Deep Semi-supervised Metric Learning with Mixed Label Propagation

Furen Zhuang[1,2,*], Pierre Moulin[1]
[1]University of Illinois at Urbana-Champaign, Dept of ECE, Urbana, Illinois 61801
[2]Institute for Infocomm Research (I^{2}R),A*STAR, Singapore 138632
zhuang_furen@i2r.a-star.edu.sg,moulin@ifp.uiuc.edu

Abstract

Metric learning requires the identification of far-apart similar pairs and close dissimilar pairs during training, and this is difficult to achieve with unlabeled data because pairs are typically assumed to be similar if they are close. We present a novel metric learning method which circumvents this issue by identifying hard negative pairs as those which obtain dissimilar labels via label propagation (LP), when the edge linking the pair of data is removed in the affinity matrix. In so doing, the negative pairs can be identified despite their proximity, and we are able to utilize this information to significantly improve LP's ability to identify far-apart positive pairs and close negative pairs. This results in a considerable improvement in semi-supervised metric learning performance as evidenced by recall, precision and Normalized Mutual Information (NMI) performance metrics on Content-based Information Retrieval (CBIR) applications.

1. Introduction

Image data have proliferated owing to ubiquitous camera ownership, widespread internet connectivity, and broad availability of software applications. The design of efficient CBIR systems is imperative in managing information content. Traditional image retrieval systems use textual annotations, but such an approach require manual labor which may be not be cost-effective. On the other hand, CBIR retrieves relevant content from an image query, alleviating the need for human annotation.

Metric learning learns a transformation mapping similar data closer than dissimilar ones. This makes it easier to cluster data [37] and also to shortlist similar data by their proximity to the query. As such, metric learning algorithms are at the cornerstone of most state-of-the-art CBIR designs [23]. As the amount of data handled in CBIR systems is very large, there is interest in utilizing the vast quantities of readily available unlabeled data to improve the metric learning training process.

Recent advances have asserted the importance of retaining within-class variance in training generalizable representations [27], particularly because the information encapsulated in representations which are irrelevant in discriminating between training labels may be important in discriminating between unseen test classes. Since close positive pairs results in little loss, there is a need to identify far-apart positive pairs for training. The paper [43] uses the method from [9] to propagate labels along manifolds in order to identify such points, by assuming data which are nearest neighbor pairs but not mutual nearest neighbors as close dissimilar pairs. However this method does not utilize label information and may not accurately identify hard negatives [3, 17, 39, 40], which are pairs of data which are close but have different labels. Label propagation (LP) using mis-identified edges would lead to inaccuracies in the model trained.

This paper proposes a novel method to identify hard negative pairs in a semi-supervised setting. Specifically, given a few labeled points from each class and an abundance of unlabeled data, we propose to identify hard negative pairs as those which obtain dissimilar labels via LP when the edge linking the two elements of the pair is removed in the affinity matrix. We obtain the dissimilarity weights of edges under this assumption quickly and efficiently, without the costly use of repeated LP to calculate these weights.

We use this negative edge information in a novel mixed LP algorithm which is able to utilize both positive and negative edge information. Specifically, the method encourages data linked by positive edges to have the same pseudolabel and data linked by negative edges to have different pseudolabels. Like LP, our mixed LP optimization problem can be solved with conjugate gradient (CG), allowing it to scale to large datasets.

As our obtained pseudolabels capture information on far-apart positive pairs and close negative pairs, they yield a significant improvement in semi-supervised metric learning. We showcase this in a CBIR setting under recall, pre-

* Work done while a student at UIUC.

cision and Normalized Mutual Information (NMI) performance metrics. This shows that our method is able to more effectively rank the database in relation to the query and return relevant articles near the top of this list.

2. Related Work

2.1. Semi-supervised metric learning

Consider a training set $\mathcal{X} = \mathcal{X}_L \cup \mathcal{X}_U$, consisting of a set of N_L labeled examples $\mathcal{X}_L = \{x_i\}_{i=0}^{N_L}$ with corresponding labels $\mathcal{Y}_L = \{y_i\}_{i=0}^{N_L}$, $y_i \in \{1, ..., C\}$ and a set of $N - N_L$ unlabeled examples $\mathcal{X}_U = \{x_i\}_{i=N_L+1}^{N}$ with unseen labels $\mathcal{Y}_U = \{y_i\}_{i=N_L+1}^{N}$. The one-hot label matrix $\mathbf{Y} \in \{0,1\}^{N \times C}$ is defined as

$$\mathbf{Y}_{ij} := \begin{cases} 1, & \text{if } y_i \in \mathcal{Y}_L \wedge y_i = j \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

Many semi-supervised metric learning algorithms [2,28] employ the triplet loss:

$$\mathcal{L}(X) = \sum_{\{(i,j,k)\,:\, y_i = y_j \neq y_k\}} \mathcal{L}(x_i, x_j, x_k), \tag{2}$$

$$\mathcal{L}(x_i, x_j, x_k) = [D(x_i, x_j) - D(x_i, x_k) + m]_+, \tag{3}$$

$$D(x_i, x_j) = \|g(x_i) - g(x_j)\|_2\,, \tag{4}$$

where m is a margin parameter and g is a function mapping the samples into embeddings. However, it is intractable to calculate the loss for all triplets in the database and it is difficult to sample loss-inducing triplets [35]. Furthermore, the triplet loss encourages similar points to be as close as possible, encouraging data of the same class to collapse into a single point. This reduces variance between data that is crucial for the embeddings to be similarity-preserving in the zero-shot setting [27].

Proxies [23] can be used to remedy this situation, specifically by assigning a single anchor point to be the representative for each class. This also allows the computational complexity to be much lower because each data point is compared to a small number of proxies. Moreover, the within-class variance is not excessively reduced because its loss only enforces that a data point is closer to the proxy of its predicted class than to other proxies. This allows the embeddings to retain variance which can be used to discriminate between classes unseen during training.

Proximity is often used to identify similar pairs [28, 30] but close pairs result in little loss, and it is difficult to reliably identify far-apart similar pairs. The paper [43] proposes to construct a graph with edges joining mutual nearest neighbor pairs and to propagate the given labels along the manifolds they lie on. In so doing, they hope to identify similar data which are far from a proxy, while also identifying dissimilar points which may be close to the proxy but are not on the same manifold as the proxy.

In effect the paper uses the assumption [9] that pairs of data which are nearest neighbor pairs but not mutual nearest neighbors, are negative pairs. However this assumption does not utilize the labels provided, and we show that we can improve the identification of far-apart positive pairs and close negative pairs by identifying hard negative pairs as those which obtain dissimilar labels via LP when the edge linking the two elements of the pair is removed in the affinity matrix. This negative edge information is used in our novel mixed LP method to significantly improve the CBIR performance of semi-supervised metric learning.

2.2. Label Propagation

LP aims to train a classification function $\mathbf{f}$ such that i) vertices that are linked by an edge are encouraged to have the same label and ii) the original labels on labeled nodes are maintained in $\mathbf{f}$. To that end, LP [42] utilizes an objective function of the form

$$\min_{\mathbf{f}} \{\mathcal{S}(\mathbf{f}) + \mu \mathcal{C}(\mathbf{f_L})\} \tag{5}$$

where $\mathcal{S}(\mathbf{f})$ is the smoothness term encouraging linked vertices to have the same label, $\mathcal{C}(\mathbf{f_L})$ is a loss function on labeled nodes penalizing the divergence of output labels from ground truth, and μ is a positive hyperparameter balancing the trade-off between the two terms.

The smoothness term is usually of the form

$$\mathcal{S}(\mathbf{f}) = \mathbf{f}^\top \mathbf{L} \mathbf{f} \tag{6}$$

where $\mathbf{L}$ is the graph Laplacian [22],

$$\mathbf{L} = \mathbf{D} - \mathbf{W}, \tag{7}$$

$$\mathbf{D} = \text{diag}(\mathbf{W}\mathbf{1}), \tag{8}$$

$\mathbf{W}$ is a symmetric, non-negative affinity matrix, and $\mathbf{1}$ is the vector with each element equal to 1.

The label loss function is of the form

$$\mathcal{C}(\mathbf{f}) = (\mathbf{f} - \mathbf{y})^\top (\mathbf{f} - \mathbf{y}) \tag{9}$$

and the overall loss function can be written as

$$\mathcal{Q}(\mathbf{F}) = \frac{1}{2}\text{tr}(\mathbf{F}^\top \mathbf{L} \mathbf{F}) + \frac{\mu}{2}\text{tr}(\mathbf{F} - \mathbf{Y})^\top (\mathbf{F} - \mathbf{Y}), \tag{10}$$

where $\mathbf{F} = [f_1, ..., f_C] \in \mathbb{R}^{N \times C}$ and f_c denotes the propagated labels for class c.

Differentiating with respect to $\mathbf{F}$ and setting the derivative to zero, we obtain the minimizing $\mathbf{F}^*$ as

$$\mathbf{F}^* = \mu(\mathbf{L} + \mu \mathbf{I})^{-1} \mathbf{Y}. \tag{11}$$

Since $(\mathbf{L} + \mu \mathbf{I})$ is positive definite for any $\mu > 0$, $\mathbf{F}^*$ can be efficiently obtained via conjugate gradient from the linear system

$$(\mathbf{L} + \mu \mathbf{I}) \mathbf{F}^* = \mu \mathbf{Y}, \tag{12}$$

The pseudolabels are typically [42] assigned as

$$\hat{y_i} = \arg\max_j F^*_{ij}. \tag{13}$$

2.3. Hard Negative Mining

Affinity matrices are usually constructed with k-nearest neighbors. Performing LP directly on these graphs would tend to result in data which are close together being assigned the same pseudolabel. Identifying hard negative pairs would improve the propagation of labels in LP allowing it to more effectively assign the correct pseudolabels to far-apart positive pairs and close negative pairs, and in turn yield better semi-supervised metric learning performance.

Hard negatives cannot be identified directly using LP class predictions. This is because LP assumes close pairs have the same label via its smoothness term – LP asserts every pair which is "hard" to be "positive". We overcome this limitation by comparing the propagated labels of two points if the edge between them was omitted in the LP affinity matrix. Removal of this edge allows the propagated labels to be of different classes and allows the identification of hard negatives.

Utilizing negative samples is central to many metric learning methods [14, 24, 29], and sampling strategies are well-studied in these works. However these methods have access to full label information. In the semi-supervised setting, it is difficult to find hard negative samples because closer pairs are more likely to be similar. In recent years, there has been an interest in utilizing hard negatives for unsupervised contrastive learning [1, 12, 31]. These methods usually identify points which the model already believes to be dissimilar as negative points. In contrast, our method is able to identify negative pairs which the model believes to be similar and hence result in a significant change to the model.

The paper [9] proposes to identify points which are nearest neighbor pairs but not mutual nearest neighbors to be negative examples. However this method does not utilize label information and we show that our method is able to outperform [9] significantly.

2.4. Mixed Label Propagation

Incorporating negative edge information into LP is not trivial. To our knowledge, there are five prominent papers on mixed LP [4,5,33,41,44]. Of these, [44] utilizes negative label information instead of negative edge information and [33] performs mixed LP on the binary classification task and cannot be directly extended to multi-class classification.

Two of the other methods have high complexity, restricting their utility to very small datasets. The quadratic program presented in [4] has a large number of inequality constraints and has $O(N^3)$ time complexity when solved using an interior-point method [38]. Similarly, [5] has a $O(N^3)$ time complexity because the paper proposes to propagate edge information to the other $N \times N$ edges.

On the other hand, LPDR [41] may encourage erroneous label assignments [4] as we discuss in section 3.2. In the same section, we formulate a scalable LP algorithm which can efficiently harness the dissimilarity information we generate.

3. Proposed Method

3.1. Obtaining Hard Negative Weights

We use a neural network with parameters ϕ to extract L_2-normalized features $\mathbf{v}_i \in \mathbb{R}^d$ from each input $\mathbf{x}_i$:

$$\mathbf{v}_i = g_\phi(x_i). \tag{14}$$

Denoting the k nearest neighbors of $\mathbf{v}$ in Euclidean distance metric by $\mathrm{NN}_k(\mathbf{v})$, we construct a sparse affinity matrix $\mathbf{A} \in \mathbb{R}^{N\times N}$ with elements

$$A_{ij} := \begin{cases} [\mathbf{v}_i^\top \mathbf{v}_j]_+^\gamma, & \text{if } i \neq j \wedge \mathbf{v}_i \in \mathrm{NN}_k(\mathbf{v}_j) \\ 0, & \text{otherwise.} \end{cases} \tag{15}$$

This matrix can be constructed efficiently for large N using the *faiss* library [11], with $O(\log^2 N)$ complexity. With $\mathbf{W} = \mathbf{A} + \mathbf{A}^\top$, we perform a slightly modified version of LP, replacing μ in (10) with a diagonal $N \times N$ matrix $\mathbf{U}$ where

$$\mathbf{U}_{ii} := \begin{cases} \mu, & \text{if } y_i \in \mathcal{Y}_L \\ 0, & \text{otherwise.} \end{cases} \tag{16}$$

In effect we are removing the L_2-regularization on unlabeled data points in the original LP. With this change, the LP objective on unlabeled data reduces to only the smoothness term (6). Differentiating with respect to $\mathbf{F}_i$ and equating to zero, we get

$$\mathbf{F}_i^* = \frac{\sum_j w_{ij}\mathbf{F}_j^*}{\mathbf{D}_{ii}}, \tag{17}$$

and hence the unlabeled $\mathbf{F}_i^*$ is a weighted average of its neighbors' labels. We can then calculate $\mathbf{Z}_{i,j}$, the weighted average of all neighbors of i excluding neighbor j, as depicted in Fig. 1,

$$\mathbf{Z}_{i,j} = \frac{\sum_l w_{il}\mathbf{F}_l^* - w_{ij}\mathbf{F}_j^*}{\mathbf{D}_{ii} - w_{ij}} \tag{18}$$

$$= \frac{\mathbf{D}_{ii}\mathbf{F}_i^* - w_{ij}\mathbf{F}_j^*}{\mathbf{D}_{ii} - w_{ij}}. \tag{19}$$

Since the labels obtained from LP can be shown to be a weighted average of its first-order neighbors, and $\mathbf{Z}_{i,j}$ is a weighted average of the first-order neighbors excluding j, we can view $\mathbf{Z}_{i,j}$ as an approximation of the propagated labels we would have obtained from LP if edge (i, j) was removed.

We estimate the class probabilities as

$$\tilde{\mathbf{Z}}_{i,j} = \mathrm{softmax}[\lambda(\mathbf{D}_{ii}\mathbf{F}_i^* - w_{ij}\mathbf{F}_j^*)], \tag{20}$$

where λ is a temperature-scaling hyperparameter.

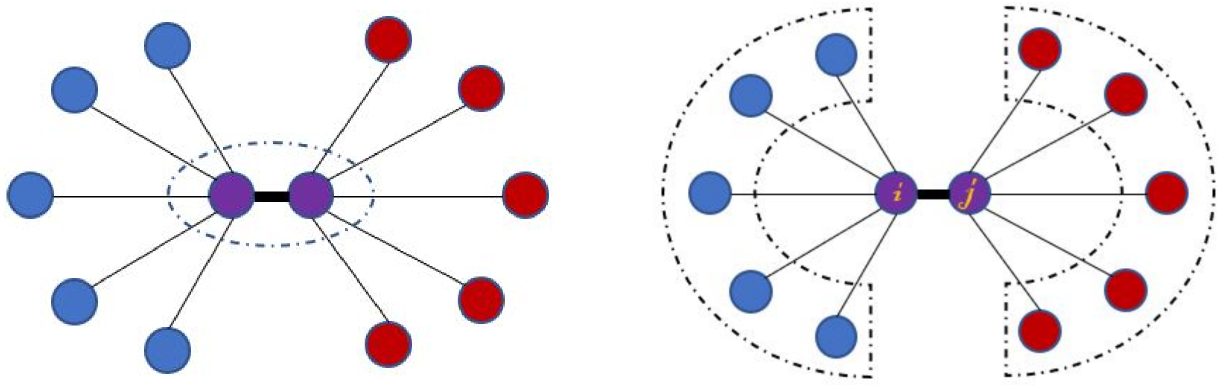

Figure 1. (left) The smoothness term ensures that pairs which are close have similar labels and it is difficult to identify them as dissimilar pairs. (right) However the influence of the red nodes on the blue nodes through the offending edge is diminished due to the increased geodesic distance and it is easier to identify dissimilar pairs by comparing their first-order neighbors.

The probability that nodes i and j are dissimilar can then be calculated as $p_{ij}^{dis} = 1 - \tilde{\mathbf{Z}}_{i,j}^{\top}\tilde{\mathbf{Z}}_{j,i}$.

We define the confidence weights [10]

$$\omega(\tilde{\mathbf{Z}}_{i,j}) = 1 - \frac{H(\tilde{\mathbf{Z}}_{i,j})}{\log(C)}, \tag{21}$$

where $H(\cdot)$ is the entropy function, and obtain dissimilarity edge weights as

$$W_{ij}^{(dis)} = \omega(\tilde{\mathbf{Z}}_{i,j})\omega(\tilde{\mathbf{Z}}_{j,i})p_{ij}^{dis}. \tag{22}$$

A high dissimilarity edge weight is hence assigned if the first-order neighbors of the pair i and j confidently belong to different classes.

3.2. Proposed Mixed Label Propagation

Given two positive matrices $\mathbf{W}$ and $\mathbf{W}^{(dis)}$ representing similar and dissimilar edge weights respectively, we propose our mixed LP method as the following optimization problem:

$$\min_{\mathbf{G}}\{\frac{1}{2}\text{tr}(\mathbf{G}^{\top}\mathbf{L}\mathbf{G}) + \frac{1}{2}\|\mathbf{U}(\mathbf{G}-\mathbf{Y})\|_F + \frac{\beta}{2}\mathcal{D}(\mathbf{G})\}, \tag{23}$$

where

$$\mathcal{D}(\mathbf{G}) = \sum_c^C \sum_{i,j} W_{ij}^{(dis)}(G_{ic}+G_{jc})^2, \tag{24}$$

is the dissimilarity loss.

The dissimilarity loss function used in LPDR [41] is

$$\mathcal{D}(\mathbf{G}) = -\sum_c^C \sum_{i,j} W_{ij}^{(dis)}(G_{ic}-G_{jc})^2. \tag{25}$$

Even though our loss function looks similar to LPDR's, there are important differences.

Consider two points i and j with differing labels. The correct assignment is $G_{ic} = G_{jc} = 0$ for $c \neq \hat{y}_i, \hat{y}_j$. However we see that the LPDR loss function erroneously enforces $G_{ic} \neq G_{jc}$ for such c.

On the other hand, for any c, our loss function gives no penalty if $G_{ic} = G_{jc} = 0$, gives a small penalty if either G_{ic} or G_{jc} equals 1, but gives an exponentially large penalty if $G_{ic} = G_{jc} = 1$, which is the main intention of the dissimilarity loss. Therefore our dissimilarity objective is able to avoid strongly penalizing the intended outcome, while effectively enforcing $\hat{y}_i \neq \hat{y}_j$.

The overall optimization problem of our mixed LP formulation can be efficiently solved in $O(N)$ time using conjugate gradient, in contrast to the quadratic program formulated in [4] which involves a large number of inequality constraints and has a high time complexity. A dense $N \times N$ matrix is also never created, allowing our method to retain $O(Nk)$ memory complexity in contrast to [5] where a dense $N \times N$ matrix is created.

3.3. Deep semi-supervised metric learning

In each epoch, we use the dissimilarity edge weights from (22) in our mixed LP method (23), to obtain propagated labels $\mathbf{G}^*$ and in turn normalized class probabilities $\tilde{\mathbf{G}}_i := \frac{\mathbf{G}_i^*}{\|\mathbf{G}_i^*\|_1}$, with confidence weights $\omega(\tilde{\mathbf{G}}_i^*)$.

A set of L_2-normalized proxies $\{p_c\}_{c=0}^{C}, p_c \in \mathbb{R}^{\delta}$ is generated and maintained, where δ is the dimension of the embeddings z, given as the final output at the top of the neural network. The pseudolabels $\hat{y}_i = \arg\max_c G_{ic}^*$ and confidence weights $\omega(\tilde{\mathbf{G}}_i^*)$ are then used in the following loss function from [43] to update the model:

$$\mathcal{L} = \frac{1}{C}\sum_i \mathcal{L}(z_i, \omega(\tilde{\mathbf{G}}_i^*)), \tag{26}$$

$$\mathcal{L}(z_i, \omega(\tilde{\mathbf{G}}_i^*)) = \omega(\tilde{\mathbf{G}}_i^*)(\mathcal{L}^+(z_i) + \mathcal{L}^-(z_i)), \tag{27}$$

$$\mathcal{L}^+(z_i) = \log\{1 + \exp[-\epsilon(z_i^\top p_{\hat{y}_i} - b)]\}, \tag{28}$$

$$\mathcal{L}^-(z_i) = \sum_{c \neq y_i} \log\{1 + \exp[\epsilon(z_i^\top p_c + b)]\}, \tag{29}$$

with ϵ and b as hyperparameters.

Given the initial affinity matrix built using *faiss*, our space complexity is $O(NkC)$ if fully-vectorized code is used, but can be reduced to $O(Nk+NC)$ using loops, with time complexity $O(NkC)$. Our method is linear in N for both space and time complexity and is therefore fairly scalable.

4. Experiments

4.1. Datasets

We mainly use 6 image datasets: FGVC-Aircraft [19], Cars-196 [15], CIFAR100 [16], CUB-200-2011 [34], Stanford Dogs [13] and Oxford-IIIT Pets [26]. For the accuracy comparison in Table 3, we additionally include results on German Traffic Sign Recognition Benchmark (traffic_sign) [7] and vgg_flowers [25].

For each class, the labels for 5 data points are given in all our experiments. For hyperparameters, we use $\beta = 1, \gamma = 3$, $\epsilon = 32$, $\lambda = 4$, $b = 0.1$, $\mu = \frac{1}{99}$, learning rate 10^{-4} and $k = 50$ in all our experiments unless otherwise stated.

We use the BN-Inception [8] architecture as the base model to obtain features. This architecture is commonly used in metric learning [43]. For our method and DSSML, the AdamW optimizer [18] was used and the weight decay was set at 10^{-4}.

4.2. CBIR Experiments

For the CBIR experiments using our metric learning method shown in Figures 2 and 3 and Table 1, we use the same setup as DSSML [43]. We merge the training, validation and test sets of the datasets and use 40% of the classes are use as the training set, 20% as the validation set, and the remaining 40% of classes as the test set. The training, validation and test set classes are hence disjoint.

We compare our results against state-of-the-art semi-supervised algorithms DSSML [43] and SDEC [30] ; fully-supervised algorithms Proxy-Anchor [14], SoftTriple [29] and ProxyNCA++ [32]; and unsupervised algorithm DEC [36].

All deep methods were run for 20 epochs using pre-trained BN-Inception [8] as the base architecture with a batch size of 32, except SDEC which used a batch size of 90. Finetuning was done for our method and DSSML for 5 epochs using only the given labelled data before running these 20 epochs. Default hyperparameter settings and optimizers for all competing methods were used. The best epoch for all methods was chosen as the epoch which gave the best P@8 (precision at 8) on the validation set. Embeddings of dimension $\delta = 64$ were used.

We use three evaluation metrics: Normalized Mutual Information (NMI), recall at k (R@k) and precision at k (P@k) [43].

NMI evaluates the clustering quality. Let $\mathcal{C} = \{h_i\}_{i=1}^C, h_i = \{j : y_j = i\}$ be the cluster assignments based on the true labels of the data. We use k-Means clustering on the embeddings to obtain a similar cluster assignment $\Omega = \{\omega_i\}_{i=1}^C, \omega_i = \{j : q_j = i\}$, where q_j is the cluster number of sample j. Let the number of test samples be N_{test}. Then NMI is calculated [20] as

$$\text{NMI} = \frac{2 \times I(\Omega, \mathcal{C})}{H(\Omega) + H(\mathcal{C})}, \tag{30}$$

where we have defined the mutual information

$$I(\Omega, \mathcal{C}) = \sum_k \sum_j \frac{|\omega_k \cap y_j|}{N_{test}} \log \frac{N_{test}|\omega_k \cap y_j|}{|\omega_k||y_j|}, \tag{31}$$

and the entropies

$$H(\Omega) = -\sum_k \frac{|\omega_k|}{N_{test}} \log \frac{|\omega_k|}{N_{test}}, \tag{32}$$

$$H(\mathcal{C}) = -\sum_k \frac{|h_k|}{N_{test}} \log \frac{|h_k|}{N_{test}}. \tag{33}$$

Denote by $r_j^{(i)}$ the class label of the j-th retrieved item of the i-th sample. Let

$$\text{R@k} = \frac{1}{N_{test}} \sum_{i=0}^{N_{test}} \mathbb{1}\{y_i \in \{r_1^{(i)}, ..., r_k^{(i)}\}\}, \tag{34}$$

which measures the average probability that at least one of the top k retrieved results has the same class label as the query.

Also

$$\text{P@k} = \frac{1}{N_{test}} \sum_{i=0}^{N_{test}} \frac{1}{k} \sum_{j=0}^{k} \mathbb{1}\{r_j^{(i)} = y_i\}, \tag{35}$$

measures the average fraction of the top k retrieved results that has the same class label as the query.

From Figures 2 and 3, we see that the use of our negative mining approach together with our mixed LP method outperforms state-of-the-art results on semi-supervised metric learning. The NMI results in Table 1 also support the observation that the incorporation of our proposal is able to

Table 1. NMI comparisons, best results are shown in bold.

	aircraft (family)	cars	CIFAR100	cub	dogs	pets
Ours	**33.837**	**36.397**	**39.377**	**57.780**	**72.677**	**77.039**
DSSML	32.798	34.755	39.150	55.824	71.427	74.867
SDEC	23.414	21.678	21.570	39.487	54.429	75.845
Proxy-Anchor	30.660	32.239	34.085	51.688	65.668	76.966
ProxyNCA++	20.279	27.954	20.808	42.209	55.949	50.420
SoftTriple	30.018	33.800	34.787	53.289	68.663	73.065
DEC	21.163	22.148	20.165	37.508	53.645	76.039

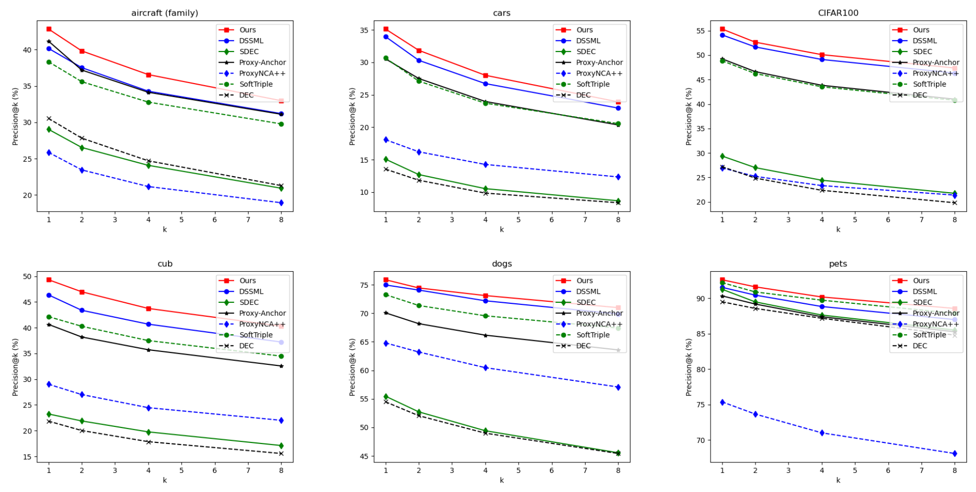

Figure 2. Comparison of precision against state-of-the-art methods on real-world public datasets.

considerably improve the semi-supervised metric learning results. These results demonstrate that our proposal is able to significantly advance state-of-the-art results in important and practical areas of high interest such as semi-supervised metric learning.

4.3. Additional comparison with recent state-of-the-art

Dutta et al [2] also considers a semi-supervised metric learning method. However the main points of difference are: 1) The paper [2] requires the creation and inversion of a dense $N \times N$ matrix which necessitates sampling, while our method does not involve the creation of such a matrix and we are able to avoid direct matrix inversion. Hence we are able to utilize all provided data. 2) The paper [2] assumes samples which rank $k/2$ apart in affinity are dissimilar. However setting k appropriately may require knowledge of the distribution of the dataset as a small k could result in erroneously declaring a same-class sample as negative, while a large k could mean only far apart negative pairs are used in forming triplets. It may also be difficult to choose k if the dataset is class imbalanced. Unlike [2], our method does not require knowledge of the distribution of the dataset, and works well even with class imbalanced datasets.

Table 2. Comparison against Dutta et al on CUB-200.

Method	**NMI**	**R@1**	**R@2**	**R@4**	**R@8**
Dutta et al [2]	54.0	44.8	56.9	69.1	79.9
Ours	**59.5 $\pm$ 0.8**	**49.9$\pm$ 1.3**	**63.1 $\pm$ 1.2**	**74.7 $\pm$ 0.8**	**84.1 $\pm$ 0.6**

To compare against [2], we run our method for CUB-200 using the experimental settings in [2]. We use BN-Inception, which is comparable to the GoogLeNet with R-MAC used by [2]. We also run our method for only 20 epochs which is lower than the 200 epochs run by [2]. We average our results over 10 runs and present them in Table 2. Our superior results support the claim that our method can alleviate the above-mentioned issues of [2].

As our pseudolabeling method could be used in the formation of triplets in [2], our methods are complementary.

4.4. Ablation studies on our hard negative mining method

We show that the negative pairs we mined are effective in improving LP performance. Four methods are compared: 1) Original LP is the method implemented in [42].

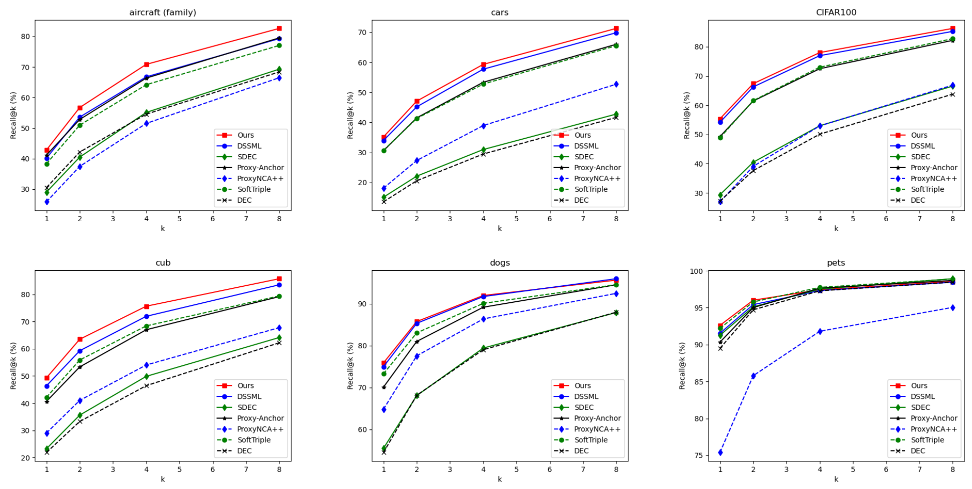

Figure 3. Comparison of recall against state-of-the-art methods on real-world public datasets.

Table 3. Accuracy comparisons between methods identifying hard negatives. Best results are shown in bold.

	Original LP	MoM	Control	Ours
aircraft (family)	0.195 ± 0.006	0.098 ± 0.002	0.197 ± 0.005	$\mathbf{0.207 \pm 0.005}$
aircraft (manufacturer)	0.189 ± 0.019	0.113 ± 0.002	0.202 ± 0.022	$\mathbf{0.218 \pm 0.020}$
aircraft (variant)	0.198 ± 0.005	0.101 ± 0.002	0.207 ± 0.003	$\mathbf{0.209 \pm 0.003}$
cars	0.190 ± 0.002	0.103 ± 0.001	0.192 ± 0.002	$\mathbf{0.193 \pm 0.001}$
CIFAR100	0.315 ± 0.004	0.210 ± 0.003	0.299 ± 0.003	$\mathbf{0.338 \pm 0.003}$
cub	0.441 ± 0.005	0.158 ± 0.002	0.458 ± 0.004	$\mathbf{0.468 \pm 0.003}$
dogs	0.620 ± 0.005	0.234 ± 0.002	0.622 ± 0.005	$\mathbf{0.639 \pm 0.004}$
pets	0.780 ± 0.011	0.242 ± 0.006	0.779 ± 0.014	$\mathbf{0.832 \pm 0.005}$
traffic_sign	0.488 ± 0.010	0.266 ± 0.004	0.482 ± 0.012	$\mathbf{0.498 \pm 0.010}$
vgg_flowers	0.750 ± 0.009	0.149 ± 0.002	0.760 ± 0.006	$\mathbf{0.797 \pm 0.003}$

2) MoM [9] considers points which are nearest neighbors but not mutual nearest neighbors as dissimilar pairs and are given $W_{ij}^{(dis)} = 1$ for such pairs. 3) Control implements our method except $\tilde{Z}_{i,j}$ is replaced with the L_1-normalized $\mathbf{F}_i, \forall j$ [10, 43], which is the usual approach to calculate the class probabilities of a data point x_i [10, 43]. This corresponds to calculating the dissimilarity between pairs directly, instead of their first-order neighbors. 4) Ours is our method.

For methods 2, 3 and 4, the dissimilarity weights are used in our mixed LP method to obtain accuracy. This accuracy is calculated as:

$$\text{accuracy} = \frac{1}{N}\sum_{i}^{N} \mathbb{1}\{\hat{y}_i = y_i\}, \tag{36}$$

The entire dataset is used for each experiment. The experiments are repeated ten times with different random selections of the five labeled data points and the 95% confidence interval is given. Table 3 shows that our method obtains consistent and significant improvements over all compared methods.

MoM results in a decline in performance from the original LP method because it does not utilize labels and could confuse the predictions of the original LP method which uses labels and is able to generate more reliable predictions.

'Control' is the straightforward extension of original LP to directly use its predictions in generating dissimilarity weights. It is able to obtain a marginal improvement over the original LP method. This could be due to its consideration of all propagated labels over all the classes allowing a slight increase in accuracy, while LP is only able to consider the propagation of each class label in isolation.

Our method is able to weaken the label smoothing constraint across potential dissimilar edges and provide dissimilar edge weights which are contradictory to the LP predictions and hence result in a considerable change to the LP objective, and a significant improvement in accuracy. This is reflected in the great improvement in accuracy afforded by our method in Table 3.

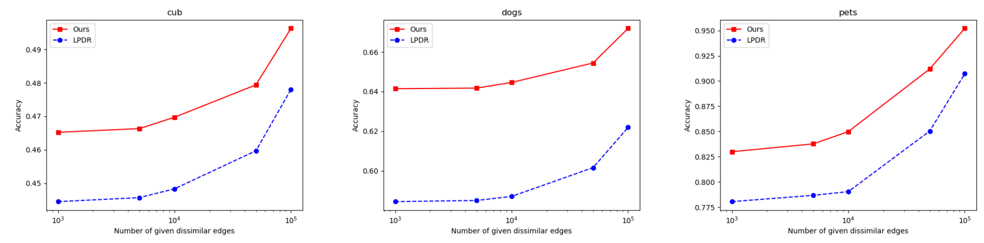

Figure 4. Comparison of accuracy against LPDR as more close negative edges are provided by an oracle.

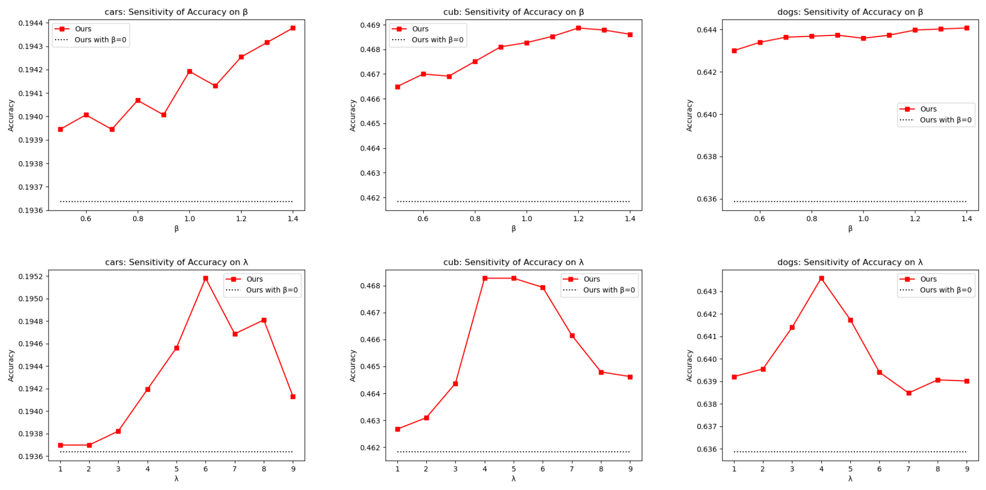

Figure 5. Sensitivity analysis of accuracy on β and λ.

4.5. Ablation studies on our mixed LP method

As the proposed methods in [4] and [5] have high time complexities and are not able to run on the datasets in reasonable time, we compare our results primarily with LPDR [41].

In the experiments shown in Figure 4, an oracle randomly identifies close negative edges between the kNN neighbors which are not already given by the labels. The accuracy is calculated as in Equation (36).

Using LPDR's notation, the default hyperparameter of LPDR's $\gamma = 100$ was used and we set LPDR's $\lambda = 1$ which is equivalent to our hyperparameter of $\beta = 1$.

From Figure 4, we see that our mixed LP method obtains a significantly higher accuracy than LPDR. LPDR is unable to effectively use the negative edge weights provided because it incorrectly enforces $G_{ic} \neq G_{jc}$ for any $c \neq \hat{y}_i, \hat{y}_j$ when the correct assignment is $G_{ic} = G_{jc} = 0$ for such c. Our method is able to avoid strongly penalizing intended outcomes such as this.

4.6. Ablation studies on hyperparameters

We conduct ablation studies on datasets cub, cars and dogs for the two hyperparameters in our method, β and λ. For each experiment, the default setting of $\beta = 1$ or $\lambda = 4$ is maintained, while the tested hyperparameter is varied. The dissimilarity weights are calculated using our method (22) and used with our mixed LP method (23). The accuracy is calculated from (36).

Figure 5 shows the sensitivity of accuracy on the two hyperparameters, β and λ. The accuracy is not very sensitive to β but is however sensitive to λ. In this work we fixed λ to a moderate value of 4, but results could be further improved by tuning λ for each dataset using a more sophisticated calibration method such as [6, 21].

5. Conclusion

In this work, we aimed to identify far-apart positive pairs and close negative pairs to improve the CBIR performance of semi-supervised metric learning. We achieved this by assuming hard negatives pairs as k-NN pairs which obtain dissimilar LP labels when the edge between them is removed and incorporated this into a novel mixed LP method. We showed that the resulting pseudolabels obtains state-of-the-art results in the semi-supervised metric learning application of CBIR.

References

[1] Ching-Yao Chuang, Joshua Robinson, Yen-Chen Lin, Antonio Torralba, and Stefanie Jegelka. Debiased contrastive learning. *Advances in neural information processing systems*, 33:8765–8775, 2020.

[2] Ujjal Kr Dutta, Mehrtash Harandi, and C Chandra Shekhar. Semi-supervised metric learning: A deep resurrection. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(8):7279–7287, May 2021.

[3] Fartash Faghri, David J Fleet, Jamie Ryan Kiros, and Sanja Fidler. Vse++: Improving visual-semantic embeddings with hard negatives. *arXiv preprint arXiv:1707.05612*, 2017.

[4] Andrew B. Goldberg, Xiaojin Zhu, and Stephen J. Wright. Dissimilarity in graph-based semi-supervised classification. In *AISTATS*, 2007.

[5] Chen Gong, Keren Fu, Qiang Wu, Enmei Tu, and Jie Yang. Semi-supervised classification with pairwise constraints. *Neurocomputing*, 139:130–137, 2014.

[6] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. In *International conference on machine learning*, pages 1321–1330. PMLR, 2017.

[7] Sebastian Houben, Johannes Stallkamp, Jan Salmen, Marc Schlipsing, and Christian Igel. Detection of traffic signs in real-world images: The German Traffic Sign Detection Benchmark. In *International Joint Conference on Neural Networks*, number 1288, 2013.

[8] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[9] Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondřej Chum. Mining on manifolds: Metric learning without labels. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7642–7651, 2018.

[10] Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondrej Chum. Label propagation for deep semi-supervised learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5070–5079, 2019.

[11] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7:535–547, 2021.

[12] Yannis Kalantidis, Mert Bulent Sariyildiz, Noe Pion, Philippe Weinzaepfel, and Diane Larlus. Hard negative mixing for contrastive learning. *Advances in Neural Information Processing Systems*, 33:21798–21809, 2020.

[13] Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Li Fei-Fei. Novel dataset for fine-grained image categorization. In *First Workshop on Fine-Grained Visual Categorization, IEEE Conference on Computer Vision and Pattern Recognition*, Colorado Springs, CO, June 2011.

[14] Sungyeon Kim, Dongwon Kim, Minsu Cho, and Suha Kwak. Proxy anchor loss for deep metric learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3238–3247, 2020.

[15] 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.

[16] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[17] Sawan Kumar, Shweta Garg, Kartik Mehta, and Nikhil Rasiwasia. Improving answer selection and answer triggering using hard negatives. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5911–5917, 2019.

[18] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[19] S. Maji, J. Kannala, E. Rahtu, M. Blaschko, and A. Vedaldi. Fine-grained visual classification of aircraft. Technical report, 2013.

[20] C.D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, 2008.

[21] Matthias Minderer, Josip Djolonga, Rob Romijnders, Frances Hubis, Xiaohua Zhai, Neil Houlsby, Dustin Tran, and Mario Lucic. Revisiting the calibration of modern neural networks. *Advances in Neural Information Processing Systems*, 34:15682–15694, 2021.

[22] Bojan Mohar, Y Alavi, G Chartrand, and OR Oellermann. The laplacian spectrum of graphs. *Graph theory, combinatorics, and applications*, 2(871-898):12, 1991.

[23] Yair Movshovitz-Attias, Alexander Toshev, Thomas K Leung, Sergey Ioffe, and Saurabh Singh. No fuss distance metric learning using proxies. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 360–368, 2017.

[24] Kevin Musgrave, Serge J. Belongie, and Ser-Nam Lim. A metric learning reality check. In *ECCV*, 2020.

[25] Maria-Elena Nilsback and Andrew Zisserman. A visual vocabulary for flower classification. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 1447–1454, 2006.

[26] Omkar M. Parkhi, Andrea Vedaldi, Andrew Zisserman, and C. V. Jawahar. Cats and dogs. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[27] Nikolaos Passalis, Alexandros Iosifidis, Moncef Gabbouj, and Anastasios Tefas. Variance-preserving deep metric learning for content-based image retrieval. *Pattern Recognition Letters*, 131:8–14, 2020.

[28] Yaxin Peng, Nijing Zhang, Ying Li, and Shihui Ying. A Local-to-Global metric learning framework from the geometric insight. *IEEE Access*, 8:16953–16964, 2020.

[29] Qi Qian, Lei Shang, Baigui Sun, Juhua Hu, Hao Li, and Rong Jin. Softtriple loss: Deep metric learning without triplet sampling. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6450–6458, 2019.

[30] Yazhou Ren, Kangrong Hu, Xinyi Dai, Lili Pan, Steven CH Hoi, and Zenglin Xu. Semi-supervised deep embedded clustering. *Neurocomputing*, 325:121–130, 2019.

[31] Joshua Robinson, Ching-Yao Chuang, Suvrit Sra, and Stefanie Jegelka. Contrastive learning with hard negative samples. *arXiv preprint arXiv:2010.04592*, 2020.

[32] Eu Wern Teh, Terrance DeVries, and Graham W Taylor. Proxynca++: Revisiting and revitalizing proxy neighborhood component analysis. *arXiv preprint arXiv:2004.01113*, 2020.

[33] Wei Tong and Rong Jin. Semi-supervised learning by mixed label propagation. In *AAAI*, pages 651–656, 2007.

[34] 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.

[35] Chao-Yuan Wu, R Manmatha, Alexander J Smola, and Philipp Krahenbuhl. Sampling matters in deep embedding learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2840–2848, 2017.

[36] Junyuan Xie, Ross Girshick, and Ali Farhadi. Unsupervised deep embedding for clustering analysis. In *International conference on machine learning*, pages 478–487, 2016.

[37] Eric Xing, Michael Jordan, Stuart J Russell, and Andrew Ng. Distance metric learning with application to clustering with side-information. *Advances in neural information processing systems*, 15, 2002.

[38] Yinyu Ye and Edison Tse. An extension of karmarkar's projective algorithm for convex quadratic programming. *Mathematical programming*, 44(1):157–179, 1989.

[39] Jingtao Zhan, Jiaxin Mao, Yiqun Liu, Jiafeng Guo, Min Zhang, and Shaoping Ma. Optimizing dense retrieval model training with hard negatives. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1503–1512, 2021.

[40] Wenzheng Zhang and Karl Stratos. Understanding hard negatives in noise contrastive estimation. *arXiv preprint arXiv:2104.06245*, 2021.

[41] Haixia Zheng and Horace Ip. Graph-based label propagation with dissimilarity regularization. volume 8294, pages 47–58, 12 2013.

[42] Dengyong Zhou, Olivier Bousquet, Thomas Lal, Jason Weston, and Bernhard Schölkopf. Learning with local and global consistency. *Advances in neural information processing systems*, 16, 2003.

[43] Furen Zhuang and Pierre Moulin. Deep semi-supervised metric learning via identification of manifold memberships. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2021, Toronto, ON, Canada, June 6-11, 2021*, pages 1755–1759. IEEE, 2021.

[44] Olga Zoidi, Anastasios Tefas, Nikos Nikolaidis, and Ioannis Pitas. Positive and negative label propagations. *IEEE Transactions on Circuits and Systems for Video Technology*, 28(2):342–355, 2016.