Medical Image Analysis 87 (2023) 102813

Contents lists available at ScienceDirect

Medical Image Analysis

journal homepage: www.elsevier.com/locate/media

Distribution based MIL pooling filters: Experiments on a lymph node metastases dataset

Mustafa Umit Oner [a,b,h,*], Jared Marc Song Kye-Jet [a,1], Hwee Kuan Lee [a,b,c,d,e,f], Wing-Kin Sung [b,g,i,j,k]

[a] A*STAR Bioinformatics Institute, 30 Biopolis Street, Singapore 138671, Singapore
[b] School of Computing, National University of Singapore, 13 Computing Drive, Singapore 117417, Singapore
[c] Singapore Eye Research Institute (SERI), 20 College Road, Singapore 169856, Singapore
[d] Image and Pervasive Access Lab (IPAL), 1 Fusionopolis Way, Singapore 138632, Singapore
[e] Rehabilitation Research Institute of Singapore, 11 Mandalay Road, Singapore 308232, Singapore
[f] Singapore Institute for Clinical Sciences, Singapore, Singapore
[g] A*STAR Genome Institute of Singapore, 60 Biopolis Street, Singapore 138672, Singapore
[h] Artificial Intelligence Engineering, Bahcesehir University, Besiktas, Istanbul 34349, Turkey
[i] Hong Kong Genome Institute, Hong Kong Science Park, Shatin, Hong Kong, China
[j] Department of Chemical Pathology, The Chinese University of Hong Kong, Hong Kong, China
[k] Laboratory of Computational Genomics, Li Ka Shing Institute of Health Sciences, The Chinese University of Hong Kong, Hong Kong, China

ARTICLE INFO

Dataset link: https://github.com/onermustafaumit/mil_pooling_filters

Keywords:
Multiple instance learning (MIL)
MIL pooling filters
Distribution pooling
Point estimate based pooling

ABSTRACT

Histopathology is a crucial diagnostic tool in cancer and involves the analysis of gigapixel slides. Multiple instance learning (MIL) promises success in digital histopathology thanks to its ability to handle gigapixel slides and work with weak labels. MIL is a machine learning paradigm that learns the mapping between bags of instances and bag labels. It represents a slide as a bag of patches and uses the slide's weak label as the bag's label. This paper introduces *distribution-based pooling filters* that obtain a bag-level representation by estimating marginal distributions of instance features. We formally prove that the distribution-based pooling filters are more expressive than the classical point estimate-based counterparts, like 'max' and 'mean' pooling, in terms of the amount of information captured while obtaining bag-level representations. Moreover, we empirically show that models with distribution-based pooling filters perform equal to or better than those with point estimate-based pooling filters on distinct real-world MIL tasks defined on the CAMELYON16 lymph node metastases dataset. Our model with a distribution pooling filter achieves an area under the receiver operating characteristics curve value of 0.9325 (95% confidence interval: 0.8798 - 0.9743) in the tumor vs. normal slide classification task.

1. Introduction

Cancer is estimated to be responsible for 9.96 million deaths in 2020, and there will be an estimated 30.2 million new cases and 16.3 million deaths by 2040 (Ferlay et al., 2020). In the early detection and successful treatment of cancer, histopathology is a crucial diagnostic tool. Recently, slide scanners have transformed histopathology into digital, where glass slides are scanned and stored as digital images, namely whole slide images (WSIs). Histopathology images provide precious data that powerful deep learning models can exploit. However, a histopathology image is a gigapixel image that traditional deep learning models cannot process. Besides, deep learning models require a lot of labeled data. Nevertheless, most histopathology images are either not annotated or annotated with some weak labels indicating coarse-level (slide-level or sample-level) properties. They are seldom annotated with region-of-interests.

This study develops novel multiple instance learning (MIL) models to address these challenges in digital histopathology. MIL is a machine learning paradigm that learns the mapping between bags of instances and bag labels. MIL models are handy for tasks where data is in the form of bags of instances, and only bag labels are provided. For example, medical image processing tasks are typical MIL tasks since an image can be treated as a bag of pixels or small patches (instances), and there usually exists only an image (bag) label without providing

* Corresponding author at: Artificial Intelligence Engineering, Bahcesehir University, Besiktas, Istanbul 34349, Turkey.
E-mail addresses: mustafaumit.oner@bau.edu.tr (M.U. Oner), leehk@bii.a-star.edu.sg (H.K. Lee), ksung@comp.nus.edu.sg (W.-K. Sung).
[1] The author contributed to this work during his internship in A*STAR Bioinformatics Institute.

https://doi.org/10.1016/j.media.2023.102813
Received 13 June 2022; Received in revised form 13 February 2023; Accepted 7 April 2023
Available online 20 April 2023

the particular region-of-interest (Zhu et al., 2017; Campanella et al., 2019; Tomita et al., 2019; Oner et al., 2022).

Different MIL methods are used to solve different MIL tasks. Some methods first classify instances inside bags and then pool the instances' scores using a MIL pooling filter (Dietterich et al., 1997; Maron and Lozano-Pérez, 1998; Andrews et al., 2003; Zhang and Goldman, 2002). Others first extract features of instances inside bags, then obtain bag-level representations using a MIL pooling filter, and finally classify the bags (Wang et al., 2018; Ilse et al., 2018; Oner et al., 2020). The common component in the two approaches is the MIL pooling filter.

A MIL pooling filter obtains a bag-level representation from extracted features of instances. An ideal MIL pooling filter would obtain a bag-level representation as joint distribution of features to capture the complete information in extracted features. However, it is computationally intractable due to the high-dimensional nature of the problem. As a solution, standard MIL pooling filters, such as 'max' pooling (Wang et al., 2018; Wu et al., 2015; Feng and Zhou, 2017), 'mean' pooling (Wang et al., 2018, 2019) or 'attention' pooling (Ilse et al., 2018), obtain point estimates (like the mean) of the features of all instances in the bag. We call this kind of pooling filters *point estimate-based pooling filters* (Section 3.2.1).

Although point estimate-based pooling filters perform well in many applications, they only capture minimal information for each feature (like feature mean of all instances in the bag). On the other hand, it is possible to capture richer information by estimating, for each feature, the distribution of the feature values of all instances in the bag. This paper introduces a new family of MIL pooling filters that obtains a bag-level representation by estimating marginal feature distributions. We call this type of pooling filters *distribution-based pooling filters* (Section 4). Distribution-based pooling filters have a notable advantage over point estimate-based pooling filters: they enable users to represent the information by the shape of the distributions rather than by the point estimates.

Furthermore, empirical comparison of different point estimate-based pooling filters have been done in some specific applications like sound event detection (Wang et al., 2019). However, they do not go beyond one specific application and have a lack of theoretical analysis. This paper performs the first systematic study of both point estimate-based and distribution-based pooling filters. We theoretically show that distribution-based pooling filters are more expressive than point estimate-based counterparts in terms of the amount of information captured while obtaining bag-level representations from extracted features (Section 5). This property of distribution-based pooling filters is of great importance since more information can improve performance.

We tested the real-life performance of pooling filters in distinct MIL tasks formulated on a lymph node metastases dataset, namely CAMELYON16 (Bejnordi et al., 2017). The dataset consists of histopathology slides of lymph node sections and it has corresponding ground-truth segmentation masks. Firstly, for an extensive performance analysis of different pooling filters, we used the histopathology image patches dataset prepared by Oner et al. (2020) from the CAMELYON16 slides. We formulated five different MIL tasks on this dataset (Section 6.1). Consistent with our theoretical analysis, distribution-based pooling filters mostly outperformed their point estimate-based counterparts in these tasks. Secondly, we evaluated the performance of our model with a distribution pooling filter in the task of CAMELYON16 WSI classification task (Section 6.2). Our model had a better performance than the models with point estimate-based pooling filters. Lastly, we tested the performance of the 'distribution' pooling filter on the MNIST-bags classification task of Ilse et al. (2018) and bag classification task on classical MIL datasets (Dietterich et al., 1997; Andrews et al., 2003) (Section 6.3 and Section 6.4). Our model mostly outperformed state-of-the-art MIL methods.

Hence, this paper has three main contributions:

1. We introduce the family of distribution-based pooling filters.
2. We theoretically analyzed different MIL pooling filters and showed that distribution-based pooling filters are more expressive than point estimate-based counterparts in terms of the amount of information captured while obtaining bag-level representations.
3. We experimentally showed that models with distribution-based pooling filters have equal or better performance than those with point estimate-based pooling filters in MIL tasks defined on the lymph node metastases dataset and two other classical MIL datasets.

2. Related work

2.1. Multiple instance learning

MIL was first introduced as a positive vs. negative bag classification task for drug activity prediction (Dietterich et al., 1997; Maron and Lozano-Pérez, 1998). After that, different versions of MIL tasks emerged: unique class count prediction (Oner et al., 2020), multi-class classification (Feng and Zhou, 2017), multi-task classification (Yang et al., 2016), or regression (Zhang et al., 2018). In order to solve these tasks, different MIL methods were derived with different assumptions (Gärtner et al., 2002; Zhang and Goldman, 2002; Chen et al., 2006; Foulds, 2008; Zhang and Zhou, 2009; Zhou et al., 2009; Ramon and De Raedt, 2000; Zhou and Zhang, 2002; Zhang and Zhou, 2004), which are reviewed in detail in Foulds and Frank (2010). Recently, there has been a massive shift towards using neural networks in MIL to exploit the power of deep learning (Wu et al., 2015; Wang et al., 2018).

2.2. MIL pooling filters

Different MIL pooling filters are used to combine extracted features of instances inside the bags, such as 'max' pooling (Wang et al., 2018; Wu et al., 2015; Feng and Zhou, 2017), 'mean' pooling (Wang et al., 2018, 2019), or 'log-sum-exp' pooling (Ramon and De Raedt, 2000). Although these filters are widely used in the literature, there are new kinds of MIL pooling filters as well, such as 'attention' pooling (Pappas and Popescu-Belis, 2017; Ilse et al., 2018; Lee et al., 2019; Wang et al., 2019) or 'sort' pooling (Lu et al., 2015; Zhang et al., 2020).

2.3. MIL in medical image analysis

Modeling an image as a bag of pixels or smaller patches and using image-level weak labels as the bag's label at the absence of pixel-level annotations make MIL models attractive in image processing (Chen and Wang, 2004; Zhang et al., 2002; Tang et al., 2010). When we consider the difficulty and cost of obtaining pixel-level annotations for medical images, MIL models becomes even more practical and useful since they help us exploit the readily available weak labels (Dundar et al., 2007; Quellec et al., 2017). Especially in processing gigapixel digital histopathology images, MIL models have recently became a commonly used approach (Campanella et al., 2019; Tomita et al., 2019; Chikontwe et al., 2020; Hashimoto et al., 2020; Li et al., 2021; Lu et al., 2021; Shao et al., 2021; Myronenko et al., 2021; Zhang et al., 2022; Oner et al., 2022).

MIL models with different pooling filters are used in histopathology image analysis. An MIL model with a max pooling filter followed by a recurrent neural network was used to classify core needle biopsy slides of prostate cancer in Campanella et al. (2019). This was the core of the AI product receiving first ever FDA approval in digital pathology. Attention pooling of Ilse et al. (2018) is another method commonly used in MIL models for histopathology image analysis (Tomita et al., 2019; Hashimoto et al., 2020; Zhang et al., 2022; Lu et al., 2021). Attention-based pooling is also implemented using transformer architectures in MIL models (Myronenko et al., 2021; Shao et al., 2021). Most of the

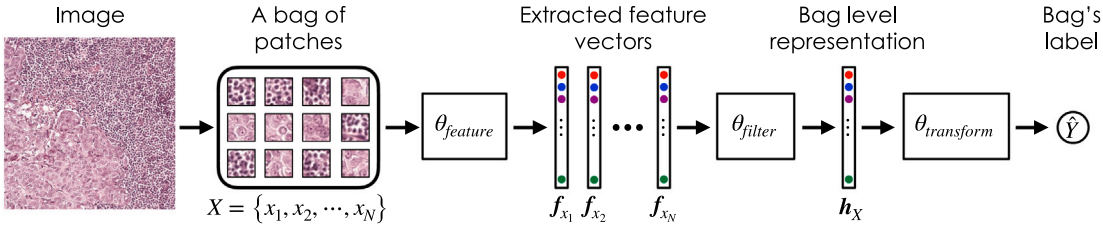


Fig. 1. Block diagram of MIL framework. Let X be a bag of patches (instances) cropped from an image and Y be the bag-level (image-level) label indicating if the bag (image) contains any cancerous patch. For each patch $x_i \in X$, the *feature extractor* module θ_{feature} extracts a feature vector $\boldsymbol{f}_{x_i} \in \mathcal{F}$. Then, the *MIL pooling filter* module θ_{filter} aggregates the extracted feature vectors and obtains a bag-level representation $\boldsymbol{h}_X \in \mathcal{H}$. Lastly, the *bag-level representation transformation* module $\theta_{\text{transform}}$ transforms bag-level representation into predicted bag label $\hat{Y} \in \mathcal{Y}$. Note that labels of individual patches are not known.

MIL models use feature vectors of a WSI's patches extracted using a pre-trained model (Lu et al., 2021; Zhang et al., 2022; Shao et al., 2021). However, some other techniques are also devised in the literature to obtain better instance features. For example, Li et al. (2021) used self-supervised contrastive learning to obtain better instance features. Besides, they used image patches from different resolution levels of slides to better capture spatial information within the slides. Similarly, Hashimoto et al. (2020) successfully used multi-scale approach in their MIL model with attention pooling. They also incorporated an auxiliary loss component over instances in a bag to improve their model's performance. The auxiliary loss approach was successfully used in Lu et al. (2021) and Chikontwe et al. (2020) as well. Different than the MIL models using point estimate-based pooling filters, this study introduces a new pooling method, namely distribution pooling, to capture more information while obtaining bag-level representations from instance features.

3. Multiple instance learning framework

In the MIL paradigm, the objective is to predict a bag label Y for a given bag of instances $X \subseteq \mathcal{I}$, where $\mathcal{I}$ is the instance space. Each instance $x_i \in X$ is endowed with an underlying label $y_i \in \mathcal{L}$, where $\mathcal{L}$ is the instance label space. However, instance labels are inaccessible during training. The definition of the MIL task imposes the relation between the bag label and instance labels. Note that although the number of instances in each bag can be different in the real world, it is treated as constant in this paper for clarity of notation. Nevertheless, all the properties stated here are also valid for bags with a variable number of instances.

Let $\mathcal{D}$ be a MIL dataset such that for each $(X, Y) \in \mathcal{D}$, $X = \{x_1, x_2, \dots, x_N\} \subseteq \mathcal{I}$ and $Y \in \mathcal{Y}$, where N is the number of instances inside the bag and $\mathcal{Y}$ is the bag label space. Given any pair $(X, Y) \in \mathcal{D}$, our objective is to predict bag label Y for bag X. We obtain predicted bag label $\hat{Y}$ using a three-stage framework (Fig. 1).

The first stage is a *feature extractor* module $\theta_{\text{feature}} : \mathcal{I} \to \mathcal{F}$, where $\mathcal{F}$ is the feature space. For each $x_i \in X$, it takes x_i as input, extracts J features and outputs a feature vector: $\boldsymbol{f}_{x_i} = \theta_{\text{feature}}(x_i) = [f^1_{x_i}, f^2_{x_i}, \dots, f^J_{x_i}] \in \mathcal{F} = \mathbb{R}^J$ where $f^j_{x_i} \in \mathbb{R}$ is the jth feature value. Let $\boldsymbol{F}_X = [\boldsymbol{f}_{x_1}, \boldsymbol{f}_{x_2}, \dots, \boldsymbol{f}_{x_N}] \in \mathbb{R}^{JN}$ be feature matrix constructed from extracted feature vectors such that ith column corresponds to $\boldsymbol{f}_{x_i}$. The second stage is a *MIL pooling filter* module $\theta_{\text{filter}} : \mathbb{R}^{JN} \to \mathcal{H}$, where $\mathcal{H}$ is the bag-level representation space. It takes the feature matrix $\boldsymbol{F}_X$ and aggregates the extracted feature vectors to obtain a bag-level representation: $\boldsymbol{h}_X = \theta_{\text{filter}}(\boldsymbol{F}_X) \in \mathcal{H}$ where $\mathcal{H}$ depends on θ_{filter}. For example, 'max' pooling gets the maximum value of each feature ($\mathcal{H} = \mathbb{R}^J$) or 'distribution' pooling estimates the marginal feature distributions ($\mathcal{H}$ is a distribution space). The last stage is a *bag-level representation transformation* module $\theta_{\text{transform}} : \mathcal{H} \to \mathcal{Y}$. It transforms bag-level representation into predicted bag label: $\hat{Y} = \theta_{\text{transform}}(\boldsymbol{h}_X)$.

We use neural networks to implement θ_{feature} and $\theta_{\text{transform}}$ so that we can fully parameterize the learning process. For θ_{filter}, we use different filters, some of which (e.g., 'attention' and 'distribution' pooling filter) also incorporate trainable components parameterized by neural networks. Note that some filters (e.g., 'mean' and 'max' pooling) have no trainable components. This system of neural networks is end-to-end trainable.

3.1. MIL tasks

MIL paradigm in its most general form can be seen as a bag classification/regression problem. This generic problem can be formulated as different types of MIL tasks. We introduce five of such tasks used in our experiments. Given a bag $X = \{x_1, x_2, \dots, x_N\}$, the bag label Y is defined as:

- **Positive vs negative bag classification:** $Y = \max_{i=1}^{N} y_i \in \{0, 1\}$ where $y_i \in \{0, 1\}$ is the label of instance $x_i \in X$. The labels 0 and 1 correspond to 'negative' and 'positive', respectively. A bag is 'negative' if and only if all instances in the bag are 'negative'; otherwise, it is 'positive'. Note that instance labels, $\{y_1, y_2, \dots, y_N\}$, are inaccessible during training.
- **Unique class count classification** (Oner et al., 2020): $Y = |\{y_1, y_2, \dots, y_N\}| \in \{1, 2, \cdots, L\}$ where $y_i \in \{1, 2, \dots, L\}$ is the label of instance $x_i \in X$.
  Unique class count (ucc) is the number of unique classes among all instances inside the bag X.
  While the instance labels 1, 2, ⋯, L correspond to *'class1'*, *'class2'*, ⋯, *'classL'*, respectively, the bag labels 1, 2, ⋯, L correspond to *'ucc1'*, *'ucc2'*, ⋯, *'uccL'*, respectively.
  Note that instance labels, $\{y_1, y_2, \dots, y_N\}$, are inaccessible during training.
- **MIL multi-class classification:** $Y = [Y^1, Y^2, \dots, Y^K] \in \{0, 1\}^K$ where $\sum_{k=1}^{K} Y^k = 1$. Label vector Y is a one-hot vector consisting of K classes, i.e. the bag X belongs to only one class (class $\bar{k}, \bar{k} \in \{1, 2, \dots, K\}$) and only the value corresponding to that class is set to 1 ($Y^{\bar{k}} = 1$).
- **MIL multi-task classification:** $Y = [Y^1, Y^2, \dots, Y^K] \in \{0, 1\}^K$. Label vector Y is a binary vector consisting of K classes, which may not be mutually exclusive. For example, if the bag X belongs to two classes (class $\bar{k}_1$ and class $\bar{k}_2$, $\bar{k}_1, \bar{k}_2 \in \{1, 2, \dots, K\}$), only the values corresponding to those classes are set to 1 ($Y^{\bar{k}_1} = 1$ and $Y^{\bar{k}_2} = 1$).
- **Regression:** $Y \in \mathbb{R}$. This is a regression task in MIL form, in which the bag X has a real-valued bag-level label.

3.2. MIL pooling filters

A MIL pooling filter obtains a bag-level representation of a bag by aggregating instance-level representations of all instances in the bag. This paper groups MIL pooling filters into two types: *point estimate-based pooling filters* (Section 3.2.1) and *distribution-based pooling filters* (Section 4). The visual summary of pooling filters used in this study is given in Fig. 2.

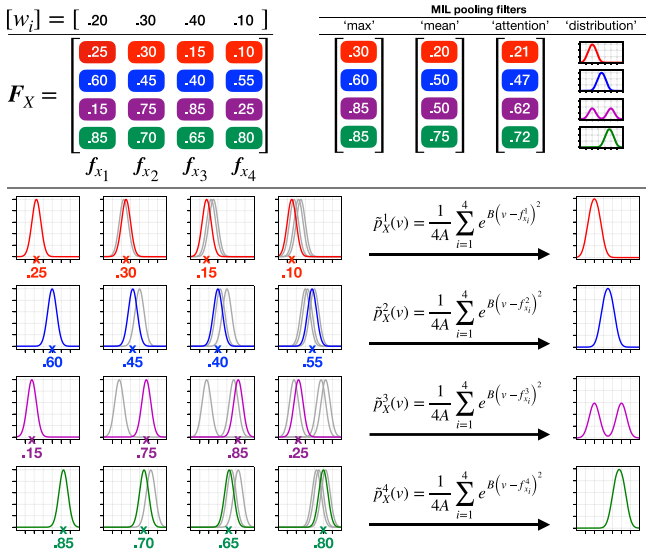


Fig. 2. MIL pooling filters. *Top:* The feature matrix $\boldsymbol{F}_X$ obtained from the bag $X = \{x_1, x_2, x_3, x_4\}$ contains 4 feature vectors $\boldsymbol{f}_{x_1}, \boldsymbol{f}_{x_2}, \boldsymbol{f}_{x_3}$ and $\boldsymbol{f}_{x_4}$. Each feature vector consists of 4 features highlighted with different colors. $[w_i]$ gives attention weights for 'attention' pooling. $\alpha_i = 1$ and $\beta_i = \frac{1}{N}\ \forall_i$ are parameters for 'distribution' pooling. Outputs of different MIL pooling filters are shown. *Bottom:* Obtaining marginal feature distributions is shown. The Gaussian kernel for each extracted feature value is illustrated with colored curves and previously accumulated kernels are shown in gray. Estimated feature distributions are obtained by employing Eq. (1). $A = \sqrt{2\pi\sigma^2}$ and $B = -\frac{1}{2\sigma^2}$.

3.2.1. Point estimate-based pooling filters

Given feature vectors $\boldsymbol{f}_{x_i} = \{f^1_{x_i}, f^2_{x_i}, \dots, f^J_{x_i}\} \in \mathbb{R}^J$ for $i = 1, 2, \dots, N$ obtained from a bag $X = \{x_1, x_2, \dots, x_N\}$, a point estimate-based pooling filter aims to find a bag-level representation $\boldsymbol{h}_X = [h^1_X, h^2_X, \dots, h^J_X] \in \mathcal{H} = \mathbb{R}^J$, where $h^j_X \in \mathbb{R}$ is the calculated point statistic of the jth feature. For 'max', 'mean', and 'attention' pooling, the values h^j_X are:

- **Max pooling:** $h^j_X = \max_{i=1}^N f^j_{x_i}\ \forall_{j=1,2,\dots,J}$.
- **Mean pooling:** $h^j_X = \frac{1}{N}\sum_{i=1}^N f^j_{x_i}\ \forall_{j=1,2,\dots,J}$.
- **Attention pooling** (Ilse et al., 2018): $h^j_X = \sum_{i=1}^N w_i f^j_{x_i}\ \forall_{j=1,2,\dots,J}$. Each instance has an attention weight w_i obtained from a neural network module $\mathcal{W}$ such that $w_i = \frac{\exp\{\mathcal{W}(\boldsymbol{f}_{x_i})\}}{\sum_{t=1}^N \exp\{\mathcal{W}(\boldsymbol{f}_{x_t})\}}\ \forall_{i=1,2,\dots,N}$. Note that $\sum_{i=1}^N w_i = 1$.

This paper included 'max', 'mean', and 'attention' pooling as basic point estimate-based pooling filters. However, most of the other point estimate-based pooling filters in the literature can be written in terms of these basic ones. For example, 'sum' pooling (Zaheer et al., 2017) is the scaled version of 'mean' pooling by N; 'exponential softmax' pooling (Wang et al., 2019) is equivalent to 'attention' pooling with attention network being 'identity' mapping; or 'top-k instance' pooling (Li and Vasconcelos, 2015) is equivalent to 'max' pooling for k=1 and 'mean' pooling for k=N.

4. Distribution-based pooling filters

Point estimate-based pooling filters obtain bag-level representations by calculating point statistics of the extracted features, which causes information loss. On the other hand, complete information can be captured by estimating the joint distribution of all extracted features. However, it is computationally intractable. This paper introduces distribution-based pooling filters that obtain a bag-level representation by estimating marginal feature distributions to capture as much information as possible.

Given feature vectors $\boldsymbol{f}_{x_i} = \{f^1_{x_i}, f^2_{x_i}, \dots, f^J_{x_i}\} \in \mathbb{R}^J$ for $i = 1, 2, \dots, N$ obtained from a bag $X = \{x_1, x_2, \dots, x_N\}$, a distribution-based pooling filter aims to find a bag-level representation $\boldsymbol{h}_X = [\tilde{p}^1_X, \tilde{p}^2_X, \cdots, \tilde{p}^J_X] \in \mathcal{H} = \mathbb{P}^J$, where $\mathbb{P}$ is the set of all marginal distributions and $\tilde{p}^j_X \in \mathbb{P}$ is the estimated marginal distribution of the jth feature. $\tilde{p}^j_X : \mathbb{R} \to \mathbb{R}^+ \cup \{0\}$ is calculated using kernel density estimation (Parzen, 1962), which employs a Gaussian kernel with standard deviation σ (Eq. (1) and Fig. 2). Note that Eq. (1) defines a family of distribution-based pooling filters, and the method employed in Oner et al. (2020) is a member of this family with $\alpha_i = 1$ and $\beta_i = \frac{1}{N}\ \forall_i$. Different than the method employed in Oner et al. (2020), this study incorporates coefficients of α_i and β_i into distribution pooling process. These coefficients are obtained using attention-based neural networks and learned during training. Moreover, this study presents a comprehensive theoretical analysis of distribution-based pooling filters.

$$\tilde{p}^j_X(v) = \sum_{i=1}^N \beta_i \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}\left(v - \alpha_i f^j_{x_i}\right)^2}\ \forall_{j=1,2,\dots,J} \tag{1}$$

Each instance $x_i \in X$ has a feature weight $\alpha_i = \mathcal{W}_\alpha(\boldsymbol{f}_{x_i})\ \forall_{i=1,2,\dots,N}$ and a kernel weight $\beta_i = \frac{\exp\{\mathcal{W}_\beta(\boldsymbol{f}_{x_i})\}}{\sum_{i=1}^N \exp\{\mathcal{W}_\beta(\boldsymbol{f}_{x_i})\}}\ \forall_{i=1,2,\dots,N}$, where $\mathcal{W}_\alpha$ and $\mathcal{W}_\beta$ are attention-based neural networks. These two weights control both the support and shape of the estimated marginal distribution. While α_i affects the support of marginal distribution by shifting the position of the Gaussian kernel, β_i affects the value of density function at that specific kernel position. Using proper attention weights, distribution-based pooling filters can prioritize some of the instances inside a bag, which is essential if all instances in the bag do not contribute equally to the bag label. For example, even one 'positive' instance makes a bag 'positive' in the positive vs. negative bag classification task (see Section 3.1). A distribution-based pooling filter can easily capture this instance while obtaining the bag-level representation.

Another notable advantage of distribution-based pooling filters is that they enable the $\theta_{\text{transform}}$ module to fully utilize the information inside the distributions rather than looking at the point estimates as in 'max', 'mean', and 'attention' pooling. Besides, the point estimates obtained by point estimate-based pooling filters, in principle, can be fully recovered from the estimated marginal distributions obtained by distribution-based pooling filters (see Section 5).

5. Theoretical analysis of MIL pooling filters

This section is reserved to theoretically show that distribution-based pooling filters are more expressive than their point estimate-based counterparts in terms of the amount of information captured.

First, we need to introduce two lemmas used to prove the propositions.

Lemma 1. *Given two sets* $R = \{r_i | r_i \in \mathbb{R},\ i = 1, 2, \dots, N_R\}$ *and* $S = \{s_j | s_j \in \mathbb{R},\ j = 1, 2, \dots, N_S\}$. *Let* $q_R(v) = \sum_{i=1}^{N_R} e^{-\frac{1}{2\sigma^2}(v - r_i)^2}$ *and* $q_S(v) = \sum_{j=1}^{N_S} e^{-\frac{1}{2\sigma^2}(v - s_j)^2}$ *where* $0 < \sigma < \infty$. *If all* r_i *and* s_j *are different (i.e.* $r_i \neq s_j \forall_{i,j}$*), then* $\exists v \in \mathbb{R}\ q_R(v) \neq q_S(v)$.

Proof. Prove by contradiction.

(i) Suppose, for the sake of contradiction, that $q_R(v) = q_S(v)\ \forall v$. Let $Z = R \cup S = \{r_1, r_2, \dots, r_{N_R}, s_1, s_2, \dots, s_{N_S}\} = \{z_1, z_2, \dots, z_{N_Z}\}$ where $N_Z = N_R + N_S$.

Let $\boldsymbol{\Phi} = [\Phi_{u_1 u_2}]$ where $\Phi_{u_1 u_2} = e^{-\frac{1}{2\sigma^2}\left(z_{u_1} - z_{u_2}\right)^2}\ \forall_{u_1,u_2}$. Then,

$$\boldsymbol{\Phi}\begin{bmatrix}1\\ \vdots\\ 1\\ -1\\ \vdots\\ -1\end{bmatrix} = \begin{bmatrix} q_R(z_1) - q_S(z_1)\\ q_R(z_2) - q_S(z_2)\\ \vdots\\ q_R(z_{N_Z}) - q_S(z_{N_Z})\end{bmatrix} = \begin{bmatrix}0\\0\\ \vdots\\0\end{bmatrix}$$

This means that $\boldsymbol{\Phi}$ is of rank $< N_Z$. Hence, $\boldsymbol{\Phi}$ has no inverse.

(ii) However, $\boldsymbol{\Phi}_{u_1 u_2}$ is in the form of a Gaussian kernel, which is a positive definite radial basis function (Schaback, 2007), and all data points constituting $\boldsymbol{\Phi}$ are different from each other (i.e. $z_{u_1} \neq z_{u_2}\ \forall_{u_1 \neq u_2}$). Therefore, $\boldsymbol{\Phi}$ is a positive definite matrix and invertible (Schoenberg, 1938; Micchelli, 1986; Buhmann, 2010).

Hence, there is a contradiction between **(i)** and **(ii)**, so $\exists v \in \mathbb{R}$ $q_R(v) \neq q_S(v)$. ■

Lemma 2. *Given two sets* $R = \{r_i | r_i \in \mathbb{R},\ i = 1, 2, \dots, N_R\}$ *and* $S = \{s_j | s_j \in \mathbb{R},\ j = 1, 2, \dots, N_S\}$. *Let* $q_R(v) = \sum_{i=1}^{N_R} e^{-\frac{1}{2\sigma^2}(v-r_i)^2}$ *and* $q_S(v) = \sum_{j=1}^{N_S} e^{-\frac{1}{2\sigma^2}(v-s_j)^2}$ *where* $0 < \sigma < \infty$. *If* $R \neq S$, *then* $\exists v \in \mathbb{R}$ $q_R(v) \neq q_S(v)$.

Proof. We can write $R = R_{inter} \cup R_{diff}$ and $S = S_{inter} \cup S_{diff}$ where $R_{inter} = S_{inter} = R \cap S$, $R_{diff} = R - S$ and $S_{diff} = S - R$. Then,

$$q_R(v) = \underbrace{\sum_{r \in R_{inter}} e^{-\frac{1}{2\sigma^2}(v-r)^2}}_{q_{R_{inter}}(v)} + \underbrace{\sum_{r \in R_{diff}} e^{-\frac{1}{2\sigma^2}(v-r)^2}}_{q_{R_{diff}}(v)} = q_{R_{inter}}(v) + q_{R_{diff}}(v)$$

$$q_S(v) = \underbrace{\sum_{s \in S_{inter}} e^{-\frac{1}{2\sigma^2}(v-s)^2}}_{q_{S_{inter}}(v)} + \underbrace{\sum_{s \in S_{diff}} e^{-\frac{1}{2\sigma^2}(v-s)^2}}_{q_{S_{inter}}(v)} = q_{S_{inter}}(v) + q_{S_{diff}}(v)$$

(i) $R_{inter} = S_{inter}$, so $q_{R_{inter}}(v) = q_{S_{inter}}(v)\ \forall v$.

(ii) Condition of lemma: $R \neq S$, so $R_{diff} \cup S_{diff} \neq \emptyset$ and $R_{diff} \cap S_{diff} = \emptyset$. Therefore, from Lemma 1 $\exists v \in \mathbb{R}$ $q_{R_{diff}}(v) \neq q_{S_{diff}}(v)$.

Hence, from **(i)** and **(ii)** $\exists v \in \mathbb{R}$ $q_R(v) \neq q_S(v)$. ■

By Lemmas 1 and 2, we are ready to state and prove our actual propositions.

Proposition 1. *Given two feature matrices obtained from bags* $X = \{x_1, x_2, \dots, x_N\}$ *and* $Z = \{z_1, z_2, \dots, z_N\}$;

- $\boldsymbol{F}_X = [f^j_{x_i} | f^j_{x_i} \in \mathbb{R},\ f^j_{x_i} \neq f^j_{x_u}\ \forall_{i \neq u},\ i, u = 1, 2, \dots, N$ *and* $j = 1, 2, \dots, J]$
- $\boldsymbol{F}_Z = [f^j_{z_i} | f^j_{z_i} \in \mathbb{R},\ f^j_{z_i} \neq f^j_{z_u}\ \forall_{i \neq u},\ i, u = 1, 2, \dots, N$ *and* $j = 1, 2, \dots, J]$

and two pooling filters; 'max' pooling filter θ^{max}_{filter} *and 'distribution' pooling filter* θ^{dist}_{filter} *with* $\alpha_i = 1$ *and* $\beta_i = \frac{1}{N}\ \forall_i$. *Let* ${}_{max}\boldsymbol{h}_X$ *and* ${}_{max}\boldsymbol{h}_Z$ *be bag level representations obtained by* θ^{max}_{filter} *from* $\boldsymbol{F}_X$ *and* $\boldsymbol{F}_Z$, *respectively. Similarly, let* ${}_{dist}\boldsymbol{h}_X$ *and* ${}_{dist}\boldsymbol{h}_Z$ *be bag level representations obtained by* θ^{dist}_{filter} *from* $\boldsymbol{F}_X$ *and* $\boldsymbol{F}_Z$, *respectively. If* ${}_{max}\boldsymbol{h}_X \neq {}_{max}\boldsymbol{h}_Z$, *then* ${}_{dist}\boldsymbol{h}_X \neq {}_{dist}\boldsymbol{h}_Z$.

Proof. Let $\boldsymbol{F}^j_X$ and $\boldsymbol{F}^j_Z$ be jth feature sets for bags X and Z such that $\boldsymbol{F}^j_X = \{f^j_{x_1}, f^j_{x_2}, \dots, f^j_{x_N}\}$ and $\boldsymbol{F}^j_Z = \{f^j_{z_1}, f^j_{z_2}, \dots, f^j_{z_N}\}$.

(i) For $\theta^{max}_{\text{filter}}$, bag level representations: ${}_{max}\boldsymbol{h}_X = [{}_{max}h^1_X, {}_{max}h^2_X, \dots, {}_{max}h^J_X]$ where ${}_{max}h^j_X = \max_{i=1}^N f^j_{x_i}\ \forall_{j=1,2,\dots,J}$ ${}_{max}\boldsymbol{h}_Z = [{}_{max}h^1_Z, {}_{max}h^2_Z, \dots, {}_{max}h^J_Z]$ where ${}_{max}h^j_Z = \max_{i=1}^N f^j_{z_i}\ \forall_{j=1,2,\dots,J}$

Condition of proposition: ${}_{max}\boldsymbol{h}_X \neq {}_{max}\boldsymbol{h}_Z$, so $\exists j$ ${}_{max}h^j_X \neq {}_{max}h^j_Z$. Thus, $\exists j$ $\boldsymbol{F}^j_X \neq \boldsymbol{F}^j_Z$.

(ii) For $\theta^{dist}_{\text{filter}}$, $\alpha_i = 1$ and $\beta_i = \frac{1}{N}\ \forall_i$. Then bag level representations:

$${}_{dist}\boldsymbol{h}_X = \begin{bmatrix} \tilde{p}^1_X(v) \\ \vdots \\ \tilde{p}^j_X(v) \\ \vdots \\ \tilde{p}^J_X(v) \end{bmatrix} \text{ where } \tilde{p}^j_X(v) = \frac{1}{N}\sum_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}\left(v - f^j_{x_i}\right)^2} \forall_{j=1,2,\dots,J}$$

$${}_{dist}\boldsymbol{h}_Z = \begin{bmatrix} \tilde{p}^1_Z(v) \\ \vdots \\ \tilde{p}^j_Z(v) \\ \vdots \\ \tilde{p}^J_Z(v) \end{bmatrix} \text{ where } \tilde{p}^j_Z(v) = \frac{1}{N}\sum_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}\left(v - f^j_{z_i}\right)^2} \forall_{j=1,2,\dots,J}$$

We can also re-write $\tilde{p}^j_X(v)$ and $\tilde{p}^j_Z(v)$ as $\tilde{p}^j_X(v) = \frac{1}{N}\frac{1}{\sqrt{2\pi\sigma^2}}\sum_{r \in \boldsymbol{F}^j_X} e^{-\frac{1}{2\sigma^2}(v-r)^2}$ and $\tilde{p}^j_Z(v) = \frac{1}{N}\frac{1}{\sqrt{2\pi\sigma^2}}\sum_{s \in \boldsymbol{F}^j_Z} e^{-\frac{1}{2\sigma^2}(v-s)^2}$, respectively.

From (i) and (ii): We know that $\exists j$ $\boldsymbol{F}^j_X \neq \boldsymbol{F}^j_Z$, so by using Lemma 2 $\exists j$ $\exists v$ $\tilde{p}^j_X(v) \neq \tilde{p}^j_Z(v)$.

Hence, since $\exists j$ $\exists v$ $\tilde{p}^j_X(v) \neq \tilde{p}^j_Z(v)$, ${}_{dist}\boldsymbol{h}_X \neq {}_{dist}\boldsymbol{h}_Z$. ■

The Proposition 1 shows that given two feature matrices, if the 'max' pooling filter produces two different representations, so does the 'distribution' pooling filter. For example, given two bags from different classes and the corresponding feature matrices satisfying the conditions of Proposition 1, if the 'max' pooling filter discriminates two bags, i.e., produces two different bag-level representations, the 'distribution' pooling filter also discriminates them. However, the converse of Proposition 1 is not valid. Obtaining two different bag-level representations by the 'distribution' pooling filter does not guarantee obtaining different representations by the 'max' pooling filter since the feature matrices may still have the same maximum feature values for both of the bags. There are cases where the 'distribution' pooling filter discriminates two bags; however, the 'max' pooling filter cannot. Hence, we conclude that the 'distribution' pooling filter is more expressive than the 'max' pooling filter in terms of the amount of information captured in bag-level representations.

Proposition 2. *Given a feature matrix* $\boldsymbol{F}_X = [f^j_{x_i} | f^j_{x_i} \in \mathbb{R},\ i = 1, 2, \dots, N$ *and* $j = 1, 2, \dots, J]$ *obtained from a bag* $X = \{x_1, x_2, \dots, x_N\}$, *'mean' pooling filter* θ^{mean}_{filter} *and 'distribution' pooling filter* θ^{dist}_{filter} *with* $\alpha_i = 1$ *and* $\beta_i = \frac{1}{N}\ \forall_i$. *Let* ${}_{mean}\boldsymbol{h}_X = [{}_{mean}h^1_X, {}_{mean}h^2_X, \cdots, {}_{mean}h^J_X]$ *and* ${}_{dist}\boldsymbol{h}_X = [\tilde{p}^1_X, \tilde{p}^2_X, \dots, \tilde{p}^J_X]$ *be bag level representations obtained from* $\boldsymbol{F}_X$ *by* θ^{mean}_{filter} *and* θ^{dist}_{filter}, *respectively. Then,* ${}_{mean}h^j_X = \mathbb{E}[V^j]\ \forall_{j=1,2,\dots,J}$ *where* $V^j \sim \tilde{p}^j_X$.

Proof. (i) For $\theta^{mean}_{\text{filter}}$, bag level representation: ${}_{mean}\boldsymbol{h}_X = [{}_{mean}h^1_X, {}_{mean}h^2_X, \dots, {}_{mean}h^J_X]$ where ${}_{mean}h^j_X = \frac{1}{N}\sum_{i=1}^N f^j_{x_i}\ \forall_{j=1,2,\dots,J}$

(ii) For $\theta^{dist}_{\text{filter}}$, $\alpha_i = 1$ and $\beta_i = \frac{1}{N}\ \forall_i$. Then bag level representation:

$${}_{dist}\boldsymbol{h}_X = \begin{bmatrix} \tilde{p}^1_X(v) \\ \vdots \\ \tilde{p}^j_X(v) \\ \vdots \\ \tilde{p}^J_X(v) \end{bmatrix} \text{ where } \tilde{p}^j_X(v) = \frac{1}{N}\sum_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}\left(v - f^j_{x_i}\right)^2} \forall_{j=1,2,\dots,J}$$

From (i) and (ii): by using definition of expected value:

$${}_{mean}h^j_X = \int v \tilde{p}^j_X(v) dv\ \forall_{j=1,2,\dots,J}$$

Hence, ${}_{mean}h^j_X = \mathbb{E}[V^j]\ \forall_{j=1,2,\dots,J}$ where $V^j \sim \tilde{p}^j_X$. ■

The Proposition 2 proves that given a feature matrix, representation obtained by the 'mean' pooling filter can be fully recovered from representation obtained by the 'distribution' pooling filter.

Proposition 3. *Given a feature matrix* $\boldsymbol{F}_X = [f^j_{x_i} | f^j_{x_i} \in \mathbb{R},\ i = 1, 2, \dots, N$ *and* $j = 1, 2, \dots, J]$ *obtained from a bag* $X = \{x_1, x_2, \dots, x_N\}$, *'attention' pooling filter* θ^{att}_{filter} *with attention weights* $w_i > 0\ \forall_i$, $\sum_{i=1}^N w_i = 1$ *and 'distribution' pooling filter* θ^{dist}_{filter}. *Let* ${}_{att}\boldsymbol{h}_X = [{}_{att}h^1_X, {}_{att}h^2_X, \cdots, {}_{att}h^J_X]$ *be bag level representation obtained from* $\boldsymbol{F}_X$ *by* θ^{att}_{filter}. *If attention weights are accessible, then* ${}_{att}h^j_X = N \times \mathbb{E}[V^j]\ \forall_{j=1,2,\dots,J}$ *where* $V^j \sim \tilde{p}^j_X$ *and* ${}_{dist}\boldsymbol{h}_X = [\tilde{p}^1_X, \tilde{p}^2_X, \dots, \tilde{p}^J_X]$ *is the bag level representation obtained by* θ^{dist}_{filter} *with* $\alpha_i = w_i$ *and* $\beta_i = \frac{1}{N}\ \forall_i$.

Proof. (i) For $\theta^{att}_{\text{filter}}$, bag level representation: ${}_{att}\boldsymbol{h}_X = [{}_{att}h^1_X, {}_{att}h^2_X, \dots, {}_{att}h^J_X]$ ${}_{att}h^j_X = \sum_{i=1}^N w_i f^j_{x_i}\ \forall_j$ where attention weight $w_i > 0\ \forall_i$ and $\sum_{i=1}^N w_i = 1$

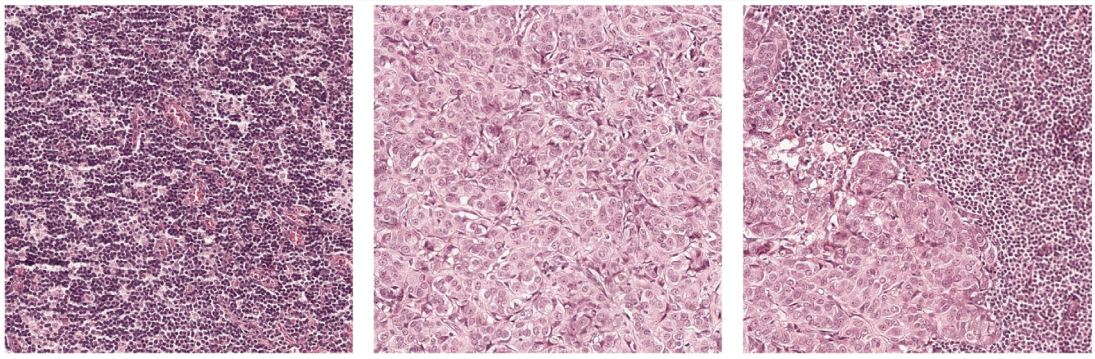
We can re-write ${}_{att}h^j_X$ as: ${}_{att}h^j_X = N \times \frac{1}{N}\sum_{i=1}^N w_i f^j_{x_i}\ \forall_j$

Now it is nothing but 'mean' pooling of weighted features multiplied with a scalar, so we can use Proposition 2.

Table 1
MIL tasks are summarized by bag labels for each kind of image and loss functions used during training. For example, in the +ve/-ve bag classification task, bag labels are 2-bit one-hot vectors such that '10': -ve bag (fully normal image) and '01': +ve bag (fully metastases or boundary image). Images are shown at the bottom.

	bag label (Y)				
	+ve/-ve (*one-hot*)	ucc (*one-hot*)	3-class (*one-hot*)	2-task (*binary*)	% metastases (*real-valued*)
Fully normal	10	10	100	1,0	0.0
Fully metastases	01	10	010	0,1	1.0
Boundary	01	01	001	1,1	0.0 < Y < 1.0
Loss	CCE	CCE	CCE	BCE	L1

CCE: Categorical Cross Entropy (PyTorch, 2020b), BCE: Binary Cross Entropy (PyTorch, 2020a)



Fully normal | Fully metastases | Boundary

(ii) For $\theta^{dist}_{\text{filter}}$, $\alpha_i = w_i$ and $\beta_i = \frac{1}{N}\ \forall_i$. Then bag level representation:

$$_{dist}\boldsymbol{h}_X = \begin{bmatrix} \tilde{p}^1_X(v) \\ \vdots \\ \tilde{p}^j_X(v) \\ \vdots \\ \tilde{p}^J_X(v) \end{bmatrix} \text{ where } \tilde{p}^j_X(v) = \frac{1}{N}\sum_{i=1}^{N} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}\left(v - w_i f^j_{X_i}\right)^2} \forall_{j=1,2,\ldots,J}$$

From (i) and (ii): by using definition of expected value:

$$_{att}h^j_X = N \int v\tilde{p}^j_X(v)dv\ \forall_{j=1,2,\ldots,J}$$

Hence, $_{att}h^j_X = N \times \mathbb{E}[V^j]\ \forall_{j=1,2,\ldots,J}$ where $V^j \sim \tilde{p}^j_X$. ■

Similar to Proposition 2, Proposition 3 proves that given a feature matrix, representation obtained by the 'attention' pooling filter (given attention weights are accessible) can be fully recovered from representation obtained by the 'distribution' pooling filter. Propositions 2 and 3 show that the representation obtained by the 'distribution' pooling filter already contains the information inside the representations obtained by the 'mean' and 'attention' pooling filters. However, the opposite is not true since the representation obtained by the 'distribution' pooling filter cannot be recovered from the representations obtained by point estimate-based pooling filters.

Hence, we conclude that distribution-based pooling filters are more expressive than point estimate-based pooling filters in terms of the amount of information contained inside bag-level representations. Note that bag-level representations containing more information help improve the model's performance.

6. Experimental analysis of MIL pooling filters

This study analyzed performances of MIL models with different pooling filters in distinct tasks: (i) five MIL tasks formulated on a digital histopathology image patches dataset, (ii) cancer vs. normal whole slide image classification, (iii) MNIST-bags classification, and (iv) positive vs. negative bag classification on classical MIL datasets.

6.1. MIL tasks on a histopathology image patches dataset

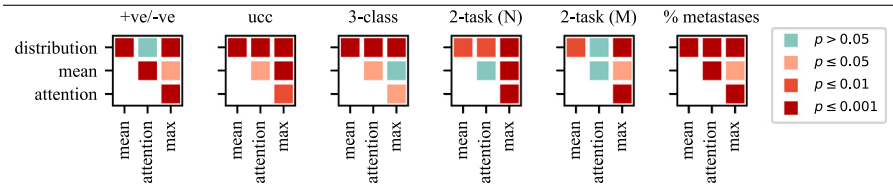
This experiment investigates the effect of MIL pooling filters on the performance of a MIL model in a real-world MIL task. We designed a neural network-based MIL framework with the same structure in Section 3. We used ResNet18 (He et al., 2016) architecture without batch normalization as feature extractor module, θ_{feature}, and a three-layer multi-layer-perceptron as bag-level representation transformation module, $\theta_{\text{transform}}$. We tested this framework with four different MIL pooling filters as θ_{filter} modules on five different MIL tasks formulated on the lymph node metastases dataset. Hence, we had 20 different models sharing the same architecture in the θ_{feature} module. Note that the code was made publicly available at: https://github.com/onermustafaumit/mil_pooling_filters.

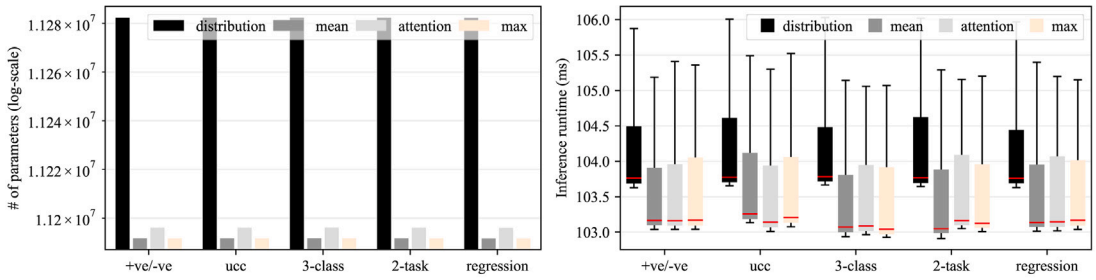
The lymph node metastases dataset is adapted from Oner et al. (2020) and has the training, validation, and test sets (see Appendix A.1 for details). The dataset consists of images cropped from histopathology slides of lymph node sections (Bejnordi et al., 2017), and it has corresponding ground-truth metastases segmentation masks. There are three types of images in this dataset: *fully normal* - all cells are normal, *fully metastases* - all cells are metastases and *boundary* - a mixture of normal and metastases cells. Similar to Section 3.1, we formulated five different MIL tasks on this dataset: positive vs. negative bag classification *(+ve/-ve: predict whether an image contains metastases cells or not)*; unique class count prediction *(ucc: predict how many types of cells exist in an image)*; multi-class classification *(3-class: predict whether an image is fully normal, fully metastases or boundary)*; multi-task classification *(2-task: 1st task — predict whether an image contains normal cells or not, 2nd task — predict whether an image contains metastases cells or not)*; and regression *(% metastases: predict percentage of metastases pixels inside an image)*. The formulated tasks are closely related to the clinical tasks. For example, unique class count prediction can help us obtain segmentation masks without requiring pixel-level annotations from pathologists (Oner et al., 2020), and percent metastases prediction is very similar to tumor purity prediction used in sample selection for genomic sequencing (Oner et al., 2022).

Table 1 summarizes the tasks by bag labels for each kind of image and loss functions used during training. Example images are also shown at the bottom of the table.

Table 2
Top: MIL models' performance on the test set in five different MIL tasks formulated on the lymph node metastases dataset. *Bottom:* Pairwise statistical test results as color-coded maps obtained by thresholding p-values at different significance levels. Best models are highlighted in bold.

	+ve/-ve	ucc	3-class	2-task (*accuracy*)		% metastases
	(*accuracy*)	(*accuracy*)	(*accuracy*)	normal (N)	metastases (M)	(*absolute error*)
distribution	**0.8995**	**0.8832**	**0.7840**	**0.9144**	**0.8696**	**0.1558**
mean	0.8139	0.6413	0.6780	0.8913	0.8438	0.2426
attention	**0.8804**	0.6957	0.7188	0.8927	**0.8614**	0.3264
max	0.7636	0.7582	0.6712	0.8356	0.8111	0.2223





(a) The number of trainable parameters (b) Inference runtime

Fig. 3. (a) The number of trainable parameters and (b) inference runtime for models with different pooling filters in distinct MIL tasks. Different colors represent the pooling filter used in the models. In the box plots, whiskers show 5th and 95th percentiles, and red lines show median values.

6.1.1. Performance analysis

On each task, we trained four different models with MIL pooling filters defined in Section 3.2, namely 'max', 'mean', 'attention', and 'distribution' pooling. Each model was randomly initialized and trained end-to-end with early-stopping criteria on validation set performance. Once we obtained the best models, we checked the models' performance on the hold-out test set (see Appendix A.2 for details). The results are summarized in Table 2 together with used performance metrics for each task. Note that we presented the performances of two tasks separately for multi-task setup.

Furthermore, we have conducted statistical tests for comparing the performance of different models in each task. For classification tasks, we used McNemar's test (Everitt, 1977) since all the models were trained on the same training set and tested on the same hold-out test set as suggested in Dietterich (1998). On the other hand, we used paired t-test (Hsu and Lachenbruch, 2005) on the absolute error values obtained for each sample in the test set to compare models in the regression task. Results of the pairwise statistical tests for each task are also presented at the bottom of Table 2 as color-coded maps obtained by thresholding p-values at different significance levels.

We observed that consistent with our theoretical analysis, models with the 'distribution' pooling filter perform the best in all MIL tasks except positive vs. negative bag classification and metastases task of multi-task classification. Models with the 'distribution' and 'attention' pooling filters perform on par in these tasks. Furthermore, we noticed that the performances of models with different point estimate-based pooling filters are different. Among the models with point estimate-based pooling filters, while the models with the 'max' pooling filter give the best results in the unique class count classification and regression tasks, the models with the 'attention' pooling filter perform the best in the others.

In short, results in our experimental analysis show that models with the 'distribution' pooling filter outperform the models with point estimate-based pooling filters. Moreover, these results are in accordance with our propositions in the theoretical analysis part and support them.

6.1.2. The number of parameters and runtime analysis

We compared the number of trainable parameters in models with different pooling filters in distinct MIL tasks (Fig. 3(a) and Appendix A.3). The models with a point estimate-based pooling filter and a distribution pooling filter, respectively, had $\approx 1.119 \times 10^7$ and $\approx 1.128 \times 10^7$ parameters. The majority of the parameters ($\approx 1.118 \times 10^7$) were in the feature extractor module, common in all models. The difference ($\approx 9.0 \times 10^4$) was due to additional parameters in the representation transformation module ($\approx 8.1 \times 10^4$) and the distribution pooling filter ($\approx 9.0 \times 10^3$). However, this difference was less than 1% of the total number of parameters. Note that attention pooling also has trainable parameters ($\approx 4.5 \times 10^3$).

Similarly, we compared the inference runtime of the models. For each model, we collected the inference runtime for 500 batches with 32 bags. Fig. 3(b) presents the results as box plots. The models with a distribution pooling filter took slightly longer (≈ 0.5 ms) to complete inference since the networks were slightly bigger. Nevertheless, the difference was less than 0.5% of the median runtime.

Hence, distribution-based pooling filters incurred additional memory and computation costs, which were negligible compared to overall network capacity and runtime. On the other hand, distribution-based pooling filters provided MIL models with a significant performance improvement in almost all MIL tasks.

6.2. CAMELYON16 WSI classification

A histopathology image, i.e., a whole slide image, is a gigapixel image that patch-based deep learning models cannot process. Besides, most WSIs do not have pixel-level annotations required by patch-based models. On the other hand, weak labels indicating coarse-level (slide-level or sample-level) properties can easily be obtained from different sources, like pathology reports and electronic health records. Recently, the importance of exploiting weak labels in cancer research have been recognized, and MIL-based deep learning models have been

Table 3
WSI classification performance on the CAMELYON16 test set. The area under the receiver operating characteristics curve (AUROC) is used as the performance metric and presented as *mean* ± *std* obtained over multiple runs for each method. Other methods' performance values are collected from (Zhang et al., 2022). We get FLOPS and model size for our model using the flop counting tool of the fvcore library (MetaResearch, 2023). *The top* accommodates the performance of models with basic MIL pooling filters used for obtaining bag-level representations. *The bottom* accommodates the performance of MIL methods that devise some techniques to improve instance-level or bag-level representations rather than new MIL pooling filters. They mostly use 'attention-based' pooling. However, they can also benefit from distribution pooling.

Method	AUROC	FLOPS	Model Size
Max	0.854 ± 0.038	62.4 M	524.3 K
Mean	0.528 ± 0.010	62.4 M	524.3 K
Attention (Ilse et al., 2018)	0.854 ± 0.006	78.1 M	655.3 K
Distribution (ours)	**0.901 ± 0.031**	5.0 M	164.9 K
RNN-MIL (Campanella et al., 2019)	0.875 ± 0.002	64.0 M	1607.7 K
DS-MIL (Li et al., 2021)	0.899 ± 0.009	117.6 M	855.7 K
CLAM (Lu et al., 2021)	0.871 ± 0.015	94.8 M	790.7 K
Trans-MIL (Shao et al., 2021)	0.906 ± 0.031	613.8 M	2723.8 K
DTFD-MIL (Zhang et al., 2022)	0.946 ± 0.005	79.4 M	986.7 K

devised (Campanella et al., 2019; Lu et al., 2021; Oner et al., 2022). This study analyzes the effects of pooling filters on the performance of MIL models in WSI classification.

We used publicly available CAMELYON16 dataset consisting of histopathology slides of lymph node sections (Bejnordi et al., 2017). Each slide has a slide-level label of either normal (N) (i.e., contains no cancer cells) or tumor (T) (i.e., contains cancer cells). There are 269 (N:158, T:111) and 129 (N:80, T:49) WSIs in the training and test sets, respectively. This is a challenging dataset since cancerous regions in a tumor slide is usually less than 10% of the tissue area, i.e., a bag with few positive instances in the MIL setup.

We represented a WSI as a bag of feature vectors of the slide's patches and use the slide's label as the bag label, similar to other studies (Lu et al., 2021; Shao et al., 2021; Zhang et al., 2022). We used feature vectors extracted by Zhang et al. (2022) using a ResNet50 (He et al., 2016) model trained on the ImageNet. We obtained our training (N:142, T:99) and validation (N:16, T:12) sets by segregating the CAMELYON16 training set slides into two with ratios of ∼90% and ∼10%, respectively. We trained the model on our training set with an early stopping criteria based on the area under the receiver operating characteristics curve (AUROC) on our validation set. Then, we tested the model on the CAMELYON16 test set slides. The model's architecture and the hyper-parameters are presented in Table B.12.

We ran the experiments five times to collect reliable statistics on the model's performance in WSI classification. We obtained the maximum AUROC value of 0.9325 (95% confidence interval: 0.8798–0.9743) on the test set (see Table B.13 for details). We also presented summary statistics of our model together with other MIL models in Table 3. The top part of the table shows that our MIL model with a distribution pooling filter outperforms MIL models with other point estimate-based pooling filters. The bottom part of the table presents the performance of models devising new techniques for obtaining better instance-level or bag-level representations rather than focusing on pooling filters. For example, DTFD-MIL (Zhang et al., 2022) assembles bag of pseudo-bags for each WSI using two attention-based models, and DS-MIL (Li et al., 2021) uses self-supervised contrastive learning to obtain a better feature extractor. Although our model with distribution pooling did not employ any of these techniques, it performed better or on par with these models (except the DTFD-MIL model). Moreover, most of these models use some 'attention-based' pooling. Based on the superior performance of MIL models with a distribution pooling filter, these models can further benefit from using distribution pooling instead of attention-based pooling.

Furthermore, we measured the memory (model size) and computation (FLOPS) requirements of our model using the flop counting tool of the fvcore library (MetaResearch, 2023). Table 3 shows that our MIL model requires less memory and computation resources compared to the other models. This provides us with faster run times and ability to work with limited computational resources.

6.3. MNIST-bags classification

This section analyzes the effects of the number of instances in a bag and the number of bags in a training dataset on the performance of MIL models with a distribution pooling filter. We used the MNIST-bags classification task of Ilse et al. (2018), in which a bag consists of images in the MNIST hand-written digits dataset. If a bag contains at least one image of digit-9, the bag's label is positive. Otherwise, it is negative. The number of images in a bag was sampled (and rounded to the closest integer) from a Gaussian distribution ($G(\mu, \sigma)$), where μ values were 10, 50, and 100 with corresponding σ values of 2, 10, and 20. The number of bags in a training dataset was 50, 100, 150, 200, 300, 400, and 500. All the training bags were created using the MNIST training set images. For performance evaluation, 1000 bags created using the MNIST test set images.

To benchmark our model's performance with the models in Ilse et al. (2018), we used the same architecture, namely LeNet5 model (LeCun et al., 1998), in the feature extractor module. We trained our MIL models with a distribution pooling filter on the training set bags and evaluated on the test set bags. Model architecture details and hyper-parameters are presented in Table C.14 and C.15. We used AUROC values on the test set as our performance metric and presented the results obtained from five runs of each model in Table 4. Our models with a distribution pooling filter outperformed the ones with point estimate-based pooling filters especially in the cases of few training bags with few instances, showing the ability of our model capturing the necessary information in few samples. In the cases of many training bags with many instances, the performance of the models with different pooling filters approximate to each other in this simple task.

6.4. Bag classification on classical MIL datasets

The performance of our neural network model with the 'distribution' pooling filter (Distribution-Net) is compared with the performance of the best MIL methods in positive vs. negative bag classification task on five classical MIL datasets: drug activity prediction datasets MUSK1 and MUSK2 (Dietterich et al., 1997) and animal image annotation datasets FOX, TIGER and ELEPHANT (Andrews et al., 2003) (see Appendix D for details).

We used 10-fold cross-validation and repeated each experiment 5 times. For each dataset, we have declared the mean of classification accuracies (± standard error). We compared the performance of Distribution-Net with the performance of state-of-the-art MIL methods on classical MIL datasets in Table 5. While the first part of the table contains methods utilizing traditional machine learning techniques (Andrews et al., 2003; Gärtner et al., 2002; Zhang and Goldman, 2002; Zhou et al., 2009; Wei et al., 2016), the second part of the table accommodates methods employing neural networks (Wang et al., 2018; Ilse et al., 2018). The last part of the table shows the performance of our Distribution-Net, which outperformed all other methods on all datasets. Neural network-based models generally outperformed the traditional machine learning-based models. Furthermore, our Distribution-Net performed even better than other neural network-based models (Wang et al., 2018; Ilse et al., 2018). The notable difference between the models was the pooling filters. While other models employed point estimate-based pooling filters (Wang et al., 2018; Ilse et al., 2018), Distribution-Net used the 'distribution' pooling filter. The reason for the improvement seems that Distribution-Net utilized the full distribution information captured by the 'distribution' pooling filter over other models.

Table 4
MNIST-bags classification. The performance of MIL models with different pooling filters are presented. The performance metric is the area under the receiver operating characteristics curve and presented as *mean* ± *std* obtained over five runs. Each model is tested on 1000 bags created using the MNIST test set images. The number of instances (images) in each bag are sampled (and rounded to the closest integer) from a Gaussian distribution ($G(\mu, \sigma)$), where μ values are 10, 50, and 100 with corresponding σ values of 2, 10, and 20. During training of the models, different number of bags (50, 100, 150, 200, 300, 400, and 500) created using the MNIST training set images are used. Please note that the performance values for MIL models with point estimate-based pooling filters are taken from Ilse et al. (2018).

# of instances	Pooling	# of bags used in training						
		50	100	150	200	300	400	500
10 per bag (on average)	max	0.713 ± 0.016	0.914 ± 0.011	0.954 ± 0.005	0.968 ± 0.001	0.980 ± 0.001	0.981 ± 0.003	0.986 ± 0.002
	mean	0.695 ± 0.026	0.841 ± 0.027	0.926 ± 0.004	0.953 ± 0.004	0.974 ± 0.002	0.980 ± 0.001	0.984 ± 0.002
	attention	0.768 ± 0.054	0.948 ± 0.007	0.949 ± 0.006	0.970 ± 0.003	0.980 ± 0.000	0.982 ± 0.001	0.986 ± 0.001
	distribution	0.902 ± 0.015	0.955 ± 0.013	0.965 ± 0.007	0.970 ± 0.005	0.976 ± 0.004	0.976 ± 0.007	0.982 ± 0.006
50 per bag (on average)	max	0.872 ± 0.039	0.984 ± 0.005	0.992 ± 0.001	0.996 ± 0.001	0.996 ± 0.001	0.997 ± 0.001	0.997 ± 0.001
	mean	0.841 ± 0.013	0.906 ± 0.046	0.983 ± 0.005	0.992 ± 0.001	0.996 ± 0.001	0.997 ± 0.001	0.997 ± 0.001
	attention	0.967 ± 0.010	0.982 ± 0.003	0.990 ± 0.002	0.993 ± 0.002	0.989 ± 0.003	0.994 ± 0.001	0.995 ± 0.001
	distribution	0.985 ± 0.004	0.987 ± 0.004	0.990 ± 0.002	0.990 ± 0.002	0.991 ± 0.003	0.989 ± 0.002	0.991 ± 0.002
100 per bag (on average)	max	0.977 ± 0.009	0.999 ± 0.001	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
	mean	0.959 ± 0.010	0.990 ± 0.003	0.998 ± 0.001	0.900 ± 0.089	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
	attention	0.996 ± 0.001	0.998 ± 0.001	0.999 ± 0.000	0.998 ± 0.001	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000
	distribution	0.997 ± 0.000	0.999 ± 0.001	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000	1.000 ± 0.000

Table 5
Performances of different MIL methods on classical MIL datasets. First part: methods utilizing traditional machine learning techniques. Second part: methods employing neural networks. Last part: our model with the 'distribution' pooling filter (Distribution-Net).

METHOD	MUSK1	MUSK2	FOX	TIGER	ELEPHANT
mi-SVM (Andrews et al., 2003)	0.874 ± N/A	0.836 ± N/A	0.582 ± N/A	0.784 ± N/A	0.822 ± N/A
MI-SVM (Andrews et al., 2003)	0.779 ± N/A	0.843 ± N/A	0.578 ± N/A	0.840 ± N/A	0.843 ± N/A
MI-Kernel (Gärtner et al., 2002)	0.880 ± 0.031	0.893 ± 0.015	0.603 ± 0.028	0.842 ± 0.010	0.843 ± 0.016
EM-DD (Zhang and Goldman, 2002)	0.849 ± 0.044	0.869 ± 0.048	0.609 ± 0.045	0.730 ± 0.043	0.771 ± 0.043
mi-Graph (Zhou et al., 2009)	0.889 ± 0.033	0.903 ± 0.039	0.620 ± 0.044	0.860 ± 0.037	0.869 ± 0.035
miVLAD (Wei et al., 2016)	0.871 ± 0.043	0.872 ± 0.042	0.620 ± 0.044	0.811 ± 0.039	0.850 ± 0.036
miFV (Wei et al., 2016)	0.909 ± 0.040	0.884 ± 0.042	0.621 ± 0.049	0.813 ± 0.037	0.852 ± 0.036
mi-Net (Wang et al., 2018)	0.889 ± 0.039	0.858 ± 0.049	0.613 ± 0.035	0.824 ± 0.034	0.858 ± 0.037
MI-Net (Wang et al., 2018)	0.887 ± 0.041	0.859 ± 0.046	0.622 ± 0.038	0.830 ± 0.032	0.862 ± 0.034
MI-Net with DS (Wang et al., 2018)	0.894 ± 0.042	0.874 ± 0.043	0.630 ± 0.037	0.845 ± 0.039	0.872 ± 0.032
MI-Net with RC (Wang et al., 2018)	0.898 ± 0.043	0.873 ± 0.044	0.619 ± 0.047	0.836 ± 0.037	0.857 ± 0.040
Attention (Ilse et al., 2018)	0.892 ± 0.040	0.858 ± 0.048	0.615 ± 0.043	0.839 ± 0.022	0.868 ± 0.022
Gated-Attention (Ilse et al., 2018)	0.900 ± 0.050	0.863 ± 0.042	0.603 ± 0.029	0.845 ± 0.018	0.857 ± 0.027
Distribution-Net (ours)	**0.923** ± 0.071	**0.932** ± 0.067	**0.680** ± 0.075	**0.864** ± 0.054	**0.900** ± 0.077

Table A.6
Lymph node metastases dataset — The number of images in training, validation, and test sets.

	Fully normal	Fully metastases	Boundary	Total
Training	395	228	310	933
Validation	267	190	211	668
Test	277	231	228	736

Table A.7
Experiments on lymph node metastases dataset — architecture and list of hyper-parameters used in the MIL models.

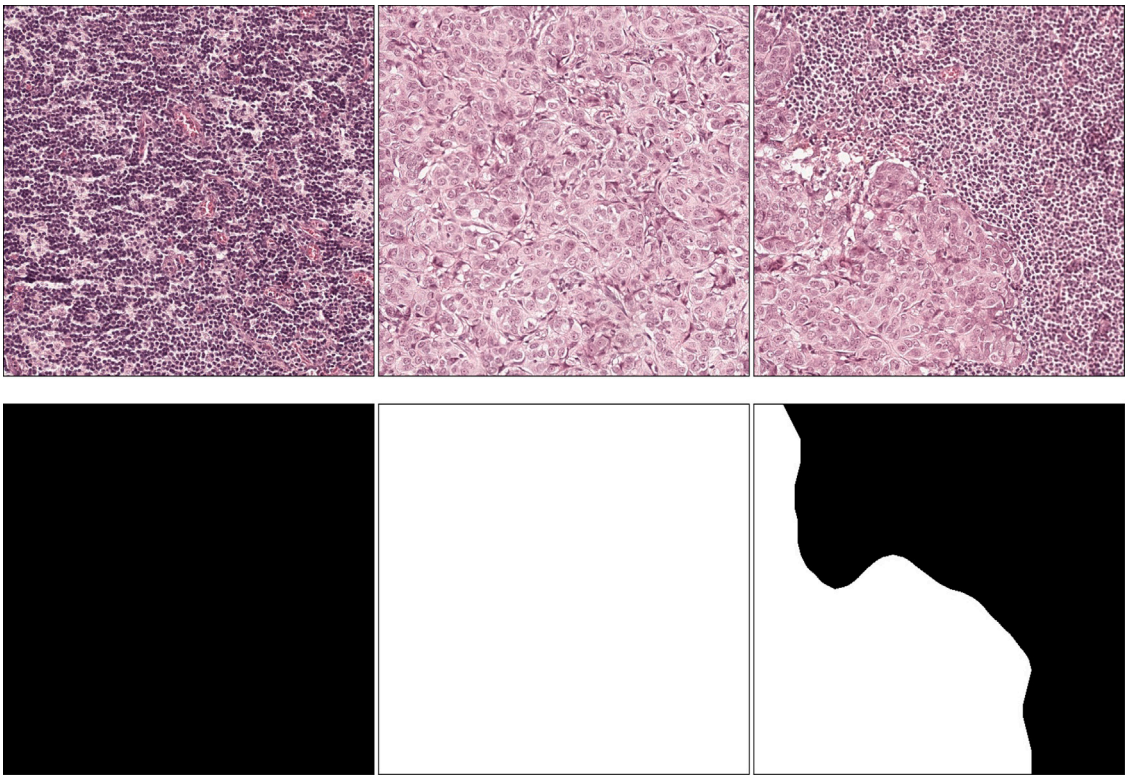
Architecture	input-32 × 32x3 ResNet18 w/o BN 'distribution'/'mean'/'attention'/'max' pooling Dropout(0.5) fc-128 + ReLU Dropout(0.5) fc-32 + ReLU Dropout(0.5) fc-2 *(+ve/-ve, ucc, 2-task)*/fc-3 *(3-class)*/fc-1 *(regression)* softmax *(+ve/-ve, ucc, 3-class)*/sigmoid *(2-task)*/None *(regression)*
image size	512 × 512
patch size	32 × 32
# instances per bag	64
# features	32
# bins in 'distribution' filters	21
σ in Gaussian kernel	0.0167
Optimizer	ADAM
Learning rate	$1e-4$
$L2$ regularization weight decay	0.0005
batch size	32

7. Conclusion

This paper introduced the family of distribution-based pooling filters that obtains a bag-level representation by estimating marginal feature distributions from extracted features of instances inside a bag. We formally proved that distribution-based pooling filters are more expressive than point estimate-based counterparts in terms of the amount of information captured while obtaining bag-level representations. This property of distribution-based pooling filters is of great importance since more information can improve the model's performance.

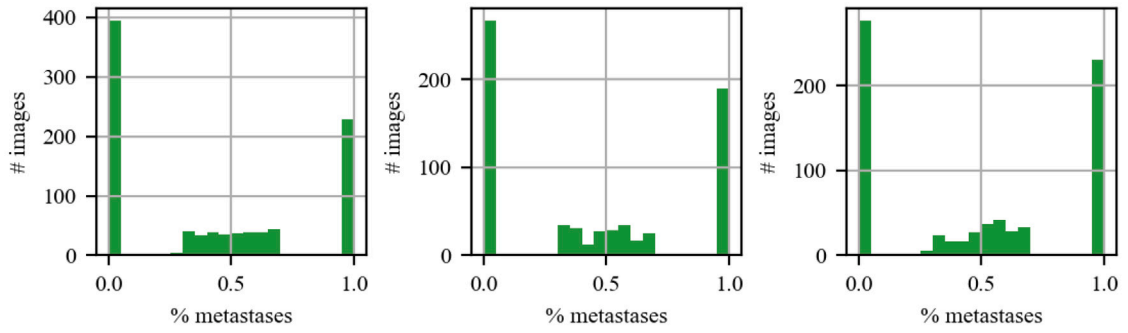
Furthermore, we extensively analyzed the performance of a MIL model with a distribution pooling filter in different MIL tasks on the lymph node metastases dataset. Models with the 'distribution' pooling filter were among the best-performing ones in all tasks. Similarly, our model with the 'distribution' pooling filter outperformed MIL models with different MIL pooling filters in the bag classification tasks on the MNIST-bags dataset and classical MIL datasets. Hence, the results were per our theoretical findings.

Besides, we analyzed the memory and computation costs of using different filters. When we had the same number of instance features, the distribution-based pooling filters incurred additional memory and computation costs, which were negligible compared to overall network capacity and runtime (Fig. 3). On the other hand, they provided MIL models with a significant performance increase in almost all MIL tasks.



(a) Fully normal (b) Fully metastases (c) Boundary

Fig. A.4. Lymph node metastases dataset — Examples of three types of images and corresponding ground-truth masks: (a) *fully normal* — all cells are normal, (b) *fully metastases* — all cells are metastases, and (c) *boundary* — a mixture of normal and metastases cells.



(a) Training (b) Validation (c) Test

Fig. A.5. Percent metastases histograms for training, validation and test sets.

We also observed that when we had fewer instance features in MIL models with distribution pooling filters, they achieved better performance than the models with point estimate-based filters and provided significant memory and computation savings (Table 3).

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The study used publicly available data and the code was made publicly available at: https://github.com/onermustafaumit/mil_pooling_filters.

Acknowledgment

This work is partly supported by the Biomedical Research Council of the Agency for Science, Technology, and Research, Singapore.

Appendix A. Experiments on lymph node metastases dataset

We investigated the effect of MIL pooling filters on the performance of a MIL model in a particular real-world MIL task. We designed a neural network-based MIL framework and analyzed the performance of our framework with four different MIL pooling filters in five distinct MIL tasks formulated on a real-world lymph node metastases dataset.

Code for the experiments is publicly available at: https://github.com/onermustafaumit/mil_pooling_filters

Table A.8
The total number of trainable parameters in each model.

	MIL tasks				
	+ve/-ve	ucc	3-class	2-task	Regression
distribution	11,282,372	11,282,372	11,282,405	11,282,372	11,282,339
mean	11,191,746	11,191,746	11,191,779	11,191,746	11,191,713
attention	11,196,099	11,196,099	11,196,132	11,196,099	11,196,066
max	11,191,746	11,191,746	11,191,779	11,191,746	11,191,713

Table A.9
The number of trainable parameters in θ_{feature}.

	MIL tasks				
	+ve/-ve	ucc	3-class	2-task	Regression
distribution	11,183,328	11,183,328	11,183,328	11,183,328	11,183,328
mean	11,183,328	11,183,328	11,183,328	11,183,328	11,183,328
attention	11,183,328	11,183,328	11,183,328	11,183,328	11,183,328
max	11,183,328	11,183,328	11,183,328	11,183,328	11,183,328

Table A.10
The number of trainable parameters in θ_{filter}.

	MIL tasks				
	+ve/-ve	ucc	3-class	2-task	Regression
distribution	8,706	8,706	8,706	8,706	8,706
mean	0	0	0	0	0
attention	4,353	4,353	4,353	4,353	4,353
max	0	0	0	0	0

Table A.11
The number of trainable parameters in $\theta_{\text{transform}}$.

	MIL tasks				
	+ve/-ve	ucc	3-class	2-task	Regression
distribution	90,338	90,338	90,371	90,338	90,305
mean	8,418	8,418	8,451	8,418	8,385
attention	8,418	8,418	8,451	8,418	8,385
max	8,418	8,418	8,451	8,418	8,385

Table B.12
CAMELYON16 WSI classification — architecture and list of hyper-parameters used in the MIL model.

Architecture	input-1024 fc-32 + ReLU 'distribution' pooling fc-2 softmax
# instances per bag	20% (train)/100% (test) of a WSI's patches
# features	32
# bins in 'distribution' filters	11
σ in Gaussian kernel	0.033
Optimizer	ADAM
Learning rate	0.0001
$L2$ regularization weight decay	0.0005
Batch size	1

A.1. Lymph node metastases dataset

The lymph node metastases dataset is adapted from Oner et al. (2020). The original dataset consists of 512 × 512 images cropped from histopathology slides of lymph node sections (Bejnordi et al., 2017), and it has corresponding ground-truth metastases segmentation masks. There are three types of images in this dataset: *fully normal* - all cells are normal, *fully metastases* - all cells are metastases and *boundary* - a mixture of normal and metastases cells. Fig. A.4 shows example images of each type. To make a clear distinction between these three types of images, we filtered out the images with (i) $0 <$ percent metastases ≤ 20 and (ii) $80 \leq$ percent metastases < 100. Moreover, to obtain a balanced dataset, we dropped some of the images in the test set coming from one specific slide.

The dataset is publicly available. The number of images and percent metastases histograms in training, validation, and test sets are given in Table A.6 and Fig. A.5, respectively.

A.2. Neural network architectures and hyper-parameters

We designed a neural network-based MIL framework. We used ResNet18 (He et al., 2016) architecture without batch normalization as feature extractor module, θ_{feature}, and a three-layer multi-layer-perceptron as bag-level representation transformation module, $\theta_{\text{transform}}$. We tested this framework with four different MIL pooling filters as θ_{filter} modules on five distinct MIL tasks formulated. Hence, we had 20 different models sharing the same architecture in the θ_{feature} module. Moreover, all models have the same hidden layers in the $\theta_{\text{transform}}$ module; however, note that the number of input nodes in the $\theta_{\text{transform}}$ module depends on θ_{filter}, and the number of output nodes in the $\theta_{\text{transform}}$ module depends on the MIL task. Please refer to the provided code for more details.

Table B.13
CAMELYON16 WSI classification — detailed performance metrics over multiple runs. AUROC: Area under receiver operating characteristics curve. 95% confidence intervals (CI) are constructed using the percentile bootstrap method (Efron, 1992).

#	Accuracy	Precision	Recall	F1-score	AUROC (95% CI)
0	0.7402	0.6230	0.7917	0.6972	0.8674 (0.7933 - 0.9283)
1	0.8031	0.7347	0.7500	0.7423	0.8601 (0.7785 - 0.9293)
2	0.8583	0.8000	0.8333	0.8163	0.9325 (0.8798 - 0.9743)
3	0.7953	0.6774	0.8750	0.7636	0.9227 (0.8724 - 0.9623)
4	0.8976	0.9268	0.7917	0.8539	0.9233 (0.8656 - 0.9704)

Table C.14
MNIST bags — architecture and list of hyper-parameters used in the MIL model.

Architecture	input-28 × 28 conv(5,1,0)-20 + ReLU maxpool(2,2) conv(5,1,0)-50 + ReLU maxpool(2,2) fc-64 + Sigmoid 'distribution' pooling fc-2 softmax
# instances per bag	all patches in a bag
# features	64
# bins in 'distribution' filters	11
σ in Gaussian kernel	0.033
Optimizer	ADAM
Learning rate	0.0003
$L2$ regularization weight decay	0.0001
Batch size	1
Epochs (max)	200
Stopping criteria	lowest validation error+loss

Table C.15
MNIST bags — architecture and list of hyper-parameters used in the MIL model of Ilse et al. (2018).

Architecture	input-28 × 28 conv(5,1,0)-20 + ReLU maxpool(2,2) conv(5,1,0)-50 + ReLU maxpool(2,2) fc-500 + ReLU 'max', 'mean', 'attention' pooling fc-1 + Sigmoid
Optimizer	ADAM
Learning rate	0.0005
$L2$ regularization weight decay	0.0001
Batch size	1
Epochs (max)	200
Stopping criteria	lowest validation error+loss

During training, each image was treated as a bag. We prepared bags on the fly during training by randomly cropping 32 × 32 patches over the images. Each bag was created with 64 cropped patches (instances), and data augmentation was applied to the cropped patches. We used a batch size of 32 and extracted 32 features for each instance inside a bag. The 'distribution' pooling filter estimated marginal feature distributions using kernel density estimation with a Gaussian kernel. The Gaussian kernel's standard deviation was $\sigma = 0.0167$, and the estimated distributions were binned into 21 bins.

Furthermore, attention weights w_i in 'attention' pooling and α_i and β_i in 'distribution' pooling were obtained from attention models $\mathcal{W}$, $\mathcal{W}_\alpha$, and $\mathcal{W}_\beta$, respectively. We used the same architecture in Ilse et al. (2018) in the attention models except for $\mathcal{W}_\alpha$, which had a *sigmoid* activation function at the last layer rather than *softmax*. We trained models using the ADAM optimizer with a learning rate of $1e-4$ and $L2$ regularization on the weights with a weight decay of 0.0005. Each model was randomly initialized and trained end-to-end with early-stopping criteria on validation set performance. Table A.7 presents the architecture and list of hyper-parameters used in MIL models.

During testing, we created 100 bags for each image in the test set and tested them with the trained model. The final prediction was obtained by averaging 100 predictions.

A.3. The number of parameters

The MIL models consist of three modules: θ_{feature}, θ_{filter}, and $\theta_{\text{transform}}$. The feature extractor module θ_{feature} and representation transformation module $\theta_{\text{transform}}$ are implemented as neural networks. Some MIL pooling filters, namely 'distribution' and 'attention' pooling filters, also contain neural networks.

Table A.8 shows the total number of trainable parameters in each model. Similarly, Table A.9, Table A.10, and Table A.11 show the number of trainable parameters in the θ_{feature}, θ_{filter}, and $\theta_{\text{transform}}$ modules for each model, respectively.

Appendix B. Experiments on CAMELYON16

See Tables B.12 and B.13.

Appendix C. Experiments on MNIST-bags

See Tables C.14 and C.15.

Appendix D. Experiments on classical MIL datasets

This section compares the performance of our neural network model with 'distribution' pooling filter (Distribution-Net) with the performance of the best MIL methods on classical MIL task of positive vs. negative bag classification on five classical MIL datasets: drug activity prediction datasets MUSK1 and MUSK2 (Dietterich et al., 1997) and animal image annotation datasets FOX, TIGER and ELEPHANT (Andrews et al., 2003). Attention weights in the 'distribution' pooling filter were fixed to $\alpha_i = 1\ \forall_i$ and $\beta_i = \frac{1}{N}\ \forall_i$, where N is the number of instances per bag. Table D.16 summarizes the classical MIL datasets.

The summary of architectures and hyper-parameters used in MIL models on 'MUSK' and 'Animal' datasets are given in Table D.17 and Table D.18, respectively. We used mini-batch training with bags that include an equal number of instances. We created bags by sampling from available instances of each sample (a drug with multiple conformations for 'MUSK' datasets and an image with multiple segments in 'Animal' datasets). When the number of available instances of a sample is less than the number of instances required to create a bag, we used available instances more than once in a bag. We have determined the number of instances with cross-validation on the validation sets.

Table D.19 summarizes the architectures and hyper-parameters used in MIL models of Wang et al. (2018), Ilse et al. (2018) on 'MUSK' and 'Animal' datasets.

Table D.16
Summary of classical MIL datasets.

	# bags			# instances per bag			
	Positive	Negative	Total	Min	Max	Average	# features
MUSK1	47	45	92	2	40	5.17	166
MUSK2	39	63	102	1	1044	64.69	166
FOX	100	100	200	2	13	6.6	230
TIGER	100	100	200	1	13	6.1	230
ELEPHANT	100	100	200	2	13	6.96	230

Table D.17
MUSK datasets — architecture and list of hyper-parameters used in the MIL models.

Architecture	input-166 fc-64 + ReLU Dropout(0.5) fc-32 + ReLU Dropout(0.5) fc-32 + Sigmoid 'distribution' pooling Dropout(0.5) fc-64 + ReLU Dropout(0.5) fc-32 + ReLU Dropout(0.5) fc-2 softmax
# instances per bag	16
# features	32
# bins in 'distribution' pooling filters	11
σ in Gaussian kernel	0.1
Optimizer	ADAM
Learning rate	$5e-4$
$L2$ regularization weight decay	0.1
batch size	8

Table D.18
Animal datasets — architecture and list of hyper-parameters used in the MIL models.

Architecture	input-230 fc-256 + ReLU Dropout(0.5) fc-128 + ReLU Dropout(0.5) fc-64 + ReLU Dropout(0.5) fc-32 + Sigmoid 'distribution' pooling Dropout(0.5) fc-384 + ReLU Dropout(0.5) fc-192 + ReLU Dropout(0.5) fc-2 softmax
# instances per bag	16
# features	32
# bins in 'distribution' pooling filters	11
σ in Gaussian kernel	0.1
Optimizer	ADAM
Learning rate	$5e-6$
$L2$ regularization weight decay	0.1
batch size	8

Table D.19
MUSK and Animal datasets — architecture and list of hyper-parameters used in the MIL models of Wang et al. (2018), Ilse et al. (2018).

Architecture	input-166 fc-256 + ReLU Dropout fc-128 + ReLU Dropout fc-64 + ReLU Dropout 'max', 'mean', 'attention' pooling fc-1 + Sigmoid
Optimizer	SGD
Learning rate	$5e-4$ (MUSK1, MUSK2, Fox)/$1e-4$ (Tiger, Elephant)
Momentum	0.9
$L2$ regularization weight decay	0.03 (MUSK2)/0.01 (Tiger)/0.005 (MUSK1, Fox, Elephant)
batch size	1

References

Andrews, S., Tsochantaridis, I., Hofmann, T., 2003. Support vector machines for multiple-instance learning. In: Advances in Neural Information Processing Systems. pp. 577–584.

Bejnordi, B.E., Veta, M., Van Diest, P.J., Van Ginneken, B., Karssemeijer, N., Litjens, G., Van Der Laak, J.A., Hermsen, M., Manson, Q.F., Balkenhol, M., et al., 2017. Diagnostic assessment of deep learning algorithms for detection of lymph node metastases in women with breast cancer. JAMA 318 (22), 2199–2210.

Buhmann, M., 2010. Radial basis function. Scholarpedia 5 (5), 9837.

Campanella, G., Hanna, M.G., Geneslaw, L., Miraflor, A., Silva, V.W.K., Busam, K.J., Brogi, E., Reuter, V.E., Klimstra, D.S., Fuchs, T.J., 2019. Clinical-grade computational pathology using weakly supervised deep learning on whole slide images. Nat. Med. 25 (8), 1301–1309.

Chen, Y., Bi, J., Wang, J.Z., 2006. MILES: Multiple-instance learning via embedded instance selection. IEEE Trans. Pattern Anal. Mach. Intell. 28 (12), 1931–1947.

Chen, Y., Wang, J.Z., 2004. Image categorization by learning and reasoning with regions. J. Mach. Learn. Res. 5 (Aug), 913–939.

Chikontwe, P., Kim, M., Nam, S.J., Go, H., Park, S.H., 2020. Multiple instance learning with center embeddings for histopathology classification. In: Medical Image Computing and Computer Assisted Intervention–MICCAI 2020: 23rd International Conference, Lima, Peru, October 4–8, 2020, Proceedings, Part V 23. Springer, pp. 519–528.

Dietterich, T.G., 1998. Approximate statistical tests for comparing supervised classification learning algorithms. Neural Comput. 10 (7), 1895–1923.

Dietterich, T.G., Lathrop, R.H., Lozano-Pérez, T., 1997. Solving the multiple instance problem with axis-parallel rectangles. Artificial Intelligence 89 (1–2), 31–71.

Dundar, M., Krishnapuram, B., Rao, R., Fung, G.M., 2007. Multiple instance learning for computer aided diagnosis. In: Advances in Neural Information Processing Systems. pp. 425–432.

Efron, B., 1992. Bootstrap methods: another look at the jackknife. In: Breakthroughs in Statistics. Springer, pp. 569–593.

Everitt, B., 1977. The Analysis of Contingency Tables. Chapman and Hall.

Feng, J., Zhou, Z.-H., 2017. Deep MIML network. In: Thirty-First AAAI Conference on Artificial Intelligence.

Ferlay, J., Laversanne, M., Ervik, M., Lam, F., Colombet, M., Mery, L., Piñeros, M., Znaor, A., Soerjomataram, I., Bray, F., 2020. Global cancer observatory: Cancer tomorrow. Lyon, France: International Agency for Research on Cancer. Available from: https://gco.iarc.fr/tomorrow, (Accessed: 6 Feb 2023).

Foulds, J.R., 2008. Learning instance weights in multi-instance learning. (Ph.D. thesis). The University of Waikato.

Foulds, J., Frank, E., 2010. A review of multi-instance learning assumptions. Knowl. Eng. Rev. 25 (1), 1–25.
Gärtner, T., Flach, P.A., Kowalczyk, A., Smola, A.J., 2002. Multi-instance kernels. In: ICML. Vol. 2, p. 7.
Hashimoto, N., Fukushima, D., Koga, R., Takagi, Y., Ko, K., Kohno, K., Nakaguro, M., Nakamura, S., Hontani, H., Takeuchi, I., 2020. Multi-scale domain-adversarial multiple-instance CNN for cancer subtype classification with unannotated histopathological images. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3852–3861.
He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 770–778.
Hsu, H., Lachenbruch, P.A., 2005. Paired t test. Encycl. Biostat. 6.
Ilse, M., Tomczak, J., Welling, M., 2018. Attention-based deep multiple instance learning. In: International Conference on Machine Learning. pp. 2127–2136.
LeCun, Y., Bottou, L., Bengio, Y., Haffner, P., et al., 1998. Gradient-based learning applied to document recognition. Proc. IEEE 86 (11), 2278–2324.
Lee, J., Lee, Y., Kim, J., Kosiorek, A., Choi, S., Teh, Y.W., 2019. Set transformer: A framework for attention-based permutation-invariant neural networks. In: International Conference on Machine Learning. pp. 3744–3753.
Li, B., Li, Y., Eliceiri, K.W., 2021. 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. pp. 14318–14328.
Li, W., Vasconcelos, N., 2015. Multiple instance learning for soft bags via top instances. In: Proceedings of the Ieee Conference on Computer Vision and Pattern Recognition. pp. 4277–4285.
Lu, X., Lin, Z., Shen, X., Mech, R., Wang, J.Z., 2015. Deep multi-patch aggregation network for image style, aesthetics, and quality estimation. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 990–998.
Lu, M.Y., Williamson, D.F., Chen, T.Y., Chen, R.J., Barbieri, M., Mahmood, F., 2021. Data-efficient and weakly supervised computational pathology on whole-slide images. Nat. Biomed. Eng. 5 (6), 555–570.
Maron, O., Lozano-Pérez, T., 1998. A framework for multiple-instance learning. In: Advances in Neural Information Processing Systems. pp. 570–576.
MetaResearch, 2023. Fvcore: Flop counter for PyTorch models. URL https://github.com/facebookresearch/fvcore/blob/main/docs/flop_count.md.
Micchelli, C.A., 1986. Interpolation of scattered data: distance matrices and conditionally positive definite functions. Constr. Approx. 2 (1), 11–22.
Myronenko, A., Xu, Z., Yang, D., Roth, H.R., Xu, D., 2021. Accounting for dependencies in deep learning based multiple instance learning for whole slide imaging. In: Medical Image Computing and Computer Assisted Intervention–MICCAI 2021: 24th International Conference, Strasbourg, France, September 27–October 1, 2021, Proceedings, Part VIII 24. Springer, pp. 329–338.
Oner, M.U., Chen, J., Revkov, E., James, A., Heng, S.Y., Kaya, A.N., Alvarez, J.J.S., Takano, A., Cheng, X.M., Lim, T.K.H., et al., 2022. Obtaining spatially resolved tumor purity maps using deep multiple instance learning in a pan-cancer study. Patterns 3 (2), 100399.
Oner, M.U., Lee, H.K., Sung, W.-K., 2020. Weakly supervised clustering by exploiting unique class count. In: International Conference on Learning Representations.
Pappas, N., Popescu-Belis, A., 2017. Explicit document modeling through weighted multiple-instance learning. J. Artificial Intelligence Res. 58, 591–626.
Parzen, E., 1962. On estimation of a probability density function and mode. Ann. Math. Stat. 33 (3), 1065–1076.
PyTorch, 2020a. PyTorch binary cross entropy loss function. URL https://pytorch.org/docs/stable/nn.html#bcewithlogitsloss.
PyTorch, 2020b. PyTorch cross entropy loss function. URL https://pytorch.org/docs/stable/nn.html#crossentropyloss.
Quellec, G., Cazuguel, G., Cochener, B., Lamard, M., 2017. Multiple-instance learning for medical image and video analysis. IEEE Rev. Biomed. Eng. 10, 213–234.
Ramon, J., De Raedt, L., 2000. Multi instance neural networks. In: Proceedings of the ICML-2000 Workshop on Attribute-Value and Relational Learning. pp. 53–60.
Schaback, R., 2007. A practical guide to radial basis functions. Electron. Resour. 11, 1–12.
Schoenberg, I.J., 1938. Metric spaces and positive definite functions. Trans. Amer. Math. Soc. 44 (3), 522–536.
Shao, Z., Bian, H., Chen, Y., Wang, Y., Zhang, J., Ji, X., et al., 2021. Transmil: Transformer based correlated multiple instance learning for whole slide image classification. Adv. Neural Inf. Process. Syst. 34, 2136–2147.
Tang, J., Li, H., Qi, G.-J., Chua, T.-S., 2010. Image annotation by graph-based inference with integrated multiple/single instance representations. IEEE Trans. Multimed. 12 (2), 131–141.
Tomita, N., Abdollahi, B., Wei, J., Ren, B., Suriawinata, A., Hassanpour, S., 2019. Attention-based deep neural networks for detection of cancerous and precancerous esophagus tissue on histopathological slides. JAMA Netw. Open 2 (11), e1914645.
Wang, Y., Li, J., Metze, F., 2019. A comparison of five multiple instance learning pooling functions for sound event detection with weak labeling. In: ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing. ICASSP, IEEE, pp. 31–35.
Wang, X., Yan, Y., Tang, P., Bai, X., Liu, W., 2018. Revisiting multiple instance neural networks. Pattern Recognit. 74, 15–24.
Wei, X.-S., Wu, J., Zhou, Z.-H., 2016. Scalable algorithms for multi-instance learning. IEEE Trans. Neural Netw. Learn. Syst. 28 (4), 975–987.
Wu, J., Yu, Y., Huang, C., Yu, K., 2015. Deep multiple instance learning for image classification and auto-annotation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3460–3469.
Yang, Y., Deng, C., Gao, S., Liu, W., Tao, D., Gao, X., 2016. Discriminative multi-instance multitask learning for 3d action recognition. IEEE Trans. Multimed. 19 (3), 519–529.
Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R.R., Smola, A.J., 2017. Deep sets. In: Advances in Neural Information Processing Systems. pp. 3391–3401.
Zhang, Q., Goldman, S.A., 2002. EM-dd: An improved multiple-instance learning technique. In: Advances in Neural Information Processing Systems. pp. 1073–1080.
Zhang, Q., Goldman, S.A., Yu, W., Fritts, J.E., 2002. Content-based image retrieval using multiple-instance learning. In: ICML. Vol. 1, Citeseer, p. 2.
Zhang, Y., Hare, J., Prügel-Bennett, A., 2020. Fspool: Learning set representations with featurewise sort pooling. In: International Conference on Learning Representations.
Zhang, H., Meng, Y., Zhao, Y., Qiao, Y., Yang, X., Coupland, S.E., Zheng, Y., 2022. 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. pp. 18802–18812.
Zhang, Y., Zhao, R., Dong, W., Hu, B.-G., Ji, Q., 2018. Bilateral ordinal relevance multi-instance regression for facial action unit intensity estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 7034–7043.
Zhang, M.-L., Zhou, Z.-H., 2004. Improve multi-instance neural networks through feature selection. Neural Process. Lett. 19 (1), 1–10.
Zhang, M.-L., Zhou, Z.-H., 2009. Multi-instance clustering with applications to multi-instance prediction. Appl. Intell. 31 (1), 47–68.
Zhou, Z.-H., Sun, Y.-Y., Li, Y.-F., 2009. Multi-instance learning by treating instances as non-iid samples. In: Proceedings of the 26th Annual International Conference on Machine Learning. ACM, pp. 1249–1256.
Zhou, Z.-H., Zhang, M.-L., 2002. Neural networks for multi-instance learning. In: Proceedings of the International Conference on Intelligent Information Technology, Beijing, China. pp. 455–459.
Zhu, W., Lou, Q., Vang, Y.S., Xie, X., 2017. Deep multi-instance networks with sparse label assignment for whole mammogram classification. In: International Conference on Medical Image Computing and Computer-Assisted Intervention. Springer, pp. 603–611.