Controlling Class Layout for Deep Ordinal Classification via Constrained Proxies Learning

Cong Wang, Zhiwei Jiang*, Yafeng Yin, Zifeng Cheng, Shiping Ge, Qing Gu

State Key Laboratory for Novel Software Technology, Nanjing University, China
cw@smail.nju.edu.cn, {jzw, yafeng}@nju.edu.cn, {chengzf, shipingge}@smail.nju.edu.cn, guq@nju.edu.cn

Abstract

For deep ordinal classification, learning a well-structured feature space specific to ordinal classification is helpful to properly capture the ordinal nature among classes. Intuitively, when Euclidean distance metric is used, an ideal ordinal layout in feature space would be that the sample clusters are arranged in class order along a straight line in space. However, enforcing samples to conform to a specific layout in the feature space is a challenging problem. To address this problem, in this paper, we propose a novel Constrained Proxies Learning (CPL) method, which can learn a proxy for each ordinal class and then adjusts the global layout of classes by constraining these proxies. Specifically, we propose two kinds of strategies: hard layout constraint and soft layout constraint. The hard layout constraint is realized by directly controlling the generation of proxies to force them to be placed in a strict linear layout or semicircular layout (i.e., two instantiations of strict ordinal layout). The soft layout constraint is realized by constraining that the proxy layout should always produce unimodal proxy-to-proxies similarity distribution for each proxy (i.e., to be a relaxed ordinal layout). Experiments show that the proposed CPL method outperforms previous deep ordinal classification methods under the same setting of feature extractor.

Introduction

Ordinal classification, also known as ordinal regression, aims to predict the label of samples on the ordinal scale. It is a learning paradigm lying between multi-class classification and regression. Compared with multi-class classification, the classes in ordinal classification are naturally ordered. Compared with regression, the number of classes in ordinal classification is finite and the distance between adjacent classes is undefined. Some examples include predicting the age group of a person (e.g., from *Infants*, *Children* to *Aged*) and the star rating of a movie (e.g., from 1 star to 5 stars).

Traditional ordinal classification methods (Frank and Hall 2001; Chu, Ghahramani, and Williams 2005; Cardoso and da Costa 2007; Lin and Li 2012) mainly work on handcrafted features, which is labor-intensive and time-consuming. Recently, with the great progress brought by deep neural networks, several deep ordinal classification methods have been proposed (Liu, Kong, and Goh 2018; Diaz and Marathe 2019; Shaham and Svirsky 2020; Li et al. 2021) and show superior performance than traditional methods (Liu, Kong, and Goh 2018). Such performance gain is mainly attributed to neural networks' strong capability of representation learning.

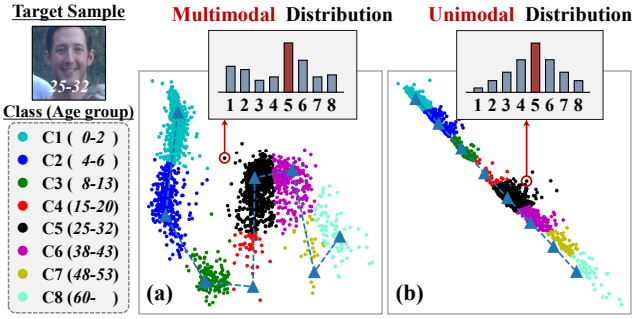


Figure 1: Illustration of (a) the unconstrained feature space and (b) our target ordinal constrained feature space.

To benefit more from the representation learning, existing deep ordinal classification methods seek to learn the feature space specific to ordinal classification. These methods fall into two fashions: classification and regression. For the case of general multi-class classification, neither feature space nor output label distribution shows any ordinal property. Therefore, researchers proposed to make implicit ordinal constraints on feature space by recoding the labels, such as transforming the N-class classification into $N-1$ binary classifications (Niu et al. 2016), predicting the relative order of samples in triplets (Liu, Kong, and Goh 2018), and softening the one-hot label distribution to unimodal distribution (Diaz and Marathe 2019). For the case of regression, the samples are directly mapped into a one-dimensional space (i.e., a line of real numbers), which is ordered in nature. But the samples are regressed into real numbers, which need to be discretized into classes by the learned boundaries, based on the unimodal transformation layer (Beckham and Pal 2017) or Gaussian process layer (Liu, Wang, and Kong 2019). The main difference between these two fashions of methods is that, the classification fashion constrains the feature space in a soft way by constraining the output label distribution, while the regression fashion constrains the feature space in a hard

*Corresponding author

way by utilizing the ordinal nature of one-dimensional space.

Inspired by these work, we consider whether we can explicitly constrain the global layout of samples in the feature space to make it reflect the ordinal nature of classes. Different from the regression fashion, which is only applicable in the one-dimensional space, we expect to do so in feature space of any dimension. When a feature space is unconstrained, as shown in Figure 1(a), the layout of samples can hardly guarantee the ordinal nature of classes. With such layout, the samples of some faraway classes may be closely distributed in space (e.g., class 1 and class 5 in Figure 1(a)), which may result in multimodal probability distributions for some samples (e.g., the target sample in Figure 1(a)). But if a feature space is ordinal constrained and the sample clusters are arranged in class order along a straight line in space, as shown in Figure 1(b), samples can always get the unimodal probability distribution (when using Euclidean distance metric). Previous studies have shown that the unimodal distribution is the ideal probability distribution for ordinal classification (Beckham and Pal 2017), which effectively reflects the ordinal nature of classes. Thus, with such ordinal constrained layout, ordinal nature of classes can be guaranteed.

However, enforcing samples to conform to a specific layout in the feature space is a challenging problem. To address this problem, we propose a novel Constrained Proxies Learning method (CPL), which can learn a proxy for each ordinal class and then adjusts the layout of classes by constraining these proxies. Similar with previous two fashions, we explore two strategies of constraining the layout of proxies: hard layout constraint and soft layout constraint. For the hard layout constraint, we directly control the generation of proxies to force them to be placed in an ordinal layout. Considering that the ordinal layout is different under different metrics, we provide two instantiations of hard layout constraint: a linear layout specific to the Euclidean distance metric, and a semicircular layout specific to the cosine similarity metric. For the soft layout constraint, we constrain the distribution shape of the similarities between each proxy and all proxies to be unimodal, so that the class layout can be a relaxed ordinal layout corresponding to a specific unimodal function.

Our main contributions are summarized as follows:

- We propose a constrained proxies learning method to explicitly control the global layout of classes in high-dimensional feature space, making it more suitable for ordinal classification.
- We propose both the hard and soft layout constraints of proxies, and explore some example layouts for both of them (i.e., two strict ordinal layouts for hard constraint and two relaxed ordinal layouts for soft constraint).
- We conduct experiments[1] on three public datasets and show that the proposed CPL achieves better performance than previous ordinal classification methods.

Related Work

Deep Ordinal Classification. Research on ordinal classification has last for about half a century (Gutiérrez et al. 2015; McCullagh 1980). Among these studies, deep ordinal classification has drawn a lot of attention in recently years, which mainly solve the task from two fashions: classification and regression. The methods fall into classification fashion usually capture the ordinal information among classes by recoding the labels. Niu et al. (2016) transformed the K-class ordinal classification task to $K-1$ binary classification tasks. Thus, the label can be recoded as a vector with $K-1$ dimensions, where the value of dimension k is the answer of *Is the target rank greater than* k. Liu, Kong, and Goh (2018) trained their models on the triplets sampled from 3 adjacent classes by computing a pairwise hinge loss. Thus, the target label can be recoded as a vector with K dimensions, where the value of dimension k is the answer of *Is the target rank greater than* $k-1$ *and smaller than* $k+1$. Diaz and Marathe (2019) trained their model with general classification paradigm, but the target one-hot label vector is softened to the label distribution with unimodal shape. Shaham and Svirsky (2020) used the modified proportional odds model to ensure that the output distribution is unimodal. The methods fall into regression usually first map the samples into real numbers, then predict their class by learning the boundaries between classes. Beckham and Pal (2017) transformed the output real number into an unimodal distribution, and predict the class by softmax operation. Liu, Wang, and Kong (2019) designed a Gaussian Process layer to map the samples and learn the boundaries of classes. Li et al. (2021) proposed probabilistic ordinal embeddings (POEs) to exploit the ordinal nature of regression. While these methods constrain the feature space implicitly, we provide a way of explicitly constraining the global layout of classes to be ordinal layout in feature space.

Proxies Learning. Proxies learning is a novel paradigm of metric learning. Previous metric learning methods (Bromley et al. 1993; Chopra, Hadsell, and LeCun 2005; Hoffer and Ailon 2015; Oh Song et al. 2017) mainly focus on constructing training pairs. To get rid of the dependence on these sampling strategies and accelerate the convergence during training, Proxy-NCA (Movshovitz-Attias et al. 2017) was proposed where each class is assigned with a learnable proxy, and the model can be trained in a way of single input. Based on Proxy-NCA, SoftTriple loss (Qian et al. 2019) was proposed to assign multiple proxies to each category to reflect intra-class variance. Manifold Proxy loss (Aziere and Todorovic 2019) extended N-pair (Sohn 2016) loss using proxies, and replaces Euclidean distance with a manifold-aware distance to improve the performance. Recently, Proxy-Anchor (Kim et al. 2020) was proposed to reformulate the triplet loss by viewing the proxy as anchor, and achieves good performance. While existing proxies learning methods often focus on modeling the local relationship between samples and proxies, the research on explicitly constraining the global layout of proxies to a specific layout is still rare.

Method

For the ordinal classification task with K classes, each sample x belongs to a class $r_k \in \mathcal{R}$, where $\mathcal{R} = \{r_0, r_1, \cdots, r_{K-1}\}$ is the set of all classes and there is an ordinal relationship among these classes $r_0 \prec r_1 \prec \cdots \prec r_{K-1}$. The objective is

[1]Code is available at https://github.com/tenvence/cpl.

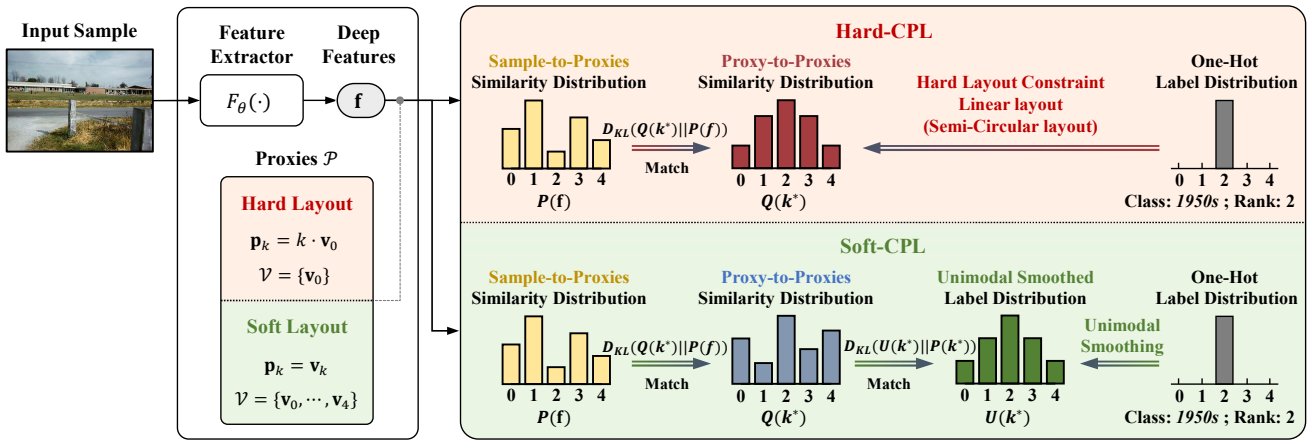


Figure 2: The framework of Constrained Proxies Learning (CPL) for deep ordinal classification.

to correctly classify each sample x into the class r_k it belongs to, while reducing the errors on the ordinal scale as many as possible. In this paper, we tackle the task from a proxies learning manner (Movshovitz-Attias et al. 2017) and propose the Constrained Proxies Learning (CPL) method.

Framework of Constrained Proxies Learning

Our CPL is designed based on the proxies learning, which can learn a proxy for each class in feature space so as to make samples belonging to the same class can be closely clustered together around the corresponding proxy. But unlike general proxies learning method, where the proxies are learned freely without constraint, our CPL aims to constrain the global layout of proxies in feature space to make it more suitable for ordinal classification. Specifically, two strategies of layout constraint are considered: hard layout constraint (Hard-CPL) and soft layout constraint (Soft-CPL). For Hard-CPL, proxies are constrained to be generated in a specific way so that they can be placed in a predefined ordinal layout. For Soft-CPL, proxies are constrained to be placed in an ordinal layout corresponding to a specific unimodal distribution.

Before describing Hard-CPL and Soft-CPL in more details, we first introduce the framework of CPL. As shown in Figure 2, the CPL model contains two components: a feature extractor $F_\theta(\cdot)$ with parameters θ, and a proxies learner $G_{\mathcal{V}}(\cdot)$ with N learnable parameter vectors $\mathcal{V} = \{\mathbf{v}_0, \cdots, \mathbf{v}_{N-1}\}$. Note that these parameters (i.e., θ and $\mathcal{V}$) can be trained together and N can be different under different layout constraints. Among these two components, the feature extractor can map a sample x to the embedding feature $\mathbf{f} = F_\theta(x) \in \mathbb{R}^d$ with d dimensions. The proxies learner can be used to generate K proxies $\mathcal{P} = \{\mathbf{p}_0, \mathbf{p}_1, \cdots, \mathbf{p}_{K-1}\}$, where the proxy $\mathbf{p}_k \in \mathbb{R}^d$ is corresponding to the ordinal class r_k.

To train the CPL model, the objective is to encourage $\mathbf{f}$ to be close to the target proxy $\mathbf{p}_{k^*}$ and to be far away from other proxies according to their relative ordinal distance with the target proxy in the feature space, where k^* denote the ground-truth class rank of the input sample.

To reach this goal, we need to first specify a similarity function $\mathrm{sim}(\cdot,\cdot)$, so that the sample-to-proxies similarity distribution of class assignment $P(\mathbf{f})$ for the input sample x can be calculated based on $\mathbf{f}$ and $\mathcal{P}$ by the softmax function:

$$P_k(\mathbf{f}) = \frac{\exp(\mathrm{sim}(\mathbf{f}, \mathbf{p}_k))}{\sum_{k'=0}^{K-1} \exp(\mathrm{sim}(\mathbf{f}, \mathbf{p}_{k'}))} \tag{1}$$

Besides, we can also calculate the proxy-to-proxies similarity distribution $Q(k^*)$ by the softmax function:

$$Q_k(k^*) = \frac{\exp(\mathrm{sim}(\mathbf{p}_{k^*}, \mathbf{p}_k)}{\sum_{k'=0}^{K-1} \exp(\mathrm{sim}(\mathbf{p}_{k^*}, \mathbf{p}_{k'})} \tag{2}$$

Then, by matching the distribution $P(\mathbf{f})$ and the distribution $Q(k^*)$, the basic loss function of CPL based on Kullback-Leibler (KL) divergence (Kullback and Leibler 1951) can be defined for model training:

$$\begin{aligned} \mathcal{L}_{\mathrm{basic}}(\mathbf{f}, k^*) &= D_{\mathrm{KL}}[Q(k^*) \| P(\mathbf{f})] \\ &= -\frac{1}{K} \sum_{k=0}^{K-1} Q_k(k^*) \log \frac{P_k(\mathbf{f})}{Q_k(k^*)} \end{aligned} \tag{3}$$

Once the CPL model is well-trained, the predicted class rank $\hat{k}$ of the input sample x can be inferred by finding the proxy most similar to its embedding feature:

$$\hat{k} = \arg\max_k P_k(F_\theta(x)) = \arg\max_k \mathrm{sim}(F_\theta(x), \mathbf{p}_k) \tag{4}$$

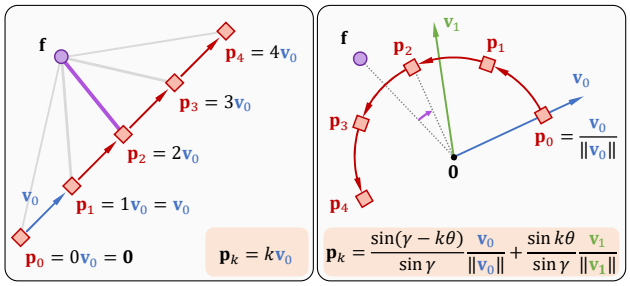
In addition, for the similarity function $\mathrm{sim}(\cdot,\cdot)$, we consider two examples based on Euclidean distance and cosine similarity, respectively. For Euclidean distance, based on Student's t-distribution (Van der Maaten and Hinton 2008), the similarity function can be formulated as:

$$\mathrm{sim}_{\mathrm{E}}(\mathbf{f}, \mathbf{p}_k) = -\log(1 + \|\mathbf{f} - \mathbf{p}_k\|^2) \tag{5}$$

For cosine similarity, the similarity function is formulated as:

$$\mathrm{sim}_{\mathrm{C}}(\mathbf{f}, \mathbf{p}_k) = s \cdot \cos(\mathbf{f}, \mathbf{p}_k) = s \cdot \frac{\mathbf{f}^{\mathrm{T}} \mathbf{p}_k}{\|\mathbf{f}\| \|\mathbf{p}_k\|} \tag{6}$$

where s is a hyperparameter to scale the range of cosine similarity and $\|\mathbf{p}_k\| = 1$. The scale parameter s is limited as $s > 1$, which is used in many cosine-based softmax loss (Deng et al. 2019; Huang et al. 2020; Wang et al. 2018; Zhang et al. 2019). Note that other similarity function can also be used in the CPL framework.



(a) Linear Layout **(b) Semicircular Layout**

Figure 3: Two example schemes of Hard-CPL.

Hard Layout Constrained Proxies Learning

For Hard-CPL, the layout of proxies is constrained in a hard way by directly constraining the generation of proxies. As shown in Figure 1, compared with unconstrained proxies learning, Hard-CPL constrains the proxies to be placed in a straight line and thus can achieve a strict ordinal layout. Considering that such linear layout guarantees the ordinal nature of classes only if Euclidean distance is used as metric, we also provide another example scheme of ordinal layout specific to the cosine similarity, called semicircle layout.

Linear Layout for Euclidean Distance (H-L). Intuitively, a straight line in a space is determined by two points (vectors). Thus, the linear layout constraint can be $\mathbf{p}_k = G_{\mathcal{V}}(k) = \mathbf{v}_0 + k \cdot \mathbf{v}_1$ which means $N = 2$ and $\mathcal{V} = \{\mathbf{v}_0, \mathbf{v}_1\}$. But when computing the Euclidean distance $\|\mathbf{f} - \mathbf{p}_k\| = \|(\mathbf{f} - \mathbf{v}_0) - k \cdot \mathbf{v}_1\|$, $\mathbf{v}_0$ can be considered as a bias for $\mathbf{f}$. Then, for simplification, the proxies learner can be formulated as:

$$\mathbf{p}_k = G_{\mathcal{V}}(k) = k \cdot \mathbf{v}_0 \tag{7}$$

where $N = 1$ and $\mathcal{V} = \{\mathbf{v}_0\}$. As illustrated in Figure 3(a), $\mathbf{p}_0$ is located in the origin point. All proxies are evenly distributed in the direction of $\mathbf{v}_0$ with the margin of $\|\mathbf{v}_0\|$.

Semicircular Layout for Cosine Similarity (H-S). Intuitively, a semicircular arc need to be determined by a plane of the feature space. Thus we use two vectors to determine a semicircular arc, which means $N = 2$ and $\mathcal{V} = \{\mathbf{v}_0, \mathbf{v}_1\}$. Specifically, $\mathbf{v}_0$ is used to determine the direction of the first proxy $\mathbf{p}_0$, and $\mathbf{v}_1$ is used to determine the expanding direction of the arc, as shown in Figure 3(b). Since the angle between $\mathbf{p}_0$ and $\mathbf{p}_{K-1}$ is π (i.e., a semicircular), the angle between adjacent proxies can be set to $\pi/(K-1)$ to equally divide the semicircle. Then the proxies learner is formulated as:

$$\mathbf{p}_k = G_{\mathcal{V}}(k) = \frac{\sin(\gamma - k\beta)}{\sin\gamma} \cdot \frac{\mathbf{v}_0}{\|\mathbf{v}_0\|} + \frac{\sin k\beta}{\sin\gamma} \cdot \frac{\mathbf{v}_1}{\|\mathbf{v}_1\|} \tag{8}$$

where $\|\mathbf{p}_k\| = 1$, $\beta = \pi/(K-1)$, and γ is the angle between $\mathbf{v}_0$ and $\mathbf{v}_1$, $\gamma = \arccos(\mathbf{v}_0^{\mathrm{T}}\mathbf{v}_1/\|\mathbf{v}_0\|\|\mathbf{v}_1\|)$.

Soft Layout Constrained Proxies Learning

For Soft-CPL, we relax the hard layout constraint, allowing proxies not to be placed in strict linear/semicircular layout. Therefore, we allow the proxies to be learned freely and only constrain that the proxy layout should always produce unimodal proxy-to-proxies similarity distribution for each proxy.

To this end, each proxy has its own learnable parameter vector, and the proxies learner can be formulated as:

$$\mathbf{p}_k = G_{\mathcal{V}}(k) = \mathbf{v}_k \tag{9}$$

where $\mathcal{V} = \{\mathbf{v}_0, \cdots, \mathbf{v}_{K-1}\}$.

To constrain the proxy-to-proxies similarity distribution $Q(k^*)$ to be unimodal, we can first define a unimodal smoothed label distribution $U(k^*)$ by a unimodal smoothing function $E(\cdot,\cdot)$ and the softmax function:

$$U_k(k^*) = \frac{\exp(E(k;k^*))}{\sum_{k'=0}^{K-1}\exp(E(k';k^*))} \tag{10}$$

Then, by matching the distributions $Q(k^*)$ and $U(k^*)$, we can define an extra unimodal loss function for Soft-CPL:

$$\mathcal{L}_{\text{unimodal}}(k^*) = D_{\text{KL}}[U(k^*)\|Q(k^*)] \tag{11}$$

While Hard-CPL only use the basic loss for model training:

$$\mathcal{L}_{\text{H}} = \mathcal{L}_{\text{basic}} \tag{12}$$

Soft-CPL use both the basic loss and unimodal loss:

$$\mathcal{L}_{\text{S}} = \mathcal{L}_{\text{basic}} + \alpha\mathcal{L}_{\text{unimodal}} \tag{13}$$

where α is tradeoff parameter.

For the unimodal smoothing function $E(\cdot,\cdot)$, we consider two classic unimodal distributions (Beckham and Pal 2017) as examples: Poisson distribution and Binomial distribution. Note that other unimodal distributions are also applicable.

Poisson Distribution (S-P). The log probability mass function (LPMF) of Poisson distribution is defined as:

$$\text{LPMF}(k;\lambda) = k\log\lambda - \lambda - \log k! \tag{14}$$

where $k \in \mathbb{N}$, $\lambda \in \mathbb{R}^+$, and the maximum value of LPMF is taken when the condition $k < \lambda < k+1$ is met. Then the ordinal smoothing function E can be formulated as:

$$E(k;k^*) = \frac{1}{\tau_p} \cdot \text{LPMF}\left(k; k^* + \frac{1}{2}\right) \tag{15}$$

where $\tau_p \in (0, +\infty)$ is a hyperparameter to control the shape of the distribution. When $\tau_p \to +\infty$ or $\tau_p \to 0$, the distribution will tend to be a uniform distribution or a one-hot distribution respectively.

Binomial Distribution (S-B). The LPMF of Binomial distribution is defined as:

$$\begin{aligned}\text{LPMF}(k;K-1,p) = &\log\binom{K-1}{k} + k\log p \\ &+ (K-1-k)\log(1-p)\end{aligned} \tag{16}$$

where $0 \le k \le K-1$, $p \in [0,1]$, and the maximum value of LPMF is taken when the condition $k < Kp < k+1$ is met. Then the ordinal smoothing function $E(\cdot,\cdot)$ can be formulated as:

$$E(k,k^*) = \frac{1}{\tau_b} \cdot \text{LPMF}\left(k; K-1, \frac{2k+1}{2K}\right) \tag{17}$$

where $\tau_b \in (0, +\infty)$ is a hyperparameter to control the shape. Same as τ_p in Equation (15), τ_b is used to control the shape of the distribution between the uniform distribution and the one-hot distribution.

Methods			Historical Color		Adience Face	
			Accuracy (%) ↑	MAE ↓	Accuracy (%) ↑	MAE ↓
Classification (Liu, Kong, and Goh 2018)			48.94 ± 2.54	0.89 ± 0.06	54.0 ± 6.3	0.61 ± 0.08
Regression (Niu et al. 2016)			42.24 ± 2.91	0.79 ± 0.03	56.3 ± 4.9	0.56 ± 0.07
Ranking (Li et al. 2021)			44.67 ± 4.24	0.81 ± 0.06	56.7 ± 6.0	0.54 ± 0.08
CNNPOR (Liu, Kong, and Goh 2018)			50.12 ± 2.65	0.82 ± 0.05	57.4 ± 5.8	0.55 ± 0.08
GP-DNNOR (Liu, Wang, and Kong 2019)			46.60 ± 2.98	0.76 ± 0.05	57.4 ± 5.5	0.54 ± 0.07
SORD (Diaz and Marathe 2019)			–	–	59.6 ± 3.6	0.49 ± 0.05
POEs (Li et al. 2021)			54.68 ± 3.21	0.66 ± 0.05	60.5 ± 4.4	0.47 ± 0.06
UPL		**Euclidean Distance**	52.20 ± 3.84	0.71 ± 0.07	58.1 ± 3.2	0.48 ± 0.05
		Cosine Similarity	51.32 ± 2.99	0.74 ± 0.05	56.8 ± 4.5	0.51 ± 0.07
CPL	**Hard-Linear**	**Euclidean Distance**	55.71 ± 3.20	$\mathbf{0.63 \pm 0.06}$	61.6 ± 2.6	$\mathbf{0.43 \pm 0.04}$
	Hard-Semicircular	**Cosine Similarity**	55.41 ± 3.21	$\underline{0.64 \pm 0.06}$	61.8 ± 3.1	$\mathbf{0.43 \pm 0.04}$
	Soft-Poisson	**Euclidean Distance**	57.28 ± 3.41	0.65 ± 0.07	61.3 ± 3.7	0.45 ± 0.05
		Cosine Similarity	56.99 ± 2.44	0.65 ± 0.05	61.1 ± 4.0	0.46 ± 0.05
	Soft-Binomial	**Euclidean Distance**	$\mathbf{57.96 \pm 3.14}$	0.66 ± 0.08	$\mathbf{62.1 \pm 3.6}$	$\underline{0.44 \pm 0.04}$
		Cosine Similarity	$\underline{57.66 \pm 3.11}$	0.65 ± 0.06	$\underline{61.9 \pm 4.5}$	$\underline{0.44 \pm 0.05}$

Table 1: The performance (accuracy and MAE) of all comparison methods on Historical Color dataset and Adience Face dataset. The feature extractors are all VGG-16. The best measures are in bold, and the second best measures are underlined.

Experiments

Datasets and Evaluation Metrics

We employ three public datasets for evaluation, which are:

- **Historical Color** (Palermo, Hays, and Efros 2012) is a small and balanced ordinal classification dataset which contains images captured on five decades, from *1930s* to *1970s*, each of which has 265 images. In each class, 210 images are randomly selected for training. For the rest 55 images, randomly selected 5 images are used for validation and 50 images are used for testing. The experiments are repeated 10 times with different partitions.
- **Adience Face** (Levi and Hassner 2015) contains 26,580 face photos from 2,284 subjects. The dataset is divided into 8 age groups, which are *0-2*, *4-6*, *8-13*, *15-20*, *25-32*, *38-43*, *48-53*, and *elder than 60 years old*, respectively. The five-fold partition follows the official repository[2].
- **Image Aesthetics** (Schifanella, Redi, and Aiello 2015) provides 15,687 Flickr image URLs, while 13,774 images are available online. The dataset contains four categories of images, namely nature, animals, people, and urban. The quality of each image is scored by at least five graders on the five scales, i.e., *unacceptable*, *flawed*, *ordinary*, *professional*, and *exceptional*. In the experiments, we randomly select 75% images as the training set, 5% images as the validation set, and 20% images as the test set. The experiments are repeated 5 times with different partitions.

The evaluation metrics are accuracy and Mean Absolute Error (MAE), which are widely-used in previous work.

Implementation Details

Feature Extractor. VGG-16 (Simonyan and Zisserman 2014) pretrained on ImageNet (Deng et al. 2009) is used as the feature extractor. The last *fc* layer of VGG-16 is replaced with a *fc* layer to map the raw feature from the default 4096 dimensions to our specified dimensions.

Proxies Learner. The parameters of the proxies learner are initialized by Xavier (Glorot and Bengio 2010) normal distribution, and are trained together with the feature extractor.

Training. We employ AdamW (Loshchilov and Hutter 2019) as the optimizer. Learning rates of feature extractor and proxies learner are set as 0.001 and 0.01. The batch size is set as 32. All models are trained with PyTorch (Paszke et al. 2019) for 48 epochs. During all training epochs, the model which achieves minimum MAE on the validation set is selected.

Image Setting. In training, the images are argumented by randomly cropping, resizing to 224×224 and randomly horizontal flipping. In testing, the images are processed by resizing to 256×256 and center cropping to 224×224.

Hyperparameter Setting. Dimension of features and proxies are 512 for fair comparison with baseline methods. α, τ_p, τ_b, and s are set as 6, 0.11, 0.13, and 6, respectively.

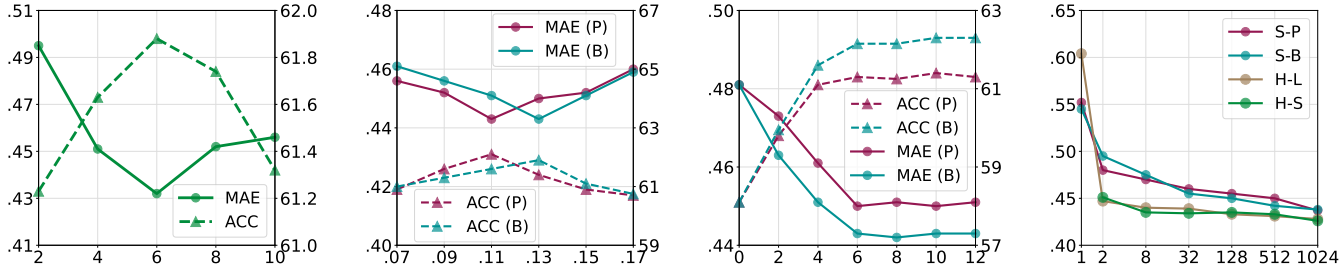
Comparison to Other Methods

Results on the three employed datasets are summarized in Table 1 and Table 2. In general, compared with all baseline methods, the proposed CPL achieves overall better performance. Hard-CPL-Linear and Soft-CPL-Binomial with Euclidean distance achieve best MAE and best accuracy, respectively. Among them, Hard-CPL-Linear outperforms the previous state-of-the-art (SOTA) method POEs (Li et al. 2021) on all three datasets by 1.03%, 1.1%, and 0.58% higher on accuracy, 0.03, 0.04, and 0.007 lower on MAE, respectively. Soft-CPL-Binomial with Euclidean distance outperforms POEs by 3.28%, 1.6%, and 0.93% higher on accuracy, 0.00, 0.03, and 0.001 lower on MAE, respectively. This demonstrates the effectiveness of our proposed CPL for the ordinal classification

[2]https://github.com/GilLevi/AgeGenderDeepLearning

Methods			Accuracy (%) ↑					MAE ↓				
			Nature	Animals	Urban	People	Overall	Nature	Animals	Urban	People	Overall
Classification (Liu, Kong, and Goh 2018)			70.97	68.02	68.19	71.63	69.45	0.305	0.342	0.374	0.412	0.376
Regression (Niu et al. 2016)			71.52	70.72	71.22	69.72	70.80	0.378	0.397	0.387	0.400	0.390
Ranking (Li et al. 2021)			69.81	69.10	66.49	66.49	68.96	0.313	0.331	0.349	0.312	0.326
CNNPOR (Liu, Kong, and Goh 2018)			71.86	69.32	69.09	69.94	70.05	0.294	0.322	0.325	0.321	0.316
SORD (Diaz and Marathe 2019)			73.59	70.29	73.25	70.59	72.03	0.271	0.308	**0.276**	0.309	0.290
POEs (Li et al. 2021)			73.62	71.14	72.78	72.22	72.44	0.273	0.299	0.281	0.293	0.287
UPL		**Euclidean Distance**	71.82	68.21	69.24	68.98	69.56	0.283	0.343	0.313	0.341	0.320
		Cosine Similarity	72.88	68.68	69.88	69.81	70.31	0.284	0.325	0.311	0.352	0.318
CPL	**Hard-Linear** **Hard-Semicircular**	**Euclidean Distance**	74.43	72.11	72.99	72.53	73.02	**0.260**	**0.289**	0.283	0.287	**0.280**
		Cosine Similarity	74.35	71.50	72.91	72.33	72.77	0.262	0.297	0.288	0.290	0.284
	Soft-Poisson	**Euclidean Distance**	74.46	71.73	72.94	72.45	72.90	0.267	0.302	0.281	0.297	0.287
		Cosine Similarity	74.53	71.39	72.97	72.38	72.82	0.270	0.299	0.287	**0.286**	0.286
	Soft-Binomial	**Euclidean Distance**	**74.97**	**72.61**	**73.28**	72.61	**73.37**	0.262	0.297	0.285	0.299	0.286
		Cosine Similarity	74.62	72.28	73.20	**72.74**	73.21	0.265	0.301	0.286	0.294	0.287

Table 2: The performance (accuracy and MAE) of all comparison methods on Image Aesthetics dataset. The feature extractors are all VGG-16. The best measures are in bold, and the second best measures are underlined.



(a) s for H-S (b) τ_p and τ_b for S-P and S-B (c) α for S-P and S-B (d) Feature dimension

Figure 4: Effects of the hyperparameters and the feature dimension on the performance of CPL. In each subfigure, the y-axis on the left represents the metric of MAE, and the y-axis on the right (if exists) represents the metric of accuracy.

task. Compared with UPL[3], CPL achieves better results by a large margin, which means that explicitly controlling the class layout is effective for the ordinal classification task.

Among all settings of our CPL, Hard-CPL generally achieves better MAE than Soft-CPL, while Soft-CPL generally achieves better accuracy than Hard-CPL. This may be because the hard layout constraints enforce the proxies to be placed in a more rigorous pre-defined ordinal layout, and thus can better catch the ordinal nature among classes. For Soft-CPL, the strategy using Euclidean distance achieves better results than that using cosine similarity. It means that Euclidean distance is more suitable for Soft-CPL.

Model Analysis

Effect of Scale Parameter s. To explore the effect of s on the performance of H-S, we test the performance by varying the value of s from 2 to 10 with step 2 on the Adience Face dataset. The results, which are summarized in Figure 4(a), show that both too small and too large s deteriorate the performance. The range of the logits in softmax function is $[-\infty, +\infty]$, while the range of cosine distance ($s = 1$) is $[-1, 1]$. Therefore, small s makes the range of logits small, which reduces the discrimination ability between classes after softmax function. Moreover, too large s causes the target distribution to be close to the one-hot distribution after softmax, which further leads to worse performance because of less modeling of ordinal relationship.

Effect of Control Parameter τ_p and τ_b. To explore the effects of τ_p and τ_b on the performance of Soft-CPL, we test the performance by varying the values of τ_p and τ_b from 0.07 to 0.17 with step 0.02 on the Adience Face dataset. The results, which are summarized in Figure 4(b), show that small τ leads to worse performance, while too large τ also deteriorates performance. This is because smaller τ makes the target distribution tend to the one-hot vector, which lacks the modeling of ordinal relationship. When τ becomes larger, the target distribution is more like uniform distribution, which reduces the discrimination ability between classes.

[3]UPL (Unconstrained Proxies Learning) encourages $P(\mathbf{f})$ to match the one-hot label distribution by the cross entropy loss.

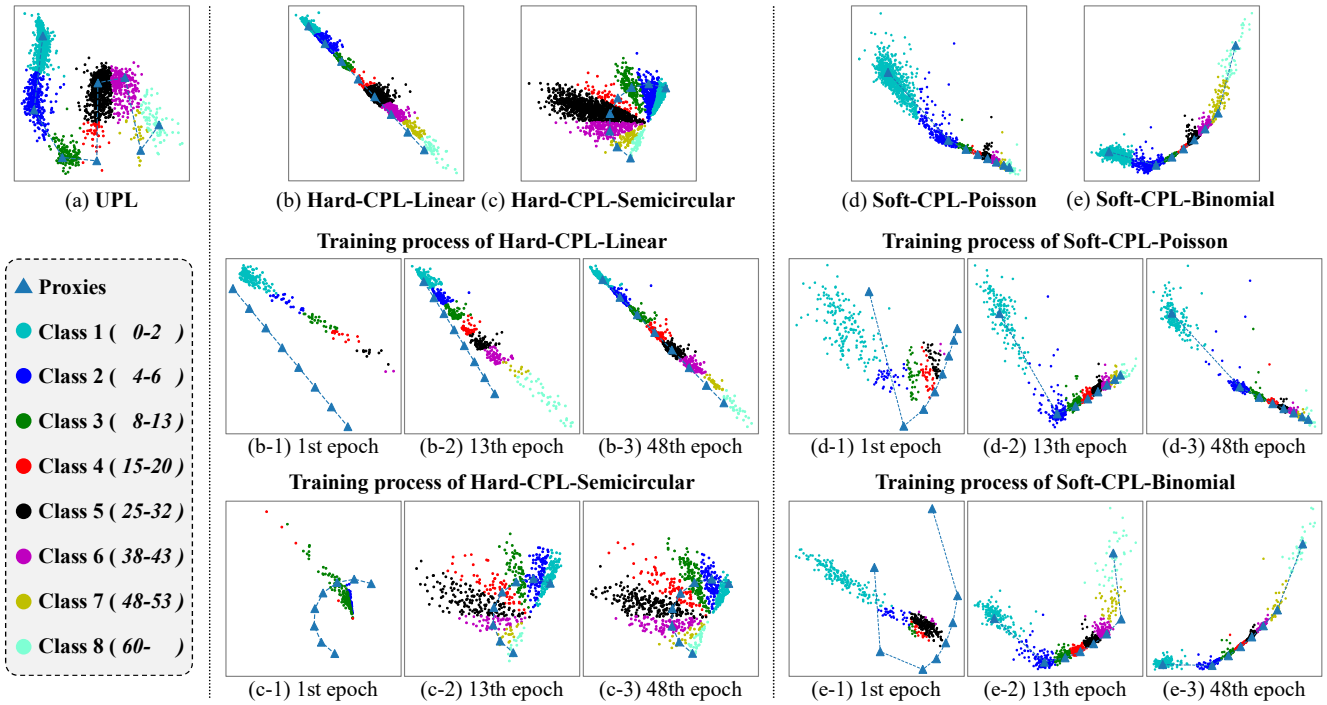


Figure 5: Visualization of CPL on the first fold of Adience Face dataset with eight age groups. The dimension of features and proxies is set as 2. Only the features of correctly-predicted samples are plotted. The visualization on the test set are plotted on top. The visualization on validation set after 1st, 13th and 48th epochs are also plotted to illustrate the training processes.

Effect of Tradeoff Parameter α. To explore the effect of tradeoff parameter α, we test the performance by varying the value of α from 0 to 12 with step 2 on Adience Face dataset. The results, which are summarized in Figure 4(c), show that smaller α leads to worse performance, while α larger than 6 achieves the stable performance. This indicates that $\mathcal{L}_{\text{unimodal}}$ can significantly help Soft-CPL to catch the ordinal nature.

Effect of Feature Dimension d. To explore the effect of feature dimension d, we test the performance by varying the value of d from 1 to 2048 on Adience Face dataset. The MAE results are summarized in Figure 4(d). For Soft-CPL, MAE decreases with the increase of feature dimension. Especially from $d = 1$ to $d = 2$, performance particularly improves. It means Soft-CPL is more sensitive to feature dimension. For Hard-CPL, MAE results are more stable under most feature dimensions, excluding $d = 1$.

Visualization

Visualization of Hard-CPL. The visualization of Hard-CPL are summarized in Figure 5(b) and Figure 5(c). Because of hard layout constraint, proxies and feature clusters are both arranged in expected ordinal layouts. Each proxy is almost the centroid of corresponding feature cluster in H-L, while in H-S each proxy is roughly located in the central angle of the corresponding feature cluster. The visualization of training processes are summarized from Figure 5(d-1) to Figure 5(e-3). For H-L, Figure 5(d-1) shows that clusters and proxies are getting closer until they completely coincide. For H-S, Figure 5(c-1) to Figure 5(e-3) show that the angle between the feature and the target proxy decreases gradually.

Visualization of Soft-CPL. The visualization of Soft-CPL are summarized in Figure 5(d) and Figure 5(e). For S-P and S-B, proxies and feature clusters are both arranged in expected ordinal layouts, where different unimodal functions produce different relaxed ordinal layouts (i.e., with different shapes). Each proxy is almost the centroid of the corresponding feature cluster. The visualization of training processes are summarized from Figure 5(b-1) to Figure 5(c-3). We can learn that the number of correctly classified features gradually increases, and the features of same-class samples are clustered more closely.

Conclusion

In this paper, we aim to learn a feature space specific to ordinal classification by explicitly constraining the layout of samples in feature space. To this end, we propose the constrained proxies learning method. From the perspectives of constraining the proxies layout in both hard way and soft way, we explore two strategies, *i.e.*, Hard-CPL and Soft-CPL. Hard-CPL directly controls the generation of proxies to force them to be placed in a strict linear layout or semicircular layout. Soft-CPL constrains that the proxy layout should always produce unimodal proxy-to-proxies similarity distribution for each proxy. We conduct experiments on three widely-used datasets of ordinal classification, and the experimental results demonstrate the effectiveness of the proposed CPL method.

Acknowledgments

This work is supported by National Nature Science Foundation of China under Grants Nos. 61972192, 62172208, 61906085, 41972111. This work is partially supported by Collaborative Innovation Center of Novel Software Technology and Industrialization.

References

Aziere, N.; and Todorovic, S. 2019. Ensemble deep manifold similarity learning using hard proxies. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 7299–7307.

Beckham, C.; and Pal, C. 2017. Unimodal probability distributions for deep ordinal classification. In *International Conference on Machine Learning*, 411–419. PMLR.

Bromley, J.; Guyon, I.; LeCun, Y.; Säckinger, E.; and Shah, R. 1993. Signature verification using a" siamese" time delay neural network. *Advances in neural information processing systems*, 6.

Cardoso, J.; and da Costa, J. P. 2007. Learning to classify ordinal data: The data replication method. *Journal of Machine Learning Research*, 8: 1393–1429.

Chopra, S.; Hadsell, R.; and LeCun, Y. 2005. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, 539–546. IEEE.

Chu, W.; Ghahramani, Z.; and Williams, C. K. 2005. Gaussian processes for ordinal regression. *Journal of machine learning research*, 6(7).

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, 248–255. Ieee.

Deng, J.; Guo, J.; Xue, N.; and Zafeiriou, S. 2019. Arcface: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 4690–4699.

Diaz, R.; and Marathe, A. 2019. Soft labels for ordinal regression. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 4738–4747.

Frank, E.; and Hall, M. 2001. A simple approach to ordinal classification. In *European conference on machine learning*, 145–156. Springer.

Glorot, X.; and Bengio, Y. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, 249–256. JMLR Workshop and Conference Proceedings.

Gutiérrez, P. A.; Perez-Ortiz, M.; Sanchez-Monedero, J.; Fernandez-Navarro, F.; and Hervas-Martinez, C. 2015. Ordinal regression methods: survey and experimental study. *IEEE Transactions on Knowledge and Data Engineering*, 28(1): 127–146.

Hoffer, E.; and Ailon, N. 2015. Deep metric learning using triplet network. In *International workshop on similarity-based pattern recognition*, 84–92. Springer.

Huang, Y.; Wang, Y.; Tai, Y.; Liu, X.; Shen, P.; Li, S.; Li, J.; and Huang, F. 2020. Curricularface: adaptive curriculum learning loss for deep face recognition. In *proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 5901–5910.

Kim, S.; Kim, D.; Cho, M.; and Kwak, S. 2020. Proxy anchor loss for deep metric learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 3238–3247.

Kullback, S.; and Leibler, R. A. 1951. On information and sufficiency. *The annals of mathematical statistics*, 22(1): 79–86.

Levi, G.; and Hassner, T. 2015. Age and gender classification using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 34–42.

Li, W.; Huang, X.; Lu, J.; Feng, J.; and Zhou, J. 2021. Learning Probabilistic Ordinal Embeddings for Uncertainty-Aware Regression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 13896–13905.

Lin, H.-T.; and Li, L. 2012. Reduction from cost-sensitive ordinal ranking to weighted binary classification. *Neural Computation*, 24(5): 1329–1367.

Liu, X.; Fan, F.; Kong, L.; Diao, Z.; Xie, W.; Lu, J.; and You, J. 2020. Unimodal regularized neuron stick-breaking for ordinal classification. *Neurocomputing*, 388: 34–44.

Liu, Y.; Kong, A. W. K.; and Goh, C. K. 2018. A constrained deep neural network for ordinal regression. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 831–839.

Liu, Y.; Wang, F.; and Kong, A. W. K. 2019. Probabilistic deep ordinal regression based on gaussian processes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 5301–5309.

Loshchilov, I.; and Hutter, F. 2019. Decoupled Weight Decay Regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

McCullagh, P. 1980. Regression models for ordinal data. *Journal of the Royal Statistical Society: Series B (Methodological)*, 42(2): 109–127.

Movshovitz-Attias, Y.; Toshev, A.; Leung, T. K.; Ioffe, S.; and Singh, S. 2017. No fuss distance metric learning using proxies. In *Proceedings of the IEEE International Conference on Computer Vision*, 360–368.

Niu, Z.; Zhou, M.; Wang, L.; Gao, X.; and Hua, G. 2016. Ordinal regression with multiple output cnn for age estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 4920–4928.

Oh Song, H.; Jegelka, S.; Rathod, V.; and Murphy, K. 2017. Deep metric learning via facility location. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 5382–5390.

Palermo, F.; Hays, J.; and Efros, A. A. 2012. Dating historical color images. In *European Conference on Computer Vision*, 499–512. Springer.

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32.

Qian, Q.; Shang, L.; Sun, B.; Hu, J.; Li, H.; and Jin, R. 2019. Softtriple loss: Deep metric learning without triplet sampling. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 6450–6458.

Schifanella, R.; Redi, M.; and Aiello, L. M. 2015. An image is worth more than a thousand favorites: Surfacing the hidden beauty of flickr pictures. In *Proceedings of the International AAAI Conference on Web and Social Media*, volume 9, 397–406.

Shaham, U.; and Svirsky, J. 2020. Deep Ordinal Regression using Optimal Transport Loss and Unimodal Output Probabilities. *arXiv preprint arXiv:2011.07607*.

Simonyan, K.; and Zisserman, A. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Sohn, K. 2016. Improved deep metric learning with multi-class n-pair loss objective. *Advances in neural information processing systems*, 29.

Van der Maaten, L.; and Hinton, G. 2008. Visualizing data using t-SNE. *Journal of machine learning research*, 9(11).

Wang, H.; Wang, Y.; Zhou, Z.; Ji, X.; Gong, D.; Zhou, J.; Li, Z.; and Liu, W. 2018. Cosface: Large margin cosine loss for deep face recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 5265–5274.

Xie, J.; Girshick, R.; and Farhadi, A. 2016. Unsupervised Deep Embedding for Clustering Analysis. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48, 478–487.

Zhang, D.; Nan, F.; Wei, X.; Li, S.-W.; Zhu, H.; McKeown, K.; Nallapati, R.; Arnold, A. O.; and Xiang, B. 2021. Supporting Clustering with Contrastive Learning. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 5419–5430.

Zhang, X.; Zhao, R.; Qiao, Y.; Wang, X.; and Li, H. 2019. Adacos: Adaptively scaling cosine logits for effectively learning deep face representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10823–10832.

A Pseudo Code of CPL

The pseudo code of our CPL can be seen in Algorithm 1.

Algorithm 1: The Pseudo Code of CPL

Input: The training set $\mathcal{D}_{\mathrm{Tr}}$ and the validation set $\mathcal{D}_{\mathrm{Val}}$
Output: The class of each sample in the test set $\mathcal{D}_{\mathrm{Te}}$
▷ **Training Stage**
Initialize the model;
for each epoch **do**
  for each batch sampled from $\mathcal{D}_{\mathrm{Tr}}$ **do**
    Forward and calculate the loss (i.e., $\mathcal{L}_H$ or $\mathcal{L}_S$);
    Back propagate and update the model;
  end for
  Evaluate the performance of the model on $\mathcal{D}_{\mathrm{Val}}$;
end for
▷ **Inference Stage**
Load the model parameters which achieves the best performance on $\mathcal{D}_{\mathrm{Val}}$;
for each sample in $\mathcal{D}_{\mathrm{Te}}$ **do**
  Forward and predict the class of the sample;
end for
return predicted classes of samples in $\mathcal{D}_{\mathrm{Te}}$

B More Unimodal Distributions for Soft-CPL

In our Soft-CPL, two unimodal distributions, i.e., Poisson distribution and Binomial distribution, are considered as the unimodal smoothing function. While other unimodal distributions are also applicable, such as exponential function (Liu et al. 2020) or triangular distribution.

For exponential function, the ordinal smoothing function $E(\cdot,\cdot)$ is formulated as

$$E(k,k^*)=\frac{\exp(-|k-k^*|/\tau_e)}{\sum_{j=1}^{K}\exp(-|j-k^*|/\tau_e)} \tag{18}$$

where τ_e is a hyperparameter to control the variance of the distribution, which is set as 30.

For triangular distribution, the ordinal smoothing function $E(\cdot,\cdot)$ is formulated as

$$f(k,k^*)=a-\frac{(a-b)\cdot|k-k^*|}{\max(k^*,K-k^*-1)} \tag{19}$$

$$E(k,k^*)=f(k,k^*)/\sum_{j=0}^{K-1}f(j,k^*) \tag{20}$$

where a and b are the hyperparameters to control the maximum and minimum values of $f(k,k^*)$, which are set as 0.9 and 0.1, respectively.

As shown in Table 3, the performances of the exponential function and the triangular distribution are similar with the performances of the Poisson distribution and the Binomial distribution. In general, trying to find more suitable unimodal distributions for Soft-CPL is a valuable direction of our further research.

C Effect of Negative Euclidean Distance

An intuitive similarity function for Euclidean distance is negative Euclidean distance, which is formulated as

$$\mathrm{sim}_{\mathrm{E}}(\mathbf{f},\mathbf{p}_k)=-\|\mathbf{f}-\mathbf{p}_k\| \tag{21}$$

But in our CPL, the similarity function for Euclidean distance is formulated as:

$$\mathrm{sim}_{\mathrm{E}}(\mathbf{f},\mathbf{p}_k)=-\log(1+\|\mathbf{f}-\mathbf{p}_k\|^2) \tag{22}$$

	HC		AF		IA	
	Acc.	**MAE**	**Acc.**	**MAE**	**Acc.**	**MAE**
Euc (P)	57.28	0.65	61.3	0.45	72.90	0.287
Euc (B)	57.96	0.66	62.1	0.44	73.37	0.286
Euc (E)	57.24	0.65	61.4	0.45	72.85	0.288
Euc (T)	57.25	0.66	61.3	0.45	73.42	0.287
Cos (P)	56.99	0.65	61.1	0.46	72.82	0.286
Cos (B)	57.66	0.65	61.9	0.44	73.21	0.287
Cos (E)	57.01	0.66	60.9	0.45	73.20	0.288
Cos (T)	56.97	0.67	60.7	0.45	73.19	0.288

Table 3: Performances of different unimodal distributions on Soft-CPL. HC, AF, and IA denote the Historical Color dataset, the Adience Face dataset, and the Image Aesthetics dataset, respectively. Euc and Cos denote using Euclidean distance and using cosine similarity. P, B, E, and T denote Poisson, Binomial, exponential, and triangular, respectively.

which is widely-used in the Euclidean-based metric learning methods (Zhang et al. 2021; Xie, Girshick, and Farhadi 2016). It has the property that $(1+||\mathbf{f}-\mathbf{p}_k||^2)^{-1}$ approaches an inverse square law for large distances. This makes the probabilities almost invariant for the large distance. It leads to smooth gradient variation, and more stable optimization (Van der Maaten and Hinton 2008).

To compare the performance of negative Euclidean distance with that of our similarity function in CPL, we conduct experiments on the three datasets. As shown in Table 4, the performances of negative Euclidean distance is a little worse than that of our similarity function. It indicates that our similarity function is a better choice for the CPL methods when using Euclidean distance.

	HC		AF		IA	
	Acc.	**MAE**	**Acc.**	**MAE**	**Acc.**	**MAE**
H-L	55.71	0.63	61.6	0.43	73.02	0.280
H-L †	53.99	0.69	61.4	0.46	72.75	0.321
S-P	57.28	0.65	61.3	0.45	72.90	0.287
S-P †	56.44	0.73	61.1	0.47	72.13	0.318
S-B	57.96	0.66	62.1	0.44	73.37	0.286
S-B †	56.84	0.71	61.7	0.47	72.99	0.311

Table 4: Comparison the performance of negative Euclidean distance with that of our similarity function in CPL. † denotes using negative Euclidean distance. HC, AF, and IA denote the Historical Color dataset, the Adience Face dataset, and the Image Aesthetics dataset, respectively.

D Effect of the Value of $\|v_0\|$ for H-L

In H-L, the vector v_0 is learnable. We conduct the experiments to study the effect of $\|v_0\|$ on the performance of H-L by fixing the $\|v_0\|$ with different values. The results in Table 5 show that the performance degrades a lot if we fix $\|v_0\|$ to some default numbers rather than make it learnable.

	HC		AF		IA	
	Acc.	**MAE**	**Acc.**	**MAE**	**Acc.**	**MAE**
learnable v_0	55.71	0.63	61.6	0.43	73.02	0.280
Fix $\|v_0\| = 1$	52.01	0.73	58.1	0.53	67.99	0.369
Fix $\|v_0\| = 3$	52.12	0.71	58.4	0.52	68.02	0.341
Fix $\|v_0\| = 5$	52.11	0.71	58.5	0.52	67.98	0.335
Fix $\|v_0\| = 7$	52.31	0.69	58.9	0.49	68.04	0.334

Table 5: Performance of H-L by fixing the v_0 with different values. HC, AF, and IA denote the Historical Color dataset, the Adience Face dataset, and the Image Aesthetics dataset, respectively.

E Proxies Learner of Hard-CPL-Semicircular

Because the proxies are all located in the plane determined by $\mathbf{v}_0$ and $\mathbf{v}_1$, the generated proxies are the linear combination of $\mathbf{v}_0$ and $\mathbf{v}_1$, which means

$$\mathbf{p}_k = a \cdot \frac{\mathbf{v}_0}{\|\mathbf{v}_0\|} + b \cdot \frac{\mathbf{v}_1}{\|\mathbf{v}_1\|} \tag{23}$$

where $a, b \in \mathbb{R}$ are the combination weights. Based on the law of sines,

$$\frac{\|\mathbf{p}_k\|}{\sin(\pi-\gamma)} = \frac{b}{\sin k\beta} = \frac{a}{\sin(\gamma - k\beta)} \tag{24}$$

Because $\|\mathbf{p}_k\| = 1$, then

$$a = \frac{\sin(\gamma - k\beta)}{\sin\gamma},\ b = \frac{\sin k\beta}{\sin\gamma} \tag{25}$$

Finally, we get

$$\mathbf{p}_k = \frac{\sin(\gamma - k\beta)}{\sin\gamma} \cdot \frac{\mathbf{v}_0}{\|\mathbf{v}_0\|} + \frac{\sin k\beta}{\sin\gamma} \cdot \frac{\mathbf{v}_1}{\|\mathbf{v}_1\|} \tag{26}$$

F More Visualization

In this section, we provide more visualization results in Figure 6, including the features of correctly-predicted samples and misclassified samples. The features of misclassified samples are not evenly dispersed outside the decision boundary, but are mostly located in the decision areas of the adjacent ordinal classes. At the beginning of training, the features of misclassified samples are scattered disorderly in the space. However, with the progress of training, the features of misclassified samples are gradually reduced, and are mainly distributed in the decision areas of the adjacent ordinal classes.

G Trivial Collapsed Solution of H-L

For the formulation of H-L, it seems obvious to contain a trivial solution which collapses all sample features and all proxies $\mathbf{p}_k$ into the zero point. However, the collapse issue did not occur in the experiments. The reason is as follows.

In the KL divergence loss of H-L, the target distribution is provided by $Q(k^*)$ which will not be optimized, and $P(\mathbf{f})$ is the distribution to be optimized. Therefore, the condition the proxies collapse into the zero point is that $Q(k^*)$ is a uniform distribution. As long as $\|\mathbf{v}_0\|$ is not initialized as 0, $Q(k^*)$ will be the unimodal distribution, and proxies always do not collapse.

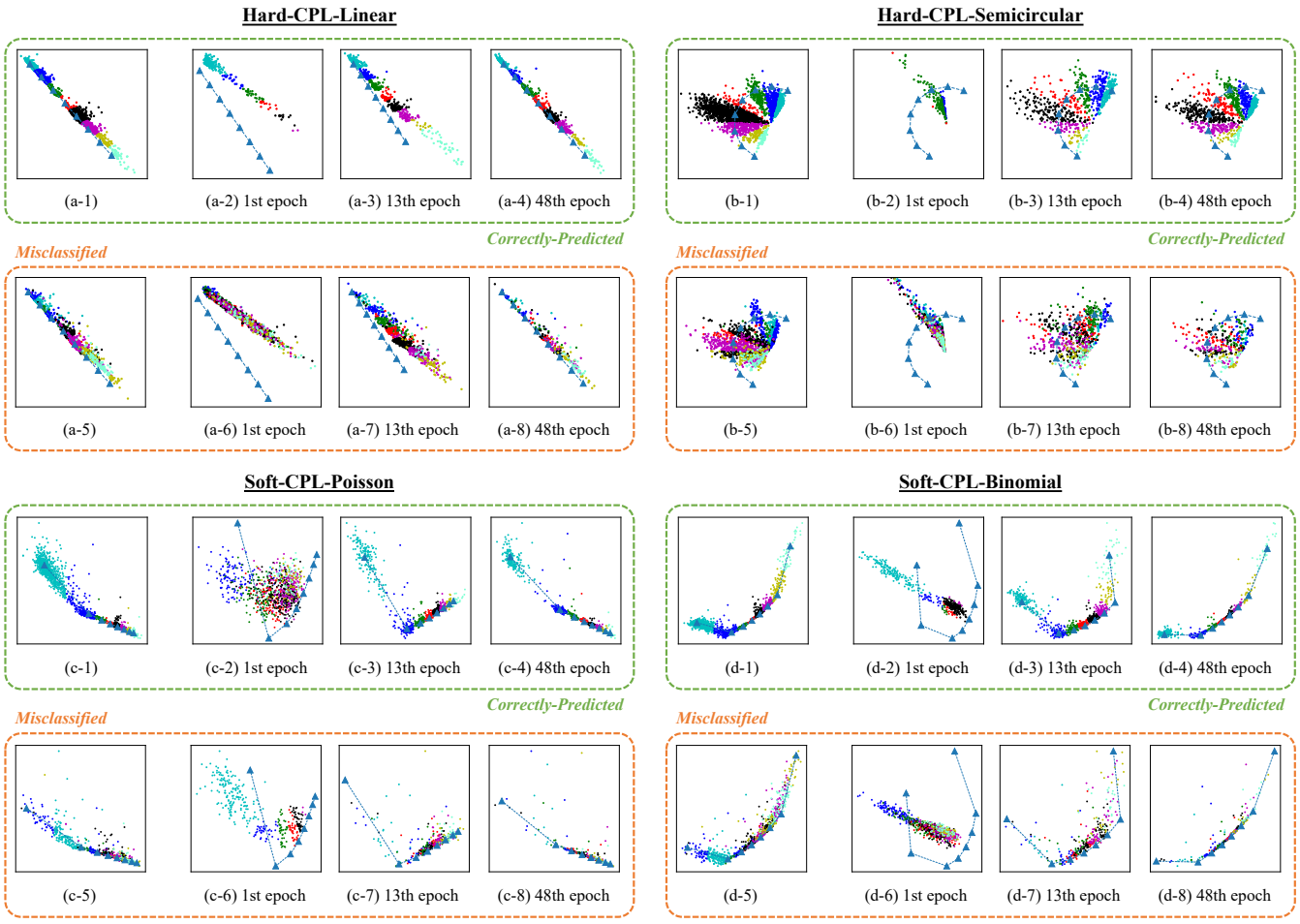


Figure 6: More Visualization of our proposed CPL on the first fold of Adience Face dataset with eight age groups.