BioMedInformatics

MDPI

Article

Physiological Data Augmentation for Eye Movement Gaze in Deep Learning

Alae Eddine El Hmimdi [1,2,*] and Zoï Kapoula [1,2,*]

[1] Orasis-Eye Analytics & Rehabilitation Research Group, Spinoff CNRS, 12 Rue Lacretelle, 75015 Paris, France
[2] LIPADE, French University Institute (IUF), Laboratoire d'Informatique Paris Descartes, University of Paris, 45 Rue des Saints Pères, 75006 Paris, France
* Correspondence: alae-eddine.el-hmimdi@etu.u-paris.fr (A.E.E.H.); zoi.kapoula@gmail.com (Z.K.)

Abstract: In this study, the challenges posed by limited annotated medical data in the field of eye movement AI analysis are addressed through the introduction of a novel physiologically based gaze data augmentation library. Unlike traditional augmentation methods, which may introduce artifacts and alter pathological features in medical datasets, the proposed library emulates natural head movements during gaze data collection. This approach enhances sample diversity without compromising authenticity. The library evaluation was conducted on both CNN and hybrid architectures using distinct datasets, demonstrating its effectiveness in regularizing the training process and improving generalization. What is particularly noteworthy is the achievement of a macro F1 score of up to 79% when trained using the proposed augmentation (EMULATE) with the three HTCE variants. This pioneering approach leverages domain-specific knowledge to contribute to the robustness and authenticity of deep learning models in the medical domain.

Keywords: data augmentation; deep learning; saccade; vergence; time series; eye movement

Citation: El Hmimdi, A.E.; Kapoula, Z. Physiological Data Augmentation for Eye Movement Gaze in Deep Learning. *BioMedInformatics* **2024**, *4*, 1457–1479. https://doi.org/10.3390/biomedinformatics4020080

Academic Editors: Moulay A. Akhloufi and Mufti Mahmud

Received: 6 March 2024
Revised: 8 April 2024
Accepted: 21 May 2024
Published: 6 June 2024

1. Introduction

Deep learning has greatly influenced a range of fields, such as computer vision, signal processing, and natural language processing. However, training deep convolutional architectures from scratch necessitates a substantial amount of data compared to traditional machine learning algorithms. This requirement becomes even more crucial when considering Transformer architecture. It is widely recognized that Transformers are data-intensive [1–4], relative to traditional CNN architectures. The advantage of this architecture lies in its enhanced model expressivity and ability to effectively learn complex tasks given ample data. Yet, the use of small datasets often leads to overfitting issues and inadequate generalization to unseen data.

Moreover, collecting annotated data in the medical domain poses additional challenges, especially when human involvement is required, due to the sensitive nature of the information gathered. Additionally, obtaining an adequate number of individuals with specific target pathologies can be complex. To overcome the issue of limited dataset size, data augmentation techniques are commonly used to artificially increase the number of training samples. These techniques involve sampling new data by applying various transformations [5,6] or interpolating new samples based on existing ones [7–9].

However, for medical datasets, particularly those related to pathologies with distinct morphologies and structural characteristics, using such augmentation methods may not be appropriate as they can introduce artifacts and alter pathological features. Mixing-based algorithms like CutMix and MixUp, which assume a linear relationship between input and label, may also have drawbacks in this context.

As a result, pathologists and researchers often use specialized augmentation methods tailored to the characteristics of pathology patterns. These techniques involve learning the data distribution using generative approaches and then sampling from it. However,

these methods are considered less effective in capturing complex or rare patterns compared to transformation-based techniques that generate diverse high-resolution samples. When trained on small-sized datasets with high-resolution images, these generative methods may not adequately capture all the patterns present in the data, as is evident in tasks like eye movement gaze classification where a two-minute recording corresponds to a multivariate time series consisting of approximately 24,000 points.

To address the challenges of limited annotated medical data, one potential solution is to enhance sample diversity by incorporating realistic physiological variations instead of directly learning the distribution. This approach leverages domain-specific knowledge and generates samples that align with the inherent characteristics of physiological data, contributing to robustness and authenticity. In this study, we introduce a physiologically based gaze data augmentation library that emulates head movements during data collection, capturing natural variability and intricacy in eye movement patterns.

The contributions are as follows:

- We introduce EMULATE, a novel library for eye movement gaze data augmentation. Named EMULATE, which stands for Eye Movement data Augmentation by Emulating Head Position and Movement, this tool pioneers its category by emulating physiological aspects. The library generates augmented eye movement data by simulating natural head movements, both prior to recording and in real-time during gaze data collection.
- We evaluate the data augmentation technique on three distinct architectures—two based on CNN and one hybrid, utilizing two separate datasets.
- We explore various augmentation settings, demonstrating the effectiveness of the proposed library in regularizing the training process of the proposed architecture and improving its generalization ability.
- We examine the complementarity between the proposed method and additional standard baseline approaches.

This paper is structured as follows: It begins with a summary of the state of the art for data augmentation, followed by an overview of the studies introducing the various architectures used. Additionally, detailed information is provided on the materials utilized in this study, including the eye movement recording setup and the resulting dataset, introduced in [10]. In this study, we explore the relevance of the proposed data augmentation method by integrating it into the existing training framework [10]. Thus, the three models are initially trained with and without the proposed method. Subsequently, comparisons are made between training with EMULATE and training with other baseline methods. Finally, we analyze the complementarity between EMULATE and the proposed baselines. In the methods section, we present the proposed data augmentation library, along with the experimental settings used to evaluate the significance of these methods. This includes the architecture used for training, as well as the augmentation and regularization methods for comparison. Implementation details such as model training and evaluation methods are also provided, together with the various hyperparameters used for the architecture, EMULATE, and the model training pipeline.

Furthermore, the experimental results are discussed in the results section and elaborated upon in the discussion section. Finally, the limitations and future directions of the proposed method are reviewed.

2. Related Work

In deep learning, data augmentation methods can be grouped into two groups: implicit distribution learning and explicit transformation modeling. The first group involves learning the underlying data distribution and sampling from it, while the second group focuses on modeling transformations to generate new samples based on existing ones.

2.1. Implicit Data Augmentation Methods

Several studies have explored the use of domain-adapted data augmentation methods, such as learning the underlying data distribution using sequence-to-sequence algorithms [11] or generative methods [12–16].

Zemblys et al. [11] studied eye-movement events through a supervised learning algorithm applied to recurrent networks, and then sampled from it to augment training sets. Additionally, when concidering generative methods, Fuhl [13] used variational autoencoders to learn the distribution of eye movement gaze and evaluated their method across three different datasets. Similarly, Ref. [12] learned to generate image-based scanpath representations and reported an improvement of up to 0.05 in the AUC score for augmented data in ASD classification tasks.

In similar studies involving electroencephalography time series data, the same approach of learning the data distribution was applied [15], by exploring, using conditional VAEs, to learn EEG distributions. Similarly, a second study [14] also studied several variants of CVAEs, GANs, and VAEs algorithms, observing an improvement of up to 10.2%.

Finally, in a different approach [16], investigated directly learning the distribution of extracted features. They trained a generative adversarial network to generate artificial EEG and eye movement parameters for a multimodal emotion recognition task, achieving an accuracy of up to 90.33%.

2.2. Explicit Transformation Modeling

On the other hand, many data augmentation techniques have been developed in the field of computer vision to improve the performance and robustness of deep learning models. Examples include RandAugment, AutoAugment, MixUp, CutMix, and Cutout [5,7,8,17,18].

RandAugment [5] and AutoAugment [17] are two transformation modeling techniques that can be used to apply random augmentations to images and improve the robustness of feature representation. RandAugment utilizes a set of predefined transformation operations for random augmentation while AutoAugment learn a set of optimal transformation, to effectively augment data and enhance feature representation.

On the other hand, MixUp, CutMix, and Cutout [7,8,18] generate new samples by combining pairs of two existing samples.

MixUp interpolates pairs of training examples to generate new samples, while CutMix combines two randomly selected samples by swapping a patch from one image onto another while preserving label information.

Lastly, Cutout, consist of masking out, square regions of an input image during training to encourage exploiting on other areas.

While data augmentation techniques like CutMix and MixUp are commonly used in the deep learning community to enhance generalization and robustness, these methods are less utilized in the medical classification domain as they may compromise the integrity and diagnostic value of data. Instead, the focus is on implementing data augmentation techniques that preserve essential diagnostic information within images. It is worth mentioning that these augmentations have not yet been applied to eye movement gaze classification studies.

Another type of explicit data augmentation technique involves learning a domain-adapted method, using algorithms that model transformations to increase the sample size while preserving the characteristics of the data manifold. Thus, as opposed to the first type discussed above, the generated data is part of the real data distribution.

For example, while several transformations of the Autoaugment library can be considered domain-adapted for computer vision tasks, they are not domain-adapted for the latter. when applied to eye movement positions in time series.

To the best of our knowledge, explicit transformation modeling methods for eye movement time series are lacking. To address this gap, we introduce a physiological data augmentation method.

2.3. Hierarchical Temporal Convolutions for Eye Movement Analysis

In a previous study [19], we investigated the screening of scholar-learning disorders using deep learning applied to clinical data. This dataset included 4243 time series of eye movement positions recorded from children across Europe. We introduced the hierarchical temporal convolutions for eye movement analysis (HTCE), a CNN architecture composed of multiple hierarchical convolutional blocks followed by a multi-layer perceptron for classification. The proposed method achieved precision and recall rates of up to 80.20% and 75.1%, respectively, when evaluated on clinical data. These results are significant considering the high variability in both input and label, particularly compared to research datasets collected under consistent protocols, with control populations consisting of healthy subjects. This setting reflects real-world scenarios more accurately, where the negative class (control) contains populations with various pathologies, making the screening task more challenging.

2.4. Multi-Segment HTCE-Based Classifier

In another study [10], we took a step further by extending the previous method to incorporate a multi-segment-based classifier. This classifier was trained to identify eight groups of pathologies, instead of exclusively focusing on screening scholar disorders, thus transitioning from a binary to a multi-label classification problem. Initially, 10 segments were randomly sampled from each recording, and each segment was processed using HTCE variants to generate embeddings. Subsequently, these embeddings were aggregated to provide a comprehensive prediction. Two aggregation strategies were explored:

- Pooling-based aggregation: Two CNN architectures, HTCE-Max and HTCE-Mean, are introduced. Each classifier consists of two stages. The initial stage constructs an embedding from a segment of eye movement recordings, employing a refined version of the HTCE classifier proposed in [10]. However, the second stage, varies in implementation. The HTCE-Max architecture aggregates the embeddings using a max-pooling layer, followed by processing the resulting feature map with a multi-layer perceptron. Similarly, HTCE-Mean employs mean-pooling instead.
- Attention-based aggregation: a second hybrid architecture is introduced, which first utilizes a lightweight version of HTCE to construct a high-dimensional representation for each segment. Subsequently, second-level feature extraction is performed using the VIT encoder [20]. This approach yields a hybrid architecture where the first stage conducts feature extraction at the temporal level, while the second stage operates at the segment level.

Additionally, contextual information is incorporated, including time series and gaze derivatives. We provide a brief overview of each time series:

- Gaze derivative: Corresponding to the first and second derivatives (velocity and acceleration) for each of the four eye movement time series.
- REMOBI target signal: Encodes the state of the different LEDs, allowing the model to infer information such as the latency.
- LED coordinates: Encodes the coordinates of the activated LED within the optical axis.
- Confidence level: Represents an estimation of the uncertainty of each eye movement coordinate estimation by the eye tracker.

In this study, the relevance of the proposed data augmentation is evaluated by utilizing the previous training setup, while incorporating the proposed augmentation method.

3. Material

3.1. Eye Movement Recording

Eye movements were recorded using the Pupil Core head-mounted video-oculography tool, which measures angular positions along the vertical and horizontal axes, forming a plane perpendicular to the optical axis, at a frequency of approximately 200 Hz.

The data collection process was stored anonymously during the eye movement analysis and complied with European regulations regarding personal data protection.

The clinical data were gathered from 20 different European clinical centers, where two technologies—REMOBI technology (patent WO2011073288) and AiDEAL technology (patent PCT/EP2021/062224)—were used to test and analyze various types of eye movement including saccade and vergence eye movements.

3.2. Eye Movement Visual Tasks

In this study, two visual tasks were explored: the saccade task and the vergence task. In the saccade task, participants responded to stimuli that appeared randomly along a horizontal axis, with analyses focusing on eye movements and fixation post-movement. The vergence task involved observing both convergent and divergent eye movements as participants fixated on a stimulus presented at various positions and durations along the optical axis.

To prevent participants from predicting motion, the duration and position of LEDs were randomized in both tests. Each test included 40 trials: 20 leftward and 20 rightward for the saccade test, and 20 coordinated and 20 uncoordinated for the vergence test.

3.3. Problem Statement

Our dataset, denoted as D and comprising N instances for $i \in [1, N]$, consists of pairs representing multivariate time series $X_i \in \mathbb{R}^{15 \times T}$ of length T and a corresponding target class y_i. The input features include both the horizontal and vertical angular positions of each eye over the duration T, along with the first two order derivatives (velocity and acceleration), as well as contextual time series, namely, the latency, LED coordinates relative to the optical axis, and confidence level. The objective is to predict the class y_i based on input X_i.

This work builds upon previous studies [10]; thus, we reuse the same problem formulation; Our objective is to tackle a multi-annotation problem by predicting the vector classes y_i based on input X_i. Additionally, to reduce the model's sensitivity to segments with high levels of noise, a multi-segment approach is adopted, with 10 segments of size $S = 1024$, corresponding to 50 s of recording.

We evaluate the relevance of the proposed data augmentation by integrating it into the existing training framework developed in previous studies. Thus, each of the three models is initially trained with and without the proposed method. Subsequently, we compare training with EMULATE and training with other baselines. Finally, we examine the complementarity between EMULATE and the proposed baselines.

3.4. Dataset Overview

We utilized the Ora23 dataset previously introduced in [10], which encompassed two distinct datasets, corresponding to two different visual tasks (saccade and vergence).

The saccade visual task was composed of 92,207 segments of 5 s duration, recorded from 3181 subjects. Similarly, the vergence visual task consisted of 95,630 segments performed by 3228 subjects. For both visual tasks, the mean duration of each recording was approximately 3 min. Note that the Ora23 dataset was generated using the same method as described in the previous study [19] for constructing the Ora22 dataset. However, it included annotated data gathered between 2022 and 2023.

Table 1 presents the corresponding group of pathologies for each class identifier, as well as the corresponding patient count for each of the two datasets, namely the saccade and the vergence datasets. It is noteworthy that there are similarities between the class

distributions in the two datasets; in the majority of cases, the same clinical protocols are used, involving performing both saccade and vergence tests.

Table 1. Presentation of the different groups of pathologies and the patient count for the saccade and the vergence datasets.

Class Identifier	Corresponding Disorder	Saccade Dataset	Vergence Dataset
0	Dyslexia	873	854
1	Reading disorder	1264	1265
2	Listening and expressing	331	321
3	Vertigo and postural	396	372
4	Attention and neurological	1016	975
5	Neuro-strabismus	455	511
6	Visual fatigue	678	567
7	Other pathologies	195	279

3.5. Data Processing

This section presents the data preprocessing steps to sample the different training batches from the dataset, following the methodology outlined in a previous study [10]. We provide a concise summary of these key procedures.

Initially, two levels of data cleansing are performed: a low-pass filtering step using a Gaussian FIR filter with a cut-off frequency of 33 Hz and a z-score filtering step, eliminating data points with z-scores exceeding 2.5. Each time series recording undergoes individual filtering using its own statistics computed from the entire recording.

Additionally, to standardize each angular coordinate, the modulo angular coordinate value of 180 is computed, and each coordinate is then divided by 180. Finally, for contextual features, min–max standardization is employed as an alternative method.

4. Methods

The proposed approach utilizes physiologically based transformation techniques to augment eye movement gaze data. To overcome the challenges, two strategies for augmenting the dataset are proposed.

- Static: This involves emulating head movements made before data acquisition without affecting head stability during acquisition. It incorporates nine parameters.
- Dynamic: This method focuses on emulating head movements during data acquisition and allows for more extensive augmentation of the dataset. It includes 15 parameters.

4.1. Motivation

The methodology incorporates the use of the REMOBI system, allowing for unrestricted movement of the subject's head instead of it being fixed. Additionally, accelerometer measurements have been included in the new data recordings to analyze head movements. Observations indicate a consistent slight variation in the initial head position and the presence of small ongoing movements. These parameters are introduced to augment highly physiological data, inspired by this physiological variability.

4.2. Algorithm

Figure 1 shows a schematic of the proposed model, with the two pupils and the center of the head projected in a two-dimensional plane. For simplicity, we will focus on the case of the right eye. Note that similar formulas are used to rotate the vector for the left eye as well.

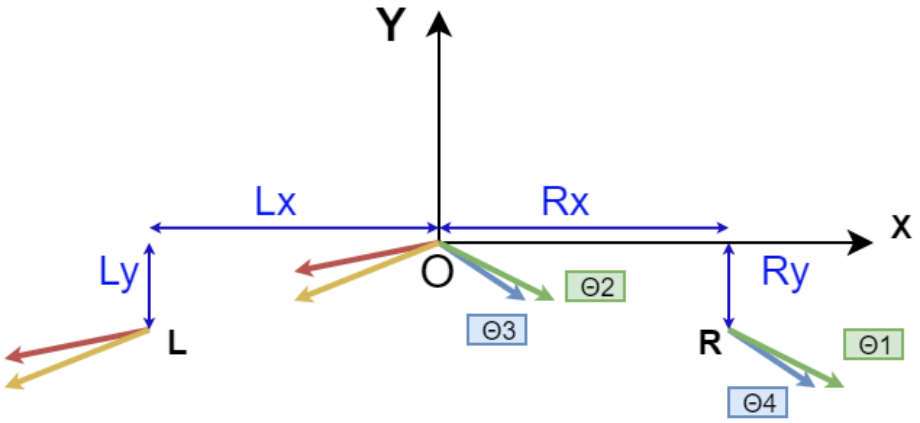

Figure 1. Illustration of the physical model used to build the proposed data augmentation method. Point R corresponds to the position of the right eye pupil center. Point L corresponds to the position of the left eye pupil center. Point O corresponds to the center of the referential system, as well as the position of the head center. Illustration of the plane (OY, OX) where the pupil and head center.

Let *rot* represent a rotation matrix, denoted as rot(β, γ, α), where β, γ, and α are the respective angles of rotation around the x-axis, y-axis, and z-axis.

$$\begin{bmatrix} \cos(\beta)\cos(\gamma) & \sin(\alpha)\sin(\beta)\cos(\gamma) - \cos(\alpha)\sin(\gamma) & \cos(\alpha)\sin(\beta)\cos(\gamma) + \sin(\alpha)\sin(\gamma) \\ \cos(\beta)\sin(\gamma) & \sin(\alpha)\sin(\beta)\sin(\gamma) + \cos(\alpha)\cos(\gamma) & \cos(\alpha)\sin(\beta)\sin(\gamma) - \sin(\alpha)\cos(\gamma) \\ -\sin(\beta) & \sin(\alpha)\cos(\beta) & \cos(\alpha)\cos(\beta) \end{bmatrix}$$

Multiplying a vector (x,y,z) by this matrix corresponds to applying rotations with angles beta, gamma, and alpha around the first, second, and third axes, respectively. The proposed algorithm consists of the following steps:

- 1. To find Θ_1 in Cartesian coordinates, the spherical coordinate triplet (α, β, r) is converted to the Cartesian reference system (x, y, z) using this equation.

$$x = \phi \sin(\alpha)\cos(\theta) \quad (1)$$
$$y = \phi \sin(\alpha)\sin(\theta) \quad (2)$$
$$z = \phi \cos(\alpha) \quad (3)$$

- 2. Each eye position vector is translated from the pupil center position (Θ_1) to the head center position (Θ_2).

$$x_{\text{head}} = x - R_x \quad (4)$$
$$y_{\text{head}} = y - R_y \quad (5)$$
$$z_{\text{head}} = z \quad (6)$$

- 3. The head rotation transformation involves rotating the head within the three axes. This operation is achieved by multiplying the vector (Θ_2) using the rotation matrix rot(β, γ, α).

$$\begin{bmatrix} x_{\text{headbar}} \\ y_{\text{headbar}} \\ z_{\text{headbar}} \end{bmatrix} = \text{Rot}(\alpha, \beta, \gamma) \cdot \begin{bmatrix} x_{\text{head}} \\ y_{\text{head}} \\ z_{\text{head}} \end{bmatrix} \quad (7)$$

- 4. Each eye position vector (Θ_3) is translated back to its corresponding eye coordinate (Θ_4).

$$x_{\text{bar}} = x_{\text{headbar}} + R_x \tag{8}$$
$$y_{\text{bar}} = y_{\text{headbar}} + R_y \tag{9}$$
$$z_{\text{bar}} = z_{\text{headbar}} \tag{10}$$

- 5. Each eye coordinate is converted back to spherical coordinates using the following equation:

$$r = -\sqrt{x^2 + y^2 + z^2} \tag{11}$$
$$\theta = \arccos\left(\frac{z}{r}\right) \cdot \frac{180}{\pi} \tag{12}$$
$$\phi = \arctan 2(y, x) \cdot \frac{180}{\pi} \tag{13}$$

Two data augmentation strategies are proposed based on this algorithm. The first strategy involves using a shared rotation matrix along the temporal axis, resulting in static data augmentation. In contrast, the second method samples different rotation matrices for each point within the temporal axis in each batch, leading to dynamic data augmentation.

4.2.1. Static Data Augmentation

The static version involves mimicking head movements before data acquisition without disrupting the stability of the head information during acquisition. To implement the proposed algorithm, we utilize eye and head coordinates from a study [21] to obtain a list of these coordinates for 10 subjects.

This method incorporates four parameters: three rotation angles and the index from the table of subject coordinates used for computation.

4.2.2. Dynamic Data Augmentation

This study explores advanced strategies for replicating dynamic head movements during data acquisition. A significant difference between the new approach and the previous one involves the handling of the rotation matrix. In the previous approach (static), the rotation matrix is shared within the temporal axis. However, in this new approach (dynamic), the real-time movement of a human head is modeled using a sinusoidal function parameterized by its initial angle, maximum angle, and period. Consequently, the rotation matrix is sampled differently for each time step based on a specific equation.

$$\gamma(t) = \gamma(0) + \gamma_{\max} \cdot \sin\left(\frac{t}{2\pi\gamma_{\text{period}}}\right) \tag{14}$$
$$\beta(t) = \beta(0) + \beta_{\max} \cdot \sin\left(\frac{t}{2\pi\beta_{\text{period}}}\right) \tag{15}$$
$$\theta(t) = \theta(0) + \theta_{\max} \cdot \sin\left(\frac{t}{2\pi\theta_{\text{period}}}\right) \tag{16}$$

In contrast to the previous method, where each batch required constructing a rotation matrix by sampling three angles (gamma, beta, and theta), this approach involves sampling three additional maximal angles and three periods. This increases the total number of free parameters from 4 to 10.

4.2.3. Interpolating Different Subject Coordinates

Different eye and head coordinates are sampled from the table presented in [21]. This table includes the left and right eye coordinates, as well as head coordinates, for 10 different

subjects. To enhance the variety of the output space, for each batch, the eye and head coordinates are dynamically generated using the following approach:

1. Randomly sample eye and head coordinates from two subjects.
2. Generate a scalar value within a specified range.
3. Construct new eye and head coordinates by linearly interpolating between the coordinate systems of the two selected subjects.

4.2.4. Radius Approximation

To convert data to the Cartesian coordinate system, angular values within the x- and y-axes are necessary along with the radius of each point. However, the eye tracker employed does not directly estimate the radius; instead, an approximation is made using the coordinates of the stimulus along the optical axis. This approximation is based on the hypothesis of optimal convergence and vergence in terms of the amplitude when focusing on an LED. It is crucial to emphasize that this approximation serves solely for converting between spherical and Cartesian coordinates.

4.3. Experimental Setting

To assess the importance of EMULATE, the three architectures are trained using the three different setups of the proposed data augmentation methods (static, dynamic, and dynamic high) and are compared first with a regime where no augmentation is applied, and then with a regime where several non-physiological data augmentation and regularization techniques are explored. Finally, the complementarity between the proposed methods and the different baseline methods is explored.

4.3.1. Incorporating the Augmentation Method Within the Three HTCE Variant Training Sessions

Firstly, the significance of the proposed augmentation method is assessed by integrating it into the initial training setup. For each of the two datasets (saccade and vergence) and the three architectures—HTCE-MAX, HTCE-MEAN, and HTCSE—a training session is conducted using the three different setups of the proposed data augmentation methods (static, dynamic, and dynamic high), and compared against a traditional regime where no augmentation is involved.

It is important to note that for these experiments, as well as all the experiments presented subsequently, the dilation mechanism is disabled to reduce training costs. The objective of these studies is to compare data augmentation methods rather than achieve the best performance. Thus, the three dilated layers and the subsequent concatenation module are replaced with a single convolution layer. This layer has a number of parameters equal to the sum of the parameters of the three previous dilated convolution layers, along with similar hyperparameters.

4.3.2. Comparing EMULATE with Other Augmentation Methods

To assess the relevance of the proposed augmentation library, in addition to a comparison with a no-augmentation regime, the three different architectures are trained with multiple standard regularization and data augmentation methods widely used in the deep learning community:

- Dropout [22]: A dropout layer with a rate of 0.2 is inserted after each ConvBlock, Attention Block, and at the input of the MLP.
- CutMix [7]: CutMix data augmentation is employed with default parameters (alpha set to 1.0), utilizing the Keras implementation [23].
- Cutout [18]: The Keras implementation [24] is utilized with default parameters (height and width factors set to 0.2).
- MixUp [8]: The Keras implementation [25] is utilized with default parameters (inverse scale parameter set to 0.2).

Exploring the Complementarity with Non-Physiological Methods

The next objective is to investigate the complementarity between EMULATE and various non-physiological methods such as CutMix, MixUp, Cutout, and Dropout. Therefore, in this approach, the physiological plausibility of the entire augmentation setup is 'sacrificed' in favor of enhancing the model's generalization ability further. For each dataset, the three models, and the four augmentation methods, each model's performance is compared with a regimen where it is trained using the corresponding augmentation method combined with the dynamics and then the dynamic high variants.

4.4. Model Training and Evaluation

4.4.1. Train/Test Split

To ensure consistency across experiments, we initially generate and store distinct training and test folds for each iteration. A three-fold stratified cross-validation approach is employed, which is a variant of the cross-validation train/test split method. This method accounts for label distribution to ensure similar label distributions between the training and test sets.

Stratification is applied at the level of anonymized patient identifiers to prevent overlap between the training and test sets, using the iterative-stratification library from the scikit-multilearn package [26]. During the process, patient IDs are divided into three folds. Subsequently, the corresponding recording time series and annotations are collected for each candidate patient ID. Finally, two folds of data are used for training, while the remaining fold is reserved for model testing.

4.4.2. Random Batch Sampling

To enhance the variability of the training dataset, a sampling heuristic similar to previous studies is employed. Rather than constructing a static training set using 10 consecutive segments, the different segments are randomly selected for each sample recording in real-time during model fitting.

This improves training regularization by diversifying the training set compared to consecutive sampling. For instance, from a recording of 50 segments, consecutive sampling yields 41 unique samples, while random sampling can generate over 10^{15} tuple samples, significantly increasing diversity. By incorporating more segments and dynamic random sampling, we aim to maximize dataset utilization and enhance model generalization.

4.4.3. Data Augmentation Hyperparameters

We explored various setups by varying the sampling law as well as the different angular values.

We experimented with angles in the range of 3 and 30 on the logarithmic scale. Additionally, we tested two sampling distributions, namely the normal and uniform distribution. We found that to increase variability, a uniform distribution is preferred. Additionally, we discovered that when sampling small angles (for example, from U[−5,5]), the regularization effect diminishes. Conversely, when allowing for larger angles (for example, sampling from U[−45,45]), it affects the model's performance.

As a result, we selected three different configurations for further experimentation: one variant for the static mode and two variants for the dynamic mode, namely "dynamic" and "dynamic high". Table 2 displays the different hyperparameters used for the three proposed configurations. In the static mode, only the initial angular value is sampled from a uniform distribution within [−10, 10]. For the first variant of the dynamic mode, both initial and peak angular rotations are sampled from a uniform distribution within [−15, 15]. In the second variant, a larger sampling interval of [−20, 20] is considered. For all three variants of dynamic modes, sampling periods were chosen using random values from a uniform distribution ranging between 4 and 40.

Table 2. An overview of the various parameters defining the configuration for each of the 5 augmentation strategies. Note that U(a,b) corresponds to the uniform distribution on the interval [a,b].

Parameter	Dynamic	Dynamic High	Static
Initial angular position	U(−15,15)	U(−20,20)	U(−10,10)
Maximum angular position	U(−15,15)	U(−20,20)	-
Period	U(4,40)	U(4,40)	-

4.4.4. Model and Training Hyperparameters

The training setup from a prior study [10] is reused to evaluate the proposed method. The setup is briefly outlined here. Additionally, the hyperparameters for the HTCE encoder and its lightweight variant are provided in Appendix A, in Tables A9 and Table A10, respectively. Refer to [10] for a comprehensive presentation of the model training set.

The different deep learning architectures are implemented using the TensorFlow package for model fitting on a single NVIDIA A100 80 GB GPU. Each model's hyperparameters are manually optimized, and training lasts for 100 epochs using the AdamW optimizer with a learning rate set to 1×10^{-4} for stability. The weight decay, set at 1×10^{-5}, aids in regularization [27]. Furthermore, we optimize the focal loss with a gamma of 5 and class balancing. The different settings for class balancing are also presented in the Appendix A, in Table A11, which lists the various training hyperparameters.

Finally, an early stopping technique is used to prevent the model from overfitting by monitoring the validation global F1 score and stopping the training after 10 consecutive epochs without improvement.

Model Evaluation

Medical datasets often exhibit high imbalances with pathologic populations being rarer compared to normal ones. As a result, precision, recall, sensitivity, and specificity are preferred over accuracy in evaluating the method performance for each class. Additionally, recall that the classification problem is a multi-annotation problem, where the model learns binary decisions for each of the eight classes. However, assessing 32 metrics simultaneously is not straightforward; therefore, the macro F1 score for each class allows for a global evaluation of screening performance across both positive and negative classes.

The model performance evaluation involves several metrics to assess the screening ability for both positive and negative cases across different pathologies. These metrics include per-class macro F1 scores, positive F1 scores, negative global F1 scores, and global F1 scores.

The positive F1 score evaluates the overall model performance in screening each pathology for each class by averaging the positive F1 scores for each class separately. Conversely, the negative global F1 score is computed as the weighted mean of micro F1 scores from normal subjects within each class.

The global F1 score, on the other hand, provides a comprehensive assessment by averaging the positive and negative F1 scores. When ranking model performances, priority is given to the global F1 score, followed by the positive F1 score, and finally the negative F1 score. Subsequently, to ensure robustness and fairness in comparison, each model is evaluated using a stratified three-fold cross-validation method. Then, all metric scores are collected during each fold, and the mean values across the three folds are reported. finally, to maintain consistency, all models are trained and evaluated using the same fold split.

5. Result

5.1. Comparison with Existing Literature

5.1.1. Comparing the Overall Model Performances

Figure 2 presents a comparison of the overall model performances in terms of the global F1 score. Additionally, in the Appendix A, in Tables A1 and A2, we present the global F1 scores for the three architectures and the applied data augmentation methods when trained on the saccade and vergence datasets, respectively.

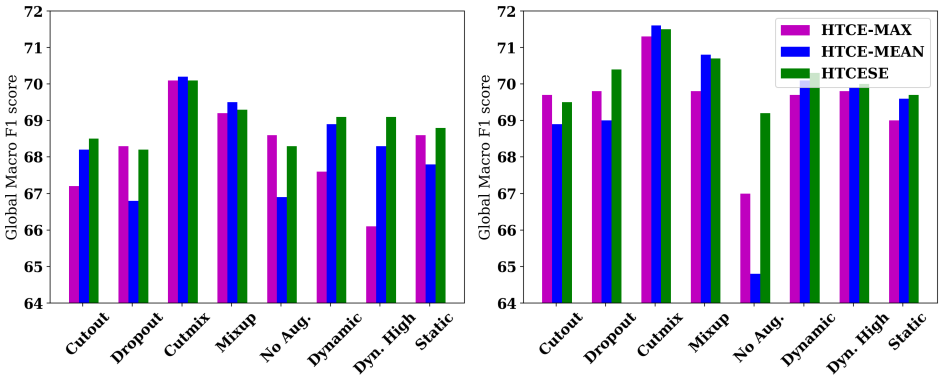

Figure 2. A comparison of the performance differences among different methods in terms of the global F1 scores for the three architectures, when trained on the saccade dataset (**right** subfigure) and the vergence dataset (**left** subfigure).

Overall, the hybrid architecture exhibits the highest performance in screening the positive samples, consistently achieving the highest positive F1 score across the two datasets and all the experiments, with a positive F1 score up to 53.9% and 51.2% on the saccade and vergence visual tasks, respectively. On the other hand, the best performance in terms of negative F1 score is achieved by HTCE-MEAN, with negative F1 scores up to 89.6% and 89.2% on the saccade and vergence visual tasks, respectively.

Saccade Visual Task

When considering the saccade visual task, the best overall performance (in terms of global F1 score and positive F1 score) is achieved when training using CutMix augmentation, using HTCE-MEAN (71.6%) and HTCSE (71.5%), respectively.

Furthermore, for the three architectures, EMULATE consistently improves performance relative to training with no augmentation, with improvements of up to 2.8, 5.3, and 0.9 points, respectively. Additionally, in comparison to other non-physiological augmentation methods, the proposed method performs competitively.

When considering only the physiological augmentation, the best performance is achieved when training the HTCSE using the dynamic variant. Additionally, the dynamic variant consistently achieves the highest score relative to the static variant. However, when inspecting the per-class macro F1 scores presented in Appendix A Table A4, the differences in performance are not consistent. For example, when considering the performance of the HTCSE on classes 0 and 1, which achieve the highest scores in these two classes, altering the head stability information (dynamic) decreases the performance relative to the static variant for classes 0 and 1, corresponding to dyslexia and reading disorders, with losses in the macro F1 score of 0.2 and 0.3 points, respectively.

Vergence Visual Task

With the vergence dataset, the best performance in terms of the global F1 score is achieved when training using CutMix, with relatively similar performances for the three models.

Additionally, EMULATE consistently improves global performance relative to training without augmentation, resulting in an improvement of the overall best score from 68.6% to

69.1%. HTCE-MEAN and HTCSE training show better performances when trained using the dynamic EMULATE variant compared to Cutout and dropout, improving the best global F1 score from 68.3% to 69.1%. Furthermore, compared to MixUp, the difference in overall best performance is 0.2 points.

When considering the overall performance of the physiological data augmentation methods, in terms of the global F1 score, the best performance is achieved when training the HTCSE using the dynamic variant (69.1%). Additionally, while training HTCE-MEAN and HTCSE using the dynamic variant achieves the best scores (68.9% and 69.1%). Similarly, in Appendix A, in Table A3, the per-class macro F1 score is presented for each model when trained on the vergence visual task.

5.2. *Extending the Baseline Methods with EMULATE*

Tables 3 and 4 showcase the different global F1 scores obtained by extending the various baseline methods using the two EMULATE dynamic variants. Additionally, in Figures 3 and 4, we present barplots comparing the different baseline performances when combined with the dynamic and dynamic high EMULATE settings, in terms of the global F1 score. Finally, detailed per-class F1 scores for the corresponding experiments are presented in Appendix A, in Tables A5–A8.

Table 3. A comparisonof the three global F1 scores across different architectures during training with baseline augmentation, both with and without the integration of the proposed methods (dynamic and dynamic high variants) on the saccade visual task. The best global F1 score for each method within the three setups is highlighted in bold.

Technique	Head	EMULATE Disabled			Dynamic			Dynamic High		
		Macro F1	Neg. F1	Pos. F1	Macro F1	Neg. F1	Pos. F1	Macro F1	Neg. F1	Pos. F1
HTCE-MAX	Cutout	69.7	89.1	50.2	**69.9**	89.0	50.8	69.7	89.1	50.4
	Dropout	69.8	87.9	51.6	**71.0**	89.1	52.8	70.5	88.8	52.3
	CutMix	**71.3**	89.4	53.3	70.5	89.0	52.0	70.1	89.2	51.1
	MixUp	69.8	89.4	50.3	**70.0**	89.3	50.7	69.8	89.3	50.2
HTCE-MEAN	Cutout	68.9	88.1	49.8	**70.1**	88.8	51.5	**70.1**	88.6	51.5
	Dropout	69.0	87.2	50.7	69.7	88.1	51.3	**70.2**	88.6	51.9
	CutMix	**71.6**	89.6	53.5	71.0	89.4	52.7	70.8	89.1	52.4
	MixUp	**70.8**	89.2	52.4	70.7	89.4	51.9	70.6	88.9	52.2
HTCSE	Cutout	69.5	88.1	50.9	70.6	88.9	52.2	**70.9**	89.1	52.7
	Dropout	70.4	88.4	52.5	70.8	88.5	53.1	**70.9**	88.5	53.3
	CutMix	**71.5**	89.1	53.9	70.7	89.0	52.4	70.6	88.9	52.3
	MixUp	70.7	88.6	52.8	**70.8**	89.2	52.4	70.6	88.9	52.2

On the vergence dataset, notable enhancements are observed across the three methods—Cutout, Dropout, and MixUp—excluding CutMix. Specifically, improvements of (2.1, 1.3, and 0.2), (1.4, 2.5, and 0.3), and (0.9, 1.3, and 0.4) points are observed for HTCE-MAX, HTCE-MEAN, and HTCSE, respectively.

Conversely, in the saccade visual task, when considering the performance of each model separately, Cutout and Dropout consistently improve performance across all three architectures.

Finally, when comparing the best performance gain of each augmentation, relatively to a training with EMULATE Disabled, across the three models, notable differences in performance emerge: 1.2, 0.6, and 0 for Cutout, Dropout, and MixUp, respectively.

Table 4. A comparison of the three global F1 scores across different architectures during training with baseline augmentations, both with and without the integration of the proposed methods (dynamic and dynamic high variants) on the vergence visual task. The best global F1 score for each method within the three setups is highlighted in bold.

Technique	Head	EMULATE Disabled			Dynamic			Dynamic High		
		Macro F1	Neg. F1	Pos. F1	Macro F1	Neg. F1	Pos. F1	Macro F1	Neg. F1	Pos. F1
HTCE-MAX	Cutout	67.2	86.9	47.5	**69.3**	89.0	49.6	69.2	88.6	49.7
	Dropout	68.3	88.0	48.6	**69.6**	88.6	50.6	**69.6**	88.6	50.6
	CutMix	**70.1**	89.1	51.0	69.8	89.3	50.3	69.7	89.3	50.1
	MixUp	69.2	89.5	48.9	**69.4**	89.1	49.6	69.2	89.3	49.1
HTCE-MEAN	Cutout	68.2	88.6	47.8	69.4	88.3	50.5	**69.6**	89.5	49.8
	Dropout	66.8	86.0	47.6	69.1	88.2	50.0	**69.3**	88.0	50.6
	CutMix	**70.2**	89.2	51.1	69.9	89.2	50.5	69.9	89.0	50.8
	MixUp	69.5	89.6	49.4	**69.8**	89.4	50.1	69.7	88.8	50.6
HTCSE	Cutout	68.5	88.2	48.8	69.3	88.2	50.3	**69.4**	88.5	50.3
	Dropout	68.2	87.4	49.1	69.3	88.0	50.6	**69.5**	88.0	51.1
	CutMix	**70.1**	88.9	51.2	69.6	88.7	50.6	69.4	88.6	50.2
	MixUp	69.3	88.3	50.4	**69.7**	88.5	50.9	69.5	88.6	50.4

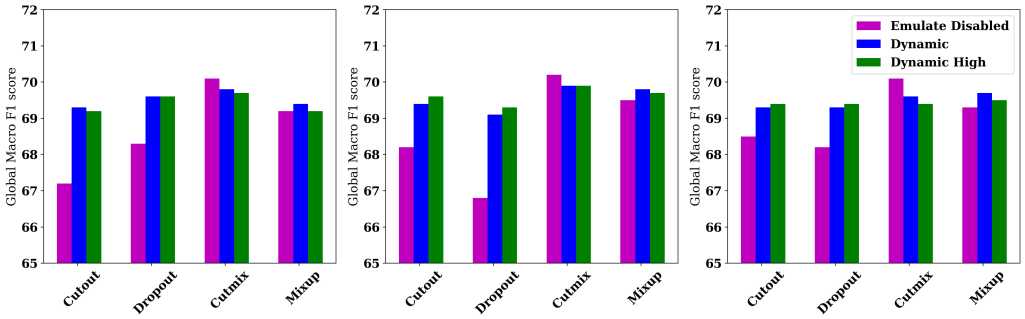

Figure 3. A barplot comparing the different baseline performances when combined with the dynamic and dynamic high EMULATE settings, and trained with the HTCE-MAX (**left** subfigure), the HTCE-MEAN (**middle** subfigure), and the HTCSE (**right** subfigure) on the vergence dataset.

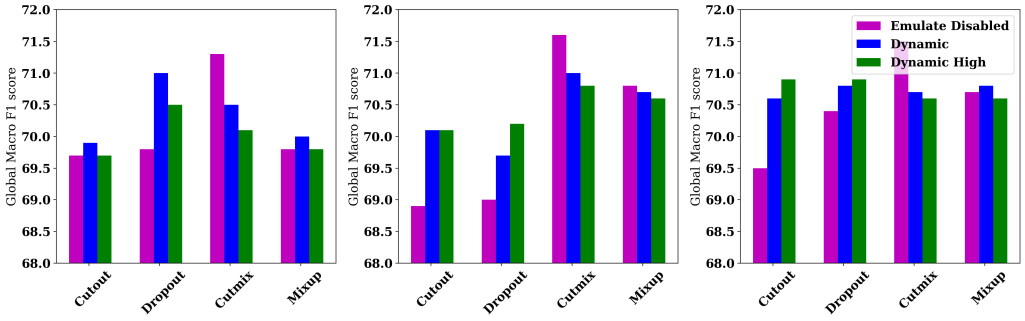

Figure 4. A barplot comparing the different baseline performances when combined with the dynamic and dynamic high EMULATE settings, and trained with the HTCE-MAX (**left** subfigure), the HTCE-MEAN (**middle** subfigure), and the HTCSE (**right** subfigure) on the saccade dataset.

6. Discussion

6.1. Major Finding

In this study, EMULATE, a novel data augmentation library tailored to time-series deep learning projects, is introduced. A comprehensive exploration of various settings was conducted, employing both a proposed CNN-based architecture and a hybrid-based architecture.

The evaluation encompasses two datasets: the saccade and the vergence visual tasks. The proposed method enhances generalization performance compared to training without augmentation and several baseline methods, such as Dropout and Cutout, and achieves competitive performance with MixUp. Moreover, although some physiological plausibility is sacrificed in the augmentation setup, incorporating EMULATE—except for CutMix—improves the overall performance across the tested regularization and augmentation methods. From a physiological perspective, it is important to consider that eye movements are not isolated from head movements and positions, as commands for eye movements are known to influence neck muscles, even when the head is artificially stabilized [28].

6.2. Analysis of the Degree of Freedom

The proposed data augmentation method significantly increases the dataset size. For instance, the dynamic mode comprises 10 different parameters, while the static mode is characterized by four parameters. These parameters correspond to the interpolation parameters and a tuple of values representing the initial angular position within each axis. Table 5 lists the specific parameters for each of the two configurations. Note that all the parameters sampled from the continuous interval increase the density of the output space.

Table 5. Presentation of the sampling parameters.

Parameters	**Configuration**	
	Dynamic	**Static**
Coordinates interpolation parameter (1 parameter)	X	X
Initial angular position (3 parameters)	X	X
Maximum angular rotation amplitude (3 parameters)	X	
Sinusoidal period (3 parameters)	X	

6.3. Physiological Data Augmentation

By using physiological data augmentation in deep learning, one can benefit from several advantages and implications. Realistic data augmentation enhances the discrimination performance by increasing the size of the dataset. On the other hand, non-realistic methods have different effects on accuracy, primarily through regularization techniques. For example, Zeshan et al. [6] compared different data augmentation techniques for the medical imaging classification task and found a strong relationship between the chosen augmentation method and the discrimination performance. In addition, the results highlight that realistic methods such as scaling, shearing, and rotating resulted in an accuracy range of 87.4% to 88.0%, while non-realistic methods like noise and power had validation accuracies of 66.0% and 73.7%, respectively. This can be attributed to realistic augmentations effectively increasing the sample size, whereas non-realistic methods serve more as regularization techniques, rather than expanding the information in the dataset itself.

6.4. Generative Method Limitation

A generative model for data augmentation does not involve sampling real data or introducing new information. Moreover, the quality of the generated data relies on the generative model's ability to capture complex or rare patterns. The task becomes more challenging when dealing with high-resolution time series data and a limited dataset size. Such constraints are especially pronounced when using a limited dataset.

6.5. On the Importance of the Transformation-Based Method

Deep learning-based data augmentation has shown promise, but it can struggle to capture rare patterns in small training datasets and introduce noise when sampling at different resolutions. Various studies have attempted to address these limitations to increase

the image resolution [29–33] or enhance the quality [34]. However, this task remains challenging, particularly with a limited training set and high-resolution samples. On the other hand, transformation-based data augmentation generates new samples through realistic transformations. For instance, in the present study, head movements are simulated to generate new rotated patterns that cannot be obtained through linear interpolation of existing samples.

Furthermore, when transformation-based methods are applied to generative-based data augmentation techniques, they can improve generalization ability, leading to better variety. Simply put, transformation-based methods can enhance generalization ability on their own or be combined with cutting-edge generative data augmentation methods to increase variability and improve augmentation. This makes them even more crucial in practice.

7. Limitations and Future Directions

7.1. Dynamic Mode Limitations

The dynamic method allows for higher augmentation capabilities. However, unlike the static mode, which makes no assumptions when performing data augmentation, the dynamic mode tends to exhibit a bias toward the sinusoidal motion model. By authorizing a near-manifold sampling, the velocity of the head is modeled using parameterization to approximate head movement during testing. This is achieved by employing a parameterized sinusoidal function and selecting the frequency range accordingly. Consequently, in contrast to the static mode, which samples from the true data manifold, the dynamic mode may introduce artifacts due to the artificial sinusoidal motion model.

Furthermore, for certain pathologies, dynamic head movement can be a pertinent criterion for screening the corresponding pathology. Additionally, small head movements can be discriminative for the same use case. For such pathologies, the static mode is recommended, as this method does not alter the real-time head instability parameters.

7.2. Computational Efficiency

While EMULATE shows promising results, it introduces significant computational costs when implemented using native Python. However, when incorporating the various computations into the TensorFlow computation graph, the introduced computational cost is relatively low compared to the initial computation cost introduced by the deep learning model. Additional techniques such as pre-fetching and preprocessing each batch in parallel further reduce the introduced computation cost.

7.3. Sensitivity to the Radius Coordinates

In order to convert the different coordinates from the spherical system to the Cartesian system, the radius distance is required, corresponding to the eye fixation coordinate within the optical axis. In the current study, this distance is approximate with the coordinate of the stimulus LED, which implies perfect accuracy in convergence and divergence, achieving exact precision.

7.4. Future Direction

At this stage, the proposed method shows promising performance when compared to other methods. One important direction involves exploring the performance of the proposed method on different standard datasets as well as other model architectures. This exploration aims to assess the relevance of EMULATE in various settings, including different datasets, models, and learning tasks.

Additionally, a noteworthy future direction involves investigating why, unlike other augmentation methods, incorporating EMULATE diminishes the generalization ability of CutMix. and push the exploration further by exploring the performance of the proposed method on different standard datasets as well as other model architectures, in order to

assess the relevance of EMULATE under different settings, including different datasets, models, as well as learning tasks.

A notable future direction involves investigating why, unlike other augmentation methods, incorporating EMULATE diminishes the generalization ability of CutMix. Further exploration could involve assessing the performance of the proposed method on various standard datasets and model architectures to evaluate the relevance of EMULATE under different settings, encompassing different datasets, models, and learning tasks.

Another area of improvement involves replacing the naive sinusoidal model, with a more complex model, thus enabling a better approximation of head motion to enhance the accuracy of generated data when utilizing the dynamic mode.

Finally, the number of head and eye coordinates taken from [21] is relatively small. Therefore, the next step would be to enhance the richness of the used coordinates by extending the database of the eye and head coordinates. A future direction will be to augment the table with the collected head and eye coordinates.

8. Conclusions

In conclusion, the challenges associated with limited annotated medical data necessitate innovative solutions for effective data augmentation. Traditional methods, such as mixing-based algorithms, may not be suitable due to their potential to introduce artifacts and alter pathological features. Generative augmentation methods tailored to the characteristics of pathology images are often preferred, yet they may be less effective at capturing complex or rare patterns compared to transformation-based techniques.

In response to these challenges, we propose a novel physiologically based head data augmentation library (EMULATE) that emulates natural head movements during data collection, contributing to enhanced sample diversity and authenticity. Our library is the first of its kind to incorporate physiological aspects, generating transformed eye movement data efficiently. Additionally, we perform a first exploration of different architectures and datasets, demonstrating the effectiveness of EMULATE in regularizing the training process and improving the generalization ability of the proposed hybrid architecture, outperforming a CNN-based approach in eye movement classification.

9. Patents

Zoï Kapoula has applied for patents for the technology used to conduct this experiment: REMOBI table (patent US8851669, WO2011073288); AiDEAL analysis software (EP20306166.8, 7 October 2020; EP20306164.3, 7 October 2020—Europe). Patent application pending EP22305903.1.

Author Contributions: Supervision, Z.K.;methodology, A.E.E.H.; software, A.E.E.H.; validation, A.E.E.H. and ZK; formal analysis, A.E.E.H.; investigation, A.E.E.H.; resources, ZK.; data curation, A.E.E.H.; Conceptualization, A.E.E.H.; writing—original draft, A.E.E.H.; writing—review and editing, Z.K.; visualization, A.E.E.H.; project administration, Z.K.; funding acquisition, Z.K. All authors have read and agreed to the published version of the manuscript.

Funding: A.E.E.H. is funded by Orasis-Ear, ANRT, and CIFRE.

Informed Consent Statement: This meta-analysis drew upon data sourced from Orasis Ear, in collaboration with clinical centers employing Remobi and Aideal technology. Participating centers agreed to store their data anonymously for further analysis.

Data Availability Statement: The datasets generated during and/or analyzed during the current study are not publicly available. This meta-analysis drew upon data sourced from Orasis Ear, in collaboration with clinical centers employing REMOBI and AiDEAL technology. Participating centers agreed to store their data anonymously for further analysis. However, upon reasonable request, they are available from the corresponding author.

Acknowledgments: This work was granted access to the HPC resources of IDRIS under the allocation 2024-AD011014231 made by GENCI.

Conflicts of Interest: Zoï Kapoula is the founder of Orasis-EAR.

Appendix A

Table A1. The presentation of the global F1 (Macro F1), global positive F1 (Pos. F1), and global negative F1 (Neg. F1) scores when trained on the vergence visual task. For each model, the best global F1 score is highlighted in bold.

	HTCE-MAX			**HTCE-MEAN**			**HTCSE**		
	Macro F1	**Neg. F1**	**Pos. F1**	**Macro F1**	**Neg. F1**	**Pos. F1**	**Macro F1**	**Neg. F1**	**Pos. F1**
Cutout	67.2	86.9	47.5	68.2	88.6	47.8	68.5	88.2	48.8
Dropout	68.3	88.0	48.6	66.8	86.0	47.6	68.2	87.4	49.1
CutMix	**70.1**	89.1	51.0	**70.2**	89.2	51.1	**70.1**	88.9	51.2
MixUp	69.2	89.5	48.9	69.5	89.6	49.4	69.3	88.3	50.4
No Aug.	68.6	89.1	48.0	66.9	88.2	45.7	68.3	88.0	48.5
Dynamic	67.6	87.0	48.1	**68.9**	88.6	49.3	**69.1**	88.4	49.8
Dynamic High	66.1	86.4	45.9	68.3	87.5	49.1	**69.1**	88.3	49.9
Static	**68.6**	88.4	48.9	67.8	86.9	48.6	68.8	88.3	49.3

Table A2. Presentation of the global F1 (Macro F1), global positive F1 (Pos. F1), and global negative F1 (Neg. F1) scores when trained on the saccade visual task. For each model, the best global F1 score is highlighted in bold.

	HTCE-MAX			**HTCE-MEAN**			**HTCSE**		
	Macro F1	**Neg. F1**	**Pos. F1**	**Macro F1**	**Neg. F1**	**Pos. F1**	**Macro F1**	**Neg. F1**	**Pos. F1**
Cutout	69.7	89.1	50.2	68.9	88.1	49.8	69.5	88.1	50.9
Dropout	69.8	87.9	51.6	69.0	87.2	50.7	70.4	88.4	52.5
CutMix	**71.3**	89.4	53.3	**71.6**	89.6	53.5	**71.5**	89.1	53.9
MixUp	69.8	89.4	50.3	70.8	89.2	52.4	70.7	88.6	52.8
No Aug.	67.0	86.8	47.3	64.8	86.0	43.6	69.2	88.0	50.4
Dynamic	69.7	88.9	50.6	**70.1**	88.4	51.8	**70.3**	88.6	52.0
Dynamic High	**69.8**	88.6	51.0	69.9	89.2	50.5	70.0	88.7	51.3
Static	69.0	88.6	49.5	69.6	88.4	50.8	69.7	88.2	51.3

Table A3. Peer class macro F1 scores for each augmentation and regularization method when separately training the three different architectures on the vergence dataset.

Model	**Augmentation Method**	**Class 0**	**Class 1**	**Class 2**	**Class 3**	**Class 4**	**Class 5**	**Class 6**	**Class 7**
HTCE-MAX	CutMix	67.4	71.3	63.9	63.5	78.0	70.2	74.0	73.1
	Cutout	64.5	68.2	63.3	62.2	73.8	66.0	70.8	69.9
	Dropout	65.9	69.4	62.4	62.7	75.7	67.5	71.7	72.3
	MixUp	66.3	70.4	62.9	63.2	76.9	68.7	73.7	72.4
	No Aug.	65.7	70.1	62.1	62.8	76.7	69.0	73.1	71.9
	Dynamic	63.4	69.9	62.0	60.1	74.9	67.7	71.9	71.6
	Dynamic High	64.3	68.0	62.5	62.5	72.4	65.7	70.2	64.3
	Static	65.8	69.6	63.4	61.2	76.4	68.9	73.2	71.6

Table A3. *Cont.*

Model	Augmentation Method	Class 0	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6	Class 7
HTCE-MEAN	CutMix	68.1	71.7	63.9	63.9	78.2	70.4	73.1	73.0
	Cutout	65.7	69.5	63.7	63.7	75.7	68.5	71.8	68.0
	Dropout	63.9	67.7	62.1	61.3	73.0	67.3	69.8	70.2
	MixUp	66.8	70.9	62.7	63.3	77.1	70.1	73.6	72.5
	No Aug.	64.9	68.4	63.3	60.6	72.9	68.3	71.7	66.4
	Dynamic	66.8	70.7	63.5	62.7	76.7	69.1	71.8	71.1
	Dynamic High	66.1	70.5	62.2	61.9	74.4	68.6	71.7	71.8
	Static	65.5	68.0	63.8	63.2	75.1	67.7	71.8	68.3
HTCSE	CutMix	67.6	71.5	64.6	63.4	77.2	69.5	74.1	73.7
	Cutout	66.5	69.7	62.7	62.9	75.2	68.4	72.0	71.7
	Dropout	66.2	69.3	63.3	63.1	74.5	67.5	71.0	71.8
	MixUp	66.8	70.1	63.4	64.0	76.6	68.3	73.1	73.3
	No Aug.	65.9	68.8	62.9	62.5	75.5	67.7	72.5	71.2
	Dynamic	67.3	69.9	63.7	62.8	76.1	68.1	72.3	73.7
	Dynamic High	67.0	69.9	63.7	62.5	76.4	68.0	73.4	72.9
	Static	67.5	70.2	63.0	62.6	76.1	68.2	72.0	71.7

Table A4. Peer class macro F1 scores for each augmentation and regularization method when separately training the three different architectures on the saccade dataset.

Model	Augmentation Method	Class 0	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6	Class 7
HTCE-MAX	CutMix	70.5	74.4	65.5	66.7	79.0	72.8	73.0	69.8
	Cutout	68.7	71.8	64.5	64.0	78.5	70.8	72.7	67.6
	Dropout	68.1	71.6	64.5	66.1	77.1	71.4	72.3	68.0
	MixUp	68.9	73.0	63.1	65.5	78.6	70.8	72.2	67.7
	No Aug.	67.3	70.6	63.2	61.4	76.6	65.6	66.9	65.5
	Dynamic	68.5	72.7	64.3	64.9	78.5	70.0	72.3	67.8
	Dynamic High	68.1	71.7	64.5	64.9	78.6	71.4	72.5	68.2
	Static	68.4	71.7	62.8	64.5	78.6	70.6	72.0	64.7
HTCE-MEAN	CutMix	70.5	74.4	65.5	67.2	79.8	73.4	74.1	68.9
	Cutout	69.0	72.2	65.6	64.8	77.7	71.3	68.4	63.5
	Dropout	69.4	72.4	66.1	65.7	74.0	70.1	68.4	66.8
	MixUp	69.7	73.1	66.2	65.5	79.5	72.1	73.8	67.5
	No Aug.	64.3	66.8	63.2	61.0	73.8	66.3	67.8	56.2
	Dynamic	69.7	72.3	66.0	65.4	78.9	71.4	72.0	66.1
	Dynamic High	68.9	72.3	64.4	65.0	79.0	72.0	71.1	67.2
	Static	69.1	71.8	64.9	65.8	77.4	71.4	72.0	65.5
HTCSE	CutMix	70.3	73.9	65.5	67.1	79.8	72.9	73.7	70.2
	Cutout	67.9	71.1	64.8	65.1	78.0	69.8	72.3	68.3
	Dropout	69.3	73.1	65.8	66.5	77.0	71.8	72.1	69.0
	MixUp	69.7	72.1	65.7	66.9	78.5	72.3	73.0	68.3
	No Aug.	68.2	71.6	64.4	64.5	76.9	69.6	71.6	67.8
	Dynamic	68.8	73.0	63.4	66.2	78.6	71.8	73.2	68.3
	Dynamic High	69.2	72.6	64.0	65.3	78.4	71.5	72.4	67.8
	Static	68.4	71.9	64.6	65.5	77.2	70.1	72.3	68.8

Table A5. Peer class macro F1 scores for each augmentation and regularization baseline method when training in combination with the dynamic variant of EMULATE, the three different architectures on the vergence dataset.

Model	Augmentation Method	Class 0	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6	Class 7
HTCE-MAX	CutMix	67.7	71.9	63.6	62.5	77.6	69.1	74.0	72.8
	Cutout	66.6	69.7	64.3	63.3	76.5	69.2	73.0	72.9
	Dropout	67.4	70.8	64.0	62.8	76.7	69.7	72.8	73.5
	MixUp	67.3	70.9	62.6	62.5	76.3	68.7	73.7	73.8
HTCE-MEAN	CutMix	68.0	71.1	63.0	62.9	76.9	70.1	73.1	74.6
	Cutout	67.5	70.6	64.6	62.4	77.2	68.7	72.8	72.4
	Dropout	67.3	70.8	63.9	63.5	75.6	69.7	70.2	73.1
	MixUp	67.3	71.2	63.1	63.7	77.1	69.4	73.6	73.8
HTCSE	CutMix	67.5	71.0	62.8	63.4	77.0	68.9	74.1	73.3
	Cutout	67.5	70.3	63.3	63.2	75.9	68.6	72.9	73.1
	Dropout	68.0	70.8	63.5	63.2	75.6	68.5	72.9	72.7
	MixUp	68.2	70.8	64.3	64.3	76.4	67.5	73.5	73.5

Table A6. Peer class macro F1 scores for each augmentation and regularization baseline method when training in combination with the dynamic High variant of EMULATE, the three different architectures on the vergence dataset.

Model	Augmentation Method	Class 0	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6	Class 7
HTCE-MAX	CutMix	67.7	70.7	64.4	62.2	77.2	69.6	73.3	73.3
	Cutout	66.9	70.4	63.1	63.0	76.4	68.3	73.4	72.8
	Dropout	67.1	71.4	63.3	64.5	75.9	69.2	72.8	73.8
	MixUp	67.4	70.5	62.9	62.1	76.9	68.5	72.9	73.5
HTCE-MEAN	CutMix	68.0	71.4	64.0	63.3	77.2	69.7	72.9	73.6
	Cutout	67.2	71.3	63.5	62.4	77.4	69.2	73.6	73.7
	Dropout	67.3	71.1	64.6	63.8	75.3	69.6	70.7	72.9
	MixUp	68.1	71.3	64.1	62.4	77.3	69.5	71.8	74.1
HTCSE	CutMix	66.8	70.8	63.4	62.9	76.8	68.1	73.4	73.9
	Cutout	67.3	69.9	63.6	63.5	76.5	67.9	74.1	73.3
	Dropout	68.6	71.2	64.3	64.1	74.9	68.7	73.2	72.3
	MixUp	67.5	70.3	64.0	63.8	76.8	67.7	73.3	73.8

Table A7. Peer class macro F1 scores for each augmentation and regularization baseline method when training in combination with the dynamic variant of EMULATE, the three different architectures on the saccade dataset.

Model	Augmentation Method	Class 0	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6	Class 7
HTCE-MAX	CutMix	69.4	73.4	65.0	65.6	79.6	71.6	72.8	67.7
	Cutout	68.9	72.6	63.5	64.0	79.5	72.0	72.3	67.6
	Dropout	69.5	73.0	64.9	65.4	79.4	73.3	72.2	70.9
	MixUp	68.1	72.7	63.5	64.3	79.8	71.1	72.6	69.0
HTCE-MEAN	CutMix	70.4	73.5	65.0	66.4	79.6	73.0	73.1	68.4
	Cutout	68.9	72.1	64.4	65.4	79.3	73.2	71.3	67.6
	Dropout	69.6	72.6	64.9	64.4	77.5	71.8	70.1	67.6
	MixUp	69.5	73.0	65.0	63.9	79.5	72.4	73.4	69.5
HTCSE	CutMix	69.6	73.1	64.1	66.3	79.6	72.4	73.1	68.7
	Cutout	69.3	73.1	64.9	66.2	78.4	71.8	73.4	68.5
	Dropout	69.4	72.3	65.4	67.1	77.8	73.7	74.0	67.8
	MixUp	69.5	73.1	64.2	67.0	79.6	72.3	73.5	68.0

Table A8. Peer class macro F1 scores for each augmentation and regularization baseline method when training in combination with the dynamic High variant of EMULATE, the three different architectures on the saccade dataset.

Model	Augmentation Method	Class 0	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6	Class 7
HTCE-MAX	CutMix	69.0	72.9	63.8	65.5	79.2	71.7	72.2	67.7
	Cutout	68.2	72.2	63.5	64.1	79.1	71.0	72.3	68.6
	Dropout	68.7	72.5	64.9	65.1	79.2	73.2	72.2	69.8
	MixUp	67.7	72.6	63.1	64.2	79.6	71.3	72.4	68.4
HTCE-MEAN	CutMix	70.0	73.4	64.6	65.4	79.3	73.4	73.3	68.0
	Cutout	69.5	72.4	64.2	65.6	79.1	73.0	70.9	66.7
	Dropout	69.7	73.5	65.5	65.6	77.7	72.0	71.6	67.2
	MixUp	69.1	73.3	64.9	65.9	78.7	72.1	73.0	68.6
HTCSE	CutMix	69.8	73.1	63.7	65.3	79.3	73.1	73.1	68.1
	Cutout	69.9	73.0	64.7	66.3	78.9	72.6	74.2	68.8
	Dropout	70.1	73.2	64.9	67.1	77.9	73.6	73.2	68.5
	MixUp	69.1	72.8	63.7	66.2	79.2	72.6	73.1	68.9

Table A9. HTCE feature extractor hyperparameters.

Stage	Filter Size	Pooling	Kernel Size	Activation
1	128-128-128	0-0-2	5-5-5	relu
2	128-128-128	0-0-2	5-5-5	relu
3	256-256-256	0-2-2	5-5-5	relu
4	512-512-512	0-2-2	3-3-3	relu

Table A10. Lightweight HTCE hyperparameters.

Stage	Filter Size	Pooling	Kernel Size	Activation
1	64-64	0-2	5-5	relu
2	128-128	0-2	5-5	relu
3	256-256	2-2	5-5	relu
4	512-512	2-2	3-3	relu

Table A11. Model Training hyperparameters.

	Value
Optimizer	
Name	AdamW
Learning rate	0.0001
Beta1	0.9
Beta2	0.999
Weight decay	0.00001
Loss	
name	Focal loss
Alpha class 0	0.73
Alpha class 1	0.61
Alpha class 2	0.90
Alpha class 3	0.88
Alpha class 4	0.67
Alpha class 5	0.83
Alpha class 6	0.81
Alpha class 7	0.32
Gamma	5

Table A11. *Cont.*

	Value
Training	
Batch size (HTCE-MAX)	128
Batch size (HTCE-MEAN)	128
Batch size (Baselines)	128
Batch size (HTCSE)	256
Epochs	100
Number of folds	3

References

1. Tagnamas, J.; Ramadan, H.; Yahyaouy, A.; Tairi, H. Multi-task approach based on combined CNN-transformer for efficient segmentation and classification of breast tumors in ultrasound images. *Vis. Comput. Ind. Biomed. Art* **2024**, *7*, 2.
2. Pan, X.; Xiong, J. DCTNet: A Hybrid Model of CNN and Dilated Contextual Transformer for Medical Image Segmentation. In Proceedings of the 2023 IEEE 6th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), Chongqing, China, 24–26 February 2023; IEEE: New York, NY, USA, 2023; Volume 6, pp. 1316–1320.
3. Lin, X.; Yan, Z.; Deng, X.; Zheng, C.; Yu, L. ConvFormer: Plug-and-Play CNN-Style Transformers for Improving Medical Image Segmentation. In Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention, Vancouver, BC, Canada, 8–12 October 2023; Springer: Berlin/Heidelberg, Germany, 2023; pp. 642–651.
4. Abibullaev, B.; Keutayeva, A.; Zollanvari, A. Deep Learning in EEG-Based BCIs: A Comprehensive Review of Transformer Models, Advantages, Challenges, and Applications. *IEEE Access* **2023**, *11*, 127271–127301. [CrossRef]
5. Cubuk, E.D.; Zoph, B.; Shlens, J.; Le, Q.V. Randaugment: Practical automated data augmentation with a reduced search space. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, Seattle, WA, USA, 14–19 June 2020; pp. 702–703.
6. Fons, E.; Dawson, P.; Zeng, X.j.; Keane, J.; Iosifidis, A. Adaptive weighting scheme for automatic time-series data augmentation. *arXiv* **2021**, arXiv:2102.08310.
7. Yun, S.; Han, D.; Oh, S.J.; Chun, S.; Choe, J.; Yoo, Y. Cutmix: Regularization strategy to train strong classifiers with localizable features. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 6023–6032.
8. Zhang, H.; Cisse, M.; Dauphin, Y.N.; Lopez-Paz, D. mixup: Beyond empirical risk minimization. *arXiv* **2017**, arXiv:1710.09412.
9. Alex, A.; Wang, L.; Gastaldo, P.; Cavallaro, A. Mixup augmentation for generalizable speech separation. In Proceedings of the 2021 IEEE 23rd International Workshop on Multimedia Signal Processing (MMSP), Tampere, Finland, 6–8 October 2021; IEEE: New York, NY, USA, 2021; pp. 1–6.
10. El Hmimdi, A.E.; Themis Palpanas, Z.K. Efficient Diagnostic Classification of Diverse Pathologies through Contextual Eye Movement Data Analysis with a Novel Hybrid Architecture. *Sci. Rep.*
11. Zemblys, R.; Niehorster, D.C.; Holmqvist, K. gazeNet: End-to-end eye-movement event detection with deep neural networks. *Behav. Res. Methods* **2019**, *51*, 840–864. [CrossRef] [PubMed]
12. Elbattah, M.; Loughnane, C.; Guérin, J.L.; Carette, R.; Cilia, F.; Dequen, G. Variational autoencoder for image-based augmentation of eye-tracking data. *J. Imaging* **2021**, *7*, 83. [CrossRef] [PubMed]
13. Fuhl, W.; Rong, Y.; Kasneci, E. Fully convolutional neural networks for raw eye tracking data segmentation, generation, and reconstruction. In Proceedings of the 2020 25th International Conference on Pattern Recognition (ICPR), Milan, Italy, 10–15 January 2021 ; IEEE: New York, NY, USA, 2021; pp. 142–149.
14. Luo, Y.; Zhu, L.Z.; Wan, Z.Y.; Lu, B.L. Data augmentation for enhancing EEG-based emotion recognition with deep generative models. *J. Neural Eng.* **2020**, *17*, 056021. [CrossRef] [PubMed]
15. Özdenizci, O.; Erdoğmuş, D. On the use of generative deep neural networks to synthesize artificial multichannel EEG signals. In Proceedings of the 2021 10th International IEEE/EMBS Conference on Neural Engineering (NER), Virtual, 4–6 May 2021; IEEE: New York, NY, USA, 2021; pp. 427–430.
16. Luo, Y.; Zhu, L.Z.; Lu, B.L. A GAN-based data augmentation method for multimodal emotion recognition. In Proceedings of the Advances in Neural Networks—ISNN 2019: 16th International Symposium on Neural Networks, ISNN 2019, Moscow, Russia, 10–12 July 2019; Proceedings, Part I 16; Springer: Berlin/Heidelberg, Germany, 2019; pp. 141–150.
17. Cubuk, E.D.; Zoph, B.; Mane, D.; Vasudevan, V.; Le, Q.V. Autoaugment: Learning augmentation strategies from data. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Beach, CA, USA, 15–20 June 2019; pp. 113–123.
18. DeVries, T.; Taylor, G.W. Improved regularization of convolutional neural networks with cutout. *arXiv* **2017**, arXiv:1708.04552.
19. El Hmimdi, A.E.; Kapoula, Z.; Sainte Fare Garnot, V. Deep Learning-Based Detection of Learning Disorders on a Large Scale Dataset of Eye Movement Records. *BioMedInformatics* **2024**, *4*, 519–541. [CrossRef]

20. Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv* **2020**, arXiv:2010.11929.
21. Singh, P.; Thoke, A.; Verma, K. A Novel Approach to Face Detection Algorithm. *Int. J. Comput. Appl.* **2011**, *975*, 8887. [CrossRef]
22. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
23. Cutmix Algorithm. Available online: https://keras.io/api/keras_cv/layers/augmentation/cut_mix (accessed on 2 February 2024).
24. Cutout Algorithm. Available online: https://keras.io/api/keras_cv/layers/augmentation/random_cutout (accessed on 2 February 2024).
25. Mixup Algorithm. Available online: https://keras.io/api/keras_cv/layers/augmentation/mix_up/ (accessed on 2 February 2024).
26. Iterative Stratification. Available online: https://scikit.ml/api/skmultilearn.model_selection.iterative_stratification.html (accessed on 2 February 2024).
27. Loshchilov, I.; Hutter, F. Decoupled weight decay regularization. *arXiv* **2017**, arXiv:1711.05101.
28. André-Deshays, C.; Berthoz, A.; Revel, M. Eye-head coupling in humans: I. Simultaneous recording of isolated motor units in dorsal neck muscles and horizontal eye movements. *Exp. Brain Res.* **1988**, *69*, 399–406. [CrossRef] [PubMed]
29. Baur, C.; Albarqouni, S.; Navab, N. MelanoGANs: High resolution skin lesion synthesis with GANs. *arXiv* **2018**, arXiv:1804.04338.
30. Radford, A.; Metz, L.; Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv* **2015**, arXiv:1511.06434.
31. Karras, T.; Aila, T.; Laine, S.; Lehtinen, J. Progressive growing of gans for improved quality, stability, and variation. *arXiv* **2017**, arXiv:1710.10196.
32. Hayat, K. Super-resolution via deep learning. *arXiv* **2017**, arXiv:1706.09077.
33. Dong, C.; Loy, C.C.; He, K.; Tang, X. Learning a deep convolutional network for image super-resolution. In Proceedings of the Computer Vision—ECCV 2014: 13th European Conference, Zurich, Switzerland, 6–12 September 2014; Proceedings, Part IV 13; Springer: Berlin/Heidelberg, Germany, 2014; pp. 184–199.
34. Zhu, J.Y.; Park, T.; Isola, P.; Efros, A.A. Unpaired image-to-image translation using cycle-consistent adversarial networks. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2223–2232.