Self-Supervised Representation Distribution Learning for Reliable Data Augmentation in Histopathology WSI Classification

Kunming Tang, Zhiguo Jiang, Kun Wu, Jun Shi, *Member, IEEE*, Fengying Xie, Wei Wang, Haibo Wu, and Yushan Zheng, *Member, IEEE*

***Abstract*—Multiple instance learning (MIL) based whole slide image (WSI) classification is often carried out on the representations of patches extracted from WSI with a pre-trained patch encoder. The performance of classification relies on both patch-level representation learning and MIL classifier training. Most MIL methods utilize a frozen model pre-trained on ImageNet or a model trained with self-supervised learning on histopathology image dataset to extract patch image representations and then fix these representations in the training of the MIL classifiers for efficiency consideration. However, the invariance of representations cannot meet the diversity requirement for training a robust MIL classifier, which has significantly limited the performance of the WSI classification. In this paper, we propose a Self-Supervised Representation Distribution Learning framework (SSRDL) for patch-level representation learning with an online representation sampling strategy (ORS) for both patch feature extraction and WSI-level data augmentation. The proposed method was evaluated on three datasets under three MIL frameworks. The experimental results have demonstrated that the proposed method achieves the best performance in histopathology image representation learning and data augmentation and outperforms state-of-the-art methods under different WSI classification frameworks. The code is available at https://github.com/lazytkm/SSRDL.**

***Index Terms*—Self-supervised Representation Learning, Data Augmentation, WSI Classification**

This work was partly supported by Beijing Natural Science Foundation (Grant No. 7242270), partly supported by the National Natural Science Foundation of China (Grant No. 62171007, 61901018, and 61906058), partly supported by the Fundamental Research Funds for the Central Universities of China (grant No. YWF-23-Q-1075 and JZ2022HGTB0285), partly supported by Emergency Key Program of Guangzhou Laboratory (Grant No. EKPG21-32), partly supported by Joint Fund for Medical Artificial Intelligence (Grant No. MAI2023C014), and partly supported by National Key Research and Development Program of China (Grant No. 2021YFF1201004). *(Corresponding authors: Yushan Zheng and Haibo Wu)*

Kunming Tang, Zhiguo Jiang, Kun Wu, and Fengying Xie are with the Image Processing Center, School of Astronautics, Beihang University, Beijing 100191, China, also with Tianmushan Laboratory, Hangzhou 311115, China, and also with Beijing Advanced Innovation Center on Biomedical Engineering, School of Engineering Medicine, Beihang University, Beijing 100191, China.

Jun Shi is with the School of Software, Hefei University of Technology, Hefei 230009, China.

Wei Wang and Haibo Wu are with the Department of Pathology, the First Affiliated Hospital of USTC, Division of Life Seciences and Medicine, University of Science and Technology of China, Hefei 230036, China and also with the Intelligent Pathology Institute, Division of Life Sciences and Medicine, University of Science and Technology of China, Hefei 230036, China.

Yushan Zheng is with Beijing Advanced Innovation Center on Biomedical Engineering, School of Engineering Medicine, Beihang University, Beijing 100191, China (e-mail: yszheng@buaa.edu.cn).

I. Introduction

The recent emergence of computational pathology, which integrates artificial intelligence (AI) techniques to address challenges in pathology, has shown promising progress in various applications [1]–[3]. These applications include metastasis detection [4], cancer subtyping [5]–[7], survival prediction [8], image retrieval [9], molecular alterations prediction [10] and gene mutation prediction [11]. For example, Zhang et al. [7] enhanced breast cancer subtyping with the introduction of a double-tier framework. Wang et al. [9] improved the retrieval of epithelial lesions of the uterine cervix patches using a clustering-guided contrastive learning framework. Chen et al. [8] enhanced pan-cancer survival prediction through multimodal learning. Yan et al. [11] advanced the prediction of clinically relevant gene mutations in bladder cancer by developing a hierarchical deep multi-instance learning framework. These advancements underscore the crucial role of whole slide image (WSI) classification as the foundation of computer-aided pathological diagnosis [5]–[7], [12]–[14]. Unlike natural images, a WSI typically contains billions of pixels that cannot be directly fed into deep neural networks for training due to computational resource limitations [15]–[17]. Moreover, pixel-wise annotations of histopathological images are scarce, and thereby sparse annotations are commonly used [18], [19]. Therefore, most of the current solutions for histopathology WSI classification are based on weakly supervised learning methods, especially multiple instance learning (MIL) [7], [20]–[22].

Under the paradigm of MIL, each WSI is regarded as a bag that consists of many instances of patches. Classification for the WSI is usually divided into two stages. In the first stage, patch-level representations are extracted by a pre-trained model. In the second stage, another model is built to take the representations of patches within WSI as input and finally output the prediction for the WSI. Convolutional neural network (CNN) [23] and vision transformer (ViT) [24] pre-trained on ImageNet dataset [25] are common choices for the patch feature extraction. Besides, self-supervised learning has recently demonstrated strong performance on representation learning [26]–[29] and effectiveness on histopathological image analysis [12], [21].

However, in the two stage frameworks, the patch-level representations will be fixed after feature extraction and during the entire period of WSI-level model training and inference. These

fixed patch representations make it challenging to produce data augmentation for the training of the second stage models, which has limited the performance and generalization of the WSI-level model.

Previous studies have proven that the composition of data augmentation operations is crucial for learning good representations. Furthermore, the advantages of appropriate data augmentation have been demonstrated in the field of patch-level classification [30]. Self-supervised representation learning also benefits from stronger data augmentation than supervised learning [26]. But, typical augmentations can be only conducted in image space [31]–[33]. Emerging methods of representation augmentations [34]–[36] for supervised learning have been proposed. However, in WSI analysis, especially for tasks of gene mutation prediction and survival prediction, we typically possess labels only for WSIs themselves, not for the individual patches within a WSI. In this case, their application to the patches is impractical due to the absence of labels. Given the constraints of computational resources, it's also not feasible to apply these methods directly to entire WSIs. These make the augmentations for supervised learning difficult to apply in weakly supervised WSI analysis frameworks.

To achieve data augmentation for WSI model training, several studies [37], [38] proposed generative methods to synthesize data augmentations in the embedding space, as shown in Fig. 1b. However, these methods require an additional generator to be trained and then deployed for augmentation, which increases the computational cost in the WSI-level model training. Meanwhile, some approaches [13], [39], [40] based on Mixup were proposed for WSI-level augmentation, as illustrated in Fig. 1c. But it is observed that the representations produced by Mixup are unpredictable for training, which occasionally leads to unstable performance [37]. Overall, the current methods either incur the additional computational cost of training or cause a loss of semantic information. It lacks data augmentation methods that are both effective and efficient for histopathology WSI-level model construction and application.

To address these problems, we propose a novel representation distribution learning technique for both patch-level representation learning and WSI-level data augmentation. We introduce a new concept "representation distribution", which distinguishes our approach from previous WSI augmentation methods. As shown in Fig. 1d, rather than viewing each patch representation as a single point in the feature space, we also consider the reasonable representation augmentations for each patch. This perspective allows us to encode each patch as a distribution within the feature space, denoted as "representation distribution". Specifically, a representation distribution estimator is trained during self-supervised representation learning. This estimator can predict a distribution of all the reasonable representation augmentations for each patch. Consequently, WSI data augmentation can be efficiently achieved by online sampling patch representations from these patch distributions. The proposed method was evaluated on a lung dataset with 754 WSIs and two public lung datasets with 696 WSIs and 3064 WSIs, where the proposed method achieved an increase in AUC of 8.7%, 8.5%, and 7.1% on

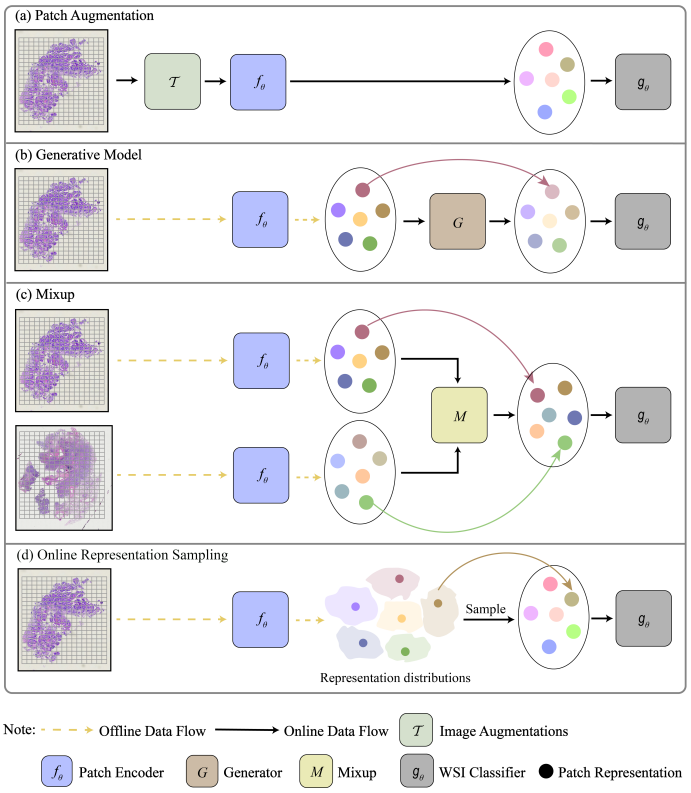


Fig. 1. Comparison between existing methods for WSI data augmentation and our method. The dotted lines indicate offline training procedures while the solid lines indicate online training procedures. (a) is the simplest WSI augmentation which is time-consuming. (b) enables WSI augmentation with an additionally trained generative model. (c) mixes two bags in multiple ways to achieve WSI augmentation. (d) is our online representation sampling strategy that dynamically samples representations from estimated distributions.

the USTC-EGFR dataset, an increase of 20.2%, 18.8%, and 26.1% on the TCGA-EGFR dataset, and an increase of 1.0%, 2.4%, and 2.4% on the TCGA-LUNG-3K dataset, respectively, under three WSI classification frameworks.

The contribution of this paper can be summarized into two aspects:

(1) We propose a representation learning framework with representation distribution estimation for histopathology WSI classification. Building on DINO [29] that is a self-supervised learning framework with augmentation-encoding branches, we enhance this architecture by adding another encoding-augmentation student branch. This new branch includes a distribution estimator specifically designed to incorporate representation augmentation into representation learning. Compared with the image-level data augmentation in DINO, the proposed method can provide more expansive while reliable augmentation and thus improve the discrimination of the patch representations and the performance of the following WSI analysis model. The visualization results show that the range of the sampled representations effectively covers the distribution of the representations extracted from different augmented views.

(2) We propose an online representation sampling (ORS) module based on the proposed representation distribution learning model. With ORS, we achieve the procedure exchange

of data augmentation and patch encoding. During the training of the WSI-level model, the representations of patches in a bag are continuously resampled from the estimated distributions, which is convenient and computationally efficient for WSI model training. The experimental results demonstrate that this data augmentation strategy can improve the generalization and robustness of the WSI classifier. Furthermore, the distribution estimator is concurrently trained with representation learning, thereby avoiding any additional training process prior to the WSI-level model.

II. RELATED WORKS

A. Self-Supervised Representation Learning

The patch encoder is crucial for MIL methods. Given the vast amount of images in ImageNet, the CNN model pre-trained on ImageNet is often considered a good choice for the patch encoder. In CLAM [5] and TransMIL [6], the patches are embedded in 1024-dimensional vector by a ResNet50 [23] model pretrained on ImageNet [25]. However, there are inevitably semantic differences between pathological and natural images, which may lead to the inapplicability of ImageNet-pretrained models in WSI analysis. Self-supervised learning (SSL) has been widely used in representation learning [41]. The representations can achieve considerable success in downstream tasks, such as classification [26], [28], [42], [43] and segmentation [44], [45].

Existing SSL methods mainly focus on contrastive learning which considers each image a different class and trains the model by discriminating them [26], [28], [43], [46]. Chen et al. [26] propose an end-to-end model SimCLR, which verifies that this architecture performs better with large batch sizes. It is deployed in DSMIL [21] to train ResNet18 for feature extraction. These discriminative approaches are similar in that they all involve calculating contrastive loss across a large number of images simultaneously.

Recent works have shown that we can learn unsupervised representations without discriminating between images [27], [29], [47], [48]. Grill et al. [27] propose a metric-learning formulation called BYOL that directly predicts the output of one view from another view. KAT [12] trains the ResNet50 using the contrastive representation learning framework proposed in BYOL [27]. Chen et al. [47] suggest that simple Siamese networks can learn meaningful representations without following the contrastive learning paradigm. Caron et al. [29] observe that self-supervised ViT [24] representations contain explicit information about the semantic segmentation of an image. Most recently, DINO [29] is proposed to utilize this information for representation learning. HIPT [15] and PAMA [49] have proven the advantage of DINO for patch representation pretraining. Thus, we take the DINO framework with the ViT architecture as our baseline model for representation learning.

B. Data Augmentation in Deep Learning

It is common knowledge that the more data a deep learning model has access to, the more effective it can be [26], [28], [29]. The amount and diversity of data affect model performance. As an effective means of expanding data, data augmentation has been widely used in both unsupervised and supervised learning.

In self-supervised representation learning, the composition of data augmentation operations has an impact on learning good representations [50], [51]. In SimCLR [26], the authors systematically study the impact of data augmentations. The augmentations involve spatial/geometric/appearance transformations, such as cropping and resizing, rotation [32], cutout, color distortion, Gaussian blur and Sobel filtering. They observe that single augmentation is not sufficient to learn good representations. BYOL [27] shows that different color histograms help the representation learning. MoCov2 [52] extends the original augmentations in MoCo [28] with the blur augmentation, which improves the baseline on ImageNet. SwAV [33] proposes a multi-crop strategy where the authors use two standard resolution crops and sample several additional low resolution crops that cover only small parts of the image. DINO [29] follows the data augmentations of BYOL [27] and multi-crop [33].

C. Data Augmentation for WSI analysis

Like data augmentation methods for natural images, the typical approach to WSI data augmentation involves continuously extracting representations from augmented patches within each bag during the training process. This is obviously inefficient for WSI model training because feature extraction takes up the vast majority of the training time. Thereby, they are only utilized in the stage of WSI preprocessing to increase the diversity of patches as in ABMIL [20]. The data augmentation methods specially designed for WSI analysis can be divided into two categories. The former is realized with generative models, while the latter generates new subsets from bags through various mixing approaches. Specifically, DAGAN [38] additionally trains a data augmentation generative adversarial network that can synthesize data augmentations in the feature space. AugDiff [37] employs the diffusion model to improve the quality of representation augmentation and control the retention of semantic information. These generative models request additional training time and confine representation augmentation to the finite feature space associated with image augmentations. ReMix [13] proposes to mix the instance prototypes obtained by clustering with four different augmentations, but clustering may entail partial loss of semantic information and lead to unstable performances. RankMix [39] rearranges the patch representations in the WSI by the patch attention scores and then formulates the Mixup of the ranked representations, which strictly follows the paradigm of Mixup that mixes the bags with linear interpolation. Mixup-MIL [40] proposes a new augmentation strategy called Intra-Mixup that uses a single WSI to mix. However, the representations produced by Mixup are unpredictable, which may lead to unstable effects on the model performance [37].

III. METHOD

A. Overview

We aim to find an approach that enables flexible data augmentation in the feature space after the image patch encoding instead of performing image augmentation prior to patch

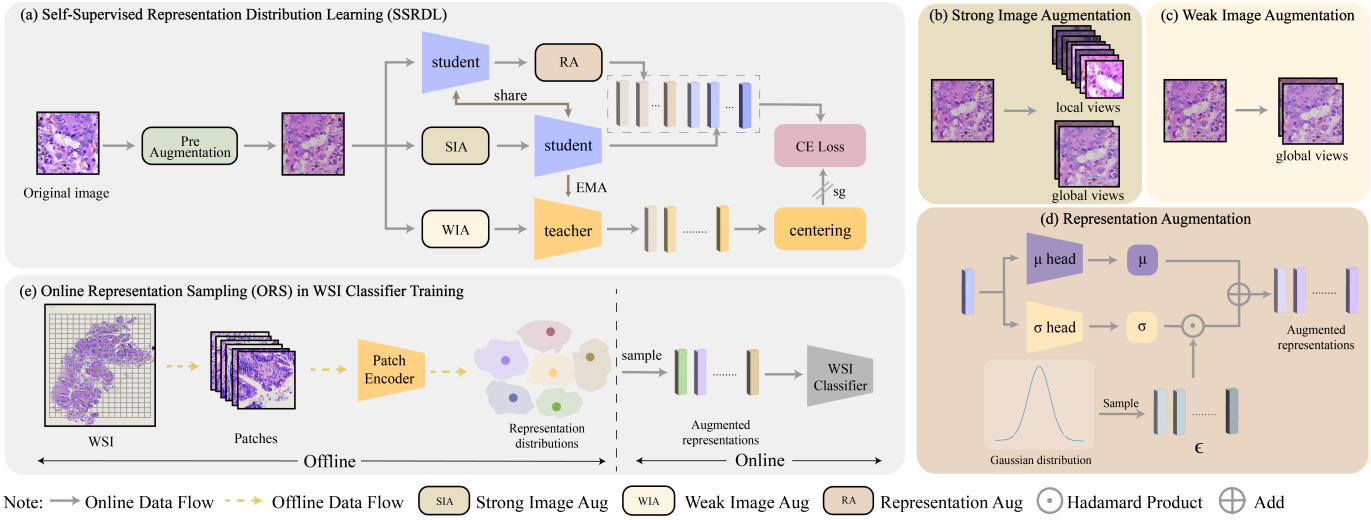


Fig. 2. Overview of the proposed representation learning and WSI data augmentation framework. (a) SSRDL: Another student branch is built by designing a representation augmentation module. The outputs of both student branches contribute to the loss calculation with the output of the teacher branch. The encoders in both student branches share weights. (b-d) Description of Augmentations: (b) and (c) illustrate the differences between strong and weak image augmentations. (d) showcases the application of the reparameterization trick in the representation augmentation module. (e) WSI Augmentation during Training: The patches within a WSI are encoded into representation distributions through an offline process. Subsequently, during the online training of the WSI classifier, patch representations are continuously updated by sampling from these distributions.

encoding. To this end, we assume that data augmentation and feature extraction are independent processes, and propose to relocate the data augmentation process after feature extraction, formulating the proposed representation distribution learning model.

As shown in Fig. 2a, the architecture of the proposed model is built based on DINO [29]. In addition to the traditional augmentation-encoding student branch, we build another encoding-augmentation student branch by designing a representation augmentation module. The self-distillation constraint is performed on both the student branches to make the representations from the encoding-augmentation branch semantically consistent with those extracted by the augmentation-encoding branch. Then, based on the trained model, we can continuously acquire diverse WSI data through online sampling representations from the distributions, as shown in Fig. 2e. This enables us to efficiently implement WSI augmentation in the training of the WSI classifier.

B. Self-Distillation Architecture

Image augmentation is the basis of our baseline DINO [29]. We first define the image augmentations required in traditional augmentation-encoding student branch and teacher branch. Given an image, we obtain $\tilde{\mathbf{x}} \in \mathbb{R}^{w\times h\times 3}$ by pre-augmentation, including flip, color distortion and gray scaling. Then, we generate a set of crops $\mathbb{V} = \mathbb{V}_g \cup \mathbb{V}_l$ for $\tilde{\mathbf{x}}$, which contains both global views $\mathbb{V}_g = \{\tilde{\mathbf{x}}_1^g, \tilde{\mathbf{x}}_2^g\}$ and local views $\mathbb{V}_l = \{\tilde{\mathbf{x}}_1^l, \tilde{\mathbf{x}}_2^l, \ldots, \tilde{\mathbf{x}}_N^l\}$. Referring to the knowledge distillation paradigm, we train a student encoder f_{θ_s} to match the output of a given teacher encoder f_{θ_t}, parameterized by θ_s and θ_t, respectively. All crops in $\mathbb{V}$ are fed into the student network while only the global views are fed into the teacher network. Both encoders output D-dimensional representations, which are represented by sets:

$$\mathbb{Z}_t = \{\mathbf{z}_i^t = f_{\theta_t}(\mathbf{x}) | \mathbf{x} \in \mathbb{V}_g\} \tag{1}$$

$$\mathbb{Z}_s = \{\mathbf{z}_i^s = f_{\theta_s}(\mathbf{x}) | \mathbf{x} \in \mathbb{V}\} \tag{2}$$

The resolution of the patches is 224^2, which matches the resolution of the images used in DINO [29]. Then, we follow the standard setting of DINO for multi-crop by using 2 global views with a resolution of 224^2 covering a large area (e.g. greater than 50%) of the patch and several local views with a resolution of 96^2 covering a small area (e.g. less than 50%) of the patch.

Then, we adapt the augmentation paradigm to self-supervised learning. We align the probability distribution of the student network output $\mathbf{P}_s$ with the probability distribution of the teacher network output $\mathbf{P}_t$, encouraging "local-to-global" correspondences. The probability distribution over D dimensions denoted by $\mathbf{P}_s$ is defined by a softmax function:

$$\mathbf{P}_s(\mathbf{z})^{(i)} = \frac{\exp\left(\mathbf{z}^{(i)}/\tau_s\right)}{\sum_{d=1}^{D}\exp\left(\mathbf{z}^{(d)}/\tau_s\right)}, \tag{3}$$

where $\mathbf{P}_s(\mathbf{z})^{(i)}$ is the i-th dimension of $\mathbf{P}_s$ and $\tau_s > 0$ is a temperature parameter that controls the sharpness of the probability distribution. Similar formulas hold for $\mathbf{P}_t$ with τ_t. We minimize the objective function:

$$L_c = \sum_{i=1}^{2}\sum_{\substack{j=1\\ j\neq i}}^{N+2} H(\mathbf{P}_t(\mathbf{z}_i^t), \mathbf{P}_s(\mathbf{z}_j^s)), \tag{4}$$

where $H(\mathbf{a}, \mathbf{b}) = -\mathbf{a}\log\mathbf{b}$.

C. Representation Distribution Learning

Different from DINO, we build another encoding-augmentation student branch. The core of the encoding-augmentation student branch is the representation distribution learning module. For convenience, we define representations extracted from different augmented views as real representations and those sampled from estimated distributions as virtual representations. To achieve the goal of exchanging the procedure of data augmentation and feature extraction, we first estimate a representation distribution for a patch and then sample representations from the estimated distribution for data augmentation.

1) Distribution Estimator: Previous studies suggest that the Gaussian prior suffices for representation augmentation by generating up-to-date and diverse representations even when the actual representations are not Gaussian [34]. In this work, we built an estimator based on the Gaussian prior to generate the patch representation distribution. Here, the estimated distribution is expected to continuously adapt to the representation change over the training procedure. Therefore, we applied trainable neural networks to fit the Gaussian distribution. It consists of a mean head ϕ_{θ_m} and a variance head ψ_{θ_v}, both composed of fully connected layers. Considering the non-negativity of the variance, we calculate its logarithm instead of calculating it directly. The corresponding representation mean $\boldsymbol{\mu} \in \mathbb{R}^D$ and standard deviation $\boldsymbol{\sigma} \in \mathbb{R}^D$ of each image that together define a Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2)$, which can be calculated by equations:

$$\boldsymbol{\mu} = \phi_{\theta_m}(f_{\theta_s}(\tilde{\mathbf{x}})), \tag{5}$$

$$\boldsymbol{\sigma} = \exp\left(\psi_{\theta_v}(f_{\theta_s}(\tilde{\mathbf{x}}))/2\right) \tag{6}$$

2) Representation Sampling: With the estimated distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2)$ for a patch, we can obtain variable representations of the patch by sampling process $\tilde{\mathbf{z}} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2)$, which is computationally efficient. Here, we adopt the reparameterization trick in VAE [53] to enable backpropagation for training. Specifically, we sample virtual representations $\tilde{\mathbf{z}}$ under the representation independence assumption:

$$\tilde{\mathbf{z}} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \in \mathcal{N}(0, \mathbf{I}_D), \tag{7}$$

where $\mathbf{I}_D$ denotes D-dimensional identity matrix.

Finally, in the encoding-augmentation student branch, we sample a set of virtual representations containing $\mathbb{Z}_v = \{\tilde{\mathbf{z}}_1, \tilde{\mathbf{z}}_2, \ldots, \tilde{\mathbf{z}}_K\}$ with K denoting the number of the sampled representations.

D. Objective and Optimization

Except for the foundational loss L_c, we encourage the probability distribution of the virtual representations $\mathbf{P}_v$ to be consistent with $\mathbf{P}_t$, enabling the virtual representations generated from the distribution estimator to cover the variants of the real representations. $\mathbf{P}_v$ are obtained in a manner akin to $\mathbf{P}_s$, utilizing the temperature parameter τ_v.

Therefore, we modify the objective function:

$$L_C = \sum_{i=1}^{2}\Big(\sum_{\substack{j=1 \\ j\neq i}}^{N+2} H(\mathbf{P}_t(\mathbf{z}_i^t), \mathbf{P}_s(\mathbf{z}_j^s)) + \sum_{k=1}^{K} H(\mathbf{P}_t(\mathbf{z}_i^t), \mathbf{P}_v(\tilde{\mathbf{z}}_k))\Big), \tag{8}$$

Moreover, we add a Kullback-Leibler (KL) divergence constraint to prevent overfitting of the distribution estimator, which is represented as

$$L_{KL} = D_{KL}(\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2), \mathcal{N}(0, \mathbf{I}_D)), \tag{9}$$

where $\mathcal{N}$ is the Gaussian distribution and D_{KL} is the K-L divergence. The final object function is composed as follows:

$$L_{total} = L_C + \beta L_{KL}, \tag{10}$$

where β controls the weight of KL loss. The parameters of the student network and the distribution estimator are optimized by the gradient descent algorithm, and the teacher network is updated by the exponential moving average (EMA) mechanism:

$$\theta_t \leftarrow \lambda\theta_t + (1-\lambda)\theta_s \tag{11}$$

with the update rule of λ following a cosine schedule from 0.996 to 1 during training.

E. WSI Analysis with Representation Augmentation

1) MIL Formulation: Given a WSI $\mathbf{X} \in \mathbb{R}^{W\times H\times 3}$ sized by $W \times H$ with label Y, it consists of patches $(\mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_n)$. Y is the bag-level label while the instance-level labels $(y_1, y_2, \ldots . y_n)$ are unknown. The majority of MIL can be summarized in two components: 1) Embedding patches into D-dimension representations $\mathbf{Z} = (\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_n)$ with patch-level representation encoder $f_{\theta_1} : \mathbf{x} \rightarrow \mathbb{R}^D$ parameterized by θ_1. 2) Integrating all patches' representations within $\mathbf{X}$ with WSI-level representation aggregator g_{θ_2} parameterized by θ_2. A common binary MIL classification method can be formulated as:

$$\hat{Y} = g_{\theta_2}(f_{\theta_1}(\mathbf{x}_1), f_{\theta_1}(\mathbf{x}_2), \ldots, f_{\theta_1}(\mathbf{x}_n)) \tag{12}$$

$$Y = \begin{cases} 0, \ if \sum y_i = 0 \quad i = 1, 2, \ldots, n \\ 1, \ otherwise \end{cases} \tag{13}$$

where $\hat{Y}$ is the WSI-level prediction. This formulation enables us to learn a WSI classifier with only WSI-level labels.

Limited by the computational cost and GPU memory, the entire MIL framework including f_{θ_1} and g_{θ_2} can hardly be trained in an end-to-end manner. Therefore, the parameters θ_1 and θ_2 are usually learned separately. The parameters θ_1 is typically pretrained on ImageNet or through representation learning methods. The parameters θ_2 is trained subsequent to the encoding process, where the patches are transformed into representations by the patch encoder using θ_1.

TABLE I
THE WSI DISTRIBUTION OF THE EXPERIMENTAL DATASETS.

USTC-EGFR	Normal	19del	L858R	Wild	Other
Train	101	69	118	85	84
Val	16	11	19	14	14
Test	48	38	47	47	43
TCGA-EGFR	Wild	Mutant			
Train	357	65			
Val	58	9			
Test	175	32			
TCGA-LUNG-3K	Normal	LUAD	LUSC		
Train	339	740	760		
Val	56	123	126		
Test	158	394	368		

TABLE II
COMPOSITION OF DIFFERENT AUGMENTATIONS.

Pre-Augmentation	RandomHorizontalFlip ColorJitter (0.4, 0.4, 0.2, 0.1) RandomGrayscale
Weak Image Augmentation	RandomResizedCrop (224) RandomHorizontalFlip ColorJitter (0.4, 0.4, 0.2, 0.1) GaussianBlur Solarization
Strong Image Augmentation	RandomResizedCrop (224) RandomResizedCrop (96) RandomHorizontalFlip ColorJitter (0.4, 0.4, 0.2, 0.1) GaussianBlur Solarization

2) Online Representation Augmentation: Through knowledge distillation in the patch-level representation learning, we achieved the procedure exchange of data augmentation and feature extraction. We adopt a representation augmentation strategy named online representation sampling (ORS) for WSI augmentation. As illustrated in Fig. 2e, patch-level representation distributions are obtained after patch encoding. During WSI classifier training, patch representations $\mathbf{Z}$ within $\mathbf{X}$ can be randomly replaced by virtual representations sampled from the corresponding distributions with a probability of p_{aug}, which can be formulated as:

$$\tilde{\mathbf{Z}} = (\tilde{\mathbf{z}}_1, \tilde{\mathbf{z}}_2, \ldots, \tilde{\mathbf{z}}_n), \quad \tilde{\mathbf{z}}_i \in \mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\sigma}_i^2). \tag{14}$$

Then, we feed $\tilde{\mathbf{Z}}$ instead of $\mathbf{Z}$ into the WSI model for training, which is represented as

$$\hat{Y} = g_{\theta_2}(\tilde{\mathbf{z}}_1, \tilde{\mathbf{z}}_2, \ldots, \tilde{\mathbf{z}}_n), \tag{15}$$

which is considered to be equivalent as the WSI-level data augmentation.

3) Inference: As in natural images, WSI augmentation is only used in the training phase to improve the generalization of the model. The augmentation-encoding student branch and the encoding-augmentation student branch share weights in patch-level representation learning, determining that WSI augmentation can be skipped in the inference phase. The original representations extracted from the patches are directly fed into the WSI classifier to obtain the prediction:

$$\hat{Y} = g_{\theta_2}(\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_n). \tag{16}$$

IV. EXPERIMENTS

A. Datasets

As shown in Table I, the proposed method was evaluated on a private dataset and two publicly available datasets, detailed as below:

USTC-EGFR is a private H&E-stained lung adenocarcinoma dataset that contains 754 WSIs for epidermal growth factor receptor (EGFR) gene mutation identification. These WSIs were captured using a Motic Easyscan Pro 6 under 20X lenses. The resolution is 0.52 μm/pixel. The dataset categorizes the WSIs into 5 classes, including EGFR 19del mutation (19del), EGFR L858R mutation (L858R), none common driver mutations (Wild), other driver gene mutation (Other), and cancer-free tissue (Normal).

TCGA-EGFR is a public lung adenocarcinoma dataset for EGFR mutation classification that contains 696 WSIs collected from the cancer genome atlas (TCGA) program of NCI (https://portal.gdc.cancer.gov). These WSIs are categorized into 2 classes, including wild type and mutant type.

TCGA-LUNG-3K is a public lung dataset that contains 3064 WSIs collected from the cancer genome atlas (TCGA) program of NCI (https://portal.gdc.cancer.gov). These WSIs are categorized into 3 classes, including lung adenocarcinoma (LUAD), lung squamous cell carcinoma (LUSC), and cancer-free tissue (Normal).

In each dataset, the WSIs were randomly split into train, validation and test subsets following the ratio of 6:1:3 at the patient-level. The train subset was used to train the models, and the validation subset was used to do the early stop and hyper-parameter tuning, and the final results were obtained within the test subset.

B. Experimental Settings

The WSIs were divided into non-overlapping patches in size of 224 × 224 for the representation learning and feature extraction by window sliding that is performed on the WSIs under 20× lenses. We followed the implementation in DINO [29] to use ViT-S/16 [24] as the backbone. We used the class token of ViT as the final feature embedding.

We evaluated our method under three WSI classification frameworks, CLAM [5], TransMIL [6] and DTFD-MIL [7]. These three methods all use attention mechanism to aggregate patch representations. CLAM focuses on several patches that most affect the diagnosis. TransMIL takes the spatial information of the patches within the WSI into consideration. DTFD-MIL utilizes pseudo-bags to increase the number of WSIs.

As shown in Table II, we used different data augmentation compositions in three different phases. Pre-augmentation does simple spatial transformations and color perturbations on the original image. Subsequent image augmentations add cropping and more complex transformations. The strong image augmentation contains the operations in the weak augmentation and adds local views cropped by small size.

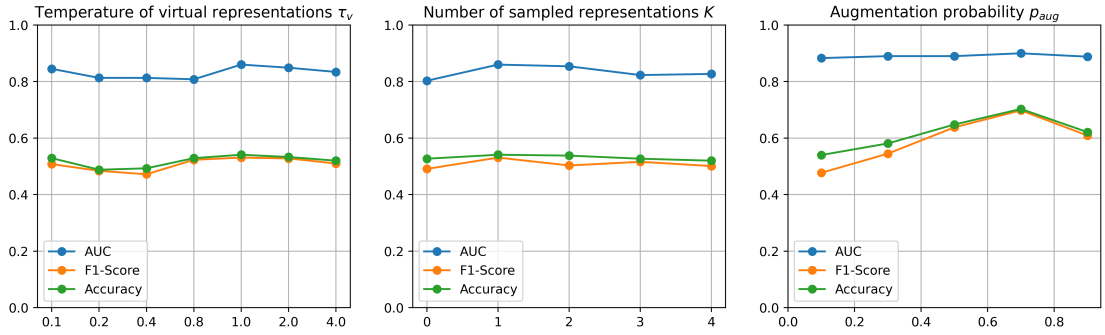


Fig. 3. Effects of the hyper-parameters on the USTC-EGFR validation subset under the CLAM framework.

TABLE III
ABLATION STUDY ON THE USTC-EGFR DATASET.

Methods	**CLAM** [5]			**TransMIL** [6]			**DTFD-MIL** [7]		
	AUC	F1-Score	ACC	AUC	F1-Score	ACC	AUC	F1-Score	ACC
SSRDL w/o RA	79.0	43.5	45.2	75.3	41.2	44.0	75.7	40.1	42.0
SSRDL w/o KL	82.5	51.5	51.1	78.1	41.7	43.2	79.9	46.2	47.0
SSRDL	85.0	57.5	58.6	80.1	41.8	44.9	81.9	**51.4**	**52.3**
SSRDL+ORS	**87.7**	**61.2**	**62.0**	**83.8**	**50.8**	**52.4**	**82.8**	49.9	51.3

For the binary classification task, average accuracy, area under the curve (AUC) and macro-average F1 score are calculated. For the sub-typing task, average accuracy, micro-average area under the curve, and macro-average F1 score are calculated. In the comparison study, we evaluated the uncertainty of the results by including the 2.5 and 97.5 percentile confidence intervals (CI), which are obtained from 1000 bootstrapping iterations.

These experiments were conducted using the original code released by the authors of the respective methods. Specifically, we implemented the methods ourselves for DAGAN [38] and AugDiff [37] for the absence of the released source code. All the methods were implemented in Python 3.8 with torch 1.8.1 and run on a computer cluster with 10 Xeon 2.66GHz CPUs and 10 Nvidia GeForce 2080Ti GPUs.

C. Hyper-parameter Verification

We verify the main factors that decide the capacity of the SSRDL with ORS, including 3 hyper-parameters (τ_v, K, p_{aug}). The results are given in Fig. 3. The first two results are obtained by SSRDL alone, and the last one is from the SSRDL with ORS. Micro-AUC, F1 score and accuracy of the validation subset are reported. The other hyper-parameters are set fixed when one hyper-parameter is being tuned.

1) The temperature of virtual representations: We control the sharpness of the virtual representations by tuning the softmax temperature parameter τ_v. We observe that the relatively good performances of the sharp targets can be achieved when the temperature is very small. The evaluation metrics decrease when the smooth targets are applied. Based on the observation, we set $\tau_v = 1$ in the following experiments.

2) The number of sampled representations: K decides the number of the representations sampled for representation learning. In this experiment, we tune $K \in \{1, 2, 3, 4\}$. $K = 0$ represents the results of SSRDL without representation augmentation. As the number of the sampled representations increases, the performance of the SSRDL decreases. This inspires us that K and the quality of the learned representations are not positively correlated. It is enough to sample just one representation for representation learning. Therefore, we set $K = 1$ in the training of the SSRDL.

3) The augmentation probability: p_{aug} determines the probability of applying the ORS strategy on the WSI. It affects how often the WSI augmentation is used during the training of the WSI classifier. The results show that p_{aug} has little impact on the Micro-AUC, but F1 score and accuracy are sensitive to the varying of the probability. Finally, we set $p_{aug} = 0.7$ for its best performance.

D. Ablation Study

The results of the ablation study are summarized in Table III. It is observed that there is a decrease of 6.0%, 4.8%, 6.2% in AUC; 14%, 0.6%, 11.3% in F1-Score; and 13.4%, 0.9% and 10.3% ACC under three WSI classification frameworks when comparing SSRDL w/o RA to SSRDL. It demonstrates that the RA module plays a crucial role in representation learning. Considering that the representation distribution is estimated based on a Gaussian prior, we constrain it with KL divergence. The results of SSRDL w/o KL show that KL divergence plays an essential role in the training of the distribution estimator.

Finally, we verify the impact of the ORS strategy on WSI classification. We observe that ORS can improve the performance of the WSI classifier even further building on SSRDL. Specifically, under the CLAM framework, the Micro-AUC increases from 85.0% to 87.7%. Under the TransMIL framework, the improvement is from 80.1% to 83.8%, and

TABLE IV
COMPARISONS WITH SOTA METHODS ON THE USTC-EGFR DATASET. † REPRESENTS THE RESULTS OBTAINED WITH THE ORIGINAL FEATURE LEARNING STRATEGY.

Methods	**CLAM** [5] AUC (2.5% CI, 97.5% CI)	**CLAM** [5] F1-Score (2.5% CI, 97.5% CI)	**CLAM** [5] ACC (2.5% CI, 97.5% CI)	**TransMIL** [6] AUC (2.5% CI, 97.5% CI)	**TransMIL** [6] F1-Score (2.5% CI, 97.5% CI)	**TransMIL** [6] ACC (2.5% CI, 97.5% CI)	**DTFD-MIL** [7] AUC (2.5% CI, 97.5% CI)	**DTFD-MIL** [7] F1-Score (2.5% CI, 97.5% CI)	**DTFD-MIL** [7] ACC (2.5% CI, 97.5% CI)
ImageNet [25]	77.8 (73.2, 82.3)	38.5 (29.9, 47.0)	43.3 (35.0, 50.0)	73.2 (68.5, 77.6)	29.4 (22.0, 37.3)	32.8 (25.0, 41.0)	73.1 (67.6, 78.5)	39.0 (30.0, 48.0)	40.1 (31.3, 49.5)
SimCLR [26]	69.2 (62.6, 75.1)	37.9 (28.5, 46.8)	39.4 (30.0, 48.0)	62.2 (56.5, 67.6)	22.2 (16.8, 27.7)	29.5 (23.0, 36.0)	70.3 (64.0, 76.3)	37.2 (28.5, 46.5)	40.2 (31.3, 48.5)
BYOL [27]	61.1 (54.5, 67.6)	24.7 (18.3, 31.5)	29.9 (22.0, 38.0)	54.5 (48.0, 61.2)	20.4 (13.8, 27.1)	25.4 (19.0, 33.0)	61.8 (55.2, 67.9)	28.5 (21.4, 36.0)	33.9 (26.3, 41.4)
MoCov3 [54]	77.3 (71.5, 82.6)	40.4 (31.6, 49.6)	43.2 (35.0, 51.0)	63.5 (58.2, 68.7)	22.9 (16.5, 29.6)	29.3 (23.0, 35.0)	74.8 (69.0, 80.2)	32.9 (25.3, 41.0)	39.3 (31.3, 47.5)
DINO [29]	79.0 (74.6, 83.1)	43.5 (34.1, 53.1)	45.2 (36.4, 53.5)	75.3 (69.4, 81.1)	41.2 (32.1, 50.6)	44.0 (35.4, 52.5)	75.7 (70.2, 80.7)	40.1 (29.8, 49.4)	42.0 (32.3, 50.5)
DINO+Random Perturbation	74.9 (70.2, 80.3)	34.9 (27.3, 42.2)	40.7 (33.3, 48.5)	74.0 (68.2, 79.8)	39.0 (30.0, 47.9)	39.7 (31.0, 49.0)	76.1 (70.7, 81.2)	43.4 (34.4, 52.5)	44.9 (36.4, 53.5)
DINO+MC Sampling [12]	75.9 (70.4, 81.5)	37.0 (28.2, 46.0)	39.5 (30.3, 48.5)	74.9 (69.6, 80.3)	35.9 (27.6, 44.0)	37.1 (29.0, 46.0)	75.9 (70.2, 81.0)	42.0 (32.8, 51.2)	43.8 (35.4, 52.5)
† ReMix [13] *(SimCLR+ResNet18)*	69.3 (63.4, 75.4)	24.8 (19.2, 31.5)	35.8 (30.3, 42.4)	64.0 (58.5, 69.1)	15.0 (10.3, 19.5)	25.6 (20.2, 31.3)	64.4 (58.3, 70.6)	22.4 (15.5, 30.1)	25.1 (18.2, 33.3)
† RankMix [39] *(SimCLR+ResNet18)*	64.0 (57.7, 70.4)	26.4 (19.5, 33.8)	30.9 (23.2, 38.4)	62.9 (57.0, 68.2)	18.8 (15.3, 22.1)	32.2 (26.3, 37.4)	64.9 (59.5, 70.4)	26.3 (19.2, 34.3)	33.3 (25.3, 41.4)
† Intra-Mixup [40] *(ImageNet+ResNet18)*	82.3 (77.7, 86.3)	32.1 (26.5, 37.7)	43.1 (37.4, 49.5)	80.4 (75.5, 85.3)	46.5 (37.3, 55.8)	47.1 (37.4, 56.6)	73.6 (67.9, 78.8)	14.7 (9.2, 21.1)	23.7 (19.2, 28.3)
† DAGAN [38] *(ImageNet+ResNet50)*	80.3 (75.2, 84.8)	43.9 (34.7, 52.7)	46.4 (38.4, 54.6)	67.5 (62.0, 72.3)	21.9 (17.4, 26.8)	33.5 (27.3, 39.4)	76.5 (70.8, 81.9)	41.9 (33.7, 51.1)	44.7 (37.4, 53.5)
† AugDiff [37] *(ImageNet+ResNet18)*	79.3 (74.1, 83.9)	37.3 (30.4, 44.7)	45.3 (37.4, 52.5)	78.7 (74.0, 83.2)	40.5 (32.9, 48.7)	47.2 (40.4, 54.6)	78.1 (72.9, 82.6)	39.9 (31.2, 48.8)	40.7 (31.3, 49.5)
DINO+ReMix [13]	78.0 (72.7, 83.1)	36.5 (28.5, 44.7)	42.7 (36.0, 50.0)	75.4 (70.1, 80.4)	36.4 (28.1, 45.0)	40.3 (32.0, 48.0)	73.7 (68.3, 79.1)	36.7 (28.1, 45.3)	41.0 (32.3, 49.5)
DINO+RankMix [39]	73.2 (68.8, 77.7)	29.3 (22.6, 36.5)	31.4 (24.0, 39.0)	75.2 (69.7, 80.4)	35.8 (28.3, 43.2)	41.3 (33.0, 49.0)	72.0 (65.8, 77.8)	33.5 (25.8, 41.3)	38.2 (29.3, 46.5)
DINO+Intra-Mixup [40]	79.1 (74.1, 83.8)	38.5 (29.4, 48.1)	40.7 (32.0, 49.0)	78.9 (74.2, 83.3)	46.0 (36.9, 55.3)	46.2 (37.4, 55.6)	76.6 (71.1, 81.8)	41.2 (32.3, 50.4)	42.5 (33.3, 51.5)
DINO+DAGAN [38]	78.8 (73.7, 83.6)	34.9 (29.3, 39.9)	45.8 (39.4, 51.5)	78.8 (73.9, 83.2)	42.3 (33.5, 50.9)	44.1 (36.0, 53.0)	76.5 (71.2, 81.6)	37.1 (28.5, 45.9)	38.6 (30.3, 47.5)
DINO+AugDiff [37]	81.7 (76.8, 86.4)	48.8 (39.3, 58.2)	52.3 (43.4, 60.6)	77.1 (71.5, 82.3)	35.2 (27.4, 43.7)	39.8 (32.3, 47.5)	75.5 (69.6, 80.8)	42.5 (33.7, 51.2)	43.8 (35.4, 52.5)
SSRDL	85.0 (80.0, 89.3)	57.5 (47.8, 67.1)	58.6 (49.0, 68.0)	80.1 (75.6, 84.6)	41.8 (33.0, 50.8)	44.9 (37.0, 53.0)	81.9 (77.0, 86.6)	**51.4** (42.0, 61.1)	**52.3** (42.4, 61.6)
SSRDL+ORS	**87.7** (83.5, 91.5)	**61.2** (52.0, 70.2)	**62.0** (53.0, 71.0)	**83.8** (80.0, 87.8)	**50.8** (42.5, 59.5)	**52.4** (44.0, 61.0)	**82.8** (78.3, 87.1)	49.9 (40.2, 59.7)	51.3 (41.4, 60.6)

under the DTFD-MIL framework, the Micro-AUC rises from 81.9% to 82.8%. This suggests that the representations generated by ORS contain useful semantic information and are effective to the model.

E. Comparison with Other Methods

We compared our method with 5 different WSI representing strategies, including ImageNet [25], SimCLR [26], BYOL [27], MoCov3 [54] and DINO [29]. The first three use ResNet50 [23] as the backbone, while the others utilize ViT-S [24]. Moreover, we compared our method with 7 WSI data augmentation frameworks, including Random Perturbation, Monte Carlo Sampling [12], ReMix [13], RankMix [39], Intra-Mixup [40], DAGAN [38], AugDiff [37]. The compared methods were also evaluated using the representations obtained by DINO, following the same experimental setup as our framework, to mitigate any potential impact caused by different feature learning strategies.

Overall, our method achieves the best performance on the **USTC-EGFR** dataset, **TCGA-EGFR** dataset and **TCGA-LUNG-3K** dataset under CLAM [5], TransMIL [6] and DTFD-MIL [7] frameworks.

1) Compare with representation learning methods: ImageNet [25] denotes extracting the patch representations using the ResNet50 pre-trained on the ImageNet dataset. The experimental results show that it sometimes performs better than self-supervised methods, which demonstrates that the representations extracted using this strategy can also achieve better results in the case of an insufficient amount of WSIs. SimCLR [26] is the contrastive learning framework widely applied in feature extraction. The results on the TCGA-EGFR dataset show that it achieves 11.7%, 2.1%, 4.9% AUC better than ImageNet under three MIL frameworks due to the semantic gap between natural and pathological images. However, on the USTC-EGFR dataset and TCGA-LUNG-3K, SimCLR and BYOL perform worse than ImageNet, indicating that self-supervised learning may not yield good performance

TABLE V
COMPARISONS WITH SOTA METHODS ON THE TCGA-EGFR DATASET. † REPRESENTS THE RESULTS OBTAINED WITH THE ORIGINAL FEATURE LEARNING STRATEGY.

Methods	**CLAM** [5] AUC (2.5% CI, 97.5% CI)	**CLAM** [5] F1-Score (2.5% CI, 97.5% CI)	**CLAM** [5] ACC (2.5% CI, 97.5% CI)	**TransMIL** [6] AUC (2.5% CI, 97.5% CI)	**TransMIL** [6] F1-Score (2.5% CI, 97.5% CI)	**TransMIL** [6] ACC (2.5% CI, 97.5% CI)	**DTFD-MIL** [7] AUC (2.5% CI, 97.5% CI)	**DTFD-MIL** [7] F1-Score (2.5% CI, 97.5% CI)	**DTFD-MIL** [7] ACC (2.5% CI, 97.5% CI)
ImageNet [25]	64.1 (36.4, 87.9)	61.3 (42.6, 77.1)	77.1 (64.1, 87.2)	65.2 (41.4, 85.9)	54.5 (40.0, 73.5)	75.6 (64.1, 87.2)	73.3 (57.6, 86.4)	61.7 (50.5, 72.9)	75.3 (66.7, 83.8)
SimCLR [26]	75.8 (55.0, 91.9)	61.3 (45.8, 76.9)	73.7 (59.0, 87.2)	67.3 (42.4, 89.4)	59.7 (43.0, 73.9)	70.3 (53.9, 84.6)	78.2 (65.4, 88.5)	60.2 (49.7, 71.6)	75.8 (67.7, 83.8)
BYOL [27]	61.0 (38.9, 82.8)	50.9 (40.9, 70.5)	78.2 (69.2, 87.2)	63.6 (40.4, 84.9)	55.7 (39.5, 73.7)	69.9 (56.4, 84.6)	58.7 (43.5, 75.1)	55.4 (44.1, 67.5)	76.9 (69.7, 83.8)
MoCov3 [54]	48.4 (25.3, 73.2)	45.8 (36.1, 60.6)	68.2 (53.9, 79.5)	58.6 (34.8, 80.8)	48.9 (37.1, 67.7)	70.4 (56.4, 82.1)	63.0 (49.4, 76.2)	55.5 (45.0, 67.5)	79.0 (71.7, 84.9)
DINO [29]	62.9 (37.9, 85.4)	54.1 (40.0, 71.1)	75.3 (61.5, 87.2)	60.9 (35.9, 83.4)	50.8 (44.3, 72.1)	84.6 (79.5, 89.7)	59.2 (42.9, 75.0)	52.0 (42.1, 63.6)	75.8 (68.7, 82.8)
DINO+Random Perturbation	61.7 (38.9, 84.3)	53.2 (38.1, 70.7)	64.5 (48.7, 79.5)	63.2 (35.3, 84.9)	57.4 (40.9, 73.7)	71.2 (56.4, 84.6)	67.1 (52.9, 80.2)	53.7 (44.7, 62.6)	60.8 (51.5, 69.7)
DINO+MC Sampling [12]	70.4 (48.5, 89.4)	59.8 (41.8, 77.1)	77.5 (66.7, 87.2)	67.3 (42.4, 88.9)	60.3 (41.8, 79.4)	76.8 (64.1, 87.2)	68.7 (52.9, 82.3)	57.3 (46.4, 70.3)	81.2 (74.8, 86.9)
† ReMix [13] *(SimCLR+ResNet18)*	66.9 (46.0, 85.9)	58.4 (40.9, 76.9)	75.8 (64.1, 87.2)	53.5 (29.3, 78.3)	45.8 (45.8, 45.8)	84.6 (84.6, 84.6)	62.2 (49.1, 74.5)	52.7 (43.3, 61.8)	61.7 (51.5, 70.7)
† RankMix [39] *(SimCLR+ResNet18)*	63.5 (41.4, 83.3)	45.4 (44.3, 45.8)	83.1 (79.5, 84.6)	65.2 (39.9, 85.9)	44.7 (42.7, 45.8)	80.8 (74.4, 84.6)	63.3 (48.7, 78.6)	45.8 (45.8, 45.8)	84.6 (84.6, 84.6)
† Intra-Mixup [40] *(ImageNet+ResNet18)*	69.3 (45.5, 89.9)	44.9 (30.8, 57.9)	47.9 (33.3, 64.1)	74.1 (52.0, 91.4)	63.1 (47.7, 79.4)	76.4 (64.1, 87.2)	67.9 (53.4, 80.6)	52.4 (42.4, 64.6)	78.4 (71.7, 83.8)
† DAGAN [38] *(ImageNet+ResNet50)*	64.7 (36.4, 87.9)	59.5 (42.6, 76.9)	74.2 (61.5, 87.2)	65.9 (38.4, 88.4)	57.1 (42.7, 80.3)	83.1 (74.4, 89.7)	78.0 (63.7, 89.4)	57.0 (48.5, 65.1)	63.9 (54.6, 72.7)
† AugDiff [37] *(ImageNet+ResNet18)*	79.0 (58.1, 96.0)	65.1 (49.4, 81.2)	76.2 (61.5, 89.7)	70.7 (44.4, 93.4)	65.4 (45.0, 84.1)	80.8 (69.2, 92.3)	64.6 (49.3, 80.5)	50.8 (44.7, 62.2)	83.9 (80.8, 86.9)
DINO+ReMix [13]	71.5 (48.5, 89.9)	47.1 (42.7, 60.8)	81.6 (74.4, 87.2)	61.7 (38.9, 81.3)	45.8 (45.8, 45.8)	84.6 (84.6, 84.6)	58.5 (43.1, 72.7)	50.5 (41.8, 60.9)	75.2 (68.7, 81.8)
DINO+RankMix [39]	71.4 (49.5, 91.4)	58.4 (42.7, 80.3)	82.5 (74.4, 89.7)	70.9 (46.5, 92.9)	50.3 (44.3, 72.1)	84.1 (79.5, 89.7)	63.7 (48.9, 78.4)	45.8 (45.8, 45.8)	84.6 (84.6, 84.6)
DINO+Intra-Mixup [40]	74.1 (48.0, 92.9)	57.8 (42.7, 77.1)	81.7 (71.8, 89.7)	63.4 (36.4, 86.4)	57.9 (42.7, 80.3)	81.7 (71.8, 89.7)	69.4 (53.5, 84.1)	59.0 (48.9, 69.8)	74.4 (66.7, 81.8)
DINO+DAGAN [38]	72.8 (49.5, 91.9)	59.9 (44.6, 75.7)	70.1 (56.4, 84.6)	64.1 (40.4, 84.9)	55.4 (39.8, 71.1)	67.6 (53.9, 82.1)	61.6 (43.6, 77.6)	55.6 (44.5, 67.4)	75.8 (68.7, 83.8)
DINO+AugDiff [37]	67.6 (43.4, 90.4)	60.6 (42.7, 80.3)	77.3 (64.1, 89.7)	67.8 (43.9, 88.4)	54.5 (40.9, 73.9)	78.8 (69.2, 87.2)	73.4 (59.6, 86.2)	61.8 (49.5, 73.9)	80.8 (73.7, 86.9)
SSRDL	78.5 (56.6, 95.0)	64.4 (43.5, 84.1)	**83.6** (74.4, 92.3)	77.9 (57.6, 95.5)	63.6 (44.3, 84.1)	85.5 (76.9, 92.3)	79.2 (67.5, 89.8)	65.5 (54.6, 76.4)	78.7 (70.7, 85.9)
SSRDL+ORS	**83.1** (62.6, 97.0)	**69.9** (50.8, 87.6)	82.7 (69.2, 92.3)	**79.7** (57.1, 96.5)	**69.7** (50.8, 88.5)	**85.7** (76.9, 94.9)	**85.3** (73.6, 93.8)	**72.5** (60.1, 83.6)	**85.0** (77.8, 91.9)

when data is scarce and the data augmentation strategies employed in these frameworks are limited. MoCov3 [54] and DINO [29] take the ViT as the backbone, making them perform better than SimCLR [26] and BYOL [27] on the USTC-EGFR and TCGA-LUNG-3K datasets. However, the performances of MoCov3 and DINO on the TCGA-EGFR dataset have gaps to the top-tier results, which could be caused by data imbalance. This indicates that ViT-based models may require more balanced data distributions or enhanced training strategies to effectively handle dataset characteristics. DINO, which adopts a multi-crop augmentation strategy, performs better compared to other representation learning methods on the USTC-EGFR and TCGA-LUNG-3K datasets. This suggests that the importance of data augmentations in generating discriminative representations for WSI classification.

In comparison, the proposed SSRDL additionally takes representation augmentation into the process of self-supervised learning, which improves the performance by increasing the diversity of representations. Compared with our baseline model DINO, SSRDL achieves increase in AUC of 6.0%, 15.6%, 1.0% under the CLAM [5] framework, 4.8%, 17.0%, 1.9% under the TransMIL [6] framework and 6.2%, 20.0%, 2.2% under the DTFD-MIL [7] framework on the three datasets, respectively. Meanwhile, the distribution estimator trained in representation learning can be utilized in the downstream task.

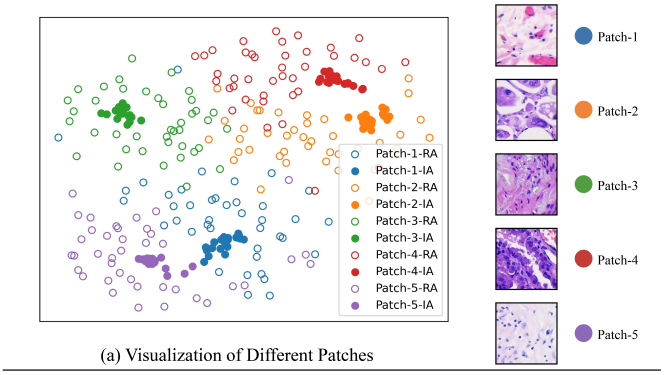
2) Compare with data augmentation methods: To carry out the WSI augmentation in feature space, we first explore whether random perturbation on representations can get an improvement. Random perturbation involves the addition of random Gaussian Noise to the patch representations. The results show that it often leads to performance degradation rather than improvement. Moreover, we utilize Monte Carlo Sampling [12] by randomly discarding some patches within a WSI. The performance on the TCGA-EGFR dataset is increased in AUC by 7.5%, 6.4%, 9.5%, which indicates that it works in some cases. However, this strategy has a high risk of losing important information for classification, thus leading

TABLE VI
COMPARISONS WITH SOTA METHODS ON THE TCGA-LUNG-3K DATASET. † REPRESENTS THE RESULTS OBTAINED WITH THE ORIGINAL FEATURE LEARNING STRATEGY.

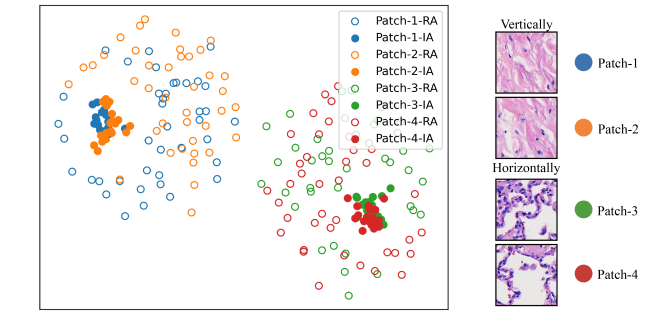
Methods	**CLAM** [5] AUC (2.5% CI, 97.5% CI)	**CLAM** [5] F1-Score (2.5% CI, 97.5% CI)	**CLAM** [5] ACC (2.5% CI, 97.5% CI)	**TransMIL** [6] AUC (2.5% CI, 97.5% CI)	**TransMIL** [6] F1-Score (2.5% CI, 97.5% CI)	**TransMIL** [6] ACC (2.5% CI, 97.5% CI)	**DTFD-MIL** [7] AUC (2.5% CI, 97.5% CI)	**DTFD-MIL** [7] F1-Score (2.5% CI, 97.5% CI)	**DTFD-MIL** [7] ACC (2.5% CI, 97.5% CI)
ImageNet [25]	94.6 (91.1, 97.6)	84.2 (76.7, 90.7)	82.7 (74.8, 89.9)	92.0 (87.5, 96.3)	80.4 (72.6, 88.1)	78.3 (69.7, 85.9)	93.7 (89.9, 97.2)	83.2 (75.5, 89.6)	81.3 (72.7, 88.9)
SimCLR [26]	93.5 (89.9, 96.6)	80.2 (72.3, 87.5)	78.7 (70.7, 86.9)	91.8 (87.2, 95.9)	81.2 (73.2, 88.5)	79.1 (70.7, 86.9)	92.8 (88.5, 96.5)	82.3 (74.7, 89.4)	80.8 (72.7, 88.9)
BYOL [27]	92.6 (88.6, 96.2)	78.5 (70.4, 86.0)	77.0 (68.7, 84.9)	89.6 (84.8, 94.1)	75.9 (67.5, 83.6)	74.6 (65.7, 82.8)	91.0 (86.1, 95.4)	77.9 (69.3, 86.1)	76.3 (67.7, 84.9)
MoCov3 [54]	94.0 (90.3, 97.1)	82.6 (75.0, 89.5)	81.5 (73.7, 88.9)	93.2 (89.0, 96.6)	79.7 (72.4, 87.0)	77.8 (69.7, 85.9)	93.9 (90.1, 97.1)	82.6 (74.9, 89.4)	81.0 (72.7, 88.9)
DINO [29]	96.6 (93.6, 98.9)	87.1 (80.3, 93.4)	85.7 (78.8, 91.9)	94.8 (91.0, 97.9)	85.5 (79.1, 91.2)	83.6 (76.8, 89.9)	95.7 (92.6, 98.4)	87.9 (81.5, 93.7)	86.2 (78.8, 92.9)
DINO+Random Perturbation	96.4 (93.3, 98.7)	86.8 (80.1, 92.9)	85.3 (78.8, 91.9)	95.0 (91.5, 97.7)	84.9 (78.2, 91.3)	83.2 (75.8, 89.9)	96.1 (92.8, 98.8)	87.0 (80.3, 92.9)	85.2 (77.8, 91.9)
DINO+MC Sampling [12]	96.1 (93.0, 98.5)	87.3 (80.7, 92.9)	85.7 (78.8, 91.9)	95.2 (91.5, 98.1)	84.1 (77.2, 90.7)	83.1 (75.8, 89.9)	96.4 (93.2, 98.8)	87.9 (81.4, 93.7)	86.6 (79.8, 92.9)
† ReMix [13] *(SimCLR+ResNet18)*	91.0 (86.3, 94.9)	79.0 (71.0, 86.5)	77.2 (68.7, 84.9)	90.1 (67.4, 78.8)	78.2 (38.8, 56.2)	76.4 (39.4, 55.6)	91.1 (86.5, 95.2)	80.0 (72.2, 87.0)	77.6 (69.7, 84.9)
† RankMix [39] *(SimCLR+ResNet18)*	91.8 (87.6, 95.7)	80.0 (71.9, 87.7)	78.3 (69.7, 86.9)	91.5 (86.7, 95.4)	79.3 (71.3, 87.3)	77.9 (69.7, 85.9)	92.0 (87.8, 95.8)	80.1 (72.2, 86.9)	78.6 (70.7, 85.9)
† Intra-Mixup [40] *(ImageNet+ResNet18)*	93.4 (89.4, 96.9)	83.3 (75.9, 90.1)	81.9 (73.7, 88.9)	92.9 (88.7, 96.5)	80.7 (73.1, 87.6)	79.0 (71.7, 85.9)	94.3 (90.6, 97.2)	81.8 (74.3, 89.0)	80.9 (72.7, 87.9)
† DAGAN [38] *(ImageNet+ResNet50)*	92.7 (88.5, 96.2)	75.9 (68.3, 83.9)	75.2 (67.7, 82.8)	91.8 (87.3, 96.0)	80.2 (72.0, 88.0)	78.9 (70.7, 86.9)	93.4 (89.4, 96.9)	82.4 (74.3, 89.6)	80.6 (72.7, 87.9)
† AugDiff [37] *(ImageNet+ResNet18)*	92.2 (88.0, 96.0)	78.0 (70.5, 85.5)	77.0 (69.7, 84.9)	91.3 (86.5, 95.4)	78.7 (70.4, 86.5)	76.8 (68.7, 84.9)	92.1 (87.7, 95.8)	79.9 (71.7, 87.4)	78.8 (70.7, 86.9)
DINO+ReMix [13]	95.4 (92.2, 98.0)	83.8 (76.6, 90.4)	82.7 (75.8, 89.9)	93.7 (89.7, 97.1)	83.0 (75.7, 89.6)	81.4 (74.7, 87.9)	96.1 (93.0, 98.5)	84.0 (76.9, 90.2)	81.4 (73.7, 87.9)
DINO+RankMix [39]	94.0 (90.4, 97.0)	81.5 (74.4, 87.7)	79.4 (71.7, 85.9)	93.4 (89.8, 96.6)	75.3 (67.9, 82.3)	73.1 (65.7, 79.8)	94.2 (91.1, 97.0)	75.9 (68.2, 82.8)	73.8 (66.7, 80.8)
DINO+Intra-Mixup [40]	94.7 (91.4, 97.5)	81.1 (73.7, 88.6)	80.7 (73.7, 88.9)	96.0 (93.1, 98.5)	87.8 (81.3, 93.5)	86.2 (78.8, 92.9)	96.6 (93.6, 98.8)	88.8 (82.6, 94.3)	87.0 (79.8, 92.9)
DINO+DAGAN [38]	95.1 (91.9, 97.8)	84.0 (76.8, 90.8)	83.3 (75.8, 90.9)	94.7 (91.1, 97.8)	85.4 (78.3, 91.9)	83.6 (75.8, 90.9)	95.6 (92.3, 98.3)	86.3 (79.7, 92.7)	84.6 (77.8, 91.9)
DINO+AugDiff [37]	95.9 (92.8, 98.4)	86.4 (79.5, 92.4)	85.1 (77.8, 91.9)	94.5 (91.1, 97.5)	84.2 (77.2, 90.5)	82.7 (75.8, 89.9)	95.8 (92.6, 98.5)	87.4 (80.9, 93.5)	85.8 (78.8, 92.9)
SSRDL	97.6 (94.9, 99.3)	**90.5** (84.5, 95.9)	**89.4** (82.8, 95.0)	96.7 (94.0, 99.0)	89.2 (83.0, 94.9)	87.8 (80.8, 93.9)	97.9 (95.7, 99.4)	91.5 (86.1, 96.8)	**90.4** (84.9, 96.0)
SSRDL+ORS	**97.6** (95.2, 99.3)	90.4 (84.4, 95.4)	89.2 (82.8, 95.0)	**97.2** (94.6, 99.2)	**90.4** (84.6, 95.4)	**89.3** (82.8, 95.0)	**98.1** (95.8, 99.6)	**91.6** (86.1, 96.7)	90.3 (83.8, 96.0)

to degradation of performance on the USTC-EGFR dataset. The results of these two strategies show that the impact of purposeless perturbation on the patch representations or the patch bag is unstable.

The methods based on Mixup produce different representations by mixing in several levels. ReMix [13] proposes to mix the prototypes within the same class obtained by K-means. The performance in AUC on the TCGA-EGFR dataset is increased by 8.6% under the CLAM framework. The results on the USTC-EGFR dataset show that this clustering based method may lead to a decline in performance due to the loss of local semantic information. RankMix [39] mixes both image and label with different WSIs. While this approach has led to improved performances on the TCGA-EGFR dataset, it has resulted in decreased performances on the USTC-EGFR and TCGA-LUNG-3K datasets. This may result from the disadvantage of Mixup that it cannot guarantee the semantic information after mixing. Intra-Mixup [40] limits mixing to a single WSI, making the mixed samples more reliable. It improves the performance in AUC on the TCGA-EGFR dataset by 11.2%, 2.5%, 10.2% under three different frameworks compared to DINO. But it can also result in the destruction of the original semantic information, thus leading to the decline on the USTC-EGFR dataset under the CLAM framework. All three methods have certain degrees of performance degradation on the TCGA-LUNG-3K dataset. These results indicate that Mixup-based methods cannot guarantee the individual semantic information after mixing and introduce noisy samples to training, thus leading to the performance decline. DAGAN [38] additionally builds a GAN [55] model for generating new representations with another training process. It improves performance in AUC by 9.9%, 3.2%, 2.4% on the TCGA-EGFR dataset. However, the generative model is trained after representation learning so that the performance is limited by the quality of the representations. The performance in F1-Score on the TCGA-LUNG-3K dataset is decreased by 3.1%, 0.1%, 1.6% under three different frameworks. Moreover, DAGAN introduces extra computation costs in both training



(a) Visualization of Different Patches



(b) Visualization of Neighbouring Patches

Fig. 4. Visualization of t-sne, where the hollow circles represent the virtual representations obtained by the proposed representation augmentation (RA) branch while the solid circles represent the real representations from the image augmentation (IA) branch, (a) displays patch representations from different categories of WSIs, (b) shows representations for neighbouring patches extracted from one WSI. The term "vertically" refers to patches that are adjacent in the vertical direction, while "horizontally" refers to patches that are adjacent in the horizontal direction.

and inference phases. AugDiff [37] utilizes a diffusion model to generate new representations. It achieves improvements with 2.7% increase in AUC, 5.3% in F1-Score, and 7.1% in ACC compared to the baseline DINO under the CLAM benchmark on the USTC-EGFR dataset. However, the use of a diffusion model increases both GPU memory consumption and the computational time required.

Our proposed SSRDL with ORS samples representations from the distributions estimated for the patches, which achieves reliable data augmentation in WSI classification and meanwhile ensures efficiency. This can also be viewed as a reasonable perturbation to representations, breaking the invariance of representations during training. The noise generated during sampling is much less than Mixup. Moreover, the distribution estimator is concurrently trained with representation learning, avoiding additional training overhead and continuously adapting to the representation change over the training procedure. With ORS, the performances on three datasets are increased in AUC of 6.0%, 9.0%, 1.7% under the CLAM [5] framework, 4.9%, 8.8%, 1.2% under the TransMIL [6] framework, 6.2%, 11.9%, 1.5% under the DTFD-MIL [7] framework compared to the second-best augmentation methods.

F. Integration with WSI-level Pretraining

WSI-level pretraining methods can be viewed as downstream tasks of the proposed method. We recognize the potential to enhance WSI classification through integrating WSI-level pretraining with our method. We have evaluated the effectiveness of our method to PAMA [49]. As shown in Table VII, these experiments were carried out using two different training strategies. The first strategy, linear probing, involves freezing the encoder while only training the classifier. This approach resulted in improvements in AUC across all three datasets, with increases of 1.2%, 9.1%, and 9.1%. The second strategy, fine-tuning, involves training all network parameters including the WSI encoder and the classifier, which also led to improvements in AUC of 5.3%, 12.2%, and 1.4%

The results demonstrate that integrating PAMA with our proposed method enhances performance across various datasets. This success supports the effectiveness of combining hierarchical pretraining with our advanced representation distribution learning framework.

G. Visualization

The dependability of the sampled representations is worth paying attention to. We conduct visualization experiments on the patches from the USTC-EGFR dataset. Fig. 4 and Fig. 5 show the t-sne [56] visualization of the representations extracted from the patches.

From Fig. 4a, we observe that the virtual representations have a wider range than the real representations but are still linearly separable. This provides a basis for the improvement of representation quality and classification performance. The representations shown in Fig. 4b are from the patches within one WSI. Two horizontally adjacent patches and two vertically adjacent patches are picked, respectively. We find that the neighboring patches are similar in both virtual and real representations, indicating that the representations obtained by ORS can maintain the original semantic relationship.

Fig. 5 shows more representations from different patches. It shows that when patches have minimal background areas, such as Patch-2, or when image augmentations are weak, as observed with Patch-8, the representations of different augmented views often gather together, leading to a lack of diversity among the augmented patch representations. Conversely, the representations sampled from distributions vary widely but remain close to the original patch representations. It demonstrates that the representations generated by ORS can cover the range of the representations extracted from image augmentations. This decides the sampled representations for MIL model training are diverse while reliable.

V. Discussion

In the field of natural images, data augmentation typically occurs before feature extraction, allowing for a diverse range of transformations directly on the raw images. However, the situation is different for WSIs due to their gigapixel properties. WSIs are generally divided into smaller patches, from which representations are extracted prior to the training of WSI-level models. This preprocessing requirement means that once

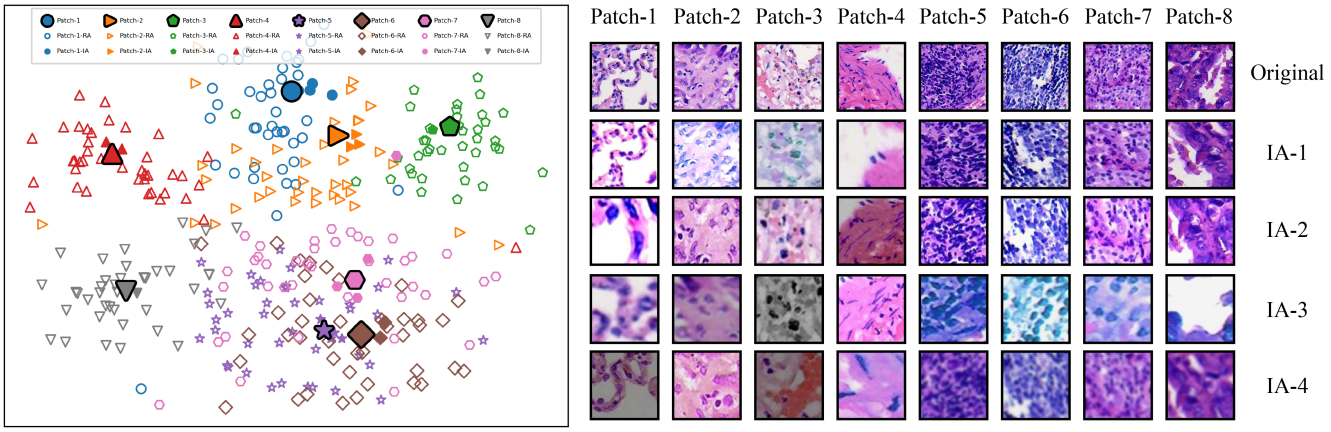


Fig. 5. Visualization of t-sne, where the ones with black outlines are original patch representations, the hollow ones are the virtual representations obtained by the proposed representation augmentation (RA) branch, and the solid ones are the real representations from the image augmentation (IA) branch.

TABLE VII
COMPARISONS UNDER THE PAMA FRAMEWORK.

Methods	Training Strategy	USTC-EGFR			TCGA-EGFR			TCGA-LUNG-3K		
		AUC	F1-Score	ACC	AUC	F1-Score	ACC	AUC	F1-Score	ACC
DINO+PAMA	linear probing	67.1	30.1	34.5	52.8	47.9	58.5	82.8	57.2	57.3
SSRDL+ORS+PAMA	linear probing	**68.3**	**31.8**	**35.9**	**61.9**	**54.8**	**70.5**	**91.9**	**80.1**	**78.9**
DINO+PAMA	fine-tuning	79.8	45.0	46.2	70.7	58.0	70.0	96.6	89.1	87.8
SSRDL+ORS+PAMA	fine-tuning	**85.1**	**53.1**	**54.3**	**82.9**	**68.1**	**85.5**	**98.0**	**91.4**	**90.3**

the patch representations are extracted from a WSI, they are fixed. As a result, common image augmentation methods that modify pixel values or apply transformations like rotation and scaling cannot be directly applied to these pre-extracted patch representations. The corresponding solution to this issue is to randomly perturbate the feature values. However, our experiments have shown that this method often leads to a decrease in performance rather than an improvement.

In our proposed method, feature extraction and augmentation are performed simultaneously. The proposed method encodes the patches into representation distributions, which integrates feature extraction with data augmentation. These augmentations are then utilized during the training of the WSI model, which takes place after feature extraction.

Regarding the calculation of P_v, it is the result of applying the softmax function to the virtual representations $\tilde{\mathbf{z}}_k$. This function maps the Gaussian distribution of $\tilde{\mathbf{z}}_k$ onto a different space where the outputs are probabilities that sum to one. Due to this transformation, P_v does not maintain the Gaussian distribution properties of $\tilde{\mathbf{z}}_k$. Consequently, P_v is not calculated directly from μ and σ.

The proposed SSRDL framework has a computational complexity of 42.02 GFLOPs, which is only marginally (8%) higher than the 38.78 GFLOPs of the standard DINO framework. Meanwhile, our augmentation method ORS is non-parameterized, thus its computational cost during the training of WSI classifier is negligible. This demonstrates that the enhancements introduced by SSRDL and ORS offer significant benefits without substantially increasing the computational burden.

In our view, a primary limitation of the proposed method is the lack of control over representation augmentation, which complicates its ability to perform as straightforwardly and controllably as traditional image augmentations. This issue is a common challenge faced by all methods of data augmentation for WSIs. Looking forward, we plan to explore potential solutions to enhance the control and applicability of representation augmentation. Despite these challenges, a key strength of our method, when compared to other WSI data augmentation methods, is its higher efficiency and superior performance.

As illustrated in Table VII, integrating WSI-level pretraining with our SSRDL framework and ORS strategy can be a promising direction to enhance the performance of WSI classification. This suggests that our method can effectively complement hierarchical pretraining approaches, such as WSI-level pretraining [49] and language-image representation learning [57]. Building on this, we can develop a multi-stage pan-cancer pathology foundation model utilizing a large medical dataset. This foundation model could enable the model to effectively extend its capabilities to more complex downstream tasks, such as survival prediction and gene mutation prediction.

VI. CONCLUSION

In this paper, we proposed a novel self-supervised representation distribution learning (SSRDL) framework with an online representation sampling (ORS) strategy for efficient data augmentation in histopathology WSI classification. Specifically, we built a distribution estimator into the DINO structure to produce representation augmentation, which has proven

effective to improving the quality of the representations in the self-supervised learning. Additionally, we designed an ORS strategy based on the distribution estimator for the WSI augmentation that can efficiently provide data augmentation for WSI-level model training. Through the visualization, we found that the representations from our representation augmentation module vary more widely than those with image augmentations. This demonstrates that representation augmentation has a stronger effect on image representations than the compositions of image augmentations, which is beneficial for patch representation learning and WSI analysis.

REFERENCES

[1] A. H. Song, G. Jaume, D. F. Williamson, M. Y. Lu, A. Vaidya, T. R. Miller, and F. Mahmood, "Artificial intelligence for digital and computational pathology," *Nature Reviews Bioengineering*, vol. 1, no. 12, pp. 930–949, 2023.

[2] J. Lipkova, R. J. Chen, B. Chen, M. Y. Lu, M. Barbieri, D. Shao, A. J. Vaidya, C. Chen, L. Zhuang, D. F. Williamson *et al.*, "Artificial intelligence for multimodal data integration in oncology," *Cancer cell*, vol. 40, no. 10, pp. 1095–1110, 2022.

[3] A. Shmatko, N. Ghaffari Laleh, M. Gerstung, and J. N. Kather, "Artificial intelligence in histopathology: enhancing cancer research and clinical oncology," *Nature cancer*, vol. 3, no. 9, pp. 1026–1038, 2022.

[4] B. E. Bejnordi, M. Veta, P. J. Van Diest, B. Van Ginneken, N. Karssemeijer, G. Litjens, J. A. Van Der Laak, M. Hermsen, Q. F. Manson, M. Balkenhol *et al.*, "Diagnostic assessment of deep learning algorithms for detection of lymph node metastases in women with breast cancer," *Jama*, vol. 318, no. 22, pp. 2199–2210, 2017.

[5] M. Y. Lu, D. F. Williamson, T. Y. Chen, R. J. Chen, M. Barbieri, and F. Mahmood, "Data-efficient and weakly supervised computational pathology on whole-slide images," *Nature Biomedical Engineering*, vol. 5, no. 6, pp. 555–570, 2021.

[6] Z. Shao, H. Bian, Y. Chen, Y. Wang, J. Zhang, X. Ji *et al.*, "Transmil: Transformer based correlated multiple instance learning for whole slide image classification," *Advances in neural information processing systems*, vol. 34, pp. 2136–2147, 2021.

[7] H. Zhang, Y. Meng, Y. Zhao, Y. Qiao, X. Yang, S. E. Coupland, and Y. Zheng, "Dtfd-mil: Double-tier feature distillation multiple instance learning for histopathology whole slide image classification," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 18 802–18 812.

[8] R. J. Chen, M. Y. Lu, D. F. Williamson, T. Y. Chen, J. Lipkova, Z. Noor, M. Shaban, M. Shady, M. Williams, B. Joo *et al.*, "Pan-cancer integrative histology-genomic analysis via multimodal deep learning," *Cancer Cell*, vol. 40, no. 8, pp. 865–878, 2022.

[9] X. Wang, Y. Du, S. Yang, J. Zhang, M. Wang, J. Zhang, W. Yang, J. Huang, and X. Han, "Retccl: Clustering-guided contrastive learning for whole-slide image retrieval," *Medical image analysis*, vol. 83, p. 102645, 2023.

[10] O. L. Saldanha, C. M. Loeffler, J. M. Niehues, M. van Treeck, T. P. Seraphin, K. J. Hewitt, D. Cifci, G. P. Veldhuizen, S. Ramesh, A. T. Pearson *et al.*, "Self-supervised attention-based deep learning for pan-cancer mutation prediction from histopathology," *NPJ Precision Oncology*, vol. 7, no. 1, p. 35, 2023.

[11] R. Yan, Y. Shen, X. Zhang, P. Xu, J. Wang, J. Li, F. Ren, D. Ye, and S. K. Zhou, "Histopathological bladder cancer gene mutation prediction with hierarchical deep multiple-instance learning," *Medical Image Analysis*, vol. 87, p. 102824, 2023.

[12] Y. Zheng, J. Li, J. Shi, F. Xie, J. Huai, M. Cao, and Z. Jiang, "Kernel attention transformer for histopathology whole slide image analysis and assistant cancer diagnosis," *IEEE Transactions on Medical Imaging*, 2023.

[13] J. Yang, H. Chen, Y. Zhao, F. Yang, Y. Zhang, L. He, and J. Yao, "Remix: A general and efficient framework for multiple instance learning based whole slide image classification," in *Medical Image Computing and Computer Assisted Intervention–MICCAI 2022: 25th International Conference, Singapore, September 18–22, 2022, Proceedings, Part II*. Springer, 2022, pp. 35–45.

[14] P. Liu, L. Ji, X. Zhang, and F. Ye, "Pseudo-bag mixup augmentation for multiple instance learning-based whole slide image classification," *IEEE Transactions on Medical Imaging*, 2024.

[15] R. J. Chen, C. Chen, Y. Li, T. Y. Chen, A. D. Trister, R. G. Krishnan, and F. Mahmood, "Scaling vision transformers to gigapixel images via hierarchical self-supervised learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 16 144–16 155.

[16] R. Nakhli, P. A. Moghadam, H. Mi, H. Farahani, A. Baras, B. Gilks, and A. Bashashati, "Sparse multi-modal graph transformer with shared-context processing for representation learning of giga-pixel images," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 11 547–11 557.

[17] J. Zhang, X. Zhang, K. Ma, R. Gupta, J. Saltz, M. Vakalopoulou, and D. Samaras, "Gigapixel whole-slide images classification using locally supervised learning," in *Medical Image Computing and Computer Assisted Intervention–MICCAI 2022: 25th International Conference, Singapore, September 18–22, 2022, Proceedings, Part II*. Springer, 2022, pp. 192–201.

[18] M. Tran, S. J. Wagner, M. Boxberg, and T. Peng, "S5cl: Unifying fully-supervised, self-supervised, and semi-supervised learning through hierarchical contrastive learning," in *Medical Image Computing and Computer Assisted Intervention–MICCAI 2022: 25th International Conference, Singapore, September 18–22, 2022, Proceedings, Part II*. Springer, 2022, pp. 99–108.

[19] P. Chikontwe, S. J. Nam, H. Go, M. Kim, H. J. Sung, and S. H. Park, "Feature re-calibration based multiple instance learning for whole slide image classification," in *Medical Image Computing and Computer Assisted Intervention–MICCAI 2022: 25th International Conference, Singapore, September 18–22, 2022, Proceedings, Part II*. Springer, 2022, pp. 420–430.

[20] M. Ilse, J. Tomczak, and M. Welling, "Attention-based deep multiple instance learning," in *International conference on machine learning*. PMLR, 2018, pp. 2127–2136.

[21] B. Li, Y. Li, and K. W. Eliceiri, "Dual-stream multiple instance learning network for whole slide image classification with self-supervised contrastive learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 14 318–14 328.

[22] G. Campanella, M. G. Hanna, L. Geneslaw, A. Miraflor, V. Werneck Krauss Silva, K. J. Busam, E. Brogi, V. E. Reuter, D. S. Klimstra, and T. J. Fuchs, "Clinical-grade computational pathology using weakly supervised deep learning on whole slide images," *Nature medicine*, vol. 25, no. 8, pp. 1301–1309, 2019.

[23] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[24] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations*, 2021.

[25] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International journal of computer vision*, vol. 115, pp. 211–252, 2015.

[26] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *International conference on machine learning*. PMLR, 2020, pp. 1597–1607.

[27] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. Richemond, E. Buchatskaya, C. Doersch, B. Avila Pires, Z. Guo, M. Gheshlaghi Azar *et al.*, "Bootstrap your own latent-a new approach to self-supervised learning," *Advances in neural information processing systems*, vol. 33, pp. 21 271–21 284, 2020.

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

[29] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin, "Emerging properties in self-supervised vision transformers," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 9650–9660.

[30] D. Tellez, G. Litjens, P. Bándi, W. Bulten, J.-M. Bokhorst, F. Ciompi, and J. Van Der Laak, "Quantifying the effects of data augmentation and stain color normalization in convolutional neural networks for computational pathology," *Medical image analysis*, vol. 58, p. 101544, 2019.

[31] C. Doersch and A. Zisserman, "Multi-task self-supervised visual learning," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2051–2060.

[32] S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised representation learning by predicting image rotations," in *International Conference on Learning Representations*, 2018.

[33] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin, "Unsupervised learning of visual features by contrasting cluster assignments," *Advances in neural information processing systems*, vol. 33, pp. 9912–9924, 2020.

[34] Y. Zang, C. Huang, and C. C. Loy, "Fasa: Feature augmentation and sampling adaptation for long-tailed instance segmentation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 3457–3466.

[35] A. Antoniou, A. Storkey, and H. Edwards, "Data augmentation generative adversarial networks," *arXiv preprint arXiv:1711.04340*, 2017.

[36] B. Li, F. Wu, S.-N. Lim, S. Belongie, and K. Q. Weinberger, "On feature normalization and data augmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 12 383–12 392.

[37] Z. Shao, L. Dai, Y. Wang, H. Wang, and Y. Zhang, "Augdiff: Diffusion based feature augmentation for multiple instance learning in whole slide image," *arXiv preprint arXiv:2303.06371*, 2023.

[38] I. Zaffar, G. Jaume, N. Rajpoot, and F. Mahmood, "Embedding space augmentation for weakly supervised learning in whole-slide images," in *2023 IEEE 20th International Symposium on Biomedical Imaging (ISBI)*. IEEE, 2023, pp. 1–4.

[39] Y.-C. Chen and C.-S. Lu, "Rankmix: Data augmentation for weakly supervised learning of classifying whole slide images with diverse sizes and imbalanced categories," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 23 936–23 945.

[40] M. Gadermayr, L. Koller, M. Tschuchnig, L. M. Stangassinger, C. Kreutzer, S. Couillard-Despres, G. J. Oostingh, and A. Hittmair, "Mixup-mil: Novel data augmentation for multiple instance learning and a study on thyroid cancer diagnosis," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2023, pp. 477–486.

[41] A. Jaiswal, A. R. Babu, M. Z. Zadeh, D. Banerjee, and F. Makedon, "A survey on contrastive self-supervised learning," *Technologies*, vol. 9, no. 1, p. 2, 2020.

[42] A. Dosovitskiy, J. T. Springenberg, M. Riedmiller, and T. Brox, "Discriminative unsupervised feature learning with convolutional neural networks," *Advances in neural information processing systems*, vol. 27, 2014.

[43] Z. Wu, Y. Xiong, S. X. Yu, and D. Lin, "Unsupervised feature learning via non-parametric instance discrimination," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 3733–3742.

[44] M. Hamilton, Z. Zhang, B. Hariharan, N. Snavely, and W. T. Freeman, "Unsupervised semantic segmentation by distilling feature correspondences," in *International Conference on Learning Representations*, 2022.

[45] H. S. Seong, W. Moon, S. Lee, and J.-P. Heo, "Leveraging hidden positives for unsupervised semantic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 19 540–19 549.

[46] I. Misra and L. v. d. Maaten, "Self-supervised learning of pretext-invariant representations," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 6707–6717.

[47] X. Chen and K. He, "Exploring simple siamese representation learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 15 750–15 758.

[48] S. Gidaris, A. Bursuc, N. Komodakis, P. Pérez, and M. Cord, "Learning representations by predicting bags of visual words," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 6928–6938.

[49] K. Wu, Y. Zheng, J. Shi, F. Xie, and Z. Jiang, "Position-aware masked autoencoder for histopathology wsi representation learning," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2023, pp. 714–724.

[50] C. Doersch, A. Gupta, and A. A. Efros, "Unsupervised visual representation learning by context prediction," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1422–1430.

[51] Y. M. Asano, C. Rupprecht, and A. Vedaldi, "A critical analysis of self-supervision, or what we can learn from a single image," in *International Conference on Learning Representations*, 2020.

[52] X. Chen, H. Fan, R. Girshick, and K. He, "Improved baselines with momentum contrastive learning," *arXiv preprint arXiv:2003.04297*, 2020.

[53] C. Doersch, "Tutorial on variational autoencoders," *arXiv preprint arXiv:1606.05908*, 2016.

[54] X. Chen, S. Xie, and K. He, "An empirical study of training self-supervised vision transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2021, pp. 9640–9649.

[55] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.

[56] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of machine learning research*, vol. 9, no. 11, 2008.

[57] D. Hu, Z. Jiang, J. Shi, F. Xie, K. Wu, K. Tang, M. Cao, J. Huai, and Y. Zheng, "Histopathology language-image representation learning for fine-grained digital pathology cross-modal retrieval," *Medical Image Analysis*, vol. 95, p. 103163, 2024.