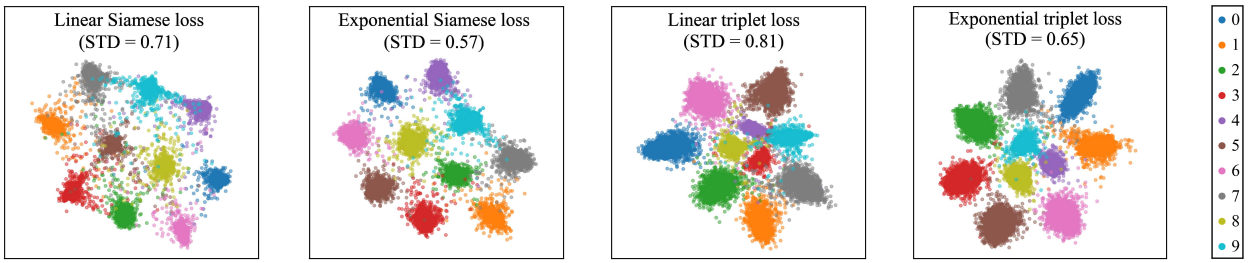
Better and Faster: Exponential Loss for Image Patch Matching

Shuang Wang[1] Yanfeng Li[1] Xuefeng Liang[1,2] Dou Quan[1] Bowu Yang[1]
Shaowei Wei[1] Licheng Jiao[1]
[1]School of Artificial Intelligence, Xidian University,Shaanxi,China [2]Kyoto University, Kyoto, Japan
xliang@xidian.edu.cn



(a) (b) (c) (d)

Figure 1: Comparison of clustering results using varied loss functions on the MNIST dataset. (a) and (b) use Siamese losses, (c) and (d) use triplet losses. (a) and (c) use linear losses, and the average standard deviations (STD) of intra-class distances are 0.71 and 0.81, respectively. (b) and (d) use our proposed exponential losses, and the average STDs of intra-class distances are 0.57 and 0.65. Clearly, the networks with exponential losses result in a smaller intra-class distance which alleviates the difficulty of classification, especially with hard samples.

Abstract

Recent studies on image patch matching are paying more attention on hard sample learning, because easy samples do not contribute much to the network optimization. They have proposed various hard negative sample mining strategies, but very few addressed this problem from the perspective of loss functions. Our research shows that the conventional Siamese and triplet losses treat all samples linearly, thus make the training time consuming. Instead, we propose the exponential Siamese and triplet losses, which can naturally focus more on hard samples and put less emphasis on easy ones, meanwhile, speed up the optimization. To assist the exponential losses, we introduce the hard positive sample mining to further enhance the effectiveness. The extensive experiments demonstrate our proposal improves both metric and descriptor learning on several well accepted benchmarks, and outperforms the state-of-the-arts on the UBC dataset. Moreover, it also shows a better generalizability on cross-spectral image matching and image retrieval tasks.

1. Introduction

Patch-based matching technology has been widely applied in structure from motion [41], stereo matching [10] and image retrieval [31]. Early methods, such as SIFT [23] and SURF [6], mainly focus on the design of hand-crafted feature descriptors. However, they lack an ability of capturing the higher-level structural information, and only work well in some specific scenarios. Recently, the deep-learning based methods [2, 3, 13, 19, 24] have been showing a promising performance on this task, especially on effectiveness and generalizability [13, 21, 36]. According to the objective functions, these methods can be divided into two categories: *metric learning* and *descriptor learning*. The former gives the pairwise matching probabilities directly [13, 21, 39], while the latter outputs the features of image patches, whose similarities are measured in the feature space [19, 27, 36, 40].

Both methods mostly use Siamese or triplet losses to optimize the network. However, they cannot deal with the problem of data redundancy because the training data are patch pairs/triplets. When the number of patches grows, the possible non-matching patch pairs/triplets are exponentially increasing. In addition, the majority of training data are the easy negative samples. Usually, they contribute little to network optimization after a few training epochs, then make the training time consuming. To address this problem, Zhang *et al*. [40] proposed a regularization term to fully utilize the descriptor space. Tian *et al*. [36] introduced a progressive sampling strategy to generate samples more effectively. Later, a *hardest-in-batch* strategy proposed in

[19, 27] was used to mine hard negative samples. The emerging studies [5, 35] have confirmed the improvement of the accuracy and efficiency while the hard sample mining was applied.

However, the above methods still have two weaknesses: (1) the loss functions are linear with respect to the feature distance (Euclidean distance), thus the network treats all training samples linearly. This makes the network penalizes less the hard examples in one iteration, and results in a time-consuming training. (2) There is no consideration of hard positive samples, which causes the positive training data to become redundant as well. To tackle these problems, we propose the exponential Siamese and triplet losses, which can naturally put more attention on hard samples than easy ones during training. Compared with the linear losses, the exponential losses achieve better clustering results as shown in Fig. 1, larger inter-class distances and smaller intra-class distances. In other words, clusters become more compact, particularly for hard samples (points far from the center of the cluster). To assist the exponential losses, we apply both positive and negative mining to select hard samples for training. Moreover, we design a shared feature network based on the work [39] as the basic network for both metric and descriptor learning, which demonstrates a better generalizability when our exponential losses are applied.

Our contribution is threefold: (1) we propose the exponential Siamese and triplet loss functions for descriptor learning and metric learning. Compared with linear functions, they enable networks to focus more on hard samples; (2) we introduce the hard positive sample mining, that makes the learning converge faster; (3) we design a shared feature network for both metric and descriptor learning. Extensive experiments on benchmarks demonstrate that our proposal achieves the state-of-the-art performance on patch matching problem, and outperforms other methods on image retrieval task.

2. Related Work

For image patch matching problem, deep learning-based methods can be grouped into two major categories: *metric learning* and *descriptor learning*. We briefly review these methods in below.

2.1. Metric Learning

MatchNet [13] designed a two-tower Siamese network for feature extraction following a three fully-connected layers that measures the similarity of feature pairs. Considering the tradeoff between efficiency and accuracy, Zagoruyko *et al*. [39] explored several different network structures, including a 2-channel network (2-ch) and a central-surround (CS) architecture. The 2-ch regards two patches as a 2-channel image, while CS learns multi-resolution information during training. Both reported considerable performance improvements. Kumar *et al*. [21] proposed a global loss to alleviate the over-fitting problem, which minimizes and maximizes the means of the similarities between non-matching pairs and matching pairs, respectively. In addition, it can minimize the variance of two distributions.

To the best of our knowledge, there was no hard sample mining strategy applied in metric learning. Instead, we introduce a hard negative mining to speed up training.

2.2. Descriptor learning

Unlike metric learning, descriptor learning learns the feature descriptors and directly measures the pairwise distance in feature space. Recently, many works applied Siamese network [21, 25] and triplet network [3, 16, 21] structures to learn descriptors. Specifically, Simo *et al*. [35] utilized a Siamese network with contrastive loss to learn discriminative descriptors. Balntas *et al*. [3] designed a new powerful loss function termed softPN which combined a softmax ratio loss [16] with soft negative mining strategy. Zhang *et al*. [40] proposed a regularization term to improve the expressive power of the feature space, which could be cooperated with any loss function such as Siamese and triplet losses. L2Net [36] proposed a progressive sampling strategy to enable the network to access a large number of samples within a few epochs, and reported a remarkable performance on UBC benchmark [8]. Later, Scale aware [19] and HardNet [27] utilized a more efficient batch-based sampling strategy to mine the hard negative samples. What's more, Scale aware combined the Siamese and triplet losses to learn consistently scaled descriptors. But, too many hyper-parameters lead to a complex training process.

To improve the performance, one effective solution in most descriptor learning methods is introducing or involving a hard negative sample mining strategy. However, very few works address this issue from perspective of loss function or think about the hard positive samples. We propose new solutions that are detailed in below.

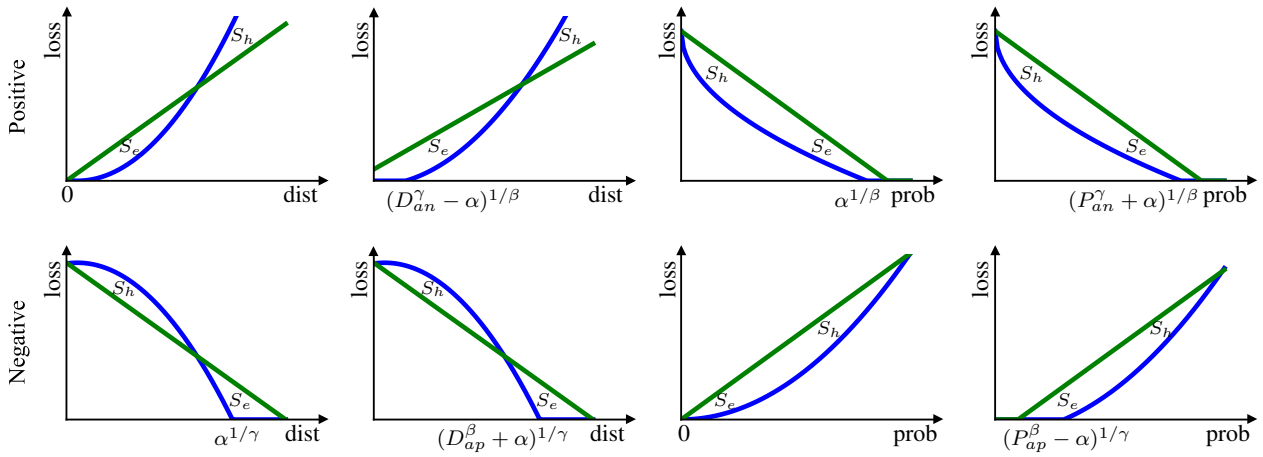
3. Methodology

We firstly give the idea and analysis of the proposed exponential loss functions, and then introduce the methodology of hard positive and negative sample mining. Last, the details of the shared feature network are given.

3.1. Exponential loss functions

Since the exponential loss functions are based on Siamese and triplet losses, we firstly review them briefly.

Siamese losses are normally designed for image patch pairs. The most popular variant of Siamese loss is Contrastive loss [11, 12, 26]. For simplicity, we first formulate a general



(a) Siamese loss (dist) (b) triplet loss (dist) (c) Siamese loss (prob) (d) triplet loss (prob)

Figure 2: The plot of exponential (blue curves) and linear (green lines) losses against the pairwise feature distance or pairwise matching probability. The top and bottom rows denote positive and negative samples, respectively. (a) and (b) are for descriptor learning, and the horizontal axis is the pairwise distance. (c) and (d) are for metric learning, and the horizontal axis is the matching probability. The vertical axis is the loss. S_e and S_h represent the ranges of easy and hard samples, respectively.

Siamese loss [19, 35] as:

$$\mathcal{L}_{Siamese} = D_{ap} + [\alpha - D_{mn}]_+, \tag{1}$$

where (a, p) denotes a matching patch pair (positive sample), (m, n) denotes a non-matching patch pair (negative sample). $D_{ap} = \|f(a) - f(p)\|_2$ and $D_{mn} = \|f(m) - f(n)\|_2$ are Euclidean distances between feature descriptors of two patches. f represents a differentiable deep network that maps image patches to normalized feature descriptors. $[z]_+ = max(z, 0)$. Siamese loss is designed to keep positive pairs as close as possible, while separate negative pairs far than a margin distance α. One can see that it only considers the absolute distance of patch pairs. Figure 2 (a) (green lines) depicts the relationship between pairwise distance and loss. The top and bottom rows represent the positive and negative samples, respectively. For a positive sample, the contributed loss for network training is linearly increasing with the increase of the feature distance between two patches. The pairs with larger distances are hard positive samples, those with smaller distances are easy samples. Analogously, the loss of a negative sample is linearly decreasing with the increase of the distance between two patches. Please note that the pairs with smaller distances are hard negative samples.

Triplet losses are designed for patch triplets. Given a patch triplet $\{a, p, n\}$, the patch pair (a, p) denotes a positive pair, and (a, n) is a negative pair. Specifically, a triplet margin loss [5] is defined as:

$$\mathcal{L}_{triplet} = [D_{ap} - D_{an} + \alpha]_+. \tag{2}$$

Triplet losses require that the negative distance must be larger than the positive distance a margin α. Therefore, they consider the relative relationship between positive and negative pairs, and usually offer a faster convergence than Siamese losses. Similar to Siamese losses, Fig. 2 (b) (green lines) shows that the conventional triplet losses are also a linear function.

Exponential losses. Recently, most works [15, 19, 29, 34, 35, 37, 38] use l_2 Euclidean distance as the function D. Several studies [11, 18, 26, 33] instead apply l_2^2 squared Euclidean distance; however, the research [15, 37] concludes that l_2^2 loss makes the model more prone to collapsing and using the l_2 loss is more stable. Our investigation shows that a few extremely hard samples could lead to the training stagnation which only occurs in early stages. For the other samples, their gradients are still in a reasonable range, and make the training procedure gradually stable. Therefore, we only apply l_2 loss during the first training epoch, and then replace it by l_2^2 loss in subsequent training. Results of the test show that l_2^2 loss offers a faster convergence and a better performance. Compared with l_2 loss, l_2^2 loss is non-linear with respect to the pairwise feature distance. We believe that this is the essence of the performance gain, and then discuss the general form of Siamese and triplet loss in descriptor learning:

$$\begin{aligned} \mathcal{L}_{Siamese(dist)} &= D_{ap}^{\beta} + [\alpha - D_{mn}^{\gamma}]_+, \\ \mathcal{L}_{triplet(dist)} &= [D_{ap}^{\beta} - D_{an}^{\gamma} + \alpha]_+, \end{aligned} \tag{3}$$

where $\beta > 0$ and $\gamma > 0$ are exponential orders that control the change rate of the loss functions. When $\beta = \gamma = 1$,

they are the conventional Siamese loss (Eq.1) and triplet loss (Eq.2). When $\beta \neq 1$ and $\gamma \neq 1$, we then name them as exponential losses. The blue curves in Fig. 2 (a) and (b) show the relations between loss and pairwise feature distance. For the positive samples with a small feature distance (easy positive samples), we expect to maintain a smaller loss which has less attention during training. The top figure in Fig. 2 (a) illustrates that the exponential Siamese loss holds a smaller value when the distance is small (left side). When the distance increases, the positive samples become harder to distinguish. The network is expected to pay more attention on hard samples. Thanks to the property of exponential function, the exponential Siamese loss grows rapidly with the distance increasing (right side), where S_e and S_h represent the ranges of easy and hard samples, respectively. For the negative samples with a larger feature distance (easy negative samples), the exponential Siamese loss has a smaller loss, as shown right side in the bottom Fig. 2 (a). Please note that loss becomes 0 when the feature distance of a easy negative sample is bigger than a certain margin. As the distance decreases, the negative samples become harder to be distinguished. The exponential Siamese loss increases quickly and keeps larger loss values for those hard samples, which enforces the network to learning more from these data.

The exponential triplet loss functions similarly as the exponential Siamese loss, as shown in Fig. 2 (b). The only difference is that no penalty is given to easy positive samples until the distance is greater than a margin. The experiment demonstrates the best parameter setting is $\beta = 2$ and $\gamma = 2$. Instead, conventional losses vary linearly with the pairwise distance, thus cannot effectively let network put less focus on easy samples and more on hard ones, which leads to a slow convergence.

This idea can be easily transferred to metric learning whose outputs are the positive-pair matching probability P_{ap} and negative-pair matching probabilities P_{mn}, P_{an}. Therefore, the general form of Siamese and triplet loss can be rewritten as:

$$\begin{aligned} \mathcal{L}_{Siamese(prob)} &= P_{mn}^{\beta} + [\alpha - P_{ap}^{\gamma}]_{+}, \\ \mathcal{L}_{triplet(prob)} &= [P_{an}^{\beta} - P_{ap}^{\gamma} + \alpha]_{+}. \end{aligned} \tag{4}$$

Figure 2 (c) and (d) depict the relations between pairwise matching probability and loss. In right side of top figures, the easy positive samples have higher matching probabilities, then, the exponential losses give less penalty for them. As the probability decreases, the losses increase slowly, which effectively weaken the impact of easy positive samples. When the probability decreases further (the left side of figures), positive samples become hard ones and the exponential losses quickly increase. Analogously, the easy negative samples keep minor penalties due to small matching probabilities, as shown in the left side of bottom figures. Note that triplet loss remains 0 until the matching probability is greater than a margin. With the probability increasing, the loss exponentially grows which enforces network to focus more on hard samples. From Fig. 2 (c) and (d), one can see the exponential loss values are less than the linear loss values. Actually, it does not affect the results because a hard sample contributes much more loss than an easy one when exponential loss is applied. Although the hard samples are the minority in training data, their accumulative loss becomes much stronger compared with the linear loss. The experiments shows the best parameter setting is $\beta = 2$ and $\gamma = 0.3$ for metric learning.

For simplicity, we name the linear Siamese and triplet losses as **Linear-SLoss** and **Linear-TLoss**, the exponential Siamese and triplet losses as **Exp-SLoss** and **Exp-TLoss**.

Toy Problem. To demonstrate the effectiveness of exponential losses, we train a simple network on the MNIST [22] dataset for a clustering task. Like the BN-net in [19], Batch Normalization (BN) [17] and ReLU [28] are added behind each convolutional layers, except for the last one. The input is a 28×28 grayscale patch. The output feature descriptor is processed by dimension reduction and visualized on a 2D plane. We train the network using exponential loss and linear loss with Adam optimizer [20]. The initial learning rate is 0.001. The margins of exponential and linear loss functions are set to 2. Figure 1 shows the clustering results.

We can see that exponential losses result in smaller intra-class distances and larger inter-class distances compared with linear losses, which benefit from the stronger penalty on hard samples (points far from the center of the class). Moreover, we can observe that the clusters using Siamese losses are more compact than those using triplet losses. It is consistent with the definition of loss functions. Siamese losses encourage positive samples to be as close as possible, while triplet losses just try to force positive samples closer to all negative ones.

3.2. Hard positive and negative sample mining

The use of exponential loss only may not guarantee the best performance. Since the proportion of easy samples usually is rather big in training data, network has less opportunity to learn from hard samples. This is the reason emerging studies introduced many data sampling and mining strategies [3, 5, 19, 24, 27, 37]. Unlike these methods mainly focusing on the hard negative samples, we notice that there also exist hard positive samples that can help the network training. Therefore, we optimize the network using our exponential losses, and do both hard positive and negative samples mining for descriptor learning. Inspired by the *hardest in batch* strategy for negative samples in HardNet [27], we select n positive pairs for each minibatch, and then generate $n(n-1)$ negative pairs by cross-pairing. These pairs are fed into a Siamese network. The network outputs

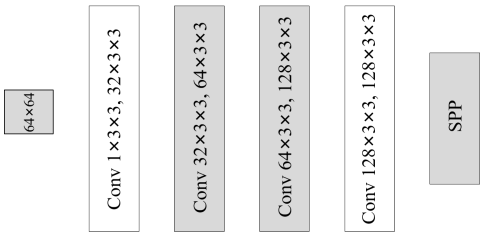


Figure 3: The architecture of our shared feature network.

pairwise feature descriptors. Then, a pairwise distance matrix is calculated to select the hard negative pairs for each positive pairs (more details can be found in [27]). In addition, inspired by Simo *et al*. [35], we propose a hard positive sample mining. After the forward-propagation of all n positive samples, only the first k pairs with larger distances are selected as hard positive samples for back-propagation. The remaining easy positive samples are not back-propagated through the network.

We adopt a triplet network [3, 21] for metric learning. The input triplets are composed of two negative pairs and one positive pair. Similar as the *in-triplet* negative mining strategy introduced by Balntas *et al*. [3], we replace the feature distance by the matching probability. Thus the hard negative mining for metric learning in this work is selecting the sample that has a larger negative matching probability.

3.3. Shared feature network

To demonstrate the effectiveness of our exponential loss functions, we require a general network for both descriptor learning and metric learning. Unfortunately, due to the specific objectives, no such network currently exists. Therefore, we design a feature network based on the SPP network [39], which can be shared by both descriptor learning and metric learning. Figure 3 shows its architecture which contains four convolutional blocks. Each block has a structure: Conv-BN-ReLU-Conv-BN-ReLU. The second and third blocks contain a dilated convolution. A SPP operator [14] is added on the top of convolutional blocks, which has a 4-level pyramid pooling $(8 \times 8, 4 \times 4, 2 \times 2, 1 \times 1)$.

For descriptor learning, we construct a Siamese network [21, 25] that has two branches. They have the same structure and consist of the shared feature network following a fully connected layer for generating feature descriptors. When the positive samples are fed into the network, the hard samples are mined according to the procedure described in Section 3.2.

For metric learning, we construct a triplet network that has three same branches. Each branch also has the shared feature network but following two fully connected layers. The last layer outputs the matching probability. Similar to the 2-ch method [1, 39], each input pair is transformed as a 2-channel image format. The three branches take two negative and one positive pairs as input respectively, and generate three matching probabilities. Then, the hard negative sample mining is applied. More details of network structures are given in supplementary material.

4. Experiments

To demonstrate the superiority of our proposal, we compare it with the state-of-the-arts [19, 21, 27, 36, 39, 40] on three benchmarks: UBC dataset [8], RGB-NIR scene dataset [9] and Hpatches [4]. The first two datasets are mainly used to evaluate patch matching performance, which is measured by the false positive rate at 95% recall (FPR95). The smaller FPR95 is, the better performance the method achieves. Hpatches is designed specifically to evaluate the robustness of descriptors. The test is evaluated by the mean average precision (mAP). The larger mAP is better.

4.1. Datasets

UBC Benchmark is also known as Brown datasets, which contains three subsets: *Liberty*, *Notredame* and *Yosemite*, and the number of unique patches is 450k for Liberty, 468k for Notredame and 634k for Yosemite. Patches in the dataset are extracted by using Difference of Gaussian (DOG) or Harris detector. Patches only corresponding to the same 3D Point are considered as matched. Following [19, 27, 36], we train on each of the three sets, and then report the FPR95 on the other two subsets, and the mean and standard deviation (STD) of all subsets.

RGB-NIR Benchmark is a cross-spectral image matching benchmark. It consists of 477 images captured in RGB and Near-infrared (NIR). All image patches are extracted by SIFT [23] in the RGB images. Half are matching pairs and the other half are non-matching ones. Following [1, 32], we train on the *country* subset, and report the FPR95 on the other 8 subsets: *field*, *forest*, *indoor*, *mountain*, *oldbuilding*, *street*, *urban* and *water*.

Hpatches Benchmark is a larger and more comprehensive dataset which is proposed recently. It consists of 116 sequences, of which 57 sequences are mainly affected by illumination and 59 sequences are mainly affected by geometric deformation. Each sequence includes a reference image and five target images. Additional geometric noise is introduced when extracting key points by DoG, Hessian-Hessian and Harris-Laplace. According to the size of noise, the samples are divided into three levels: EASY, HARD and TOUGH. The benchmark defines three tasks: patch verification, image matching, and patch retrieval. We use mAP to measure the performance for all three tasks.

4.2. Training settings

The input in the experiments is 64*64 gray image patches. For descriptor learning, networks are trained by the SGD optimizer [7] with an initial learning rate of 0.1,

β \ γ	0.1	0.3	0.5	1.0
1.0	1.05	0.90	1.16	1.40
2.0	1.14	**0.75**	1.12	0.95
3.0	1.01	0.78	0.79	0.81
4.0	1.55	0.91	0.78	0.79

(a) Exp-TLoss (prob) with varied β and γ.

β \ γ	0.5	1.0	2.0	3.0
0.5	1.97	1.09	2.24	6.23
1.0	2.22	1.37	1.12	2.00
2.0	1.15	2.23	**1.03**	1.11
3.0	1.11	1.99	1.22	1.07

(b) Exp-TLoss (dist) with varied β and γ.

bs \ *mr*	0:1	1:8	1:4	1:2	1:1
64	1.13	1.07	1.11	1.23	1.27
128	1.09	1.12	1.07	**1.03**	1.13
256	1.11	1.14	1.10	1.09	1.35
512	1.25	1.22	1.29	1.34	1.49

(c) FPR95 under varied *bs* and *mr*.

Table 1: Ablation study on varied exponential orders, β and γ, batch size, and mining ratio. (a) FPR95 achieved by Exp-TLoss with varied β and γ in metric learning; (b) FPR95 achieved by Exp-TLoss with varied β and γ in descriptor learning; (c) FPR95 achieved by varied batch size (bs) and mining ratio (mr) in descriptor learning.

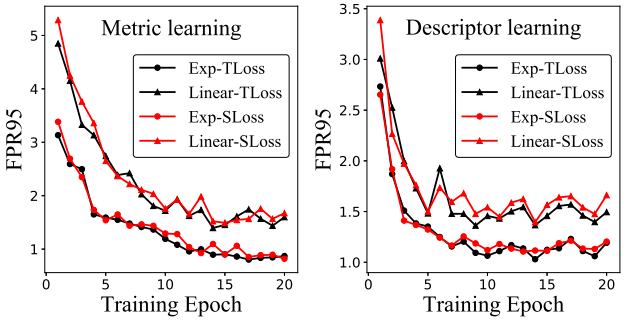


Figure 4: The training efficency of linear and exponential losses in metric (left) and descriptors (right) learning.

momentum of 0.9 and weight decay of 1e-5. We set the margin $\alpha = 2$, train our networks 20 epochs totally. The learning rate is halved by every 5 epochs. For metric learning, we set the margin $\alpha = 1$, and use Adam optimizer [20] with an initial learning rate of 2e-4. For a fair comparison, only immediate flip and horizontal rotation are applied for data augmentation. The training is done from scratch based on Pytorch [30] and NVIDIA GeForce RTX 2080 Ti.

4.3. Ablation study

The exponential order selection. As exponential orders β and γ control the change rate of loss with respect to the pairwise matching probability or the feature distance, we first exam how they affect the performance of network. We vary β and γ and train networks on the *Liberty* subset from UBC datasets [8]. The average FPR95 of the other two subsets are computed and shown in Table 1 (a) and (b). One can see that the metric learning network reaches the lowest FPR95 $= 0.75$ when $\beta = 2$ and $\gamma = 0.3$, the descriptor learning network achieves the best result when $\beta = 2$ and $\gamma = 2$. Then, we apply these settings to exam the training efficiency while varied loss functions are applied. Figure 4 shows the FPR95 against training epochs. Clearly, exponential losses converge faster than linear losses, and triplet losses perform slightly better than Siamese losses. Therefore, we apply the exponential triplet loss function, **Exp-TLoss**, with the best parameter settings for subsequent experiments.

Methods	Fature dim	Mean
Linear-TLoss (prob) +	128	1.56
Exp-TLoss (prob) +	128	**0.98**
Linear-TLoss (dist) +	128	1.25
Exp-TLoss (dist) +	128	**1.00**

Table 3: FPR95 of HardNet [27] using exponential loss functions. + indicates data augmentation.

The effectiveness of hard positive sample mining. Since the top k hard samples are selected from n positive samples in a batch, we test how k and batch size affect the performance of network. The mining ratio is defined as $(n - k) : k$. The results using varied batch sizes 64,128,256,512 and mining ratios 0:1,1:8,1:4,1:2,1:1 are listed in Table 1 (c). One can see the training with a batch size of 128 reaches lower FPR95 values on average, meanwhile, the mining ratio, 1:2, yields the best result.

The generalizability of exponential losses. We would like to test how our exponential loss functions affect other networks, and then apply them into HardNet [27], which reported a better performance than other methods. For a fair comparison, we only replace the loss function and keep other settings unchanged. Table 3 shows that exponential losses reduce FPR95 by 37% and 20% for metric and descriptor learning, respectively. Moreover, we observed that HardNet in descriptor learning only needs half of training time to reach the best performance.

4.4. UBC Benchmark Dataset

We first compare our proposal with eight state-of-the-arts on the most widely used UBC datasets, and list the FPR95 of all subsets, mean and STD in Table 2. For metric learning, the performance varies significantly on different subsets. Compared with SNet [21], our proposal reduces the FPR95 by 88% and 78% when trained on *Liberty*. Because the patches in *Liberty* are mainly affected by rotation and perspective change, exponential loss forces the network to learn more from these hard samples for a better accuracy. When trained on *Notredame*, our proposal is inferior to SNet. We think there are two reasons: (1) *Notredame*

Method	Feature Dim	NOT	YOS	LIB	YOS	LIB	NOT	Mean	STD
		LIB		NOT		YOS			
Metric Learning									
SIFT[23]	128	29.84		22.5		27.29		26.55	3.04
MatchNet[13]	4096	6.90	10.77	3.87	5.67	10.88	8.39	7.74	2.56
SNet-GLoss+[21]	256	6.39	8.43	1.84	2.83	6.61	5.57	5.27	2.27
DeepCompare 2ch-deep+[39]	256	4.55	7.40	2.01	2.52	4.75	4.38	4.27	1.75
DeepCompare 2ch-2stream+[39]	256	4.85	7.20	1.90	2.11	5.00	4.10	4.19	1.81
CS SNet-GLoss+[21]	384	3.69	4.91	**0.77**	**1.14**	**3.09**	2.67	2.71	1.42
Exp-TLoss (prob)+	128	**0.44**	**1.07**	1.63	1.85	3.78	**2.27**	**1.84**	**1.04**
Descriptor Learning									
TL+AS+GOR[40]	128	1.95	5.40	4.80	5.15	6.45	2.38	4.36	1.63
L2-Net[36]	128	3.64	5.29	1.15	1.62	4.43	3.30	3.23	1.46
CS L2-Net[36]	256	2.55	4.24	0.87	1.39	3.81	2.84	2.61	1.20
L2-Net+[36]	128	2.36	4.70	0.72	1.29	2.57	1.71	2.22	1.27
CS L2-Net+[36]	256	1.71	3.87	**0.56**	**1.09**	2.07	1.30	1.76	1.05
Scale aware[19]	128	0.68	2.51	1.79	1.64	2.96	1.02	1.64	0.79
HardNet+[27]	128	0.53	1.96	1.49	1.84	2.51	0.78	1.51	0.69
Exp-TLoss (dist)	128	**0.47**	**1.32**	1.16	1.10	**2.01**	**0.67**	**1.12**	**0.49**

Table 2: PFR95 comparisons among our proposal and state-of-the-arts on UBC Benchmark. + indicates data augmentation. LIB: Liberty, NOT: Notredame, YOS: Yosemite.

suffers more geometric deformation and results in some extremely hard samples, which affect the generalizability of our proposal; (2) SNet uses a central-surround (CS) network [39] that learns multi-scale information and is robust to geometric deformation. Although CS improves the performance, the complexity of network and training cost rise significantly. To maintain efficiency, we do not adopt CS in our proposal and still outperforms other competing methods. Although our network architecture is comparable to DeepCompare, we achieve a significant improvement.

For descriptor learning, we have similar results. Our proposal is slightly inferior to CS L2-Net [36] when trained on *Notredame*, because it involves the CS structure. Compared with the best descriptor learning method HardNet [27], our proposal achieves the lower FPR95 values on all subsets without using any data augmentation. Moreover, the lower mean and STD of FPR95 indicate that our method is more robust on different data.

To illustrate the effectiveness and efficiency, we also plot FPR95 against training cost in Fig. 5. It shows that our proposal outperforms others on performance and efficiency.

4.5. RGB-NIR scene dataset

To demonstrate the generalizability of our proposal, we conduct an experiment on this cross-spectral image patch matching benchmark. Due to the different imaging mechanisms, there exist a significant appearance difference between image pairs. We compare our proposal with the state-of-the-arts that can work on cross-spectral data, and show

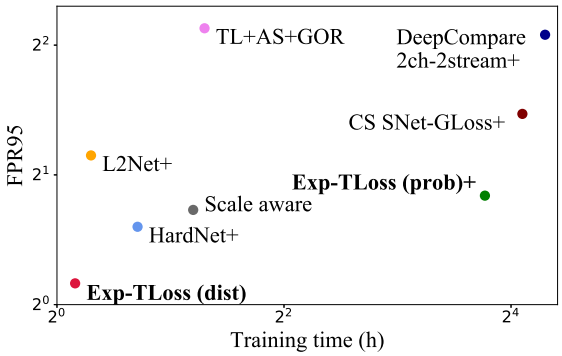


Figure 5: FPR95 against training cost of all competing descriptor and metric learning methods. Ours are in **bold**.

the results in Table 4. For metric learning, our proposal has the lowest mean of FPR95. However, the performance on *field* is inferior to SCFDM [32], which leads to a larger STD over all subsets. We find that *Filed* is the most challenging subset, and SCFDM was designed particularly for cross-spectral data. For descriptor learning, our proposal performs better than HardNet [27] by reducing the mean and STD of FPR95 by 26% and 55%, respectively. Especially, the FPR95 values on two most challenging subsets: *field* and *mountain* are reduced by 50%.

4.6. Hpatches benchmark

We compare our proposal in descriptor learning with TFeat-M* [5], L2Net [36] and HardNet [27] on a more chal-

Methods	Field	Forest	Indoor	Mountain	Oldbuilding	Street	Urban	Water	Mean	STD
Metric Learning										
pseudo-siamese+[1]	17.01	9.82	11.17	11.86	6.75	8.25	5.65	12.04	10.31	3.35
Siamese+[1]	15.79	10.76	11.60	11.15	5.27	7.51	4.60	10.21	9.61	3.44
2-channel+[1]	9.96	0.12	4.40	8.89	2.30	2.18	1.58	6.40	4.47	3.37
SCFDM+[32]	**7.91**	0.87	3.93	5.07	2.27	2.22	**0.85**	4.75	3.48	**2.26**
Exp-TLoss (prob)+	10.15	**0.55**	**1.05**	**1.44**	**1.38**	**1.34**	1.29	**1.84**	**2.42**	2.96
Descriptor Learning										
PN-Net[3]	24.56	3.91	6.56	15.99	6.84	9.51	4.41	15.62	10.92	6.74
PN-Net+[3]	20.09	3.27	6.36	11.53	5.19	5.62	3.31	10.72	8.26	5.32
Q-Net[2]	20.80	3.12	6.11	12.32	5.42	6.57	3.30	11.24	8.61	5.57
Q-Net+[2]	17.01	2.70	6.16	9.61	4.61	3.99	2.83	8.44	6.86	4.48
L2-Net+[36]	16.77	0.76	2.07	5.98	1.89	2.83	**0.62**	11.11	5.25	5.44
HardNet+[27]	10.89	**0.22**	**1.87**	3.09	**1.32**	**1.30**	1.19	2.54	2.80	3.17
Exp-TLoss (dist)+	**5.55**	0.24	2.30	**1.51**	1.45	2.15	1.44	**1.95**	**2.07**	**1.44**

Table 4: FPR95 comparisons among our proposal and state-of-the-arts on RGB-NIR scene dataset.

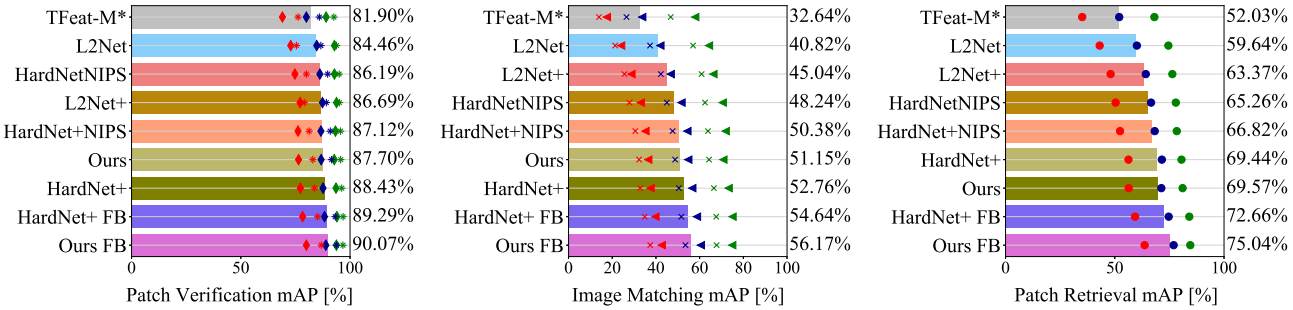


Figure 6: Performance comparisons in verification, matching and retrieval tasks on HPatches dataset. Color of the marker indicates noise level: easy (green), hard (blue), tough (red). DIFFSEQ ($*$) and SAMESEQ ($\blacklozenge$) indicate the source of negative examples in verification task. ILLUM ($\times$) and VIEWPT ($\blacktriangleleft$) indicate the influence of illumination and viewpoint changes in matching task. Suffix indicates the training set (FB: the entire UBC dataset, no suffix: Liberty).

lenging Hpatches benchmark [4]. Figure 6 shows the results. When methods are trained on *Liberty*, our proposal outperforms HardNet slightly only in patch retrieval task. When trained on the entire UBC dataset, our proposal performs the best in all tasks. We think the performance gain comes from a larger scale hard sample mining. Then exponential loss forces network to learn more from these informative data. Eventually, network becomes more robust on different tasks. Note that our proposal improves the performance greater in *tough* and *hard* tests than *easy* test without applying data augmentation, and consistently achieves improvement for sequences with illumination (ILLUM) and viewpoint (VIEWPT) changes which are the most challenging task in image matching. All above results demonstrate a better robustness of our proposal.

5. Conclusion

In this paper, we proposed exponential triplet and Siamese losses for patch matching tasks, which enable the network to naturally put less attention on easy training samples and focus more on hard samples to accelerate learning. To verify the effectiveness of exponential losses, we designed a shared feature network that can be applied to both descriptor learning and metric learning. With the assistance of hard sample mining, our proposal outperforms other state-of-the-arts in terms of effectiveness and efficiency on UBC benchmark. In addition, it also performs well in cross-spectral image matching and image retrieval tasks, demonstrates a better generalizability.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (No.61771379), the Foundation for Innovative Research Groups of the National Natural Science Foundation of China (No.61621005), the Fundamental Research Funds of the Central Universities of China (No.JC1904), the Program for Cheung Kong Scholars and Innovative Research Team in University (No.IRT_15R53).

References

[1] Cristhian A Aguilera, Francisco J Aguilera, Angel D Sappa, Cristhian Aguilera, and Ricardo Toledo. Learning cross-spectral similarity measures with deep convolutional neural networks. In *CVPR*, 2016. 5, 8

[2] Cristhian A Aguilera, Angel D Sappa, Cristhian Aguilera, and Ricardo Toledo. Cross-spectral local descriptors via quadruplet network. *Sensors*, 17(4):873, 2017. 1, 8

[3] Vassileios Balntas, Edward Johns, Lilian Tang, and Krystian Mikolajczyk. Pn-net: Conjoined triple deep network for learning local image descriptors. *CoRR*, abs/1601.05030, 2016. 1, 2, 4, 5, 8

[4] Vassileios Balntas, Karel Lenc, Andrea Vedaldi, and Krystian Mikolajczyk. Hpatches: A benchmark and evaluation of handcrafted and learned local descriptors. In *CVPR*, 2017. 5, 8

[5] Vassileios Balntas, Edgar Riba, Daniel Ponsa, and Krystian Mikolajczyk. Learning local feature descriptors with triplets and shallow convolutional neural networks. In *BMVC*, 2016. 2, 3, 4, 7

[6] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *ECCV*, 2006. 1

[7] Léon Bottou. *Stochastic Gradient Descent Tricks*, volume 7700, pages 421–436. 01 2012. 5

[8] Matthew Brown and David G. Lowe. Automatic panoramic image stitching using invariant features. *IJCV*, 74(1):59–73, 2007. 2, 5, 6

[9] Matthew Brown and Sabine Süsstrunk. Multi-spectral sift for scene category recognition. In *CVPR*, 2011. 5

[10] Jia-Ren Chang and Yong-Sheng Chen. Pyramid stereo matching network. In *CVPR*, 2018. 1

[11] Christopher B Choy, JunYoung Gwak, Silvio Savarese, and Manmohan Chandraker. Universal correspondence network. In *NIPS*, 2016. 2, 3

[12] R. Hadsell, S. Chopra, and Y. Lecun. Dimensionality reduction by learning an invariant mapping. In *CVPR*, 2006. 2

[13] Xufeng Han, Thomas Leung, Yangqing Jia, Rahul Sukthankar, and Alexander C. Berg. Matchnet: Unifying feature and metric learning for patch-based matching. In *CVPR*, 2015. 1, 2, 7

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *ECCV*, 2014. 5

[15] Alexander Hermans, Lucas Beyer, and Bastian Leibe. In defense of the triplet loss for person re-identification. *CoRR*, abs/1703.07737, 2017. 3

[16] Elad Hoffer and Nir Ailon. Deep metric learning using triplet network. *CoRR*, abs/1412.6622, 2014. 2

[17] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015. 4

[18] Wang Jian, Zhou Feng, Shilei Wen, Liu Xiao, and Yuanqing Lin. Deep metric learning with angular loss. In *ICCV*, 2017. 3

[19] Michel Keller, Zetao Chen, Fabiola Maffra, Patrik Schmuck, and Margarita Chli. Learning deep descriptors with scale-aware triplet networks. In *CVPR*, June 2018. 1, 2, 3, 4, 5, 7

[20] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014. 4, 6

[21] Vijay Kumar B G, Gustavo Carneiro, and Ian Reid. Learning local image descriptors with deep siamese and triplet convolutional networks by minimising global loss functions. In *CVPR*, June 2016. 1, 2, 5, 6, 7

[22] Yann Lecun, Leon Bottou, Y Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86:2278 – 2324, 12 1998. 4

[23] David G. Lowe. Distinctive image features from fpr95-invariant keypoints. *IJCV*, 60(2):91–110, 2004. 1, 5, 7

[24] Zixin Luo, Tianwei Shen, Lei Zhou, Siyu Zhu, Runze Zhang, Yao Yao, Tian Fang, and Long Quan. Geodesc: Learning local descriptors by integrating geometry constraints. In *ECCV*, 2018. 1, 4

[25] Jonathan Masci, Davide Migliore, Michael M. Bronstein, and Jürgen Schmidhuber. Descriptor learning for omnidirectional image matching. *CoRR*, abs/1112.6291, 2011. 2, 5

[26] Iaroslav Melekhov, Juho Kannala, and Esa Rahtu. Siamese network features for image matching. In *ICPR*, 2017. 2, 3

[27] Anastasiya Mishchuk, Dmytro Mishkin, Filip Radenovic, and Jiri Matas. Working hard to know your neighbor's margins: Local descriptor learning loss. *CoRR*, abs/1705.10872, 2017. arXiv. 1, 2, 4, 5, 6, 7, 8

[28] Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, 2010. 4

[29] Jiazhi Ni, Liu Jie, Chenxin Zhang, Ye Dan, and Zhirou Ma. Fine-grained patient similarity measuring using deep metric learning. In *CIKM*, 2017. 3

[30] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017. 6

[31] James Philbin, Ondrej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *CVPR*, 2007. 1

[32] Dou Quan, Shuai Fang, Xuefeng Liang, Shuang Wang, and Licheng Jiao. Cross-spectral image patch matching by learning features of the spatially connected patches in a shared space. In *ACCV*, 2018. 5, 7, 8

[33] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, 2015. 3

[34] Johannes L Schönberger, Hans Hardmeier, Torsten Sattler, and Marc Pollefeys. Comparative evaluation of hand-crafted and learned local features. In *CVPR*, 2017. 3

[35] Edgar Simo-Serra, Eduard Trulls, Luis Ferraz, Iasonas Kokkinos, Pascal Fua, and Francesc Moreno-Noguer. Discriminative learning of deep convolutional feature point descriptors. In *ICCV*, 2016. 2, 3, 5

[36] Yurun Tian, Bin Fan, and Fuchao Wu. L2-net: Deep learning of discriminative patch descriptor in euclidean space. In *CVPR*, 2017. 1, 2, 5, 7, 8

[37] Chao-Yuan Wu, R Manmatha, Alexander J Smola, and Philipp Krähenbühl. Sampling matters in deep embedding learning. In *ICCV*, 2017. 3, 4

[38] Kwang Moo Yi, Eduard Trulls, Vincent Lepetit, and Pascal Fua. LIFT: learned invariant feature transform. In *ECCV*, 2016. 3

[39] Sergey Zagoruyko and Nikos Komodakis. Learning to compare image patches via convolutional neural networks. In *CVPR*, 2015. 1, 2, 5, 7

[40] Xu Zhang, Felix X. Yu, Sanjiv Kumar, and Shih-Fu Chang. Learning spread-out local feature descriptors. In *ICCV*, 2017. 1, 2, 5, 7

[41] Siyu Zhu, Runze Zhang, Lei Zhou, Tianwei Shen, Tian Fang, Ping Tan, and Long Quan. Very large-scale global sfm by distributed motion averaging. In *CVPR*, June 2018. 1