Riemannian Geometry-Preserving Variational Autoencoder for MI-BCI Data Augmentation

Bachelor's Project Thesis

Victoria Polaka, s4641930, a.v.p.polaka@student.rug.nl,
Supervisors: Ivo De Jong MSc & Dr Andreea Sburlea

Abstract: This paper addresses the challenge of generating synthetic electroencephalogram (EEG) covariance matrices for Motor Imagery Brain-Computer Interface data augmentation. *Objective.* To develop a generative model capable of producing high-fidelity synthetic covariance matrices for BCI data augmentation, accounting for the crucial constraint that these matrices are Symmetric Positive-Definite and reside on a non-Euclidean, Riemannian manifold. *Approach.* A novel Riemannian geometry-preserving Variational Autoencoder architecture is proposed. This model uniquely integrates geometric mappings and employs a composite loss function that combines Riemannian distance for manifold fidelity with objectives promoting Euclidean tangent space reconstruction accuracy and generative diversity. *Results.* The model successfully generates valid and representative EEG covariance matrices. The utility of the synthetic data was evaluated in a cross-subject, Leave-One-Subject-Out Cross-Validation classification setting and found to be highly classifier-dependent. While the augmentation significantly hindered the performance of a Support Vector Classifier, it maintained performance using Minimum Distance to Mean classifier and even provided a statistically significant improvement for the geometry-aware K-Nearest Neighbors classifier, increasing its balanced accuracy by up to 3.49%. *Contribution.* This work validates a new architecture for generating Motor Imagery EEG covariance matrices and concludes that its effectiveness as an augmentation tool is directly linked to the algorithm of the classifier it is paired with.

1 Introduction

A Brain-Computer Interface (BCI) is a system that measures bio-signals, such as electroencephalogram (EEG), and utilizes specialized algorithms to interpret specific aspects of a user's cognitive state (Bonci et al., 2021). In active systems such as Motor Imagery Brain-Computer Interfaces (MI-BCI), users voluntarily modulate their brain activity to generate specific commands for controlling external devices or environments (Bonci et al., 2021; Prapas et al., 2022; Cariello et al., 2023). The core principle behind MI-BCIs involves the user imagining a motor action (e.g left-hand movement), without any physical execution. This mental task causes measurable changes in the sensorimotor rhythms of the brain's cortex—a phenomenon known as Event-Related Desynchronization/Synchronization—which can be captured using EEG and thus translated into commands (Bonci et al., 2021). While traditionally, a MI-BCI pipeline often consists of preprocessing (i.e., bandpass filter), spatial filtering (e.g., Common Spatial Pattern filters), feature extraction (e.g band power) and specialized classification models such as Linear Discriminant Analysis (LDA) (Yger et al., 2017), recent approaches employing Riemannian geometry have demonstrated superior performance over classical approaches, scoring the highest in multiple brain signal classification competitions (Congedo et al., 2017; Yger et al., 2017). These state-of-the-art pipelines often replace the traditional feature extraction steps and it's corresponding models in favor of EEG covariance matrices paired with Riemannian-based classification models such as the Minimum Distance to Mean (MDM) classifier and Support Vector Classifier (SVC) (Chevallier et al., 2024; Yger et al., 2017). Even deep learning models, despite their success in other domains, have generally not exceeded the performance of existing Riemannian pipelines, which may be explained by the limited availability of subject-level data within the BCI community (Chevallier et al., 2024). Limited data along with long calibration time between subjects prove to be significant hurdles in the advancement of mainstream BCI applications (Chevallier et al., 2024; Yger et al., 2017).

A promising path to tackling these issues may lie in data augmentation techniques—artificially expanding the training set by generating new, synthetic samples from an original dataset—which has already been long practiced in the field of machine learning to undertake data scarcity (Chevallier et al., 2024; Rommel et al., 2022). Therefore, this paper focuses on developing and evaluating a novel method for data augmentation specifically tailored to the unique geometric properties of EEG covariance matrices.

Previous work has explored data augmentation directly on the Symmetric Positive Definite (SPD) manifold on which EEG covariance matrices exist on. One notable approach generates new artificial trials by geometrically interpolating between pairs of existing covariance matrices of the same class (Kalunga et al., 2015). This is accomplished by creating points along the geodesic—the shortest path between two points on the manifold—which ensures that the newly generated matrices remain on the manifold (Kalunga et al., 2015). This scheme has been shown to successfully boost steady state visually evoked potential (SSVEP) and error potential (ERP) classification accuracies in the scenarios with few training samples and imbalanced classes (Kalunga et al., 2015) respectively.

However, this method has a distinct limitation: geodesic interpolation is restricted to generating data within the convex hull of the original training samples (Kalunga et al., 2015). It can densify the space between existing data points but cannot produce novel examples that explore the entire manifold. While the resulting points are new, their creation is an interpolative process, not a truly generative one. It cannot produce samples that represent plausible variations of the task that might exist in unexplored regions of the data manifold, away from these direct paths. This constraint further motivates the need for a more powerful, non-linear generative model. A Variational Autoencoder (VAE), which can learn a latent distribution of a manifold (Shao et al., 2017), may offer an alternative to bypass this convex-hull limitation and generate diverse yet representative synthetic data.

A VAE is a common generative model first introduced in the paper by Kingma & Welling (2013). A Variational Autoencoder comprises of two neural networks: a probabilistic encoder and decoder. The encoder maps an input data point to the mean and variance of a probability distribution typically within a lower-dimensional latent space. This distribution, $q_\phi(\mathbf{z}|\mathbf{x})$, serves as an approximation of the true intractable posterior distribution. The decoder network then takes a sample $\mathbf{z}$ from this learned latent distribution and aims to reconstruct the original input data, defining the likelihood $p_\theta(\mathbf{x}|\mathbf{z})$ (Kingma & Welling, 2013).

The model is trained by optimizing a single objective (i.e., Evidence Lower Bound) which balances two competing goals. The first is a reconstruction loss, which ensures the decoded samples are faithful to the original data. The second is a regularization term, the Kullback-Leibler (KL) divergence, which encourages the learned latent distributions toward a chosen prior distribution (Kingma & Welling, 2013). This training encourages a continuous and well-structured latent space, enabling the VAE to generate novel yet plausible data by decoding points sampled from this space.

The standard VAE architecture, however, assumes data lies in a Euclidean space, creating a fundamental conflict when working with SPD covariance matrices. A naive application that simply vectorizes the matrices and feeds them to a VAE ignores their essential geometric structure on the Riemannian manifold. This oversight, as will be empirically confirmed in Section 4.3.1, leads to an outcome where the model generates a high percentage of invalid matrices that are not symmetric and positive-definite, rendering them unusable for Riemannian-based classifiers. This establishes the central technical problem: a framework is required that can utilize the generative power of a VAE while explicitly respecting the geometry of the SPD manifold.

This thesis addresses this challenge by proposing and evaluating a modified Variational Autoencoder; an architecture specifically designed to preserve the geometric integrity of the data. The core innovation is in its use of Riemannian geometry to bridge the Euclidean and Riemannian domains, rendering it suitable for neural networks while also enabling the model to create geometrically faithful representations of the original data through the use of a composite loss function. For an in depth description of the architecture, refer to Section 3.2.

A major obstacle to widespread adoption is the significant inter-subject variability, which necessitates lengthy and user-specific calibration sessions to maximize performance (Congedo et al., 2017; Yger et al., 2017; Blankertz et al., 2007). While BCI performance can often be high in within-subject scenarios, this paper focuses specifically on the more challenging and practically relevant problem of cross-subject generalization (Congedo et al., 2017). The goal is to contribute towards BCI systems that are more robust and readily deployable, reducing the need for extensive data collection from every new user. By demonstrating the efficacy of a generative model in a cross-subject setting, the present work aims to validate an approach for learning subject-invariant features of motor imagery on the SPD manifold through the use of parallel transport (Yair et al., 2019) for data alignment. Accordingly, this investigation aims to first establish if a Riemannian geometry-preserving VAE can generate valid synthetic EEG covariance matrices, and second, to evaluate whether the synthetic data improves cross-subject MI-BCI performance.

The remainder of the paper is organized to systematically justify the proposed methodology. Section 2 first lays the essential theoretical concepts in Riemannian geometry. Building on this, Section 3 details the complete methodology, including the datasets, the specific preprocessing pipeline, the proposed architecture, and the training and evaluation protocols. Section 4 presents the empirical results, assessing both the fidelity of the generated synthetic data and its impact on cross-subject classification performance. Finally, Section 5 discusses the interpretation and implications of these findings, and further outlines promising directions for future research.

2 Theoretical Foundations

To fully understand the mechanisms that enable the proposed architecture, the following section provides the necessary theoretical foundation in Riemannian geometry.

2.1 The Manifold of SPD Matrices

In state-of-the-art BCI pipelines, EEG trials are represented by their spatial covariance matrices. These matrices are Symmetric Positive-Definite; they are symmetric and all their eigenvalues are strictly positive. Capturing the spatial correlations between EEG channels, these matrices are treated as points on a curved mathematical space known as a Riemannian manifold. Formally, The space of Symmetric Positive-Definite matrices is defined as:

$$\mathcal{M} = \{\mathbf{X} \in \mathbb{R}^{N \times N} \mid \mathbf{X} = \mathbf{X}^{\top}, \mathbf{X} \succ 0\} \qquad (2.1)$$

where $X \succ 0$ indicates that the matrix is composed of strictly positive eigenvalues (Yger et al., 2017). In the context of this work, $\mathbf{N}$ is the number of EEG channels, and each matrix $\mathbf{X}$ contains the spatial covariances measured between their signals.

An SPD matrix can be interpreted as a multi-dimensional generalization of variance. Due to the positivity constraint, these matrices do not occupy a standard Euclidean space; for instance, the space of 2×2 SPD matrices forms an open cone, creating a smoothly curved space known as a differentiable manifold (Yger et al., 2017; Congedo et al., 2017). Although the manifold is curved, its structure allows for local applications of standard linear algebra and calculus. Accounting for this inherent geometry is important to avoid geometric distortions, such as the "swelling effect," that can occur during naive Euclidean operations (Kalunga et al., 2015).

2.2 The Tangent Space

A Riemannian manifold, on a local level, behaves like a euclidean space (Yger et al., 2017). The local approximation at any point $\mathbf{P}$ on the manifold $\mathcal{M}$ is the tangent space $T_{\mathbf{P}}\mathcal{M}$, defined as the set of all $N \times N$ symmetric matrices at said point (Congedo et al., 2017). As illustrated in Figure 2.1, the tangent space can be visualized as a flat plane touching the curved manifold at a single point. By projecting SPD matrices onto this tangent space, the curved data is transformed into a standard symmetric vector, making it suitable for processing by the encoder and decoder networks.

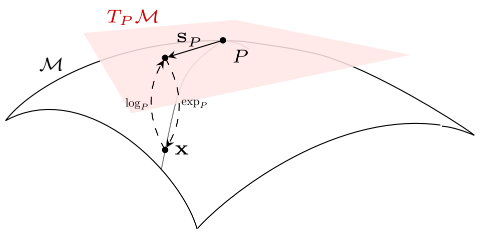


Figure 2.1: Visualization of the relationship between the Riemannian manifold $\mathcal{M}$ and its Euclidean-like tangent space $T_{\mathbf{P}}\mathcal{M}$, at a point $\mathbf{P}$. The Logarithmic map ($\log_{\mathbf{P}}$) projects a point $\mathbf{X}$ from the manifold to a symmetric matrix $\mathbf{S}_{\mathbf{P}}$ in the tangent space, while the Exponential map ($\exp_{\mathbf{P}}$) is the inverse transformation.

2.3 The Exponential and Logarithmic Maps

Transporting points between the Riemannian manifold and a tangent space is enabled by two fundamental operations:

(1) the logarithmic map $\log_{\mathbf{P}}(\mathbf{X_i})$, which projects a point $\mathbf{X_i}$ from the manifold onto the tangent space at point $\mathbf{P}$, and

(2) the exponential map $\exp_{\mathbf{P}}(\mathbf{S_i})$, which projects a symmetric matrix $\mathbf{S_i}$ from the tangent space at point $\mathbf{P}$ back onto the manifold (Yger et al., 2017; Congedo et al., 2017).

The formulas for these maps when using the Affine-Invariant metric are as follows (Yger et al., 2017), where $\mathbf{P}$ and $\mathbf{X_i}$ are SPD matrices on the manifold $\mathcal{M}$ and $\mathbf{S}_i$ is the corresponding symmetric matrix in the tangent space $T_{\mathbf{P}}\mathcal{M}$ at point $\mathbf{P}$:

$$\mathbf{S}_i = \log_{\mathbf{P}}(\mathbf{X_i}) = \mathbf{P}^{1/2}\log\left(\mathbf{P}^{-1/2}\mathbf{X_i}\mathbf{P}^{-1/2}\right)\mathbf{P}^{1/2} \tag{2.2}$$

$$\mathbf{X_i} = \exp_{\mathbf{P}}(\mathbf{S}_i) = \mathbf{P}^{1/2}\exp\left(\mathbf{P}^{-1/2}\mathbf{S}_i\mathbf{P}^{-1/2}\right)\mathbf{P}^{1/2} \tag{2.3}$$

In these formulas, $\log(\cdot)$ and $\exp(\cdot)$ denote the matrix logarithm and matrix exponential, respectively.

2.4 Distances and Geodesics

In a Euclidean space, the distance between two points is a straight line while on a curved manifold, the shortest path between two points, is a curve known as a geodesic. The length of this geodesic is the Riemannian distance, which also provides a way to measure similarity between SPD matrices. This work utilizes the Affine-Invariant Riemannian Metric (AIRM), defining the distance between two SPD matrices, $\mathbf{P}_1$ and $\mathbf{P}_2$, as follows:

$$d_r(\mathbf{P}_1, \mathbf{P}_2) = \left\| \log\left(\mathbf{P}_1^{-1/2}\mathbf{P}_2\mathbf{P}_1^{-1/2}\right)\right\|_F \tag{2.4}$$

where $\log(\cdot)$ and $\exp(\cdot)$ again denote the matrix logarithm and exponential (Yger et al., 2017).

The metric is invariant under congruence transformations; the distance between matrices is unchanged by spatial filtering, contributing to robust riemannian methods across subjects and sessions, thus making it very popular for BCI applications (Congedo et al., 2017).

2.5 The Riemannian Mean

Calculating the center of a cluster of points on a manifold requires a geometric approach, since it is not guaranteed that the simple arithmetic mean of SPD matrices will be an SPD matrix itself. Therefore, the Riemannian mean (often referred to as the Fréchet Mean) is utilized—a unique point on the manifold that minimizes the sum of squared Riemannian distances to all other matrices in a set $\{\mathbf{X}_i\}_{i=1}^{M_{\text{set}}}$ (Fletcher et al., 2004; Moakher, 2005). It is formally defined as:

$$\mathbf{G} = \underset{\mathbf{P}\in\mathcal{M}}{\arg\min} \sum_{i=1}^{M_{set}} d_r^2(\mathbf{P}, \mathbf{X}_i) \tag{2.5}$$

where $\mathbf{P}$ is the candidate SPD matrix over which the minimization occurs, $\mathcal{M}$ is the manifold of $N \times N$ SPD matrices, $d_r(\mathbf{P}, \mathbf{X}_i)$ is the Affine-Invariant Riemannian metric (as defined in Eq. 2.4) between $\mathbf{P}$ and $\mathbf{X}_i$. Unlike the arithmetic mean, the Riemannian mean for more than two matrices does not have a closed-form solution and must be found using iterative optimization algorithms (Yger et al., 2017).

For further in-depth overview on Riemannian Geometry, please refer to the works of Lee (2018) and Lang (1995).

3 Methodology

This section details the methods employed for generating synthetic EEG covariance matrices using the proposed Variational Autoencoder architecture that incorporates Riemannian geometry for MI-BCI data augmentation. The pipeline encompasses dataset description and preprocessing, model architecture, training, synthetic data generation, and evaluation.

3.1 Data and Preprocessing

In this subsection, descriptions of the datasets used and the preprocessing pipeline applied are provided. The primary objective of this pipeline is to transform EEG signals into standardized and subject-aligned SPD covariance matrices that serve as input for the proposed architecture, referred to as the Riemannian Variational Autoencoder (RVAE) onward.

3.1.1 Datasets

To rigorously evaluate the performance and generative plausibility of the proposed Riemannian VAE architecture, two distinct EEG datasets sourced from the "Mother of All BCI Benchmarks" (MOABB) framework (Aristimunha et al., 2023) are utilized. MOABB provides standardized access to a wide range of publicly available EEG datasets, facilitating comparisons with established BCI algorithms. The datasets selected are from Faller et al. (2012) and Leeb et al. (2008). These datasets were deliberately chosen to represent diverse potential BCI application scenarios based on EEG channel configurations. Both datasets feature recordings from multiple subjects performing cue-based, two-class Motor Imagery (MI) tasks. Specifically, the 3-channel dataset (Leeb et al., 2008) serves as a proxy for scenarios where BCI systems might employ fewer sensors, reflecting potential applications in the real world (Congedo et al., 2017). Conversely, the 13-channel dataset (Faller et al., 2012) aligns more closely with typical research settings where higher channel density provides richer spatial information for detailed neurological analysis. The use of two datasets with differing numbers of subjects, channels, and trial counts allows for a more comprehensive assessment of the model's capabilities.

The 13-channel dataset contains recordings from 12 subjects. The MI paradigm involved discriminating between sustained imagination of right-hand movement versus movement of both feet. EEG data was acquired from three Laplacian derivations around the C3, Cz, and C4 electrode positions (specifically: FC3, FCz, FC4, C5, C3, C1, Cz, C2, C4, C6, CP3, CPz, CP4), sampled at 512 Hz. The experimental paradigm presented a cue at second 3, after which subjects performed the designated motor imagery task until second 8, resulting in a 5-second MI activity period per trial, of which the last 4 seconds were used. The data loading procedure, using the MOABB framework, resulted in a total of 5572 trials across the 12 subjects, with individual participants contributing either 398 or 597 trials.

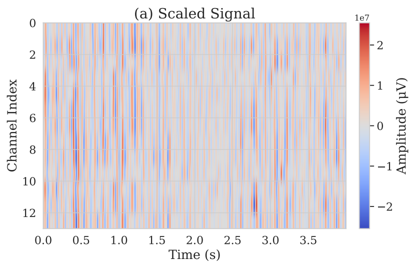
The 3-channel dataset consists of recordings from 9 right-handed subjects performing an MI task; left-hand versus right-hand imaginary movements. Signals were captured using three bipolar EEG channels (C3, Cz, C4) at a sampling frequency of 250 Hz. Trials consisted of a visual cue followed by a 4-second MI execution period. By utilizing data aggregated across all available sessions (including screening and feedback runs), this dataset provides up to 360 trials per MI class for each subject.

For both datasets, the original data acquisition included bandpass filtering between 0.5 Hz and 100 Hz and a 50 Hz notch filter to mitigate power line noise. However, further preprocessing tailored to the requirements of the model was applied, as detailed below.

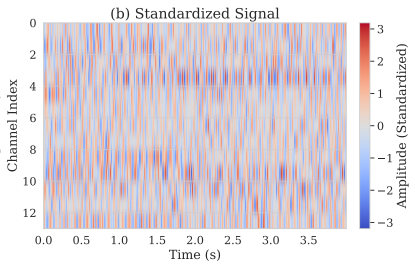
3.1.2 Preprocessing

The raw EEG data, loaded using the MOABB library, underwent a standardized preprocessing pipeline designed to enhance signal quality and extract features leading to a suitable input format for the geometry-aware VAE model.

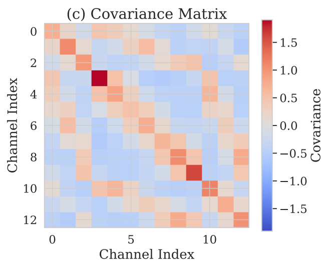
The data loading process, facilitated by the MOABB library, yielded the set of processed EEG epochs for each trial. During this loading phase, the relevant signal segments corresponding to individual trials were bandpass-filtered between 8 Hz and 30 Hz. This frequency band targets the alpha and beta rhythms most relevant to sensorimotor activity during motor imagery. Concurrently, these filtered segments were temporally cut to extract the core MI activity window, resulting in the epochs used before subject alignment preprocessing. Each epoch captures multi-channel EEG data from 4 to 8 seconds representing the activity period for the



(a) Raw EEG signals scaled to microvolts across 13 channels. The signal amplitudes are visually less dispersed, making subtle variations challenging to discern before further processing.



(b) Signals after EMS, highlighting high-frequency fluctuations enhancing temporal patterns. This process normalizes the signal amplitude, allowing for clearer dynamic changes.



(c) Final 13×13 SPD covariance matrix after applying the OAS estimator, showing channel variance on the diagonal and covariance on the off-diagonal.

Figure 3.1: EEG preprocessing pipeline transformations for a single trial from subject 1 (fold 2).

13-channel dataset, and 3 to 7 seconds for the 3-channel dataset.

The raw voltage signals were then scaled to microvolts (μV), a standard unit in EEG analysis, using a multiplication factor of 10^6. Following scaling, Exponential Moving Standardization (EMS) was applied independently to the time series data within each epoch. The preprocessing technique, adopted from the work of Schirrmeister et al. (2017), adaptively smooths the EEG signal using exponentially weighted moving statistics (mean and variance). EMS can be seen applied to data for training deep learning models as these models tend to be sensitive to the input scale (Zhu et al., 2022).

The necessary step for obtaining input compatible with the Riemannian geometry-based methods involved transforming the standardized EEG epochs into SPD covariance matrices. Spatiotemporal covariance matrices were estimated for each epoch using the Oracle Approximating Shrinkage (OAS), a method optimized for high-dimensional data where sample sizes are small (Chen et al., 2010). In our case, the use of sample covariance matrix would suffice to convert the EEG signals to covariance matrices, but the use of a shrinkage estimator provides a stronger guarantee that the resulting input matrices are well-conditioned (not close to a singular matrix) and strictly positive definite, which is required for the mathematical operations that are utilized during the training phase. The sequential transformations can be seen in Figure 3.1. This robust estimation technique yields an $N \times N$ SPD matrix for each trial where N represents the number of EEG channels.

To address the inherent variability between individuals' EEG signal characteristics, which manifests as geometric differences in their respective covariance matrices and thus their location on the Riemannian manifold, parallel transport (Yair et al., 2019) was applied as a final preprocessing step. This approach aligned the covariance matrices, thereby facilitating the learning of subject-invariant features and preventing deformation of distant covariance matrices when using a global reference point for the tangent space. Parallel transport maps matrices from different subjects (each potentially having their own geometric 'center' or reference) onto a common reference frame on the SPD manifold, which in turn preserves the intrinsic geometric relationships relevant to the MI task.

Data leakage during the alignment procedure was prevented by fitting the transport exclusively on the training data for each validation fold (i.e., data from all subjects except the subject held-out for testing). The fitting step involved:

(1) Computing a global reference point $\mathbf{G}$, defined as the Affine-Invariant Riemannian Mean (as defined in Section 2.5) of a set of SPD matrices $\{\mathbf{X}_i\}_{i=1}^{M_{set}}$ belonging to the subjects in the training set. Utilizing the implementation provided by Barachant et al. (2025), this mean is found using an iterative gradient descent algorithm, which converges based on a tolerance parameter of 10^-9 within a maximum of 50 iterations.

(2) Computing subject-specific reference points $\mathbf{S}_k$ for each subject k within the training set, cal-

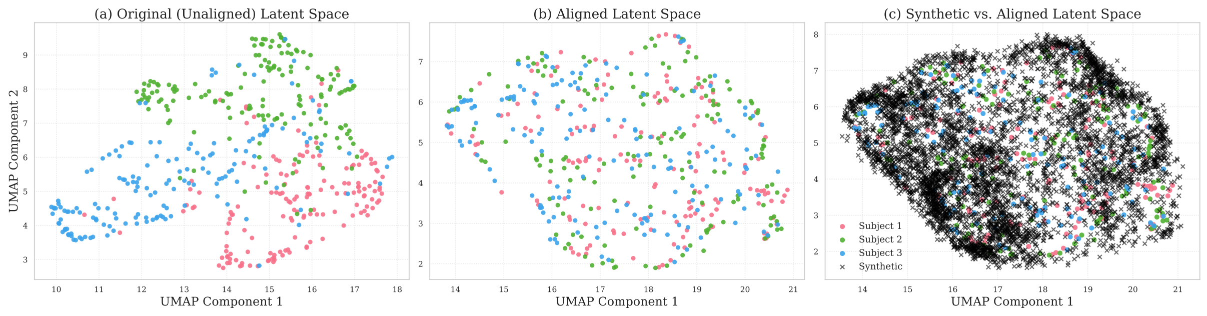


Figure 3.2: UMAP visualization of covariance matrix distributions across different pipeline stages (trained on right hand data). (a) Unaligned covariance matrices show distinct, subject-specific clusters. (b) Parallel transport aligns matrices, merging clusters and reducing inter-subject variance. (c) Aligned real data with synthetic samples generated using prior sampling, demonstrating the model's generative capacity.

culated as the Riemannian mean (Eq. 2.5) of *all* the subject's matrices irrespective of class label. This class-agnostic subject reference aims to capture the average geometric characteristics specific to that individual, thereby focusing the alignment on reducing overall subject geometry relative to the global reference before subsequent class-specific modeling.

(3) Deriving subject-specific transport operators $\mathbf{E}_{\mathbf{S}_k\to\mathbf{G}}$. These operators, aligning with the formulation detailed in Yair et al. (2019) (specifically Appendix A), are computed as $\mathbf{E}_{\mathbf{S}_k\to\mathbf{G}} = \mathbf{S}_k^{1/2}(\mathbf{S}_k^{-1/2}\mathbf{G}\mathbf{S}_k^{-1/2})^{1/2}\mathbf{S}_k^{-1/2}$. This method first computes an intermediate matrix $\mathbf{X}_0 = \mathbf{S}_k^{-1/2}\mathbf{G}\mathbf{S}_k^{-1/2}$, which is itself Symmetric Positive-Definite if $\mathbf{G}$ and $\mathbf{S}_k$ are. The principal square root of $\mathbf{X}_0$ is then calculated, followed by multiplication with $\mathbf{S}_k^{1/2}$ and then multiplication with $\mathbf{S}_k^{-1/2}$ to form the final operator $\mathbf{E}_{\mathbf{S}_k\to\mathbf{G}}$. This construction ensures that all matrix square root operations are performed on SPD matrices, making it compatible with numerically stable eigendecomposition-based methods, which were used for these matrix operations. The resulting operator $\mathbf{E}_{\mathbf{S}_k\to\mathbf{G}}$ maps points from the geometric vicinity of the subject's reference $\mathbf{S}_k$ to that of the global reference $\mathbf{G}$.

Once fitted on the training data, the transport was applied to both the training and the held-out test set for that fold. Each covariance matrix $\mathbf{X}_i$ belonging to a subject k was transformed using the corresponding pre-computed operator $\mathbf{E}_{\mathbf{S}_k\to\mathbf{G}}$ via the congruence transformation:

$$\mathbf{X}_{\text{aligned},i} = \mathbf{E}_{\mathbf{S}_k\to\mathbf{G}}\mathbf{X}_i\mathbf{E}_{\mathbf{S}_k\to\mathbf{G}}^T \tag{3.1}$$

Throughout these geometric calculations, a small epsilon value ($\epsilon = 1 \times 10^{-6}$) was used to ensure numerical stability, particularly during matrix square root and inverse computations. The positive-definiteness of the resulting aligned matrices was explicitly enforced by checking if the minimum eigenvalue λ_{min} of a matrix falls below ϵ, and if so, adding a scaled identity matrix ($(\epsilon - \lambda_{min})\mathbf{I}$) to shift the minimum eigenvalue up to the ϵ threshold. This SPD enforcement method is also used in the model architecture.

The parallel transport alignment, as seen in Figure 3.2, yielded the final set of SPD matrices used as input for the RVAE model, effectively reducing inter-subject geometric variability while aiming to retain task discriminative information embedded within the manifold structure.

3.2 Model Architecture

Building upon the foundational concepts of Variational Autoencoders and Riemannian geometry discussed in Sections 1 and 2, the current section details the specific architecture of the RVAE developed for processing SPD matrices. The overall architecture, as visualized in Figure 3.3, integrates Riemannian geometric operations with standard neural network components. At its core, the RVAE aims to learn a latent representation $\mathbf{z}_i$ from input EEG covariance matrices $\mathbf{X}_i$. This involves

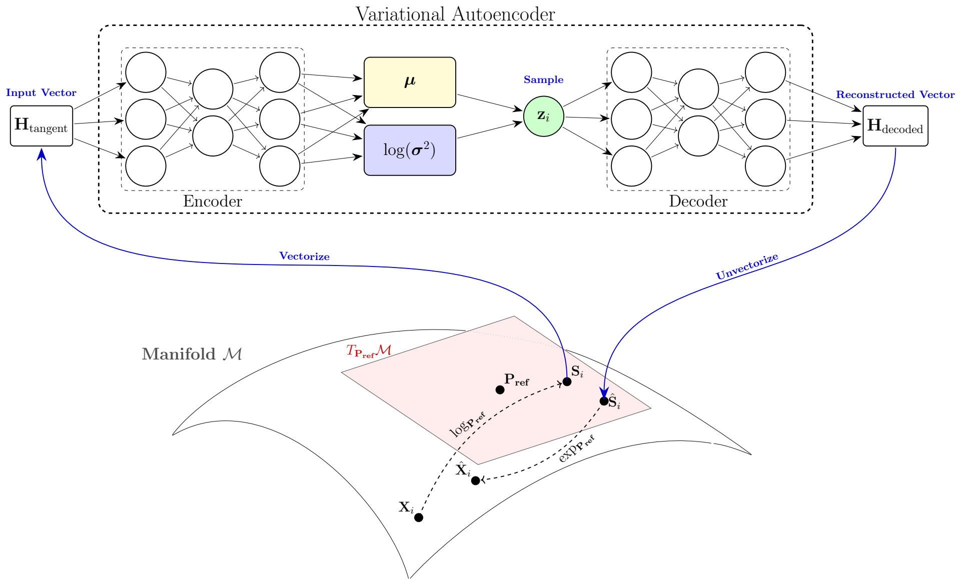


Figure 3.3: An overview of the proposed Riemannian Variational Autoencoder, illustrating the integration of a standard VAE with geometric operations on the SPD manifold. An input SPD matrix $\mathbf{X}_i$ is first projected onto the tangent space at a reference point $\mathbf{P}_{\text{ref}}$ using the logarithmic map $\log_{\mathbf{P}_{\text{ref}}}$ (Eq. 2.2). This tangent representation $\mathbf{S}_i$ is then vectorized to serve as the encoder input $\mathbf{H}_{\text{tangent}}$. The encoder maps this input to a latent distribution parameterized by $\boldsymbol{\mu}$ and $\log(\boldsymbol{\sigma}^2)$, from which a latent vector $\mathbf{z}_i$ is sampled and passed to the decoder to produce the reconstructed vector $\mathbf{H}_{\text{decoded}}$. The vector is unvectorized back into a tangent space representation $\hat{\mathbf{S}}_i$, which is finally mapped back onto the SPD manifold via the exponential map ($\exp_{\mathbf{P}_{\text{ref}}}$) (Eq. 2.3) to produce the reconstructed SPD matrix $\hat{\mathbf{X}}_i$.

an initial mapping of the input SPD matrices to a representation within a tangent space, which is then vectorized for processing by the neural network components (encoder and decoder) operating in Euclidean space. From the learned latent space, the model can generate new SPD matrices by decoding sampled latent vectors and mapping them back onto the SPD manifold, effectively preserving the inherent geometric properties of the data.

The transformations between the SPD manifold and the Euclidean tangent space representations, are critically depend on a global reference point $\mathbf{P}_{\text{ref}}$. This reference point is established once at the very beginning of the training phase. It is computed as the Riemannian mean (as defined in Equation 2.5 of Section 3.1.2) of all SPD matrices present in the training set corresponding to the specific MI-class for which the particular RVAE model instance is being trained. Once computed, $\mathbf{P}_{\text{ref}}$ is fixed and serves as the constant anchor for all geometric mapping operations—both projecting data onto a tangent space and mapping data back from the tangent space to the SPD manifold—throughout the model's training and the subsequent inference/generation phase.

The sequence of operations that constitutes the model's forward pass, detailing how a batch of input SPD matrices $\mathcal{X}$ is transformed to produce their reconstructions $\hat{\mathcal{X}}$ while simultaneously learning their latent representations $\mathcal{Z}$, is outlined below:

1. **Input**: The model processes a batch of input SPD covariance matrices, denoted as $\mathcal{X} = \{\mathbf{X}_1, \mathbf{X}_2, ..., \mathbf{X}_B\}$, where each $\mathbf{X}_i$ is an aligned $N \times N$ SPD matrix and B is the batch size.

2. **Map to Tangent Representation**: The entire batch of input SPD matrices $\mathcal{X}$ is transformed into a corresponding batch of tangent space matrices $\mathcal{S}'$. The projection is

achieved by applying a version of the logarithmic map (see Section 2.3) where the data is first whitened with respect to the reference point $\mathbf{P}_{\text{ref}}$ before the matrix logarithm is taken. Each resulting matrix $\mathbf{S}'_i \in \mathcal{S}'$ resides in a vector space (the space of $N \times N$ symmetric matrices, which is Euclidean) and its vectorized form (specifically, its upper triangular part) serves as the input to the subsequent neural network layers.

The transformation process begins with the computation of the inverse square root of the reference point $\mathbf{P}_{\text{ref}}^{-1/2}$. This operator is derived from the eigendecomposition of $\mathbf{P}_{\text{ref}}$. Since $\mathbf{P}_{\text{ref}}$ is eigendecomposed as $\mathbf{P}_{\text{ref}} = \mathbf{V}_{\text{ref}}\mathbf{\Lambda}_{\text{ref}}\mathbf{V}_{\text{ref}}^T$, where $\mathbf{V}_{\text{ref}}$ is the matrix of orthonormal eigenvectors and $\mathbf{\Lambda}_{\text{ref}} = \text{diag}(\lambda_{\text{ref},1}, \ldots, \lambda_{\text{ref},N})$ is the diagonal matrix of its positive eigenvalues, its inverse square root is constructed as (Horn & Johnson, 2013; Petersen & Pedersen, 2008):

$$\mathbf{P}_{\text{ref}}^{-1/2} = \mathbf{V}_{\text{ref}}\mathbf{\Lambda}_{\text{ref}}^{-1/2}\mathbf{V}_{\text{ref}}^T \tag{3.2}$$

where $\mathbf{\Lambda}_{\text{ref}}^{-1/2}$ is the diagonal matrix with entries $\lambda_{\text{ref},j}^{-1/2}$. This matrix $\mathbf{P}_{\text{ref}}^{-1/2}$ serves as a whitening operator.

Subsequently, the batch of input matrices $\mathcal{X}$ is whitened using $\mathbf{P}_{\text{ref}}^{-1/2}$ via batched matrix multiplication:

$$\mathcal{X}' = \mathbf{P}_{\text{ref}}^{-1/2}\mathcal{X}(\mathbf{P}_{\text{ref}}^{-1/2})^T \tag{3.3}$$

This whitening transformation congruence transforms each matrix $\mathbf{X}_i \in \mathcal{X}$ by $\mathbf{P}_{\text{ref}}^{-1/2}$. Geometrically, the transformation maps the matrices from the vicinity of $\mathbf{P}_{\text{ref}}$ on the SPD manifold to the vicinity of the identity matrix $\mathbf{I}$. For instance, if an input $\mathbf{X}_i = \mathbf{P}_{\text{ref}}$, then its whitened form $\mathbf{X}'_i = \mathbf{I}$. The whitening process is vital for stabilizing the subsequent matrix logarithm operation.

The core mapping is then achieved by applying the matrix logarithm to the whitened batch $\mathcal{X}'$. The calculation for each symmetric matrix $\mathbf{S}'_i$ in the final output batch is defined by the following equation:

$$\mathbf{S}'_i = \log(\mathbf{X}'_i) = \mathbf{V}_{w,i}\log(\mathbf{\Lambda}_{w,i})\mathbf{V}_{w,i}^T \tag{3.4}$$

where $\mathbf{X}'_i = \mathbf{V}_{w,i}\mathbf{\Lambda}_{w,i}\mathbf{V}_{w,i}^T$ is the eigendecomposition of a single sample from the whitened batch $\mathcal{X}'$ (with the subscript 'w' denoting 'whitened'), and $\log(\mathbf{\Lambda}_{w,i})$ is a diagonal matrix of the natural logarithms of its eigenvalues. This completes the transformation, where the final batch $\mathcal{S}'$ represents the set of symmetric matrices in the tangent space at the identity. The conceptual formula for the entire batch-wise logarithmic mapping is therefore $\mathcal{S}' = \log(\mathbf{P}_{\text{ref}}^{-1/2}\mathcal{X}(\mathbf{P}_{\text{ref}}^{-1/2})^T)$.

3. **Vectorize Symmetric Matrix**: The batch of symmetric matrices $\mathcal{S}' \in \mathbb{R}^{B\times N\times N}$ (representing points in the tangent space), obtained from the previous step, is then transformed into a batch of flat vector representations, denoted $\mathbf{h}_{\text{tangent}} \in \mathbb{R}^{B\times D_{spd}}$. This vectorization is necessary for processing by standard fully connected neural network layers.

   The operation consists of extracting the $D_{spd} = N(N+1)/2$ unique elements from each matrix $\mathcal{S}'_i$ in the batch. Specifically, for each matrix, the elements from the main diagonal and the upper triangular part (i.e., elements s'_{jk} where $j \leq k$) are selected and arranged into a vector. This is achieved by systematically taking these elements in a row-wise order (e.g., $s'_{11}, s'_{12}, \ldots, s'_{1N}, s'_{22}, s'_{23}, \ldots, s'_{NN}$). The extraction process is applied in parallel to every matrix in the batch $\mathcal{S}'$ to produce the corresponding batch of vectors $\mathbf{H}_{\text{tangent}}$.

4. **Encoder Network**: The batch of vectorized tangent space representations $\mathbf{H}_{\text{tangent}} \in \mathbb{R}^{B\times D_{spd}}$ serves as the input to the encoder network. The encoder's primary role is to learn a non-linear mapping from the input—which can range in dimensionality (e.g., $D_{spd} = 6$ for 3 channels, $D_{spd} = 91$ for 13 channels)—to an intermediate representation. This representation is then used to parameterize the $D_{lat} = 64$ dimensional latent distribution. The encoder architecture is a feedforward neural network designed to transform and extract features from the entire batch $\mathbf{H}_{\text{tangent}}$.

   The transformation process within the encoder is composed of five sequential blocks, each following a consistent structure: a Linear layer, a Batch Normalization layer, and finally a

LeakyReLU activation function. Batch Normalization is applied directly after each linear transformation to standardize the inputs to the next stage, which stabilizes training dynamics, reduces internal covariate shift, and aids convergence (Ioffe & Szegedy, 2015)—a particularly helpful feature given the complex nature of the geometrically-derived input data. Following this normalization, the LeakyReLU activation (Maas et al., 2013) is applied. This particular activation function is essential because the tangent space vectors contain both positive and negative values; standard ReLU activations would map all negative inputs to zero, leading to the problem where neurons become permanently inactive (Maas et al., 2013). By allowing a small, non-zero gradient for negative inputs, LeakyReLU ensures all neurons remain adaptive and preserves the network's representational capacity.

The model's encoder, transforms the initial batch $\mathbf{H}_{\text{tangent}}$ into a final batch of encoded feature vectors $\mathbf{H}_{\text{encoded}} \in \mathbb{R}^{B\times 64}$. The selection of hidden layer dimensions, as shown below, aims to balance sufficient representational capacity with model complexity.

Input

$$\begin{tikzcd} & & & & & & D_{\text{lat}} \\ D_{\text{spd}} \arrow[r] & 32 \arrow[r] & 64 \arrow[r] & 16 \arrow[r] & 32 \arrow[r] & 64 \arrow[ur, "\log\sigma^2"] \arrow[dr, "\mu"'] & \\ & & & & & & D_{\text{lat}} \end{tikzcd}$$

For the 13-channel dataset, this initial step reduces dimensionality ($91 \to 32$), encouraging a compact summary, while for the 3-channel dataset, it acts as an expansion ($6 \to 32$), potentially allowing for richer feature interaction from a lower-dimensional input. These resulting encoded vectors capture the learned characteristics from the tangent space and serve as the input for parameterizing the latent variables.

5. **Latent Variables**: The batch of 64-dimensional feature vectors $\mathbf{H}_{\text{encoded}} \in \mathbb{R}^{B\times 64}$, produced by the final layer of the encoder network, serves as the input to two distinct linear projection layers. These layers map the entire batch $\mathbf{H}_{\text{encoded}}$ to the parameters that define the approximate posterior distribution over the latent variables for each sample.

   Specifically, one linear layer projects $\mathbf{H}_{\text{encoded}}$ to a batch of mean vectors $\mathbf{M} \in \mathbb{R}^{B\times D_{\text{lat}}}$, and a second, separate linear layer projects it to a batch of log-variance vectors $\log \boldsymbol{\Sigma}^2 \in \mathbb{R}^{B\times D_{\text{lat}}}$, where the latent dimension is set to $D_{\text{lat}} = 64$. Each component $(\log \boldsymbol{\Sigma}^2)_{ij}$ represents the log-variance for the j-th latent dimension of the i-th sample.

   These batches of parameters, $\mathbf{M}$ and $\log \boldsymbol{\Sigma}^2$, are used to define a corresponding batch of approximate posterior distributions $q_\phi(\mathbf{z}_i|\mathbf{X}_i)$, which are then regularized with a Gaussian prior distribution $p(\mathbf{z}) = \mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{I})$.

6. **Reparameterization**: To sample a batch of latent vectors $\mathbf{Z}$ from the learned approximate posterior distributions in a manner that allows for the backpropagation of gradients, the reparameterization trick is employed. The construction for each latent vector $\mathbf{z}_i$ within the batch is as follows:

$$\mathbf{z}_i = \boldsymbol{\mu}_i + \boldsymbol{\epsilon}_i \odot \exp(0.5 \cdot \log \boldsymbol{\sigma}_i^2) \tag{3.5}$$

   In this equation, $\boldsymbol{\mu}_i$ and $\log \boldsymbol{\sigma}_i^2$ are the mean and log-variance vectors for the i-th sample, taken from the parameter batches $\mathbf{M}$ and $\log \boldsymbol{\Sigma}^2$. The term $\boldsymbol{\epsilon}_i$ is a random noise vector drawn from a standard Gaussian distribution, which is part of a batch of noise vectors $\mathbf{E} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. This method separates the stochastic part of the sampling (the random noise $\mathbf{E}$) from the learned parameters, allowing gradients to flow back through $\mathbf{M}$ and $\log \boldsymbol{\Sigma}^2$ during training.

7. **Decoder Network**: The batch of sampled latent vectors $\mathbf{Z} \in \mathbb{R}^{B\times D_{\text{lat}}}$ serves as the input to the decoder network. By learning a non-linear mapping from the latent space, the decoder produces a batch of output vectors, $\mathbf{H}_{\text{decoded}} \in \mathbb{R}^{B\times D_{\text{spd}}}$. The objective is for these outputs to closely match the original input vectors $\mathbf{H}_{\text{tangent}}$. The decoder's feedforward architecture, as shown below, is designed to build the complexity needed for this reconstruction, by effectively mirroring the encoder's structure—Linear Layer, Batch Normalization layer, followed by a LeakyReLU activation function.

Output

$D_{\text{lat}} \to 32 \to 16 \to 64 \to 32 \to D_{\text{spd}}$

The final layer, consisting of a single Linear layer, projects the 32-dimensional representation to the target D_{spd}-dimensional output batch, $\mathbf{H}_{\text{decoded}}$. No activation or normalization is applied to this final output, as the components of the tangent space vectors can be any real value.

8. **Unvectorize**: The batch of decoded vectors $\mathbf{H}_{\text{decoded}} \in \mathbb{R}^{B \times D_{spd}}$ is transformed back into a batch of symmetric $N \times N$ matrices, denoted $\hat{\mathcal{S}}'$, reversing the initial vectorization step. For each matrix $\hat{\mathbf{S}}'_i \in \hat{\mathcal{S}}'$, corresponding vector elements from $\mathbf{H}_{\text{decoded}}$ populate the upper and lower triangles, ensuring the resulting matrices in $\hat{\mathcal{S}}'$ are symmetric.

9. **Map to SPD Manifold**: To produce the reconstructed covariance matrices $\hat{\mathcal{X}}$, the batch of symmetric matrices $\hat{\mathcal{S}}'$ (reconstructed tangent space representations) is mapped back onto the SPD manifold $\mathcal{M}$. The process is an application of the exponential map (see Section 2.3) which reverses the initial logging operation.

   Beginning with numerical stabilization, the batch $\hat{\mathcal{S}}'$ is explicitly re-symmetrized to ensure that any potential asymmetries, possibly introduced by floating-point arithmetic, are eliminated. For each matrix $\hat{\mathbf{S}}'_i$ in the batch, the operation is:

$$\hat{\mathbf{S}}''_i = \frac{\hat{\mathbf{S}}'_i + (\hat{\mathbf{S}}'_i)^T}{2} \tag{3.6}$$

   The matrix exponential, $\exp(\cdot)$, of this symmetrized batch $\hat{\mathcal{S}}''$ is computed using an eigendecomposition-based method. First, for each matrix $\hat{\mathbf{S}}''_i \in \hat{\mathcal{S}}''$, its eigenvalues $\lambda_{i,j}$ are found. To ensure numerical stability and prevent overflow, these eigenvalues are conditionally scaled. Setting a threshold $T = 20$ and defining $\lambda_{i,\max} = \max_j |\lambda_{i,j}|$, the scaled eigenvalues $\lambda'_{i,j}$ are:

$$\lambda'_{i,j} = \begin{cases} \lambda_{i,j} \cdot (T/\lambda_{i,\max}) & \text{if } \lambda_{i,\max} > T \\ \lambda_{i,j} & \text{otherwise} \end{cases}$$

   Next, the matrix exponential for each sample, $\mathbf{E}_i$, is calculated using its eigenvectors ($\mathbf{V}_i$) and the diagonal matrix of its exponentiated scaled eigenvalues ($\exp(\mathbf{\Lambda}'_i)$) as follows (Horn & Johnson, 2013; Petersen & Pedersen, 2008):

$$\mathbf{E}_i = \mathbf{V}_i \exp(\mathbf{\Lambda}'_i)\mathbf{V}_i^T \tag{3.7}$$

   where $\exp(\mathbf{\Lambda}'_i) = \text{diag}(e^{\lambda'_{i,1}}, \ldots, e^{\lambda'_{i,N}})$. The resulting batch of matrices $\mathcal{E}$, is guaranteed to be SPD and represents the mapping of the symmetric matrices from the tangent space at the identity.

   Finally, to map the batch $\mathcal{E}$ back to the vicinity of the class-specific reference point $\mathbf{P}_{\text{ref}}$, the matrix square root of $\mathbf{P}_{\text{ref}}$ is required. This square root $\mathbf{P}_{\text{ref}}^{1/2}$, is computed via eigendecomposition, where the eigenvalues of $\mathbf{P}_{\text{ref}}$ are clamped to be non-negative before the square root is taken to ensure stability. The batch of SPD matrices $\mathcal{E}$ is then transformed using $\mathbf{P}_{\text{ref}}^{1/2}$ via a congruence transformation to yield an intermediate batch $\tilde{\mathcal{X}}$:

$$\tilde{\mathcal{X}} = \mathbf{P}_{\text{ref}}^{1/2}\mathcal{E}(\mathbf{P}_{\text{ref}}^{1/2})^T \tag{3.8}$$

   To ensure every reconstructed matrix in the final batch $\hat{\mathcal{X}}$ is a valid SPD matrix, a final enforcement step is applied. This procedure uses the same mathematical principle for ensuring positive definiteness as detailed in the preprocessing stage (see Section 3.1.2), but is implemented here using PyTorch (Paszke et al., 2019) operations for end-to-end differentiability. The procedure first computes the minimum eigenvalue, $\lambda_{\min}$, for each matrix in the batch $\tilde{\mathcal{X}}$. If $\lambda_{\min}$ is less than a small threshold $\epsilon = 1 \times 10^{-5}$, a scaled identity matrix, $(\epsilon - \lambda_{\min})\mathbf{I}$, is added to shift the minimum eigenvalue up to ϵ. The resulting matrices are re-symmetrized one last time (as implemented in Eq. 3.6) to correct for minor floating-point errors from the exponential mapping that break the symmetry constraint, yielding the final reconstructed batch of SPD matrices $\hat{\mathcal{X}}$. The conceptual formula for this entire mapping is $\hat{\mathcal{X}} = \mathbf{P}_{\text{ref}}^{1/2}\exp(\hat{\mathcal{S}}')(\mathbf{P}_{\text{ref}}^{1/2})^T$, supplemented by these steps for numerical stability and correctness.

10. **Output**: The final output is the batch of reconstructed SPD covariance matrices $\hat{\mathcal{X}}$. Each individual matrix $\hat{\mathbf{X}}_i \in \hat{\mathcal{X}}$ is the reconstruction corresponding to an input matrix $\mathbf{X}_i$.

A separate instance of this architecture is trained independently for each class of motor imagery data. Each class-specific model therefore determines its own reference point $\mathbf{P}_{\text{ref}}$ based on the training samples belonging exclusively to that class. Consequently, all geometric operations (mapping to/from tangent space) within a class-specific model are relative to its own learned class reference.

3.3 Model Training

The class-specific training approach allows each model to specialize in capturing the unique geometric characteristics of its assigned class across different individuals. The training procedure for each model consists of the following components:

Loss Function Formulation: The model's parameters are optimized using a composite loss function L_{total}, designed to balance data fidelity, latent space regularization, and generative diversity. This function comprises several key components.

Firstly, the reconstruction loss L_{recon} targets accurate reconstruction through two distinct, fully differentiable mechanisms. The primary component, the manifold reconstruction loss L_{manifold}, incorporates the Affine-Invariant Riemannian metric (Equation 2.4) to ensure geometric fidelity on the SPD manifold, calculated as:

$$L_{\text{manifold}} = \text{mean}(d_r(\mathcal{X}, \hat{\mathcal{X}})) \tag{3.9}$$

This entire operation is implemented using batch-wise PyTorch (Paszke et al., 2019) functions, allowing the geometric error between the input batch $\mathcal{X}$ and the reconstructed batch $\hat{\mathcal{X}}$ to be back-propagated through the network.

Additionally, L_{recon} includes a component to promote accurate reconstruction within the Euclidean tangent space. This measures the error between the original batch of tangent vectors $\mathbf{H}_{\text{tangent}}$, and the batch of decoded tangent vectors $\mathbf{H}_{\text{decoded}}$. This is achieved by calculating the squared error between the decoded tangent vectors ($\mathbf{H}_{\text{decoded}}$) and the original tangent vectors ($\mathbf{H}_{\text{tangent}}$), normalized by the squared magnitude of the original vectors. This approach may provide the benefit of balancing the loss contribution from tangent vectors with different magnitudes. The formula for this normalized reconstruction error, averaged over the batch, is:

$$L_{\text{tangent}} = \frac{1}{B} \sum_{i=1}^{B} \frac{\sum_j \left(\mathbf{h}_{\text{decoded},i,j} - \mathbf{h}_{\text{tangent},i,j}\right)^2}{\sum_j \mathbf{h}^2_{\text{tangent},i,j} + \epsilon} \tag{3.10}$$

where j indexes the vector dimensions and ϵ is a stability constant equal to 1×10^{-6}. These two reconstruction components are averaged over the batch and summed together:

$$L_{\text{recon}} = L_{\text{manifold}} + L_{\text{tangent}} \tag{3.11}$$

Secondly, the KL Divergence regularizes the latent space. It minimizes the divergence between the learned approximate posterior distribution for each sample, $q_\phi(\mathbf{z}_i|\mathbf{X}_i)$, and a standard Gaussian prior $p(\mathbf{z}) = \mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{I})$. This encourages the latent space to be well-structured. Its analytical form, averaged over the batch, is:

$$L_{\text{KL}} = \frac{1}{B} \sum_{i=1}^{B} \left[-0.5 \sum_{k=1}^{D_{\text{lat}}} (1 + \log \sigma^2_{i,k} - \mu^2_{i,k} - \exp(\log \sigma^2_{i,k})) \right] \tag{3.12}$$

where B is the number of samples in the batch. Its influence is controlled by β (as shown in Eq. 3.14), which undergoes annealing during training, a technique introduced as KL cost annealing by Bowman et al. (2016). This annealing linearly increases β during training from a starting value of 0.0001 to an ending value of 0.2, thereby aiding stability and ensuring appropriate KL regularization without overwhelming the reconstruction loss early in training. These specific start and end values were determined empirically with the goal of achieving a final KL divergence value of approximately 2.0. This target represents a balance; a very low KL value can lead to posterior collapse where the latent space is not informative for generation (Bowman et al., 2016), while a value too high can overpower the reconstruction loss, leading to poor data fidelity.

Thirdly, to encourage the generation of diverse samples, a diversity loss $L_{\text{diversity}}$ is included. This loss promotes a larger geometric volume within the tangent space representation. Since the determinant of a covariance matrix is geometrically related to the volume spanned by the data points, it serves as a measure of generalized variance or data spread. Maximizing this determinant therefore encourages the generated vectors to be more diverse and cover a wider region of the space. Accordingly, the loss is formulated to maximize the determinant by minimizing its negative logarithm. Let $\mathbf{H}_{\text{decoded}}$ be the matrix where each row is a decoded tangent vector $\mathbf{h}_{\text{decoded},i}$ from the current batch. The loss is then calculated as the negative log-determinant, weighted by an empirically determined factor $\gamma = 0.035$ and stabilized with an identity matrix scaled by $\epsilon_{\text{cov}} = 1 \times 10^{-6}$:

$$L_{\text{diversity}} = -\log\det(\text{Cov}(\mathbf{H}_{\text{decoded}}^T) + \epsilon_{\text{cov}}\mathbf{I}) \tag{3.13}$$

The total loss, incorporating these components with their respective weights, is calculated as:

$$L_{\text{total}} = L_{\text{recon}} + \beta \cdot L_{\text{KL}} + \gamma \cdot L_{\text{diversity}} \tag{3.14}$$

Optimization Strategy: The AdamW optimizer (Loshchilov & Hutter, 2017) is employed with an empirically found learning rate of 1×10^{-4} and a weight decay parameter of 1×10^{-6}, which applies L2 regularization directly to the model's parameters. To ensure stable convergence, the learning rate was adaptively reduced by a factor of 0.5 if the loss stagnated for 20 epochs. The training was further regularized by clipping the gradient's L2 norm to a threshold of 1.0.

For the covariance matrix in equation 3.13 to be well-conditioned and likely non-singular, the number of samples in the batch must ideally be greater than the dimensionality of the tangent vectors (D_{spd}). A sufficiently large batch size ensures a more robust estimation of the data's covariance structure. A batch size of 128 is used for the 13-channel dataset and 32 for the 3-channel dataset, selected as a balance between computational load and the requirements for the diversity loss.

This class-specific training procedure yields a set of specialized RVAE models, optimized for generating cross-subject SPD matrices characteristic of each motor imagery class by balancing several important aspects; accurate geometric reconstruction on both the SPD manifold and its tangent space, a well-regularized latent space for meaningful sampling, and generative diversity.

3.4 Synthetic Data Generation

Once the class-specific models are trained, they are employed to generate synthetic SPD covariance matrices. Two main strategies are considered for generation:

Posterior-Based Sampling: This approach leverages the existing training data to generate augmented samples. The process iterates through each real training matrix $\mathbf{X}_i$. Each matrix is passed individually through its corresponding class-specific trained encoder to obtain its latent parameters, $\boldsymbol{\mu}_i$ and $\log \boldsymbol{\sigma}_i^2$. The reparameterization trick (Eq. 3.5) is then applied. By introducing the random noise vector $\boldsymbol{\epsilon}_i$, this step samples a point $\mathbf{z}_i$ in the latent space that is close to the original sample's representation but includes stochastic variation. This single latent vector $\mathbf{z}_i$ is then passed through the decoder to produce a synthetic SPD matrix $\hat{\mathbf{X}}_i$. This method effectively creates augmented matrices that are variations of the original training samples, preserving their core characteristics while introducing plausible diversity.

Prior-Based Sampling: To generate entirely novel synthetic data, we begin by drawing a full batch of latent vectors, $\mathbf{Z}_{\text{prior}}$, directly from the Gaussian prior distribution $\mathbf{Z}_{\text{prior}} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. This entire batch of randomly sampled latent vectors is then passed through the trained decoder network of a specific class model in a single forward pass to generate a corresponding batch of new SPD matrices. This method explores the latent space to produce data points that do not directly correspond to any single original training sample but should still conform to the learned class distribution.

3.5 Evaluation

The evaluation framework assesses data performance in a cross-subject context using a Leave-One-Subject-Out Cross-Validation (LOSO-CV) protocol to test model generalization to unseen individuals. This evaluation encompasses classifier performance under various augmentation conditions and an intrinsic analysis of the generated synthetic data.

3.5.1 LOSO-CV Protocol

The Leave-One-Subject-Out Cross-Validation protocol is executed for each subject s_j in the dataset.

In each fold, subject s_j constitutes the test set, while data from all other subjects form the training set. The parallel transport alignment, as detailed in Section 3.1.2, is recomputed and fitted using only the current fold's training data. Both this training data and the test subject's data are then aligned using these fold-specific transport operators. Subsequently, the class-specific RVAE models are trained using only the aligned training data of the current fold. Following training, synthetic data is generated using these fold-trained models via two distinct strategies outlined in Section 3.4: a posterior-based approach creating augmented data at a 1:5 real-to-synthetic ratio, and a prior-based approach generating 5000 novel samples per class. The ratio for the posterior method was selected to substantially augment the training data, aiming to improve classifier robustness by densely sampling the learned local manifold around each real data point while mitigating the risk of overfitting to the original training set's characteristics. Conversely, the generation of 5000 novel samples via the prior ensures the synthetic-only dataset is sufficiently large to rigorously test the generative model's ability to capture the entire data distribution, allowing for fair classifier training without data scarcity as a confounding factor. Finally, classifiers are trained and tested, as detailed in the next Section 3.5.2. This entire process is iterated, with each subject serving as the test set once. Performance metrics are subsequently averaged across all folds.

3.5.2 Classifiers and Metrics

The impact of augmentation is assessed using three Riemannian geometry-aware classifiers from the `pyriemann` library (Barachant et al., 2025): Minimum Distance to Mean, K-Nearest Neighbors (KNN), and Support Vector Classifier. The MDM classifier utilizes the Affine-Invariant Riemannian metric (defined in Eq. 2.4) to assign samples to the class with the nearest Riemannian mean (defined in Eq. 2.5). The KNN classifier employs the same metric (AIRM) to find the k nearest neighbors, where $k = 5$. The SVC is adapted for Riemannian manifolds, using parameters including regularization parameter $C = 1.0$ and the Affine-Invariant Riemannian metric.

Within each LOSO-CV fold, these classifiers are trained and then evaluated on the held-out test subject's data under three distinct conditions. The first condition is the baseline, where training relies solely on the original aligned fold-specific training data (No Augmentation). The second condition involves training on the combined set of original and synthetic data for the fold (Augmented Data). The third condition uses only the synthetic data generated for the fold for training (Synthetic-Only Data), which helps to assess its standalone quality. Balanced Accuracy serves as the primary performance measure, averaged across all LOSO-CV folds to yield final scores for each condition. It is defined as the average of recall obtained on each class (Brodersen et al., 2010). This metric was chosen over conventional accuracy due to its robustness against potential class imbalances within evaluation folds. While the overall MI datasets are generally balanced, balanced accuracy was chosen to account for potential class imbalances that might arise, for instance, from artifact removal. Standard deviations are also reported to indicate performance consistency. While confusion matrices are generated for more detailed per-class analysis, the main comparisons in this study rely on balanced accuracy.

3.5.3 Synthetic Data Quality Analysis

In addition to classifier-based evaluation, the intrinsic quality of the generated synthetic SPD matrices is examined within each LOSO-CV fold. Generated matrices are first checked for symmetry and positive definiteness to verify their SPD properties; the number of matrices that deviate from these properties are reported. Secondly, a variance analysis is performed to compare the data spread of the original and synthetic datasets from two complementary perspectives: statistical variance and geometric diversity.

To assess statistical variance, each covariance matrix is first vectorized by flattening it into a one-dimensional array. On these vectors, the average variance is computed both globally across all samples and on a per-class basis. A more granular element-wise analysis compares the variance at each corresponding position within the covariance matrices.

To evaluate the geometric spread directly on the SPD manifold, the analysis calculates the mean pair-wise Riemannian distance for samples within each class, measuring how dispersed the data is in its native geometric space. Both variances are analyzed to ensure the generated data realistically reflects the overall geometric distribution on the manifold and the statistical variance of individual covariance elements.

Lastly, a scrambled label test is implemented as a diagnostic. Classifiers are trained on data with deliberately scrambled labels. Performance in this scenario is expected to be at chance level; consistent significant deviations might suggest that non-task-related characteristics are being learned or that the synthetic data exhibits spurious correlations.

These procedures collectively offer a comprehensive view of the augmentation's benefits and the synthetic data's geometric integrity. Results from these evaluations are aggregated for comparative analysis. The source code for the RVAE and the experiment results presented in this paper are publicly available on GitHub at `https://github.com/641e16/DA_RVAE`.

4 Results

This section presents the results of the model(s), focusing on the evaluation of the synthetic SPD matrices for data augmentation in a cross-subject context for MI-BCIs. The results presented are from experiments conducted on the 13-channel dataset, which was chosen as the representative example. The findings from the 3-channel dataset showed the practically the same performances as the original data and are therefore included in Appendix A.

This chapter presents the results in a logical progression, designed to first establish the validity of the generative model before testing its utility. It begins with the examination of the RVAE's training dynamics and the structure of its learned latent space. Following that, the fidelity of the generated synthetic data is thoroughly assessed, addressing the fundamental question of whether the model can produce valid and realistic covariance matrices. With the quality of the synthetic data established, the chapter proceeds to its final analysis: quantifying the impact of this data on cross-subject classification performance to determine the practical value of the proposed method. The proposed RVAE is finally compared to a standard VAE approach to confirm its benefit.

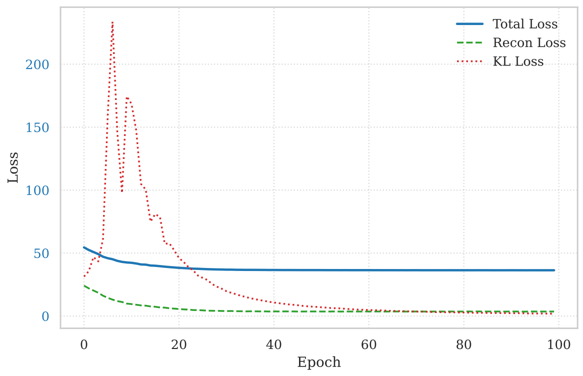


Figure 4.1: Representative training loss curves for a right-hand class RVAE (fold 1). The smooth convergence of the total loss demonstrates stable training.

4.1 VAE Model Training and Latent Space Characterization

The effectiveness of the generative model hinges on two key aspects: stable training and the emergence of a well-structured latent space. This section evaluates both.

4.1.1 Training Dynamics

The RVAE models were trained for a fixed 100 epochs. The incorporation of KL annealing shifts the model's focus from initial reconstruction fidelity to eventual latent space regularization. Furthermore, a diversity loss term was active during the entire training. The representative training curves in Figure 4.1 demonstrate the success of this strategy. The steady decrease of the total loss indicates stable convergence. The spike and subsequent decrease of the KL divergence, paired with a decreasing and then consistently low reconstruction loss, suggests the model first prioritized learning the data's structure for accurate reconstruction and then established a well-regularized latent space suitable for generative sampling.

4.1.2 Latent Space Structure

To assess the structure learned by the RVAE, the latent space is visualized using Uniform Manifold Approximation and Projection (UMAP) (McInnes et al., 2018).

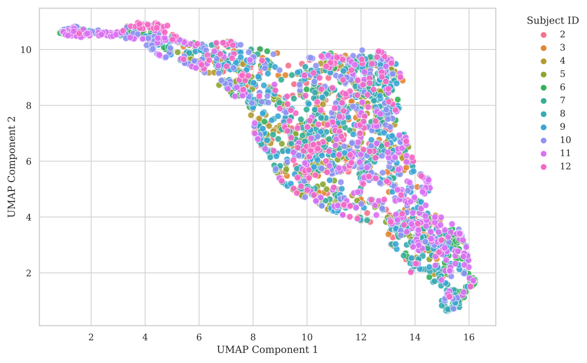


Figure 4.2: 2D UMAP visualization of the learned latent space for right-hand movement data, generated from the model trained with Subject 1 held out. The points are colored by Subject ID, and their significant overlap indicates the model learned a subject-invariant representation.

As shown in Figure 4.2, the latent codes organize into a distinct, elongated structure. Critically, when these points are colored by subject ID, they appear heavily intermingled. There are no large, isolated clusters that correspond to any single individual—instead, the data from all subjects are distributed throughout this shared structure. This arrangement of the latent space suggests that the RVAE has successfully achieved a representation that is largely subject-invariant—a primary objective for effective cross-subject generalization. This property, enabled by the parallel transport alignment preprocessing step, is highly relevant as it implies that generated data will reflect generalized patterns of motor imagery rather than the details of specific subjects. Such samples could therefore be more beneficial for augmentation in a LOSO-CV evaluation.

4.2 Fidelity Assessment of Generated Covariance Matrices

Before assessing the impact of the synthetic data on classifier performance in a cross-subject setting, it is essential to first evaluate its intrinsic quality. This involves verifying that the generated matrices adhere to the necessary geometric properties and that their statistical distributions are faithful to the original data.

4.2.1 SPD Property Verification

A fundamental requirement for the generated data is that each sample constitutes a valid SPD matrix. Across all LOSO-CV folds and for both prior- and posterior-based generation, 100% of the synthetic matrices successfully passed verification checks for symmetry and positive-definiteness. This confirms that the architectural constraints and numerical stabilization steps within the RVAE are effective. This adherence to the required geometric properties is a necessary prerequisite, establishing that the generated data is fundamentally valid and compatible with the Riemannian geometry-aware classifiers used in the subsequent analysis.

4.2.2 Variance Analysis

To quantitatively assess the fidelity of the synthetic data, its variance and geometric spread were compared against the original training data. The average variance ratios and geometric distances are presented in Table 4.1, with a detailed per-fold breakdown available in Appendix B Table B.1.

The fidelity analysis highlights a key trade-off between matching the data's statistical variance and its geometric diversity. The chosen γ hyperparameter value maintained a statistical variance ratio (synthetic/original) very close to 1.0. While the diversity loss term substantially improved the geometric diversity, initial generation still produced samples that were more concentrated than the original data. Increasing the γ value further to close the geometric diversity gap during training came at the cost of unacceptably inflating the statistical variance. Therefore, an optimal γ was chosen empirically—maximizing geometric diversity under the primary constraint of maintaining realistic statistical properties.

To bridge the remaining geometric diversity gap, a complementary approach was utilized by setting the diversity scale parameter to 2.2 during data generation. The parameter modifies the sampling process by directly scaling the random noise vector (ϵ_i) within the reparameterization equation 3.5. This offered a more direct way to modulate geometric diversity during data generation increasing the mean paired riemannian distance to 1.946 as seen in Table 4.1, allowing for synthetic data that more closely matches the original data on both metrics without altering the trained model's learned statistical distribution.

4.3 Data Augmentation Impact on Cross-Subject Classification

Having established the fidelity of the generated SPD matrices, we now assess their practical utility in a cross-subject classification task. This section details the effect of data augmentation on the balanced accuracy of the MDM, KNN, and SVC classifiers.

The statistical significance of the observed performance changes was assessed using a Wilcoxon signed-rank test (Wilcoxon, 1945). Suitable for comparing paired samples, the test is used on the baseline accuracy versus the accuracy achieved with data augmentation for each subject. To account for multiple comparisons across the different classifiers and generation methods, Bonferroni correction (Dunn, 1961) is applied. With six hypotheses being evaluated, the significance threshold for the p-value is adjusted to 0.0083, and any value below this corrected threshold is considered statistically significant.

4.3.1 Classification Performance

The main classification results are summarized in Table 4.2. The baseline performance, using only the original training data, yielded average accuracies of 59.52% for MDM, 53.19% for KNN, and 60.67% for SVC.

For the MDM classifier, performance was largely stable with all conditions showing only minor, non-significant fluctuations.

In contrast, data augmentation had a consistent and significant negative impact on the SVC across almost all conditions. Performance dropped by as much as 4.01% ($p = 0.002$) when trained on synthetic-only data from the posterior generator.

However, for the distance-based KNN classifier, data augmentation provided a clear and consistent benefit in every tested condition. The posterior generator produced the strongest results, with its synthetic-only data significantly increasing accuracy by 3.49 percentage points to 56.68% ($p = 0.002$) and its augmented set providing a significant gain of 2.45% ($p = 0.002$). The prior generator produced a similar positive trend; its synthetic-only data also achieved a significant improvement of 3.00% ($p < 0.001$) and an improvement of 2.19% ($p = 0.003$) on the augmented data underscoring the consistency of this improvement for KNN.

These divergent outcomes strongly suggest that the benefit of data augmentation depends on the dynamic between the generated data and the classifier's internal logic. With the synthetic data consistently being less diverse, with a lower mean distance between samples compared to the original data suggests that the RVAE is creating 'cleaner', more prototypical examples of each class, which are clustered more tightly around the geometric mean on the manifold. This lower diversity appears to help the local, distance-based KNN classifier, while hindering the margin-based SVC, which may benefit from more varied data to define a robust decision boundary. Data from the prior is sampled from a generalized latent distribution, likely tending to create centrally-located examples far from the decision boundary. The posterior, which augments existing training points, is constrained by the initial data's distribution and is less likely to generate novel samples in the sparsely populated boundary regions that SVC needs to define an effective decision boundary.

4.3.2 Generation Strategies Comparison

Both prior- and posterior-based sampling proved effective for augmenting data for the KNN classifier. The posterior generator led to slightly larger and more statistically significant improvements compared to the prior generator. For the SVC and MDM classifiers, both generation methods resulted in a similar degradation/stagnation of performance. The distribution of accuracy improvements across subjects for each classifier and generator is shown in Figure 4.3 and Figure 4.4. For the KNN classifier, the median improvement is consistently positive, whereas for SVC it is consistently negative.

4.3.3 Subject-wise Performance Analysis

While the average results are informative, the effect of augmentation varied across subjects. The subject-wise performance for each classifier is detailed in Appendix A (Figures B.1, B.2, B.3). For the KNN classifier (the best-performing case), aug-

Table 4.1: Fidelity analysis of synthetic data, averaged across all 12 LOSO-CV folds. The table compares the statistical variance ratio and the mean intra-class Riemannian distance between original and synthetic datasets for both prior and posterior generators.

Generator	Statistical Variance		Geometric Diversity	
	Original	Synthetic (Ratio)	Original	Synthetic
Prior	0.208	0.221 (1.061)	2.032	1.946
Posterior	0.208	0.221 (1.063)	2.032	1.918

Table 4.2: Average balanced accuracy (%) across 12 subjects (LOSO-CV folds). Results are shown for Baseline (no augmentation), Augmented (Original + Synthetic), and Synthetic-Only training conditions for both prior and posterior generators and the corresponding p-values.

Generator	Classifier	Baseline	Augmented Scenario			Synthetic-Only Scenario		
		Acc. (%)	Acc. (%)	Improvement	p-value	Acc. (%)	Improvement	p-value
Prior	MDM	59.52 ± 5.52	58.92 ± 5.40	-0.59%	0.092	58.36 ± 5.03	-1.16%	0.043
	KNN	53.19 ± 4.00	55.38 ± 4.17	+2.19%	**0.003**	56.19 ± 4.19	+3.00%	$<$ **0.001**
	SVC	60.67 ± 5.33	57.43 ± 6.32	-3.24%	0.016	56.75 ± 6.37	-3.92%	**0.002**
Posterior	MDM	59.52 ± 5.52	58.83 ± 5.29	-0.69%	0.092	58.95 ± 5.51	-0.57%	0.151
	KNN	53.19 ± 4.00	55.64 ± 4.13	+2.45%	**0.002**	56.68 ± 4.06	+3.49%	**0.002**
	SVC	60.67 ± 5.33	57.18 ± 6.57	-3.48%	**0.007**	56.66 ± 6.25	-4.01%	**0.002**

mentation provided substantial accuracy gains for some subjects (e.g. subject 1, 2, and 8), while for others the positive effect was small or slightly negative in very rare cases.

4.3.4 Scrambled Label Test

A diagnostic scrambled label test was conducted to ensure that classifiers were learning task-relevant features rather than artifacts of the data generation process. In this test, classifiers were trained on both original and synthetic data with randomly shuffled class labels. The resulting average balanced accuracies were consistently at chance level for all classifiers and conditions (e.g., prior generator synthetic data: MDM 50.7%, KNN 50.7%, SVC 50.6%). This confirms that the observed performance improvements are due to the model learning and generating meaningful, class-conditional data structures.

4.4 Standard VAE Comparison

To validate the necessity of the proposed Riemannian framework, a standard Euclidean VAE was trained as a comparative baseline. This model used an identical encoder/decoder architecture but operated directly on the vectorized form of the covariance matrices, without the geometric mappings (log and exp maps), geometric loss or constraints.

The standard VAE fundamentally failed to generate valid data. While all generated matrices were symmetric by construction, a substantial percentage in every fold were not positive-definite (e.g. over 40% in the first fold). This is a critical failure, as the output is not compatible with Riemannian classifiers, violating the primary requirement of the task.

Augmenting the training set with the portion of valid SPD data from the standard VAE was detrimental to classification. As shown in the results in Appendix B, it caused a statistically significant performance degradation for the MDM classifier (up to -9.49% with $p < 0.001$) and offered no significant benefit to the KNN or SVC classifiers. This stands in contrast to the RVAE, which provided a significant boost to KNN performance.

This provides clear evidence that a naive application of a VAE in the Euclidean space is insufficient. The geometric framework of the RVAE is not just beneficial but essential for both generating valid SPD matrices and for achieving effective data augmentation in this cross-subject context.

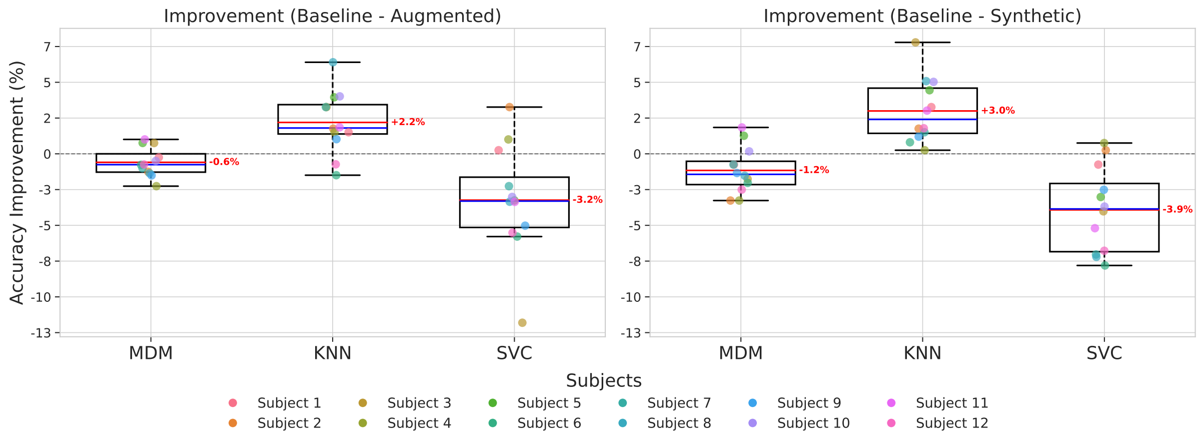


Figure 4.3: Distribution of accuracy improvement for each classifier using the prior generator. The plot shows the percentage point difference between the 'Augmented' and 'Synthetic-Only' conditions relative to the 'Baseline' across all subjects. The red line signifies the mean whilst the blue line is the median.

5 Conclusions

This thesis set out to investigate whether a novel Riemannian geometry-preserving Variational Autoencoder could generate high-fidelity EEG covariance matrices and if this data can improve cross-subject classification in MI-BCI. This final section directly addresses the core research questions, discusses the implications of the experimental results in the broader context of BCI research, and provides directions for future work.

5.1 Key Findings

The empirical results presented in the previous chapter can be narrowed down into two main areas: the fidelity of the generative model, and second, its subsequent impact on classification.

A key success was the generation of valid SPD matrices, a non-trivial task achieved through the initial alignment of training data to a common tangent space allowing for the use of logarithmic and exponential mapping in order to traverse between the manifold and the corresponding tangent space. The addition of the parallel transport alignment preprocessing technique enabled the RVAE to learn a subject-invariant latent space—a desired property for cross-subject generalization. The resulting synthetic data demonstrated high, although slightly inflated, statistical variance (a variance ratio of approx. 1.06), while its geometric diversity was somewhat reduced (a distance ratio of approx. 0.95), indicating that the synthetic samples were slightly more concentrated and prototypical than the more varied real-world data.

When this high-fidelity synthetic data was applied, the classification outcomes were highly divergent, underscoring that the utility of the data is not universal but is instead classifier-dependent. For the K-Nearest Neighbors classifier, the utilization of the generated data yielded a statistically significant improvement in balanced accuracy, boosting it by up to 3.49% ($p = 0.002$) using posterior sampling. Conversely, the same data provided no significant changes in performance for the Minimum Distance to Mean classifier and significantly hindered the performance of the Support Vector Classifier with its accuracy decreasing by as much as 4.01% ($p = 0.002$).

5.2 Discussion and Interpretation

The results of this study, particularly their dependency on the chosen classifier, offer several important insights into the nature of generative data augmentation using the RVAE on the SPD manifold for EEG covariance matrices.

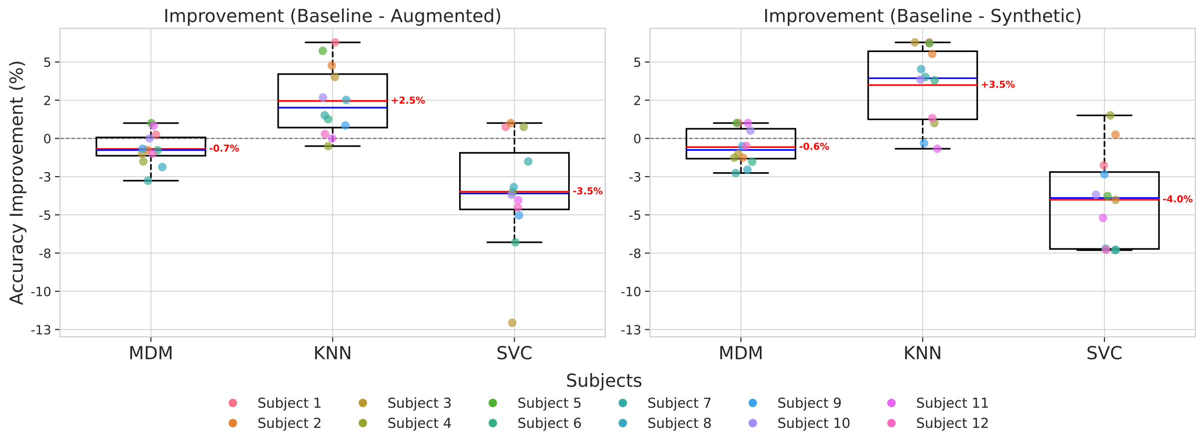


Figure 4.4: Distribution of accuracy improvement for each classifier using the posterior generator, showing similar trends to the prior generator but with more pronounced fluctuations.

5.2.1 Generative model capabilities

A primary contribution of this work is the confirmation that the RVAE framework is inherently capable of generating valid SPD matrices, a fundamental requirement that standard VAEs often fail to meet. This success is directly attributable to the underlying Riemannian geometry that enforces the SPD constraint by design. The geometric framework proved to be exceptionally well-suited for this task.

However, generating matrices that are merely valid is insufficient—they must also be faithful representations of the original data. This requires the model to learn the underlying geometry of the EEG covariance matrices to reconstruct them with high fidelity. Fidelity was assessed by comparing the statistical and geometric variance of the synthetic data against the original. The results indicate that by carefully tuning generation parameters and structuring the loss function, a high degree of fidelity in variance can be achieved.

The significance of this achievement is underscored when contrasted with a standard VAE, where even the valid portion of its generated output significantly worsened the MDM classifier. The fact that the RVAE-generated data led to performance improvements in some cases—and at least maintained baseline performance in others—is a promising indication of its utility. It demonstrates that the RVAE is not simply producing random SPD matrices; it is successfully learning and embedding meaningful manifold information from the training data into its generative process.

5.2.2 The Classifier-Dependent Nature of Augmentation

The divergent impact of augmentation on KNN and SVC classifiers strongly suggests that there is no universal solution for RVAE data augmentation in MI-BCIs. The effectiveness of synthetic data depends on not only the data generation technique but also the classifiers internal workings.

Given the results, it may be that the KNN classifier, which relies on local neighborhood density and geometric distance, benefits from the VAE-generated data for two reasons: firstly, because the data is more concentrated and prototypical, and secondly, because the sheer increase in sample size creates a much denser and more reliable neighborhood for classification. The synthetic samples may act as clean, prototypical exemplars that densify the center of the class manifold. Thus for a new test sample, this would creates a more well-defined local neighborhood of a single class, improving classification.

Conversely, the SVC, which seeks to find a maximum-margin separating hyperplane, may be obstructed by this lower diversity. The less varied synthetic data might cause the SVC to learn a decision boundary that is too tightly fitted to the center of the class distributions, making it brittle and less generalizable to real-world test samples that exhibit higher diversity.

The performance of the Minimum Distance to Mean classifier, which remained largely unchanged, provides another layer to this interpretation. As MDM classifies samples based on their Riemannian distance to the geometric mean of each class, a naive geometric model could easily "hack" this benchmark by simply learning to reproduce the Riemannian class means producing similar results. The fact that the RVAE maintained MDM performance while closely matching the variance characteristics of the original data is therefore significant. It stands in contrast to the standard VAE, whose generated data was detrimental, causing a performance drop of 9.49% ($p < 0.001$). Accordingly, the stable MDM performance is not a neutral result, but rather a positive confirmation of the generator's capabilities.

Due to this classifier-dependence, this work, therefore, refines the common goal of "generating realistic data" in BCI data augmentation to a more precise objective: generating data that complements the specific learning algorithm being used.

5.3 Future Research

This study provides a foundational proof of concept that opens avenues for future research. The evaluation was conducted on two datasets and a specific set of three geometry-aware classifiers. Building on this work, future research could proceed in several directions:

- **Advanced Latent Space Sampling:** The framework could be extended by incorporating more advanced sampling techniques. Rather than sampling from the prior, one could explore interpolation between latent codes of different subjects. Implementing a Riemannian Hamiltonian Variational Autoencoder (RHVAE) along with Riemannian random walks or Riemannian Hamiltonian Monte Carlo (RHMC) sampling may further capture a more faithful latent distribution for data generation as demonstrated in recent RHVAE data augmentation research for Magnetic Resonance Imaging (MRI) (Chadebec & Allassonnière, 2021; Chadebec et al., 2023).

- **Integrating Discriminative Frameworks:** A highly promising direction is to integrate the proposed geometric framework with architectures known for their strong discriminative power, such as vEEGNet (Zancanaro et al., 2024). While the RVAE creates a valid, subject-invariant space, vEEGNet has shown potential in learning latent spaces where MI classes are explicitly pushed apart. By incorporating the Riemannian architecture features (e.g custom Riemannian loss function and geometric mappings) into a vEEGNet-like architecture, a hybrid model for EEG covariance matrices could be created that is optimized to learn a latent space that is simultaneously geometrically valid, subject-invariant, and class-discriminative. This combination could lead to a significant performance increase beyond the improvements already achieved, fully leveraging the latent space for classification.

- **Classifier-Specific Augmentation:** A key insight from this work is the classifier-dependency. Future research could therefore focus on developing loss functions or data generation techniques for the VAE that are tailored to a specific classifier. For example, a targeted strategy for the SVC could be developed to generate class-specific samples along the boundary where the two classes meet—which may directly improve it's performance specifically.

5.4 Concluding Remarks

The presented work successfully developed and validated a novel Riemannian geometry-preserving VAE for EEG covariance matrix data augmentation in the challenging cross-subject MI-BCI context, revealing that the model is not only capable of consistently generating valid SPD matrices but also that the synthetic data can be used to maintain or even significantly improve classification performance for specific classifiers. This work shows that while generative modeling on the SPD manifold is a powerful and promising approach, its success is not universal. Its success hinges on the dynamic between the generation techniques and the learning mechanisms of the downstream classifier. This finding underscores the importance of developing data augmentation techniques in tandem with the specific classifiers they are intended to support.

References

Aristimunha, B., Carrara, I., Guetschel, P., Sedlar, S., Rodrigues, P., Sosulski, J., … Chevallier, S. (2023). *Mother of all BCI Benchmarks.* Retrieved from https://github.com/NeuroTechX/moabb doi: 10.5281/zenodo.10034223

Barachant, A., Barthélemy, Q., King, J.-R., Gramfort, A., Chevallier, S., Rodrigues, P. L. C., … Lebeurrier, A. (2025, February). *pyriemann.* Zenodo. Retrieved from https://doi.org/10.5281/zenodo.593816 doi: 10.5281/zenodo.593816

Blankertz, B., Dornhege, G., Krauledat, M., Müller, K. R., & Curio, G. (2007, aug). The non-invasive Berlin Brain-Computer Interface: fast acquisition of effective performance in untrained subjects. *NeuroImage*, *37*(2), 539–550. doi: 10.1016/j.neuroimage.2007.01.051

Bonci, A., Fiori, S., Higashi, H., Tanaka, T., & Verdini, F. (2021). An introductory tutorial on brain–computer interfaces and their applications. *Electronics*, *10*(560). Retrieved from https://doi.org/10.3390/electronics10050560 doi: 10.3390/electronics10050560

Bowman, S. R., Vilnis, L., Vinyals, O., Dai, A., Jozefowicz, R., & Bengio, S. (2016, August). Generating sentences from a continuous space. In S. Riezler & Y. Goldberg (Eds.), *Proceedings of the 20th SIGNLL conference on computational natural language learning* (pp. 10–21). Berlin, Germany: Association for Computational Linguistics. Retrieved from https://aclanthology.org/K16-1002/ doi: 10.18653/v1/K16-1002

Brodersen, K. H., Ong, C. S., Stephan, K. E., & Buhmann, J. M. (2010). The balanced accuracy and its posterior distribution. In *20th international conference on pattern recognition (icpr)* (pp. 3121–3124).

Cariello, S., Sanalitro, D., Micali, A., Buscarino, A., & Bucolo, M. (2023). Brain–computer-interface-based smart-home interface by leveraging motor imagery signals. *Inventions*, *8*(4), 91. doi: 10.3390/inventions8040091

Chadebec, C., & Allassonnière, S. (2021). Data augmentation with variational autoencoders and manifold sampling. In *Deep generative models, and data augmentation, labelling, and imperfections* (pp. 184–192). Springer.

Chadebec, C., Thibeau-Sutre, E., Burgos, N., & Allassonnière, S. (2023). Data augmentation in high dimensional low sample size setting using a geometry-based variational autoencoder. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *45*(3), 2879-2896. doi: 10.1109/TPAMI.2022.3185773

Chen, Y., Wiesel, A., Eldar, Y. C., & Hero, A. O. (2010). Shrinkage algorithms for mmse covariance estimation. *IEEE Transactions on Signal Processing*, *58*(10). Retrieved from http://dx.doi.org/10.1109/TSP.2010.2053029 doi: 10.1109/tsp.2010.2053029

Chevallier, S., Carrara, I., Aristimunha, B., Guetschel, P., Sedlar, S., Lopes, B., … Moreau, T. (2024). *The largest eeg-based bci reproducibility study for open science: the moabb benchmark.* doi: 10.48550/arXiv.2404.15319

Congedo, M., Barachant, A., & Bhatia, R. (2017). Riemannian geometry for EEG-based brain-computer interfaces; a primer and a review. *Brain-Computer Interfaces*, *4*(3), 155–174. doi: 10.1080/2326263X.2017.1297192

Dunn, O. J. (1961). Multiple comparisons among means. *Journal of the American Statistical Association*, *56*(293), 52–64. Retrieved 2025-07-31, from http://www.jstor.org/stable/2282330

Faller, J., Vidaurre, C., Solis-Escalante, T., Neuper, C., & Scherer, R. (2012). Autocalibration and recurrent adaptation: Towards a plug and play online erd-bci. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, *20*(3), 313-319. doi: 10.1109/TNSRE.2012.2189584

Fletcher, P. T., Lu, C., Pizer, S. M., & Joshi, S. (2004). Principal geodesic analysis for the study of nonlinear statistics of shape. *IEEE Transactions on Medical Imaging*, *23*(8), 995–1005. doi: 10.1109/TMI.2004.828353

Horn, R. A., & Johnson, C. R. (2013). *Matrix analysis* (2nd ed.). Cambridge University Press.

Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, *abs/1502.03167*. Retrieved from http://arxiv.org/abs/1502.03167

Kalunga, E., Chevallier, S., & Barthélemy, Q. (2015, June). Data augmentation in Riemannian space for Brain-Computer Interfaces. In *STAMLINS 2015 proceedings.* Lille, France. Retrieved from https://hal.science/hal-01351990

Kingma, D. P., & Welling, M. (2013). *Auto-encoding variational bayes.* Retrieved from https://arxiv.org/abs/1312.6114

Lang, S. (1995). *Differential and riemannian manifolds.* Springer.

Lee, J. M. (2018). *Introduction to riemannian manifolds* (Second ed.) (No. 176). Springer. doi: 10.1007/978-3-319-91755-9

Leeb, R., Lee, F., Keinrath, C., Scherer, R., Bischof, H., & Pfurtscheller, G. (2008, 01). Brain–computer communication: Motivation, aim, and impact of exploring a virtual apartment. *IEEE transactions on neural systems and rehabilitation engineering : a publication of the IEEE Engineering in Medicine and Biology Society*, *15*, 473-82. doi: 10.1109/TNSRE.2007.906956

Loshchilov, I., & Hutter, F. (2017). Fixing weight decay regularization in adam. *CoRR*, *abs/1711.05101*. Retrieved from http://arxiv.org/abs/1711.05101

Maas, A. L., Hannun, A. Y., & Ng, A. Y. (2013). Rectifier nonlinearities improve neural network acoustic models. In *Proceedings of the 30th international conference on machine learning* (Vol. 28).

McInnes, L., Healy, J., & Melville, J. (2018). Umap: Uniform manifold approximation and projection for dimension reduction. *Journal of Open Source Software*, *3*(29), 861. Retrieved from https://doi.org/10.21105/joss.00861 doi: 10.21105/joss.00861

Moakher, M. (2005). A differential geometric approach to the geometric mean of symmetric positive-definite matrices. *SIAM Journal on Matrix Analysis and Applications*, *26*(3), 735–747. doi: 10.1137/S0895479803436937

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., ... Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. In *Advances in neural information processing systems 32* (pp. 8024–8035).

Petersen, K. B., & Pedersen, M. S. (2008, October). *The matrix cookbook.* Technical University of Denmark. Retrieved from http://www2.imm.dtu.dk/pubdb/p.php?3274 (Version 20081110)

Prapas, G., Glavas, K., Tzallas, A. T., Tzimourta, K. D., Giannakeas, N., & Tsipouras, M. G. (2022, September). Motor imagery approach for BCI game development. In *2022 7th south-east europe design automation, computer engineering, computer networks and social media conference (SEEDA-CECNSM)* (p. 5). IEEE. doi: 10.1109/SEEDA-CECNSM57760.2022.9932937

Rommel, C., Paillard, J., Moreau, T., & Gramfort, A. (2022). Data augmentation for learning predictive models on EEG: a systematic comparison. *Journal of Neural Engineering*, *20*(2), 026002. doi: 10.1088/1741-2552/aca220

Schirrmeister, R. T., Springenberg, J. T., Fiederer, L. D. J., Glasstetter, M., Eggensperger, K., Tangermann, M., ... Ball, T. (2017). Deep learning with convolutional neural networks for brain mapping and decoding of movement-related information from the human EEG. *CoRR*, *abs/1703.05051*. Retrieved from http://arxiv.org/abs/1703.05051

Shao, H., Kumar, A., & Fletcher, P. T. (2017). The riemannian geometry of deep generative models. *CoRR*, *abs/1711.08014*. Retrieved from http://arxiv.org/abs/1711.08014

Wilcoxon, F. (1945). Individual comparisons by ranking methods. *Biometrics Bulletin*, *1*(6), 80–83. Retrieved 2025-07-31, from http://www.jstor.org/stable/3001968

Yair, O., Ben-Chen, M., & Talmon, R. (2019). Parallel transport on the cone manifold of SPD matrices for domain adaptation. *IEEE Transactions on Signal Processing*, *67*(7), 1797–1811. doi: 10.1109/TSP.2019.2894801

Yger, F., Berar, M., & Lotte, F. (2017). Riemannian approaches in brain-computer interfaces: A review. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, *25*(10), 1753-1762. doi: 10.1109/TNSRE.2016.2627016

Zancanaro, A., Cisotto, G., Zoppis, I., & Manzoni, S. L. (2024). veegnet: Learning latent representations to reconstruct eeg raw data via variational autoencoders. In M. Ziefle, M. D. Lozano, & M. Mulvenna (Eds.), *Information and communication technologies for ageing well and e-health* (pp. 114–129). Cham: Springer Nature Switzerland.

Zhu, H., Forenzo, D., & He, B. (2022, 08). On the deep learning models for eeg-based brain-computer interface using motor imagery. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, *PP*, 1-1. doi: 10.1109/TNSRE.2022.3198041

A 3-channel Dataset Additional Results

Table A.1: Average Classification Performance for the 3-Channel (Left/Right Hand) MI Dataset. The table shows balanced accuracy (mean % ± std. dev. %) and improvement percentages across 9 LOSO-CV folds. Due to the small dataset sample size, no p-values were calculated.

Generator	Classifier	Baseline	Augmented Scenario		Synthetic-Only Scenario	
		Acc. (%)	Acc. (%)	Improvement	Acc. (%)	Improvement
Prior	MDM	55.93 ± 5.02	55.89 ± 5.37	-0.05%	56.04	+0.11%
	KNN	52.95 ± 2.60	52.91 ± 3.13	-0.04%	53.45	+0.49%
	SVC	56.23 ± 5.15	56.20 ± 5.00	-0.03%	55.93	-0.30%
Posterior	MDM	55.93 ± 5.02	56.04 ± 5.67	+0.11%	55.88	-0.05%
	KNN	52.95 ± 2.60	52.43 ± 3.08	-0.53%	52.45	-0.50%
	SVC	56.23 ± 5.15	56.56 ± 5.16	+0.33%	56.55	+0.32%

Table A.2: Fidelity analysis of 3-channel synthetic data, averaged across all 9 LOSO-CV folds. The table compares the statistical variance ratio and the mean intra-class Riemannian distance between original and synthetic datasets for both prior and posterior generators.

Generator	Statistical Variance		Geometric Diversity	
	Original	Synthetic (Ratio)	Original	Synthetic
Prior	0.505	0.499 (0.9939)	0.2311	0.2224
Posterior	0.505	0.502 (0.9882)	0.2263	0.1862

B 13-channel Dataset Additional Results

Table B.1: Detailed Data Fidelity Metrics Per Fold. For each fold of the LOSO-CV, the table shows statistical variance metrics (Ratio, Original, and Synthetic) and the mean paired intra-class Riemannian distance for both original (O) and synthetic (S) data.

Generator	**Test Subject**	**Statistical Variance**			**Mean Intra-Class Geo. Dist.**	
		Ratio (S/O)	Original	Synthetic	Original (O)	Synthetic (S)
Prior	1	1.0637	0.208322	0.221585	2.0627	1.9276
	2	1.0663	0.206122	0.219777	2.0448	1.9372
	3	1.0677	0.206693	0.220679	2.0397	1.9286
	4	1.0685	0.207656	0.221883	2.0470	1.9402
	5	1.0620	0.207457	0.220319	2.0038	1.9208
	6	1.0525	0.209491	0.220500	2.0528	1.8538
	7	1.0577	0.207586	0.219572	2.0448	1.9201
	8	1.0599	0.209262	0.221792	2.0073	1.9786
	9	1.0580	0.205711	0.217637	2.0560	1.9855
	10	1.0624	0.211294	0.224470	2.0031	1.9806
	11	1.0543	0.208945	0.220293	2.0257	2.0313
	12	1.0570	0.211065	0.223096	1.9982	1.9442
	Average	**1.0608**	**0.208300**	**0.220967**	**2.0322**	**1.9457**
Posterior	1	1.0685	0.208322	0.222599	2.0660	1.9225
	2	1.0699	0.206122	0.220527	2.0689	1.8890
	3	1.0723	0.206693	0.221636	2.0592	1.8892
	4	1.0711	0.207656	0.222428	2.0640	1.9050
	5	1.0675	0.207457	0.221459	1.9222	1.7077
	6	1.0604	0.209491	0.222151	1.9910	1.8253
	7	1.0593	0.207586	0.219887	2.0570	1.9384
	8	1.0563	0.209262	0.221034	2.1603	2.0985
	9	1.0570	0.205711	0.217430	1.9434	1.9370
	10	1.0573	0.211294	0.223401	2.0750	2.0235
	11	1.0630	0.208945	0.222101	1.9539	1.9224
	12	1.0577	0.211065	0.223235	2.0169	1.9617
	Average	**1.0633**	**0.208300**	**0.221491**	**2.0315**	**1.9183**

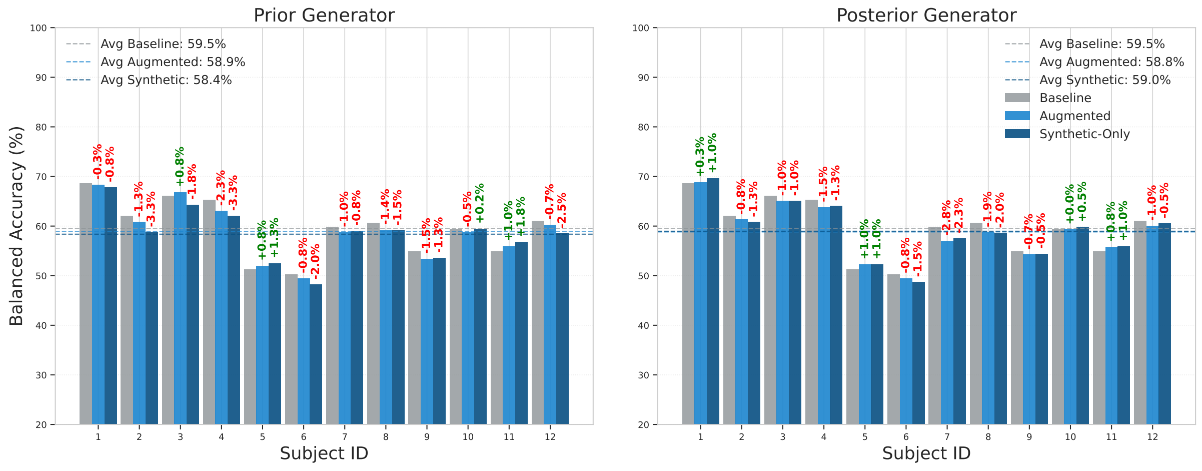


Figure B.1: Per-subject balanced accuracy for the MDM classifier under three conditions: Baseline (no augmentation), Augmented (original + synthetic), and Synthetic-Only, comparing the Prior (left) and Posterior (right) generators. Horizontal dashed lines indicate the average performance for each condition. The plot visually demonstrates the stable performance of the MDM classifier, with data augmentation resulting in only minor fluctuations in accuracy across most subjects.

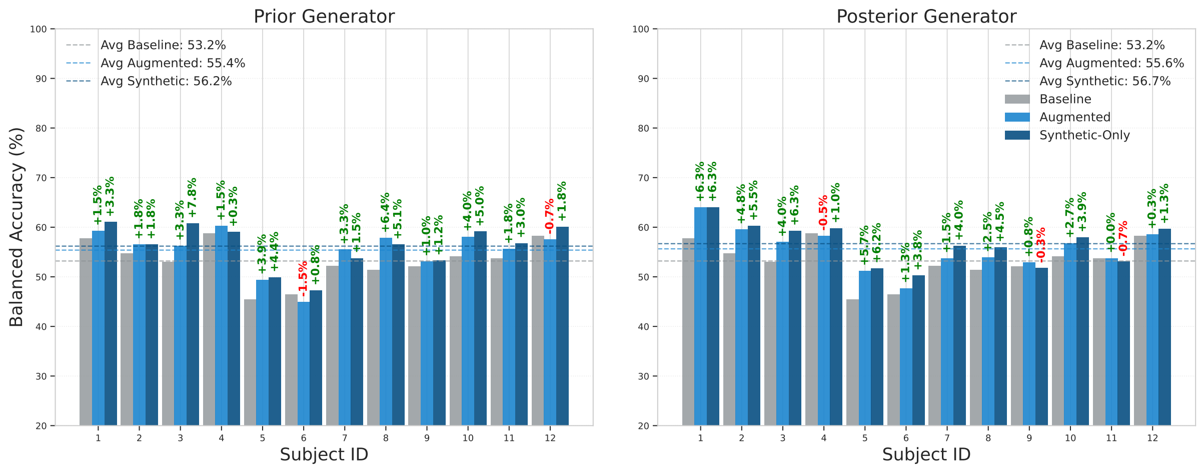


Figure B.2: Per-subject balanced accuracy for the KNN classifier under three conditions: Baseline (no augmentation), Augmented (original + synthetic), and Synthetic-Only, comparing the Prior (left) and Posterior (right) generators. Horizontal dashed lines indicate the average performance for each condition. The plot visually demonstrates the general trend of performance improvement for the KNN classifier, reaching as high as 7.8% for the third subject.

Table B.2: Average Classification Performance for the Standard VAE Baseline. The table shows balanced accuracy (mean % ± std. dev. %), improvement percentage, and the corresponding p-values from a Wilcoxon signed-rank test.

Generator	Classifier	Baseline	Augmented Scenario			Synthetic-Only Scenario		
		Acc. (%)	Acc. (%)	Improvement	p-value	Acc. (%)	Improvement	p-value
Prior	MDM	59.52 ± 5.52	51.08 ± 2.09	-8.43%	**0.001**	50.03 ± 0.20	-9.49%	$<$ **0.001**
	KNN	53.19 ± 4.00	54.75 ± 4.32	+1.56%	0.054	52.68 ± 2.92	-0.51%	0.970
	SVC	60.67 ± 5.33	59.60 ± 5.04	-1.07%	0.160	56.30 ± 5.00	-4.37%	**0.005**
Posterior	MDM	59.52 ± 5.52	50.97 ± 2.00	-8.55%	$<$ **0.001**	50.07 ± 0.24	-9.45%	$<$ **0.001**
	KNN	53.19 ± 4.00	54.14 ± 4.16	+0.95%	0.129	52.04 ± 2.19	-1.16%	0.339
	SVC	60.67 ± 5.33	59.28 ± 5.22	-1.39%	0.036	56.50 ± 4.82	-4.17%	0.009

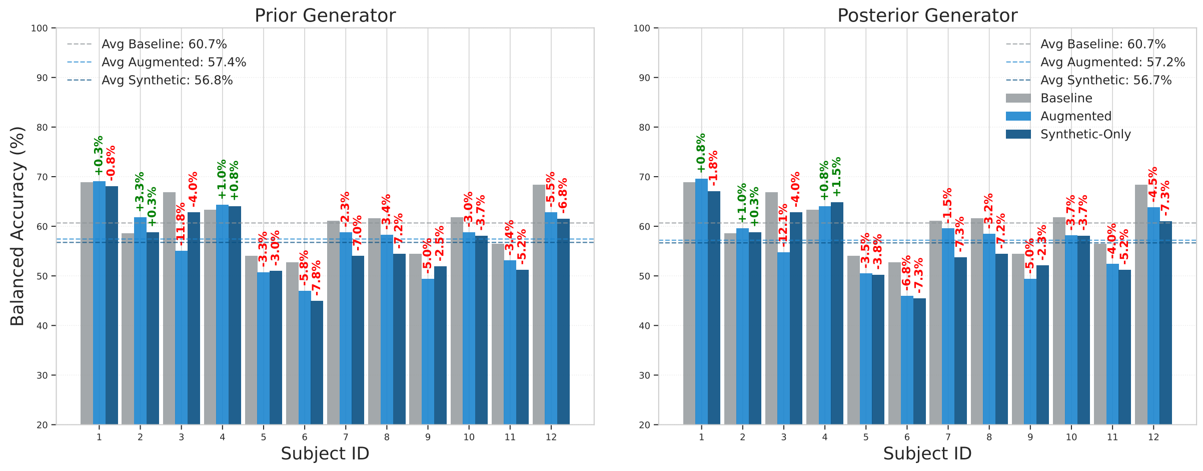


Figure B.3: Per-subject balanced accuracy for the SVC classifier under three conditions: Baseline (no augmentation), Augmented (original + synthetic), and Synthetic-Only. The charts compare results from the Prior Generator (left) and Posterior Generator (right). Horizontal dashed lines indicate the average performance across all subjects for each condition. The plot visually demonstrates the consistent, though variable in magnitude, degradation in SVC performance with data augmentation across most subjects.