Published in final edited form as:
KDD. 2023 August ; 2023: 1073–1085. doi:10.1145/3580305.3599483.

R-Mixup: Riemannian Mixup for Biological Networks

Xuan Kan[*],
Department of Computer Science, Emory University, Atlanta, GA, USA

Zimu Li[*],
Pritzker School of Molecular, Engineering, University of Chicago, Chicago, IL, USA

Hejie Cui,
Department of Computer Science, Emory University, Atlanta, GA, USA

Yue Yu,
School of Computational Science and Engineering, Georgia Institute of Technology, Atlanta, GA, USA

Ran Xu,
Department of Computer Science, Emory University, Atlanta, GA, USA

Shaojun Yu,
Department of Computer Science, Emory University, Atlanta, GA, USA

Zilong Zhang,
School of Statistics, University of International Business and Economics, Beijing, China

Ying Guo,
Department of Biostatistics and Bioinformatics, Emory University, Atlanta, GA, USA

Carl Yang[†]
Department of Computer Science, Emory University, Atlanta, GA, USA

Abstract

Biological networks are commonly used in biomedical and healthcare domains to effectively model the structure of complex biological systems with interactions linking biological entities. However, due to their characteristics of high dimensionality and low sample size, directly applying deep learning models on biological networks usually faces severe overfitting. In this work, we propose R-Mixup, a Mixup-based data augmentation technique that suits the symmetric positive definite (SPD) property of adjacency matrices from biological networks with optimized training efficiency. The interpolation process in R-Mixup leverages the log-Euclidean distance metrics from the Riemannian manifold, effectively addressing the *swelling effect* and *arbitrarily incorrect label* issues of vanilla Mixup. We demonstrate the effectiveness of R-Mixup with five real-world

[†]To whom correspondence should be addressed. j.carlyang@emory.edu.
[*]Both authors contributed equally to this research.

biological network datasets on both regression and classification tasks. Besides, we derive a commonly ignored necessary condition for identifying the SPD matrices of biological networks and empirically study its influence on the model performance. The code implementation can be found in Appendix E.

Keywords

biological network; data augmentation; geometric deep learning

1 INTRODUCTION

As a ubiquitous type of data in biomedical studies, biological networks are used to depict a complex system with a set of interactions between various biological entities. For example, in a brain network, the correlations extracted from functional Magnetic Resonance Imaging (fMRI) are modeled as interactions among human-divided brain regions [16, 44, 45, 51, 62, 75, 76, 83]. Meanwhile, in a co-expression gene-protein network, interactions are built to discover disease genes and potential modules for clinical intervention [66]. There are diverse ways to define the connections among entities in biological networks, such as interactions [60, 97], reactions [23], and relations [25–27, 52, 89]. One of the most widespread practices is calculating the covariance and correlation among entities to summarize and quantify interactions [8, 32, 75, 81, 82, 96]. Therefore, developing powerful computational methods to predict disease outcomes based on profiling datasets from such correlation matrices has attracted great interest from biologists [1, 58, 59, 61, 90, 91, 98].

Deep learning methods have achieved state-of-the-art performance in various downstream applications [21, 54], especially when the training sample size is large enough. However, biological network datasets often suffer from limited samples due to the complicated and expensive collection and annotation processes of scientific data [13, 88, 102]. Another key property of biological networks is that the dimension of such networks is typically very high, i.e., $O(n^2)$ correlation edges among n entities. Therefore, directly applying Deep Neural Networks (DNNs) to such biological network datasets can easily cause severe overfitting [2, 29, 87, 92, 99].

Mixup is a widely used data augmentation technique that can improve the model performance by linearly interpolating new samples from pairs of existing instances [101]. In the scenario of biological network analysis, since the node identities and their corresponding order are usually fixed across network samples within the same dataset [53], the Mixup technique can be easily applied via linear interpolation. Empirically, Figure 1 (a) and (b) compare the processes of training a transformer model [53] without Mixup and with the vanilla Mixup (V-Mixup) [101] technique on the brain network dataset from the PNC studies [73] to perform binary classification. In Figure 1 (a), the training loss without the Mixup technique diminishes quickly while the test loss continues to increase, which apparently indicates a severe overfitting problem. In contrast, in Figure 1 (b) with V-Mixup, the training process becomes more stable, and the model achieves higher performance with a lower test loss, even though the training loss is relatively high.

Although the vanilla Mixup can mitigate the overfitting issue for biological networks, there are two critical limitations in existing Mixup methods. The first noticeable issue is that the linear Mixup of correlation matrices in the Euclidean space would cause a *swelling effect*, where the determinant of the interpolated matrix is larger than any of the original ones. The inflated determinant, which equals the product of eigenvalues, also indicates an increase in eigenvalues. This can be interpreted as exaggerated variances of the data points in the principal eigen-directions. As a result, an unphysical augmentation from original data is generated, which may change the characteristics, e.g., the correlations of different brain functional areas, of the original dataset and violate the intuition of linear interpolations that the determinant from a mixed sample should be intuitively *between* the original pair of samples [15, 31, 33, 69]. On the other hand, the vanilla Mixup cannot properly handle regression tasks due to *arbitrarily incorrect label* [93], which means that linearly interpolating a pair of examples and their corresponding labels cannot ensure that the synthetic sample is paired with the correct label. Although several existing works like RegMix [47] and C-Mixup [93] have attempted to avoid this issue by restricting the mixing process only to samples with a similar label, their practice leads to less various sample generation and weakens the ability of Mixup towards improving the robustness and generalization of deep neural network models.

Recently, investigating covariance and correlation matrices in the view of symmetric positive definite matrices (SPD) with Riemannian manifold has demonstrated impressive advantages in biological domains [4, 67, 95], which helps to improve the model performance and capture informative sample features. Inspired by these studies, we pinpoint a promising direction to mitigate these two identified issues when adapting the Mixup technique for biological networks from the perspective of SPD analysis. However, existing works that leverage the Riemannian manifold for SPD analysis of biological networks often directly treat covariance and correlation matrices as SPD matrices without rigorous verification. We clarify that covariance and correlation matrices are not equal to SPD matrices: a necessary condition for the covariance and correlation matrices generated from a sample $X \in \mathbb{R}^{n \times t}$ to be positive definite is that *the sequence length t is no less than the sample variable number n*. We provide theoretical proof for this condition in Appendix A. The collection of positive definite matrices mathematically forms a unique geometric structure called the *Riemannian manifold*, which generalizes curves and surfaces to higher dimensional objects [35, 57, 72]. From a mathematical perspective, augmenting samples along geodesics on the manifold of SPDs with the log-Euclidean metric effectively (a) preserves the intrinsic geometric structure of the original data and eases the *arbitrarily incorrect label* and (b) eliminates the *swelling effect* as is shown in Figure 2. The advantages are further proved theoretically in Section 3.

Based on this insight, we propose R-Mixup, a Mixup-based data augmentation approach for SPD matrices of biological networks, which augments samples based on Riemannian geodesics (i.e., Eq.(2)) instead of straight lines (i.e., Eq.(1)). We theoretically analyze the advantages of R-Mixup by incorporating tools from differential geometry, probability and information theory. Besides, a simple and efficient preprocess optimization is proposed to reduce the actual training time of R-Mixup considering the costly eigenvalue decomposition operation. Sufficient experiments on five datasets spanning both regression and classification tasks demonstrate the superior performance and generalization ability of R-Mixup. For regression tasks, R-Mixup can achieve the best performance based on the same random sampling strategy as vanilla Mixup, demonstrating its ability to overcome the *arbitrarily incorrect label* issue by adequately leveraging the intrinsic geometric structure of SPD. This advantage is also proved by a case study in Appendix G. Furthermore, we observe that the performance gain of R-Mixup over existing methods is especially prominent when the annotated samples are extremely scarce, verifying its practical advantage under the low-resource settings.

We summarize the contributions of this work as three folds:

- We propose R-Mixup, a data augmentation method for SPD matrices in biological networks, which leverages the intrinsic geometric structure of the dataset and resolves the *swelling effect* and *arbitrarily incorrect label* issues. Different Riemannian metrics on manifold are compared, and the effectiveness of R-Mixup is theoretically proved from the perspective of statistics. We also proposed a pre-computing optimization step to reduce the burden from eigenvalue decomposition.
- Thorough empirical studies are conducted on five real-world biological network datasets, demonstrating the superior performance of R-Mixup on both regression and classification tasks. Experiments on low-resource settings further stress its practical benefits for biological applications often with limited annotation.
- We emphasize a commonly ignored necessary condition for viewing covariance and correlation matrices as SPD matrices. We believe the clarification of this pre-requirement for applying SPD analysis can enhance the rigor of future studies.

2 RELATED WORK

2.1 Mixup for Data Augmentation

Mixup is a simple but effective principle to construct new training samples for image data by linear interpolating input pairs and forcing the DNNs to behave linearly in-between training examples [101]. Many follow-up works extend Mixup from different perspectives. For example, [79, 80] interpolate training data in the feature space, [20, 37] learn the mixing ratio for Mixup to alleviate the under-confidence issue for predictions. Besides, [28, 48, 93, 104] strategically select the sample pairs for Mixup to prevent low-quality mixing examples and produce more reasonable augmented data. To further improve the quality of the augmented data, [42, 56, 100] create mixed examples by only interpolating a specific region (often most salient ones) of examples. Mixup has also been extended to other data modalities such as text [17, 103] and audio [63]. There are several attempts to study Mixup on non-Euclidean data, graphs, like NodeMixup [84], GraphMixup [85] and G-Mixup [39]. However, less attention has been paid to adapting Mixup for graphs from a manifold perspective, which is the focus of this study.

2.2 Geometric Deep Learning

Geometric deep learning aims to adapt commonly used deep network architectures from euclidean data to non-euclidean data, such as graphs and manifolds, with a broad spectrum of applications from the domains of radar data processing [10], graph analysis [3, 64], image and video processing [14, 41, 46, 64], and Brain-Computer Interfaces [67, 77]. For example, SPDNet [46] builds a Riemannian neural network architecture with special convolution-like layers, rectified linear units (ReLU)-like layers, and modified backpropagation operations for the non-linear learning of SPD matrices. ManifoldNet [14] defines the analog of convolution operations for manifold-valued data. MoNet [64] generalizes CNN architectures to the non-Euclidean domain with pseudo-coordinates and weight functions. [10] designs a Riemannian batch normalization for SPD matrices by leveraging geometric operations on the Riemannian manifold. MAtt [67] proposes the manifold attention mechanism to represent spatiotemporal representations of EEG data. Though widely recognized as being effective for images, tabular and graph data, to the best of our knowledge, data augmentation methods in geometric deep learning have rarely been explored.

3 R-MIXUP

In this section, we first provide some preliminary facts, including a necessary condition for treating covariance and correlation matrices as SPD matrices. Next, we elaborate on the detailed process of applying R-Mixup for data augmentation, compare possible mathematical metrics designs, and finally provide the theoretical analysis of the advantages of using R-Mixup.

3.1 Notations and Preliminary Results

Given n variables of biological entities, we extract a t length sequence for each variable and compose the input sequences $X \in \mathbb{R}^{n \times t}$. The correlation matrix or biological network $S = \operatorname{Cor}(X) \in \mathbb{R}^{n \times n}$ is obtained by taking the pairwise correlation among each pair of the biological variables. The value y is the network-level prediction label for the prediction task.

Definition 3.1. A symmetric $n \times n$ matrix S is *positive semi-definite* if for any vector $u \in \mathbb{R}^n, u^T S u \geq 0$. Equivalently, this means that the eigenvalues of S are all nonnegative. If the inequality holds strictly, S is said to be *positive definite*, or *symmetric positive definite*, or SPD for short.

Let $\operatorname{Sym}(n)$ be the collection of all positive semi-definite matrices, and $\operatorname{Sym}^+(n)$ denotes the collection of all SPDs. The collection $\operatorname{Sym}(n)$ can be seen as an $\frac{1}{2}n(n-1)$-dimensional Euclidean space, but $\operatorname{Sym}^+(n) \subset \mathbb{R}^{n \times n}$ admits a more general structure call *manifold* in differential geometry which resembles the Euclidean space in its local regions. To set up the modeling on the manifold $\operatorname{Sym}^+(n)$, the covariance matrix $\operatorname{Cor}(X)$ for the input X should be positive definite. However, it is worth mentioning that previous studies that use the Riemannian manifold for analyzing biological networks often treat covariance and correlation matrices as SPD without proper validation. Towards this common negligence, we bring out the following basic fact:

Proposition 3.2. Covariance and correlation matrices are positive semi-definite. A necessary condition for them to be positive definite is that the sample length is no less than the variable number, i.e., $t \geq n$.

This proposition indicates that covariance and correlation matrices only have the opportunity to be positive definite when $t \geq n$. The detailed proof can be found in Appendix A. This is the case for the datasets involved in this study, where most of the correlation matrices are SPD. The few exceptions would have very few zero eigenvalues, which we manually set as 10^{-6} to eliminate their influence. More discussions on adjusting correlation matrices to be SPDs can be found in [22, 43].

3.2 R-Mixup Deduction

In this section, we explain on the detailed process of R-Mixup for SPD matrice augmentation. Let S_i, S_j represent two different correlation matrices constructed based on X_i, X_j. In the vanilla Mixup [101], the augmented samples $(\widetilde{S}, \widetilde{y})$ are created through the straight line connecting S_i, S_j and y_i, y_j,

$$\widetilde{S} = (1-\lambda)S_i + \lambda S_j,$$
$$\widetilde{y} = (1-\lambda)y_i + \lambda y_j,$$

where $\lambda \sim \mathrm{Beta}(\alpha, \alpha)$, Beta is the Beta distribution, given $\alpha \in (0, \infty)$.

To facilitate the illustration of R-Mixup in geometry notions, we briefly introduce the main concepts here while more detailed explanations can be found in [35, 57]. To define R-Mixup, we replace Eq.(1) by a certain Riemannian geodesics. *Geodesics* are the generalization of *straight lines* in the Euclidean space, which is intuitively the shortest path between two given points on Riemannian manifolds. Riemannian manifolds (M, g) are manifolds M equipped with *Riemannian metrics* g which measure distances between points in the manifold and induces geodesic equations [35, 57]. It is generally hard to solve geodesics equations in the simple analytical form as straight lines, however, for $\mathrm{Sym}^+(n)$, there are lots of well-defined choices of Riemannian metrics with known geodesics [31, 35, 50, 69], and we employ the *log-Euclidean metric* with the following geodesic:

$$\widetilde{S} = \exp((1-\lambda)\log S_i + \lambda \log S_j), \tag{2}$$

where exp, log are matrix exponential and logarithm. Figure 3 sketches the geodesic as the purple dotted curve and a rigorous deduction of Eq.(2) can be found in [35]. Implementation of the matrix exponential for positive definite matrix S is straightforward: by basic linear algebra,

$$S = O\mathrm{diag}\,(\mu_1, \ldots, \mu_n)O^T, \tag{3}$$

where O is an orthogonal matrix with μ_i being eigenvalues of S. Then by definition,

$$\begin{aligned} \exp S &= O\mathrm{diag}\,(\exp \mu_1, \ldots, \exp \mu_n)O^T, \\ \log S &= O\mathrm{diag}\,(\log \mu_1, \ldots, \log \mu_n)O^T. \end{aligned} \tag{4}$$

3.3 Comparison with Other Metrics

There are various choices of Riemannian metrics and hence different geodesics on $\mathrm{Sym}^+(n)$ [7, 31, 35, 50, 69], such as the Cholesky metric defined by Cholesky decompositions L_i of positive definite matrices S_i, the well-known Affine-invariant metrics on $\mathrm{Sym}^+(n)$ [78], and the Bures-Wasserstein studied in statistics and information theory [6, 7]. We compare the most popular ones with the proposed log-Euclidean for mixing up biological networks on prediction tasks. The comparisons are summarized in Table 1. To be specific, different geodesics are analyzed from two perspectives: (*a*) whether it causes the swelling effect, (*b*) whether it is numerically stable on our dataset.

Swelling Effect.—The detailed definition and rigorous proof of the swelling effect can be found in Section 3.4 and Appendix B. As exemplified by the motivation in Figure 2, Euclidean, Cholesky, and Bures-Wasserstein metrics evidently suffer from the swelling effect.

Numerical Stability.—Augmenting matrices from the geodesic with the Affine-invariant metric requires the computation of $S_i^{-1/2}$ and hence the calculation of the inverse square root of its eigenvalues as we define matrix exponential and logarithm in Eq.(4). For SPDs with small eigenvalues μ, such computations may not be numerically stable since $\mu^{-1/2} \to \infty$. Furthermore, with awareness to the following limit relation:

$$\lim_{\mu \to 0} \frac{\log \mu}{\mu^{-1/2}} = 0, \tag{5}$$

which indicates that $\log \mu \ll \mu^{-1/2}$ for small μ, we know that computing matrix logarithm when using log-Euclidean metric should be more stable. Similarly, for Bures-Wasserstein geodesics, to compute $(S_iS_j)^{1/2}$, we notice the following fact:

$$S_iS_j = S_i^{1/2}\left(S_i^{-1/2}(S_iS_j)S_i^{1/2}\right)S_i^{-1/2} = S_i^{1/2}\left(S_i^{1/2}S_jS_i^{1/2}\right)S_i^{-1/2}. \tag{6}$$

Thus,

$$(S_i S_j)^{1/2} = S_i^{1/2}\left(S_i^{1/2} S_j S_i^{1/2}\right)^{1/2} S_i^{-1/2}, \tag{7}$$

where the undesirable $\mu^{-1/2}$ appears again in the calculation.

Considering these two points, we stick with log-Euclidean metric. Experimental results in Section 4.2 further showcase the effectiveness of this choice.

3.4 R-Mixup Theoretical Justification

Using geodesics when conducting data augmentation demonstrates unique advantages over straight lines. The first advantage is that R-Mixup will not cause the *swelling effect* which exaggerates the determinant and certain eigenvectors of the samples as discussed in Section 1 and 3.3. Mathematically, suppose $\det S_i \le \det S_j$, then the determinant of $\widetilde{S}$ defined by Eq. (2) satisfies:

$$\det S_i \le \det \widetilde{S} \le \det S_j\,. \tag{8}$$

Detailed proof can be found in Appendix B.

The second advantage is that, by leveraging the manifold structure, we can fit better estimators compared with linear interpolation in the Euclidean space. To be precise, as illustrated after Proposition 3.2, our samples are distributed over $\mathrm{Sym}^+(n)$ rather than the whole ambient Euclidean space $\mathbb{R}^{n \times n}$, which is accepted as a *prior knowledge* in the sense of Bayesian modeling fitting. Then the purpose of implementing R-Mixup becomes clear: we augment the non-trivial geometric information for the learning architectures later used in our experiments as an analogy to transforming images to enhance the translation and rotational invariance before a training of image identification [18]. We theoretically justify this point from the perspective of both statistics on Riemannian manifolds [9, 30, 49, 50, 55] and information theory [12, 65, 70].

Specifically, we treat the data augmentation process as a regression conducted on the manifold $\mathrm{Sym}^+(n)$ which is explicitly constrained by its geometric structure and based on the distribution of the dataset as the prior knowledge. Given any $\widetilde{S}$, let $\widetilde{m}(\widetilde{S})$ denote the *estimator/prediction function* of the regression whose analytical form depends on the concrete regression methods. We take geodesic regression and kernel regression [9, 34, 74] on $\mathrm{Sym}^+(n)$ to address the problem. Roughly speaking, geodesic regression generalizes multi-linear regression on Euclidean space to manifold with the Euclidean distance being replaced by Riemannian metric. Kernel regression embeds data into higher dimensional feature space with kernel functions K to grasp more non-linear relationship of the dataset.

Since the exact distribution of augmented data is unknown, we follow the common practice [19, 40, 74] and apply *Gauss kernel*

$$K_E(S_i, \widetilde{S}) = \frac{1}{(2\pi\sigma^2)^{\frac{n}{2}}} \exp\left(-\frac{1}{2\sigma^2} \| S_i - \hat{S} \|^2\right), \tag{9}$$

which possess the *universal property* to approximate any continuous bounded function in principle. However, the Gauss kernel K_E is defined on the Euclidean space which *unreasonably* implies nonzero density of samples outside $\mathrm{Sym}^+(n)$ contradicting the prior knowledge. To remedy the problem, we introduce a method from the *heat kernel theory* in differential geometry [5, 35] to generalize K_E to

$$K_R(S_i, \hat{S}) = \frac{1}{(2\pi\sigma^2)^{\frac{n(n-1)}{4}}} \exp\left(-\frac{1}{2\sigma^2} d(S_i, \hat{S})^2\right), \tag{10}$$

with

$$d(S_i, S_j) = \| \log S_i - \log S_j \| \tag{11}$$

being the *Riemannian distances function* on $\mathrm{Sym}^+(n)$. Then we prove in details in the Appendix C.

Theorem 3.3. For $Sym^+(n)$ with log-Euclidean metric, comparing R-Mixup with estimators $\widetilde{m}$ obtained by regressions with respect to the manifold structure, the square loss for augmented data $\widetilde{S}$ from Riemannian geodesics Eq.(2) is no more than those $\widetilde{S}'$ from straight lines Eq.(1):

$$\sum \left(\widetilde{m}(\widetilde{S}) - \widetilde{y}\right)^2 \le \left(\widetilde{m}(\widetilde{S}') - \widetilde{y}\right)^2. \tag{12}$$

A less empirical loss from regression on manifold is recognized as an evidence that R-Mixup captures some geometric features of $\mathrm{Sym}^+(n)$, thereby providing the learning algorithm an opportunity to learn this feature. Finally, the proposed R-Mixup is formally defined as

$$\begin{aligned} \widetilde{S} &= \exp((1-\lambda)\log S_i + \lambda \log S_j), \\ \widetilde{y} &= (1-\lambda)y_i + \lambda y_j, \end{aligned} \tag{13}$$

where $\lambda \sim \text{Beta}(\alpha, \alpha)$, for $\alpha \in (0, \infty)$.

3.5 Time Complexity and Optimization

One potential concern of the proposed R-Mixup lies in its timeconsuming operations of the eigenvalue decomposition and matrix multiplication (with the time complexity of $O(n^3)$), which dominate the overall running time of R-Mixup. In practice, we find that most common modern deep learning frameworks such as PyTorch [68] have been optimized for accelerating matrix multiplication. Thus, the main extra time consumption of the R-Mixup is the exp and log operations of the three eigenvalue decompositions. We propose a sample strategy to optimize the running time of R-Mixup by precomputing the eigenvalue decomposition and saving the orthogonal matrix O and eigenvalues $\{\mu_1, \ldots, \mu_n\}$ of each sample. This precomputing process can reduce the three computations of eigenvalue decomposition to once for each sample. Formally,

$$\widetilde{S} = \exp\left((1-\lambda)O_i \text{diag}(\log \mu_1, \ldots, \log \mu_n)O_i^T + \lambda O_j \text{diag}(\log v_1, \ldots, \log v_n)O_j^T\right), \tag{14}$$

where $O_i \text{diag}(\mu_1, \ldots, \mu_n)O_i^T$ and $S_j = O_j \text{diag}(v_1, \ldots, v_n)O_j^T$ are the eigenvalue decompositions of S_i and S_j, respectively. The efficiency of this optimization is further discussed in Section 4.4.

4 EXPERIMENTS

We evaluate the performance of R-Mixup comprehensively on real-world biological network datasets with five tasks spanning classification and regression. The dataset statistics are summarized in Table 2. The empirical studies aim to answer the following three research questions:

- RQ1: How does R-Mixup perform compared with existing data augmentation strategies on biological networks with various sample sizes on different downstream tasks?
- RQ2: How does the sequence length of each sample affect the characteristics of correlation matrices and consequently the choice of augmentation strategies?
- RQ3: Is R-Mixup efficient in the training process and robust to hyperparameter changes?

4.1 Experimental Setup

4.1.1 Datasets and Tasks.

Adolescent Brain Cognitive Development Study (ABCD).: The dataset used in this study is one of the largest publicly available fMRI datasets, with access restricted by a strict data requesting process [13]. From this dataset, we define two tasks: *BioGender Prediction* and *Cognition Summary Score Prediction*. The data used in the experiments are fully anonymized brain networks based on the HCP 360 ROI atlas [36] with only biological sex labels or cognition summary scores. BioGender Prediction is a binary classification problem, which includes 7901 subjects after the quality control process, with 3961 (50.1%)

females among them. Cognition Summary Score Prediction is a regression task whose label is Cognition Total Composite Score containing seven computer-based instruments assessing five cognitive sub-domains: Language, Executive Function, Episodic Memory, Processing Speed, and Working Memory, ranging from 44.0 to 117.0.

Autism Brain Imaging Data Exchange (ABIDE).: The dataset includes anonymous resting-state functional magnetic resonance imaging (rs-fMRI) data from 17 international sites [11]. It includes brain networks from 1009 subjects, with a majority of 516 (51.14%) being patients diagnosed with Autism Spectrum Disorder (ASD). The task is to perform the binary classification for ASD diagnosis. The region definition is based on Craddock 200 atlas [24]. Given the blood-oxygen-level-dependent (BOLD) signal length of the samples in this dataset is 100, which reflects whether neurons are active or reactive, we randomly select 100 nodes to satisfy the necessary condition discussed in Proposition 3.2 for SPD matrices.

Philadelphia Neuroimaging Cohort (PNC).: The dataset is a collaborative project from the Brain Behavior Laboratory at the University of Pennsylvania and the Children's Hospital of Philadelphia. It includes a population-based sample of individuals aged 8–21 years [73]. After the quality control, 503 subjects were included in our analysis. Among these subjects, 289 (57.46%) are female. In the resulting data, each sample contains 264 nodes with time series data collected through 120 timesteps. Hence, we randomly select 120 nodes to satisfy the necessary condition mentioned in Proposition 3.2 for treating generated correlation matrices as SPD. BioGender Prediction is used as the downstream task.

TCGA-Cancer Transcriptome.: The Cancer Genome Atlas (TCGA) dataset is a large-scale collection of multi-omics data from over 20,000 primary cancer and matched normal bio-samples spanning 33 cancer types. In this study, we select non-redundant cancer subjects with gene expression data and valid clinical information. The gene expression data is normalized, and the top 50 highly variable genes (HVG) are selected as the nodes for network construction. The subjects are then assigned to different samples based on their cancer subtype. The final dataset consists of 459 subjects from 66 cancer subtypes. We extract 240 correlation matrices from these subjects with 24 cancer types, each type includes ten samples, and each sample contains 50 nodes. The downstream task of this study is to predict cancer subtypes based on the HVG expression network.

4.1.2 Metrics.—For binary classification tasks on datasets ABCD-BioGender, PNC, and ABIDE, we adopt AUROC and accuracy for a fair performance comparison. The classification threshold is set as 0.5. For the regression task on ABCD-Cog, the mean square error (MSE) is used to reflect model performance. For the multiple class classification task on TCGA-Cancer, since it contains 24 classes and each class has a balanced sample size, we take the macro Precision and macro Recall so that all classes are treated equally to reflect the overall performance. All the reported results are based on the average of five runs using different random seeds.

4.1.3 Implementation Details.—We equip the proposed R-Mixup with two most popular deep backbone models for biological networks, Transformer [53] and GIN [86], to verify its universal effectiveness with different models. For the architecture of Transformer,

the number of transformer layers is set to 2, followed by an MLP function to make the prediction. For each transformer layer, the hidden dimension is set to be the same as the number of nodes n, and the number of heads is set to 4. Regarding the GCN backbone, we set the number of GCN layers as 3. The graph representation is obtained with a sum readout function to make the final prediction. We randomly select 70% of the datasets for training, 10% for validation, and the remaining for testing. In the training process, we use the Adam optimizer with an initial learning rate of 10^{-4} and a weight decay of 10^{-4}. The batch size is set as 16. All the models are trained for 200 epochs, and the epoch with the best performance on the validation set is selected for the final report.

4.1.4 Baselines.—We include a variety of Mixup approaches as baselines. Given $\Lambda \in [0,1]^{v \times v}, \alpha \in (0, \infty), \pi \in (0,1), \cdot$ is the dot product.

V-Mixup [101] is the vanilla Mixup by the linear combination of two random samples,

$$\begin{aligned} \tilde{S} &= (1-\lambda)S_i + \lambda S_j, \tilde{y} = (1-\lambda)y_i + \lambda y_j, \\ \lambda &\sim \mathrm{Beta}(\alpha, \alpha). \end{aligned} \tag{15}$$

D-Mixup is the discrete Mixup, a naive baseline designed by ourselves. Given two randomly selected samples, a synthetic sample is generated by obtaining parts of the edges from one sample and the rest from the other,

$$\begin{aligned} &\tilde{S} = (1-\Lambda) \cdot S_i + \Lambda \cdot S_j, \tilde{y} = (1-\lambda)y_i + \lambda y_j, \\ &\Lambda^{i,j} \sim \mathrm{B}(\lambda), \lambda \sim \mathrm{Beta}(\alpha, \alpha). \end{aligned} \tag{16}$$

DropNode [38] randomly selects nodes given a sample and sets all edge weights related to these selected nodes as zero,

$$\tilde{S} = \Lambda \cdot S, \Lambda^{p,:} = \Lambda^{:,p} = z, z \sim \mathrm{Bernoulli}(\pi). \tag{17}$$

DropEdge [71] randomly selects edges given a sample and assigns their weights as zero,

$$\tilde{S} = \Lambda \cdot S, \Lambda^{p,q} \sim \mathrm{Bernoulli}(\pi). \tag{18}$$

G-Mixup [39] is originally proposed for classification tasks, which augments graphs by interpolating the generator of different classes of graphs. Since each cell in a covariance and correlation matrix represents a specific edge in a graph, we can convert a graph generator into a group of generator for each edge. We model each edge generator as a conditional multivariate normal distribution $P(S^{p,q} \mid y)$. The augmentation process can be formulated as,

$$\tilde{S}^{p,q} \sim (1-\lambda)P\left(S^{p,q} \mid y_i\right) + \lambda P\left(S^{p,q} \mid y_j\right), \tilde{y} = (1-\lambda)y_i + \lambda y_j,$$
$$\lambda \sim \mathrm{Beta}(\alpha, \alpha)\,. \tag{19}$$

For the setting of classification,

$$P\left(S^{p,q} \mid y = c\right) \sim \mathcal{N}\left(\mu_c^{p,q}, \left(\sigma_c^{p,q}\right)^2\right), \tag{20}$$

To extend G-Mixup for regression, we slightly modify the augmentation process to adapt it for regression tasks as

$$P\left(S^{p,q} \mid y\right) \sim \mathcal{N}\left(\mu^{p,q} + \frac{\sigma^{p,q}}{\sigma_y}\rho^{p,q}(y - \mu_y), \left(1 - \left(\rho^{p,q}\right)^2\right)\left(\sigma^{p,q}\right)^2\right), \tag{21}$$

where μ and σ are the mean and standard deviation of the weight for each edge, ρ is the correlation coefficient between $S^{p,q}$ and y.

C-Mixup [93] shares the same process with the V-Mixup. Instead of randomly selecting two samples, C-Mixup picks samples based on label distance to ensure the mixed pairs are more likely to share similar labels $(S_j, y_j) \sim P(\cdot \mid (S_i, y_i))$, where P is a sampling function which can sample closer pairs of examples with higher probability. For classification tasks, it degenerates into the intra-class V-Mixup.

4.2 RQ1: Performance Comparison

Overall Performance.—The overall comparison based on the Transformer and GCN backbone are presented in Table 3 and Table 5 respectively, where *ABCD-BioGender*, *PNC, ABIDE*, and *TCGA-Cancer* focus on classification tasks, while *ABCD-Cog* is a regression task. Since the performance of the two backbones demonstrates similar patterns, we focus on the result discussion of the Transformer due to the space limit. Specifically, for classification tasks, incorporating the Mixup technique can constantly improve the performance, especially on the *TCGA-Cancer* dataset, which features a small sample size with high dimensional matrices. Among the various Mixup techniques, our proposed R-Mixup performs the best across datasets and tasks, indicating the further advantage of using log-Euclidean metrics instead of Euclidean metrics for SPD matrices mixture. Besides, for datasets with a relatively smaller sample size, such as PNC, ABIDE, and TCGA-Cancer, R-Mixup can further reduce training variance and stabilize the final performance compared with other data augmentation methods.

Compared with the improvements on classification tasks, R-Mixup demonstrates a more significant advantage on the regression task. It is shown that R-Mixup can significantly reduce the MSE compared with the baseline without Mixup (5.5% with the transformer backbone) and archive a large advantage over the second runner (4.8% with the transformer backbone). It is also noted that other Mixup approaches sometimes hurt the model performance, indicating the Euclidean space cannot measure the distance between SPD matrices very well, and the mixed samples may not be paired with the correct labels. In contrast, our proposed log-Euclidean metric can correctly represent the distance among SPD matrices and therefore address the problem of *arbitrarily incorrect label*.

Performance with Different Sample Sizes.—As collecting labeled data can be extremely expensive for biological networks in practice, we adopt R-Mixup for the challenging low-resource setting to justify its efficacy with limited labeled data only. For this set of experiments, we vary the training sample size from 10% to 100% of the full datasets to show the performance of R-Mixup based on transformers with different sample sizes. Specifically, the ABCD dataset is adopted in this detailed analysis due to its relatively large sample size and supports for both classification and regression tasks. The selected comparing methods are the strongest baselines, namely V-Mixup and C-Mixup, from the overall performance in Table 3. Results are presented in Table 4.

On the classification task of BioGender prediction, impressively, the proposed R-Mixup can already achieve a decent performance with only 10% percent of full datasets and demonstrates a large margin over other compared methods. As the sample size becomes larger, the performance of different data augmentation methods tends to be close, while the proposed R-Mixup reaches the best performance for most of the cases (7 out of 10 setups). On the more challenging regression task of Cognition Summary Score prediction, R-Mixup consistently outperforms the other two baselines under different portions of the training data, which stresses the absolute advantages of our proposed R-Mixup in its flexible and effective adaption for the regression settings. Note that when equipped with an inappropriate augmentation method (i.e., V-Mixup), the regression performance can always deteriorate under different volumes of training data. This implies the necessity of proposing appropriate Mixup techniques tailored for biological networks to address specific challenges for regression tasks. Furthermore, we propose a case study in Appendix G to show why R-Mixup can achieve the best performance for the Regression task in the ABCD-Cog dataset.

4.3 RQ2: The Relations of Sequence Length, SPD-ness and Model Performance

To quantitatively verify the necessary conditions of SPD matrices in Proposition 3.2, we vary the length of sequences whose pairwise correlations compose the network matrices and observe its influence on the percentage of positive eigenvalues and the final prediction performance. For better illustration, we define a new terminology *SPD-ness* to reflect the percentage of positive values among all eigenvalues. The higher the percentage of positive eigenvalues, the higher SPD-ness, and a full SPD matrice requires all the eigenvalues to be positive. Specifically, we choose the dataset with the longest time sequence, namely ABCD, to facilitate this study. Since samples in the ABCD dataset are of different sequence lengths, we simply select those with sequence length longer than 1024 and truncate them to 1024 to form a length-unified dataset *ABCD-1024*, leading to 4613 samples for the *ABCD-BioGender* classification task and 4533 samples for the *ABCD-Cog* regression task.

First, we investigate the relationship between the length of biological sequences t and the *SPD-ness* of the corresponding network matrix. The results are shown in Figure 4(a), where the value of sequence length t is varied from 90 to 900 with a step size of 90. For each given t, we construct the correlation matrices based on each pair of the truncated sequences with only the first t elements from the original sequences. Then the eigenvalue decomposition is applied to each obtained correlation matrix, and the percentage of positive ($>10^{-6}$) eigenvalues are calculated. The reported results are the average over all the correlation matrices. From this curve, we observe that the percentage of positive eigenvalues grows gradually as the time-series length increases. The growth trend gradually slows down, reaching a percentage point saturation at about the length of 540, where the full percentage indicates full *SPD-ness*. Note that the number of variables n for the ABCD dataset is 360. This aligns with our conclusion in Proposition 3.2 that a necessary condition for correlation matrices satisfying SPD matrices is $t \geq n$.

Second, with the verified relation between the sequence length t and *SPD-ness*, we study the influence of sequence length t or *SPD-ness* on the prediction performance. We observe that directly truncating the time series to length t will lose a huge amount of task-relevant signals, resulting in a significant prediction performance drop. As an alternative, we reduce the original sequence to length t by taking the average of each $1024/t$ consecutive sequence unit. Results on the classification task *ABCD-BioGender* and the regression task *ABCD-Cognition* with the input of different time-series length t are demonstrated in Figure 4(b). It shows that for the classification task, although V-Mixup and C-Mixup demonstrate an advantage when the percentage of positive eigenvalues is low, the performance of the proposed R-Mixup continuously improves as the sequence length t increases and finally beats the other baselines. For the regression task, our proposed R-Mixup consistently performs the best regardless of the *SPD-ness* of the correlation matrices. The gain is more observed when the dataset matrices are full SPD. Combining these observations from both classification and regression tasks, we prove that the proposed R-Mixup demonstrates superior advantages for mixing up SPD matrices and facilitating biological network analysis that satisfies full *SPD-ness*.

4.4 RQ3: Hyperparameter and Efficiency Study

The Influence of Key Hyperparameter α.—We study the influence of the key hyperparameter α in R-Mixup, which correspondingly changes the Beta distribution of λ in Equation (13). Specifically, the value of α is adjusted from 0.1 to 1.0, and the corresponding prediction performance under the specific values is demonstrated in Figure 5. We observe that the prediction performance of both classification and regression tasks are relatively stable as the value of α varies, indicating that the proposed R-Mixup is not sensitive to the key hyperparameter α.

Efficiency Study.—To further investigate the efficiency of different Mixup methods, we compare the training time of different data augmentation methods on the large-scale dataset, ABCD, to highlight the difference. The results are shown in Figure 6. Besides, the running time comparison on three smaller datasets, ABIDE, PNC, and TCGA-Cancer are also included in appendix D for reference. All the compared methods are trained with the same backbone model [53]. It is observed that with the precomputed eigenvalue decomposition, the training speed of the optimized R-Mixup on the large ABCD dataset can be 2.5 times faster than the original model without optimization. Besides, on the smaller datasets such as PNC, ABIDE, and TCGA-Cancer, there is no significant difference in elapsed time between different methods.

5 CONCLUSION

In this paper, we present R-Mixup, an effective data augmentation method tailored for biological networks that leverage the log-Euclidean distance metrics from the Riemannian manifold. We further propose an optimized strategy to improve the training efficiency of R-Mixup. Empirical results on five real-world biological network datasets spanning both classification and regression tasks demonstrate the superior performance of R-Mixup over existing commonly used data augmentation methods under various data scales and downstream applications. Besides, we theoretically verify a necessary condition overlooked by prior works to determine whether a correlation matrix is SPD and empirically demonstrate how it affects the prediction performance, which we expect to guide future applications spreading the biological networks.

ACKNOWLEDGMENTS

This research was supported in part by the University Research Committee of Emory University and the National Institute Of Diabetes And Digestive And Kidney Diseases of the National Institutes of Health under Award Number K25DK135913. The authors also gratefully acknowledge support from NIH under award number R01MH105561 and R01MH118771. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health. Besides, the acknowledgment of used datasets can be found in Appendix H.

A: COVARIANCE, CORRELATION AND POSITIVE DEFINITE MATRICES

We provide detailed definitions on covariance, correlation and positive definite matrices with necessary properties here.

Definition A.1. Let $X = (X_i) = (x_{ik})$ with $i = 1, \ldots, n$ and $k = 1, \ldots, t$ be t-dimensional vectors of n variables. The corresponding *covariance matrix* $\mathrm{Cov}(X)$ is defined as

$$\begin{aligned}\mathrm{Cov}\left(X\right)_{ij} &= \frac{1}{t}\left(\sum_k (x_{ik} - E(X_i))(x_{jk} - E(X_j))\right) \\ &= E(X_iX_j) - E(X_i)E(X_j)\,.\end{aligned} \tag{22}$$

The *correlation matrix* is normalized as:

$$\begin{aligned}&\mathrm{Cor}\left(X\right) = \mathrm{diag}\left(\frac{1}{\sqrt{\mathrm{Cov}\,(X)_{11}}}, \ldots, \frac{1}{\sqrt{\mathrm{Cov}\,(X)_{vv}}}\right). \\ &\mathrm{Cov}\,(X) \cdot \mathrm{diag}\left(\frac{1}{\sqrt{\mathrm{Cov}\,(X)_{11}}}, \ldots, \frac{1}{\sqrt{\mathrm{Cov}\,(X)_{vv}}}\right).\end{aligned}$$

(23)

Expressed by matrix entries, we restore the familiar *Pearson correlation coefficients*:

$$\mathrm{Cor}\left(X\right)_{ij} = \frac{\mathrm{Cov}\,(X)_{ij}}{\sqrt{\mathrm{Cov}\,(X)_{ii}}\sqrt{\mathrm{Cov}\,(X)_{jj}}}\,. \tag{24}$$

Remark A.2. It should be noted that to make the definition of $\mathrm{Cor}\,(X)$ valid, $\mathrm{Cov}\,(X)_{ii} \neq 0$ for all i. Since

$$\mathrm{Cov}(X)_{ii} = E(X_iX_i) - E(X_i)E(X_i) = \frac{1}{t}\sum_k x_{ik}^2 - \left(\frac{1}{t}\sum_k x_{ik}\right)^2, \tag{25}$$

the geometric mean inequality says that $\mathrm{Cov}\,(X)_{ii}$ vanishes only when x_{ik} are identical, which does not happen in our case.

Definition A.3. A symmetric $n \times n$ matrix S is *positive semi-definite* if for any vector $u \in \mathbb{R}^n, u^T S u \geq 0$. Equivalently, this means that the eigenvalues of S are all nonnegative. If the inequality holds strictly, S is said to be *positive definite*, or *symmetric positive definite*, or SPD for short.

Proposition A.4. *Covariance and correlation matrices are positive semi-definite. A necessary condition for them to be positive definite is that the length of each sample is no less than the number of variables, i.e., $t \geq n$.*

Proof. Recall Eq.(22) from Definition A.1, let us consider column vectors $Y_k = (x_{ik} - E(X_i))$. Then $\mathrm{Cov}\left(X\right) = \frac{1}{t}\sum_k Y_k Y_k^T$. Given any vector $u \in \mathbb{R}^n$,

$$u^T \mathrm{Cov}\left(X\right) u = \frac{1}{t}\sum_i u^T Y_k Y_k^T u = \frac{1}{t}\sum_i \left(Y_k^T u\right)^2 \geq 0\,. \tag{26}$$

On the other hand, by Eq.(23)

$$\begin{aligned} & u^T \mathrm{Cor}(X) u = u^T \mathrm{diag}\left(\frac{1}{\sqrt{\mathrm{Cov}(X)_{11}}}, \ldots, \frac{1}{\sqrt{\mathrm{Cov}(X)_{vv}}}\right) \cdot \\ & \mathrm{Cov}(X) \cdot \mathrm{diag}\left(\frac{1}{\sqrt{\mathrm{Cov}(X)_{11}}}, \ldots, \frac{1}{\sqrt{\mathrm{Cov}(X)_{vv}}}\right) u \\ & = \tilde{u}^T \mathrm{Cov}(X) \tilde{u} \geq 0\,. \end{aligned} \tag{27}$$

If $\{Y_k\}_{k=1}^{t}$ spans the whole vector space $\mathbb{R}^n$, in which case t must be no less than n, then $\mathrm{Cov}(X)$ is positive definite. Otherwise, there must be some vector u perpendicular to all Y_k, which leads to $\sum_i (Y_k^T u)^2 = 0$. □

B: GEODESICS AND SWELLING EFFECT

We list the geodesic equation and Riemannian distance function induced from log-Euclidean metric on $\mathrm{Sym}^+(n)$ here followed by a rigorous proof on the swelling effect.

Definition B.1. Let $\mathrm{Sym}^+(n)$ denote the manifold of positive definite matrices equipped with the *log-Euclidean* metric. Analytically, the induced distance function reads

$$d(S_i, S_j) = \| \log S_i - \log S_j \|, \tag{28}$$

which measures the distance between different two points $S_i, S_j \in \mathrm{Sym}^+(n)$, with the following geodesic connecting two points:

$$\gamma(\lambda) = \exp((1-\lambda)\log S_i + \lambda \log S_j)\,. \tag{29}$$

Detailed derivation of the geodesics equation can be found in [35, 57, 72].

Proposition B.2 (Swelling Effect). *Given arbitrary* $S_i, S_j \in \mathrm{Sym}^+(n)$, *then*

$$\begin{aligned} &\det(\exp((1-\lambda)\log S_i + \lambda \log S_j)) \\ &\le \det(((1-\lambda)\log S_i + \lambda \log S_j))\,. \end{aligned} \tag{30}$$

Especially,

$$\begin{aligned} &\min\{\det S_i, \det S_j\} \\ &\le \det(\exp((1-\lambda)\log S_i + \lambda \log S_j)) \le \max\{\det S_i, \det S_j\}, \end{aligned} \tag{31}$$

while $\det((1-\lambda)\log S_i + \lambda \log S_j)$ would exceed the determinants of both S_i and S_j as shown in Figure 2 *in the main text.*

Proof. To prove the first inequality, we note one basic fact of matrix exponential: $\det(\exp(A)) = \exp(\mathrm{Tr}\, A)$. Thus,

$$\begin{aligned}
&\det\left(\exp\left((1-\lambda)\log S_i + \lambda\log S_j\right)\right)\\
&= \exp\left(\operatorname{Tr}\left((1-\lambda)\log S_i + \lambda\log S_j\right)\right)\\
&= \exp\left((1-\lambda)\operatorname{Tr}\log S_i\right)\exp\left(\lambda\operatorname{Tr}\log S_j\right)\\
&= (\exp\operatorname{Tr}\log S_i)^{1-\lambda}(\exp\operatorname{Tr}\log S_j)^{\lambda}\\
&= (\det S_i)^{1-\lambda}(\det S_2)^{\lambda}.
\end{aligned} \tag{32}$$

Then we make use of the following identity of n-dimensional Gaussian integral:

$$\int \exp\left(-\frac{1}{2}x^T S x\right)dx = \sqrt{\frac{(2\pi)^n}{\det S}}, \tag{33}$$

where $x \in \mathbb{R}^n$ and $S \in \operatorname{Sym}^+(n)$. In our case,

$$\begin{aligned}
&\sqrt{\frac{(2\pi)^n}{\det\left(((1-\lambda)S_i + \lambda S_j)\right)}}\\
&= \int \exp\left(-\frac{1}{2}x^T((1-\lambda)S_i + \lambda S_j)x\right)dx\\
&= \int \left(\exp\left(-\frac{1}{2}x^T S_i x\right)\right)^{1-\lambda}\left(\exp\left(-\frac{1}{2}x^T S_j x\right)\right)^{\lambda}dx\\
&\le \left(\int \exp\left(-\frac{1}{2}x^T S_i x\right)dx\right)^{1-\lambda}\left(\int \exp\left(-\frac{1}{2}x^T S_j x\right)dx\right)^{\lambda}\\
&= \sqrt{\frac{(2\pi)^n}{(\det S_i)^{1-\lambda}(\det S_j)^{\lambda}}}.
\end{aligned} \tag{34}$$

We use Hölder's inequality in the last step from above, which yields

$$\begin{aligned}
&\det\left(\exp\left((1-\lambda)\log S_i + \lambda\log S_j\right)\right)\\
&= (\det S_i)^{1-\lambda}(\det S_j)^{\lambda} \le \det\left(((1-\lambda)S_i + \lambda S_j)\right).
\end{aligned} \tag{35}$$

To prove the second inequality, let us assume $\det S_i \le \det S_j$ and let $a = \det S_j / \det S_i \ge 1$. It is straightforward to check that $a^{\lambda} - 1 \le (a-1)\lambda$ when $0 \le \lambda \le 1$. This fact indicates that

$$\begin{aligned}
&\left(\frac{\det S_j}{\det S_i}\right)^{\lambda} - 1 \le \left(\frac{\det S_j}{\det S_i} - 1\right)\lambda\\
&\Longrightarrow \left(\frac{\det S_j}{\det S_i}\right)^{\lambda} \le \left(\frac{\det S_j}{\det S_i}\right)\lambda + \left(1-\lambda\right)\\
&\Longrightarrow (\det S_i)^{1-\lambda}(\det S_j)^{\lambda} \le (\det S_i)\left(1-\lambda\right) + (\det S_j)\lambda,
\end{aligned} \tag{36}$$

and finishes the proof. □

C: KERNEL REGRESSION ON $\mathrm{Sym}^+(n)$

We now present the proof details of Theorem 3.3 from the main text. To begin with, we introduce a method from heat kernel theory [5] to generalize to Euclidean Gauss kernel

$$K_E(S_i, \widetilde{S}) = \frac{1}{(2\pi\sigma^2)^{n^2/2}} \exp\left(-\frac{1}{2\sigma^2} \| S_i - \hat{S} \|^2\right) \tag{37}$$

on $\mathrm{Sym}^+(n)$ over which our samples are distributed. The notion of *geodesic regression* [9, 34] would also become apparent as we move forward. Let us first consider the following classical heat equation on Euclidean space

$$\left(\frac{\partial}{\partial t} - \sum_i \frac{\partial^2}{\partial x_i^2}\right) f = \frac{\partial f}{\partial t} + \Delta f = 0, \tag{38}$$

where $\Delta = \sum_i \frac{\partial^2}{\partial x_i^2}$ is the Laplacian. A solution $f(x, t)$ to this equation is interpreted as the temperature at position x and time t. Substituting $t = \sigma^2/2$ into Eq.(9), it can be check by definition that the function

$$K_t\left(x, y\right) = \frac{1}{(4\pi t)^{n^2/2}} \exp\left(-\frac{1}{4t} \| x - y \|^2\right) \tag{39}$$

solves Eq.(38). It is called the *fundamental solution* or *heat kernel* as any other solutions to Eq.(38) can be written as the convolution with a certain function $f(y)$:

$$f(x, t) = \int_{\mathbb{R}^{n \times n}} K_t(x, y) f(y) dy \,. \tag{40}$$

Based on this fact, it is natural to define Riemannian Gauss kernel as the fundamental solution to heat equation on Riemannian manifolds. To this end, we need to replace the Laplacian on Euclidean spaces by *Laplace–Beltrami operator*, still denoted by Δ, on manifolds. The formal definition of this operator is unnecessary here and we recommend interested readers to [5, 35] for more details. It is enough to known its local coordinate expression for our purpose. Specifically, being different from Euclidean spaces with a standard and explicit *coordinate system*, i.e., any vector $x \in \mathbb{R}^n$ can be explicitly expressed by its components (coordinates) x_i, the coordinates of points p in a manifold M always need being defined exclusively depending on the concerned manifold. In the most general case, it is only known that manifolds admit local coordinate parametrizations for its local regions as they resemble Euclidean spaces. Expressed by any local coordinates,

$$\Delta f = \sum_{i,j} g^{ij}\left(\frac{\partial^2 f}{\partial x_i^2} - \Gamma_{ij}^k \frac{\partial f}{\partial x_k}\right) \tag{41}$$

where g^{ij}, Γ_{ij}^k are called dual metric and Christoffel symbols and are all determined by the Riemannian metric [57, 72]. Now we wish to analyze the heat equation expressed by local coordinates. However due to its intricate form involving the Riemannian metric, it is generally impossible to solve the equation analytically. Even though, the following theorem is established in heat kernel theory using advanced tools from differential geometry:

Theorem C.1. [5] Let M be a complete Riemannian manifold, then there exists a function $K_t(p,q)$, called heat kernel, with the following properties

1. $K_t(p,q) = K_t(q,p)$ *for all* $p, q \in M$.
2. $\lim_{t\to 0} K_t(p,q)$ *equals the Dirac delta function* $\delta_x(y)$.
3. $K_t(p,q)$ is positive definite and solves the heat equation.
4. $K_t(p,q) = \int_M K_{t-s}(p,p')K_s(p',q)dp'$ *for any* $s > 0$.

We are only interested in the third property as its confirms that the heat kernel truly determines a feature map from $\mathrm{Sym}^+(n)$ into a higher dimensional feature space [74]. Let $\boldsymbol{y} = (y_i)$ denote the column vector consisting of training data labels, let $\boldsymbol{K}_{\widetilde{S}}$ denote the column vector consisting of $K_R(S_i, \widetilde{S})$ and let $G = (K_R(S_i, S_j))$ denote the kernel matrix evaluated by K_R on the data set. Then the predictor function regressed through *kernel ridge regression* is

$$\widetilde{m}(\widetilde{S}) = \boldsymbol{y}^T G^{-1} \boldsymbol{K}_{\widetilde{S}}. \tag{42}$$

As a reminder, if $\det G = 0$ (when does not happen here since the kernel function is positive definite), a regularization $\zeta \geq 0$ can be chosen as a trade-off between weights and square errors when optimizing the regression. The log-Euclidean metric is now explicitly used in evaluating $\boldsymbol{K}_{\widetilde{S}}$ as well as G. On the other hand, a vanilla geodesic regression could be intuitively treat as the multi-linear regression on manifold with the Riemannian metric substituting for the Euclidean metric, which is fairly easy to deal with when we have a coordinate system and this is what we are going to do in the following proof.

Theorem C.2. *For* $\mathrm{Sym}^+(n)$ with log-Euclidean metric and estimators $\widetilde{m}$ obtained with either geodesic regression or Gaussian kernel regression on samples. Augmented data from Riemannian geodesics bear the mean square error no more than those from straight lines.

Proof. We first note the fact that the exponential and logarithm functions

$$\exp : \mathrm{Sym}(n) \to \mathrm{Sym}^{+}(n), \quad \log : \mathrm{Sym}^{+}(n) \to \mathrm{Sym}(n) \tag{43}$$

are *isometries* between $\mathrm{Sym}^{+}(n)$ and $\mathrm{Sym}(n)$. That is: (*a*) they are bijective and (*b*) preserve the Riemannian distance functions:

$$\| H_i - H_j \| = d(\exp H_i, \exp H_j), \quad d(S_i, S_i) = \| \log S_i - \log S_j \| \tag{44}$$

for any $S_i, S_j \in \mathrm{Sym}^{+}(n)$ and any $H_i, H_j \in \mathrm{Sym}(n)$. A detailed proof on the RHS equation from can be found in [35] and with the bijectivity of exp and log, we obtain the LHS equation from above. Besides, being defined as collection of all symmetric $n \times n$ matrices, $\mathrm{Sym}(n)$ is an $\frac{1}{2}n(n-1)$-dimensional Euclidean space with a standard coordinate system introduced above. Combining with the logarithm function $\log : \mathrm{Sym}^{+}(n) \to \mathrm{Sym}(n)$, then we obtain a coordinate system for $\mathrm{Sym}^{+}(n)$ within which we can express the heat equation Eq.(38) explicitly.

Since log is an isometry and since $\mathrm{Sym}(n)$ is a Euclidean space, as a basic result in Riemannian geometry, the Christoffel symbols Γ_{ij}^{k} in Eq.(41) vanishes [57] in the defined coordinate system and hence the Laplace–Beltrami operator degenerates to the common Laplacian. As a result, the fundamental solution is exactly Eq.(39) expressed by the coordinates. After taking the inverse map exp, the Euclidean distance is replaced by Riemannian distance in Eq.(39). Hence,

$$K_R(S_i, \hat{S}) = \frac{1}{(2\pi\sigma^2)^{\frac{n(n-1)}{4}}} \exp\left(-\frac{1}{2\sigma^2} d(S_i, \hat{S})^2\right), \tag{45}$$

is the heat kernel on $\mathrm{Sym}^{+}(n)$ with the property being positive definite by Theorem C.1.

Recall that the Riemannian geodesic of log-Euclidean metric is

$$\gamma(\lambda) = \widetilde{S} = \exp((1-\lambda)\log S_i + \lambda \log S_j)\,. \tag{46}$$

Its coordinate representations is then

$$\log(\gamma(\lambda)) = (1-\lambda)\log S_i + \lambda \log S_j, \tag{47}$$

which is a straight line connecting $\log S_i$ and $\log S_j \in \mathrm{Sym}(n)$. As a contrary, the coordinate representation of

$$\eta(\lambda) = \widetilde{S}' = (1-\lambda)S_i + \lambda S_j \tag{48}$$

is highly curved as

$$\log(\eta(\lambda)) = (\log(1-\lambda)S_i + \lambda S_j). \tag{49}$$

Since $\mathrm{Sym}(n)$ is an Euclidean space and since we verified above that the function log is an isometry, conducting geodesic regression for the samples $\{(S_i, y_i) \mid i = 1, \ldots, N\}$ is merely solving the linear model of $\{(\log S_i, y_i) \mid i = 1, \ldots, N\}$. Since the total sample number N in our case is less than the dimension of the ambient Euclidean space n^2, the optimal solution is just a hyperplane encompassing all samples as well as those synthesized via Eq.(47). However, curves like Eq.(49) are manifestly deviated from the regression hyperplane which leads to large square loss.

To verify the case involving Gaussian kernel, we make use of the following operator inequalities [12]:

$$\log((1-\lambda)S_i + \lambda S_j) \geq (1-\lambda)\log S_i + \lambda \log S_j. \tag{50}$$

Intuitively, the logarithm is a concave function on $(0, +\infty)$, which is generalized to hold in the setting of positive semidefinite matrices with $A \geq B$ meaning $A - B$ is positive semidefinite. For simplicity, we only analyze Eq.(42) for a pair of samples S_i, S_j as an augmented sample $\widetilde{S}$ is coined in this way through our mixup method. In statistics, penalty functions [94] can be employed weaken the influence of other samples and achieve this effect. With these preparation,

$$\begin{aligned}\widetilde{m}(S) = \boldsymbol{y}^T G^{-1} \boldsymbol{K}_S &= (y_i, y_j)\begin{pmatrix} K_R(S_i, S_i) & K_R(S_i, S_j) \\ K_R(S_j, S_i) & K_R(S_i, S_i) \end{pmatrix}^{-1}\begin{pmatrix} K_R(S_i, S) \\ K_R(S_j, S) \end{pmatrix} \\ &= \frac{1}{1 - K_{ij}^2}((y_i - K_{ij}y_j)K_{i,S} + (y_j - K_{ij}y_i)K_{j,S})\Big),\end{aligned} \tag{51}$$

where $K_{ij} = \exp\left(-\frac{1}{2\sigma^2} d\left(S_i, \hat{S}\right)^2\right)$ is an abbreviation for the non-normalized Gaussian distribution of log-Euclidean distance with $K_{i,S}$ being denoted analogously. Substituting $\widetilde{S}$ and $\widetilde{S}'$ from Eq.(46) and Eq.(48) into the above equation, we then compare the estimators with $\widetilde{y} = (1-\lambda)y_i + \lambda y_j$ directly.

For predictions of $\widetilde{S}$, we note that

$$K_{ij} = \exp\left(-\frac{1}{2\sigma^2}\| \log S_i - \log S_j \|\right), \tag{52}$$

$$K_{i,\widetilde{S}} = \exp\left(-\frac{1}{2\sigma^2}\lambda\| \log S_i - \log S_j \|\right) = K_{ij}^{\lambda} \tag{53}$$

$$K_{j,\widetilde{S}} = \exp\left(-\frac{1}{2\sigma^2}\left(1-\lambda\right)\| \log S_i - \log S_j \|\right) = K_{ij}^{1-\lambda} \tag{54}$$

with

$$\widetilde{m}\left(\widetilde{S}\right) = \frac{1}{1-K_{ij}^2}\left(K_{ij}^{\lambda}(y_i - K_{ij}y_j) + K_{ij}^{1-\lambda}(y_j - K_{ij}y_i)\right) \tag{55}$$

being a *concave function* for $\lambda \in [0,1]$. This can be demonstrated by examining that the second order derivative

$$\frac{d^2\widetilde{m}\left(\widetilde{S}\left(\lambda\right)\right)}{d\lambda^2} = \frac{\ln^2 K_{ij}}{K_{ij}^{\lambda}\left(1-K_{ij}^2\right)}\left(\left(K_{ij} - K_{ij}^{2\lambda+1}\right)y_j + \left(K_{ij}^{2\lambda} - K_{ij}^2\right)y_i\right), \tag{56}$$

which is nonnegative because $K_{ij} \le 1$ and $K_{ij} - K_{ij}^{2\lambda+1}, K_{ij}^{2\lambda} - K_{ij}^2 \ge 0$. As a result, $\widetilde{m}\left(\widetilde{S}\right) \le \widetilde{y}$. On the other hand,

$$K_{i,\widetilde{S}'} = \exp\left(-\frac{1}{2\sigma^2}\| \log\left((1-\lambda)S_i - \lambda S_j\right) - \log S_i \|\right) \tag{57}$$

$$K_{j,\widetilde{S}'} = \exp\left(-\frac{1}{2\sigma^2}\| \log\left((1-\lambda)S_i - \lambda S_j\right) - \log S_j \|\right) \tag{58}$$

are intricate as the linear combination of matrices $((1-\lambda)S_i - \lambda S_j)$ does not commute with the logarithm. Despite of this difficulty, we are still above to compare $\widetilde{m}\left(\widetilde{S}\right)$ and $\widetilde{m}\left(\widetilde{S}'\right)$ based on their general expansion in Eq.(51). By (50),

$$\begin{aligned}
&\log\left((1-\lambda)S_i+\lambda S_j\right)-\log S_i \geq \lambda(\log S_i+\log S_j)\\
&\Longrightarrow \|\log\left((1-\lambda)S_i+\lambda S_j\right)-\log S_i\| \geq \|\lambda(\log S_i+\log S_j)\|\\
&\Longrightarrow K_{i,\widetilde{S}'} = \exp\left(-\frac{1}{2\sigma^2}\|\log\left((1-\lambda)S_i-\lambda S_j\right)-\log S_i\|\right)\\
&\leq \exp\left(-\frac{1}{2\sigma^2}\lambda\|\log S_i-\log S_j\|\right) = K_{i,\widetilde{S}}.
\end{aligned} \tag{59}$$

The second inequality is due to the fact that the operator norm || - || equals the largest eigenvalue of any positive semidefinite operator. Similar argument also implies case with S_j. Together with the concavity of $\widetilde{m}(\widetilde{S})$, Eq.(51) and the range of our labels, we conclude that

$$0 \leq \widetilde{m}(\widetilde{S}') \leq \widetilde{m}(\widetilde{S}) \leq \widetilde{y} \Longrightarrow \sum\left(\widetilde{m}(\widetilde{S})-\widetilde{y}\right)^2 \leq \left(\widetilde{m}(\widetilde{S}')-\widetilde{y}\right)^2, \tag{60}$$

which are finally summed over the samples to show that the square error of estimation using geodesics is no more than that using straight lines on $\mathrm{Sym}^+(n)$. □

Remark C.3. For affine-invariant metric, it has been shown that the induced *Riemannian curvature tensor* R is nonzero [72, 78] and hence it is impossible to find coordinate systems within which $\Gamma_{ij}^k = 0$ [57]. Therefore, the fundamental solution to the heat equation can never take in the concise form as Eq.(45) and Theorem 3.3 becomes invalid to appraise the case when using affine-invariant metric.

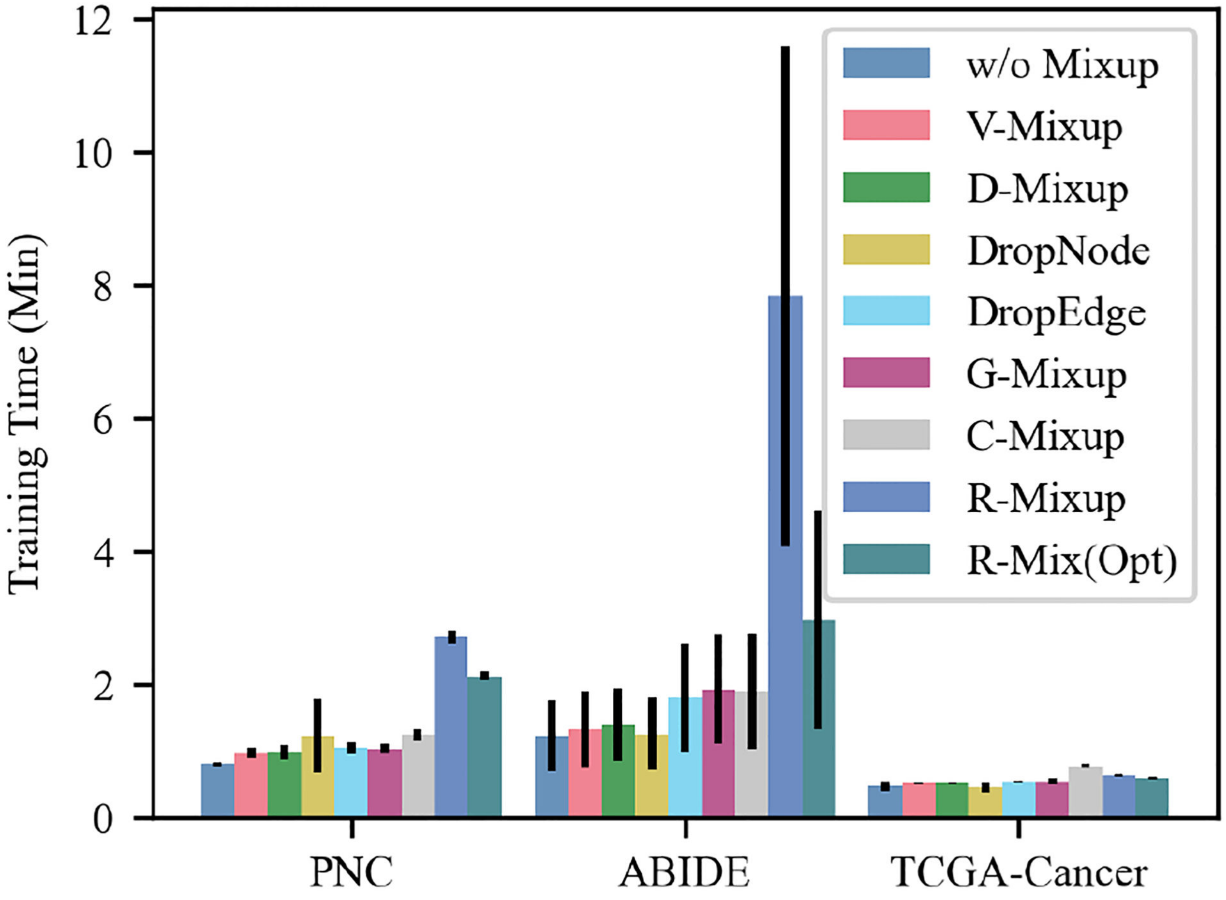


Figure 7:
Running Time in PNC, ABIDE and TCGA-Cancer. R-Mixup is the original version of our method while R-Mix(Opt) is the proposed optimized version in Section 3.5.

D: RUNNING TIME ON THREE SMALLER DATASETS

As shown in Figure 7, on the smaller datasets, PNC, ABIDE, and TCGA-Cancer, there is no significant difference in elapsed time between the different methods. Notably, the proposed R-Mixup is magically faster than C-Mixup on the TCGA-Cancer dataset. This is mainly due to the small node size of TCGA-Cancer, which reduces the main barrier of the eigenvalue decomposition in R-Mixup, while the time cost of the sampling operation in the C-Mixup baseline does not change dynamically with the node size.

E: CODE IMPLEMENTATION

```python
import torch
import numpy as np

def tensor_log(t):
    # condition: t is symmetric.
    s, u = torch.linalg.eigh(t)
    s[s <= 0] = 1e-8
    return u @ torch.diag_embed(torch.log(s)) @ u.permute
     (0, 2, 1)

def tensor_exp(t):
    # condition: t is symmetric.
    s, u = torch.linalg.eigh(t)
    return u @ torch.diag_embed(torch.exp(s)) @ u.permute
     (0, 2, 1)

def r_mixup(x, y, alpha=1.0, device='cuda'):
    if alpha > 0:
        lam = np.random.beta(alpha, alpha)
    else:
        lam = 1
    batch_size = y.size()[0]
    index = torch.randperm(batch_size).to(device)
    x = tensor_log(x)
    x = lam * x + (1 - lam) * x[index, :]
    y = lam * y + (1 - lam) * y[index]
    return tensor_exp(x), y
```

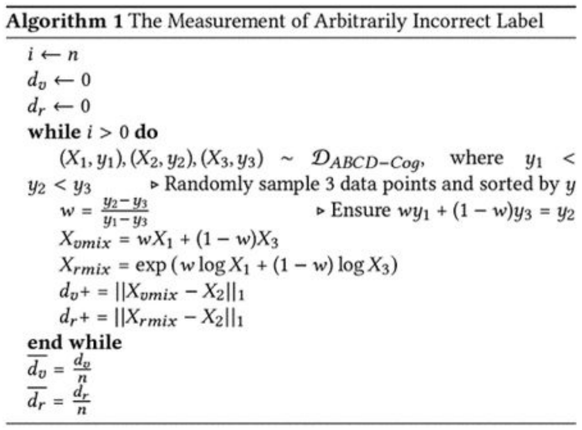
Listing 1:
Python Example

F: GCN BACKBONE PERFORMANCE

The performance of models with the GCN backbones can be found in Table 5.

G: CASE STUDY ABOUT ARBITRARILY INCORRECT LABEL PROBLEM

To verify how R-Mixup migrate the *arbitrarily incorrect label* problem, we design the following process:



Algorithm 1 The Measurement of Arbitrarily Incorrect Label

```
$i \leftarrow n$
$d_v \leftarrow 0$
$d_r \leftarrow 0$
while $i > 0$ do
    $(X_1, y_1), (X_2, y_2), (X_3, y_3) \sim \mathcal{D}_{ABCD-Cog}$, where $y_1 < y_2 < y_3$   ▷ Randomly sample 3 data points and sorted by $y$
    $w = \frac{y_2 - y_3}{y_1 - y_3}$   ▷ Ensure $wy_1 + (1 - w)y_3 = y_2$
    $X_{vmix} = wX_1 + (1 - w)X_3$
    $X_{rmix} = \exp(w \log X_1 + (1 - w) \log X_3)$
    $d_v + = ||X_{vmix} - X_2||_1$
    $d_r + = ||X_{rmix} - X_2||_1$
end while
$\overline{d_v} = \frac{d_v}{n}$
$\overline{d_r} = \frac{d_r}{n}$
```

We set n as 1000 and obtain $\overline{d}_v = 24{,}416.04 \pm 4{,}066.60$, $\overline{d}_r = 22{,}622.41 \pm 3{,}873.05$, where the sample distance $\overline{d}_r$ from R-Mixup is significantly smaller (7.3%) than the sample distance $\overline{d}_v$ from V-Mixup. The phenomenon shows our R-Mixup indeed can migrate the *arbitrarily incorrect label* problem.

H: DATASET ACKNOWLEDGMENTS

Part of data used in the preparation of this article were obtained from the Philadelphia Neurodevelopmental Cohort (PNC) study[1] and the Adolescent Brain Cognitive Development (ABCD) Study[2]. The PNC study support for the collection of the datasets was provided by grant RC2MH089983 awarded to Raquel Gur and RC2MH089924 awarded to Hakon Hakorson. All subjects were recruited through the Center for Applied Genomics at The Children's Hospital in Philadelphia. held in the NIMH Data Archive (NDA). This is a multisite, longitudinal study designed to recruit more than 10,000 children age 9–10 and follow them over 10 years into early adulthood. The ABCD Study® is supported by the National Institutes of Health and additional federal partners under award numbers U01DA041048, U01DA050989, U01DA051016, U01DA041022, U01DA051018, U01DA051037, U01DA050987, U01DA041174, U01DA041106, U01DA041117, U01DA041028, U01DA041134, U01DA050988, U01DA051039, U01DA041156, U01DA041025, U01DA041120, U01DA051038, U01DA041148, U01DA041093, U01DA041089, U24DA041123, U24DA041147. A full list of supporters is available at https://abcdstudy.org/federal-partners.html. A listing of participating sites and a complete listing of the study investigators can be found at https://abcdstudy.org/consortium_members/. ABCD consortium investigators designed and implemented the study and/or provided data but did not necessarily participate in the analysis or writing of this report. This manuscript reflects the views of the authors and may not reflect the opinions or views of the NIH or ABCD consortium investigators. The ABCD data repository grows and changes over time. The ABCD data used in this report came from NIMH Data Archive Release 4.0 (DOI 10.15154/1523041). DOIs can be found at https://nda.nih.gov/abcd.

[1] https://www.med.upenn.edu/bbl/philadelphianeurodevelopmentalcohort.html
[2] https://abcdstudy.org

Table 5:

Overall performance comparison based on the GCN backbone. The best results are in bold, and the second best results are underlined. The ↑ indicates a higher metric value is better and ↓ indicates a lower one is better.

Method	ABCD-BioGender		ABCD-Cog	PNC		ABIDE		TCGA-Cancer	
	AUROC↑	Accuracy↑	MSE↓	AUROC↑	Accuracy↑	AUROC↑	Accuracy↑	Precision↑	Recall↑
w/o Mixup	78.82±0.62	71.55±0.43	80.85±4.69	59.14±5.66	60.00±4.72	55.09±6.91	55.20±6.18	30.49±6.89	40.83±6.18
V-Mixup	81.47±0.79	73.97±0.79	80.35±3.09	63.63±3.80	61.76±3.25	58.49±6.58	56.40±3.91	38.50±9.22	46.67±11.18
D-Mixup	81.30±0.35	73.67±0.39	80.90±9.48	58.68±6.24	58.43±5.99	60.19±6.58	55.40±6.15	37.67±5.47	**50.00±4.17**
DropNode	80.77±2.02	73.18±2.09	88.11±10.59	63.65±5.04	61.57±5.16	59.49±4.99	56.60±5.55	29.58±7.69	39.17±10.46
DropEdge	79.98±1.54	72.23±1.37	85.98±2.31	56.61±2.72	56.67±2.89	56.58±6.78	54.80±4.76	39.44±7.72	**50.00±7.80**
G-Mixup	81.30±1.07	73.90±0.86	81.28±3.46	57.25±3.75	57.45±2.91	62.43±2.94	**60.40±3.44**	38.64±8.47	49.17±9.03
C-Mixup	81.62±1.65	73.62±1.80	78.86±3.51	60.88±7.24	58.24±7.61	60.22±9.32	57.40±5.32	34.17±11.74	46.67±15.14
R-Mixup	**82.85±1.86**	**75.86±1.88**	**74.88±2.03**	**64.39±5.05**	**62.31±3.32**	**63.03±5.58**	59.67±5.96	**44.78±8.64**	48.44±8.61

1 https://www.med.upenn.edu/bbl/philadelphianeurodevelopmentalcohort.html

2 https://abcdstudy.org

REFERENCES

[1]. Anirudh Rushil and Thiagarajan Jayaraman J.. 2019. Bootstrapping Graph Convolutional Neural Networks for Autism Spectrum Disorder Classification. In ICASSP. IEEE, 3197–3201.

[2]. Arpit Devansh, Jastrzebski Stanislaw, Ballas Nicolas, Krueger David, Bengio Emmanuel, Kanwal Maxinder S., Maharaj Tegan, Fischer Asja, Courville Aaron C., Bengio Yoshua, and Lacoste-Julien Simon. 2017. A Closer Look at Memorization in Deep Networks. In Proc. of ICML PMLR, 233–242.

[3]. Aykent Sarp and Xia Tian. 2022. Gbpnet: Universal geometric representation learning on protein structures. In KDD. 4–14

[4]. Barachant Alexandre, Bonnet Stéphane, Congedo Marco, and Jutten Christian. 2010. Riemannian geometry applied to BCI classification. In International conference on latent variable analysis and signal separation. Springer, 629–636.

[5]. Berline Nicole, Getzler Ezra, and Vergne Micheèle. 2004. Heat kernels and Dirac operators. Springer.

[6]. Bhatia Rajendra, Gaubert Stephane, and Jain Tanvi. 2019. Matrix versions of the Hellinger distance. Letters in Mathematical Physics (2019), 1777–1804.

[7]. Bhatia Rajendra, Jain Tanvi, and Lim Yongdo. 2019. On the Bures-Wasserstein distance between positive definite matrices. Expositiones Mathematicae (2019).

[8]. Bhavsar Ronakben, Sun Yi, Helian Na, Davey Neil, Mayor David, and Steffert Tony. 2018. The Correlation between EEG Signals as Measured in Different Positions on Scalp Varying with Distance. Procedia Computer Science (2018), 92–97.

[9]. Bickel Peter J. and Li Bo. 2007. Local Polynomial Regression on Unknown Manifolds. Lecture Notes-Monograph Series (2007), 177–186.

[10]. Brooks Daniel A., Schwander Olivier, Barbaresco Frédéric, Schneider Jean-Yves, and Cord Matthieu. 2019. Riemannian batch normalization for SPD neura networks. In NeurIPS. 15463–15474.

[11]. Cameron Craddock, Yassine Benhajali, Carlton Chu, Francois Chouinard, Alan E. Aykan, András Jakab, Budhachandra Khundrakpam, John Lewis, Qingyang Liub, Michael Milham, Chaogan

Yan, and Pierre Bellec. 2013. The Neuro Bureau Preprocessing Initiative: open sharing of preprocessed neuroimaging data and derivatives. Frontiers in Neuroinformatics (2013).

[12]. Carlen Eric. 2010. Trace inequalities and quantum entropy: an introductory course. In Contemporary Mathematics. American Mathematical Society.

[13]. Casey BJ, Cannonier Tariq, and Conley May I.et al. 2018. The Adolescent Brain Cognitive Development (ABCD) study: Imaging acquisition across 21 sites. Developmental Cognitive Neuroscience (2018), 43–54. [PubMed: 29567376]

[14]. Chakraborty Rudrasis, Bouza Jose, Manton Jonathan H., and Vemuri Baba C.. 2022. ManifoldNet: A Deep Neural Network for Manifold-Valued Data With Applications. TPAMI 2 (2022), 799–810.

[15]. Chefd'hotel C, Tschumperlé D, Deriche R, and Faugeras O. 2004. Regularizing Flows for Constrained Matrix-Valued Images. Fournal of Mathematical Imaging and Vision 1/2 (2004), 147–162.

[16]. Chen Junru, Yang Yang, Yu Tao, Fan Yingying, Mo Xiaolong, and Yang Carl 2022. BrainNet: Epileptic Wave Detection from SEEG with Hierarchical Graph Diffusion Learning. In KDD. 2741–2751.

[17]. Chen Jiaao, Yang Zichao, and Yang Diyi. 2020. MixText: Linguistically-Informed Interpolation of Hidden Space for Semi-Supervised Text Classification. In Proc of ACL. Association for Computational Linguistics, 2147–2157.

[18]. Chen Shuxiao, Dobriban Edgar, and Lee Jane H.. 2020. A Group-Theoretic Framework for Data Augmentation. In NeurIPS

[19]. Cheng Philip E.. 1990. Applications of kernel regression estimation:survey. Communications in Statistics - Theory and Methods 11 (1990), 4103–4134

[20]. Chou Hsin-Ping, Chang Shih-Chieh, Pan Jia-Yu, Wei Wei, and Juan Da-Cheng. 2020. Remix: rebalanced mixup. In ECCV. Springer, 95–110.

[21]. Chu Xiangxiang, Tian Zhi, Wang Yuqing, Zhang Bo, Ren Haibing, Wei Xiaolin, Xia Huaxia, and Shen Chunhua. 2021. Twins: Revisiting the Design of Spatial Attention in Vision Transformers. In NeurIPS. 9355–9366.

[22]. Congedo Marco and Barachant Alexandre. 2015. A special form of SPD covariance matrix for interpretation and visualization of data manipulated with Riemannian geometry. 495–503

[23]. Craciun Gheorghe and Feinberg Martin. 2006. Multiple equilibria in complex chemical reaction networks: II. The species-reaction graph. SIAM J. Appl. Math 4 (2006), 1321–1338.

[24]. R Cameron Craddock G James Andrew, Holtzheimer Paul E III, Hu Xiaoping P, and Mayberg Helen S. 2012. A whole brain fMRI atlas generated via spatially constrained spectral clustering. Human Brain Mapping (2012), 1914–1928. [PubMed: 21769991]

[25]. Cui Hejie, Dai Wei, Zhu Yanqiao, Kan Xuan, Chen Gu Antonio Aodong, Lukemire Joshua, Zhan Liang, He Lifang, Guo Ying, and Yang Carl. 2023. BrainGB: A Benchmark for Brain Network Analysis With Graph Neural Networks. IEEE Transactions on Medical Imaging 42, 2 (2023), 493–506. [PubMed: 36318557]

[26]. Cui Hejie, Dai Wei, Zhu Yanqiao, Li Xiaoxiao, He Lifang, and Yang Carl. 2022. Interpretable Graph Neural Networks for Connectome-Based Brain Disorder Analysis. In MICCAI.

[27]. Cui Hejie, Lu Jiaying, Wang Shiyu, Xu Ran, Ma Wenjing, Yu Shaojun, et al. 2023. A Survey on Knowledge Graphs for Healthcare: Resources, Application Progress, and Promise. arXiv (2023).

[28]. Dabouei Ali, Soleymani Sobhan, Taherkhani Fariborz, and Nasrabadi Nasser M.. 2021. SuperMix: Supervising the Mixing Data Augmentation. In CVPR. 13794–13803.

[29]. Dai Wei, Cui Hejie, Kan Xuan, Guo Ying, and Yang Carl. 2022. Transformer-Based Hierarchical Clustering for Brain Network Analysis. In 2022 IEEE International Conference on Big Data (Big Data). 4970–4971.

[30]. Dodero Luca, Hà Quang Minh Marco San Biagio, Murino Vittorio, and Sona Diego. 2015. Kernel-based classification for brain connectivity graphs on the Riemannian manifold of positive definite matrices. In ISBI. 42–45.

[31]. Dryden Ian L., Koloydenko Alexey, and Zhou Diwei. 2009. Non-Euclidean statistics for covariance matrices, with applications to diffusion tensor imaging. The Annals of Applied Statistics 3 (2009), 1102–1123.

[32]. Du Yuanqi, Wang Shiyu, Guo Xiaojie, Cao Hengning, Hu Shujie, Jiang Junji, Varala Aishwarya, Angirekula Abhinav, and Zhao Liang. 2021. Graphgt: Machine learning datasets for graph generation and transformation. In NeurIPS.

[33]. Feddern Christian, Weickert Joachim, Burgeth Bernhard, and Welk Martin. 2006. Curvature-Driven PDE Methods for Matrix-Valued Images. IJCV 1 (2006), 93–107.

[34]. Fletcher Thomas. 2011. Geodesic Regression on Riemannian Manifolds. In Proceedings of the Third International Workshop on Mathematical Foundations of Computational Anatomy. 75–86.

[35]. Gallier Jean and Quaintance Jocelyn. 2020. Differential Geometry and Lie Groups: A Computational Perspective. Springer International Publishing.

[36]. Glasser Matthew F., Sotiropoulos Stamatios N., Wilson J. Anthony, Coalson Timothy S., Fischl Bruce, Andersson Jesper L., Xu Junqian, Jbabdi Saad, Webster Matthew, Polimeni Jonathan R., Van Essen David C., and Jenkinson Mark. 2013. The minimal preprocessing pipelines for the Human Connectome Project. Neurolmage (2013), 105–124.

[37]. Guo Hongyu, Mao Yongyi, and Zhang Richong. 2019. MixUp as Locally Linear Out-of-Manifold Regularization. In AAAI. AAAI Press, 3714–3722.

[38]. Hamilton William L., Ying Zhitao, and Leskovec Jure. 2017. Inductive Representation Learning on Large Graphs. In NeurIPS. 1024–1034.

[39]. Han Xiaotian, Jiang Zhimeng, Liu Ninghao, and Hu Xia. 2022. G-Mixup: Graph Data Augmentation for Graph Classification. In ICML. PMLR, 8230–8248.

[40]. Härdle Wolfgang. 1990. Applied nonparametric regression. Number 19.

[41]. He Wenchong, Jiang Zhe, Zhang Chengming, and Sainju Arpan Man. 2020. CurvaNet: Geometric Deep Learning based on Directional Curvature for 3D Shape Analysis. In KDD. ACM, 2214–2224.

[42]. Hendrycks Dan, Zou Andy, Mazeika Mantas, Tang Leonard, Li Bo, Song Dawn, and Steinhardt Jacob. 2022. PixMix: Dreamlike Pictures Comprehensively Improve Safety Measures. In CVPR. IEEE, 16762–16771.

[43]. Huang Chao, Farewell Daniel, and Pan Jianxin. 2017. A calibration method for non-positive definite covariance matrix in multivariate data analysis. Journal of Multivariate Analysis (2017), 45–52.

[44]. Huang Shuai, Lah James J., Allen Jason W., and Qiu Deqiang. 2022. A probabilistic Bayesian approach to recover R2* map and phase images for quantitative susceptibility mapping. Magnetic Resonance in Medicine 88, 4 (2022), 1624–1642. [PubMed: 35672899]

[45]. Huang Shuai, James J Lah, Jason W Allen, and Deqiang Qiu. 2023. Robust quantitative susceptibility mapping via approximate message passing with parameter estimation. Magnetic Resonance in Medicine (2023).

[46]. Huang Zhiwu and Van Gool Luc. 2017. A Riemannian Network for SPD Matrix Learning. In Proc. of AAAI. AAAI Press, 2036–2042.

[47]. Hwang Seonghyeon and Whang Steven Euijong. 2021. MixRL: Data Mixing Augmentation for Regression using Reinforcement Learning. CoRR (2021).

[48]. Hwang Seong-Hyeon and Whang Steven Euijong. 2021. MixRL: Data mixing augmentation for regression using reinforcement learning. ArXiv preprint (2021).

[49]. Jayasumana Sadeep, Hartley Richard, Salzmann Mathieu, Li Hongdong, and Harandi Mehrtash. 2015. Kernel Methods on Riemannian Manifolds with Gaussian RBF Kernels. TPAMI 12 (2015), 2464–2477.

[50]. Jayasumana Sadeep, Hartley Richard I., Salzmann Mathieu, Li Hongdong, and Harandi Mehrtash Tafazzoli. 2013. Kernel Methods on the Riemannian Manifold of Symmetric Positive Definite Matrices. In CVPR. 73–80.

[51]. Kan Xuan, Cui Hejie, Lukemire Joshua, Guo Ying, and Yang Carl. 2022. Fbnetgen: Task-aware gnn-based fmri analysis via functional brain network generation. In MIDL. PMLR, 618–637.

[52]. Kan Xuan, Cui Hejie, and Yang Carl. 2021. Zero-shot scene graph relation prediction through commonsense knowledge integration. In Machine Learning and Knowledge Discovery in Databases. Springer, 466–482.

[53]. Kan Xuan, Dai Wei, Cui Hejie, Zhang Zilong, Guo Ying, and Yang Carl. 2022. BRAIN NETWORK TRANSFORMER. In NeurIPS.

[54]. Kawahara Jeremy, Brown Colin J., Miller Steven P., Booth Brian G., Chau Vann, Grunau Ruth E., Zwicker Jill G., and Hamarneh Ghassan. 2017. BrainNetCNN: Convolutional neural networks for brain networks; towards predicting neurodevelopment. NeuroImage (2017), 1038–1049. [PubMed: 27693612]

[55]. Kim Hyunwoo J., Bendlin Barbara B., Adluru Nagesh, Collins Maxwell D., Chung Moo K., Johnson Sterling C., Davidson Richard J., and Singh Vikas. 2014. Multivariate General Linear Models (MGLM) on Riemannian Manifolds with Applications to Statistical Analysis of Diffusion Weighted Images. In CVPR. IEEE Computer Society, 2705–2712

[56]. Kim Jang-Hyun, Choo Wonho, and Oh Song Hyun. 2020. Puzzle Mix: Exploiting Saliency and Local Statistics for Optimal Mixup. In ICML. PMLR, 5275–5285.

[57]. Lee John M.. 2018. Introduction to Riemannian Manifolds. Springer International Publishing.

[58]. Li Xiaoxiao, Dvornek Nicha C., Zhou Yuan, Zhuang Juntang, Ventola Pamela, and Duncan James S.. 2019. Graph Neural Network for Interpreting Task-fMRI Biomarkers. In MICCAI.

[59]. Li Xiaoxiao, Zhou Yuan, Gao Siyuan, Dvornek Nicha, Zhang Muhan, Zhuang Juntang, Gu Shi, Scheinost Dustin, Staib Lawrence, Ventola Pamela, et al. 2021 BrainGNN: Interpretable Brain Graph Neural Network for fMRI Analysis. Medical Image Analysis (2021).

[60]. Li Zimeng, Zhu Shichao, Shao Bin, Zeng Xiangxiang, Wang Tong, and Liu Tie-Yan. 2023. DSN-DDI: an accurate and generalized framework for drug–drug interaction prediction by dual-view representation learning. Briefings in Bioinformatics 1 (2023).

[61]. Lin Qixiang, Shahid Salman, Antoine Hone-Blanchet Shuai Huang, Wu Junjie, Bisht Aditya, Loring David, Goldstein Felicia, Levey Allan, Crosson Bruce, et al. 2023. Magnetic resonance evidence of increased iron content in subcortical brain regions in asymptomatic Alzheimer's disease. Human Brain Mapping (2023).

[62]. Lin Sikun, Tang Shuyun, Grafton Scott T, and Singh Ambuj K. 2022. Deep Representations for Time-varying Brain Datasets. In KDD. 999–1009.

[63]. Meng Linghui, Xu Jin, Tan Xu, Wang Jindong, Qin Tao, and Xu Bo. 2021. Mixspeech: Data augmentation for low-resource automatic speech recognition. In ICASSP 2021. IEEE, 7008–7012.

[64]. Monti Federico, Boscaini Davide, Masci Jonathan, Emanuele Rodolà Jan Svoboda, and Bronstein Michael M.. 2017. Geometric Deep Learning on Graphs and Manifolds Using Mixture Model CNNs. In CVPR. 5425–5434.

[65]. Nielsen Michael A. and Chuang Isaac L.. 2010. Quantum computation and quantum information (10th anniversary ed ed.). Cambridge University Press.

[66]. Paci Paola, Fiscon Giulia, Conte Federica, Wang Rui-Sheng, Farina Lorenzo, and Loscalzo Joseph. 2021. Gene co-expression in the interactome: moving from correlation toward causation via an integrated approach to disease module discovery. NPJ systems biology and applications 1 (2021), 1–11.

[67]. Pan Yue-Ting, Chou Jing-Lun, and Wei Chun-Shu. 2022. MAtt: A Manifold Attention Network for EEG Decoding. NeurIPS (2022)

[68]. Paszke Adam, Gross Sam, Massa Francisco, Lerer Adam, Bradbury James, Chanan Gregory, Killeen Trevor, Lin Zeming, Gimelshein Natalia, Antiga Luca, Desmaison Alban, Andreas Köpf Edward Yang, Zachary DeVito Martin Raison, Tejani Alykhan, Chilamkurthy Sasank, Steiner Benoit, Fang Lu, Bai Junjie, and Chintala Soumith. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In NeurIPS. 8024–8035.

[69]. Pennec Xavier. 2009. Statistical Computing on Manifolds: From Riemannian Geometry to Computational Anatomy. Springer Berlin Heidelberg, 347–386.

[70]. Roberts Daniel A.. 2022. The principles of deep learning theory: an effective theory approach to understanding neural networks. Cambridge University Press.

[71]. Rong Yu, Huang Wenbing, Xu Tingyang, and Huang Junzhou. 2020. DropEdge: Towards Deep Graph Convolutional Networks on Node Classification. In Proc. of ICLR. OpenReview.net.

[72]. Sakai Takashi. 1996. Riemannian Manifolds.

[73]. Satterthwaite Theodore D., Elliott Mark A., Ruparel Kosha, Loughead James, Prabhakaran Karthik, Calkins Monica E., Hopson Ryan, Jackson Chad, Keefe Jack, Riley Marisa, Mentch Frank D., Sleiman Patrick, Verma Ragini, Davatzikos Christos, Hakonarson Hakon, Gur Ruben

C., and Gur Raquel E.. 2014. Neuroimaging of the Philadelphia Neurodevelopmental Cohort. NeuroImage (2014), 544–553.

[74]. Shawe-Taylor Johnand Cristianini Nello. 2004. Kernel methods for pattern analysis. Cambridge University Press.

[75]. Sean L Simpson F Bowman DuBois, and Laurienti Paul J. 2013. Analyzing complex functional brain networks: fusing statistics and network science to understand the brain. Statistics Surveys (2013), 1. [PubMed: 25309643]

[76]. Smith Stephen M., Miller Karla L., Gholamreza Salimi-Khorshidi Matthew Webster, Beckmann Christian F., Nichols Thomas E., Ramsey Joseph D., and Woolrich Mark W. 2011. Network modelling methods for FMRI. NeuroImage (2011).

[77]. Suh Yoon-Je and Kim Byung Hyung. 2021. Riemannian Embedding Banks for Common Spatial Patterns with EEG-based SPD Neural Networks. In AAAI. 854–862.

[78]. Thanwerdas Yann and Pennec Xavier. 2023. O(n)-invariant Riemannian metrics on SPD matrices. Linear Algebra Appl. (2023), 163–201.

[79]. Venkataramanan Shashanka, Kijak Ewa, Amsaleg Laurent, and Avrithis Yannis. 2022. AlignMixup: Improving Representations By Interpolating Aligned Features. In CVPR. 19174–19183

[80]. Verma Vikas, Lamb Alex, Beckham Christopher, Najafi Amir, Mitliagkas Ioannis, Lopez-Paz David, and Bengio Yoshua. 2019. Manifold Mixup: Better Representations by Interpolating Hidden States. In Proc. of ICML PMLR, 6438–6447.

[81]. Wang Shiyu, Bai Guangji, Zhu Qingyang, Qin Zhaohui, and Zhao Liang. 2023. Domain Generalization Deep Graph Transformation. arXiv:2305.11389 [cs.LG]

[82]. Wang Shiyu, Guo Xiaojie, Lin Xuanyang, Pan Bo, Du Yuanqi, Wang Yinkai, Ye Yanfang, Petersen Ashley, Leitgeb Austin, Alkhalifa Saleh, Minbiole Kevin, Wuest William M., Shehu Amarda, and Zhao Liang. 2022. Multi-objective Deep Data Generation with Correlated Property Control. In NeurIPS.

[83]. Wang Yikai, Kang Jian, Kemmer Phebe B., and Guo Ying. 2016. An Efficient and Reliable Statistical Method for Estimating Functional Connectivity in Large Scale Brain Networks Using Partial Correlation. Frontiers in Neuroscience (2016).

[84]. Wang Yiwei, Wang Wei, Liang Yuxuan, Cai Yujun, and Hooi Bryan. 2021. Mixup for node and graph classification. In the Web Conference. 3663–3674.

[85]. Wu Lirong, Lin Haitao, Gao Zhangyang, Tan Cheng, Li Stan, et al. 2021. Graphmixup: Improving class-imbalanced node classification on graphs by self-supervised context prediction. ArXiv preprint (2021).

[86]. Xu Keyulu, Hu Weihua, Leskovec Jure, and Jegelka Stefanie. 2019. How Powerful are Graph Neural Networks? In Proc. of ICLR. OpenReview.net.

[87]. Xu Ran, Yu Yue, Cui Hejie, Kan Xuan, Zhu Yanqiao, Ho Joyce, Zhang Chao, and Yang Carl. 2023. Neighborhood-Regularized Self-Training for Learning with Few Labels. In AAAI Conference on Artificial Intelligence.

[88]. Xu Ran, Yu Yue, Ho Joyce C, and Yang Carl. 2023. Weakly-Supervised Scientific Document Classification via Retrieval-Augmented Multi-Stage Training. In SIGIR.

[89]. Xu Ran, Yu Yue, Zhang Chao, Ali Mohammed K, Ho Joyce C, and Yang Carl. 2022. Counterfactual and factual reasoning over hypergraphs for interpretable clinical predictions on ehr. In Machine Learning for Health. PMLR, 259–278.

[90]. Yan Yujun, Zhu Jiong, Duda Marlena, Solarz Eric, Sripada Chandra, and Koutra Danai. 2019. GroupINN: Grouping-based Interpretable Neural Network-based Classification of Limited, Noisy Brain Data. In KDD.

[91]. Yang Yi, Cui Hejie, and Yang Carl. 2023. PTGB: Pre-Train Graph Neural Networks for Brain Network Analysis. In The Conference on Health, Inference, and Learning.

[92]. Yang Yi, Zhu Yanqiao, Cui Hejie, Kan Xuan, He Lifang, Guo Ying, and Yang Carl. 2022. Data-Efficient Brain Connectome Analysis via Multi-Task Meta-Learning. In KDD ‘22 (Washington DC, USA). New York, NY, USA, 4743–4751.

[93]. Yao Huaxiu, Wang Yiping, Zhang Linjun, Zou James, and Finn Chelsea. 2022. C-Mixup: Improving Generalization in Regression. In NeurIPS.

[94]. Yeniay Özgür. 2005. Penalty Function Methods for Constrained Optimization with Genetic Algorithms. Mathematical and Computational Applications (2005).

[95]. You Kisung and Park Hae-Jeong. 2022. Geometric learning of functional brain network on the correlation manifold. Scientific Reports (2022), 1–13. [PubMed: 34992227]

[96]. Yu Haiyuan, Philip M Kim Emmett Sprecher, Trifonov Valery, and Gerstein Mark. 2007. The importance of bottlenecks in protein networks: correlation with gene essentiality and expression dynamics. PLoS computational biology 4 (2007).

[97]. Yu Yue, Huang Kexin, Zhang Chao, Lucas M Glass Jimeng Sun, and Xiao Cao. 2021. SumGNN: multi-typed drug interaction prediction via efficient knowledge graph summarization. Bioinformatics 18 (2021), 2988–2995.

[98]. Yu Yue, Kan Xuan, Cui Hejie, Xu Ran, Zheng Yujia, Song Xiangchen, Zhu Yanqiao, Zhang Kun, et al. 2022. Learning Task-Aware Effective Brain Connectivity for fMRI Analysis with Graph Neural Networks. ArXiv preprint (2022).

[99]. Yu Yue, Zhang Rongzhi, Xu Ran, Zhang Jieyu, Shen Jiaming, and Zhang Chao. 2022. Cold-start data selection for few-shot language model fine-tuning: A prompt-based uncertainty propagation approach. ArXiv preprint (2022).

[100]. Yun Sangdoo, Han Dongyoon, Chun Sanghyuk, Seong Joon Oh Youngjoon Yoo, and Choe Junsuk. 2019. CutMix: Regularization Strategy to Train Strong Classifiers With Localizable Features. In ICCV. IEEE, 6022–6031.

[101]. Zhang Hongyi, Moustapha Cissé Yann N. Dauphin, and Lopez-Paz David. 2018. mixup: Beyond Empirical Risk Minimization. In Proc. of ICLR. OpenReview.net.

[102]. Zhang Rongzhi, Yu Yue, Shen Jiaming, Cui Xiquan, and Zhang Chao. 2023. Local Boosting for Weakly-supervised Learning. In KDD.

[103]. Zhang Rongzhi, Yu Yue, and Zhang Chao. 2020. SeqMix: Augmenting Active Sequence Labeling via Sequence Mixup. In Proc. of EMNLP. Association for Computational Linguistics, 8566–8579.

[104]. Zhang Shaofeng, Liu Meng, Yan Junchi, Zhang Hengrui, Huang Lingxiao, Yang Xiaokang, and Lu Pinyan. 2022. M-Mix: Generating Hard Negatives via Multi-sample Mixing for Contrastive Learning. In KDD. 2461–2470.

CCS CONCEPTS

• Computing methodologies → Regularization; • Applied computing → Biological networks.

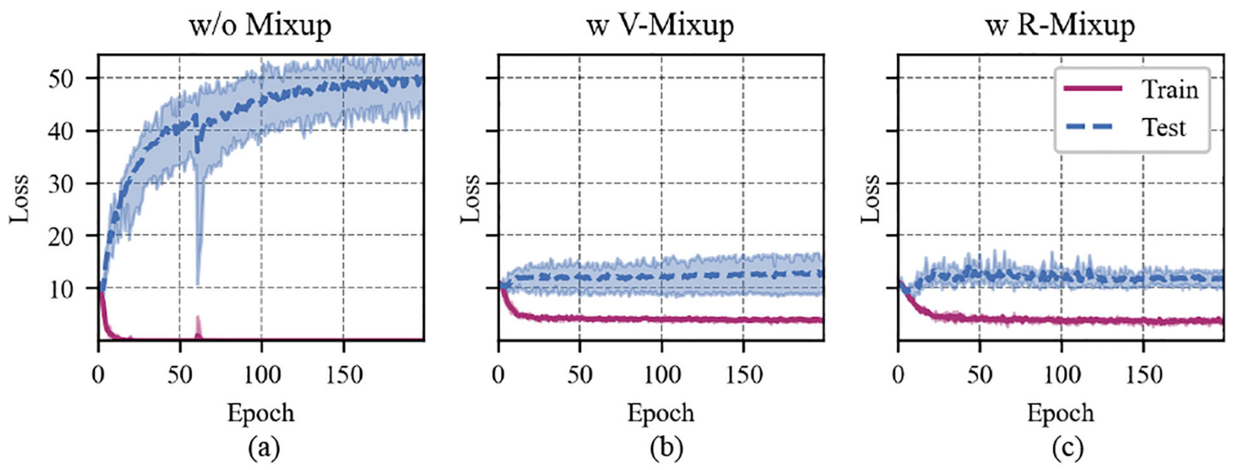


Figure 1:
Train/Test performance of a Transformer on the biological network dataset PNC with 503 samples. Each sample is represented as a 120 × 120 adjacency matrix. V-Mixup is the vanilla Mixup and R-Mixup is our proposed method.

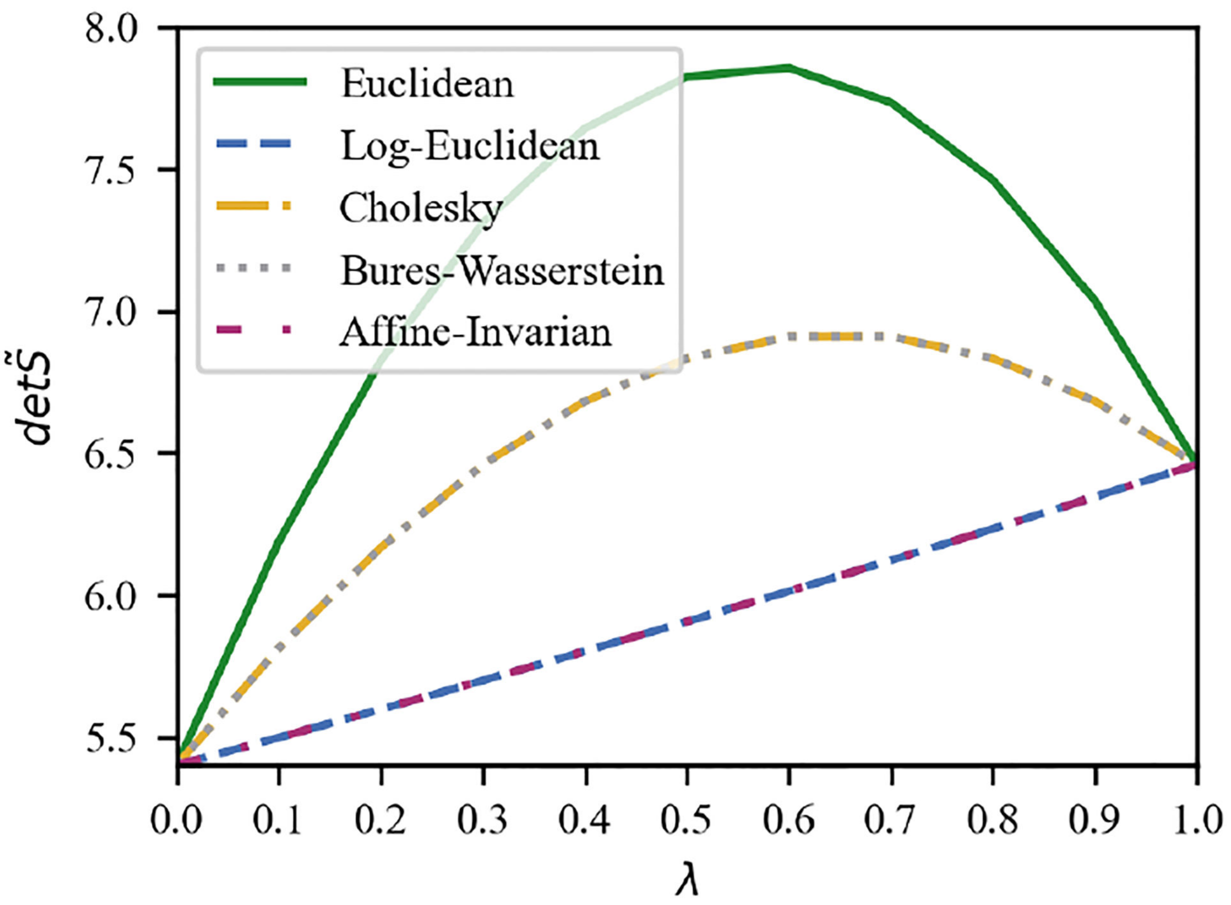


Figure 2:

The *swelling* effect of Mixing up with different metrics. $\widetilde{S}$ is the augmented sample mixed by samples S_i and S_j, where det $S_i = 5.40$ and det $S_j = 6.46$. Ideally, the determinant of the mixed sample $\widetilde{S}$ should be between det S_i and det S_j. The results indicate that mixing samples with Euclidean (widely used in existing Mixup methods), Cholesky, and Bures-Wasserstein metrics leads to unphysical inflations.

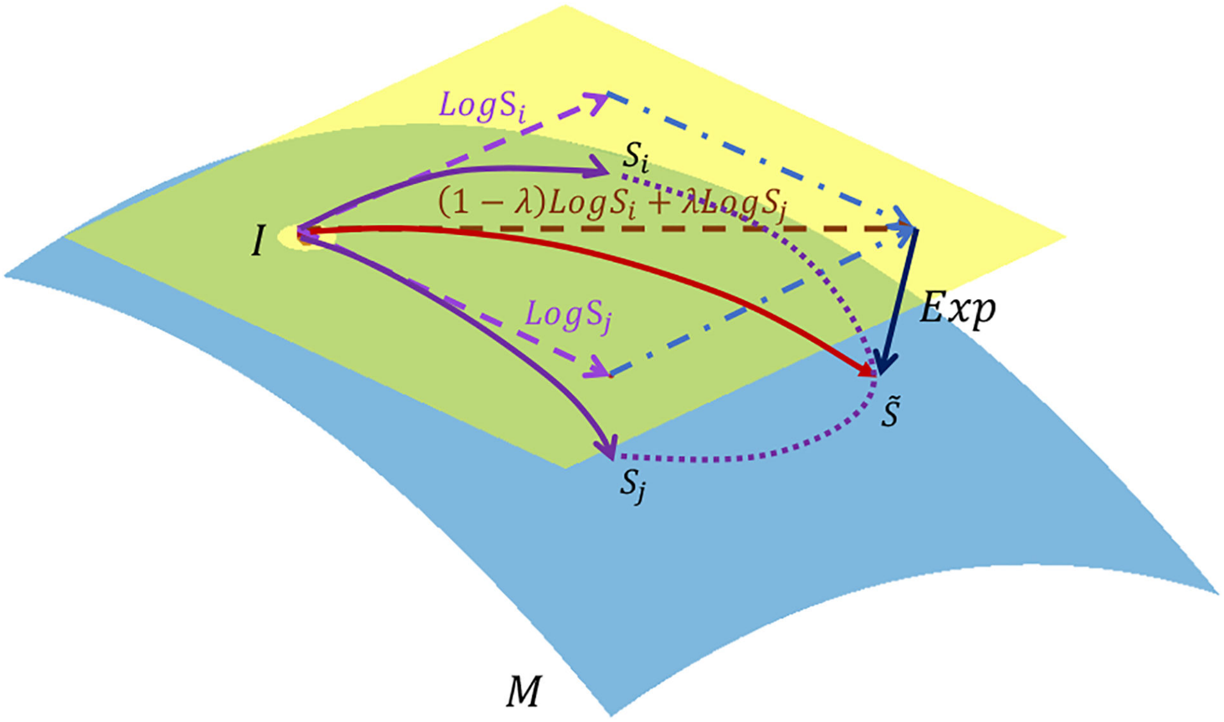


Figure 3:

The process of R-Mixup generating sample $\widetilde{S}$, where the blue surface M represents the *Riemannian manifold* and the yellow plane is the tangent plane of M at the origin I . S_i, S_j are the original samples in M, and $\log S_i, \log S_j$ are tangent vectors. R-Mixup creates the augmented sample $\widetilde{S}$ by combining the initial tangent vectors of both trajectories connecting I with S_i, S_j, i.e., $(1-\lambda)\log S_i + \lambda \log S_j$, and push it back to the *Riemannian manifold* M via exponential map.

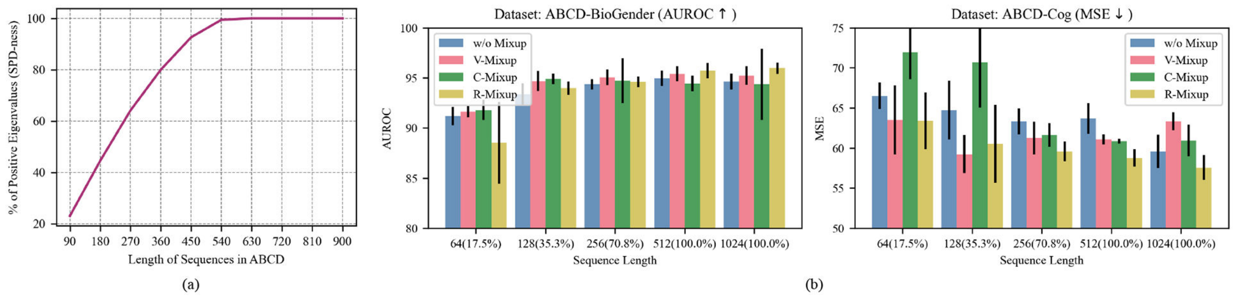


Figure 4:
(a) The influence of time-series sequence length *t* on the percentage of the positive eigenvalues (%). (b) The influence of the sequence length *t* or *SPD-ness* (%) on the prediction performance of classification and regression tasks.

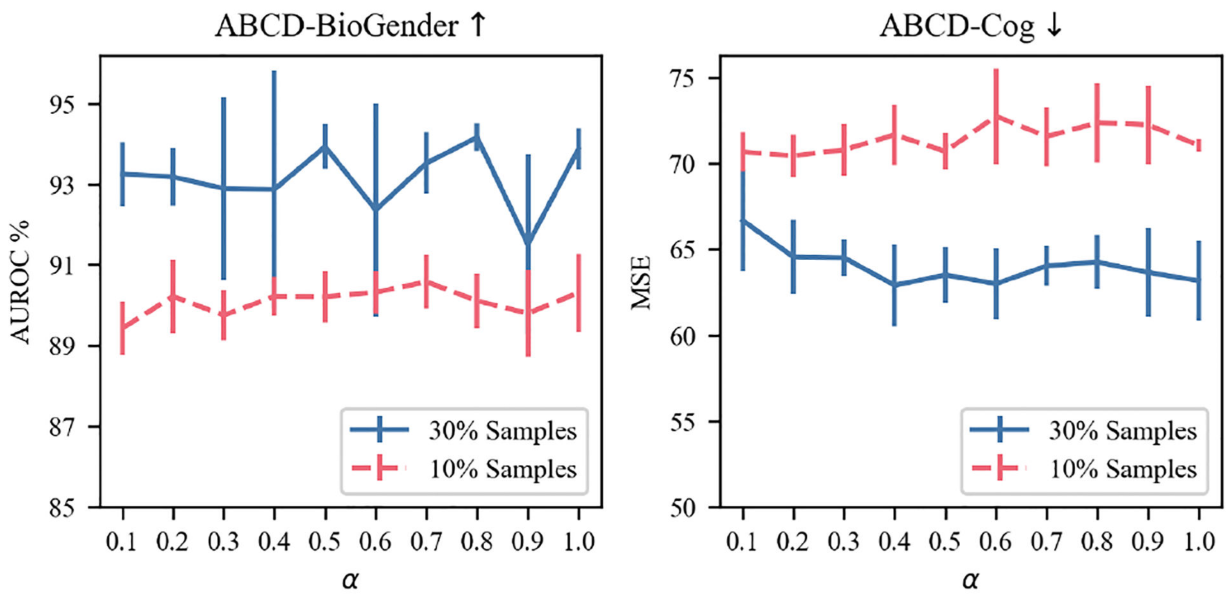


Figure 5:
The influence of the key hyperparameter ($\boldsymbol{\alpha}$) value on the performance of classification and regression tasks.

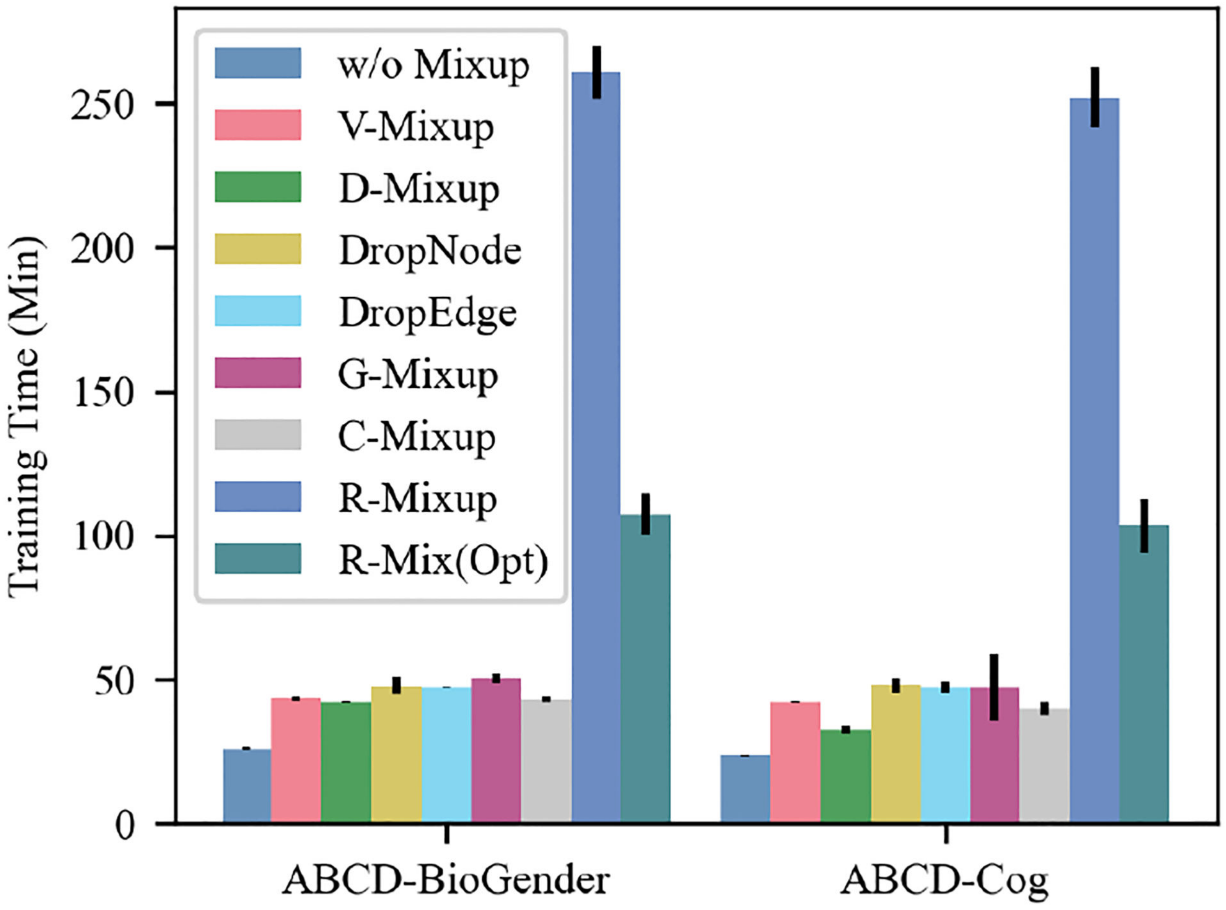


Figure 6:

Training Time of different Mixup methods on the large ABCD dataset. R-MIXUP is the original model while R-Mix(Opt) is time-optimized as discussed in Section 3.5.

Table 1:

Comparison of Different Metrics Choices

Metric	Geodesics	Swelling Effect	Numerical Stability
Euclidean	$(1-\lambda)S_i + \lambda S_j$	Yes	Stable
Cholesky	$((1-\lambda)L_i + \lambda L_j)((1-\lambda)L_i + \lambda L_j)^T$	Yes	Stable
Bures-Wasserstein	$(1-\lambda)^2 S_i + \lambda^2 S_j + \lambda(1-\lambda)\left((S_i S_j)^{1/2} + (S_j S_i)^{1/2}\right)$	Yes	Unstable
Affine-invariant	$S_i^{1/2}\left(S_i^{-1/2} S_j S_i^{-1/2}\right)^{\lambda} S_i^{1/2}$	No	Unstable
Log-Euclidean	$\exp((1-\lambda)\log S_i + \lambda \log S_j)$	**No**	**Stable**

Table 2:

Dataset Summary.

Dataset	Sample Size	Variance Number (n)	Sequence Length (t)	Task	Class Number
ABCD-BioGender	7901	360	Variable Length	Classification	2
ABCD-Cog	7749	360	Variable Length	Regression	-
PNC	503	120	120	Classification	2
ABIDE	1009	100	100	Classification	2
TCGA-Cancer	240	50	50	Classification	24

Table 3:

Overall performance comparison based on the Transformer backbone. The best results are in bold, and the second best results are underlined. The ↑ indicates a higher metric value is better and ↓ indicates a lower one is better.

Method	ABCD-BioGender		ABCD-Cog	PNC		ABIDE		TCGA-Cancer	
	AUROC↑	Accuracy↑	MSE↓	AUROC↑	Accuracy↑	AUROC↑	Accuracy↑	Precision↑	Recall↑
w/o Mixup	95.28±0.32	87.68±1.31	60.21±1.53	74.85±4.93	66.57±6.29	73.32±4.11	66.00±3.66	35.33±11.52	45.00±10.79
V-Mixup	95.85±0.63	87.86±1.45	60.43±2.67	76.02±2.54	65.88±7.89	**75.03±5.04**	66.80±5.40	69.58±9.39	77.50±6.97
D-Mixup	94.55±2.84	87.17±3.45	60.96±1.82	76.15±4.58	68.82±6.29	72.92±4.93	67.40±5.64	70.28±12.30	75.83±12.98
DropNode	95.65±0.35	88.07±0.76	65.35±2.97	75.47±4.27	67.45±4.35	73.49±4.09	66.00±3.16	53.96±11.34	61.67±10.79
DropEdge	95.28±0.39	87.54±0.60	76.44±1.82	72.89±5.70	66.27±5.31	70.68±6.14	64.20±5.12	67.57±5.14	75.00±5.10
G-Mixup	95.24±0.92	88.16±0.63	62.16±2.04	76.01±3.04	69.41±3.21	73.68±5.67	65.60±4.56	59.72±7.77	69.44±6.27
C-Mixup	96.01±0.48	88.40±1.44	59.68±1.15	75.29±2.52	69.02±5.48	74.69±4.40	66.40±3.36	67.50±6.90	76.67±6.32
R-MIXUP	**96.20±0.33**	**89.44±1.06**	**56.89±1.66**	**77.01±2.59**	**69.80±3.63**	74.79±4.90	**68.20±4.19**	**71.39±9.59**	**78.33±9.03**

Table 4:

Detailed performance comparison of different sample sizes with Transformer as the backbone.

Percentage (in %)	Dataset: ABCD-BioGender (AUROC↑)				Dataset: ABCD-Cog (MSE↓)			
	w/o Mixup	V-Mixup	C-Mixup	R-Mixup	w/o Mixup	V-Mixup	C-Mixup	R-Mixup
10	87.14±1.15	88.99±0.75	88.72±1.13	**90.21±0.64**	73.07±2.75	77.00±4.58	71.22±1.68	**70.69±1.06**
20	90.60±0.91	91.11±0.54	91.49±0.89	**92.72±0.64**	69.70±2.75	69.80±2.42	69.30±3.21	**66.50±2.50**
30	92.60±0.51	93.45±0.35	93.33±0.78	**93.93±0.55**	65.97±2.48	65.84±1.11	64.31±0.57	**63.50±1.61**
40	92.84±0.40	94.06±0.48	93.95±0.53	**94.12±0.21**	63.91±4.07	63.14±1.08	61.88±2.93	**61.15±1.80**
50	94.18±0.51	**95.20±0.39**	95.03±0.57	94.78±0.98	61.89±3.85	63.45±1.65	61.26±1.31	**60.82±2.71**
60	94.22±0.44	95.19±0.54	95.17±0.32	**95.65±0.37**	59.47±1.59	60.32±0.94	60.20±1.58	**58.75±1.65**
70	94.18±0.40	**95.51±0.18**	95.49±0.28	95.07±0.18	62.35±2.28	61.15±1.51	60.54±3.57	**60.17±0.50**
80	95.18±0.31	95.60±0.42	95.73±0.51	**95.94±0.31**	59.85±1.47	60.31±1.07	60.85±3.84	**56.78±2.05**
90	95.55±0.86	**95.92±0.34**	95.49±0.73	95.24±0.65	61.17±3.36	61.51±0.78	60.35±0.93	**57.45±3.39**
100	95.28±0.32	95.85±0.63	96.01±0.48	**96.20±0.33**	60.21±1.53	60.43±2.67	59.68±1.15	**56.89±1.66**