R-Mixup: Riemannian Mixup for Biological Networks

Xuan Kan*
xuan.kan@emory.edu
Department of Computer Science, Emory University
Atlanta, GA, USA

Zimu Li*
lizm@mail.sustech.edu.cn
Pritzker School of Molecular Engineering, University of Chicago
Chicago, IL, USA

Hejie Cui
hejie.cui@emory.edu
Department of Computer Science, Emory University
Atlanta, GA, USA

Yue Yu
yueyu@gatech.edu
School of Computational Science and Engineering, Georgia Institute of Technology
Atlanta, GA, USA

Ran Xu
ran.xu@emory.edu
Department of Computer Science, Emory University
Atlanta, GA, USA

Shaojun Yu
shaojun.yu@emory.edu
Department of Computer Science, Emory University
Atlanta, GA, USA

Zilong Zhang
201957020@uibe.edu.cn
School of Statistics, University of International Business and Economics
Beijing, China

Ying Guo
yguo2@emory.edu
Department of Biostatistics and Bioinformatics, Emory University
Atlanta, GA, USA

Carl Yang†
j.carlyang@emory.edu
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 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 D.

CCS CONCEPTS

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

KEYWORDS

biological network, data augmentation, geometric deep learning

ACM Reference Format:
Xuan Kan, Zimu Li, Hejie Cui, Yue Yu, Ran Xu, Shaojun Yu, Zilong Zhang, Ying Guo, and Carl Yang. 2023. R-Mixup: Riemannian Mixup for Biological Networks. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '23), August 6–10, 2023, Long Beach, CA, USA.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3580305.3599483

*Both authors contributed equally to this research.
†To whom correspondence should be addressed.

KDD '23, August 6–10, 2023, Long Beach, CA, USA

ACM ISBN 979-8-4007-0103-0/23/08…$15.00
https://doi.org/10.1145/3580305.3599483

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.,

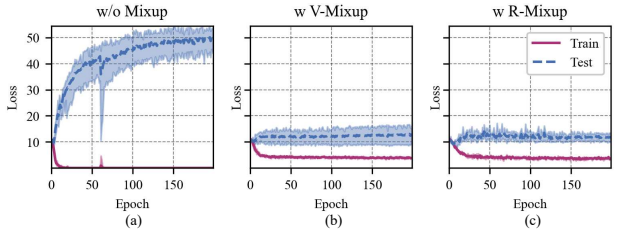

Figure 1: Train/Test performance of a Transformer on the biological network dataset PNC with 503 samples. Each sample is represented as a 120×120 adjacency matrix. V-Mixup is the vanilla Mixup and R-Mixup is our proposed method.

$O(n^2)$ 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.

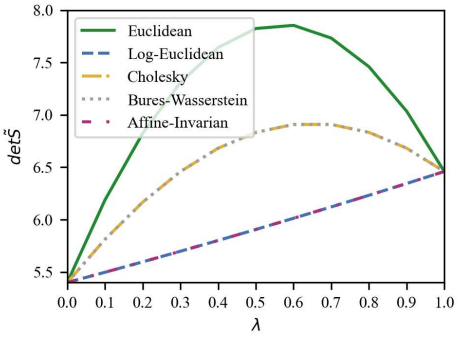

Figure 2: The *swelling effect* of Mixing up with different metrics. $\tilde{S}$ is the augmented sample mixed by samples S_i and S_j, where $\det S_i = 5.40$ and $\det S_j = 6.46$. Ideally, the determinant of the mixed sample $\tilde{S}$ should be between $\det S_i$ and $\det S_j$. The results indicate that mixing samples with Euclidean (widely used in existing Mixup methods), Cholesky, and Bures-Wasserstein metrics leads to unphysical inflations.

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 B. 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. 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 $S = \mathrm{Cor}(X) \in \mathbb{R}^{n\times n}$ 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 $u \in \mathbb{R}^n$, $u^T Su \geq 0$. 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 $\mathrm{Sym}(n)$ be the collection of all positive semi-definite matrices, and $\mathrm{Sym}^+(n)$ denotes the collection of all SPDs. The collection $\mathrm{Sym}(n)$ can be seen as an $\frac{1}{2}n(n-1)$-dimensional Euclidean space, but $\mathrm{Sym}^+(n) \subset \mathbb{R}^{n\times n}$ 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 $\mathrm{Sym}^+(n)$, the covariance matrix $\mathrm{Cor}(X)$ 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 B. 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 matrices 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 $(\tilde{S}, \tilde{y})$ are created through the straight line connecting S_i, S_j and y_i, y_j,

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

where $\lambda \sim \text{Beta}(\alpha, \alpha)$, 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 $\text{Sym}^+(n)$, 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:

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

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,

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

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

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

3.3 Comparison with Other Metrics

There are various choices of Riemannian metrics and hence different geodesics on $\text{Sym}^+(n)$ [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 $\text{Sym}^+(n)$ [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.

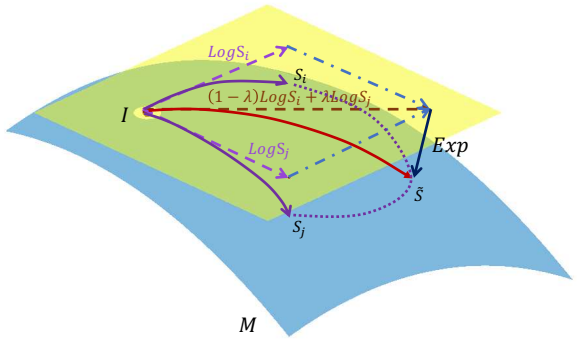

Figure 3: The process of R-Mixup generating sample $\tilde{S}$, where the blue surface M represents the *Riemannian manifold* and the yellow plane is the tangent plane of M at the origin I. S_i, S_j are the original samples in M, and $\log S_i, \log S_j$ are tangent vectors. R-Mixup creates the augmented sample $\tilde{S}$ 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.

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 $(S_i S_j)^{1/2}$, we notice the following fact:

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

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

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	$(1-\lambda)^2 S_i + \lambda^2 S_j + \lambda(1-\lambda)((S_i S_j)^{1/2} + (S_j S_i)^{1/2})$	Yes	Unstable
Affine-invariant	$S_i^{1/2}(S_i^{-1/2} S_j S_i^{-1/2})^{\lambda} S_i^{1/2}$	No	Unstable
Log-Euclidean	$\exp((1-\lambda)\log S_i + \lambda \log S_j)$	**No**	**Stable**

in Section 1 and 3.3. Mathematically, suppose $\det S_i \leq \det S_j$, then the determinant of $\tilde{S}$ defined by Eq.(2) satisfies:

$$\det S_i \leq \det \tilde{S} \leq \det S_j. \tag{8}$$

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 $\tilde{S}$, let $\tilde{m}(\tilde{S})$ 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*

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

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 non-zero 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 B.

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

$$\sum(\tilde{m}(\tilde{S}) - \tilde{y})^2 \leq (\tilde{m}(\tilde{S}') - \tilde{y})^2. \tag{12}$$

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

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

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

3.5 Time Complexity and Optimization

One potential concern of the proposed R-Mixup lies in its time-consuming 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 $\{\mu_1, ..., \mu_n\}$ of each sample. This precomputing process can reduce the three computations of eigenvalue decomposition to once for each sample. Formally,

$$\begin{aligned} \tilde{S} = \exp\Big((1-\lambda) O_i \mathrm{diag}(\log \mu_1, ..., \log \mu_n) O_i^T \\ + \lambda O_j \mathrm{diag}(\log \nu_1, ..., \log \nu_n) O_j^T\Big), \end{aligned} \tag{14}$$

where $O_i \mathrm{diag}(\mu_1, ..., \mu_n) O_i^T$ and $S_j = O_j \mathrm{diag}(\nu_1, ..., \nu_n) O_j^T$ 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

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

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.

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**

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

$$\tilde{S} = (1-\lambda)S_i + \lambda S_j, \tilde{y} = (1-\lambda)y_i + \lambda y_j, \quad \lambda \sim \text{Beta}(\alpha, \alpha). \tag{15}$$

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,

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

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

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

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

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

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,

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

For the setting of classification,

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

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

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

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

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**

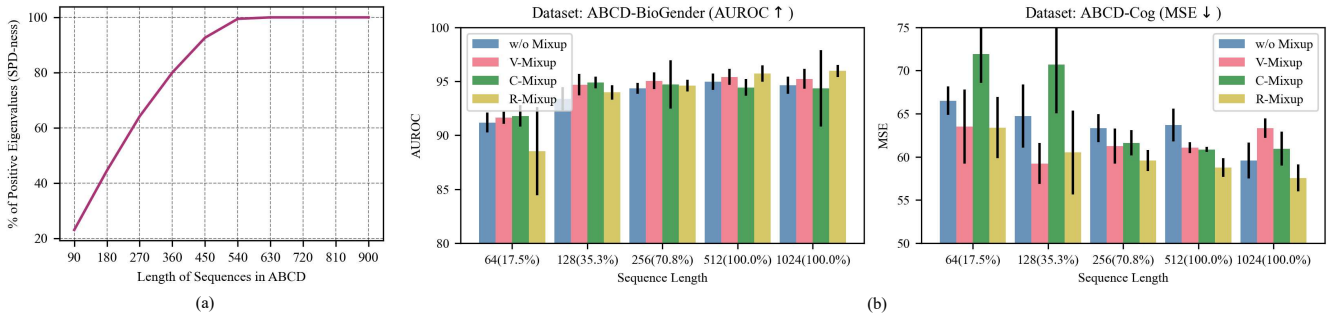

Figure 4: (a) The influence of time-series sequence length t on the percentage of the positive eigenvalues (%). (b) The influence of the sequence length t or *SPD-ness* (%) on the prediction performance of classification and regression tasks.

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.

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,

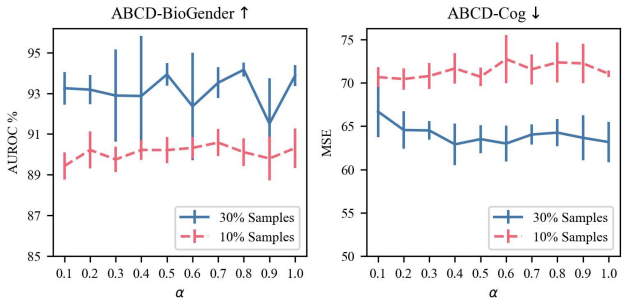

Figure 5: The influence of the key hyperparameter (α) value on the performance of classification and regression tasks.

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 C 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.

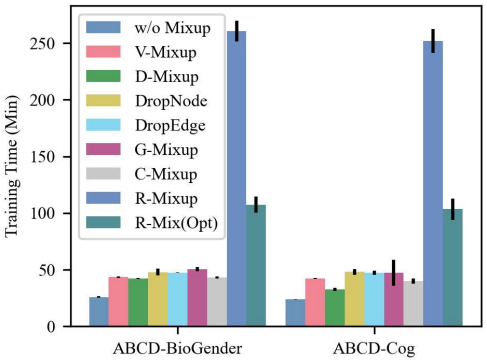

Figure 6: Training Time of different Mixup methods on the large ABCD dataset. R-Mixup is the original model while R-Mix(Opt) is time-optimized as discussed in Section 3.5.

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.

6 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.

REFERENCES

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

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

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

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

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

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

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

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

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

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

[11] Craddock Cameron, Benhajali Yassine, Chu Carlton, Chouinard Francois, E. Aykan Alan, Jakab András, Khundrakpam Budhachandra, Lewis John, Liub Qingyang, Milham Michael, Yan Chaogan, and Bellec Pierre. 2013. The Neuro Bureau Preprocessing Initiative: open sharing of preprocessed neuroimaging data and derivatives. *Frontiers in Neuroinformatics* (2013).

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

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

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

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

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

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

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

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

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

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

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

[23] Gheorghe Craciun and Martin Feinberg. 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 Andrew James, Paul E Holtzheimer III, Xiaoping P Hu, and Helen S Mayberg. 2012. A whole brain fMRI atlas generated via spatially constrained spectral clustering. *Human Brain Mapping* (2012), 1914–1928.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

[44] Shuai Huang, James J. Lah, Jason W. Allen, and Deqiang Qiu. 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.

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

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

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

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

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

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

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

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

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

[55] Hyunwoo J. Kim, Barbara B. Bendlin, Nagesh Adluru, Maxwell D. Collins, Moo K. Chung, Sterling C. Johnson, Richard J. Davidson, and Vikas Singh. 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] Jang-Hyun Kim, Wonho Choo, and Hyun Oh Song. 2020. Puzzle Mix: Exploiting Saliency and Local Statistics for Optimal Mixup. In *ICML*. PMLR, 5275–5285.

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

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

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

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

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

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

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

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

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

[66] Paola Paci, Giulia Fiscon, Federica Conte, Rui-Sheng Wang, Lorenzo Farina, and Joseph Loscalzo. 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] Yue-Ting Pan, Jing-Lun Chou, and Chun-Shu Wei. 2022. MAtt: A Manifold Attention Network for EEG Decoding. *NeurIPS* (2022).

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

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

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

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

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

[73] Theodore D. Satterthwaite, Mark A. Elliott, Kosha Ruparel, James Loughead, Karthik Prabhakaran, Monica E. Calkins, Ryan Hopson, Chad Jackson, Jack Keefe, Marisa Riley, Frank D. Mentch, Patrick Sleiman, Ragini Verma, Christos Davatzikos, Hakon Hakonarson, Ruben C. Gur, and Raquel E. Gur. 2014. Neuroimaging of the Philadelphia Neurodevelopmental Cohort. *NeuroImage* (2014), 544–553.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

A GCN BACKBONE PERFORMANCE

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

B PROOF DETAILS

Proof of Proposition 3.2

Proof. Let us consider column vectors $Y_k = (x_{ik} - E(X_i))$. Then $\mathrm{Cov}(X) = \frac{1}{t}\sum_k Y_k Y_k^T$. Given any vector $u \in \mathbb{R}^n$,

$$u^T \mathrm{Cov}(X) u = \frac{1}{t}\sum_i u^T Y_k Y_k^T u = \frac{1}{t}\sum_i (Y_k^T u)^2 \geq 0. \tag{22}$$

On the other hand,

$$\begin{aligned} u^T \mathrm{Cor}(X) u = u^T \mathrm{diag}(&\frac{1}{\sqrt{\mathrm{Cov}(X)_{11}}}, ..., \frac{1}{\sqrt{\mathrm{Cov}(X)_{vv}}})\cdot \\ &\mathrm{Cov}(X)\cdot \mathrm{diag}(\frac{1}{\sqrt{\mathrm{Cov}(X)_{11}}}, ..., \frac{1}{\sqrt{\mathrm{Cov}(X)_{vv}}}) u \\ &= \tilde{u}^T \mathrm{Cov}(X) \tilde{u} \geq 0. \end{aligned} \tag{23}$$

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 $\sum_i (Y_k^T u)^2 = 0$. □

Proof of Swelling Effects

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

$$\begin{aligned} &\det\left(\exp((1-\lambda)\log S_i + \lambda \log S_j)\right) \\ =&\exp\left((1-\lambda)\mathrm{Tr}\log S_i\right)\exp\left(\lambda \mathrm{Tr}\log S_j\right) \\ =&(\det S_i)^{1-\lambda}(\det S_2)^{\lambda}. \end{aligned} \tag{24}$$

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

$$\int \exp(-\frac{1}{2}\boldsymbol{x}^T S \boldsymbol{x}) d\boldsymbol{x} = \sqrt{\frac{(2\pi)^n}{\det S}}, \tag{25}$$

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

$$\begin{aligned} &\sqrt{\frac{(2\pi)^n}{\det\left(((1-\lambda)S_i + \lambda S_j)\right)}} \\ \leq&\Big(\int \exp(-\frac{1}{2}\boldsymbol{x}^T S_i \boldsymbol{x}) d\boldsymbol{x}\Big)^{1-\lambda}\Big(\int \exp(-\frac{1}{2}\boldsymbol{x}^T S_j \boldsymbol{x}) d\boldsymbol{x}\Big)^{\lambda} \\ =&\sqrt{\frac{(2\pi)^n}{(\det S_i)^{1-\lambda}(\det S_j)^{\lambda}}}. \end{aligned} \tag{26}$$

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

$$\begin{aligned} &\det\left(\exp((1-\lambda)\log S_i + \lambda \log S_j)\right) \\ =&(\det S_i)^{1-\lambda}(\det S_j)^{\lambda} \leq \det\left(((1-\lambda)S_i + \lambda S_j)\right). \end{aligned} \tag{27}$$

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

$$\begin{aligned} &(\frac{\det S_j}{\det S_i})^{\lambda} - 1 \leq (\frac{\det S_j}{\det S_i} - 1)\lambda \\ \Longrightarrow\ &(\det S_i)^{1-\lambda}(\det S_j)^{\lambda} \leq (\det S_i)(1-\lambda) + (\det S_j)\lambda, \end{aligned} \tag{28}$$

and finishes the proof. □

Proof of Theorem 3.3

Proof. The representation for Riemannian geodesic is

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

which is a straight line connecting $\log S_i$ and $\log S_j \in \mathrm{Sym}(n)$. As a contrary, the coordinate representation of straight line is now highly curved. 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 $\{(S_i, y_i) | i = 1, ..., N\}$ is merely solving the linear model of $\{(\log S_i, y_i) | i = 1, ..., N\}$. 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.(2). However, curves like Eq.(1) 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]:

$$\log((1-\lambda)S_i + \lambda S_j) \geq (1-\lambda)\log S_i + \lambda \log S_j. \tag{30}$$

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 the case for a pair of samples S_i, S_j as an augmented sample $\tilde{S}$ 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,

$$\begin{aligned} \tilde{m}(S) = \boldsymbol{y}^T G^{-1} \boldsymbol{K}_S &= (y_i, y_j)\begin{pmatrix} K_R(S_i, S_i) & K_R(S_i, S_j) \\ K_R(S_j, S_i) & K_R(S_i, S_i) \end{pmatrix}^{-1}\begin{pmatrix} K_R(S_i, S) \\ K_R(S_j, S) \end{pmatrix} \\ &= \frac{1}{1-K_{ij}^2}\Big((y_i - K_{ij}y_j)K_{i,S} + (y_j - K_{ij}y_i)K_{j,S}\Big), \end{aligned} \tag{31}$$

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

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

$$K_{ij} = \exp\left(-\frac{1}{2\sigma^2}\|\log S_i - \log S_j\|\right), \tag{32}$$

$$K_{i,\tilde{S}} = \exp\left(-\frac{1}{2\sigma^2}\lambda\|\log S_i - \log S_j\|\right) = K_{ij}^{\lambda} \tag{33}$$

$$K_{j,\tilde{S}} = \exp\left(-\frac{1}{2\sigma^2}(1-\lambda)\|\log S_i - \log S_j\|\right) = K_{ij}^{1-\lambda} \tag{34}$$

with

$$\tilde{m}(\tilde{S}) = \frac{1}{1-K_{ij}^2}\Big(K_{ij}^{\lambda}(y_i - K_{ij}y_j) + K_{ij}^{1-\lambda}(y_j - K_{ij}y_i)\Big) \tag{35}$$

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

$$\frac{d^2\tilde{m}(\tilde{S}(\lambda))}{d\lambda^2} = \frac{\ln^2 K_{ij}}{K_{ij}^{\lambda}(1-K_{ij}^2)}\Big((K_{ij} - K_{ij}^{2\lambda+1})y_j + (K_{ij}^{2\lambda} - K_{ij}^2)y_i\Big), \tag{36}$$

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

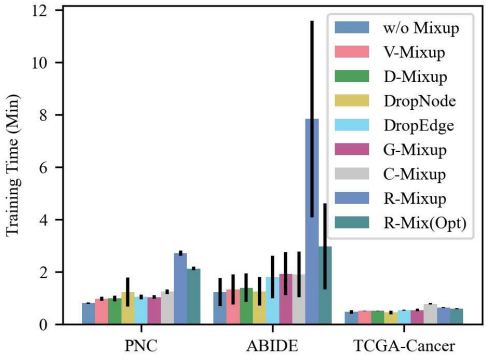

Figure 7: Running Time in PNC, ABIDE and TCGA-Cancer. R-Mixup is the original version of our method while R-Mix(Opt) is the proposed optimized version in Section 3.5.

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

$$K_{i,\tilde{S}'} = \exp\left(-\frac{1}{2\sigma^2}\|\log((1-\lambda)S_i - \lambda S_j) - \log S_i\|\right) \quad (37)$$

$$K_{j,\tilde{S}'} = \exp\left(-\frac{1}{2\sigma^2}\|\log((1-\lambda)S_i - \lambda S_j) - \log S_j\|\right) \quad (38)$$

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 $\tilde{m}(\tilde{S})$ and $\tilde{m}(\tilde{S}')$ based on their general expansion in Eq.(31). By (30),

$$\begin{aligned} &\log((1-\lambda)S_i + \lambda S_j) - \log S_i \geq \lambda(\log S_i + \log S_j) \\ \Longrightarrow\ &\|\log((1-\lambda)S_i + \lambda S_j) - \log S_i\| \geq \|\lambda(\log S_i + \log S_j)\| \\ \Longrightarrow\ &K_{i,\tilde{S}'} = \exp\left(-\frac{1}{2\sigma^2}\|\log((1-\lambda)S_i - \lambda S_j) - \log S_i\|\right) \quad (39) \\ &\leq \exp\left(-\frac{1}{2\sigma^2}\lambda\|\log S_i - \log S_j\|\right) = K_{i,\tilde{S}}. \end{aligned}$$

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 $\tilde{m}(\tilde{S})$, Eq.(31) and the range of our labels, we conclude that

$$0 \leq \tilde{m}(\tilde{S}') \leq \tilde{m}(\tilde{S}) \leq \tilde{y} \implies \sum(\tilde{m}(\tilde{S}) - \tilde{y})^2 \leq (\tilde{m}(\tilde{S}') - \tilde{y})^2, \quad (40)$$

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 $\mathrm{Sym}^+(n)$. □

C RUNNING TIME ON 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.

D 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