A Appendix

A.1 Pseudo-Code of DEMIPL

Given an unseen multi-instance bag $\boldsymbol{X}_* = [\boldsymbol{x}_{*,1}, \boldsymbol{x}_{*,2}, \cdots, \boldsymbol{x}_{*,n_*}]$ with n_* instances, DEMIPL initially utilizes the feature extractor to obtain the instance-level representation as follows:

$$\boldsymbol{H}_* = h(\boldsymbol{X}_*) = \{\boldsymbol{h}_{*,1}, \boldsymbol{h}_{*,2}, \cdots, \boldsymbol{h}_{*,n_*}\}. \tag{9}$$

Subsequently, DEMIPL maps the instance-level representation $\boldsymbol{H}_*$ into a vector representation $\boldsymbol{z}_*$ using the disambiguation attention mechanism, which is described as follows:

$$a_{*,j} = \frac{1}{1 + \exp\left\{-\boldsymbol{W}^\top \left(\tanh\left(\boldsymbol{W}_v^\top \boldsymbol{h}_{*,j} + \boldsymbol{b}_v\right) \odot \operatorname{sigm}\left(\boldsymbol{W}_u^\top \boldsymbol{h}_{*,j} + \boldsymbol{b}_u\right)\right)\right\}}, \tag{10}$$

$$\boldsymbol{z}_* = \frac{1}{\sum_{j=1}^{n_*} a_{*,j}} \sum_{j=1}^{n_*} a_{*,j} \boldsymbol{h}_{*,j}. \tag{11}$$

Finally, we employ the trained classifier $\mathbf{f}$ to predict the label of $\boldsymbol{X}_*$ by:

$$Y_* = \underset{c \in \mathcal{Y}}{\arg\max}\, f_c(\boldsymbol{z}_*), \tag{12}$$

where $f_c(\cdot)$ is the c-th element of $\mathbf{f}(\cdot)$, and the normalized function $f_c(\cdot)$ represents the probability of class label c being the ground-truth label.

Algorithm 1 summarizes the complete procedure of DEMIPL. First, the algorithm uniformly initializes the weights of the momentum-based disambiguation loss (Step 1). Next, the model training can be divided into two sub-steps (Steps 2-13). The initial sub-step involves extracting features for each mini-batch and aggregating them into bag-level vector representations (Steps 5-7). The subsequent sub-step encompasses calculating the loss function and updating the model (Steps 8-11). Finally, for an unseen multi-instance bag, instance-level features are extracted and aggregated into a bag-level vector representation, which is used to predict the label (Steps 14-16).

Algorithm 1 $Y_* =$ DEMIPL ($\mathcal{D}$, λ_a, T, $\boldsymbol{X}_*$)

Inputs:
$\mathcal{D}$: the multi-instance partial-label training set $\{(\boldsymbol{X}_i, \boldsymbol{S}_i) \mid 1 \leq i \leq m\}$, where $\boldsymbol{X}_i = \{\boldsymbol{x}_{i,1}, \boldsymbol{x}_{i,2}, \cdots, \boldsymbol{x}_{i,n_i}\}$, $\boldsymbol{x}_{i,j} \in \mathcal{X}$, $\mathcal{X} = \mathbb{R}^d$, $\boldsymbol{S}_i \subset \mathcal{Y}$, $\mathcal{Y} = \{l_1, l_2, \cdots, l_q\}$
λ_a : the weight for the attention loss
T: the number of iterations
$\boldsymbol{X}_*$: the unseen multi-instance bag with n_* instances
Outputs:
$\boldsymbol{Y}_*$: the predicted label for $\boldsymbol{X}_*$
Process:

```
 1: Initialize uniform weights $\boldsymbol{w}^{(0)}$ as stated by Equation (6)
 2: for $t = 1$ to $T$ do
 3:   Shuffle training set $\mathcal{D}$ into $B$ mini-batches
 4:   for $b = 1$ to $B$ do
 5:     Extract the instance-level features of $\boldsymbol{X}$ according to Equation (1)
 6:     Calculate the attention scores as stated by Equation (2)
 7:     Map the instance-level features into a single vector representation according to Equation (3)
 8:     Update weights $\boldsymbol{w}^{(t)}$ according to Equation (7)
 9:     Calculate $\mathcal{L}$ according to Equation (8)
10:     Set gradient $- \bigtriangledown_{\Phi} \mathcal{L}$
11:     Update $\Phi$ by the optimizer
12:   end for
13: end for
14: Extract the instance-level features of $\boldsymbol{X}_*$ according to Equation (9)
15: Calculate the attention scores and map the instance-level features into a single vector representa-
    tion according to Equations (10) and (11)
16: Return $Y_*$ according to Equation (12)
```

A.2 Additional Experiment Results

Table 1: Classification accuracy (mean±std) of each comparing algorithm on the benchmark datasets in terms of the different number of false positive labels $[r \in \{1, 2, 3\}]$. $\bullet/\circ$ indicates whether the performance of DeMipl is statistically superior/inferior to the compared algorithm on each dataset (pairwise t-test at a significance level of 0.05).

Algorithm	r	MNIST-MIPL	FMNIST-MIPL	Birdsong-MIPL	SIVAL-MIPL
DeMipl	1	0.976±0.008	0.881±0.021	0.744±0.016	0.635±0.041
	2	0.943±0.027	0.823±0.028	0.701±0.024	0.554±0.051
	3	0.709±0.088	0.657±0.025	0.696±0.024	0.503±0.018
		Mean			
Proden	1	0.555±0.033●	0.652±0.033●	0.303±0.016●	0.303±0.020●
	2	0.372±0.038●	0.463±0.067●	0.287±0.017●	0.274±0.022●
	3	0.285±0.032●	0.288±0.039●	0.278±0.006●	0.242±0.009●
Rc	1	0.660±0.031●	0.697±0.166●	0.329±0.014●	0.344±0.014●
	2	0.577±0.039●	0.684±0.029●	0.301±0.014●	0.299±0.015●
	3	0.362±0.029●	0.414±0.050●	0.288±0.019●	0.256±0.013●
Lws	1	0.605±0.030●	0.702±0.033●	0.344±0.018●	0.346±0.014●
	2	0.431±0.024●	0.547±0.040●	0.310±0.014●	0.312±0.015●
	3	0.335±0.029●	0.411±0.033●	0.289±0.021●	0.286±0.018●
		MaxMin			
Proden	1	0.465±0.023●	0.358±0.019●	0.339±0.010●	0.322±0.018●
	2	0.338±0.031●	0.315±0.023●	0.329±0.016●	0.295±0.021●
	3	0.260±0.037●	0.265±0.031●	0.305±0.015●	0.244±0.018●
Rc	1	0.518±0.022●	0.421±0.016●	0.379±0.014●	0.304±0.015●
	2	0.462±0.028●	0.363±0.018●	0.359±0.015●	0.268±0.023●
	3	0.366±0.039●	0.294±0.053●	0.332±0.024●	0.244±0.014●
Lws	1	0.457±0.028●	0.346±0.033●	0.349±0.013●	0.345±0.013●
	2	0.351±0.043●	0.323±0.031●	0.336±0.013●	0.314±0.019●
	3	0.274±0.037●	0.267±0.034●	0.307±0.016●	0.268±0.019●

Table 2: Classification accuracy (mean±std) of each comparing algorithm on the real-world dataset.

Algorithm	CRC-MIPL-Row	CRC-MIPL-SBN	CRC-MIPL-KMeansSeg	CRC-MIPL-SIFT
DeMipl	0.408±0.010	0.486±0.014	0.521±0.012	0.532±0.013
	Mean			
Proden	0.405±0.012	0.515±0.010○	0.512±0.014●	0.352±0.015●
Rc	0.290±0.010●	0.394±0.010●	0.304±0.017●	0.248±0.008●
Lws	0.360±0.008●	0.440±0.009●	0.422±0.035●	0.338±0.009●
	MaxMin			
Proden	0.453±0.009○	0.529±0.010○	0.563±0.011○	0.294±0.008●
Rc	0.347±0.013●	0.432±0.008●	0.366±0.010●	0.204±0.008●
Lws	0.381±0.011●	0.442±0.009●	0.335±0.049●	0.287±0.009●

In Section 3.2 and Section 3.3, we report the results of Proden, Rc, and Lws using linear models. In this section, we supplement the results of the compared PLL algorithms with multi-layer perceptrons (MLPs) as described in the respective literature. Table 1 and Table 2 present the experimental results of DeMipl compared to PLL algorithms on the benchmark and real-world datasets, respectively. It is noteworthy that DeMipl employs a two-layer CNN network for feature extraction on the MNIST-MIPL and FMNIST-MIPL datasets, while on the remaining datasets, DeMipl only utilizes linear models.

On the benchmark datasets, DeMipl consistently outperforms the compared algorithms in almost all cases. Moreover, the compared algorithms using MLPs do not consistently yield superior results compared to those using linear models, especially when the benchmark datasets exhibit relatively simple features. This suggests that linear models are sufficient to achieve satisfactory results given the benchmark datasets, while MLPs might introduce unnecessary complexity.

On the real-world dataset, DeMipl outperforms the compared algorithms in 19 out of 24 cases. When combined with complex image bag generators such as CRC-MIPL-KMeansSeg and CRC-MIPL-SIFT, DeMipl outperforms the compared algorithms in 11 out of 12 cases. In the majority of cases, the compared algorithms using MLPs demonstrate better performance than those using linear models.

However, on the CRC-MIPL-SIFT dataset, the improvement provided by MLPs is not particularly evident and sometimes even leads to a decline in performance. Therefore, when dealing with complex multi-instance features, the bag features obtained through the Mean or MaxMin strategies do not accurately reflect the characteristics of multi-instance bags. This highlights the need for specialized MIPL algorithms to accurately capture the features of multi-instance bags.

A.3 Theoretical Analysis

Theorem 1. *In a multi-instance bag $\boldsymbol{X}_i$, when the normalized attention score of an instance $\boldsymbol{x}_{i,j'}$ approaches* 1*, e.g.,* $\frac{a_{i,j'}}{\sum_{j=1}^{n_i} a_{i,j}} \rightarrow 1$*, the probability of the multi-instance bag $\boldsymbol{X}_i$ being classified as the c-th class is approximately equal to that of the instance $\boldsymbol{x}_{i,j'}$ belonging to the c-th class.*

Proof. Equation (2) demonstrates that the attention score for each instance ranges between 0 and 1. After normalizing by $\frac{1}{\sum_{j=1}^{n_i} a_{i,j}}$, the sum of attention scores for all instances within a multi-instance bag becomes equal to 1. When the normalized attention score of an instance $\boldsymbol{x}_{i,j'}$ is approach 1, the normalized attention scores of the remaining instances $\{\boldsymbol{x}_{i,1}, \boldsymbol{x}_{i,2}, \cdots, \boldsymbol{x}_{i,n_i}\} \setminus \{\boldsymbol{x}_{i,j'}\}$ approach 0. Based on Equation (3), the aggregated bag-level vector representation $\boldsymbol{z}_i = \frac{1}{\sum_{j=1}^{n_i} a_{i,j}} \sum_{j=1}^{n_i} a_{i,j} \boldsymbol{h}_{i,j} \approx \boldsymbol{h}_{i,j'} = h(\boldsymbol{x}_{i,j'})$. Therefore, it is confirmed that instances with higher attention scores contribute significantly to the bag-level predictions, underlining the significance of attention mechanisms in multi-instance partial-label learning. □

Theorem 1 suggests that high attention scores of individual instances can play a crucial role in determining the bag-level class prediction, emphasizing the significance of accurately capturing and interpreting attention scores in multi-instance partial-label learning scenarios.

A.4 Image Bag Generator

We utilize four image bag generators to extract multi-instance features from the CRC-MIPL dataset. The detailed descriptions of these image bag generators are provided below:

- **Row Generator**: It treats each row of the image as an individual instance. To extract the feature for each instance, the Row generator computes the average RGB color value of the row and the color differences in the rows above and below it.
- **SBN Generator**: It considers each 2×2 blob within the image and includes the RGB color values of the blob itself and its four neighboring blobs as features for each instance. It generates instances by iteratively moving one pixel at a time. However, it should be noted that the SBN generator ignores the feature information at the four corners of the image.
- **KMeansSeg Generator**: It divides the image into K segments or partition blocks. For each segment, it generates a 6-dimensional feature. The first three dimensions represent color values in the YCbCr color space, while the last three dimensions represent values obtained by applying the wavelet transform to the luminance (Y) component of the image.
- **SIFT Generator**: It applies the scale-invariant feature transform (SIFT) algorithm to extract features, which partitions each instance into multiple 4×4 subregions and assigns the gradients of the pixels within these subregions to 8 bins. Consequently, the SIFT generator produces a 128-dimensional feature vector for each instance.

The implementations of the four image bag generators are available at `http://www.lamda.nju.edu.cn/code_MIL-BG.ashx`.

A.5 Data and Code Availability

The implementations of the compared algorithms are publicly available. MIPLGP and PL-AGGD are implemented at `http://palm.seu.edu.cn/zhangml/`. PRODEN is implemented at `https://github.com/Lvcrezia77/PRODEN`. RC is implemented at `https://lfeng-ntu.github.io/codedata.html`. LWS is implemented at `https://github.com/hongwei-wen/LW-loss-for-partial-label`. Additionally, the code of DEMIPL, the benchmark datasets, and the real-world dataset are publicly available at `http://palm.seu.edu.cn/zhangml/`.