy that adapts model expressivity during training, only expanding the architecture when required by the data. Building upon a lightweight convolutional neural network (CNN) architecture commonly used for a recent Brain-Computer Interface paradigm, code modulated visual evoked potential, decoding, we propose to grow only the classification head with two different growing modes, directed acyclic graph (DAG) and growing linear layers (GLL). This design explicitly targets subject-specific variability while preserving architectural priors learned from the literature. We evaluate our approaches on a five-class EEG dataset comprising 24 participants and compare them against fixed CNN, Riemannian TS-LDA, and GREEN architectures. Results show that the growing CNN significantly outperforms fixed CNN by increasing the accuracy by 1% while having 80% less neurons in the last layers. Our findings suggest that selectively growing the final layers of a deep network provides the capacity to adapt to subject-specific signatures, thus addressing the inter-subject variability. Moreover, we find an interesting trend where participants with better accuracy need fewer additional neurons to optimize their performance.**

***Index Terms*—Brain-computer interfaces, growing networks, convolutional neural networks, EEG, multivariate time series**

I. Introduction

Deep learning for time-series analysis remains a challenging problem, as no single model architecture has demonstrated consistent performance across the wide diversity of real-world signals. This difficulty is particularly pronounced for electroencephalography (EEG), where the recorded time series are highly non-stationary, noisy, and have strong inter-individual and intra-individual variability.

Inter-individual factors such as age, anatomy, neurophysiological signatures, and environment, as well as intra-individual states including fatigue, workload, and attentional engagement, continuously modulate EEG activity [1]. These sources of variability induce strong subject-specific signatures and significantly complicate the decoding of brain signals. In addition, the intrinsically low signal-to-noise ratio (SNR) of EEG—despite its portability—further limits the robustness and generalization capability of machine learning models for applications such as Brain-Computer Interface (BCI) that translates neural activity into commands. These problems induce a need to perform an often lengthy calibration, from 5 to 40 minutes, at every usage of the BCI. One promising approach, known as the modulated code visual evoked potential (c-VEP), involves toggling the stimulus pattern between black and white according to an aperiodic binary code—a random sequence of zeros and ones—that elicits a series of discrete visually evoked potentials [2]. This approach is promising as it needs less data to be trained and induces a more stable neural response than usual BCI paradigms, meaning we can reduce the size and

the complexity of the classifier architecture [3], [4]. However, the intra- and inter- variability is still present and induces the need for repeated calibration and the search for an architecture adapted to the participant's EEG data.

Numerous deep learning architectures have been proposed for EEG decoding, ranging from compact convolutional neural networks to more complex hybrid and Riemannian-based models [5]. Although the NeurIPS competition suggests that deep architectures can achieve strong performance, no single model has emerged as a reliable reference across tasks and datasets [6]. As a result, identifying an appropriate architecture remains a non-trivial and task-dependent process.

Finding the correct architecture for a brain decoding task is a complex task, depending on the expected outcomes (accuracy, robustness, embedding capability, etc.). Neural architecture search (NAS) provides strong methods for identifying and selecting the best architectures for a given task, at the expense of high computational cost [7]. Certain one-shot techniques have been proposed to reduce computational costs, for example, training super-networks and searching for the most fitting architecture within them [8], [9], thus following the lottery ticket hypotheses [10]. The cost, however, as well as the memory and data consumption, remains high. This is problematic for EEG signals, as inter-subject variability requires repeating the lengthy and costly search for each subject. Moreover, in practical BCI settings, subject-specific calibration must be performed quickly, with limited data and computational resources. Methods that require extensive architecture search or full network retraining are therefore impractical. This drives computational costs too high for real-world applications, especially for patients who cannot endure hours of recordings.

Growing Neural Networks, an alternative to NAS, expand the architecture over the course of training, searching for the best architecture for a specific task without discrete search phases or multiple model retrainings [11]–[13]. One of the objectives of these methods is to acquire the best architecture as fast as possible, which is also our goal. Based on this intent, we will use the method of TINY [12]. The main criterion for increasing or adding a layer is to detect that the expressivity of the network is insufficient to fulfill the task. This approach yields a trained network of the correct size at the end of the training. It is possible to grow the complete architecture or just a part of it. In the latter, this allows reusing existing architecture choices.

In this work, we propose to build upon an existing architecture, a convolutional neural network (CNN) found in the literature, expanding only the last layer (called the expressivity layer). We evaluate this approach by applying either growing linear layers or directed acyclic graph layers at the end of a CNN and compare it against the fixed CNN, which is, beforehand, compared against a Riemannian non-deep model, Tangent Space Linear Discriminant Analysis (TS-LDA) [14], and a complex deep model, Gabor Riemann EEGNet (GREEN) [15]. Our results demonstrate that, by adapting the expressivity layer to the subject-specific brain signal, our approach outperforms existing fixed-architecture methods while reducing the number of parameters compared to fully fixed or more flexible growing designs. These findings highlight growing architectures as a computationally efficient and principled solution to subject-specific EEG variability.

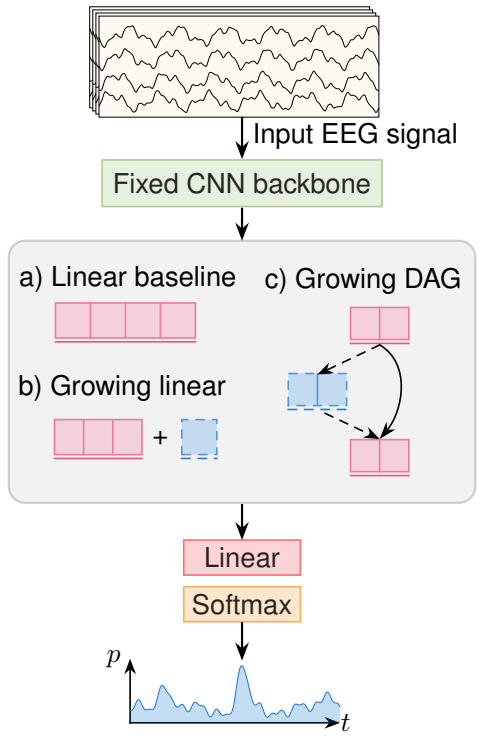

Fig. 1. Our method uses a fixed CNN backbone, based on the EEG2Code model [16], but grows the classification head during training to find a subject-specific architecture.

We will make our data and code available upon acceptance of this paper.

II. Methods

A. Pre-processing steps

The initial step in classifying EEG data involves applying a pre-processing pipeline [17]. Below, we outline the sequential procedures employed in this study.

In order to restrict ourselves to the frequency bands involved during c-VEP neural processes, we apply a finite impulse response filter between 1 Hz and 45 Hz. As the time between two onsets of the flashes in a code exceeds 0.35 s, we create an overlapping epoch of size 0.35 s every $1/500$s. This gives $N_e = 15 * (2.2 - 0.35) * 500 = 13875$ chunks of continuous signals, referred to as epochs, per participant per class. Then, to ensure comparable amplitude across participants, we normalize the data by dividing it by the standard deviation of each participant's training set. To improve the SNR of the captured event-related potential (ERP) response, which corresponds to a response to a specific sensory, cognitive, or motor event, to the flashes of the c-VEP, and improve the quality of the signal, we add a spatial filtering step using the spatial filter xDAWN, which enhances performance for ERP-based BCI [18]. For xDAWN, we select the 4 highest-ranked components.

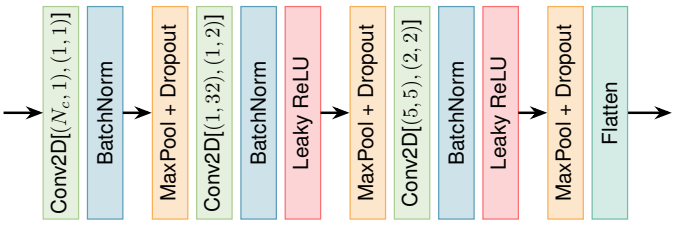

Fig. 2. Fixed CNN backbone based on the EEG2Code model [16].

To allow preservation of the information contained in the average target response, we create super epochs by combining the average of the spatially filtered epoch with each spatially filtered target response. Let $\mathbf{E_Z} = \frac{1}{N_e}\sum_{k=1}^{N_e}\mathbf{Z}_k$ be the evoked response of the spatially filtered data where N_e is the number of epochs and $\mathbf{Z}_k$ is the k^{th} epoch of the spatially filtered target data. $\mathbf{Z}_k \in \mathbb{R}^{N_c \times N_t}$, where N_c is the number of electrodes channels, and N_t is the number of time samples in one epoch. We create the super epochs where the k^{th} sample will be $\mathbf{Z}'_k = \{Z_{k,1}, Z_{k,2}, Z_{k,3}, Z_{k,4}, E_{\hat{Z},1}, E_{\hat{Z},2}, E_{\hat{Z},3}, E_{\hat{Z},4}\}$. Combining this preserved information with the application of the spatial filter, intra-epoch variability is mitigated. We compute the evoked response of the spatially filtered data using the calibration data, and we concatenate it to all epochs, including both the training and test sets.

To align the different participants onto the same feature space and reduce the variability between the subjects, we apply a Riemannian spatial alignment. For this purpose, we first use the calibration phase of the participant as a training batch and compute the associated covariance matrices with the Ledoit-Wolf estimator [19]. Then, we measure the Riemannian mean [20] of these matrices, and finally, we transport the samples of the participant with this mean as follows:

$$\hat{\mathbf{Z}}'_i = \bar{\mathbf{R}}^{-\frac{1}{2}}\mathbf{Z}'_i \quad \forall i \in [1, ..., N_e] \tag{1}$$

where $\mathbf{Z}'_i \in \mathbb{R}^{N_c \times N_t}$ is an element of $\{\mathbf{Z}'_i\}_{i\in[1,....,N_e]}$, $\bar{\mathbf{R}}$ is the Riemannian mean of of the covariances matrices of the set $\{\mathbf{Z}'_1, ..., \mathbf{Z}'_T\}$ and T is the number of samples in the calibration set of the participant.

B. Description of the models

1) CNN: We use a variant [21] of the original EEG2Code model [16] that was optimized for burst-c-VEP decoding [22]. The model is structured with three convolutional blocks. The CNN backbone is displayed on figure 2, with Conv2D$[k, d]$ indicating a convolutional layer with kernel k and dilation d. In this fixed version of CNN, the architecture is connected to a 256-unit dense layer with a leaky ReLU followed by a 2-unit dense output layer with a softmax activation function. The model has 28353 parameters and is optimized with Adam, using a learning rate of 0.001.

2) TS-LDA: This model is a Riemannian machine learning algorithm that is faster in computation time, has a smaller number of parameters than the CNN (1050 parameters only), and is used in BCI and attained 98% accuracy in [3]. It can be summarized as a spatial filter combined with a classifier that uses second-order statistics, that is, the covariance information. The first layer spatially filters the EEG data with the xDAWN filter. Covariances, estimated with Ledoit-Wolf [19] for each of the xDAWN outputs components, are concatenated with the target signal covariance to create a super-covariance matrix. To be able to predict the label, the super-covariance matrices are projected onto the tangent space, where a linear discriminant analysis (LDA) classifier is applied on the tangent vectors.

3) GREEN: GREEN is an algorithm introduced in [15] to detect novel neuro-markers applicable in a medical context. To get temporal and frequency information from the EEG data and to increase the quantity of usable information, GREEN starts by performing a Gabor wavelet transform, followed by a 1D convolutional layer. This classifier coupled with the pre-processing steps succeed to counter the EEG variability [23].

In this paper, we choose to separate the signal into 22 frequency bands that are learned through training, all included between 0 and 4.4 octaves. Then the wavelet data are transformed in covariances matrices and go through an SPDNet [24]. A fully connected layer with 2 hidden layers of size 20 and 10 units is processing the tangent space projection of the latent space to make classification decisions. GREEN counts a total of 5922 parameters.

4) Growing CNN: To allow the architecture to adapt to the complexity of each participant, we adapt the previous architecture by replacing the last 2 linear layers with growing linear layers (GLL). The motivation to increase the size of the layers is to augment the network's expressivity and tackle each specific subject more optimally. The objective is to acquire an architecture just large enough to handle the complexity of the task at hand. To accomplish that, we use the notion of expressivity bottleneck described in [12], [25], [26]. Given a predefined architecture $\mathcal{A}$ with parameterization θ, denoting the output by f_θ, we can define the tangent space $\mathcal{T}_\mathcal{A}$ of all possible functions we can approximate using gradient descent, $\mathcal{T}_\mathcal{A} := \left\{\frac{\partial f_\theta}{\partial \theta}\delta\theta \mid \delta\theta \in \Theta\right\}$. By taking the curvature of the space into account, we can compute a direction derived from the natural gradient [12] that indicates the optimal update $v_\nabla(x)$ to decrease loss $\mathcal{L}$, when we are not constrained by the tangent space, $v_\nabla(x) := \nabla_{f_\theta(x)}\mathcal{L}(f_\theta(x), y)$. However, the best move we can make with a fixed architecture is the projection of this direction onto the tangent space, $v^* := \text{Proj}_{\mathcal{T}_\mathcal{A}}(v_\nabla)$. This geometric explanation gives us an orthogonal triangle whose hypotenuse is the desired direction, and the vertical defines the inability of the architecture to completely express the direction of the loss $\Psi := \underset{v^*\in\mathcal{T}_\mathcal{A}}{\arg\min}\ \underset{(x,y)\sim D}{\mathbb{E}}||v_\nabla(x) - v^*(x)||^2$. This expressivity bottleneck is what we try to minimize by increasing the size of the classification head.

The most straightforward solution is to maintain the sequential nature of the last two layers and gradually increase the size of their hidden state. In that case, every growth step will increase the output dimension of the first layer and the input dimension of the second layer, with the starting number of hidden neurons being set at 10. The completion of the CNN with these growing linear layers will be named GLL-CNN in the rest of the paper.

Another design choice is to start with a single linear layer and allow the classification head to grow freely, meaning that it can add new layers parallel to the starting one. This creates a directed acyclic graph (DAG) where every edge is a linear layer, and every node is a hidden state. This free-form design adds one more level of complexity, meaning that we have to choose exactly how and where the DAG will grow, such as by increasing the size of a hidden dimension, as we do in the sequential case, adding a new edge, or adding a new node. The completion of the CNN with these DAG layers will be named DAG-CNN in the rest of the paper.

C. Training methods

For all architectures and each participant, we take the first 4/15 of the data for the training, 1/15 for the validation, and 10/15 for the testing set. For the non-growing architecture, we adopt a classical training-validation-testing procedure where we train for 300 epochs on the training set. The validation set is used to fine-tune the hyperparameters of the pre-processing steps. Then we calculate the best model's performance on the testing set.

For the growing architecture, we have different steps. These steps are done for each individual to create a participant-specific model. The growing schedule is set to alternate for 300 steps long between 4 training steps (normal training procedure) and 1 growth step. A growth step is decomposed into several actions. First, one needs to choose the layer to grow: we pick the one with the most expressivity bottleneck. For GLL-CNN, we constrain ourselves to the head's first layer; for DAG-CNN, it is constrained to be a part of the DAG (network head). Second, we gather the necessary gradient-related statistics that allow to design the possible new neurons. In practice those combine into a matrix $\mathbf{SN}$. Third, we compute the optimal updates to perform. However, we are checking if the eigenvalues of $\mathbf{SN}^{-\frac{1}{2}}$ are greater than 0.01, a threshold that is fine-tuned on the validation set. If not, no neurons are added. If so, all neurons associated with an eigenvalue above 0.01 are added, up to a maximum of 10 neurons. New neurons are initialized with weights chosen to minimize the loss (on a subset of the training set). Finally, we apply the change to the model and resetting the optimizer to take the changes into account.

We first compare 3 non-growing neural networks, TS-LDA / GREEN / CNN (Section III-B). We then compare the best non-growing neural network (CNN) to its growing version, GLL-CNN and DAG-CNN (Section III-C). Finally, we analyze how the number of neurons that grew in the last layers correlates with participant performance.

III. Results

A. Experimental setup

1) Dataset: All the results presented in this study are based on an already collected dataset called Ricker patch-based StAR c-VEP (ISAE-SUPAERO Dataset [27]). The EEG data were collected using a dry 8-electrode Enobio system, with a sampling rate of 500 Hz to capture the surface brain activity.

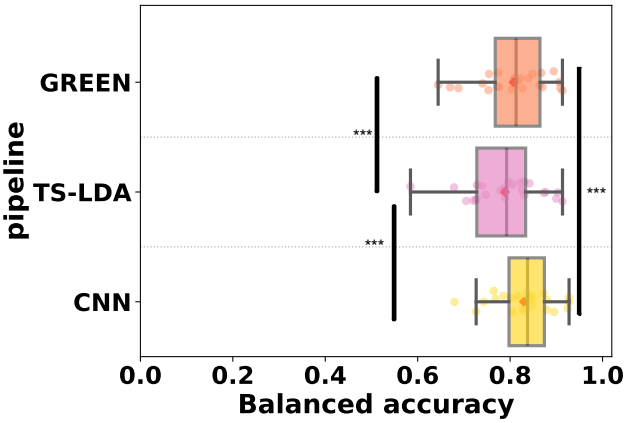

Fig. 3. Balanced accuracy for the 3 fixed architecture GREEN, TS-LDA, CNN. The boxes correspond to the distribution between the first and third quartiles. The mean is shown as a red square. Each point corresponds to the performance of one participant. The stars indicate a significant difference (***: $p < 0.001$)

The $N_c = 8$ electrodes were positioned over the occipital and parieto-occipital sites, namely: PO7, O1, Oz, O2, PO8, PO3, POz, PO4. This dataset comprises 24 healthy individuals (4 women, 20 men, mean age = 29.3 years, SD = 7.5 years) performing a visual task where they were instructed to direct their attention sequentially to five presented targets for 2.2 seconds each, which were cued in a random order for 0.5 seconds each. Each participant performed 15 runs, each of which was composed of 5 trials of 2.2 seconds each. In other words, we have a total of 75 trials, including 15 trials of 2.2 seconds per codes.

2) Statistical analysis and Evaluation metrics: To analyze the results of III-B and III-C, we use the Wilcoxon signed rank test. The Wilcoxon signed-rank test is chosen due to the non-Gaussian distribution of subject-wise accuracies. The effect size between two decoding models is measured via standardized mean difference (SMD) estimated over the participants.

The performance of the model is evaluated with the classification balanced accuracy and the stability of the results.

B. Best classifier without growing

In Fig. 3 we compare three fixed architectures. The CNN performs significantly better than the 2 other decoding models ($p < 0.001$). The performance gain of the CNN over GREEN and TS-LDA corresponds to a large effect size ($SMD > 0.8$), indicating a practically meaningful improvement. CNN obtains an average accuracy of 0.83, when GREEN obtains 0.807, and TS-LDA has the worst performance with an accuracy of 0.789. Moreover, CNN provides more stable results compared to the other two architectures.

C. Growing vs non-growing

We can see in Fig. 4 the results of the CNN and two growing versions of the CNN. The GLL-CNN performs significantly better than the two other architectures ($p < 0.001$ for DAG-CNN and $p < 0.01$ for the CNN). The CNN performs statistically better than the DAG-CNN ($p < 0.001$). The improvement

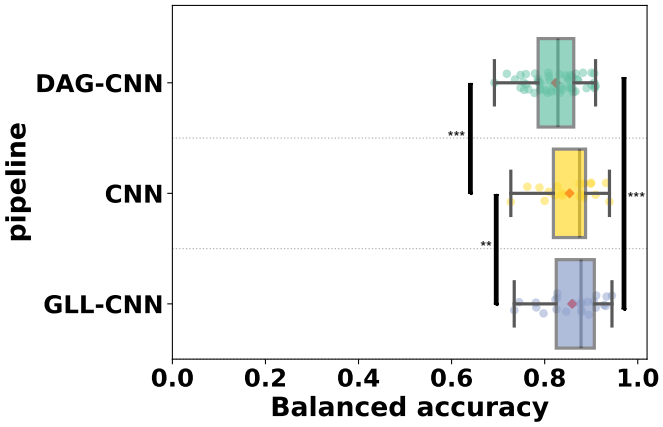

Fig. 4. Balanced accuracy for the best fixed architecture CNN and two growing architectures DAG-CNN and growing linear layer (GLL-CNN). The boxes correspond to the distribution between the first and third quartiles. The mean is shown as a red square. Each point corresponds to the performance of one participant. The stars indicate a significant difference (***: $p < 0.001$, **: $p < 0.01$).

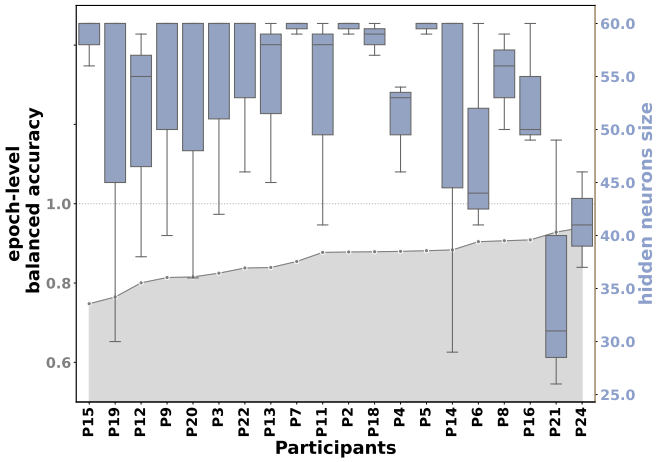

Fig. 5. Boxplot of the size of the hidden neurons in the growing layers of the GLL-CNN. Balanced accuracy on the test set for each participant of the CNN architecture. The participants are sorted in an ascending order of their balanced accuracy. Accuracy starts at 0.5 as it corresponds to the chance level.

of GLL-CNN over the fixed CNN exhibits a medium effect size ($SMD = 0.82$), confirming that the observed gains are not only statistically significant but also substantial in magnitude. The GLL-CNN obtains an average accuracy of 0.860, whereas DAG-CNN obtains only an average accuracy of 0.824, and CNN obtains 0.853 as average accuracy.

D. Neurons size with performance

Figure 5 presents the number of hidden neurons in the growing layers of the GLL-CNN architecture for each participant. We observe a trend: the number of neurons added decreases as participant performance increases, reaching for all participants less than in the fixed-sized CNN (up to 85% less). For most of the participants with the worst accuracy, we observe that the number of hidden neurons is saturated, which is due to an insufficient number of epochs.

IV. Discussion

A. Growing vs Not-growing

In this study, we investigate whether a growing network outperforms static architectures. To establish a baseline, we first select the optimal non-growing architecture from three established models commonly employed in the BCI literature. A non-deep Riemannian architecture (TS-LDA), a complex deep architecture using wavelet transform (GREEN), and a light CNN. The CNN is the architecture with the best and most stable performance. This leads us to choose the CNN to adapt it with growing layers. We construct the DAG-CNN and GLL-CNN architectures and compare them. The growing architecture GLL-CNN perform better than the CNN, showing the effectiveness of this strategy for adapting to the subject-specific brain signal. However, DAG-CNN perform worse than the CNN and GLL-NN. Moreover, the GLL-CNN has, on average, 200 fewer neurons than the DAG-CNN and the CNN. The DAG-CNN goes from 80 to 600 hidden neurons depending on the participants. This difference compared to the GLL-CNN comes from the presence of an early stopping and a threshold to stop the growing in the GLL-CNN which is not present in the DAG-CNN. This will be added in future researches. Therefore, we have shown that adding growing layers as the expressivity layer allows us to efficiently adapt the classifier to each participant's brain signal.

In a BCI framework, we do not have enough data to train really huge and complex models. This data scarcity explains why a model with 200 fewer neurons (around 80% less neurons) can have a better accuracy than a fixed model with more neurons. A smaller size of parameters seems more adapted to the quantity of data we have.

B. Number of hidden neurons with performance

In Section III-D, we observed an inverse relationship between the final number of hidden neurons in the CNN and its classification performance. This trend may stem from differences in data quality across participants: high-performing individuals tend to exhibit more stable, less variable, and less noisy EEG signals. Consequently, the classifier requires fewer hidden neurons to extract relevant features from its data, resulting in reduced network growth. Conversely, participants with lower performance may exhibit greater intra-subject variability or higher noise levels in their EEG data. In such cases, the classifier must expand its capacity—by increasing the number of hidden neurons—to effectively identify and leverage discriminative features.

However, some of the worst participants do not reach the saturation level. We could explain it as the fact that the data of these participants are in a state where adding neurons does not seem to help to improve the performance. In contrast, if we look at the good performers, some participants reach a smaller final number of hidden neurons despite having a smaller accuracy. Some participants could have more complex

data and still have small variability and noise in the data. This complexity in the data could induce variable growth in the growing layers.

The question of the relationship between the performance and the final number of hidden neurons remains open and would imply to increase the number of growing steps, increase the number of epochs, and decrease the threshold for the eigenvalues of the extension.

V. Conclusion

In this paper, we investigate the use of growing neural network architectures for subject-specific EEG decoding. We first compare several fixed architectures commonly used in the BCI literature and identify a lightweight CNN as the most accurate and stable baseline. Building upon this model, we introduce two growing variants that dynamically adapt their classification capacity during training.

Our results demonstrate that selectively growing the final classification layers significantly improves decoding performance compared to fixed architectures, while maintaining a reduced parameter count. Among the proposed designs, the growing linear layer (GLL-CNN) achieves the best trade-off between accuracy, stability, and model complexity. These findings suggest that subject-specific variability in EEG signals can be efficiently addressed by adapting only the expressivity of the final layers, without modifying the shared feature extraction backbone. Furthermore, the observed relationship between the final number of hidden neurons and the baseline decoding performance provides insight into how the growing mechanisms reflect data complexity and signal variability between participants. This supports the interpretation of growing layers as an implicit measure of task difficulty and subject-specific signal structure.

Future work will focus on refining growth criteria, extending training schedules, and exploring the growth thresholds to better characterize the relationship between expressivity and performance. Additionally, applying this framework to other datasets, EEG tasks and modalities and online BCI settings could further establish growing architectures as a practical alternative to computationally expensive architecture search methods. Furthermore, while evaluated on EEG data, the proposed growing strategy is applicable to any subject-dependent or non-stationary time-series classification problem where computational efficiency and adaptive capacity are critical.

Acknowledgment

The European Union also funded this work under GA no.101135782 (MANOLO project).

References

[1] S. Saha and M. Baumert, "Intra- and inter-subject variability in EEG-based sensorimotor brain computer interface: A review," *Frontiers in Human Neuroscience*, 2020.

[2] V. Martinez-Cagigal, J. Thielen, E. Santamaria-Vazquez, S. Perez-Velasco, P. Desain, and R. Hornero, "Brain-computer interfaces based on code-modulated visual evoked potentials (c-VEP): a literature review," *Journal of Neural Engineering*, 2021.

[3] F. Dehais, K. Cabrera Castillos, S. Ladouce, and P. Clisson, "Leveraging textured flickers: A leap toward practical, visually comfortable, and high-performance dry EEG code-VEP BCI," *Journal of Neural Engineering*, vol. 21, no. 6, p. 066023, Dec. 2024.

[4] J. Thielen, "Addressing BCI inefficiency in c-VEP-based BCIs: A comprehensive study of neurophysiological predictors, binary stimulus sequences, and user comfort," *Biomedical Physics & Engineering Express*, vol. 11, no. 4, p. 045017, 2025.

[5] S. Chevallier, I. Carrara, B. Aristimunha, P. Guetschel, S. Sedlar, B. Lopes, S. Velut, S. Khazem, and T. Moreau, "The largest EEG-based BCI reproducibility study for open science: the MOABB benchmark," *arXiv*, 2024.

[6] B. Aristimunha, D. Truong, P. Guetschel, S. Y. Shirazi, I. Guyon, A. R. Franco, M. P. Milham, A. Dotan, S. Makeig, A. Gramfort, J.-R. King, M.-C. Corsi, P. A. Valdés-Sosa, A. Majumdar, A. Evans, T. J. Sejnowski, O. Shriki, S. Chevallier, and A. Delorme, "EEG Foundation Challenge: From Cross-Task to Cross-Subject EEG Decoding," 2025.

[7] C. White, M. Safari, R. Sukthanker, B. Ru, T. Elsken, A. Zela, D. Dey, and F. Hutter, "Neural Architecture Search: Insights from 1000 Papers," 2023.

[8] H. Liu, K. Simonyan, and Y. Yang, "DARTS: Differentiable architecture search," in *ICLR*, 2019.

[9] H. Pham, M. Guan, B. Zoph, Q. Le, and J. Dean, "Efficient neural architecture search via parameter sharing," in *ICML*, 2018.

[10] J. Frankle and M. Carbin, "The lottery ticket hypothesis: Finding sparse, trainable neural networks," in *ICML*, 2019.

[11] U. Evci, B. van Merrienboer, T. Unterthiner, F. Pedregosa, and M. Vladymyrov, "GradMax: Growing neural networks using gradient information," in *ICLR*, 2022.

[12] M. Verbockhaven, T. Rudkiewicz, S. Chevallier, and G. Charpiat, "Growing tiny networks: Spotting expressivity bottlenecks and fixing them optimally," *Transactions on Machine Learning Research*, 2024.

[13] K. Maile, E. Rachelson, H. Luga, and D. G. Wilson, "When, where, and how to add new neurons to ANNs," in *AutoML*, 2022.

[14] A. Barachant, S. Bonnet, M. Congedo, and C. Jutten, "Multiclass Brain–Computer Interface Classification by Riemannian Geometry," *IEEE Transactions on Biomedical Engineering*, 2012.

[15] J. Paillard, J. F. Hipp, and D. A. Engemann, "GREEN: A lightweight architecture using learnable wavelets and Riemannian geometry for biomarker exploration with EEG signals," *Patterns*, 2024.

[16] S. Nagel and M. Spuler, "Modelling the brain response to arbitrary visual stimulation patterns for a flexible high-speed brain-computer interface," *PLOS ONE*, 2018.

[17] R. Kessler, A. Enge, and M. A. Skeide, "How EEG preprocessing shapes decoding performance," *Communications Biology*, 2025.

[18] B. Rivet, A. Souloumiac, V. Attina, and G. Gibert, "xDAWN algorithm to enhance evoked potentials: Application to brain-computer interface." *IEEE Transactions on Biomedical Engineering*, 2009.

[19] O. Ledoit and M. Wolf, "A well-conditioned estimator for large-dimensional covariance matrices," *Journal of Multivariate Analysis*, 2004.

[20] R. Bhatia, *Matrix Information Geometry*. Springer Nature, 2013, ch. The Riemannian Mean of Positive Matrices, pp. 35–51.

[21] S. Nagel and M. Spuler, "World's fastest brain-computer interface: Combining EEG2code with deep learning," *PLOS ONE*, 2019.

[22] K. Cabrera Castillos, S. Ladouce, L. Darmet, and F. Dehais, "Burst c-VEP based BCI: Optimizing stimulus design for enhanced classification with minimal calibration data and improved user experience," *NeuroImage*, p. 11, 2023.

[23] S. Velut, J. Thielen, S. Chevallier, M.-C. Corsi, and F. Dehais, "Neurophysiological screening of individual variability for robust decoding in c-vep-based bci," *Imaging Neuroscience*, vol. 4, p. IMAG.a.1172, 03 2026.

[24] Z. Huang and L. Van Gool, "A riemannian network for SPD matrix learning," in *AAAI*, 2017.

[25] M. Verbockhaven, "Spotting expressivity bottlenecks in neural networks and fixing them by optimal architecture growth," Theses, Université Paris-Saclay, Mar. 2025. [Online]. Available: https://theses.hal.science/tel-05232707

[26] S. Douka, M. Verbockhaven, T. Rudkiewicz, S. Rivaud, F. P. Landes, S. Chevallier, and G. Charpiat, "Growth strategies for arbitrary DAG neural architectures," in *ESANN*, 2025.

[27] K. Cabrera Castillos and F. Dehais, "5-Class Burst C-VEP with Dry EEG," 2024.