Deep Metric Learning with False Positive Probability: Trade off Hard Levels in a Weighted Way

Jia-Xing Zhong[1,2], Ge Li[1(✉)] and Nannan Li[1]

[1] School of Electronic and Computer Engineering, Peking University, Shenzhen, China
[2] School of Data and Computer Science, Sun Yat-sen University, Guangzhou, China
jxzhong@pku.edu.cn, geli@ece.pku.edu.cn, linn@pkusz.edu.cn

Abstract. In recent years, deep metric learning has been an end-to-end fashion in computer vision community due to the great success of deep learning. However, existing deep metric learning frameworks are faced with a dilemma about the hard level trade-off for training examples. Namely, the "harder" examples we feed to neural networks, the more likely we attain highly discriminative models, but the more easily neural networks get stuck into poor local minimal in practice. To fight against this dilemma, we propose a deep metric learning method with **FA**lse **P**ositive **P**robabilit**Y** (FAPPY) to gradually incorporate different hard levels. Unlike mainstream deep metric learning schemes, the presented approach optimizes similarity probability distribution among training samples, instead of the similarity itself. Experimental results on *CUB-200-2011*, *Stanford Online Products* and *VehicleID* datasets show that our FAPPY method achieves or outperforms state-of-the-art metric learning methods on fine-grained image retrieval and vehicle re-identification tasks. Besides, the presented method has relatively low sensitivity of hyper-parameters and it requires minor changes on traditional classification networks.

Keywords: Convolutional Neural Network, Deep Metric Learning, Hard Example Mining, Fine-grained Image Retrieval, Vehicle Re-identification.

1 Introduction

Deep metric learning aims to learn a metric space that pushes away dissimilar data points farther and pulls similar ones closer by deep neural networks. As an end-to-end feature embedding approach, deep metric learning is widely used on computer vision area, such as fine-grained image recognition [1,2], face identification/verification [3, 4], vehicle/person re-identification [5, 6], zero-shot learning [7, 8], *etc*.

However, deep metric learning suffers from a dilemma about mining hard training examples, where "hard" describes the distinguishable degree as shown in **Fig. 1**. On one hand, if we treat all the possible training samples without distinction, it is impossible or time-infeasible for the effective convergence of discriminative neural networks; on the other hand, if we force neural networks to pay excessive attention to hard examples, neural networks tend to get stuck in the local minimal in practice.

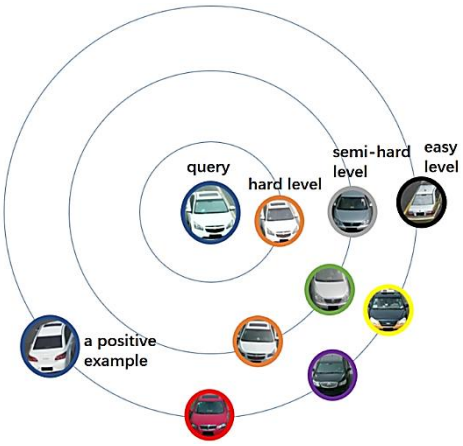

Fig. 1. Illustration of negative examples with different hard levels: In the metric space, shorter distance refers to higher similarity. The circles of the same color correspond to objects of the same label. Therefore, harder negative examples are closer to the query so that it is more difficult to tell the difference between them.

In this paper, we propose a metric learning approach to deal with the above dilemma in a weighted way. Our FAPPY method is an end-to-end framework as shown in **Fig. 2**, which aims to minimize the false positive probability. Our implementation just needs to replace the last loss layer with the FAPPY metric learning unit on traditional classification networks.

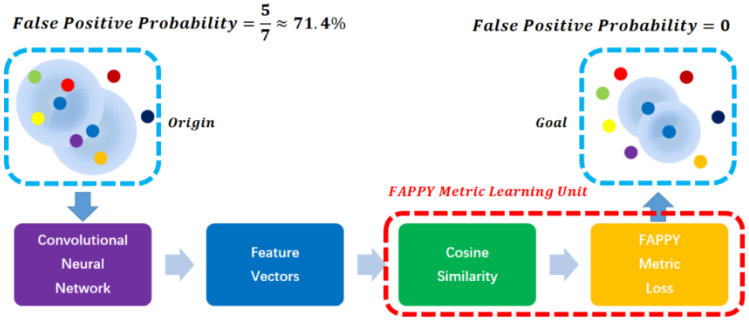

Fig. 2. Framework of the proposed approach: The neural network is designed to project the input data points into a cosine space at the lowest false positive rate. In the left-side original metric space, for a positive pair with the same label (two blue data points), there are 7 negative examples totally, 5 of which are closer to one positive example than the other one. Thus, the false positive probability is $\frac{5}{7} \approx 71.4\%$. Our goal is to minimize the false positive rate.

Specifically, our FAPPY method optimizes probability distribution of relative similarity among training samples, rather than directly adjusts similarity as other methods. Compared with other deep metric learning algorithms, the proposed method

is robust to hyper-parameter settings and do not require troublesome "hard example mining" [3] schemes.

Extensive experiments show that the performance of our approach is equal or better than state-of-the-art metric learning methods, without introducing any sensitive hyper-parameter. Our source code with *Caffe* and the trained models are available at: https://github.com/jx-zhong-for-academic-purpose/fappy_metric_learning.

2 Related Work

The first deep metric learning framework is siamese network [9], which strives to maximize the distance of positive examples and minimize the distance of negative ones. Sharing a similar goal, histogram loss [10] takes effort to separate similarity distributions of positive pairs from negative pairs. These two algorithms are based on absolute similarity/dissimilarity, however, the relative constraint seems of great importance on a majority of problems.

FaceNet [3] introduces triplet loss in the seminal work on deep metric learning with relative similarity, which optimizes relative distances from the anchor point. Later, Song *et al.* [11] proposes lifted structured embedding, which makes full use of the whole training batch. In recent works done by Yuan *et al.* [13] and Song *et al.* [12], relative similarity is adopted, as well as our FAPPY approach.

The above methods attempt to solve the problem on hard level trade-off in various ways. For instance, the lifted structured embedding [11] utilizes "log-sum-exp" function as a smooth upper bound, which may cause arithmetic overflow/underflow problems. FaceNet [3] applies the "semi-hard mining" strategy, so the batch size dramatically reaches up to 1800 for the selection of enough hard negative examples. To avoid these drawbacks, we fuse the false positive probability on various hard levels to settle this problem.

3 Deep Metric Learning with False Positive Probability

There are two stages in our FAPPY method, false positive probability estimation and weighted fusion. For clarity, we detail these two stages respectively in this section.

3.1 False Positive Probability Estimation

First of all, we estimate the false positive rate with different hard ranges respectively, where ***false positive*** means that a negative pair (two images with different labels) is determined as the positive one, as in **Fig. 2**.

Inspired by the histogram loss [10], we estimate the probability distribution with histograms. For each positive pair $\langle i, j \rangle$ belonging to a training batch set S, with a specific bin width Δ, we build a histogram to figure out its false positive rate.

we denote the cosine similarity between the training example $x, y \in \mathrm{S}$ as $s_{xy} = cos\langle \overrightarrow{f_x}, \overrightarrow{f_y} \rangle$, where $\overrightarrow{f_x}$ and $\overrightarrow{f_y}$ are respective feature vectors. Since the cosine similarity

ranges from -1 to +1, we define the bin boundaries of histograms as $b_1 = -1,\ b_2,\ \dots,\ b_{n-1},\ b_n = +1$ with n bins, where $n = \frac{2}{\Delta} + 1$ is determined by the bin width Δ.

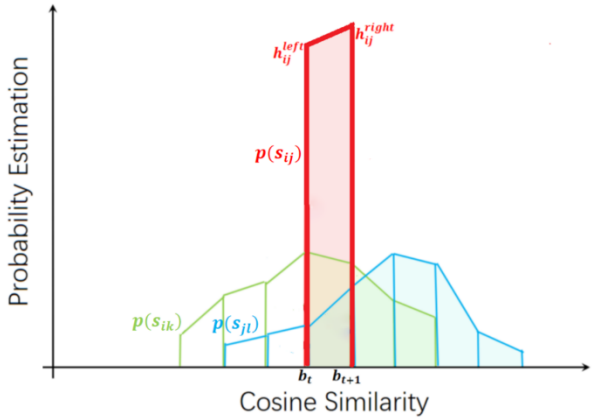

Fig. 3. For a specific positive pair and the bin width Δ, we build a histogram to estimate the false positive rate. The probability density function of s_{ij} is approximated by the red bins, and the green and blue portions indicate the similarity distribution of negative examples upon i and j in the same manner.

Assuming $s_{ij} \in [b_t, b_{t+1}]$, we denote the heights of bins b_t and b_{t+1} as:

$$h_{ij}^{left} = w_{ij}^{t}, \tag{1}$$

$$h_{ij}^{right} = w_{ij}^{t+1}, \tag{2}$$

where w_{ij}^{t} indicates the probability weight for training pair $\langle i, j \rangle$ on the t^{th} bin. The value of w_{ij}^{t} is determined by s_{ij} with triangular smoothing kernel [14]:

$$w_{ij}^{t} = \begin{cases} \frac{(s_{ij} - b_{t-1})}{\Delta}, s_{ij} \in [b_{t-1}, b_t] \\ \frac{(b_{t+1} - s_{ij})}{\Delta}, s_{ij} \in [b_t, b_{t+1}] \\ 0 \quad , otherwise. \end{cases} \tag{3}$$

In the same way, for this positive pair $\langle i, j \rangle$, the similarity density of negative examples on the u^{th} bin, are estimated respectively as follows, illustrated by the green and blue bins in **Fig. 3**:

$$h_i^u = \frac{1}{|N_{ij}|} \sum_k w_{ik}^u,\ k \in N_{ij}, \tag{4}$$

$$h_j^u = \frac{1}{|N_{ij}|} \sum_l w_{jl}^u,\ l \in N_{ij}, \tag{5}$$

where the negative set N_{ij} is composed of training examples whose labels are inconsistent with the example i (or j).

In **Fig. 2**, we have given an instance for the false positive probability, and its mathematical definition for the specific positive pair $\langle i,j\rangle$ is as follows:

$$P_{ij} = P(s_{ik} \geq s_{ij}) + P(s_{jl} \geq s_{ij}) = \int_{-1}^{1} p(s_{ij})[\int_{s_{ij}}^{1} p(s_{ik})\, ds_{ik}]ds_{ij} + \int_{-1}^{1} p(s_{ij})[\int_{s_{ij}}^{1} p(s_{jl})\, ds_{jl}]ds_{ij}\,,\ \forall k,l \in N_{ij}. \quad (6)$$

As shown in **Fig. 3**, the estimation of $P(s_{ik} \geq s_{ij})$ is obtained by the multiplication of the shade red area and the shade green area. Similarly, we approximate $P(s_{jl} \geq s_{ij})$ with the shade red area timed by blue ones. Thus, the false positive probability P_{ij} for this positive pair $\langle i,j\rangle$ can be estimated as:

$$P_{ij}^{\Delta} = (h_{ij}^{left} \sum_{u_1} h_i^{u_1} + h_{ij}^{right} \sum_{u_2} h_i^{u_2}) + (h_{ij}^{left} \sum_{u_3} h_j^{u_3} + h_{ij}^{right} \sum_{u_4} h_j^{u_4})\,, \quad (7)$$

where Δ is the bin width, and u_1, u_1, u_1, u_4 satisfy that $b_{u_1}, b_{u_3} \geq b_t$ while $b_{u_2}, b_{u_4} \geq b_{t+1}$ since $s_{ij} \in [b_t, b_{t+1}]$.

The corresponding relationship between **Eq. (6)** and **Eq. (7)** is: $P_{ij} \approx P_{ij}^{\Delta}, P(s_{ik} \geq s_{ij}) \approx (h_{ij}^{left} \sum_{u_1} h_i^{u_1} + h_{ij}^{right} \sum_{u_2} h_i^{u_2})$ and $P(s_{jl} \geq s_{ij}) \approx (h_{ij}^{left} \sum_{u_3} h_j^{u_3} + h_{ij}^{right} \sum_{u_4} h_j^{u_4})$.

3.2 Weighted Fusion

Lemma 1. Given the bin width Δ and a positive pair $\langle i,j\rangle$, assuming $s_{ij} \in [b_t, b_{t+1}]$, $\forall k,l \in N_{ij}$, the backward propagation gradients of false positive probability estimation P_{ij}^{Δ} satisfy: $\frac{\partial P_{ij}^{\Delta}}{\partial s_{ik}} \neq 0 \Leftrightarrow s_{ik} \in [b_t, b_{t+1}]$, $\frac{\partial P_{ij}^{\Delta}}{\partial s_{jl}} \neq 0 \Leftrightarrow s_{jl} \in [b_t, b_{t+1}]$.

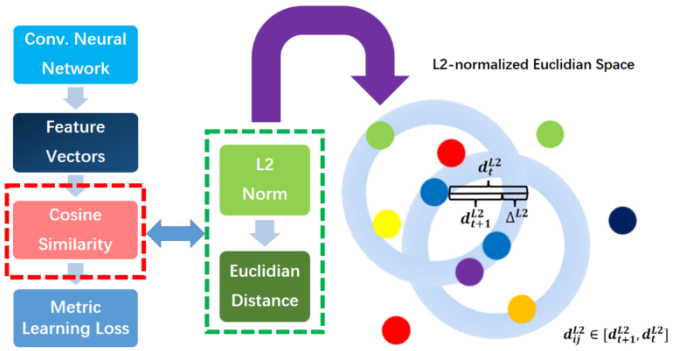

Fig. 4. In the equivalent L2-normalized Euclidian space, given the Δ^{L2} and a positive pair $\langle i,j\rangle$ with the distance $d_{ij}^{L2} \in [d_{t+1}^{L2}, d_t^{L2}]$, where $d_t^{L2} - d_{t+1}^{L2} = \Delta^{L2}$; d_{ij}^{L2}, d_t^{L2}, d_{t+1}^{L2} and Δ^{L2} correspond to s_{ij}, b_{t+1}, b_t and Δ in the cosine space. During the backward propagation to minimize P_{ij}^{Δ}, ***only the blue-shaded area is taken into consideration***. The radius of blue shade circular ring locates in the range of $[d_{t+1}^{L2}, d_t^{L2}]$ from the data points i and j. The proof is given in **Appendix.**

In other words, the bin width Δ strongly constrains the range of hard levels. To make it more understandable, we illustrate **Lemma 1** in **Fig. 4** with the L2-normalized Euclidian space, an equivalent form of the cosine metric space. The proof of **Lemma 1** and this equivalency is given in **Appendix.**

The larger Δ value we set, the bigger metric space around positive examples we focus on, so that the more hard levels we choose. Therefore, we can merge different hard levels together and compute the loss function for $\langle i,j\rangle$: $L_{ij} = \alpha_1 P_{ij}^{\Delta_1} + \alpha_2 P_{ij}^{\Delta_2} + \cdots + \alpha_n P_{ij}^{\Delta_n}$ where α is the fusion weight and $\Delta_1, \Delta_2, \ldots, \Delta_n$ are different bin widths.

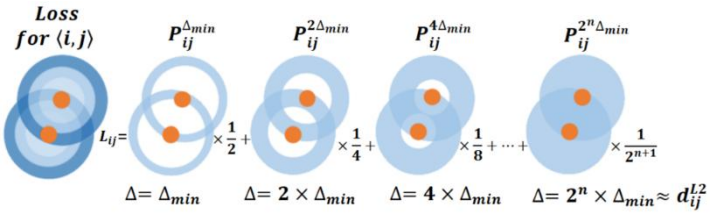

Fig. 5. Fusion Strategy: The false positive rate estimation takes more hard levels into account with gradually reducing weights. As with the final loss for $\langle i,j\rangle$, the blue ring farthest away from a positive data point has the darkest color, which means that we actually give more weights to negative examples with the easier "hard level".

As is shown in **Fig. 5**, we gradually increase the delta value by 2x, while decrease its weight by 0.5x, and compute L_{ij} with the summation over these results. By this means, we attach greater importance to the easier "hard level". In the meanwhile, we also take harder levels into account. As for our FAPPY method, the only hyper-parameter we need to set is the minimum bin width Δ_{min}. It is worth mentioning that experimental results in **Section 4** show that Δ_{min} is remarkably robust.

Algorithm 1 Fusion to Compute the Loss Function

Input: A training batch set S and the minimum bin width Δ_{min}
Output: The loss function L

$L \leftarrow 0$, $count_{positive\_pairs} \leftarrow \left|\{\langle i,j\rangle \in S \middle| label_i = label_j\}\right|$, $\Delta_{cur} \leftarrow 2.0$
repeat
  $L_{\Delta_{cur}} \leftarrow 0$
  for all positive pair $\langle i,j\rangle$ such that $i,j \in S$ **do**
    if $1.0 - s_{ij} \geq \Delta_{cur}$ **then**
      compute $P_{ij}^{\Delta_{cur}}$ as **Eq. (7)**
      $L_{\Delta_{cur}} \leftarrow L_{\Delta_{cur}} + P_{ij}^{\Delta_{cur}}$
    end if
  end for
  $L \leftarrow L + L_{\Delta_{cur}}/count_{positive\_pairs}$
  $L \leftarrow L/2.0$
  $\Delta_{cur} \leftarrow \Delta_{cur}/2.0$
until $\Delta_{cur} < \Delta_{min}$

The average loss on each positive pair is the final loss function. The pseudocode of loss function computation is shown in **Algorithm 1**. To minimize the false posi-

tive error rate, the neural network is updated with $\frac{\partial L}{\partial \vec{f_i}}$ during the backward propagation process, where $\vec{f_i}$ is the feature vector of the training example i.

4 Experiments

We evaluate our FAPPY method on 3 datasets, *i.e. CUB-200-2011* [16], *Stanford Online Products* [11] and *VehicleID* [5]. The experiments consist of two typical tasks, fined-grained image retrieval and similar vehicle re-identification, implemented with the *Caffe* [15] framework. Furthermore, we conduct a series of experimental studies about the sensitivity of hyper-parameters for both tasks, at various minimum bin widths $\Delta_{min} = \{0.01,\ 0.001,\ 0.0001\}$.

4.1 Datasets

The retrieval experiments are conducted on the *CUB-200-2011* and *Stanford Online Products* datasets, while the re-identification experiments are performed upon the benchmark *VehicleID* dataset. To ensure the apples-to-apples comparison to existing methods, we follow the standard train/test data split.

- *CUB-200-2011* dataset consists of 200 bird species with 11,788 images, where the first 100 species with 5,864 images make up the training set and the rest 100 species with 5,924 images constitute the test set.
- *Stanford Online Products* dataset consists of 22,634 classes with 120,053 online products images from eBay, where 59,551 images of 11,318 classes are for training and 60,502 images of 11,316 classes are for testing.
- *VehicleID* dataset consists of 26,267 vehicles with 221,763 images, where the training set includes 110,178 images of 13,134 vehicles and the test set comprises 111,585 images of 13,133 vehicles. In accordance with [5], we choose 3 test data splits with different scales. The small test set is composed of 800 vehicles with 7,332 images. The medium test set is composed of 1,600 vehicles with 12,995 images. The large test set is composed of 2,400 vehicles with 20,038 images.

4.2 Fine-grained Image Retrieval

Fine-grained image recognition is such a challenging problem that researchers can just improve its performance by around 1% every year despite of their great efforts. Typically, fined-grained image retrieval is a recognition problem on unseen categories, which means the object classes are entirely disjoint between train and test sets.

On fined-grained image retrieval tasks, we compare our FAPPY method with state-of-the-art deep metric learning approaches on two frequently-used datasets, *CUB-*

200-2011 and *Stanford Online Products*. We apply the standard Recall@K metric[1] [17] on the experiment result.

Table 1. Recall@K (%) of Fine-grained Image Retrieval

Dataset	CUB-200-2011						Stanford Online Products			
K	1	2	4	8	16	32	1	10	100	1000
Contrastive [9]	26.4	37.7	49.8	62.3	76.4	85.3	42.0	58.2	73.8	89.1
Triplet [3]	36.1	48.6	59.3	70.0	80.2	88.4	42.1	63.5	82.5	94.8
LiftStruct [11]	47.2	58.9	70.2	80.2	89.3	93.2	62.1	79.8	91.3	97.4
Histogram [10]	50.3	61.9	72.6	82.4	88.8	93.7	**63.9**	**81.7**	**92.2**	**97.7**
Ours$^{0.01}$	50.4	**62.6**	74.8	**84.0**	90.5	94.9	63.3	**80.8**	**91.7**	**97.4**
Ours$^{0.001}$	50.5	**62.6**	**74.9**	**84.0**	**90.7**	**95.0**	**63.5**	80.7	91.6	97.2
Ours$^{0.0001}$	**50.6**	62.4	**74.9**	83.8	90.3	**95.0**	63.4	**80.8**	**91.7**	**97.3**

Following [11] and [10], our basic network architecture is the *GoogLeNet* (up to "pool5") [18] pre-trained on *ImageNet ILSVRC* [19], appended by a fully connected product layer initialized with random weights. In addition, we keep the implementation details closely with above two papers.

As is shown in **Table 1**, the performance of FAPPY approach is marginally better than that of state-of-the-art on *CUB-200-2011* and just slightly inferior to the histogram loss on *Stanford Online Products*. Besides, the results of FAPPY by Recall@K metric remain stable (change less than 0.5%) even though the minimum bin width Δ_{min} varies across several orders of magnitude.

4.3 Vehicle Re-identification

For deep metric learning, vehicle re-identification is the touchstone of discernment, because numerous vehicles with the same vehicle model are almost exactly the same in appearance.

We make experience on the *VehicleID* dataset with different scales of gallery sets (*i.e.* small, medium and large). Following the paper [5], we adopt the pre-trained network *VGG_CNN_M_1024* [20] with the batch size of 150, and repeat the query 10 times in test phase.

The evaluation metric is the Top-k match rate and we draw the cumulative match characteristic (CMC) curve [21]. Our FAPPY method significantly outperforms the existing approaches and the result is pretty stable as shown in **Table 2** and **Fig. 6**.

[1] Recall@K is the average recall scores over all the query images in testing set. For each query image, the recall score is 1 if at least one positive image in the nearest K returned images and 0 otherwise

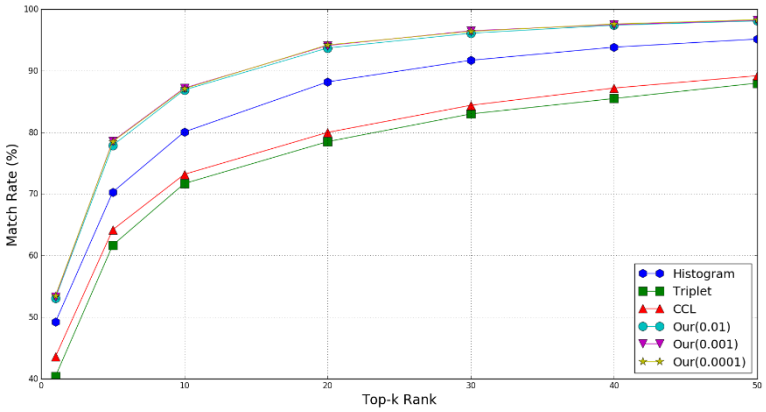

Fig. 6. CMC on the Small Gallery Set

Table 2. Match Rate (%) of Vehicle Re-identification Task

Match Rate	Small		Medium		Large	
Top K	Top 1	Top 5	Top 1	Top 5	Top 1	Top 5
Triplet [3]	40.4	61.7	35.4	54.6	31.9	50.3
CCL [5]	43.6	64.2	37.0	57.1	32.9	53.3
LiftStruct[2][11]	/	/	/	/	/	/
Histogram [10]	49.3	70.9	43.7	65.1	39.5	59.9
Ours$^{0.01}$	53.1	77.9	46.9	71.9	42.4	66.2
Ours$^{0.001}$	53.3	**78.6**	**47.5**	**72.6**	**43.0**	**66.7**
Ours$^{0.0001}$	**53.5**	78.5	47.3	72.2	42.8	66.6

5 Conclusion

In this paper, we propose a deep metric learning framework with false positive probability (FAPPY), which can gradually incorporate training example with different hard levels. Neither complex "hard example mining" nor intricate hyper-parameters tuning is required in the presented method. The experimental results show that our FAPPY method is comparable to existing metric learning approaches on fine-grained image retrieval, and outperforms state-of-the-art on vehicle re-identification tasks.

Acknowledgements. This work is partially supported by National Science Foundation of China (No. U1611461), Shenzhen Peacock Plan (20130408-183003656), Science, Technology Planning Project of Guangdong Province (No. 2014B090910001) and National Natural Science Foundation of China (61602014). In addition, we would

[2] The LiftStruct metric learning gets no results for it failed to converge in the experiment.

like to thank Guangzhou Supercomputer Center for providing us with Tianhe-2 system to conduct the experiment and giving us technical supports.

References

1. Cui, Y., Zhou, F., Lin, Y., & Belongie, S.: Fine-grained categorization and dataset bootstrapping using deep metric learning with humans in the loop. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition, pp. 1153-1162 (2016).
2. Zhang, X., Zhou, F., Lin, Y., & Zhang, S.: Embedding label structures for fine-grained feature representation. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition, pp. 1114-1123 (2016).
3. Schroff, F., Kalenichenko, D., & Philbin, J.: Facenet: A unified embedding for face recognition and clustering. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition, pp. 815-823 (2015).
4. Wen, Y., Zhang, K., & Qiao, Y.: A discriminative feature learning approach for deep face recognition. In: 2016 European Conference on Computer Vision, pp. 499-515 (2016).
5. Liu, H., Tian, Y., Yang, Y., Pang, L., & Huang, T.: Deep relative distance learning: Tell the difference between similar vehicles. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition, pp. 2167-2175 (2016).
6. You, J., Wu, A., & Zheng, W. S.: Top-push video-based person re-identification. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition, pp. 1345-1353 (2016).
7. Zhang, Z., & Saligrama, V.: Zero-shot learning via joint latent similarity embedding. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition, pp. 634-642 (2016).
8. Bucher, M., Herbin, S., & Jurie, F.: Improving semantic embedding consistency by metric learning for zero-shot classiffication. In: 2016 European Conference on Computer Vision, pp. 730-746 (2016).
9. Chopra, S., Hadsell, R., & LeCun, Y.: Learning a similarity metric discriminatively, with application to face verification. In: 2016 European Conference on Computer Vision, pp. 539-546 (2016).
10. Ustinova, E., & Lempitsky, V.: Learning deep embeddings with histogram loss. In: Advances in Neural Information Processing Systems, pp. 4170-4178 (2016).
11. Oh Song, H., Xiang, Y., Jegelka, S., & Savarese, S.: Deep metric learning via lifted structured feature embedding. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition, pp. 4004-4012 (2016).
12. Song, H. O., Jegelka, S., Rathod, V., & Murphy, K.: Deep Metric Learning via Facility Location (2017).
13. Yuan, Y., Yang, K., & Zhang, C.: Hard-Aware Deeply Cascaded Embedding. arXiv preprint:1611.05720 (2016).
14. Zucchini, Walter, A. Berzel, and O. Nenadic: Applied smoothing techniques. Part I: Kernel Density Estimation, pp. 5 (2003).
15. Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., ... & Darrell, T.: Caffe: Convolutional architecture for fast feature embedding. In: 22nd ACM international conference on Multimedia, pp. 675-678 (2014).
16. Welinder, P., Branson, S., Mita, T., Wah, C., Schroff, F., Belongie, S., & Perona, P.: Caltech-UCSD birds 200 (2011).
17. Jegou, H., Douze, M., & Schmid, C.: Product quantization for nearest neighbor search. IEEE transactions on pattern analysis and machine intelligence, 33(1), 117-128 (2011).

18. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... & Rabinovich, A.: Going deeper with convolutions. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition, pp.1-9 (2015).
19. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., ... & Berg, A. C.: Imagenet large scale visual recognition challenge. International Journal of Computer Vision, 115(3), 211-252 (2015).
20. Chatfield, K., Simonyan, K., Vedaldi, A., & Zisserman, A.: Return of the devil in the details: Delving deep into convolutional nets. arXiv preprint arXiv:1405.3531 (2014).
21. Gray, D., Brennan, S., & Tao, H.: Evaluating appearance models for recognition, reacquisition, and tracking. In: IEEE International Workshop on Performance Evaluation for Tracking and Surveillance (PETS) (Vol. 3, No. 5) (2007).

Appendix.

Lemma 1. *Proof.* We compute the gradients of these pairs:

$$\frac{\partial P_{ij}^{\Delta}}{\partial s_{ij}} = h_i^t + h_j^t, \quad \frac{\partial P_{ij}^{\Delta}}{\partial s_{ik}} = \begin{cases} -h_{ij}^{right}, s_{ik} \in [b_t, b_{t+1}] \\ 0, \quad otherwise \end{cases}, \quad \frac{\partial P_{ij}^{\Delta}}{\partial s_{jl}} = \begin{cases} -h_{ij}^{right}, s_{jl} \in [b_t, b_{t+1}] \\ 0, \quad otherwise \end{cases}$$

Strictly speaking, the necessary and sufficient condition of $\frac{\partial P_{ij}^{\Delta}}{\partial s_{ik}} \neq 0$ is $s_{ik},\ s_{ij} \in (b_t, b_{t+1}]$. Likewise, $\frac{\partial P_{ij}^{\Delta}}{\partial s_{jl}} \neq 0$ if and only if $s_{jl},\ s_{ij} \in (b_t, b_{t+1}]$.

The Equivalency *Proof.*

In the L2-normalized space, the feature vectors are defined as: $\overrightarrow{f_x^{L2}} = \frac{\overrightarrow{f_x}}{\|\overrightarrow{f_x}\|}$.

The Euclidian distance d_{xy}^{L2} and Δ^{L2} are as follows:

$$d_{xy}^{L2} = \left\| \overrightarrow{f_x^{L2}} - \overrightarrow{f_y^{L2}} \right\| = \left\| \frac{\overrightarrow{f_x}}{\|\overrightarrow{f_x}\|} - \frac{\overrightarrow{f_y}}{\|\overrightarrow{f_y}\|} \right\| = \sqrt{1 - 2cos\langle \overrightarrow{f_x}, \overrightarrow{f_y} \rangle + 1} = \sqrt{2 - 2s_{xy}}$$

$$\Delta^{L2} = d_t^{L2} - d_{t+1}^{L2} = \sqrt{2 - 2b_t} - \sqrt{2 - 2b_{t+1}}$$

Therefore, Δ^{L2} is positively related to Δ while d_{xy}^{L2} is negatively related to s_{xy}, and the explanation in **Fig. 4** is reasonable.