DiffAug: Enhance Unsupervised Contrastive Learning with Domain-Knowledge-Free Diffusion-based Data Augmentation

Zelin Zang [1 2 3] **Hao Luo** [2 4] **Kai Wang** [3] **Panpan Zhang** [3] **Fan Wang** [2 4] **Stan.Z Li**[† 1] **Yang You** [3]

Abstract

Unsupervised Contrastive learning has gained prominence in fields such as vision, and biology, leveraging predefined positive/negative samples for representation learning. Data augmentation, categorized into hand-designed and model-based methods, has been identified as a crucial component for enhancing contrastive learning. However, hand-designed methods require human expertise in domain-specific data while sometimes distorting the meaning of the data. In contrast, generative model-based approaches usually require supervised or large-scale external data, which has become a bottleneck constraining model training in many domains. To address the problems presented above, this paper proposes DiffAug, a novel unsupervised contrastive learning technique with diffusion mode-based positive data generation. DiffAug consists of a semantic encoder and a conditional diffusion model; the conditional diffusion model generates new positive samples conditioned on the semantic encoding to serve the training of unsupervised contrast learning. With the help of iterative training of the semantic encoder and diffusion model, DiffAug improves the representation ability in an uninterrupted and unsupervised manner. Experimental evaluations show that DiffAug outperforms hand-designed and SOTA model-based augmentation methods on DNA sequence, visual, and bio-feature datasets. The code for review is released at `https://github.com/zangzelin/code_diffaug`.

[1]AI Lab, Research Center for Industries of the Future, Westlake University, China [2]DAMO Academy, Alibaba Group [3]National University of Singapore [4]Hupan Lab, Zhejiang Province. Correspondence to: Stan.Z Li <Stan.Z.Li@westlake.edu.cn>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

1. Introduction

Contrastive learning, as shown by many studies (He et al., 2020b; Chen et al., 2020; Cui et al., 2021; Wang & Qi, 2022; Assran et al., 2022; Zang et al., 2023), has become important in areas like vision (He et al., 2021; Zang et al., 2022b), natural language processing (Rethmeier & Augenstein, 2023), and biology (Yu et al., 2023; Krishnan et al., 2022). The key to contrastive learning (CL) lies in designing appropriate data augmentation methods. Many studies (Tian et al., 2020; Zhang & Ma, 2022; Peng et al., 2022; Zhang et al., 2023b) have found that data augmentation helps CL by making it more robust and preventing model collapse problems.

Data augmentation falls into two main types: *hand-designed methods* and *model-based methods* (Xu et al., 2023). Hand-designed methods require humans to understand the meaning of the data and then change the input features while maintaining or extending that meaning. For example, several methods in visual tasks (such as color change (Yan et al., 2022), random cropping (Cubuk et al., 2020), and rotation (Maharana et al., 2022)) and DNA sequence representation tasks (such as mutations, insertion, and noise (Lee et al., 2023)) are used to aid in model training. However, the problem is that the above techniques must be more data-specific. For some data (genes or proteins or others), it isn't easy to understand due to the complexity of its meaning. Consequently, it isn't easy to design a good augmentation strategy. Semantics-independent augmentation methods such as adding noise (Huang et al., 2022) and random hiding (Theodoris et al., 2023) are used, but only sometimes with significant results. Another issue with hand-designed methods is their inability to subtly alter the semantics of the data. For instance, a minor mutation in a DNA sequence can lead to significant semantic changes, akin to a gene mutation (as illustrated in Figure. 1). As a result, more positive/negative samples are needed to distribute these risks to obtain a stable representation. It is also challenging to train CL models with fewer samples for certain domains where data acquisition is costly, such as biology.

Given the challenges mentioned earlier, model-based methods (generative models) based on deep learning are used to create better data. In the vision domain, techniques using VAE (Kingma & Welling, 2014), GAN (Goodfellow

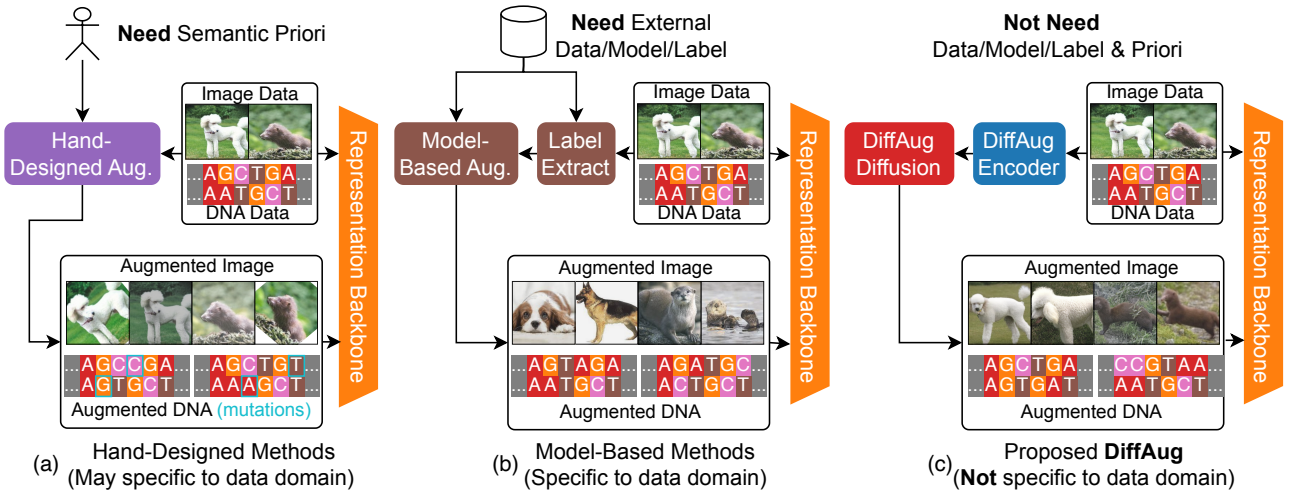


Figure 1. **Comparison of DiffAug with existing augmentation strategy.** (a) Hand-designed augmentation is based on human priori that generate new data with different feature but semantically similar semantic. (b) Model-based augmentation methods generate new data with the same labels by training generative models with large amount of data, labels. These methods often require large amounts of data and target specific data domains. (c) DiffAug attempts to reduce the dependence on external data and prior knowledge through iterative training with encoders and diffusion. Expanding the application areas of unsupervised CL.

et al., 2014), and diffusion models (Ho et al., 2020; Nichol & Dhariwal, 2021; Saharia et al., 2022; Nichol et al., 2022; Ramesh et al., 2022) have been developed to improve model training. For supervised learning, several studies have received attention. Du et al. (2023) proposed the DREAM-OOD framework, which uses diffusion models to generate photo-realistic outliers from in-distribution data for improved OOD detection. Zhang et al. (2023a) developed the Guided Imagination Framework (GIF) using generative models like DALL-E2 and Stable Diffusion for dataset expansion, enhancing accuracy in both natural and medical image datasets. The detailed related works are in Appendix.A.

However, there are concerns about these methods, especially about their diversity and how well they generalize. Moreover, most of these generative models are trained with supervision or need much external data. This makes them less suitable for areas like DNA sequence and bio-feature data (in Figure. 1). This leads to an important question: *Is it possible to design a data augmentation framework to enhance unsupervised CL in different domains without requiring expert knowledge or additional data?*

We introduce DiffAug, a novel diffusion mode-based positive data generation technique for unsupervised CL to address the posed problem. DiffAug eliminates the need for training labels. Instead, we employ a semantic estimator to gauge the semantics of the input data, subsequently guiding the augmentation process. At its core, DiffAug operates through two synergistic modules: a semantic encoder and a diffusion generator. Utilizing a soft contrastive loss, the semantic encoder crafts latent representations that act as guiding vectors for the diffusion generator. This generator then methodically produces augmented data in the input space, ensuring varying levels of semantic consistency based on the guiding vectors and specific adjustable hyperparameters.

We demonstrate DiffAug pioneering on DNA sequence and biometric datasets, and pioneering on highly competitive visual datasets. Our findings indicate that the proposed method can produce sensible data augmentations, subsequently enhancing the performance of unsupervised CL that utilizes these augmentations. Notably, DiffAug performs superior classification and clustering tasks compared to all benchmark methods. The primary contributions of this paper are: (a) We introduce DiffAug, a novel unsupervised CL technique with diffusion mode-based positive data generation. DiffAug's data augmentation improves traditional domain-specific hand-designed data augmentation strategy. (b) DiffAug operates independently of external data or manually designed rules. Its versatility allows for integration with various models, encompassing domains like vision or biology studies. (b) The experimental results show the efficacy of DiffAug in enhancing the performance of CL in different tasks. This suggests that DiffAug can generate positive sample data unsupervised, which in turn promotes the development of unsupervised learning techniques.

2. Methods

In the context of unsupervised data augmentation, the training dataset providing potential semantic categories is denoted as $\mathcal{D}_t = \{\mathbf{x}_i\}_{i=1}^{N}$, where N is the size of the training

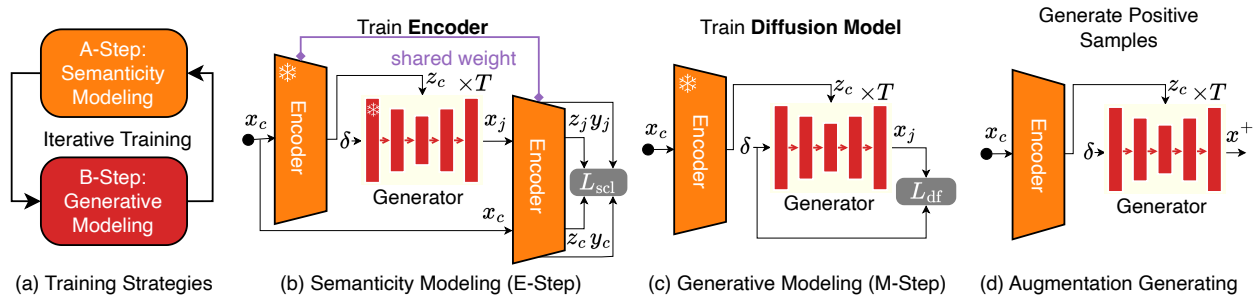


Figure 2. **The DiffAug framework and training strategy.** DiffAug includes a semantic encoder Enc$(\cdot|\theta)$ and a diffusion generator Gen$(\cdot|\phi)$. (a) shows how Enc$(\cdot|\theta)$ and Gen$(\cdot|\phi)$ are interative trained. (b) and (c) show how to calculate the loss functions. (d) shows how to generate new augmentation data with the trained model.

set. To boost the training efficiency of unsupervised contrastive learning with positive samples generated by the diffusion model, a novel framework called DiffAug is proposed.

2.1. Preliminaries of Contrastive Learning

Contrastive Learning. Contrastive learning learns visual representation via enforcing the similarity of the positive pairs and enlarging distance of negative pairs. Formally, loss is defined as, $\mathcal{L}_{\text{cl}} =$

$$-\log\mathcal{Q}\left(\mathbf{z}_i,\mathbf{z}_i^+\right)+\log[\mathcal{Q}\left(\mathbf{z}_i,\mathbf{z}_i^+\right)+\sum_{\mathbf{z}_i^-\in V^-}\mathcal{Q}\left(\mathbf{z}_i,\mathbf{z}_i^-\right)] \quad (1)$$

where $\mathbf{z}_i$ is the low dimensional embedding $\mathbf{z}_i = \text{Enc}^{\text{cl}}(\mathbf{x}_i)$, $Q\left(\mathbf{z}_i,\mathbf{z}_i^+\right)$ indicates the similarity between positive pairs while $Q\left(\mathbf{z}_i,\mathbf{z}_i^-\right)$ is the similarity between negative pairs. For the traditional scheme, in the computer vision domain, data augmentation methods such as random cropping (Cubuk et al., 2020) or data Mixup (Zhang et al., 2017) are used to generate new positive data. The negative samples v_i^- are sampled from negative distribution V^-.

Soft Contrastive Learning. To address the performance degradation due to view noise in contrastive learning and to accomplish unsupervised learning on smaller scale datasets, Zang et al. (2023) designed soft contrastive learning, which smoothes sharp positive and negative sample pair labels by evaluating the credibility of the sample pairs. Consider the loss form for multiple positive samples and multiple negative samples as,

$$\begin{aligned}\mathcal{L}_{\text{scl}}(\mathbf{y}_c,\mathbf{y}_j,\mathbf{z}_c,\mathbf{z}_j) = -\sum_{j=1}^{\mathcal{B}}\{\mathcal{P}(\mathbf{y}_c,\mathbf{y}_j)\log(\mathcal{Q}(\mathbf{z}_c,\mathbf{z}_j))+\\ (1-\mathcal{P}(\mathbf{y}_c,\mathbf{y}_j))\log\left(1-\mathcal{Q}(\mathbf{z}_c,\mathbf{z}_j)\right)\},\\ \mathcal{P}(\mathbf{a},\mathbf{b}) = \left(1+\mathcal{H}_{ij}\left(e^{\beta}-1\right)\right)\mathcal{Q}(\mathbf{a},\mathbf{b}),\end{aligned} \quad (2)$$

where the $\mathbf{y}_i, \mathbf{z}_i$ are the high dimensional embedding and low dimensional embedding $\mathbf{y_i}, \mathbf{z_i} = \text{Enc}(\mathbf{x_i})$. The $\mathcal{P}(\mathbf{a},\mathbf{b})$ is soft learning weight and calculated by the positive/negative pair indicator $\mathcal{H}_{cj}$. The hyper-parameter $\beta \in [0,1]$ introduces prior knowledge of data augmentation relationship $\mathcal{H}_{cj}$ into the model training. Details of contrastive and soft contrastive learning are in Appendix. B.

2.2. DiffAug Design Details and Training Strategies

DiffAug Framework. DiffAug accomplishes the tasks of *positive sample generation* and *data representation* by iterating the two modules over each other (in Figure. 2). DiffAug consists of two main modules, a semantic encoder Enc$(\cdot|\theta)$ and a diffusion generator Gen$(\cdot|\phi)$, where θ and ϕ are model parameters. The Enc$(\cdot|\theta)$ maps the input data $\mathbf{x}_i$ to the discriminative latent space $\mathbf{v}_i$, and the generator Gen$(\cdot|\phi)$ generates new data with a semantic vector $\mathbf{v}_i$. Similar to the Expectation maximization algorithm (Gupta et al., 2011), the semantic encoder Enc$(\cdot|\theta)$ and the diffusion generator Gen$(\cdot|\phi)$ are trained in turn by two different loss functions (see Figure. 2(a) and Figure. 2(b)).

Semanticity Modeling (A-Step). In the semanticity modeling step, given a central data $\mathbf{x}_c$, we generate a background set $\mathcal{B}_c = \{\mathbf{x}_1,\cdots,\mathbf{x}_j,\cdots,\mathbf{x}_{\mathcal{N}_b}\}$,

$$\begin{cases}\mathbf{x}_j \sim \mathcal{D}_t & \text{if} \quad \mathcal{H}_{cj}=0\\ \mathbf{x}_j \sim A_{\text{ug}}(\mathbf{x}_c) & \text{if} \quad \mathcal{H}_{cj}=1\end{cases} \quad (3)$$

where N_b is the number of background data points. The $\mathcal{H}_{cj}=0$ indicates $\mathbf{x}_j$ is sampled from the dataset $\mathcal{D}_t$, and $\mathbf{x}_c$ and $\mathbf{x}_j$ are negative pair. Meanwhile, $\mathcal{H}_{cj}=1$ indicates $\mathbf{x}_c$ and $\mathbf{x}_j$ are positive pair and $\mathbf{x}_j$ is sampled from data augmentation. For details, new positive data are generated by the diffusion model according to DDPM (Ho et al., 2020),

$$\mathbf{x}_j = \text{Gen}(\delta,\mathbf{z}_c|\phi^*),\mathbf{y}_c,\mathbf{z}_c = \text{Enc}(\mathbf{x_c}|\theta^*), \quad (4)$$

where $\text{Gen}(\delta,\mathbf{z}_c|\phi^*)$ is the generation process of the diffusion model, and the generating details are in Eq. (7). The

Table 1. **Comparison of Linear probing results on DNA sequence datasets.** The compared methods including SOTA DNA sequence methods (DNA-BERT (Ji et al., 2021), NT (Dalla-Torre et al., 2023), Hyena (Nguyen et al., 2023)) and contrastive methods with human-designed DNA-augmentation. The 'AVE' represents the average performance. The best results are marked in **bold**.

Datasets	Genomic Benchmarks (Grešová et al., 2023)								
	MoEnEn	CoIn	HuWo	HuEnCo	HuEnEn	HuEnRe	HuNoPr	HuOcEn	AVE
CNN	69.0	87.6	93.0	58.6	69.5	93.3	84.6	68.0	76.7
DNA-BERT	69.4	92.3	96.3	**74.3**	81.1	87.7	85.8	73.1	82.5
NT	70.2	90.0	92.3	71.5	80.8	87.9	84.0	77.2	81.7
Hyena	80.9	89.0	96.4	73.2	88.1	88.1	**94.6**	79.2	86.2
SSL+Translocation	83.8	88.2	95.5	73.8	77.4	88.2	84.7	52.5	80.5
SSL+RC	84.5	88.3	95.8	71.9	82.3	86.8	91.0	74.4	84.3
SSL+Insertion	80.9	89.8	96.6	73.7	83.4	87.1	91.8	77.3	85.0
SSL+Mixup	80.9	89.4	96.2	73.2	85.2	88.6	91.6	77.9	85.4
DiffAug	**86.0**(+1.5)	**94.9**(+2.6)	**96.8**(+0.2)	74.0(-0.3)	**94.9**(+6.8)	**91.8**(+3.2)	94.5(-0.1)	**79.9**(+0.7)	**89.1**(+2.9)

Algorithm 1 The DiffAug Training Algorithm:

Input: Data: $\mathcal{D}_t = \{\mathbf{x}_c\}_{i=1}^N$, Learning rate: η, E or M State: S, Batch size: B, Network parameters: θ, ϕ,
Output: Updateed Parameters: θ, ϕ.

while $b = 0; b < [|\mathcal{X}|/B]; b$++ **do**
  $\mathbf{x}_c \sim \mathcal{D}_t$; # Sample the centering data
  $\mathbf{y}_c, \mathbf{z}_c \leftarrow \text{Enc}(\mathbf{x}_c|\theta)$; # Generate frozen condition vector
  if S==B-Step **then**
    $L_1 \leftarrow L_{\text{df}}(\mathbf{x}_c, \text{SG}(\mathbf{z}_c)|\phi)$ by Eq. (6);
    $\phi \leftarrow \phi - \eta \frac{\partial \mathcal{L}_1}{\partial \phi}$, # Calculate diffusion loss
  else
    $\mathcal{B}_c = \{\mathbf{x}_1, \cdots, \mathbf{x}_\mathcal{B} | \mathbf{x}_j \sim \mathcal{D}_t$ **if** $\mathcal{H}_{ij} = 0; \mathbf{x}_j \sim \text{Aug}(\mathbf{x}_c)$ **else** $\}$; # Generate/sample data
    $\mathcal{Y} = \{\mathbf{y}_1, \cdots, \mathbf{z}_j, \cdots, \mathbf{y}_\mathcal{B}\}$,
    $\mathcal{Z} = \{\mathbf{z}_1, \cdots, \mathbf{z}_j, \cdots, \mathbf{z}_\mathcal{B}\}, \mathbf{y}_j, \mathbf{z}_j = \text{Enc}(\mathbf{x}_j|\theta)$
    $L_2 \leftarrow L_{\text{scl}}(\mathcal{Y}, \mathcal{Z})$ by Eq. (2);
    $\theta \leftarrow \theta - \eta \frac{\partial \mathcal{L}_2}{\partial \theta}$ # Calculate scl loss
  end if
end while

$\delta \sim \mathcal{N}(0, \mathbf{1})$ is the random initialized data, and $\mathbf{z}_c$ is a conditional vector. The $*$ in ϕ^* and θ^* means the parameter is frozen. To avoid unstable positive samples from untrained generative models, training starts exclusively with traditional data augmentation tools, and then, the data generated by DiffAug is replaced with data generated by DiffAug, with a replacement probability of the hyperparameter λ, an oversized λ introduce toxicity, which we will discuss in Sec. 3.6. We update the parameter of the semantic encoder with the soft contrastive learning loss,

$$\theta = \theta - \eta \sum_{\mathbf{x}_j \in \mathcal{B}_c} \{\mathcal{L}_{\text{scl}}(\mathbf{y}_c, \mathbf{z}_c, \mathbf{y}_j, \mathbf{z}_j)\}, \quad \text{where } \mathbf{y}_j, \mathbf{z}_j = \text{Enc}(\mathbf{x}_j|\theta), \tag{5}$$

where the η is the learning rate, and the $\mathcal{L}_{\text{scl}}(\mathbf{y}_c, \mathbf{z}_c, \mathbf{y}_j, \mathbf{z}_j)$ is in Eq. (2).

Generative Modeling (B-Step). In the generative modeling step, the conditional diffusion generator $\text{Gen}(\cdot|\phi)$ is trained by the vanilla diffusion loss $\mathcal{L}_{\text{df}}(\mathbf{x}_c, \mathbf{z}_c|\phi)$ (Ho et al., 2020),

$$\phi = \phi - \eta \sum_{t=1}^{T} \left\{ \left\| \delta - g_\phi \left(\sqrt{\bar{\alpha}_t} \widetilde{\mathbf{x}}_c^t + \sqrt{1 - \bar{\alpha}_t}, t, \mathbf{z}_c \right) \right\|_2^2 \right\}, \tag{6}$$

where the conditional vector $\mathbf{z}_c$ is generated from the semantic encoder in Eq.(4). The $g_\phi(\cdot)$ is the conditional diffusion neural network. The α_t is the noise parameter in the diffusion process, and $\bar{\alpha}_t = 1 - \alpha_t$. The $\widetilde{\mathbf{x}}_c^t$ is the intermediate data in the diffusion process, and the $\widetilde{\mathbf{x}}_c^0 = \mathbf{x}_c$. T is the time step of the generation process. When $g_\phi(\cdot)$ is trained, the detailed generating process is,

$$\text{Gen}(\delta, \mathbf{z}_c|\phi^*) = \left\{ \widetilde{\mathbf{x}}^0 \mid \widetilde{\mathbf{x}}^{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(\widetilde{\mathbf{x}}^t - \hat{\alpha} \right) + \sigma_t \mathcal{N}(0, 1), \right\}, \quad \hat{\alpha} = \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} g_\phi(\widetilde{\mathbf{x}}^t, t, \mathbf{z}_c^*), \tag{7}$$

where $t \in \{T, \cdots, 1\}$, the $g_\phi(\cdot)$ is a neural network approximator intended to predict δ with $\widetilde{\mathbf{x}}$ and the condition vector $\mathbf{z}_c^*$.

Augmentation Generation. Given the trained semantic encoder $\text{Enc}(\cdot|\theta)$ and diffusion generator $D(\cdot)$, and DiffAug generate new augmented data $\mathbf{x}_i^+$ from any input data $\mathbf{x}_i$.

$$\mathbf{x}_i^+ = \text{Gen}(\delta|\mathbf{z}_i), \quad \mathbf{y}_i, \mathbf{z}_i = \text{Enc}(\mathbf{x}_i). \tag{8}$$

Meanwhile, DiffAug's semantic encoder can be seen as a feature extractor. It is considered to have good discriminative performance because it is trained simultaneously as the diffusion generator.

3. Results

We conduct experiments on various datasets, including DNA sequences, vision, and bio-feature datasets. We aim to demonstrate that Diffaug can operate effectively and facilitate improvements across diverse domains.

Table 2. **Comparison of the Linear probing results on Bio-feature dataset.** dimensional reduction (DR) methods and contrastive learning methods are list in the table. The DR methods are widely used on Bio-feature analysis including TopoAE (Moor et al., 2019), PaCMAP (Wang et al., 2022), and hNNE (Sarfraz et al., 2022). CL methods wtih mixup augmentation including Simclr (Chen et al., 2020), BYOL (Grill et al., 2020), MoCo (He et al., 2020b), and DLME (Zang et al., 2022b).

	Method Type	GA1457	SAM561	MC1374	HCL500	WARPARIOP	PROT579	AVE
TopoAE	DR	74.6	72.4	61.3	56.0	73.0	88.3	70.9
PaCMAP	DR	85.3	83.7	61.3	36.2	76.9	87.2	71.7
hNNE	DR	77.4	83.8	62.3	62.2	72.5	82.4	73.4
Simclr+Mixup	CL	84.8	82.4	62.3	61.7	74.3	74.7	73.3
BYOL+Mixup	CL	82.8	73.3	60.9	61.3	72.6	70.5	70.2
MoCo+Mixup	CL	84.2	83.1	70.2	61.7	82.8	84.2	77.7
DLME+Mixup	CL	85.7	83.6	71.4	62.3	83.5	84.5	78.5
DiffAug	CL	**92.7**(+7.0)	**89.3**(+5.5)	**71.8**(+0.4)	**64.7**(+2.4)	**84.8**(+1.3)	**91.2**(+2.9)	**82.4**(+3.9)

Table 3. **Comparison of Linear probing results on vision dataset.** The SimC.+Mix. and Mo.V2+Mix. are SimCLR and MoCoV2 with Mixup data augmentation which processed by Zhang et al. (2022). The SimC./Mo.V2+VAE/GAN means SimCLR/MoCoV2 with VAE/GAN generative model as data augmentation.

Datasets	CF10	CF100	STL10	TINet
SimCLR	89.6	60.3	89.0	45.2
Mo.V2	86.7	56.1	89.1	47.1
BYOL	92.0	62.7	91.8	46.1
SimSiam	91.6	64.7	89.4	43.0
DINO	91.8	67.4	91.7	44.2
SimC.+Mixup	90.9	62.9	89.6	—
Mo.V2+Mixup	91.5	62.7	90.1	—
SimC.+VAE	89.6	64.2	91.7	46.0
Mo.V2+VAE	89.3	65.9	91.2	43.3
SimC.+GAN	90.0	64.3	89.9	44.6
Mo.V2+GAN	91.1	62.9	91.2	43.6
DiffAug	**93.4**(+1.6)	**69.9**(+2.5)	**92.5**(+0.8)	**49.7**(+2.1)

3.1. Comparations on DNA Sequence Datasets

First, we demonstrate the efficacy of DiffAug in improving DNA sequence representation and classification. DNA sequence representation is challenging for contrastive learning because one cannot easily design data augmentation manually by visualizing and understanding the data. Lee et al. (2023) explores how various natural genetic alterations can enhance model training performance.

Test Protocols. Our experiments utilize the Genomic Benchmarks (Grešová et al., 2023), encompassing datasets that target regulatory elements (such as promoters, enhancers, and open chromatin regions) from three model organisms: humans, mice, and roundworms[1]. We adopted a methodology akin to that of Hyena-DNA (Nguyen et al., 2023) for evaluating linear-test performance. To mitigate the influence of the pre-training dataset, we exclusively employed the training data from Genomic Benchmarks for self-supervised pre-training and fine-tuning, followed by an evaluation of the test dataset. The comparison includes various methods such as CNN (as per Genomic Benchmarks), DNA-BERT (Ji et al., 2021), NT (Dalla-Torre et al., 2023), and Hyena (Nguyen et al., 2023). Both DiffAug and 'SSL+' leverage Hyena-tiny[2] backbone, replacing Hyena's pre-training approach with a unique pre-training methodology. 'SSL+' means the model is pre-trained with SimCLR (Chen et al., 2020) with the augmentation of natural DNA augmentation strategies in (Lee et al., 2023). Further details on the dataset are provided in Appendix C.

Analysis. DiffAug outperforms competing methods in eight evaluations across four datasets, achieving performance improvements ranging up to 6.8%. Notably, DiffAug demonstrates several significant benefits, particularly in classification metrics: (a) DiffAug enhances the sequence model's performance on classification accuracy, surpassing traditional DNA augmentation approaches. (b) By learning distributional knowledge from the training data, DiffAug facilitates data augmentation with minimal human intervention, potentially enabling the generation of more stable enhanced samples.

3.2. Comparations on Bio-feature Datasets

Next, we benchmark DiffAug against SOTA unsupervised contrastive learning (CL) methods and traditional dimensional reduction (DR) methods in bio-feature datasets. Unlike DNA sequence data, bio-feature datasets are the format for most proteomics(Suhre et al., 2021), genomics (Bustamante et al., 2011), and transcriptomics(Aldridge & Teichmann, 2020) data. The dataset contains an equal-length vector per sample, and each element in the vector represents a gene, protein abundance, or bio-indicator. The data of training time consumption is in the Table 11.

[1]https://github.com/ML-Bioinfo-CEITEC/genomic_benchmarks.

[2]https://huggingface.co/LongSafari/hyenadna-tiny-1k-seqlen

Table 4. **Ablation study of the semantic encoder includes DiffAug's encoder is necessary and can efficiently generate conditional vectors.** Linear-tests performance of different ablation setups on on vision dataset and biological dataset.

Datasets	Vision Datasets				Bio-feature Datasets			
	CF10	CF100	STL10	TINet	GA1457	SAM561	MC1374	HCL500
A1. Gen$(\cdot)$ + Sup. Condition	**93.4**	**70.9**	**92.9**	45.9	92.5	**89.6**	71.1	63.9
A2. Gen$(\cdot)$ + Rand. Condition	34.2	10.4	30.1	7.3	10.5	16.9	13.9	10.0
A3. Gen$(\cdot)$ + Enc$(\cdot\|\theta)$ (DiffAug)	**93.4**	69.9	92.5	**49.7**	**92.7**	88.3	**71.8**	**64.7**

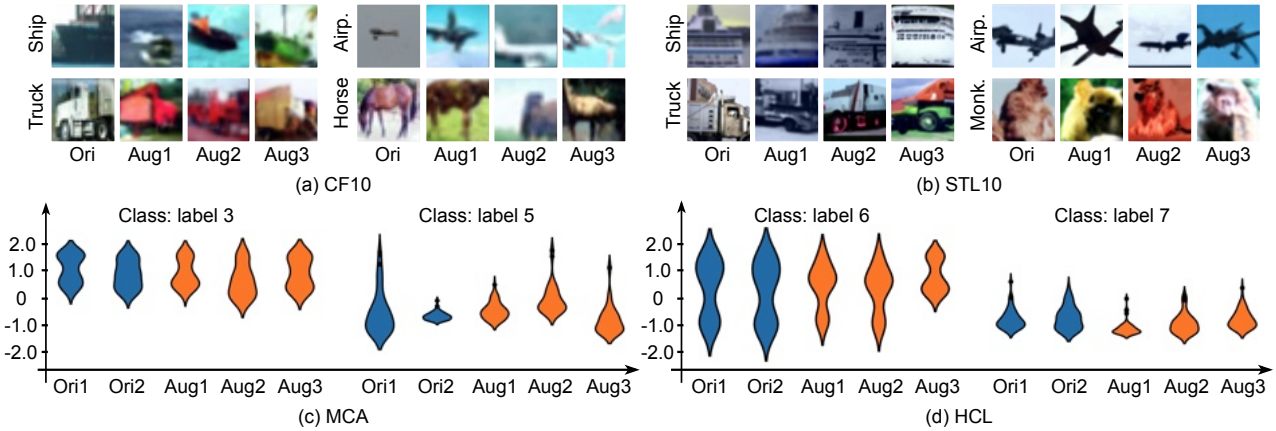


Figure 3. **The display of original and generated images illustrates that DiffAug generates semantically similar augmented data.** The 'Ori' means original data and Aug1, Aug2 and Aug3 are augmentated data. For bio-feature data, we use violin plots to plot the distribution of features.

Test protocols. Our experiments are conducted using a variety of bio-feature datasets, namely GA1457 (Rouillard et al., 2016), SAM (Weber & Robinson, 2016), MC1374 (Han et al., 2018), HCL500 (Han et al., 2020), PROT579 (Sun et al., 2022), and WARPARIOP. To evaluate the effectiveness of our DiffAug, we adopted a linear Support Vector Machine (SVM) performance assessment akin to the methodologies described in Wang et al. (2022) and Sarfraz et al. (2022). In this assessment, dataset embeddings are split, allocating 90% for training purposes and the remaining 10% for testing. The comparative results are shown in Table 2. Further details regarding this experimental setup can be found in the Appendix F. The training regimen for DiffAug is structured as follows: an initial A-Step of 330 epochs, followed by an B-Step of 330 epochs and concluding with a final A-Step of 340 epochs. Comprehensive training specifics, along with the evolution of accuracy throughout training, are depicted in Figure 7.

Analysis. DiffAug consistently surpasses all other methods across eight evaluations spanning four datasets, registering a performance enhancement between 0.4% and 7.0% over its counterparts. (a) It is worth noting that the benefits of DiffAug are not limited to DNA sequence data. It also excels in areas such as bio-feature and has broader applications in traditional bioanalysis. (b) Data processed through DiffAug exhibits reduced overlap among distinct groups, facilitating enhanced classification. This suggests DiffAug delineates more explicit boundaries between data categories, culminating in more precise outcomes. Corresponding evidence is demonstrated in Figure 4. (c) Experiments with DNA sequence and bio-feature data have demonstrated that DiffAug is versatile and an important complement to other unsupervised learning techniques. Searching for effective data augmentation strategies in biology has been difficult.

3.3. Comparations on Vision Datasets

Next, we benchmark DiffAug against the SOTA unsupervised comparative learning (CL) method on a visual dataset. This field is much more active and has seen the emergence of excellent human-designed data augmentation methods. We focus on data augmentation techniques that can be used for unsupervised CL. Therefore, the comparison does not include some labeling-based methods (Nichol et al., 2021; He et al., 2023; Trabucco et al., 2023; Zhang et al., 2023a).

Test Protocols. Experiments are performed on CIFAR-10 [CF10] and CIFAR-100 [CF100] (Krizhevsky et al., 2009), STL10 (Coates et al., 2011), TinyImageNet [TINet] (Le & Yang, 2015) dataset. Notably, we did not use a larger dataset in our experiments. It is because the proposed method aims to efficiently utilize data to train models when data is limited (or expensive). Simple augmen-

Table 5. **Ablation study of scl loss function and training strategy.** The classifier accuracy of each setting is displayed in this table. Soft contrastive learning is improved with typical contrast learning, and AB-Step training is more stable.

Datasets	Vision Datasets				Bio-feature Datasets			
	CF10	CF100	STL10	TINet	GA1457	SAM561	MC1374	HCL500
B1. SimCLR	89.6	60.3	89.0	45.2	84.8	82.4	62.3	61.7
B2. DiffAug w/o $\mathcal{L}_{\text{df}}$	91.3	66.1	90.1	44.9	89.1	82.1	59.3	62.3
B3. DiffAug w/o $\mathcal{L}_{\text{scl}}$	92.7	68.4	90.9	45.1	89.2	82.4	69.2	61.3
B4. DiffAug Syn. Training	92.9	69.7	**92.7**	45.3	90.1	**89.6**	68.1	62.3
B5. DiffAug AB Training	**93.4**	**69.9**	92.5	**49.7**	**92.7**	88.3	**71.8**	**64.7**

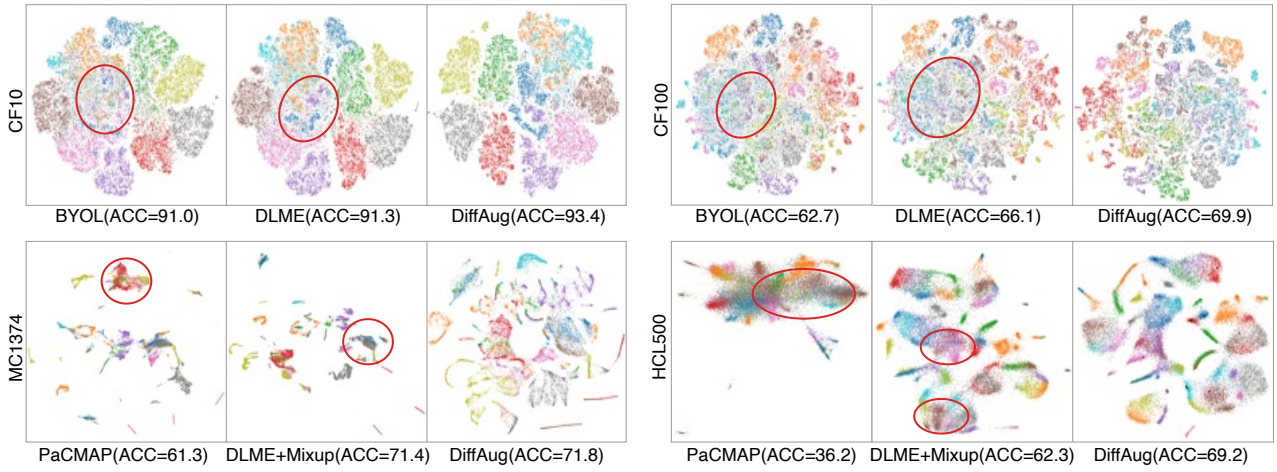


Figure 4. **The scatter visualization of representation indicates DiffAug's encoder learns cleaner embedding.** The colors represent different categories; there are 100 categories in CF100; we used the superclasses label provided by (Deng et al., 2021).

tation techniques can train models without fully adequate and well-sampled data. We followed a procedure similar to SimCLR (Chen et al., 2020) for the Linear-test performance assessment. The baseline of SimCLR (Chen et al., 2020), BYOL (Grill et al., 2020), MoCo v2 (He et al., 2020a), and SimSiam (Chen & He, 2021) is from Peng et al. (2022). The results of SimCLR and MoCoV2 with Mixup data augmentation (SimC.+Mix. and Mo.V2+Mix.) are from Zhang et al. (2022). Since we did not find the corresponding GAN and VAE as a baseline for data augmentation, we tested the corresponding results ourselves. For DiffAug, its semantic encoder served as the CL backbone, trained using DiffAug-augmented images. Comparative results are shown in Table 3. Details of the experimental setup are in Appendix D. The training strategy of DiffAug is A-Step: 200 epochs → B-Step: 400 epoch → A-Step: 800 epoch. The data of training time consumption is in the Table 9.

Analysis. From Table 3, it is evident that DiffAug consistently outperforms SOTA methods across all datasets. It surpasses other techniques by at least 0.8% in five out of the four projects. This showcases the effectiveness of DiffAug's data augmentation. (a) *Beyond hand-designed augmentation methods.* DiffAug's versatility indicates that its approach is on par with, or even better than, traditional hand-crafted methods. The encoder in DiffAug produces robust features. (b) *Beyond Mixup improved CL methods.* DiffAug outperforms the Mixup improved CL method of typical contrast learning methods, and additionally, models trained using DiffAug-generated data and contrast learning methods bring some improvement. (c) For datasets with many classes, like CF100 and TINet, DiffAug's encoder might only sometimes capture some local detail. Still, augmented data is crucial in guiding CL to produce better results.

3.4. DiffAug Effectiveness Analysis

Effectiveness analysis of diffusion generator. The diffusion module generates new positive data by inputting the provided condition vector. To demonstrate that our DiffAug works appropriately, we show the generation results for both the image and bio-feature datasets (in Figure. 3). A more detailed implementation and more results are in the Appendix D and Appendix F. We can observe that the generated data retains semantic similarity to the original data. For example, the objects described in the image data are consistent, while the gene distribution is also consistent. At the same time, the generated data is not simply copied

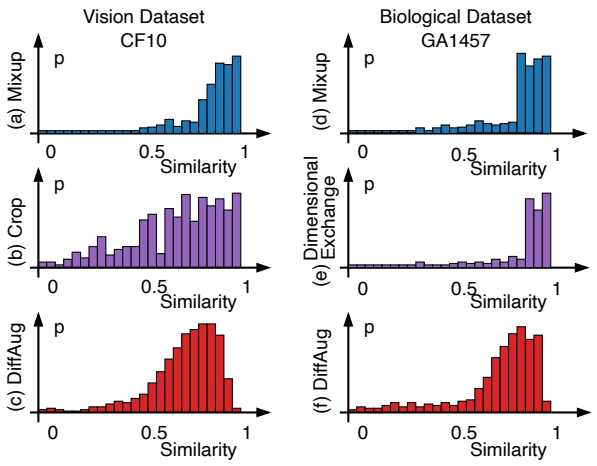


Figure 5. **Hist plot of the cosine similarity between original data and the augmentation data in latent space indicate that DiffAug generates semantically smooth augmentations.** For the image data, we compared similar mixups with random cropping. For bio-feature datasets, we compared same-label Mixup and random dimension swapping.

but varied without changing the semantic information.

In addition, we computed the cos-similarity of the original augmented sample in latent space to explore further the semantic differences between the newly generated and original data. As depicted in Figure. 5, DiffAug's similarity distribution is smoother and broader. In comparison, Mixup tends to produce augmentations that are very similar semantically, while methods like cropping might introduce data with semantically distinct noise samples. In addition, we computed the cos-similarity of the original augmented sample in latent space to explore further the semantic differences between the newly generated and original data. As depicted in Figure. 5, DiffAug's similarity distribution is smoother and broader. In comparison, Mixup tends to produce augmentations that are very similar semantically, while methods like cropping might introduce data with semantically distinct noise samples.

Effectiveness analysis of semantic encoder. Next, we confirm that the semantic encoder of DiffAug works well by visualizing the representation of DiffAug and baseline methods (in Figure. 4). The t-SNE (Van der Maaten & Hinton, 2008) is used to analyze the BYOL, DLME, and DiffAug embedding on CF10, CF100, MC1374, and HCL500 datasets. The results show that DiffAug's encoder learns cleaner embedding than the baseline methods in Figure. 4, DiffAug E means the first A-Step's results of the DiffAug. By comparing DiffAug E and DiffAug, we observe that the augmented data further improves the embedding quality, significantly enhancing the depiction of local structures. The same conclusion is shown in Figure. 7.

3.5. Ablation Study and Effectiveness of Component

Ablation study of the semantic encoder. In the ablation study presented in Table. 4, we consider three configurations: A1 and A2 confirm the significance of DiffAug's semantic encoder by ablating it in two ways. A1 directly uses supervised one hot label as the conditional, bypassing the condition vectors generated by the unsupervised neural network. A2 employs random conditional vectors instead of those the encoder produces. A3 means the proposed DiffAug method. The results from these experiments can be found in Table 4. We observe that the average performance of A1 is highest due to the access to the label. And not accessing the label at all brings a huge performance drop. The results in A3 illustrate that DiffAug's performance is comparable to the fully supervised condition, demonstrating its ability to model supervised annotation within an unsupervised framework.

Ablation study of training strategy and scl loss function. For Ablation in Table. 5, B1 means that the model is trained by SimCLR (Chen et al., 2020). B2 omits the diffusion loss and trains the encoder with only the soft CL loss. B3 omits the soft CL loss and trains the encoder with InfoNCE loss. B4 and B5 talk about the training strategy of DiffAug. B4 denotes training the model by integrating two loss functions, i.e., mixing A-Step and B-Step to update the parameters of both networks simultaneously through a single forward propagation.B5 denotes the default training strategy, which trains the model by alternating the two loss functions. The results from these experiments can be found in Table 5. First, we observe that either replacing the scl loss or replacing the diff model (B2 or B3) brings about performance degradation, which implies that the two modules of DiffAug work in conjunction with each other. Second, we observe that on some datasets, the performance of the two training strategies (B4 and B5) is comparable, but on others, the EM method demonstrates higher stability. We attribute this to the fact that the difficulty of diffusion model training varies from data to data, and simultaneous training may result in the two modules being unable to match at all times, bringing about instability in training. However, the E-M training approach avoids this problem.

3.6. Hyperparametric Analysis and Toxicity Analysis

Finally, we investigate the performance improvement and potential toxicity of the DiffAug method through hyperparametric analysis. The hyperparameter λ determines how often the model generated by DiffAug affects the training of the semantic encoder. Introducing the augmentation data ($\lambda = 0$) brings the method back to traditional CL methods, while too much ($\lambda = 1$) will lead to the encoder

crashing. To demonstrate this, we tested the model performance of different λ counterparts on two visual datasets (CF10, CF100) and two bio-feature datasets (SAM561 and MC1374). As shown in Figure. 3, the change in performance brought about by λ is consistent across datasets. Specifically, setting $\lambda = 0.1$ or $\lambda = 0.15$ provides the most significant gain. We believe that $\lambda = 0.1$ may be a suitable default setting for most datasets.

4. Conclusion

In summary, we presented DiffAug, an innovative contrastive learning framework that leverages diffusion-based augmentation to enhance the robustness and generalization of unsupervised learning. Unlike many existing methods, DiffAug operates independently of prior knowledge or external labels, positioning it as a versatile augmentation tool with notable performance in vision and life sciences. Our tests reveal that DiffAug consistently boosts classification and clustering accuracy across multiple datasets.

Acknowledgements

This work was supported by the Science & Technology Innovation 2030 Major Program Project No. 2021ZD0150100, National Natural Science Foundation of China Project No. U21A20427, Project No. WU2022A009 from the Center of Synthetic Biology and Integrated Bioengineering of Westlake University, and Project No. WU2023C019 from the Westlake University Industries of the Future Research. Finally, we thank the Westlake University HPC Center for providing part of the computational resources. This work was done when Zelin Zang was intern at DAMO Academy. This work was supported by Alibaba Group through Alibaba Research Intern Program.

This research is supported by the National Research Foundation, Singapore under its AI Singapore Programme (AISG Award No: AISG2-PhD-2021-08-008). Yang You's research group is being sponsored by NUS startup grant (Presidential Young Professorship), Singapore MOE Tier-1 grant, ByteDance grant, ARCTIC grant, SMI grant (WBS number: A-8001104-00-00), Alibaba grant, and Google grant for TPU usage.

Impact Statement

The introduction of DiffAug, our novel approach to unsupervised contrastive learning, presents an opportunity to reflect on the ethical considerations inherent in the advancement of machine learning technologies. A key aspect of DiffAug is its reliance on a conditional diffusion model for the generation of new, positive data samples. This process, rooted in unsupervised learning, underscores our commitment to minimizing human intervention and, by extension, the potential for human bias in the initial stages of data handling. By automating the generation of training data, we aim to reduce the subjective influences that may arise from hand-designed methods, thus promoting a more objective and equitable development of machine learning models. Furthermore, the iterative training process of the semantic encoder and diffusion model in DiffAug is designed to ensure that the generated data remains true to the original dataset's semantic integrity, thereby upholding the principles of fairness and transparency in AI.

The deployment of DiffAug is poised to have a transformative effect on a wide array of sectors, leveraging the untapped potential of unsupervised learning to interpret complex datasets across disciplines. In healthcare, for example, DiffAug's ability to enhance representation learning without extensive labeled datasets could revolutionize the early detection and diagnosis of diseases, making healthcare more accessible and efficient. The fundamental shift towards unsupervised learning, exemplified by DiffAug, also heralds a new era of innovation in which the reliance on large, labeled datasets is significantly reduced, thereby democratizing access to advanced machine learning technologies.

References

Aldridge, S. and Teichmann, S. A. Single cell transcriptomics comes of age. *Nature Communications*, 11(1): 4307, 2020.

Antoniou, A., Storkey, A., and Edwards, H. Data augmentation generative adversarial networks, 2017. URL https://arxiv.org/abs/1711.04340.

Assran, M., Caron, M., Misra, I., Bojanowski, P., Bordes, F., Vincent, P., Joulin, A., Rabbat, M., and Ballas, N. Masked siamese networks for label-efficient learning. In *ECCV*, pp. 456–473. Springer, 2022.

Besnier, V., Jain, H., Bursuc, A., Cord, M., and Pérez, P. This dataset does not exist: Training models from generated images. In *ICASSP 2020*, pp. 1–5. IEEE, 2020. doi: 10.1109/ICASSP40776.2020.9053146. URL https://doi.org/10.1109/ICASSP40776.2020.9053146.

Blanco-Gonzalez, A., Cabezon, A., Seco-Gonzalez, A., Conde-Torres, D., Antelo-Riveiro, P., Pineiro, A., and Garcia-Fandino, R. The role of ai in drug discovery: challenges, opportunities, and strategies. *Pharmaceuticals*, 16(6):891, 2023.

Botton, A., Barberi, G., and Facco, P. Data augmentation to support biopharmaceutical process development through

digital models—a proof of concept. *Processes*, 10(9): 1796, 2022.

Brock, A., Donahue, J., and Simonyan, K. Large scale GAN training for high fidelity natural image synthesis. In *ICLR 2019*, 2019. URL https://openreview.net/forum?id=B1xsqj09Fm.

Bustamante, C. D., De La Vega, F. M., and Burchard, E. G. Genomics for the world. *Nature*, 475(7355):163–165, 2011.

Chen, T., Kornblith, S., Norouzi, M., and Hinton, G. A Simple Framework for Contrastive Learning of Visual Representations. *arXiv:2002.05709 [cs, stat]*, June 2020. URL http://arxiv.org/abs/2002.05709. arXiv: 2002.05709.

Chen, X. and He, K. Exploring simple siamese representation learning. In *CVPR*, pp. 15750–15758, 2021.

Coates, A., Ng, A., and Lee, H. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 215–223. JMLR Workshop and Conference Proceedings, 2011.

Cubuk, E. D., Zoph, B., Shlens, J., and Le, Q. V. Randaugment: Practical automated data augmentation with a reduced search space. In *CVPR*, pp. 702–703, 2020.

Cui, J., Zhong, Z., Liu, S., Yu, B., and Jia, J. Parametric contrastive learning. In *ICCV*, pp. 715–724, 2021.

Dalla-Torre, H., Gonzalez, L., Mendoza-Revilla, J., Carranza, N. L., Grzywaczewski, A. H., Oteri, F., Dallago, C., Trop, E., de Almeida, B. P., Sirelkhatim, H., et al. The nucleotide transformer: Building and evaluating robust foundation models for human genomics. *bioRxiv*, pp. 2023–01, 2023.

Dat, P. T., Dutt, A., Pellerin, D., and Quénot, G. Classifier training from a generative model. In *2019 International Conference on Content-Based Multimedia Indexing (CBMI)*, pp. 1–6, 2019. doi: 10.1109/CBMI.2019.8877479.

Deng, D., Chen, G., Hao, J., Wang, Q., and Heng, P.-A. Flattening sharpness for dynamic gradient projection memory benefits continual learning. *Advances in Neural Information Processing Systems*, 34:18710–18721, 2021.

Dhariwal, P. and Nichol, A. Q. Diffusion models beat gans on image synthesis. In Ranzato, M., Beygelzimer, A., Dauphin, Y. N., Liang, P., and Vaughan, J. W. (eds.), *NeurIPS 2021*, pp. 8780–8794, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/49ad23d1ec9fa4bd8d77d02681df5cfa-Abstract.html.

Du, X., Sun, Y., Zhu, X., and Li, Y. Dream the impossible: Outlier imagination with diffusion models, 2023.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., and Weinberger, K. (eds.), *NeurIPS*, volume 27. Curran Associates, Inc., 2014.

Grešová, K., Martinek, V., Čechák, D., Šimeček, P., and Alexiou, P. Genomic benchmarks: a collection of datasets for genomic sequence classification. *BMC Genomic Data*, 24(1):25, 2023.

Grill, J.-B., Strub, F., Altché, F., Tallec, C., Richemond, P. H., Buchatskaya, E., Doersch, C., Pires, B. A., Guo, Z. D., Azar, M. G., Piot, B., Kavukcuoglu, K., Munos, R., and Valko, M. Bootstrap Your Own Latent: A New Approach to Self-Supervised Learning. *arXiv:2006.07733 [cs, stat]*, June 2020. URL http://arxiv.org/abs/2006.07733. arXiv: 2006.07733.

Gupta, M. R., Chen, Y., et al. Theory and use of the em algorithm. *Foundations and Trends® in Signal Processing*, 4(3):223–296, 2011.

Halko, N., Martinsson, P.-G., and Tropp, J. A. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions, 2010.

Han, X., Wang, R., Zhou, Y., Fei, L., Sun, H., Lai, S., Saadatpour, A., Zhou, Z., Chen, H., Ye, F., et al. Mapping the mouse cell atlas by microwell-seq. *Cell*, 172(5):1091–1107, 2018.

Han, X., Zhang, L., Zhou, K., and Wang, X. Progan: Protein solubility generative adversarial nets for data augmentation in dnn framework. *Computers & Chemical Engineering*, 131:106533, 2019.

Han, X., Zhou, Z., Fei, L., Sun, H., Wang, R., Chen, Y., Chen, H., Wang, J., Tang, H., Ge, W., et al. Construction of a human cell landscape at single-cell level. *Nature*, 581(7808):303–309, 2020.

Haque, A. EC-GAN: low-sample classification using semi-supervised algorithms and gans (student abstract). In *AAAI*, pp. 15797–15798, 2021. URL https://ojs.aaai.org/index.php/AAAI/article/view/17895.

He, K., Zhang, X., Ren, S., and Sun, J. Deep Residual Learning for Image Recognition. *arXiv:1512.03385 [cs]*, December 2015. URL http://arxiv.org/abs/1512.03385. arXiv: 1512.03385.

He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9729–9738, 2020a.

He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. Momentum Contrast for Unsupervised Visual Representation Learning. *arXiv:1911.05722 [cs]*, March 2020b. URL http://arxiv.org/abs/1911.05722. arXiv: 1911.05722.

He, K., Chen, X., Xie, S., Li, Y., Dollár, P., and Girshick, R. Masked autoencoders are scalable vision learners. *CVPR*, 2021.

He, R., Sun, S., Yu, X., Xue, C., Zhang, W., Torr, P., Bai, S., and Qi, X. Is synthetic data from generative models ready for image recognition?, 2022. URL https://arxiv.org/abs/2210.07574.

He, R., Sun, S., Yu, X., Xue, C., Zhang, W., Torr, P., Bai, S., and Qi, X. Is synthetic data from generative models ready for image recognition? *ICLR*, 2023.

Ho, J. and Salimans, T. Classifier-free diffusion guidance, 2022. URL https://arxiv.org/abs/2207.12598.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H. (eds.), *NeurIPS*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/4c5bcfec8584af0d967f1ab10179ca4b-Abstract.html.

Huang, H.-H., Rao, H., Miao, R., and Liang, Y. A novel meta-analysis based on data augmentation and elastic data shared lasso regularization for gene expression. *BMC bioinformatics*, 23(Suppl 10):353, 2022.

Jahanian, A., Puig, X., Tian, Y., and Isola, P. Generative models as a data source for multiview representation learning. In *ICLR*, 2022. URL https://openreview.net/forum?id=qhAeZjs7dCL.

Ji, Y., Zhou, Z., Liu, H., and Davuluri, R. V. Dnabert: pre-trained bidirectional encoder representations from transformers model for dna-language in genome. *Bioinformatics*, 37(15):2112–2120, 2021.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. In Bengio, Y. and LeCun, Y. (eds.), *ICLR,*, 2014. URL http://arxiv.org/abs/1312.6114.

Krishnan, R., Rajpurkar, P., and Topol, E. J. Self-supervised learning in medicine and healthcare. *Nature Biomedical Engineering*, pp. 1–7, 2022.

Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. 2009.

Le, Y. and Yang, X. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7):3, 2015.

Lee, N. K., Tang, Z., Toneyan, S., and Koo, P. K. Evoaug: improving generalization and interpretability of genomic deep neural networks with evolution-inspired data augmentations. *Genome Biology*, 24(1):105, 2023.

Li, B., Han, Z., Li, H., Fu, H., and Zhang, C. Trustworthy long-tailed classification. *arXiv: Learning*, 2021.

Li, M. and Zhang, W. Phiaf: prediction of phage-host interactions with gan-based data augmentation and sequence-based feature fusion. *Briefings in Bioinformatics*, 23(1): bbab348, 2022.

Maharana, K., Mondal, S., and Nemade, B. A review: Data pre-processing and data augmentation techniques. *Global Transitions Proceedings*, 3(1):91–99, 2022.

McGibbon, M., Money-Kyrle, S., Blay, V., and Houston, D. R. Scorch: improving structure-based virtual screening with machine learning classifiers, data augmentation, and uncertainty estimation. *Journal of Advanced Research*, 46:135–147, 2023.

Moon, K. R. and van Dijk. Visualizing structure and transitions in high dimensional biological data. *Nature biotechnology*, 37(12):1482–1492, 2019.

Moor, M., Horn, M., Rieck, B., and Borgwardt, K. Topological Autoencoders. *arXiv preprint arXiv:1906.00722*, 2019.

Nguyen, E., Poli, M., Faizi, M., Thomas, A. W., Wornow, M., Birch-Sykes, C., Massaroli, S., Patel, A., Rabideau, C. M., Bengio, Y., et al. Hyenadna: Long-range genomic sequence modeling at single nucleotide resolution. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

Nichol, A., Dhariwal, P., Ramesh, A., Shyam, P., Mishkin, P., McGrew, B., Sutskever, I., and Chen, M. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021.

Nichol, A. Q. and Dhariwal, P. Improved denoising diffusion probabilistic models. In Meila, M. and Zhang, T. (eds.), *ICML*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8162–8171. PMLR, 2021. URL http://proceedings.mlr.press/v139/nichol21a.html.

Nichol, A. Q., Dhariwal, P., Ramesh, A., Shyam, P., Mishkin, P., McGrew, B., Sutskever, I., and Chen, M. GLIDE: towards photorealistic image generation and editing with text-guided diffusion models. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvári, C., Niu, G., and Sabato, S. (eds.), *ICML*, volume 162 of *Proceedings of Machine Learning Research*, pp. 16784–16804. PMLR, 2022. URL https://proceedings.mlr.press/v162/nichol22a.html.

Peng, X., Wang, K., Zhu, Z., Wang, M., and You, Y. Crafting better contrastive views for siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16031–16040, 2022.

Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., and Chen, M. Hierarchical text-conditional image generation with clip latents, 2022. URL https://arxiv.org/abs/2204.06125.

Razavi, A., van den Oord, A., and Vinyals, O. Generating diverse high-fidelity images with VQ-VAE-2. In Wallach, H. M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E. B., and Garnett, R. (eds.), *NeurIPS 2019*, pp. 14837–14847, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/5f8e2fa1718d1bbcadf1cd9c7a54fb8c-Abstract.html.

Rethmeier, N. and Augenstein, I. A primer on contrastive pretraining in language processing: Methods, lessons learned, and perspectives. *ACM Computing Surveys*, 55 (10):1–17, 2023.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution image synthesis with latent diffusion models. In *CVPR 2022*, pp. 10674–10685. IEEE, 2022. doi: 10.1109/CVPR52688.2022.01042. URL https://doi.org/10.1109/CVPR52688.2022.01042.

Rouillard, A. D., Gundersen, G. W., Fernandez, N. F., Wang, Z., Monteiro, C. D., McDermott, M. G., and Ma'ayan, A. The harmonizome: a collection of processed datasets gathered to serve and mine knowledge about genes and proteins. *Database*, 2016, 2016.

Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E., Ghasemipour, S. K. S., Ayan, B. K., Mahdavi, S. S., Lopes, R. G., Salimans, T., Ho, J., Fleet, D. J., and Norouzi, M. Photorealistic text-to-image diffusion models with deep language understanding, 2022. URL https://arxiv.org/abs/2205.11487.

Sainburg, T., McInnes, L., and Gentner, T. Q. Parametric UMAP embeddings for representation and semi-supervised learning. *arXiv:2009.12981 [cs, q-bio, stat]*, April 2021. URL http://arxiv.org/abs/2009.12981. arXiv: 2009.12981.

Sarfraz, S., Koulakis, M., Seibold, C., and Stiefelhagen, R. Hierarchical nearest neighbor graph embedding for efficient dimensionality reduction. In *CVPR*, pp. 336–345, 2022.

Schuhmann, C., Beaumont, R., Vencu, R., Gordon, C., Wightman, R., Cherti, M., Coombes, T., Katta, A., Mullis, C., Wortsman, M., Schramowski, P., Kundurthy, S., Crowson, K., Schmidt, L., Kaczmarczyk, R., and Jitsev, J. Laion-5b: An open large-scale dataset for training next generation image-text models, 2022. URL https://arxiv.org/abs/2210.08402.

Suhre, K., McCarthy, M. I., and Schwenk, J. M. Genetics meets proteomics: perspectives for large population-based studies. *Nature Reviews Genetics*, 22(1):19–37, 2021.

Sun, Y., Selvarajan, S., Zang, Z., Liu, W., Zhu, Y., Zhang, H., Chen, W., Chen, H., Li, L., Cai, X., et al. Artificial intelligence defines protein-based classification of thyroid nodules. *Cell discovery*, 8(1):85, 2022.

Szubert, B., Cole, J. E., Monaco, C., and Drozdov, I. Structure-preserving visualisation of high dimensional single-cell datasets. *Scientific Reports*, 9(1):8914, June 2019. ISSN 2045-2322.

Tanaka, F. H. K. d. S. and Aranha, C. Data augmentation using gans, 2019. URL https://arxiv.org/abs/1904.09135.

Theodoris, C. V., Xiao, L., Chopra, A., Chaffin, M. D., Al Sayed, Z. R., Hill, M. C., Mantineo, H., Brydon, E. M., Zeng, Z., Liu, X. S., et al. Transfer learning enables predictions in network biology. *Nature*, pp. 1–9, 2023.

Tian, Y., Sun, C., Poole, B., Krishnan, D., Schmid, C., and Isola, P. What makes for good views for contrastive learning? *NeurIPS*, 33:6827–6839, 2020.

Trabucco, B., Doherty, K., Gurinas, M., and Salakhutdinov, R. Effective data augmentation with diffusion models. *arXiv preprint arXiv:2302.07944*, 2023.

Tran, T., Pham, T., Carneiro, G., Palmer, L. J., and Reid, I. D. A bayesian data augmentation approach for learning deep models. In Guyon, I., von Luxburg, U., Bengio, S., Wallach, H. M., Fergus, R., Vishwanathan, S. V. N., and Garnett, R. (eds.), *NeurIPS 2017*, pp. 2797–2806, 2017.

Van der Maaten, L. and Hinton, G. Visualizing data using t-sne. *JMLR*, 9(11), 2008.

Wang, X. and Qi, G.-J. Contrastive learning with stronger augmentations. *TPAMI*, 45(5):5549–5560, 2022.

Wang, Y., Huang, H., Rudin, C., and Shaposhnik, Y. Understanding how dimension reduction tools work: An empirical approach to deciphering t-sne, umap, trimap, and pacmap for data visualization. *JMLR*, 22(201):1–73, 2022. URL http://jmlr.org/papers/v22/20-1061.html.

Weber, L. M. and Robinson, M. D. Comparison of clustering methods for high-dimensional single-cell flow and mass cytometry data. *Cytometry Part A*, 89(12):1084–1096, 2016.

Wickramaratne, S. and Mahmud, M. S. Conditional-gan based data augmentation for deep learning task classifier improvement using fnirs data. *Frontiers in Big Data*, 4: 659146, 07 2021. doi: 10.3389/fdata.2021.659146.

Xiong, W., He, Y., Zhang, Y., Luo, W., Ma, L., and Luo, J. Fine-grained image-to-image transformation towards visual recognition. In *CVPR 2020*, June 2020.

Xu, M., Yoon, S., Fuentes, A., and Park, D. S. A comprehensive survey of image augmentation techniques for deep learning. *Pattern Recognition*, pp. 109347, 2023.

Yamaguchi, S., Kanai, S., and Eda, T. Effective data augmentation with multi-domain learning gans. In *AAAI 2020*, pp. 6566–6574. AAAI Press, 2020. URL https://ojs.aaai.org/index.php/AAAI/article/view/6131.

Yan, Z., Xu, L., Suzuki, A., Wang, J., Cao, J., and Huang, J. Rgb color model aware computational color naming and its application to data augmentation. In *2022 IEEE International Conference on Big Data (Big Data)*, pp. 1172–1181. IEEE, 2022.

Yu, T., Cui, H., Li, J. C., Luo, Y., Jiang, G., and Zhao, H. Enzyme function prediction using contrastive learning. *Science*, 379(6639):1358–1363, 2023.

Zang, Z., Cheng, S., Lu, L., Xia, H., Li, L., Sun, Y., Xu, Y., Shang, L., Sun, B., and Li, S. Z. Evnet: An explainable deep network for dimension reduction. *TVCG*, 2022a.

Zang, Z., Li, S., Wu, D., Wang, G., Wang, K., Shang, L., Sun, B., Li, H., and Li, S. Z. Dlme: Deep local-flatness manifold embedding. *ECCV*, pp. 576–592, 2022b.

Zang, Z., Shang, L., Yang, S., Wang, F., Sun, B., Xie, X., and Li, S. Z. Boosting novel category discovery over domains with soft contrastive learning and all-in-one classifier. *ICCV*, 2023.

Zhang, H., Cissé, M., Dauphin, Y. N., and Lopez-Paz, D. mixup: Beyond empirical risk minimization. *CoRR*, abs/1710.09412, 2017. URL http://arxiv.org/abs/1710.09412.

Zhang, J. and Ma, K. Rethinking the augmentation module in contrastive learning: Learning hierarchical augmentation invariance with expanded views. In *CVPR 2022*, pp. 16650–16659, 2022.

Zhang, S., Liu, M., Yan, J., Zhang, H., Huang, L., Yang, X., and Lu, P. M-mix: Generating hard negatives via multi-sample mixing for contrastive learning. In *KDD*, pp. 2461–2470, 2022.

Zhang, Y., Chen, W., Ling, H., Gao, J., Zhang, Y., Torralba, A., and Fidler, S. Image gans meet differentiable rendering for inverse graphics and interpretable 3d neural rendering. In *ICLR*, 2021a.

Zhang, Y., Ling, H., Gao, J., Yin, K., Lafleche, J., Barriuso, A., Torralba, A., and Fidler, S. Datasetgan: Efficient labeled data factory with minimal human effort. In *CVPR*, pp. 10145–10155, 2021b.

Zhang, Y., Zhou, D., Hooi, B., Wang, K., and Feng, J. Expanding small-scale datasets with guided imagination, 2023a.

Zhang, Y., Zhu, H., and Yu, S. Adaptive data augmentation for contrastive learning. *ICASSP 2023*, pp. 1–5, 2023b.

Zheng, Z., Zheng, L., and Yang, Y. Unlabeled samples generated by GAN improve the person re-identification baseline in vitro. In *ICCV 2017*, pp. 3774–3782. IEEE Computer Society, 2017. doi: 10.1109/ICCV.2017.405. URL https://doi.org/10.1109/ICCV.2017.405.

Zheng, Z., Le, N. Q. K., and Chua, M. C. H. Maskdna-pgd: An innovative deep learning model for detecting dna methylation by integrating mask sequences and adversarial pgd training as a data augmentation method. *Chemometrics and Intelligent Laboratory Systems*, 232: 104715, 2023.

Appendix

Contents

A. Appendix: Related Works

Generative Models Generative models have been the subject of growing interest and rapid advancement. Earlier methods, including VAEs (Kingma & Welling, 2014) and GANs (Goodfellow et al., 2014), showed initial promise generating realistic images, and were scaled up in terms of resolution and sample quality (Brock et al., 2019; Razavi et al., 2019). Despite the power of these methods, many recent successes in photorealistic image generation were the result of diffusion models (Ho et al., 2020; Nichol & Dhariwal, 2021; Saharia et al., 2022; Nichol et al., 2022; Ramesh et al., 2022). Diffusion models have been shown to generate higher-quality samples compared to their GAN counterparts (Dhariwal & Nichol, 2021), and developments like classifier free guidance (Ho & Salimans, 2022) have made text-to-image generation possible. Recent emphasis has been on training these models with internet-scale datasets like LAION-5B (Schuhmann et al., 2022). Generative models trained at internet-scale (Rombach et al., 2022; Saharia et al., 2022; Nichol et al., 2022; Ramesh et al., 2022) have unlocked several application areas where photorealistic generation is crucial.

Synthetic Image Data Generation Training neural networks on synthetic data from generative models was popularized using GANs (Antoniou et al., 2017; Tran et al., 2017; Zheng et al., 2017). Various applications for synthetic data generated from GANs have been studied, including representation learning (Jahanian et al., 2022), inverse graphics (Zhang et al., 2021a), semantic segmentation (Zhang et al., 2021b), and training classifiers (Tanaka & Aranha, 2019; Dat et al., 2019; Yamaguchi et al., 2020; Besnier et al., 2020; Xiong et al., 2020; Wickramaratne & Mahmud, 2021; Haque, 2021). More recently, synthetic data from diffusion models has also been studied in a few-shot setting (He et al., 2022). These works use generative models that have likely seen images of target classes and, to the best of our knowledge, we present the first analysis for synthetic data on previously unseen concepts. (Du et al., 2023) proposed the DREAM-OOD framework, which uses diffusion models to generate photo-realistic outliers from in-distribution data for improved OOD detection. By learning a text-conditioned latent space, it visualizes imagined outliers directly in pixel space, showing promising results in empirical studies. (Zhang et al., 2023a) developed the Guided Imagination Framework (GIF) using generative models like DALL-E2 and Stable Diffusion for dataset expansion, enhancing accuracy in both natural and medical image datasets.

Synthetic Biology Data Generation The realm of synthetic biology has witnessed a surge in the utilization of data-driven approaches, particularly with the advent of advanced computational models. The generation of synthetic biological data has been instrumental in predicting protein structures (McGibbon et al., 2023). The use of Generative Adversarial Networks (GANs) has also found its way into this domain, aiding in the c reation of synthetic DNA sequences (Zheng et al., 2023; Li & Zhang, 2022; Han et al., 2019) and simulating cell behaviors (Botton et al., 2022). Furthermore, the integration of machine learning with synthetic biology has paved the way for innovative solutions in drug discovery (Blanco-Gonzalez et al., 2023; McGibbon et al., 2023). Unlike the synthetic image data generation, where models have often seen images of target classes, synthetic biology data generation often grapples with the challenge of generating data for entirely novel biological entities. This presents a unique set of challenges and opportunities, pushing the boundaries of what synthetic data can achieve in the realm of biology.

B. Appendix: Details of Contrastive Learning and Soft Contrastive Learning

B.1. The t-kernel similarity in soft contrastive learning

To map the high-dimensional embedding vector to a probability value, a kernel function $\mathcal{S}(\cdot)$ is used. In this paper, we use the t-distribution kernel function $\mathcal{S}^{\nu}(\cdot)$ because it exposes the degrees of freedom and allows us to adjust the closeness of the distribution in the dimensionality reduction mapping (Li et al., 2021). The t-distribution kernel function is defined as follows,

$$\mathcal{S}(\mathbf{z}_i, \mathbf{z}_j) = \Gamma\left((\nu+1)/2\right)\left(1+\|z_i - z_j\|_2^2/\nu\right)^{-\frac{\nu+1}{2}} / \sqrt{\nu\pi}\Gamma\left(\nu/2\right), \tag{9}$$

where $\Gamma(\cdot)$ is the Gamma function. The degrees of freedom ν control the shape of the kernel function. The different degrees of freedom (ν^y, ν^z) is used in $\mathcal{R}^y$ and $\mathcal{R}^z$ for the dimensional reduction mapping.

B.2. Why Soft Contrastive Learning is a softened version of Contrastive Learning

Lemma B.1. *Let* $\mathcal{L}_{\text{cl}} = -\log \mathcal{Q}\left(\mathbf{z}_i, \mathbf{z}_i^+\right) + \log\left[\mathcal{Q}\left(\mathbf{z}_i, \mathbf{z}_i^+\right) + \sum_{\mathbf{z}_i^- \in V^-} \mathcal{Q}\left(\mathbf{z}_i, \mathbf{z}_i^-\right)\right]$ *and* $\mathcal{L}_{\text{cl}}^p = -\sum_{j=1}^{N_K+1}\left\{\mathcal{H}_{ij}\log Q_{ij} + (1-\mathcal{H}_{ij})\log \dot{Q}_{ij}\right\}$ *Then* $\lim_{x\to\infty}\mathcal{L}_{\text{cl}} - \mathcal{L}_{\text{cl}}^p = 0$.

Proof. We start with $L_{\text{CL}} = -\log\frac{\exp(S(z_i,z_j))}{\sum_{k=1}^{N_K}\exp(S(z_i,z_k))}$ (Eq. (3)), then

$$L_{\text{CL}} = \log N_K - \log\frac{\exp(S(z_i,z_j))}{\frac{1}{N_K}\sum_{k=1}^{N_K}\exp(S(z_i,z_k))}.$$

We are only concerned with the second term that has the gradient. Let (i,j) are positive pair and $(i,k_1),\cdots,(i,k_N)$ are negative pairs. The overall loss associated with point i is:

$$\begin{aligned}
&-\log\frac{\exp(S(z_i,z_j))}{\frac{1}{N_K}\sum_{k=1}^{N_K}\exp(S(z_i,z_k))}\\
=&-\left[\log\exp(S(z_i,z_j)) - \log\frac{1}{N_K}\sum_{k=1}^{N_K}\exp(S(z_i,z_k))\right]\\
=&-\left[\log\exp(S(z_i,z_j)) - \sum_{k=1}^{N_K}\log\exp(S(z_i,z_k)) + \sum_{k=1}^{N_K}\log\exp(S(z_i,z_k)) - \log\frac{1}{N_K}\sum_{k=1}^{N_K}\exp(S(z_i,z_k))\right]\\
=&-\left[\log\exp(S(z_i,z_j)) - \sum_{k=1}^{N_K}\log\exp(S(z_i,z_k)) + \log\Pi_{k=1}^{N_K}\exp(S(z_i,z_k)) - \log\frac{1}{N_K}\sum_{k=1}^{N_K}\exp(S(z_i,z_k))\right]\\
=&-\left[\log\exp(S(z_i,z_j)) - \sum_{k=1}^{N_K}\log\exp(S(z_i,z_k)) + \log\frac{\Pi_{k=1}^{N_K}\exp(S(z_i,z_k))}{\frac{1}{N_K}\sum_{k=1}^{N_K}\exp(S(z_i,z_k))}\right]
\end{aligned}$$

We focus on the case where the similarity is normalized, $S(z_i,z_k)\in[0,1]$. The data i and data k is the negative samples, then $S(z_i,z_k)$ is near to 0, $\exp(S(z_i,z_k))$ is near to 1, thus the $\frac{\Pi_{k=1}^{N_K}\exp(S(z_i,z_k))}{\frac{1}{N}\sum_{k=1}^{N_K}\exp(S(z_i,z_k))}$ is near to 1, and $\log\frac{\Pi_{k=1}^{N_K}\exp(S(z_i,z_k))}{\frac{1}{N}\sum_{k=1}^{N_K}\exp(S(z_i,z_k))}$ near to 0. We have

$$L_{\text{CL}} \approx -\left[\log\exp(S(z_i,z_j)) - \sum_{k=1}^{N_K}\log\exp(S(z_i,z_k))\right]$$

We denote ij and ik by a uniform index and use $\mathcal{H}_{ij}$ to denote the homology relation of ij.

$$\begin{aligned}
L_{\text{CL}} &\approx -\left[\log\exp(S(z_i,z_j)) - \sum_{k=1}^{N_K}\log\exp(S(z_i,z_k))\right]\\
&\approx -\left[\mathcal{H}_{ij}\log\exp(S(z_i,z_j)) - \sum_{j=1}^{N_K}(1-\mathcal{H}_{ij})\log\exp(S(z_i,z_j))\right]\\
&\approx -\left[\sum_{j=1}^{N_K+1}\{\mathcal{H}_{ij}\log\exp(S(z_i,z_j)) + (1-\mathcal{H}_{ij})\log\{\exp(-S(z_i,z_j))\}\}\right]
\end{aligned}$$

we define the similarity of data i and data j as $Q_{ij} = \exp(S(z_i,z_j))$ and the dissimilarity of data i and data j as $\dot{Q}_{ij} = \exp(-S(z_i,z_j))$.

$$L_{\text{CL}} \approx -\left[\sum_{j=1}^{N_K+1}\left\{\mathcal{H}_{ij}\log Q_{ij} + (1-\mathcal{H}_{ij})\log \dot{Q}_{ij}\right\}\right]$$

□

The proposed SCL loss is a smoother CL loss:

This proof tries to indicate that the proposed SCL loss is a smoother CL loss. We discuss the differences by comparing the two losses to prove this point. the forward propagation of the network is, $z_i = H(\hat{z}_i), \hat{z}_i = F(x_i), z_j = H(\hat{z}_j), \hat{z}_j = F(x_j)$. We found that we mix y and $\hat{z}$ in the main text, and we will correct this in the new version. So, in this section $z_i = H(y_i), y_i = F(x_i), z_j = H(y_j), y_j = F(x_j)$ is also correct.

Let $H(\cdot)$ satisfy K-Lipschitz continuity, then $d^z_{ij} = k^* d^y_{ij}, k^* \in [1/K, K]$, where k^* is a Lipschitz constant. The difference between L_{SCL} loss and L_{CL} loss is,

$$L_{\text{CL}} - L_{\text{SCL}} \approx \sum_j \left[\left(\mathcal{H}_{ij} - [1+(e^\alpha-1)\mathcal{H}_{ij}]\kappa\left(d^y_{ij}\right)\right)\log\left(\frac{1}{\kappa\left(d^z_{ij}\right)} - 1\right)\right]. \tag{10}$$

Because the $\alpha > 0$, the proposed SCL loss is the soft version of the CL loss. if $\mathcal{H}_{ij} = 1$, we have:

$$(L_{\text{CL}} - L_{\text{SCL}})|_{\mathcal{H}_{ij}=1} = \sum\left[\left((1-e^\alpha)\kappa\left(k^* d^z_{ij}\right)\right)\log\left(\frac{1}{\kappa\left(d^z_{ij}\right)} - 1\right)\right] \tag{11}$$

then:

$$\lim_{\alpha\to 0}(L_{\text{CL}} - L_{\text{SCL}})|_{\mathcal{H}_{ij}=1} = \lim_{\alpha\to 0}\sum\left[\left((1-e^\alpha)\kappa\left(k^* d^z_{ij}\right)\right)\log\left(\frac{1}{\kappa\left(d^z_{ij}\right)} - 1\right)\right] = 0 \tag{12}$$

Based on Eq.(12), we find that if i, j is neighbor ($\mathcal{H}_{ij} = 1$) and $\alpha \to 0$, there is no difference between the CL loss L_{CL} and SCL loss L_{SCL}. When if $\mathcal{H}_{ij} = 0$, the difference between the loss functions will be the function of d^z_{ij}. The CL loss L_{CL} only minimizes the distance between adjacent nodes and does not maintain any structural information. The proposed SCL loss considers the knowledge both comes from the output of the current bottleneck and data augmentation, thus less affected by view noise.

Details of Eq. (10). Due to the very similar gradient direction, we assume $\dot{Q}_{ij} = 1 - Q_{ij}$. The contrastive learning loss is written as,

$$L_{\text{CL}} \approx -\sum\left\{\mathcal{H}_{ij}\log Q_{ij} + (1-\mathcal{H}_{ij})\log(1-Q_{ij})\right\} \tag{13}$$

where $\mathcal{H}_{ij}$ indicates whether i and j are augmented from the same original data.

The SCL loss is written as:

$$L_{\text{SCL}} = -\sum\left\{P_{ij}\log Q_{ij} + (1-P_{ij})\log(1-Q_{ij})\right\} \tag{14}$$

According to Eq. (4) and Eq. (5), we have

$$\begin{aligned} P_{ij} &= R_{ij}\kappa(d^y_{ij}) = R_{ij}\kappa(y_i, y_j), R_{ij} = \begin{cases} e^\alpha & \text{if } \mathcal{H}(x_i, x_j) = 1 \\ 1 & \text{otherwise} \end{cases}, \\ Q_{ij} &= \kappa(d^z_{ij}) = \kappa(z_i, z_j), \end{aligned} \tag{15}$$

For ease of writing, we use distance as the independent variable, $d^y_{ij} = \|y_i - y_j\|_2$, $d^z_{ij} = \|z_i - z_j\|_2$.

The difference between the two loss functions is:

$$
\begin{aligned}
& L_{\mathrm{CL}}-L_{\mathrm{SCL}} \\
=&-\sum\left[\mathcal{H}_{ij}\log\kappa\left(d_{ij}^{z}\right)+\left(1-\mathcal{H}_{ij}\right)\log\left(1-\kappa\left(d_{ij}^{z}\right)\right)-R_{ij}\kappa\left(d_{ij}^{y}\right)\log\kappa\left(d_{ij}^{z}\right)-\left(1-R_{ij}\kappa\left(d_{ij}^{y}\right)\right)\log\left(1-\kappa\left(d_{ij}^{z}\right)\right)\right] \\
=&-\sum\left[\left(\mathcal{H}_{ij}-R_{ij}\kappa\left(d_{ij}^{y}\right)\right)\log\kappa\left(d_{ij}^{z}\right)+\left(1-\mathcal{H}_{ij}-1+R_{ij}\kappa\left(d_{ij}^{y}\right)\right)\log\left(1-\kappa\left(d_{ij}^{z}\right)\right)\right] \\
=&-\sum\left[\left(\mathcal{H}_{ij}-R_{ij}\kappa\left(d_{ij}^{y}\right)\right)\log\kappa\left(d_{ij}^{z}\right)+\left(R_{ij}\kappa\left(d_{ij}^{y}\right)-\mathcal{H}_{ij}\right)\log\left(1-\kappa\left(d_{ij}^{z}\right)\right)\right] \\
=&-\sum\left[\left(\mathcal{H}_{ij}-R_{ij}\kappa\left(d_{ij}^{y}\right)\right)\left(\log\kappa\left(d_{ij}^{z}\right)-\log\left(1-\kappa\left(d_{ij}^{z}\right)\right)\right)\right] \\
=&\sum\left[\left(\mathcal{H}_{ij}-R_{ij}\kappa\left(d_{ij}^{y}\right)\right)\log\left(\frac{1}{\kappa\left(d_{ij}^{z}\right)}-1\right)\right]
\end{aligned}
\tag{16}
$$

Substituting the relationship between $\mathcal{H}_{ij}$ and R_{ij}, $R_{ij}=1+(e^{\alpha}-1)\mathcal{H}_{ij}$, we have

$$
L_{\mathrm{CL}}-L_{\mathrm{SCL}}=\sum\left[\left(\mathcal{H}_{ij}-[1+(e^{\alpha}-1)\mathcal{H}_{ij}]\kappa\left(d_{ij}^{y}\right)\right)\log\left(\frac{1}{\kappa\left(d_{ij}^{z}\right)}-1\right)\right] \tag{17}
$$

We assume that network $H(\cdot)$ to be a Lipschitz continuity function, then

$$
\frac{1}{K}H(d_{ij}^{z})\leq d_{ij}^{y}\leq KH(d_{ij}^{z})\quad\forall i,j\in\{1,2,\cdots,N\} \tag{18}
$$

We construct the inverse mapping of $H(\cdot)$ to $H^{-1}(\cdot)$,

$$
\frac{1}{K}d_{ij}^{z}\leq d_{ij}^{y}\leq Kd_{ij}^{z}\quad\forall i,j\in\{1,2,\cdots,N\} \tag{19}
$$

and then there exists k^{*}:

$$
d_{ij}^{y}=k^{*}d_{ij}^{z}\quad k^{*}\in[1/K,K]\quad\forall i,j\in\{1,2,\cdots,N\} \tag{20}
$$

Substituting the Eq.(20) into Eq.(17).

$$
L_{\mathrm{CL}}-L_{\mathrm{SCL}}=\sum\left[\left(\mathcal{H}_{ij}-[1+(e^{\alpha}-1)\mathcal{H}_{ij}]\kappa\left(k^{*}d_{ij}^{z}\right)\right)\log\left(\frac{1}{\kappa\left(d_{ij}^{z}\right)}-1\right)\right] \tag{21}
$$

C. Appendix: Details of DNA Sequence Experiments

C.1. Experimental Setups and Datasets Information

In our research, we utilized the 'genomic-benchmarks' dataset, a comprehensive collection of curated sequence classification datasets specifically designed for genomic studies. This repository encompasses a range of datasets derived from both novel compilations mined from publicly accessible databases and existing datasets gathered from published studies. It focuses on regulatory elements such as promoters, enhancers, and open chromatin regions from three model organisms: humans, mice, and roundworms. Accompanying these datasets is a simple convolutional neural network provided as a baseline model, enabling researchers to benchmark their algorithms effectively. The entire collection is made available as a Python package, facilitating easy integration with popular deep learning libraries and serving as a valuable resource for the genomics research community.

The initiative behind the 'genomic-benchmarks' dataset aims to address the critical need for standardized benchmarks in genomics, akin to the role that carefully curated benchmarks have played in advancing other biological fields, notably demonstrated by AlphaFold's success in protein folding. By offering a structured and easily accessible set of benchmarks, this collection not only promotes comparability and reproducibility in machine learning applications within genomics but also lowers the entry barrier for researchers new to this domain. Consequently, it fosters a healthy competitive environment that is likely to spur innovation and discovery in genomic research, paving the way for significant advancements in the understanding and annotation of genomes.

Table 6. GenomicBenchmarks Dataset Metadata

Name	Acronyms	**Num. Seqs**	**Num. Classes**	**Median Len**	**Std**
dummy_mouse_enhancers_ensembl	MoEnEn	1,210	2	2,381	984.4
demo_coding_vs_intergenomic_seqs	CoIn	100,000	2	200	0
demo_human_or_worm	HuWo	100,000	2	200	0
human_enhancers_cohn	HuEnCo	27,791	2	500	0
human_enhancers_ensembl	HuEnEn	154,842	2	269	122.6
human_ensembl_regulatory	HuEnRe	289,061	3	401	184.3
human_nontata_promoters	HuNoPr	36,131	2	251	0
human_ocr_ensembl	HuOcEn	174,756	2	315	108.1

D. Appendix: Details of Vision Experiments

D.1. Dataset Setups

Experiments are performed on CIFAR-10 [CF10][3] and CIFAR-100[4] [CF100] (Krizhevsky et al., 2009), STL10[5] (Coates et al., 2011), TinyImageNet[6] [TINet] (Le & Yang, 2015) dataset.

To compare with the two different baseline methods, the setting of the dataset is shown in Table. 7.

Table 7. Dataset setting of linear-test Performance.

Dataset	Train Split	Test Split	Train Samples	Test Samples	Classes
CF10	Train	Test	50,000	10,000	10
CF100	Train	Test	50,000	10,000	100
STL10	Train + Unlabeled	Test	5,000+100,000	8,000	10
TINet	Train	Test	100,000	100,000	200

Table 8. Dataset setting of clustering test.

Dataset	Train & Test Split	Train & Test Samples	Classes
CF10	Train+Test	60,000	10
CF100	Train+Test	60,000	20
STL10	Train+Test	13,000	10
TIN	Train	100,000	200

D.2. Baseline Methods and Implementation Details

The contrastive learning methods, including SimCLR (Chen et al., 2020), MOCO v2 (He et al., 2020b), BYOL (Grill et al., 2020), SimSiam (Chen & He, 2021), and DLME (Zang et al., 2022b) are chosen for comparison. The SimC.+Mix. and MoCo.+Mix. are SimCLR and MoCoV2 with Mixup data augmentation which processed by (Zhang et al., 2022). The

[3]https://www.cs.toronto.edu/ kriz/cifar.html
[4]https://www.cs.toronto.edu/ kriz/cifar.html
[5]https://cs.stanford.edu/ acoates/stl10/
[6]https://www.kaggle.com/c/tiny-imagenet

SimC.+Dif. and MoCo.+Mix. are SimCLR and MoCoV2 with DiffAug data augmentation. Improvements over the best baseline are shown in parentheses.

For the Linear-test performance assessment, we followed a procedure similar to SimCLR (Chen et al., 2020). We evaluated the model's representations linearly on top of the frozen features. This ensures that the quality of the representations is attributed only to the pre-training task, without any influence from potential fine-tuning. We used the ResNet-50 (He et al., 2015) backbone as the encoder and a standard diffusion backbone as diffusion model (in code below). In contrast, for DiffAug, its semantic encoder served as the contrastive learning backbone, trained using DiffAug-augmented images. For the kMeans clustering evaluation, we extracted feature vectors from the models, leaving out the top classification layer. We then applied kMeans clustering to these features. The primary metric for evaluation was clustering accuracy.

Listing 1. DiffusionModel for Vision Task

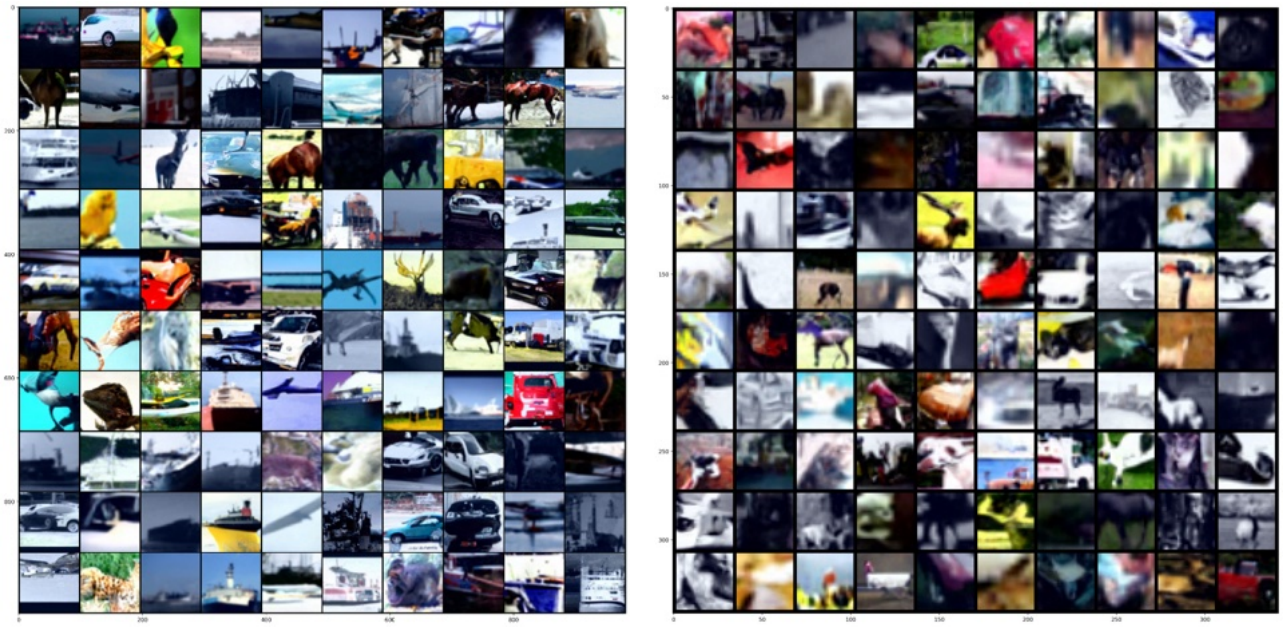
```
class DiffusionModelVision(nn.Module):
    def __init__(self, c_in=3, c_out=3, time_dim=256):
        super().__init__()
        self.time_dim = time_dim
        self.remove_deep_conv = remove_deep_conv
        self.inc = DoubleConv(c_in, 16)
        self.down1 = Down(16, 32)
        self.sa1 = SelfAttention(32)
        self.down2 = Down(32, 64)
        self.sa2 = SelfAttention(64)
        self.down3 = Down(64, 64)
        self.sa3 = SelfAttention(64)
        self.up1 = Up(128, 32)
        self.sa4 = SelfAttention(32)
        self.up2 = Up(64, 16)
        self.sa5 = SelfAttention(16)
        self.up3 = Up(32, 16)
        self.sa6 = SelfAttention(16)
        self.outc = nn.Conv2d(16, c_out, kernel_size=1)
        def pos_encoding(self, t, channels):
        inv_freq = 1.0 / (10000 ** (
            torch.arange(0, channels, 2,
            device=one_param(self).device).float() / channels
            ))
        pos_enc_a = torch.sin(t.repeat(1, channels // 2) * inv_freq)
        pos_enc_b = torch.cos(t.repeat(1, channels // 2) * inv_freq)
        pos_enc = torch.cat([pos_enc_a, pos_enc_b], dim=-1)
        return pos_enc

    def forward(self, x, t):
        t = t.unsqueeze(-1)
        t = self.pos_encoding(t, self.time_dim)
        return self.unet_forwad(x, t)
```

Our training strategy is as follows: A-step: 200 epochs → B-Step: 400 epoch → A-Step: 800 epoch. Continued training will further improve performance, but we did not increase the amount of computation due to computational resource constraints. The time loss of the method does improve due to the use of the diffusion model. However, on small datasets, this boost is acceptable. In this way at the same time DiffAug gives the possibility to accomplish unsupervised comparison learning training on small datasets.

Table 9. Details of the training process in vision dataset.

CF10	ν	Learning Rate	Weight Decay	Batch Size	GPU	pix	Training Time
CF10	1	0.001	1e-6	256	1*V100	32×32	7.1 hours
CF100	2	0.001	1e-6	256	1*V100	32×32	7.2 hours
STL10	5	0.001	1e-6	256	1*V100	96×96	15.1 hours
TINet	3	0.001	1e-6	256	1*V100	64×64	20.6 hours



STL10 (96*96*3) CF10 (32*32*3)

Figure 6. **The display of original and generated images illustrates that DiffAug generates semantically similar augmented images.** Ori means original image and Aug1, Aug2 and Aug3 are augmentated images. More detailed results are in the appendix.

D.3. Data Augmentation of the Compared Methods

BYOL augmentation. The BYOL augmentation method is a hand-designed method. It is composed of four parts: random cropping, left-right flip, color ji

- Random cropping: A random patch of the image is selected, with an area uniformly sampled between 8% and 100% of that of the original image, and an aspect ratio logarithmically sampled between 3/4 and 4/3. This patch is then resized to the target size of 224 $\times$ 224 using bicubic interpolation.
- Optional left-right flip.
- Color jittering: The brightness, contrast, saturation, and hue of the image are shifted by a uniformly random offset applied to all the pixels of the same image. The order in which these shifts are performed is randomly selected for each patch.
- Color dropping: An optional conversion to grayscale. When applied, the output intensity for a pixel (r, g, b) corresponds to its luma component, computed as 0.2989r + 0.5870g + 0.1140b1.

SimCLR augmentation.

- Random Cropping: This involves taking a random crop of the image and then resizing it back to the original size. This can be seen as a combination of zooming and spatial location changes.
- Random Flipping: Randomly flip the image horizontally.
- Color Distortion: Apply a random color distortion. In the SimCLR paper, they use a combination of random brightness, random contrast, random saturation, and random hue changes. The strength of these distortions is controlled by a factor.
- Gaussian Blur: Apply a random Gaussian blur to the image. The extent of blurring is controlled by a factor.

MoCo v2 augmentation. For MoCo v2, the data augmentations are similar to those used in SimCLR, but there might be slight differences in implementation details. Here are the main augmentations used in MoCo v2:

- Random Cropping: This involves taking a random crop of the image and then resizing it back to the original size. Random Flipping: Randomly flip the image horizontally.
- Color Jitter: Randomly change the brightness, contrast, saturation, and hue of the image.
- Gaussian Blur: Apply Gaussian blur to the image with a certain probability.
- Solarization: This is an augmentation introduced in MoCo v2. It inverts pixel values above a threshold, which can create a unique visual effect.

MAE augmentation. The core idea behind MAE is to mask out parts of an image and then train an autoencoder to reconstruct the original image from the masked version. This is somewhat analogous to the masked language modeling task used in models like BERT for NLP, where parts of the text are masked out and the model is trained to predict the masked words.

E. Appendix: Time Consumption of Vision Experiments

F. Appendix: Details of Biology Experiments

F.1. Dataset Setups

Experiments are performed on biological datasets, including MC1374[7] (Han et al., 2018), GA1457[8] (Rouillard et al., 2016), SAM[9] (Weber & Robinson, 2016), , and HCL500[10] (Han et al., 2020) datasets.

To ensure a fair comparison, we first embed the data into a 2D space using the method under evaluation. We then assess the method's performance through 10-fold cross-validation. Classification accuracy is determined by applying a linear SVM classifier in the latent space, while clustering accuracy is gauged using k-means clustering in the same space. Further details about the datasets, baseline methods, and evaluation metrics can be found in Table 10.

Table 10. Datasets information of simple manifold embedding task

Dataset	Train Samples	Test Samples	Input Dimension	Class Number in label
MC1374	24,000	6,000	1,374	98
GA1457	8,510	2,127	1,457	49
SAM561	69,491	17,373	561	52
HCL500	48,000	12,000	500	45

F.2. Baseline Methods and Implementation Details

Dimension reduction methods that have been widely used on biological analyze are compared, including kPCA (Halko et al., 2010), Ivis (Szubert et al., 2019), PHATE (Moon & van Dijk, 2019), PUMAP (Sainburg et al., 2021), PaCMAP (Wang et al., 2022), DMTEV (Zang et al., 2022a) and hNNE (Sarfraz et al., 2022).

For DiffAug, both the semantic encoder Enc$(\cdot)$, and the diffusion generator Gen$(\cdot)$, are implemented using a Multi-Layer Perceptron (MLP). Their respective architectures are defined as: Enc$(\cdot)$: [-1, 500, 300, 80]. The Gen$(\cdot)$: is defined below,

[7]https://bis.zju.edu.cn/MCA/
[8]https://maayanlab.cloud/Harmonizome/gene/GAST
[9]https://github.com/abbioinfo/CyAnno
[10]https://db.cngb.org/HCL/

Listing 2. DiffusionModel for Biology Task

```
class AE(nn.Module):
def __init__( self,in_dim, mid_dim=2000, time_step=1000,):
    super().__init__()
    self.enc1 = self.diff_block(in_dim, mid_dim)
    self.enc2 = self.diff_block(in_dim, mid_dim)
    self.enc3 = self.diff_block(in_dim, mid_dim)
    self.enc4 = self.diff_block(in_dim, mid_dim)

    self.dec1 = self.diff_block(in_dim, mid_dim)
    self.dec2 = self.diff_block(in_dim, mid_dim)
    self.dec3 = self.diff_block(in_dim, mid_dim)
    self.dec4 = self.diff_block(in_dim, mid_dim)
    self.time_encode = nn.Embedding(time_step, in_dim)

def diff_block(in_dim, mid_dim):
    return nn.Sequential(
    nn.LeakyReLU(), nn.InstanceNorm1d(in_dim),
    nn.Linear(in_dim, mid_dim), nn.LeakyReLU(),
    nn.InstanceNorm1d(mid_dim), nn.Linear(mid_dim, in_dim),)

def forward(self, input, time, cond=None):
    input_shape = input.shape
    if len(input.size()) > 2:
        input = input.view(input.size(0), -1)
    ti = self.time_encode(time)
    cd = self.cond_model(cond).reshape(input.shape[0], -1)
    ee1 = self.enc1(input + ti + cd)
    ee2 = self.enc2(ee1 + ti+ cd) + ee1
    ee3 = self.enc3(ee2 + ti+ cd) + ee1 + ee2
    ee4 = self.enc4(ee3 + ti+ cd) + ee1 + ee2 + ee3

    ed1 = self.dec1(ee4 + ti+ cd)
    ed2 = self.dec2(ed1 + ti+ cd) + ee3 + ed1
    ed3 = self.dec3(ed2 + ti+ cd) + ee2 + ed1 + ed2
    ed4 = self.dec4(ed3 + ti+ cd) + ee1 + ed1 + ed2 + ed3
    return ed4.reshape(input_shape)
```

To assess the efficacy of the proposed methods, following (Wang et al., 2022; Sarfraz et al., 2022), we utilized linear SVM performance to evaluate the performance of differences methods. For the linear SVM evaluation, embeddings were partitioned with 90% designated for training and 10% for testing; the training set facilitated the linear SVM training, while the test set yielded the performance metrics. Detailed specifics of this configuration are elaborated in the Table 11.

Table 11. Details of the training process in biological dataset.

CF10	ν	Learning Rate	Weight Decay	Batch Size	GPU	Training Time
MC1374	1	0.0001	1e-6	300	1*V100	4.2 hours
GA1457	1	0.0001	1e-6	300	1*V100	4.6 hours
SAM561	1	0.0001	1e-6	300	1*V100	12.1 hours
HCL500	0.1	0.0001	1e-6	300	1*V100	20.1 hours

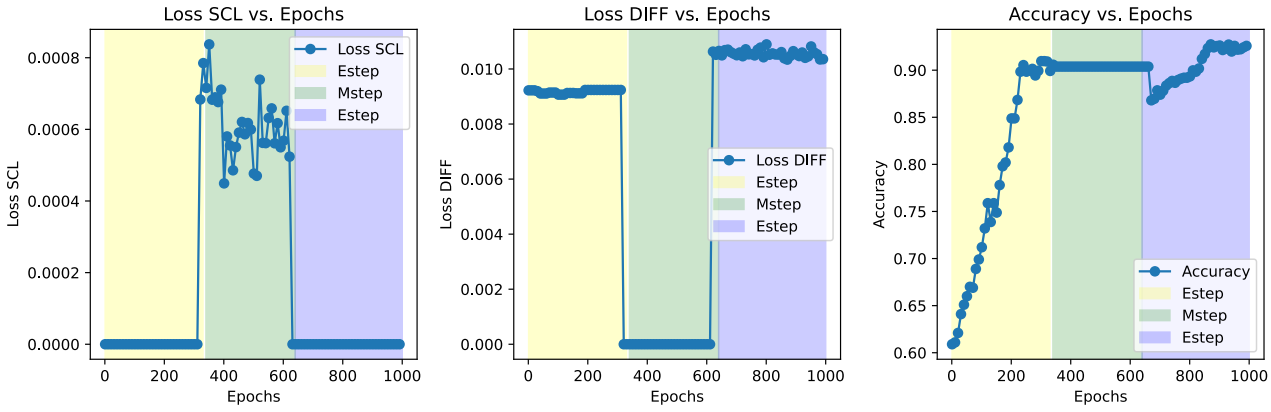


Figure 7. Training curves on the GA1457 dataset, including two Esteps and one Mstep. We can observe that the new generated data improves the correctness of E step.