Proceedings of the
9th Graz Brain-Computer Interface Conference 2024
10.3217/978-3-99161-014-4-040

ANA-E: A NOVEL APPROACH FOR PRE-TRAINED ERROR DETECTION MODELS IN BRAIN-COMPUTER INTERFACES

Alexandros Christopoulos[1], Matias Valdenegro-Toro[1], Andreea Ioanna Sburlea[1]

[1] Department of Artificial Intelligence,Faculty of Science and Engineering, University of Groningen, Groningen, The Netherlands.

E-mail: a.i.sburlea@rug.nl

ABSTRACT: Error-related potentials hold the potential to enhance self-correcting behaviors in Brain-Computer Interfaces (BCIs), pivotal for human-machine interactions. However, integrating error detection mechanisms poses challenges, notably in lengthy calibration sessions required for different BCI modules. To address this, we propose a novel approach using Self-Supervised Learning (SSL) with an autoencoder architecture, called Ana-E, to develop pre-trained error detection pipelines. We recorded EEG data from participants navigating a game scenario imposed with errors. Offline analyses within and between participants were conducted for both preprocessed EEG trials and Ana-E features with two classifiers. Within-participants analysis showed comparable performance between Ana-E features and EEG trials. While in between-participants analysis, Ana-E exhibited an 8% performance improvement (72%) over the second-best pipeline (64%). Our study offers valuable insights into the future of pre-trained models for error detection in BCIs, providing a baseline for more complex architectures with the goal of significantly enhancing BCI usability and reducing dependency on calibration sessions, thereby improving user experience and applicability.

INTRODUCTION

In the field of Brain-Computer Interfaces (BCIs), Error-related Potentials (ErrPs) have been utilized as error detection instruments to expand the usability of architectures and develop a smoother experience between the user and external device [1]. Such implementations have been applied both in offline [2] and online paradigms [3]. Although the integration of errors as corrective instruments could improve the usability of BCIs as assistive tools, an immediate challenge emerges. Different BCI modules would require multiple calibration sessions [4], making the existence of pretrained models a requirement for self-correcting BCI implementation to prove applicable. In this paper, we attempt to develop pretrained models by adapting an autoencoder architecture to conduct a Self-Supervised Learning (SSL) task.

SSL reflects a subset of unsupervised learning methods in which neural networks are trained with automatically generated labels (pretext task) and then tested on a supervised task (downstream task), where human annotations are utilized to evaluate the performance of the model [5]. SSL relies on the premise that input information has distinguishable characteristics, and learned feature representations from the pretext task can be transferred to the downstream task [5]. SSL has been used successfully in visual feature learning tasks like image colorization [6], temporal order verification [7], and visual-audio correspondence verification [8]. SSL methods have further been deployed for time series data [9], where a common method is that of masked autoencoders, which randomly mask patches of the original time series data and learn temporal dynamics by recovering the masked patches [10]. Recently, SSL methods were implemented in EEG data for sleep stage recognition and pathology detection, outperforming purely supervised deep neural networks in low-labeled cases [11].

Autoencoders represent an unsupervised learning technique where the core idea is to conduct a representation learning task [12]. To do so, a deterministic encoder-decoder network pair is trained to learn a feature vector, often referred to as a 'bottleneck,' capable of encoding the underlying structural characteristics of the input samples. The learned feature vector could then be used by the decoder to fully reconstruct the input data samples [13].

In this paper, we used a 1D convolutional autoencoder architecture for EEG reconstruction that we coined as Ana-E. After training our model on EEG reconstructions, we extracted the encoder part and used it as a feature extractor of EEG trials, which we then fed to a classification head (CH). Our goal was to develop an architecture capable of deconstructing and reconstructing the input EEG as our pretext task. We then expect that the learned features from our encoder would be robust enough to classify errors in human participants in a downstream task. By doing so, we hope to address the issue of pre-trained BCI models. To investigate the novelty of our approach, we tested our architecture both within-participants and between-participants.

MATERIALS AND METHODS

Participants: We recorded 10 participants (6 females) with a mean age of 22 years (SD = 2.3), each undergoing a single recording session. One participant was removed due to incomplete markers. Participants were recruited

Published by
Verlag der Technischen Universität Graz

10.3217/978-3-99161-014-4-040

via advertisement fliers and compensated at a rate of 8 euros per hour. They provided informed consent and were informed of their right to withdraw at any time during the experiment. Ethical approval was obtained from the Ethical Committee of the Faculty of Arts, University of Groningen, The Netherlands (ID 92123476).

Procedure: Upon arrival, participants were introduced to the laboratory, briefed on the study, and signed informed consent forms. EEG cap placement took approximately 30 minutes. Following this, participants underwent a brief training session (5-10 minutes) to familiarize themselves with the experimental paradigm, including game rules and controls.After training, participants explained the game rules to researchers and began playing, with the game duration lasting 60-90 minutes. Rest periods were provided between trials as needed. The total experiment duration ranged from 120 to 150 minutes.

EEG Recordings: Participants EEG was recorded with antiCAP slim/snap 32 gel based active electrodes according to the 10-20 international system with a sampling rate of 500Hz using the LiveAmp BrainProducts amplifier. The measured EEG channels were: FP1, FPz, FP2, AF3, AFz, AF4, F3, F1, Fz, F2, F4, FC3, FC1, FCz, FC2, FC4, C3, C1, Cz, C2, C4, CP5, CP1, CPz, CP2, CP6, P3, Pz, P4, O1, Oz, O2. Ground and reference electrodes were placed on the left and right mastoid, respectively. EEG was recorded with the Brainvision recorder. Finally, impedance of electrodes was kept below 20 Ω for all participants.

Experimental Procedure: The experiment was developed in the Unity game engine [14] and had the code name Honey Heist. The experiment consisted of two phases: the training phase (approximately 5 minutes) and the testing phase (approximately 60–90 minutes). In this game, participants had to control a 3D avatar (bear) using keyboard buttons (W, S, D, A, or arrow keys) to reach a target (acquiring the honey) and then escape from the predefined boundaries to reach the finish line (forest). After participants passed the starting line (fence), they were chased by an artificial agent (chicken) throughout the rest of the trial.

Each trial had two possible outcomes: either participants acquired the target (honey) and reached the finish line, resulting in "winning" the trial, or the agent caught them before reaching the finish line, resulting in "losing" the trial. Participants were instructed to complete the task as quickly as possible. This experiment consisted of 400 trials, divided into three experimental conditions: 1) Normal Trials: 280 trials, 2) Control Error: 60 trials, and 3) Environment Error: 60 trials. Each trial ranged from 9 to 13.5 seconds depending on the participants' performance. In the normal trials, the procedure was identical to what is described above. In the control error condition, after crossing a specified threshold, the player lost the ability to jump over fences (Fig. 1), resulting in the agent catching up with the player and subsequently "losing" the trial. The threshold was an invisible box that was randomly selected between 3 – 3.6 units on the Z axis, in the game environment.

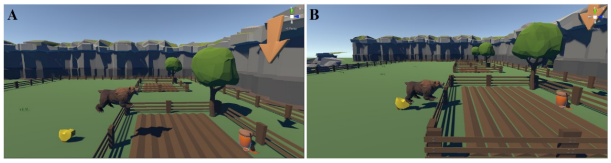

Figure 1: A) Normal trial where the participant is able to jump over the fence. B) Control error trial where the participant loses the ability to jump.

Figure 2: Environment Error Condition: The artificial agent is teleported onto the participant while enlarged, resulting in a jittering effect.

In the environment error conditions, after participants crossed the threshold, the agent was teleported inside the player's avatar while doubling its size, resulting in jitter effects and instant loss of the trial (Fig. 2).
Participants were not informed about the different error conditions, which comprised 30% (15% per error condition) of the total trials. The total number of trials was divided into 10 blocks with a uniform distribution to ensure that only six errors per condition occurred per block. Furthermore, the onset of error trials occurred after participants crossed the specified threshold, randomly selected in each trial.

EEG Pre-processing: EEG recordings were first bandpass filtered (FIR) between 1-30 Hz (filter length of 1651 samples) to remove slow drifts and power line noise. Then, to remove eye artifacts, an Extended Infomax Independent Component Analysis (ICA) [15] was computed with as many components as EEG electrodes (32). The ICA components were then visually inspected and scored by adaptive z-scoring based on the three frontal electrodes (FP1, FPz, FP2). After correcting for eye artifacts, the data were filtered again between 1-15 Hz (FIR) with a filter length of 1651 samples (3.302 sec) and epoched for each condition from 0 to 1 s, where 0 s was the onset of our markers. To ensure that no significant artifacts were maintained in our epochs, we dropped epochs based on maximum peak-to-peak signal amplitude (PTP) with a rejection threshold of 100 x 10^{-6} V. Epochs were then saved per participant to be later used for the training of our models.

Data preparation: The dataset consisted of epochs x channels x time-points. Additionally, we selected only the 11 central electrodes AFz, F1, Fz, F2, FC1, FCz, FC2,

Published by
Verlag der Technischen Universität Graz

10.3217/978-3-99161-014-4-040

C1, Cz, C2, CPz, based on the brain regions frequently associated with the encoding of error processing [16]. Furthermore, to prepare the dataset for our model, min-max normalization was computed per electrode, transforming the amplitude of the electrodes between 0 and 1. Finally, to account for the unbalanced dataset (as normal trials accounted for 60% of the trials), the number of epochs was equalized between the error conditions and the normal trials, with the aim of the remaining epochs occurring as close as possible in time. Thus, we removed those normal trials that fell further in time from the epochs of control and environment error conditions and maintained normal trials that fell closer. Finally, we combined the two error conditions (control, environment) into a single class, resulting in a binary classification task between normal and error trials

Autoencoder Architecture Ana-E: The architecture we developed, termed Ana-E, is a 1D convolutional autoencoder comprised of an encoder, decoder, and an intermediate dense layer for reshaping the encoded representation. The encoder consists of three layers, with each layer applying a 1D convolution with kernel sizes of 32, 64, and 128, respectively. The first layer's input size mirrors the 11 selected electrode, where a 1D convolution with a kernel size of 32 is applied over the 500 time points per electrode.This results in EEG epochs as input, with the first layer's output producing 22 filters. Subsequent layers double the number of filters, culminating in 88 in the final encoder layer. No padding is applied in any layer. For the first two layers, batch normalization and ReLU activation functions are applied after each convolutional layer, while the final encoder layer consists of a 1D convolution with a kernel size of 128, followed by flattening the output. This results in a high-dimensional tensor, which is then passed through a linear layer to reduce the dimensionality to 750.

Between the encoder and decoder, we integrated an intermediate dense layer with a linear transformation from 750 to 88*280 dimensions, reshaping the flattened encoder output for decoding. The decoder network mirrors the encoder, excluding the final flatten and linear layers. Additionally, the decoder's output is passed through a sigmoid function to reconstruct the original EEG signal.

For the EEG feature representations, we utilized the output of our encoder. For the classification task, our classification head (CH) consisted of five linear layers with dimensions 750, 500, 250, 125, 60, and 1, respectively. The first input layer corresponds to the output number of Ana-E's encoder. Each linear layer is followed by a ReLU activation function. The output of the final layer undergoes a sigmoid function for binary classification between normal and error trials.

Training and evaluation: We trained and evaluated Ana-E both within-participants and between participants. For the within-participants case, we split each participant's session into train/val/test splits of 0.6, 0.2, and 0.2, respectively. In the between-participants case, the model was trained on sessions from all participants except the one being tested, resulting in 8 training sessions and 1 testing session. The 8 sessions, after being combined, were split into train and validation sets of 0.8 and 0.2, respectively. This process was conducted iteratively for each participant. The model was trained for 250 iterations with a batch size of 64.

We selected Adam as an optimizer with a learning rate of 1×10^{-3} and weight decay of 1×10^{-6}. For the loss function, we chose the Mean Squared Error (MSE) as we wanted our model to be fine-tuned based on the difference between the original input and the reconstructed output. The most optimal parameters for the models, such as training iterations, batch size, learning rate, weight decay, and the number of neurons of the final linear layer of the encoder, were selected based on GridSearch.

For our classification head (CH), we used 200 training iterations with a batch size of 10 using Adam with a learning rate of 1×10^{-3} and weight decay of 1×10^{-6}. We employed Binary Cross-Entropy (BCE) as the loss function.

Ana-E: Error classification as downstream task: To assess the effectiveness of our architecture in extracting reliable features for developing pre-trained error detection models, we compared the features extracted by Ana-E with the raw (preprocessed) EEG trials within and between participants. In each comparison, we employed two classifiers: our CH and Linear Discriminant Analysis (LDA) [17], resulting in four different pipelines: AnaE-LDA, AnaE-CH, RAW-LDA, and RAW-CH. In the RAW pipelines, we flattened the 3D EEG trials into 2D. Each epoch's input for LDA and CH in the RAW pipelines consisted of 11 electrodes multiplied by 500 time points. To meet the specified input size of 750 in the first layer for CH, we added an extra layer with an input of size 11 * 500 and an output size of 750.

We evaluated the quality of each pipeline and its ability to differentiate between classes by examining accuracy scores, True Negative Rates (TNRs), and True Positive Rates (TPRs) for 2-class classification within and between participants.

RESULTS

First, to assess the quality of our feature extractor in the within-participants case, we provide the grand average (GA) per pipeline, together with each participant's accuracy and TNRs and TPRs. We observe that the best GA is achieved by RAW-CH (M = 79%, SD = 0.08). For the second-best performance, both AnaE-LDA (M = 78%, SD = 0.11) and AnaE-CH (M = 78%, SD = 0.11) performed equally well, while the worst accuracy was achieved from RAW-LDA (M = 73%, SD = 0.13).

Further inspection of the accuracy per participant reveals that the top three pipelines performed equally well across participants, as each pipeline resulted in the best performance across three participants. Differences in GA are reflective of the variation of classifiers' performances within each participant. For example, our custom classi-

Published by
Verlag der Technischen Universität Graz

10.3217/978-3-99161-014-4-040

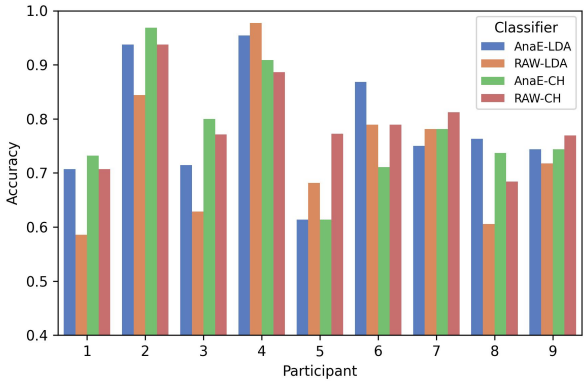

Figure 3: Within-Participants: Classification accuracy

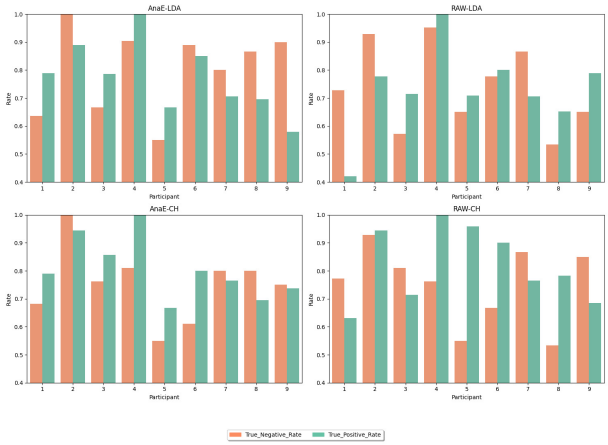

Figure 4: True Negative and True Positive Rates: Within-Participants

fication head seems to achieve the best GA due to the difference it has with the other classifiers in participant 6 and not due to being the most optimal classification method. Additionally, we notice that all pipelines across all participants scored higher than the binary classification chance level (50%), with the best performances achieved in participants 3 and 5 (>80%). Our top three pipelines consistently perform around the 70 mark for almost all participants. (Fig.3).

To gain a better understanding of the quality of the classification made by our tested pipelines, we further investigated the TNRs and TPRs (Fig. 4). We observed that the LDA pipelines predict both classes more equally, with the normal class being predicted slightly more frequently. In contrast, the Ana-E pipelines seem to predict the error condition more strongly, as evidenced by the average TNRs and TPRs (Fig. 5).

Similarly, to assess the quality of our approach between participants, we provide the GA for each pipeline together with the accuracy performances per participant and then TNRs and TPRs. The best GA is reached by AnaE-CH with a score of M = 72% (SD = .07), the second-best metric is achieved by both RAW-CH (M = 64%, SD = .14), and AnaE-LDA (M = 64%, SD = .05).

By investigating the accuracy metric per participant, we

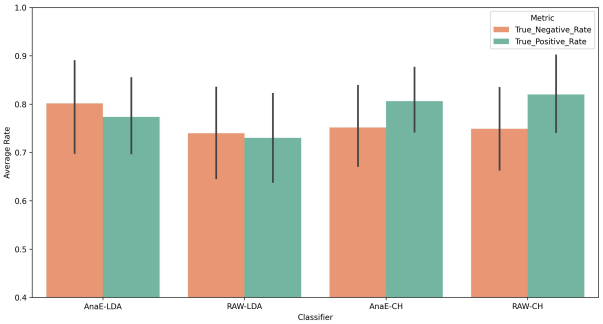

Figure 5: Average True Negative and True Positive Rates: Within-Participants

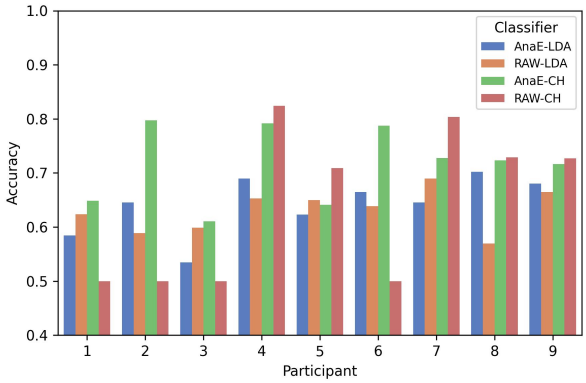

Figure 6: Between-Participants: Classification accuracy

observe that our pipeline (Ana-E) performed above average in all participants and achieved the best accuracy in 4 out of 9 participants. Additionally, we notice that our custom classifier coupled with the preprocessed data (epochs) performed the best in 5 out of the 9 participants but next to chance levels in the remaining participants (Fig. 6).

By further inspecting the average TNRs and TPRs per pipeline, we notice that pipelines utilizing our feature extractor perform the best in terms of error condition recognition. Although RAW-CH seems to achieve the best performances in 5 out of the 9 participants in terms of accuracy metrics, we now notice that the classifier mainly learns to predict the normal trials and performs poorly in terms of error detection.

Furthermore, by inspecting the average TNRs and TPRs, we can deduce that when classifiers were utilizing the features extracted by our encoder, they were firstly able to better predict error conditions while secondly maintaining more stable performances across the different participants (Fig. 8).Finally, by inspecting the TNRs and TPRs per participant we can observe the effects that are responsible for the below chance level of error condition predictions as in 4 out 9 participants the custom classifier trained on the preprocessed data predicts every class as normal trials (Fig.7).

DISCUSSION

Published by
Verlag der Technischen Universität Graz

10.3217/978-3-99161-014-4-040

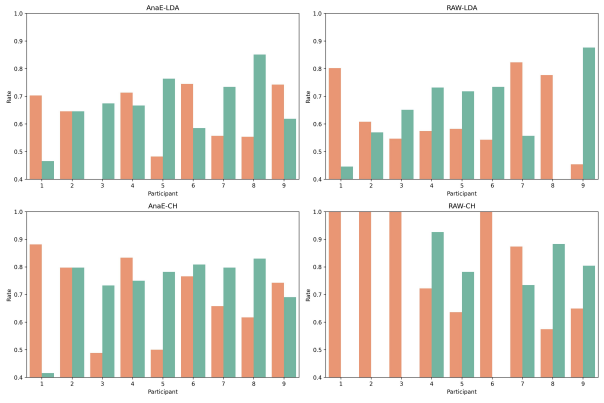

Figure 7: True Negative and True Positive Rates: Between-Participants

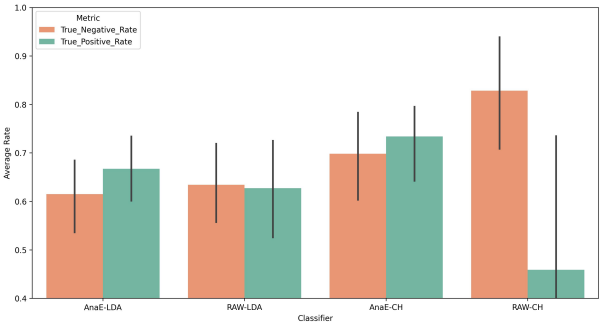

Figure 8: Average True Negative and True Positive Rates: Between-Participants

The study aimed to explore the development of pre-trained error detection models using an autoencoder in a semi-supervised learning (SSL) setup. Initially, a game simulation was designed in Unity, where participants navigated obstacles, with 30% of trials containing simulated errors. EEG data were recorded during the game. We adapted an autoencoder, termed Ana-E, for SSL, training it on a reconstruction task and utilizing EEG features for classification. Our pipeline (AnaE) was compared against preprocessed EEG data (RAW) using two classifiers. Both within-participant and between-participant analyses were conducted. In the within-participant analysis, the top three approaches (RAW-CH, AnaE-LDA, AnaE-CH) performed equally. In the between-participant analysis, AnaE-CH outperformed RAW-CH and AnaE-LDA.

Our within-participant analysis offers insights into the performance of our architecture within a classical BCI framework, where models are trained and tested on the same participant session. Results suggest comparable performance among our top three pipelines (RAW-CH, AnaE-LDA, AnaE-CH), with minor variation. Further examination of TNRs and TPRs reveals a tendency for our custom architectures (RAW-CH and AnaE-CH) to exhibit stronger predictions for positive class instances (error conditions). Moreover, it is noteworthy that the sample size for training and testing within participants was significantly smaller compared to between-participant analysis. Given the complexity of Ana-E, the available data may not have been sufficient for our model to learn highly discriminable features. Potentially a solution could be implemented by integrating variational autoencoders (VAE) [18] or other generative models [19] in the processes and amplifying the total number of EEG samples [20]. Furthermore, while the within-participant analysis highlights the usability of our architecture as a "classical" BCI paradigm, the between-participant analysis will provide evidence regarding the feasibility of developing pre-trained error detection models.

The between-participants analysis provided insight into the potential of pre-trained models, as it underscores the capacity of an architecture to generalize to unseen participants while being trained on a sizable dataset comprising multiple individuals. However, in the context of BCIs, certain constraints, such as the non-stationarity of EEG recordings [21], hinder the application of classical training approaches similar to those used in image processing. In this study, we implemented an architecture designed to learn generalizable features from an unsupervised task (pretext) and subsequently transfer the learned EEG feature representations to a downstream classification task. Our architecture (AnaE-CH) achieved the highest GA, surpassing the second-best approach by 8%. Further examination of the TNRs and TPRs revealed that our classifiers were more reliable in predicting the error condition only when our encoder (AnaE) was used to extract features. Conversely, when preprocessed trials were utilized to train the classifiers, they primarily predicted normal trials and struggled to identify the error condition. Our approach provides support for the idea of generalizable features across participants, laying the foundation for pre-trained error detection models. Integration of such models into classical BCI scenarios could potentially reduce the need for calibration sessions.

In the current study, there were certain limitations that could have hindered the performance of our architecture, such as the lack of sufficient datasets in our within-participants analysis. The small sample size for training and testing within participants might have limited Ana-E's robustness and generalizability in terms of feature learning. Insufficient data can impact the model's capacity to learn complex EEG signal patterns and features, potentially leading to suboptimal performance. A potential solution could involve the integration of VAE [18] or GANs [19] to amplify the number of trials within participants without increasing the duration of participants' sessions. By doing so, we could further examine the quality of our approach as a classical BCI pipeline. Furthermore it should be highlighted that the results of this study are based on an offline analysis where excessive attenuation of artifacts was possible.

Moreover, our study diverged from the classical autoencoders. Typically, autoencoders aim to reduce dimensions, but in our case, the number of dimensions increases with each layer. Existing literature suggests that autoencoders with large inter-layer dimensions may sim-

Published by
Verlag der Technischen Universität Graz

10.3217/978-3-99161-014-4-040

ply copy input to output without learning meaningful features [22]. Despite the increasing dimensions, our model achieved improved classification performance based on the learned features, indicating meaningful feature representations from our encoder.Further investigation could explore classical autoencoder principles to determine if the improvements seen in this study stem from our architectural choices.

CONCLUSION

In conclusion, our study investigated the feasibility of developing pre-trained models for error detection using an autoencoder architecture in a semi-supervised learning (SSL) setting. We designed an experimental setup simulating a game scenario to collect EEG data, which were then employed to train our proposed approach. The results demonstrated comparable performance across different pipelines in the within-participant analysis and a notable enhancement in classification performance in the between-participant analysis when utilizing Ana-E. Particularly promising was our architecture's ability to generalize to unseen participants, indicating its potential utility in real-world applications. Moving forward, further research should explore alternative architectural modifications to enhance the adaptability and robustness of Ana-E. Overall, our study provides valuable insights into the opportunity of developing pre-trained models for error detection in BCI scenarios, laying the foundation for future advancements in the field.

REFERENCES

[1] Dias CL, Sburlea AI, Müller-Putz GR. Masked and unmasked error-related potentials during continuous control and feedback. Journal of Neural Engineering. 2018;15(3):036031.
[2] Omedes J, Iturrate I, Minguez J, Montesano L. Analysis and asynchronous detection of gradually unfolding errors during monitoring tasks. Journal of Neural Engineering. 2015;12(5):056001.
[3] Lopes-Dias C, Sburlea AI, Müller-Putz GR. Online asynchronous decoding of error-related potentials during the continuous control of a robot. Scientific Reports. 2019;9(1):17596.
[4] Krauledat M, Tangermann M, Blankertz B, Müller KR. Reducing calibration time for brain-computer interfaces: A clustering approach. Jan. 2006, 753–760.
[5] Jing L, Tian Y. Self-supervised visual feature learning with deep neural networks: A survey. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2021;43(11):4037–4058.
[6] Larsson G, Maire M, Shakhnarovich G. Colorization as a proxy task for visual understanding. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2017, 840–849.
[7] Misra I, Zitnick CL, Hebert M. Shuffle and learn: Unsupervised learning using temporal order verification. In: Computer Vision – ECCV 2016. Springer International Publishing: Cham, 2016, 527–544.
[8] Korbar B, Tran D, Torresani L. Cooperative learning of audio and video models from self-supervised synchronization. In: Advances in Neural Information Processing Systems. Curran Associates, Inc., 2018.
[9] Zerveas G, Jayaraman S, Patel D, Bhamidipaty A, Eickhoff C. A transformer-based framework for multivariate time series representation learning. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining. Association for Computing Machinery: Virtual Event, Singapore, 2021, 2114–2124.
[10] Zha M, Wong S, Liu M, Zhang T, Chen K. Time series generation with masked autoencoder. 2022.
[11] Banville H, Chehab O, Hyvärinen A, Engemann DA, Gramfort A. Uncovering the structure of clinical eeg signals with self-supervised learning. Journal of Neural Engineering. 2021;18(4):046020.
[12] Tschannen M, Bachem O, Lucic M. Recent advances in autoencoder-based representation learning. 2018. arXiv: 1812.05069 [cs.LG].
[13] Michelucci U. An introduction to autoencoders. 2022. arXiv: 2201.03898 [cs.LG].
[14] Haas JK. "A history of the unity game engine." Worcester. Tech. Rep. 2014. [Online]. Available: https://digital.wpi.edu/show/2f75r821k.
[15] Lee TW, Girolami M, Sejnowski T. Independent component analysis using an extended infomax algorithm for mixed sub-gaussian and super-gaussian sources. Neural Computation. 1999;11:417–441.
[16] Rousseau S, Jutten C, Congedo M. The error-related potential and bcis. 2012.
[17] Mika S, Ratsch G, Weston J, Scholkopf B, Mullers K. Fisher discriminant analysis with kernels. In: Neural Networks for Signal Processing IX: Proceedings of the 1999 IEEE Signal Processing Society Workshop (Cat. No.98TH8468). 1999, 41–48.
[18] Kingma DP, Welling M. An introduction to variational autoencoders. Foundations and Trends® in Machine Learning. 2019;12(4):307–392.
[19] Ruthotto L, Haber E. An introduction to deep generative modeling. 2021.
[20] Luo Y, Zhu LZ, Wan ZY, Lu BL. Data augmentation for enhancing eeg-based emotion recognition with deep generative models. 2020.
[21] Krumpe T, Baumgärtner K, Rosenstiel W, Spüler M. Non-stationarity and inter-subject variability of eeg characteristics in the context of bci development. Sep. 2017.
[22] Bourlard H, Kabil S. Autoencoders reloaded. Biological Cybernetics. 2022;116.

Published by
Verlag der Technischen Universität Graz