ProcSim: Proxy-based Confidence for Robust Similarity Learning

Oriol Barbany[1†] Xiaofan Lin[2] Muhammet Bastan[2] Arnab Dhua[2]
[1]Institut de Robòtica i Informàtica Industrial, CSIC-UPC [2]Visual Search & AR, Amazon
obarbany@iri.upc.edu, {xiaofanl,mbastan,adhua}@amazon.com

Abstract

Deep Metric Learning (DML) methods aim at learning an embedding space in which distances are closely related to the inherent semantic similarity of the inputs. Previous studies have shown that popular benchmark datasets often contain numerous wrong labels, and DML methods are susceptible to them. Intending to study the effect of realistic noise, we create an ontology of the classes in a dataset and use it to simulate semantically coherent labeling mistakes. To train robust DML models, we propose ProcSim, a simple framework that assigns a confidence score to each sample using the normalized distance to its class representative. The experimental results show that the proposed method achieves state-of-the-art performance on the DML benchmark datasets injected with uniform and the proposed semantically coherent noise.

1. Introduction

The problem of quantifying the similarity between images is typically framed in the context of metric learning, which aims at learning a metric space in which distances closely relate to underlying semantic similarities. Deep Metric Learning (DML) is based on transforming the images using a neural network and then applying a predefined metric, *e.g.*, the Euclidean distance, or cosine similarity.

Identifying visual similarities is crucial for tasks such as image retrieval [28], zero-shot learning [6], and person identification [44, 45]. Solving these problems with DML allows the introduction of new classes without retraining, a desirable feature in applications such as retail [53]. Moreover, the learned similarity model can be easily paired with efficient nearest-neighbor inference techniques [19].

DML requires labeled datasets, but manual labeling is cumbersome and, in some cases, infeasible. Automated labeling, while efficient, introduces errors like duplicates and irrelevant images, often necessitating manual correction [46]. Conversely, manual annotations often involve non-expert annotators on crowdsourcing platforms, leading to occasional labeling errors [22]. Labeling mistakes are especially problematic for DML, which suffer a higher drop in performance than classification models as the number of noisy labels increases [10].

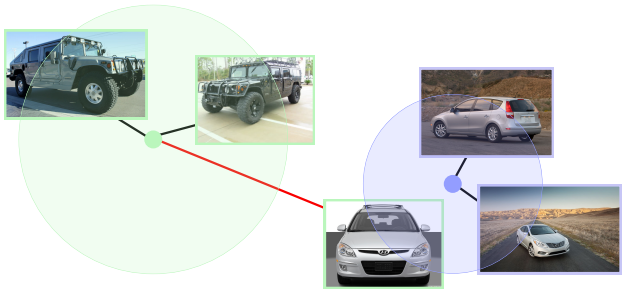

Figure 1. ProcSim handles incorrect labels by reducing the contribution of samples whose learned embeddings are too far away from their class representatives.

While DML with noisy labels has garnered attention, prior research has mostly focused on building robust models against uniform noise [25, 56, 60]. However, due to the annotation techniques in image retrieval, real datasets often exhibit noise concentrated in clusters of similar images [10].

This paper proposes ProcSim, a new confidence-aware framework for training robust DML models by estimating the reliability of samples in an unsupervised fashion. To test the benefits of our method on noisy datasets, we present a new procedure for injecting semantically coherent label errors. The empirical results show the superior performance of ProcSim trained on benchmark datasets injected with uniform and the proposed semantic noise in front of alternative approaches.

The main contributions of this paper are:

- We propose ProcSim, a novel framework for robust visual similarity learning usable on top of any general-purpose DML loss to improve performance on noisy datasets. ProcSim assigns a per-sample confidence that indicates the reliability of its label and is used to determine the influence of such a sample during training.
- We introduce a new noise model based on swapping semantically similar class labels. Sec. 3.6 describes how to automatically obtain a hierarchy of the classes in a dataset and use it to inject label noise.

† Work performed during an internship at Amazon.

2. Related work

2.1. Learning with noisy labels

Some approaches dealing with noisy data estimate the noise transition matrix [35, 55, 60], which requires prior knowledge or a subset of clean data. Another class of methods uses the model predictions to correct the labels [20, 24, 60]. However, this technique can lead to confirmation bias, where prediction errors accumulate and harm performance [57]. Alternatively, one can estimate which samples are incorrectly annotated [15, 17, 24]. These methods typically assume that significant loss instances can be associated with incorrect labels, a technique commonly known as the small-loss trick.

The small-loss trick is rooted in the observation that deep neural networks often learn clean samples before noisy samples [1], resulting in inputs with accurate labels exhibiting lower-magnitude losses [7].

Some works on noisy classification train two semi-independent networks that exchange information about noisy samples to prevent their memorization [15,23,51,59]. Directly adopting these methods to DML is not feasible [56], but there exist similar approaches in the DML literature using self-distillation to determine soft labels [60] or detect noisy samples [17].

If the noise probability is known and the small-loss trick assumption is satisfied, one can spot noisy samples as those whose loss value is over a given percentile determined by the noise probability [17,25]. However, the amount of noise present in a dataset is generally unknown.

Under the more realistic case where the noise probability is unknown, an interesting approach is to fit a bimodal distribution to explain the loss values [24]. Then, following the small-loss trick, the samples belonging to the distribution with the higher mode are treated as noisy.

Once noisy samples are detected, we can split the training dataset into disjoint sets representing correct and incorrect labels. In the context of DML, when we identify a sample as noisy, we can discard it [25] or only consider it for negative interactions [17].

Instead of treating all correct and incorrect samples equally, an option is to use a confidence-aware loss, in which the loss amplitude is modulated proportionally to the sample confidence [32]. Ideally, noisy samples will be assigned a low confidence score to reduce or even suppress their contribution. SuperLoss [5] offers a task-agnostic approach to converting any loss into a confident-aware loss without additional learnable parameters.

2.2. Inter-class similarities

Inter-class similarities can be considered by clustering image features and creating a class tree [13] or promoting the clusters formed during training [58]. Another approach is to modify a margin-based objective so that the margin depends on the attribute similarity [28]. One compelling alternative is to distill the knowledge of a Large Language Model (LLM) to learn semantically consistent metric spaces [40]. One can also learn a hyperbolic space [12, 56], which naturally embeds hierarchies.

2.3. Non-uniform noise generation

Swapping labels using semantic similarities results in plausible labeling mistakes and noisy samples that are more challenging to spot [24]. Using this idea, some works on noisy classification considered injecting class label errors based on the structure of recurring mistakes in real datasets, *e.g.*, Truck→Automobile, Bird→Airplane, and Dog↔ Cat [20, 24]. However, inferring these rules is specific to each dataset and requires statistics about the errors.

In the context of noisy DML, Liu *et al*. [25] proposed an iterative procedure to introduce noise. In each iteration, they choose a class and group its samples by employing a similarity measure computed using a pre-trained DML model. Then, they assign the same class label to all cluster members. Although this method incorporates a notion of visual similarity for the clustering step, label assignment is performed uniformly at random, and the number of classes decreases at each iteration. Dereka *et al*. [10] introduced the large and small class label noise models based on only corrupting the most frequent or rarest classes. While this method restricts the set of possible labels assigned (asymmetric noise), the choice is purely based on class frequencies, not semantics.

3. Methodology

3.1. Preliminaries

Let $\mathcal{D} := \{(\mathbf{x}_i, y_i)\}_{i\in[n]}$ be a dataset with pairs of images $\mathbf{x}_i \in \mathcal{X}$ and class labels $y_i \in [C]$. DML aims to learn a metric space (Ψ, d) with fixed $d : \Psi \times \Psi \to \mathbb{R}$ and a learned transformation $\phi : \mathcal{X} \to \Psi$ such that $d(\phi(\mathbf{x}_i), \phi(\mathbf{x}_j)) < d(\phi(\mathbf{x}_i), \phi(\mathbf{x}_k))$ if $\mathbf{x}_i$ is semantically more similar to $\mathbf{x}_j$ than it is to $\mathbf{x}_k$ [2]. Commonly, the space Ψ is normalized to the unit hypersphere for training stability [43,44,54], and d is chosen to be the Euclidean or cosine distance.

Instead of computing the confidence of the sample using a learnable model [9, 32, 42], we prefer to follow a parsimonious approach inspired by SuperLoss [5], a technique that computes a confidence score from the training loss and uses it for the task of automatic curriculum learning. In the curriculum learning training, the samples are fed in increasing order of difficulty, which improves the speed of convergence and the quality of the models obtained [3, 14].

For the DML problem, SuperLoss assigns a confidence σ_{ij} to each pair of samples. Doing that requires an objective expressed as a double sum over pairs, *e.g.*, the contrastive

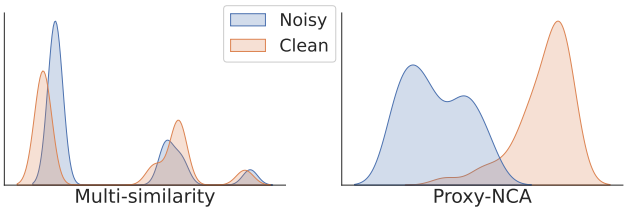

Figure 2. Distribution of loss values for clean and noisy samples in the late stages of training on CUB200 [49] with 50% uniform noise. While the Multi-similarity (MS) loss [50] is a powerful objective for training DML models, it is unsuited for label noise identification. Classification of noisy samples using Otsu's threshold [33] achieved 50% and 90% recall, respectively. More details in the supplementary.

loss [8]. For a pair of samples (i, j) with loss ℓ_{ij}, instead of directly minimizing $\mathbb{E}_{(i,j)}[\ell_{ij}]$ as in regular training, SuperLoss proposes to minimize

$$\mathbb{E}_{(i,j)}\left[\min_{\sigma_{ij}}(\ell_{ij}-\tau_{ij})\sigma_{ij}+\lambda(\log\sigma_{ij})^2\right], \qquad (1)$$

where $\lambda \in \mathbb{R}^+$, and τ_{ij} is the global average of all positive (resp. negative) pair losses across all iterations if $y_i = y_j$ (resp. $y_i \neq y_j$). The optimization of the pair confidence has the closed form solution

$$\sigma_{ij} = \exp\left[-W\left(\frac{1}{2}\max\left\{-\frac{2}{e}, \frac{\ell_{ij}-\tau_{ij}}{\lambda}\right\}\right)\right], \qquad (2)$$

where $W(\cdot)$ is the principal branch of the Lambert W function. The authors of SuperLoss [5] use this analytical solution to compute the optimal confidence and avoid the minimization in Eq. (1). The confidence is treated as a constant, meaning that they don't propagate gradients through it.

3.2. Identifying noisy samples

Curriculum learning down-weights the contribution of challenging samples, sometimes resulting in the omission of noisy samples [18, 27]. However, inputs considered hard in the curriculum learning context change across iterations while the number of incorrect annotations in a dataset remain the same. Particularly for DML, the loss is obtained by considering interactions–pairs, triplets, or tuples of a higher order–with the other samples in a batch. Hence, large loss values may be either because of a wrong label of the anchor sample or others included in the considered interactions. Therefore, data points that are hard to explain under the training objective are not necessarily those with an incorrect class label.

Fig. 2 shows the distribution of noisy and clean samples when using two well-known DML losses. The MS [50] objective penalizes the positive pairs with lower similarity and the negative pairs with higher similarity. Thus, a clean sample interacting with a noisy one will almost exclusively consider the latter, which will cause large loss values. Hence, this loss is unsuited for spotting noisy samples.

Let $\{\mathbf{p}_i\}_{i\in[C]}$ be a set of points representing classes and $\mathbf{x}$ an unlabeled sample. The nearest neighbor search on ϕ returns $\arg\max_{i\in[C]}\langle\phi(\mathbf{x}), \mathbf{p}_i\rangle$. Softmax is a smooth approximation of $\arg\max$, and replacing it in the previous expression yields a stochastic nearest neighbor classifier. The Proxy-NCA [30] loss for sample i, which we will refer to as ℓ_i^{Proxy}, is precisely the negative $log(\cdot)$ of the probability that a stochastic nearest neighbor classifier assigns a sample to its correct label when $\{\mathbf{p}_i\}_{i\in[C]}$ are class proxies.

The class proxies are learnable embeddings representing data groups and have the desirable feature that they are robust to noisy labels [21]. Therefore, even when some class contains wrong annotations, their proxies will be close to the embeddings of the clean samples of that class. Overall, Proxy-NCA loss is fundamentally a normalized distance to the class representative. This observation provides a theoretical explanation of why large sample loss values can be associated with a possibly incorrect label.

3.3. Separating noisy and clean samples

In Fig. 2, we present some empirical evidence of the identifiability of noisy samples under the Proxy-NCA [30]. Indeed, the distribution follows a bimodal pattern, with wrongly annotated data points falling within the mode exhibiting higher losses. One option to separate clean and noisy samples is to use a Gaussian mixture model [24]. However, this method assumes that each distribution is a Gaussian, which is not the case for the skewed distributions of clean and noisy samples in Fig. 2. Moreover, this approach requires an iterative procedure to estimate the sufficient statistics of each distribution.

An alternative is using Otsu's method, a one-dimensional discrete analog of Fisher's discriminant analysis. This approach selects a threshold that minimizes the intra-class variance (equivalently, maximizing the inter-class variance) and is typically used to perform image thresholding. Otsu's method does not require any optimization, has no hyperparameters, and achieves the same result as globally optimal K-means [26].

In Alg. 1, we describe the procedure to determine the Otsu threshold for our case. Note that the tested thresholds $\mathcal{T}$ correspond to the midpoints between consecutive loss values. Each of these thresholds divides the samples into two groups with at least two items each, which allows for computing the variance. Then, Otsu's method [26] exhaustively tests all thresholds and selects the one with a lower cost.

3.4. Sample confidence

We previously showed that ℓ_i^{Proxy} behaves as a bimodal distribution and that we can use Otsu's method [33] to sep-

Algorithm 1 COMPUTATION OF OTSU'S THRESHOLD

Inputs: Proxy loss values $\{\ell_i^{\text{Proxy}}\}_i$
Output: Threshold τ
Sort loss values $\mathtt{L} \leftarrow \mathtt{sorted}(\ell_i^{\text{Proxy}})$
Define thresholds $\mathcal{T} \leftarrow \{\frac{\mathtt{L}[i]+\mathtt{L}[i+1]}{2}\}_{i\in\{2,3,\dots,|\mathcal{B}|-2\}}$
for all $\tau' \in \mathcal{T}$ **do**
  Let $\mathcal{C}_0 \leftarrow \{\ell_i^{\text{Proxy}} | \ell_i^{\text{Proxy}} < \tau'\}$
  Let $\mathcal{C}_1 \leftarrow \{\ell_i^{\text{Proxy}} | \ell_i^{\text{Proxy}} \geq \tau'\}$
  Let $\text{Cost}(\tau) \leftarrow \frac{1}{|\mathcal{B}|}\left(|\mathcal{C}_0|\text{Var}[\mathcal{C}_0] + |\mathcal{C}_1|\text{Var}[\mathcal{C}_1]\right)$
end for
$\tau \leftarrow \arg\min_{\tau'\in\mathcal{T}} \text{Cost}(\tau')$

arate clean and noisy samples. Having this, we want to design a confidence score. Unlike SuperLoss [5], we advocate for computing a confidence score for each data point instead of doing so for each pair. Concretely, we want a confidence score σ_i satisfying the following criteria:

(i) σ_i is translation invariant w.r.t. ℓ_i^{Proxy}.

(ii) $\sigma_i \geq \sigma_j \iff \ell_i^{\text{Proxy}} \leq \ell_j^{\text{Proxy}}$ (i, j in the same batch).

(iii) $\sigma_i \in [0, 1]$.

(iv) As $\lambda \to 0$, $\sigma_i \to 1$ if clean, $\sigma_i \to 0$ otherwise.

(v) As $\lambda \to \infty$, $\sigma_i \to 1$.

Claim 1 *The choice*

$$\sigma_i := \exp\left\{-W\left(\left[\frac{\ell_i^{Proxy} - \tau}{2\lambda}\right]_+\right)\right\}, \tag{3}$$

where $[\cdot]_+$ is the positive part, and τ computed with Alg. 1 satisfies Conditions (i) to (v).

Equation (3) draws inspiration from SuperLoss [5]. The reason is that the sample-level version of the SuperLoss confidence yields a clean expression and already satisfies Conditions (i), (ii), and (v). While the proposed changes might seem subtle, they conceptually make a huge difference and improve the performance by a large margin (see Tab. 1). Refer to the supplementary material for the proof of Claim 1 and further discussion.

In stark contrast with SuperLoss [5], the confidence in ProcSim is not computed from the training loss. Having a different loss for the confidence computation and the parameter update can avoid biases, something considered in the works leveraging two models for unbiased noise sample identification [15, 17, 23, 51, 59, 60].

3.5. ProcSim

ProcSim can work with any DML objectives writable as a sum over sample losses, a prerequisite for enabling independent scaling of the sample loss through σ_i. In this scenario, the gradients of the loss monotonously increase with σ_i, and low-confidence samples result in diminished gradient updates.

DML model training typically relies on binary similarities, *i.e.*, identifying whether a pair of samples belong to the same class. However, evaluation involves unseen classes, so DML requires learning a notion of similarity rather than discriminating between training classes.

With this in mind, we add a self-supervised regularization loss to implicitly enforce a semantic structure among classes. Directly applying the confidence score to the regularized objective would alter the magnitude of both the supervised and unsupervised losses. Since the computation of the unsupervised loss does not rely on labels, we want it to be unaffected by the confidence.

The final objective then becomes

$$\mathcal{L} = \frac{1}{|\mathcal{B}|} \sum_{(\mathbf{x}_i, y_i)\in\mathcal{B}} \sigma_i \cdot \ell_i^{\text{DML}} + \omega \ell_i^{\text{SSL}}, \tag{4}$$

where ω a hyperparameter weighting the importance of the regularization loss. Note that setting $\sigma_i = 1$ amounts to regular training, while for $\sigma_i = 0$ the metric space is only learned with the semantic knowledge of the LLM. An overview of the proposed method is presented in Fig. 3.

By default, ProcSim uses MS [50] as the supervised DML loss, but we also assess the performance using other losses in Sec. 4.2. In the case of using the MS objective, the DML sample loss is

$$\begin{aligned}\ell_i^{\text{DML}} := &\frac{1}{\alpha}\log\left[1 + \sum_{j\in\mathcal{P}_i} e^{-\alpha(S_{ij}-\delta)}\right] \\ &+ \frac{1}{\beta}\log\left[1 + \sum_{j\in\mathcal{N}_i} e^{-\beta(\delta - S_{ij})}\right],\end{aligned} \tag{5}$$

where $S_{ij} := \langle\phi(\mathbf{x}_i), \phi(\mathbf{x}_j)\rangle$, which is equivalent to the cosine distance because we enforce $\|\phi(\mathbf{x}_i)\| = 1\ \forall i$, and $\alpha, \beta, \delta \in \mathbb{R}$ are hyperparameters.

Unless explicitly stated, we choose the Pseudolabel Language Guidance (PLG) loss [40] as the self-supervised objective. To compute the PLG loss images are input to a classifier pre-trained on ImageNet [41]. For each image, the top$-k$ class names are passed to the language part of CLIP [37] using the prompt "`A photo of a {label}`". Subsequently, k similarity matrices are generated from the similarities of text embeddings. The PLG loss is the row-wise KL divergence between the matrix of visual similarities and the mean of the k matrices of language similarities. We refer the interested reader to the PLG paper for further details.

3.6. Semantically coherent noise generation

Artificial noise models allow injecting a controlled amount of noise to assess the robustness of different meth-

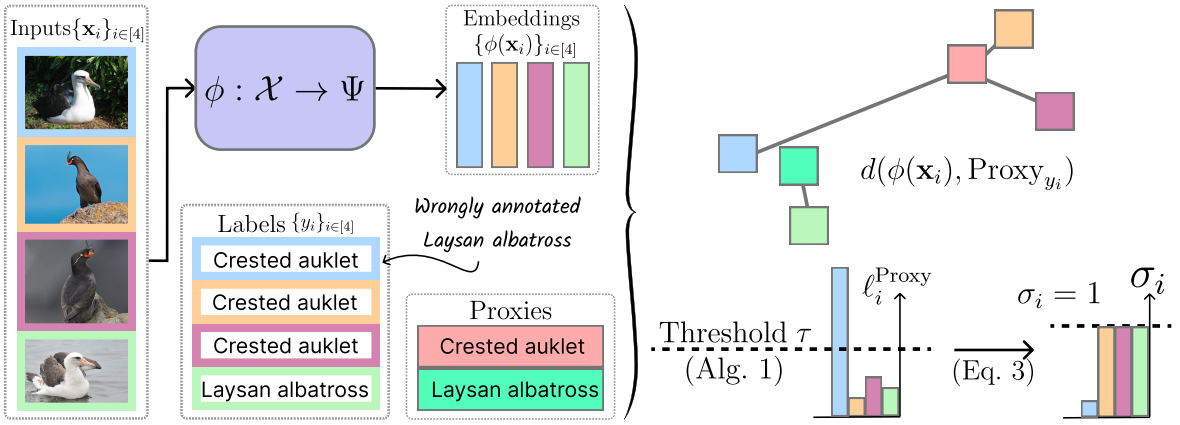

Figure 3. ProcSim model overview using an illustrative example. Here, we showcase the ProcSim model's functionality with four images $\{\mathbf{x}_i\}_{i\in[4]}$ from the CUB200 dataset [49]. These images have class labels $y_1 = y_2 = y_3 \neq y_4$, where y_1 has been erroneously assigned; it should be $y_1 = y_4 \neq y_2 = y_3$. The DML model projects images into the metric space, yielding visual embeddings $\{\psi(\mathbf{x}_i)\}_{i\in[4]}$. Then we compute the proxy loss ℓ_i^{Proxy}, which is obtained by evaluating the distance from an embedding to its associated proxy. We determine a threshold for proxy loss values using Alg. 1, and then calculate the sample confidence $\{\sigma_i\}_{i\in[4]}$ using Eq. (3). Samples with proxy loss values below the threshold possess unit confidence, while others have a smaller value that decreases as they move farther away from the proxies. Notably, $(\mathbf{x}_1, y_1)$ is assigned a low confidence score, resulting in its limited contribution to updating the model parameters compared to other samples.

ods. A simple and ubiquitous noise model is the symmetric noise model [48], based on assigning an incorrect label picked uniformly at random from all the classes. However, labeling mistakes are often due to the semantic similarity of the correct and wrong classes. For this reason, noisy labels contained in real datasets follow a non-uniform distribution among classes, differing from the symmetric model.

To mimic label errors where semantically similar images are confused, we propose computing the inherent taxonomy of the dataset's classes and using that in the noise injection process. Among the considered benchmark datasets, Standard Online Products (SOP) [46] is the only one that provides a grouping of classes. Concretely, the 22,634 products belong to one of twelve categories. Hence, for SOP, one can inject semantic noise by swapping the class label of a training sample to another class in the train partition that falls within the same category.

To build a semantic taxonomy for the CUB200 [49] and Cars196 [22] datasets, we group the natural language class names in the dataset by finding their hypernyms with WordNet [29], as done by Rohrbach *et al*. [38]. Given that a word can have multiple meanings, captured by WordNet synsets [29], and hence several potential hypernyms, we ensure that all the class names are a hyponym of *bird* and *car*, respectively. In other words, we enforce a common root node grouping all the classes. Refer to the supplementary material for further details and visualizations of the obtained class hierarchies.

To inject noise into the training splits of the datasets, we first filter the taxonomy to the training classes and treat each sample independently. Then, we traverse the class hierarchy starting at the leaf node corresponding to the original label until we find a node with several children. Finally, we select the incorrect class label uniformly at random among all children except the original class. We compute the class taxonomies only once and generate noisy versions of each dataset offline.

The fact that the noise model differs from the principles motivating ProcSim has two main reasons. On the one hand, using the same class hierarchy for noise generation and training could lead to unfair biases favoring our method. On the other hand, using word hierarchies such as WordNet [29] to resolve inter-class similarities empirically achieves lower retrieval performance than other methods such as PLG [40].

4. Experiments

4.1. Experimental details

Datasets: We report results on CUB200 [49], Cars196 [22], and SOP [46]. For all datasets, the sets of train and test classes are disjoint.

Implementation details: We implement ProcSim using PyTorch [34], which also provides the utilized ResNet-50 [16] backbone model with pre-trained ImageNet [41] weights. We replace the last layer of the backbone model with a fully connected layer that provides embeddings of dimension 512. The PLG [40] and the MS losses [50] are adapted from the original implementations and use the hyperparameters proposed by the authors for each dataset. The reported metrics are obtained by retrieving the nearest neighbors using the cosine similarity. For a fair compar-

Table 1. Recall@1 on the CUB200 [49] dataset for different types and levels of noise. The methods included in the ablation study are classified depending on how the confidence (if any) is computed. All the methods in each group share the same hyperparameters. Best results are shown in **bold**. ProcSim and its variants consistently outperform all the other baselines, and ProcSim (base) achieves the best performance overall in terms of the harmonic mean on all corrupted datasets.

NOISE TYPE →	NONE	SEMANTIC			UNIFORM			HARMONIC MEAN
METHODS ↓	-	10%	20%	50%	10%	20%	50%	
Pair-level confidence								
SuperLoss [5]	49.8	49.7	48.8	48.3	49.2	48.8	47.3	48.8
Base non-confidence-aware losses								
Proxy-NCA [30]	58.0	57.8	56.4	51.9	57.3	56.9	55.7	56.2
MS [50]	67.9	64.8	60.6	49.0	64.0	60.7	49.5	58.6
MS + PLG [40]	69.4	68.7	67.7	62.3	68.5	68.4	55.5	65.4
ProcSim and variants of it (ours)								
ProcSim (base)	70.1	**72.2**	**71.0**	**67.9**	69.3	70.4	**60.8**	**68.6**
Threshold on MS instead of Proxy-NCA	69.1	69.2	66.9	67.7	67.8	66.0	54.1	65.4
Proxy-NCA instead of MS as DML loss	59.0	58.1	56.9	51.3	58.2	59.2	56.4	56.9
Regularization affected by confidence	65.7	63.4	62.7	56.9	63.0	62.5	52.3	60.6
Global average instead of Otsu's method	69.6	69.6	69.2	64.1	**70.5**	71.1	59.0	67.3
Gaussian Mixture Model instead of Otsu's method	**70.2**	64.9	69.1	64.4	70.4	**71.2**	58.0	66.6

ison, we do not apply learning rate scheduling [39]. We also report the results with fixed hyperparameters for each dataset to show that our method achieves good performance without requiring fine-tuning for different types and probabilities of noise. Please refer to the supplementary for additional implementation details.

4.2. Ablation study

This section presents a study in which we assess the boost in image retrieval performance obtained with each of ProcSim's components. We report the Recall@1 achieved on the CUB200 [49] dataset and its corrupted versions in Tab. 1. As baselines, we consider the base DML losses, which treat all samples equally, and SuperLoss [5].

We implement the SuperLoss framework using the details provided by Castells *et al.* [5]: learning rate, weight decay, scheduling[1], contrastive loss [8], and λ hyperparameter. We compute the contrastive loss using the PyTorch Metric Learning library [31] and weight each loss term by the confidence in (2) before reducing the loss.

SuperLoss [5] yields poor results, which can be due to its susceptibility to techniques such as hard-negative mining and hyperparameter tuning [17]. However, its surprising robustness to noise motivates the usage of a confidence-aware objective. Computing confidence scores at the sample level, as we do in ProcSim, yields much better results than the pair-level scheme of SuperLoss. Moreover, it can use any objective written as a sum over samples. Waiving this restriction allows the incorporation of more powerful DML objectives that alone outperform the pair-level confidence scheme.

[1]We apply learning rate scheduling to this method since its absence led to significantly worse results. All the other methods don't use scheduling to avoid confounders in the performance boost [39].

Table 2. Recall@1 when ProcSim uses BERT [11] instead of CLIP [37] for the computation of the self-supervised loss. Difference with ProcSim inside parentheses.

UNIFORM NOISE (%) →	10%	20%	50%
CUB200 [49]	71.3 (+2.0)	71.2 (+0.8)	60.3 (-0.5)
CARS196 [22]	86.9 (-0.3)	86.3 (+0.3)	75.6 (+0.4)
SOP [46]	79.1 (-0.2)	77.9 (-0.5)	73.1 (-0.2)

Proxy-NCA loss [30] is preferable for noise identification, but its base performance falls behind the MS loss [50]. Adding the PLG term [40] promotes learning a representation that captures semantics. When using this regularization, we achieve a consistently better performance than plain MS loss and improved robustness against semantic noise compared to uniform noise.

We can see that weighting the DML loss by the confidence score and not on the regularization term yields a consistent improvement. In this case, noisy samples rely more on the regularization objective than the supervised DML loss, which is affected by label noise. Finally, using other thresholding methods like global average, as in SuperLoss [5], or Gaussian mixtures, as in [24], results in generally worse performance.

ProcSim does not have a monotonically decreasing performance with noise, a behavior only observed for the CUB200 dataset [49]. On the one hand, this can be due to using the same hyperparameters across all corrupted datasets and Otsu's method [33] separating the samples into two groups. Note that this assumes that the Proxy-NCA loss [30] follows a bimodal distribution, which may decrease the contribution of correctly labeled samples when there are no wrong labels. Solving this is as easy as setting a larger λ, which accounts for a more equal treatment of the two sets of samples separated by the threshold. However, we wanted to show that even if not tuning λ, Proc-

Table 3. Recall@K (%) on the benchmark datasets corrupted with 30% uniform noise for different values of K. The reported results for all methods except ProcSim (ours) are taken from Yan *et al*. [56], and the asterisk (*) indicates that their method was applied on top of the indicated DML loss. Best results are shown in **bold**. ProcSim achieves a superior performance according to most of the metrics. Note that, similarly to the runner-up method, ProcSim is a robustness framework built on top of the MS loss [50].

BENCHMARKS →	CUB200 [49]				CARS196 [22]				SOP [46]		
METHODS ↓	R@1	R@2	R@4	R@8	R@1	R@2	R@4	R@8	R@1	R@10	R@100
Triplet [44]	54.3	67.1	77.4	85.6	44.3	57.0	69.0	79.1	51.7	69.2	84.1
Triplet* [56]	55.5	68.1	78.2	85.9	46.1	58.2	69.6	79.3	52.9	70.1	84.6
LiftedStruct [46]	61.6	73.0	82.1	89.1	77.1	85.3	91.6	94.8	67.9	82.0	91.5
LiftedStruct* [56]	64.3	75.5	83.6	90.1	79.2	87.1	82.0	95.0	69.1	83.0	92.1
MS [50]	62.0	73.8	82.5	89.6	79.5	86.7	91.7	95.1	72.0	85.7	94.1
MS* [56]	65.3	76.1	84.7	90.7	82.4	89.5	93.8	95.9	73.6	86.9	94.8
ProcSim (ours)	**68.8**	**79.8**	**87.4**	**92.4**	**84.1**	**90.6**	**94.7**	**97.0**	**77.7**	**89.5**	**95.0**

Table 4. Recall@1 (%) on the benchmark datasets corrupted with different probabilities of uniform noise. The reported results for all methods except ProcSim (ours) are taken from the PRISM paper [25] and rounded to one decimal place for consistency with the other tables. Best results are shown in **bold**. While MCL+PRISM [25] performs slightly better than ProcSim for low levels of noise on SOP [46], our method consistently and considerably outperforms it in the other datasets.

BENCHMARKS →	CUB200 [49]			CARS196 [22]			SOP [46]		
METHODS ↓	10%	20%	50%	10%	20%	50%	10%	20%	50%
DML with Proxy-based Losses									
FastAP [4]	54.1	53.7	51.2	66.7	66.4	58.9	69.2	67.9	65.8
nSoftmax [61]	52.0	49.7	42.8	72.7	70.1	54.8	70.1	68.9	57.3
ProxyNCA [30]	47.1	46.6	41.6	69.8	70.3	61.8	71.1	69.5	61.5
Soft Triple [36]	51.9	49.1	41.5	76.2	71.8	52.5	68.6	55.2	38.5
DML with Pair-based Losses									
MS [50]	57.4	54.5	40.7	66.3	67.1	38.2	69.9	67.6	59.6
Circle [47]	47.5	45.3	13.0	71.0	56.2	15.2	72.8	70.5	41.2
Contrastive Loss [8]	51.8	51.5	38.6	72.3	70.9	22.9	68.7	68.8	61.2
MCL [52]	56.7	50.7	31.2	74.2	69.2	46.9	79.0	76.6	67.2
MCL + PRISM [25]	58.8	58.7	56.0	80.1	78.0	72.9	**80.1**	**79.5**	72.9
ProcSim (ours)	**69.3**	**70.4**	**60.8**	**87.2**	**86.0**	**75.2**	79.3	78.4	**73.3**

Sim obtains good results. Note that in any case finding λ is equivalent to finding the noise level of the data, but to the severity by which we decrease the importance of noisy sample. On the other hand, surprisingly, the best results are achieved with some semantic noise. Note that along with PLG regularization, having some labels swapped to semantically similar samples can force the model to learn a space with semantically related groups.

4.3. Influence of the language model

The PLG loss uses the language part of CLIP [37], which is trained on vision-language paired datasets. While this means CLIP is well-aligned to learn semantic information for a visual similarity task, it also means that its training set might overlap with vision datasets [37]. For this reason, we tested ProcSim with a pre-trained BERT base model [11] as LLM. The performance in Tab. 2 shows the generalization capacity of ProcSim and factors out the possibility of unfair advantages by using CLIP.

Another possible issue arising from the PLG loss is its limitation by the performance of the image classification model. Concretely, the classifier discretizes the number of language embeddings and limits it by the number of classes. Moreover, the categories may not align with the downstream dataset. One possible solution to bypass the classifier is to distill information from CLIP image embeddings. This approach takes advantage of the multi-modality of the model and achieves comparable performance in all datasets with slight improvements on SOP. Refer to the supplementary for the results and additional discussions.

4.4. Comparison to state-of-the-art

Previous methods for robust DML report results on the benchmark datasets corrupted with uniform noise. For an extensive and exhaustive comparison, we present the image retrieval performance that ProcSim obtains compared to state-of-the-art approaches. We facsimile the results reported in the papers, which means that although the noise

Table 5. Recall@1 (%) on the benchmark datasets injected with different probabilities and models of noise. Best results are shown in **bold**. ProcSim obtains a consistently better performance and is significantly more robust to semantic noise than the alternatives.

BENCHMARKS →	CUB200 [49]			CARS196 [22]			SOP [46]		
METHODS ↓	10%	20%	50%	10%	20%	50%	10%	20%	50%
Uniform noise									
LSD [60]	63.0	62.1	57.2	78.5	72.3	65.2	76.6	75.4	68.7
MCL + PRISM [25]	58.1	56.4	54.7	78.7	74.8	68.6	76.4	76.6	72.6
ProcSim (ours)	**69.3**	**70.4**	**60.8**	**87.2**	**86.0**	**75.2**	**79.3**	**78.4**	**73.3**
Semantic noise									
LSD [60]	62.8	61.9	58.5	77.5	76.6	73.0	76.8	73.7	69.1
MCL + PRISM [25]	57.7	57.9	50.6	77.8	75.9	63.4	76.6	75.8	72.2
ProcSim (ours)	**72.2**	**71.0**	**67.9**	**86.9**	**86.3**	**81.1**	**79.0**	**77.8**	**73.3**

statistics are the same, the corrupted samples could differ. We also found methods like MS [50] to be inconsistent across papers, likely due to different implementations and hyperparameters.

Tab. 3 presents the results obtained using adaptive hierarchical similarity [56] on top of common DML objectives trained on datasets with a 30% of wrong annotations. Among all DML objectives augmented with adaptive hierarchical similarity [56], MS attains the best performance, further motivating utilizing the MS loss as the base DML objective for ProcSim. The model trained with ProcSim outperforms all the other methods in all metrics, proving to be a better alternative to enhance the MS loss [50].

Liu *et al*. [25] report results on the benchmark datasets corrupted with 10%, 20%, and 50% of uniform noise. In Tab. 4, we report their results along the ProcSim performance. We can see further evidence of the superiority of MS [50] in front of Proxy-NCA loss [30] and of the vastly higher performance of Procsim on the CUB200 [49] and Cars196 [22] datasets.

We can observe a slightly lower performance on SOP. On the one hand, this is because the SOP dataset is much more fine-grained than the others, and MCL + PRISM [25] is focusing on it and not on the other datasets, where ProcSim occasionally outperforms it by a 10% difference. On the other hand, the PLG is less effective on SOP due to its higher class-to-sample ratio [40].

4.5. Effect of semantic noise

In Tab. 5, we compare the effect of uniform and semantic noise on the state-of-the-art methods. To assess the performance of LSD [60] and MCL + PRISM [25], we use the code provided by the authors with the proposed hyperparameters and include the obtained results on uniform noise. MCL + PRISM [25] requires an estimate of the noise probability, and although not specified, we used the ground truth probabilities, thus favoring this method. Doing so achieved the closest results to those reported by Liu *et al*. [25] for CUB200 [49] and Cars196 [22], but not for SOP [46]. We can observe that the results on the SOP dataset [46] for both types of noise are alike as expected. The reason being that semantic noise assigns a label chosen uniformly at random over only one of the twelve categories for SOP.

ProcSim attains the best performance in all cases. The competing approaches, especially MCL + PRISM [25], are more affected by semantic noise. These results show that semantic noise can be more harmful as it generates samples with wrong labels that are harder to spot. Instead, ProcSim shows the opposite behavior, which we attribute to the resolution of inter-class relationships.

5. Conclusions

This paper proposed ProcSim, an approach for training DML models for visual search on datasets with wrong annotations. ProcSim is a confidence-aware framework that is usable on top of any DML loss to improve its performance on noisy datasets. ProcSim is superior to existing alternatives when applied to datasets with injected noise without even fine-tuning for different types and levels of noise.

This work also introduced a new noise model inspired by plausible labeling mistakes. The proposed semantic noise model yields samples with wrong class labels that are harder to spot and can occasionally be more harmful than the omnipresent uniform noise model. While real noise is complex and a mixture of different types of noise, including but not limited to semantic errors, we believe this is a step towards closing the gap between real-world and simulated noise.

Acknowledgments

The authors thank Amit Kumar K C, Michael Huang, and René Vidal for fruitful discussions and useful suggestions. O.B. is part of CLOTHILDE ("CLOTH manIpulation Learning from DEmonstrations") which has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (Advanced Grant agreement No. 741930). O.B. thanks the European Laboratory for Learning and Intelligent Systems (ELLIS) for PhD program support.

References

[1] Devansh Arpit, Stanisław Jastrzundefinedbski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S. Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, and Simon Lacoste-Julien. A closer look at memorization in deep networks. In *ICML*, 2017. 2

[2] Aurélien Bellet, Amaury Habrard, and Marc Sebban. *Metric Learning*. Morgan & Claypool Publishers (USA), Synthesis Lectures on Artificial Intelligence and Machine Learning, pp 1-151, 2015. 2

[3] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *ICML*, 2009. 2

[4] Fatih Cakir, Kun He, Xide Xia, Brian Kulis, and Stan Sclaroff. Deep metric learning to rank. In *CVPR*, 2019. 7

[5] Thibault Castells, Philippe Weinzaepfel, and Jerome Revaud. Superloss: A generic loss for robust curriculum learning. In *NeurIPS*, 2020. 2, 3, 4, 6

[6] Binghui Chen and Weihong Deng. Hybrid-attention based decoupled metric learning for zero-shot image retrieval. In *CVPR*, 2019. 1

[7] Pengfei Chen, Ben Ben Liao, Guangyong Chen, and Shengyu Zhang. Understanding and utilizing deep neural networks trained with noisy labels. In *ICML*, 2019. 2

[8] Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *CVPR*, 2005. 3, 6, 7

[9] Roberto Cipolla, Yarin Gal, and Alex Kendall. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *CVPR*, 2018. 2

[10] Stanislav Dereka, Ivan Karpukhin, and Sergey Kolesnikov. Deep Image Retrieval is not Robust to Label Noise. In *CVPR*, 2022. 1, 2

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACLR*, 2019. 6, 7

[12] Aleksandr Ermolov, Leyla Mirvakhabova, Valentin Khrulkov, Nicu Sebe, and Ivan Oseledets. Hyperbolic Vision Transformers: Combining Improvements in Metric Learning. In *CVPR*, 2022. 2

[13] Weifeng Ge. Deep metric learning with hierarchical triplet loss. In *ECCV*, September 2018. 2

[14] Guy Hacohen and Daphna Weinshall. On the power of curriculum learning in training deep networks. In *ICML*, 2019. 2

[15] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *NeurIPS*, 2018. 2, 4

[16] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CVPR*, 2016. 5

[17] Sarah Ibrahimi, Arnaud Sors, Rafael Sampaio de Rezende, and Stéphane Clinchant. Learning with Label Noise for Image Retrieval by Selecting Interactions. In *WACV*, 2022. 2, 4, 6

[18] Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *ICML*, 2018. 3

[19] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 2019. 1

[20] Nazmul Karim, Mamshad Nayeem Rizve, Nazanin Rahnavard, Ajmal Mian, and Mubarak Shah. UniCon: Combating Label Noise Through Uniform Selection and Contrastive Learning. In *CVPR*, 2022. 2

[21] Sungyeon Kim, Dongwon Kim, Minsu Cho, and Suha Kwak. Proxy anchor loss for deep metric learning. In *CVPR*, 2020. 3

[22] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, Sydney, Australia, 2013. 1, 5, 6, 7, 8

[23] Kuang-Huei Lee, Xiaodong He, Lei Zhang, and Linjun Yang. Cleannet: Transfer learning for scalable image classifier training with label noise. In *CVPR*, 2018. 2, 4

[24] Junnan Li, Richard Socher, and Steven C.H. Hoi. DivideMix: Learning with Noisy Labels as Semi-supervised Learning. In *ICLR*, 2020. 2, 3, 6

[25] Chang Liu, Han Yu, Boyang Li, Zhiqi Shen, Zhanning Gao, Peiran Ren, Xuansong Xie, Lizhen Cui, and Chunyan Miao. Noise-resistant Deep Metric Learning with Ranking-based Instance Selection. In *CVPR*, 2021. 1, 2, 7, 8

[26] Dongju Liu and Jian Yu. Otsu method and k-means. In *International Conference on Hybrid Intelligent Systems*, 2009. 3

[27] Yueming Lyu and Ivor W. Tsang. Curriculum loss: Robust learning and generalization against label corruption. In *ICLR*, 2020. 3

[28] Dipu Manandhar, Muhammet Bastan, and Kim-Hui Yap. Semantic granularity metric learning for visual search. *Journal of Visual Communication and Image Representation*, 2020. 1, 2

[29] George A. Miller. WordNet: A Lexical Database for English. *Communications of the ACM*, 38(11):39–41, 1995. 5

[30] Yair Movshovitz-Attias, Alexander Toshev, Thomas K Leung, Sergey Ioffe, and Saurabh Singh. No fuss distance metric learning using proxies. In *ICCV*, 2017. 3, 6, 7, 8

[31] Kevin Musgrave, Serge Belongie, and Ser-Nam Lim. Pytorch metric learning. arXiv:2008.09164, 2020. 6

[32] David Novotny, Samuel Albanie, Diane Larlus, and Andrea Vedaldi. Self-supervised learning of geometrically stable features through probabilistic introspection. In *CVPR*, 2018. 2

[33] Nobuyuki Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 1979. 3, 6

[34] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison,

Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019. 5

[35] Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. Making deep neural networks robust to label noise: A loss correction approach. In *CVPR*, 2017. 2

[36] Qi Qian, Lei Shang, Baigui Sun, Juhua Hu, Hao Li, and Rong Jin. Softtriple loss: Deep metric learning without triplet sampling. In *ICCV*, 2019. 7

[37] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *ICML*, 2021. 4, 6, 7

[38] Marcus Rohrbach, Michael Stark, and Bernt Schiele. Evaluating knowledge transfer and zero-shot learning in a large-scale setting. In *CVPR*, 2011. 5

[39] Karsten Roth, Timo Milbich, Samarth Sinha, Prateek Gupta, Bjorn Ommer, and Joseph Paul Cohen. Revisiting training strategies and generalization performance in deep metric learning. In *ICML*, 2020. 6

[40] Karsten Roth, Oriol Vinyals, and Zeynep Akata. Integrating Language Guidance into Vision-based Deep Metric Learning. In *CVPR*, 2022. 2, 4, 5, 6, 8

[41] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *IJCV*, 2015. 4, 5

[42] Artsiom Sanakoyeu, Vasil Khalidov, Maureen S. McCarthy, Andrea Vedaldi, and Natalia Neverova. Transferring dense pose to proximal animal classes. In *CVPR*, 2020. 2

[43] Artsiom Sanakoyeu, Vadim Tschernezki, Uta Büchler, and Björn Ommer. Divide and conquer the embedding space for metric learning. In *CVPR*, 2019. 2

[44] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, 2015. 1, 2, 7

[45] Bing Shuai, Xinyu Li, Kaustav Kundu, and Joseph Tighe. Id-free person similarity learning. In *CVPR*, 2022. 1

[46] Hyun Oh Song, Yu Xiang, Stefanie Jegelka, and Silvio Savarese. Deep Metric Learning via Lifted Structured Feature Embedding. In *CVPR*, 2016. 1, 5, 6, 7, 8

[47] Yifan Sun, Changmao Cheng, Yuhan Zhang, Chi Zhang, Liang Zheng, Zhongdao Wang, and Yichen Wei. Circle loss: A unified perspective of pair similarity optimization. In *CVPR*, 2020. 7

[48] Brendan van Rooyen, Aditya Menon, and Robert C Williamson. Learning with Symmetric Label Noise: The Importance of Being Unhinged. In *NeurIPS*, 2015. 5

[49] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011. 3, 5, 6, 7, 8

[50] Xun Wang, Xintong Han, Weilin Huang, Dengke Dong, and Matthew R Scott. Multi-similarity loss with general pair weighting for deep metric learning. In *CVPR*, 2019. 3, 4, 5, 6, 7, 8

[51] Xiaobo Wang, Shuo Wang, Hailin Shi, Jun Wang, and Tao Mei. Co-mining: Deep face recognition with noisy labels. In *ICCV*, 2019. 2, 4

[52] Xun Wang, Haozhi Zhang, Weilin Huang, and Matthew R Scott. Cross-batch memory for embedding learning. In *CVPR*, 2020. 7

[53] Yuchen Wei, Son Tran, Shuxiang Xu, Byeong Kang, and Matthew Springer. Deep Learning for Retail Product Recognition: Challenges and Techniques. *Computational Intelligence and Neuroscience*, 2020. 1

[54] Chao-Yuan Wu, R Manmatha, Alexander J Smola, and Philipp Krahenbuhl. Sampling matters in deep embedding learning. In *ICCV*, 2017. 2

[55] Xiaobo Xia, Tongliang Liu, Nannan Wang, Bo Han, Chen Gong, Gang Niu, and Masashi Sugiyama. Are anchor points really indispensable in label-noise learning? In *NeurIPS*, 2019. 2

[56] Jiexi Yan, Lei Luo, Cheng Deng, and Heng Huang. Adaptive hierarchical similarity metric learning with noisy labels. *IEEE Transactions on Image Processing*, 2023. 1, 2, 7, 8

[57] Jiexi Yan, Lei Luo, Chenghao Xu, Cheng Deng, and Heng Huang. Noise Is Also Useful: Negative Correlation-Steered Latent Contrastive Learning. In *CVPR*, 2022. 2

[58] Zhibo Yang, Muhammet Bastan, Xinliang Zhu, Doug Gray, and Dimitris Samaras. Hierarchical Proxy-based Loss for Deep Metric Learning. In *WACV*, 2022. 2

[59] Xingrui Yu, Bo Han, Jiangchao Yao, Gang Niu, Ivor Tsang, and Masashi Sugiyama. How does disagreement help generalization against label corruption? In *ICML*, 2019. 2, 4

[60] Zelong Zeng, Fan Yang, Zheng Wang, and Shin'ichi Satoh. Improving Generalization of Metric Learning via Listwise Self-distillation. arXiv:2206.08880, 2022. 1, 2, 4, 8

[61] Andrew Zhai and Hao-Yu Wu. Classification is a strong baseline for deep metric learning. In *BMVC*, 2019. 7