Learning Spread-out Local Feature Descriptors

Xu Zhang[1], Felix X. Yu[2], Sanjiv Kumar[2], Shih-Fu Chang[1]

[1]Columbia University, [2] Google Research

{xu.zhang, sc250}@columbia.edu, {felixyu, sanjivk}@google.com

Abstract

We propose a simple, yet powerful regularization technique that can be used to significantly improve both the pairwise and triplet losses in learning local feature descriptors. The idea is that in order to fully utilize the expressive power of the descriptor space, good local feature descriptors should be sufficiently "spread-out" over the space. In this work, we propose a regularization term to maximize the spread in feature descriptor inspired by the property of uniform distribution. We show that the proposed regularization with triplet loss outperforms existing Euclidean distance based descriptor learning techniques by a large margin. As an extension, the proposed regularization technique can also be used to improve image-level deep feature embedding.

1. Introduction

Computing image patch correspondences based on local descriptor matching is important in many computer vision problems such as image retrieval, wide baseline stereo matching and panorama building. The main challenge of finding correct correspondences is that the appearance of the image patches varies due to changes of scaling, view angle, illumination and imaging condition etc. Designing local feature descriptors that are invariant to such changes is therefore essential. Efforts of local descriptor fall into two categories: hand-crafted and learning-based. Hand-crafted descriptors try to achieve the invariance by manually selected rules. One of the most popular hand-crafted descriptors is SIFT [10] and its variants [2, 24], which are widely used in the computer vision community. The main issue of the hand-crafted descriptors is that they can only consider a limited predefined set of variations.

One approach to take all variations into consideration is learning local descriptors from a large patch correspondence dataset [3, 20]. The state-of-the-art descriptor learning methods are based on neural networks [1, 8, 19, 26]. In addition to the model itself, the most important aspect of learning-based method is the loss function which defines

the goal of descriptor learning: matching patches should be close in the descriptor space, while the non-matching patches should be far-away[1]. The pairwise loss and triplet loss (Section 2) are the commonly used loss functions to achieve the desired properties. Recently, there are a lot of works such as smart sampling strategies [1, 12] and structured loss [23] that improve the triplet loss. In particular, Kumar *et al*. [8] propose to use a global loss to separate the distance distributions of the matching pairs and non-matching pairs. This approach avoids the design of complicated sampling strategies and is also shown to provide results that are robust to training with outliers.

The success of global loss motivates us to further explore the desired properties of the descriptor space and design a robust regularization term based on these properties. Our main idea is that the good local feature descriptors should be sufficiently "spread-out" in the descriptor space in order to fully utilize the expressive power of the space. Specifically, we introduce a regularization term that induces the spread-out condition, inspired by the properties of the uniform distribution on unit sphere (Section 3). The regularization can be easily used to improve all methods where pairwise or triplet loss is used. We show that the proposed regularization with triplet loss, without hard sample mining, outperforms all the Euclidean distance based descriptors by a large margin (Section 5). In particular, it outperforms the global loss [8] in the patch pair classification task. As an extension of descriptor learning, we show that the proposed regularization can also be used in improving image-level deep feature embedding (Section 6).

2. Background

We begin by reviewing some commonly used loss functions in learning local feature descriptors. Let $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, $\mathbf{x}_i \in \mathbb{R}^{m \times n}$ denote a set of N training patches with $m \times n$ pixels. $\{y_{ij}, 1 \leq i, j \leq N\}$ is a set of pairwise labels for $\mathcal{X}$ indicating whether $\mathbf{x}_i$ and $\mathbf{x}_j$ belong to the same class ($y_{ij} = 1$) or not ($y_{ij} = 0$). In this paper, we call the pairs with $y_{ij} = 1$ the matching pairs,

---

[1]We use Euclidean distance in this paper. See Section 2 for more details, and Section 3 for the discussion on alternatives.

and the pairs with $y_{ij} = 0$ the non-matching pairs. The goal of descriptor learning is to learn a feature embedding $f(\cdot) : \mathbb{R}^{m \times n} \mapsto \mathbb{R}^d$ that maps raw patch pixels to a d dimensional vector, such that $\| f(\mathbf{x}_i) - f(\mathbf{x}_j) \|_2$ is small when $y_{ij} = 1$ and $\| f(\mathbf{x}_i) - f(\mathbf{x}_j) \|_2$ is large when $y_{ij} = 0$. In this paper, we assume that $f(\cdot)$ lives on the unit sphere, i.e., $\| f(\mathbf{x}) \|_2 = 1, \forall \mathbf{x} \in \mathbb{R}^{m \times n}$.

2.1. Pairwise loss

The pairwise loss tries to directly induce small distance for matching pairs and large distance for non-matching pairs. An input for pairwise loss is of the form $(\mathbf{x}_i, \mathbf{x}_j, y_{ij})$, consisting of a pair of samples and their corresponding label. The most widely used pairwise loss is the contrastive loss:

$$
\begin{aligned}
\ell_{\mathrm{con}} = \; & y_{ij} \max(0, \| f(\mathbf{x}_i) - f(\mathbf{x}_j) \|_2 - \epsilon^+) \\
& + (1 - y_{ij}) \max(0, \epsilon^- - \| f(\mathbf{x}_i) - f(\mathbf{x}_j) \|_2),
\end{aligned} \tag{1}
$$

where $f(\cdot)$ is the feature embedding. ϵ^+ and ϵ^- control the margins of the matching and non-matching pairs respectively. Contrastive loss was originally proposed in [4] with $\epsilon^+ = 0$. As shown in [9], this often leads to overfitting. And a proper relaxed margin ($\epsilon^+ > 0$) can achieve better performance. The main problem with the pairwise loss is that the margin parameters are often difficult to choose [25].

2.2. Triplet loss

Triplet loss takes a triplet of samples as input. One triplet consists of three samples: $(\mathbf{x}_i, \mathbf{x}_j, \mathbf{x}_k)$, with $y_{ij} = 1$ and $y_{ik} = 0$. To simplify the notation, we denote one triplet as $(\mathbf{x}_i, \mathbf{x}_i^+, \mathbf{x}_i^-)$, where $\mathbf{x}_i^+ = \mathbf{x}_j$ and $\mathbf{x}_i^- = \mathbf{x}_k$. One commonly used triplet loss is the ranking loss [18]:

$$
\begin{aligned}
\ell_{\mathrm{tri}} = \max \Big(0, & \epsilon - (\| f(\mathbf{x}_i) - f(\mathbf{x}_i^-) \|_2 \\
& - \| f(\mathbf{x}_i) - f(\mathbf{x}_i^+) \|_2) \Big)
\end{aligned} \tag{2}
$$

where ϵ is a margin. The idea of ranking loss is to separate the matching sample and the non-matching sample by at least a margin ϵ. The main difference between pairwise loss and triplet loss is that pairwise loss considers the absolute distances of the matching pairs and non-matching pairs, while triplet loss considers the relative difference of the distances between matching and non-matching pairs. Since the quality of the embeddings largely depends on the relative ordering of the matching pairs and non-matching pairs, triplet loss shows better performance than pairwise loss in local descriptor learning [1, 8].

2.3. Improvements

The main issue of triplet loss and pairwise loss is that as the number of training samples grows, sampling all the possible triplets/pairs becomes infeasible, and only a relatively small portion of triplets/pairs can be used in training. As

observed in practice, the training is often ineffective since many of the sampled triplets/pairs will satisfy the constraint within just a few training steps. One possible solution is to remove the "easy samples" and add new "hard samples" to the training set. However, determining which samples to remove or add is a challenging task [17, 19]. Additionally, focusing only on the samples that violate the training constraints the most will lead to overfitting [17].

Balntas *et al*. [1] propose an improved version of triplet loss by applying in-triplet hard negative mining. The idea is that one triplet contains two non-matching pairs $(\mathbf{x}_i, \mathbf{x}_i^-)$ and $(\mathbf{x}_i^+, \mathbf{x}_i^-)$, and choosing the one that violates the triplet constraint more will make training more effective. They call this technique "anchor swap".

Kumar *et al*. [8] propose a global loss and combine it with the traditional triplet loss to address the sampling issue in pairwise and triplet loss. Instead of considering sample pair or triplet, the global loss considers all matching and non-matching pairs in one training batch, and calculate the empirical mean and variance of the distance of the matching and non-matching pairs. The main idea of the global loss is to separate two empirical means by a margin and minimize the variances. There are two drawbacks of this method. First, the distribution of the distance of the matching pairs can vary greatly across different classes, and using a batch of randomly sampled matching pairs to estimate that distribution is unstable. Second, the extra margin in global loss adds extra complexity for training.

Structured loss [12, 13, 14, 21, 23] considers all the possible matching and non-matching pairs in one batch of samples. By carefully designing the loss functions, structured loss has the ability to focus on the "hard" pairs in training. Song *et al*. [14] propose the lifted structured similarity softmax loss (LSSS). N-pair loss [21] further develops the idea by using a more effective batch construction method.

The motivation of this paper is that good descriptors should fully utilize the expressive power of the whole space ("spread-out" in the descriptor space). We also propose a simple regularization term, global orthogonal regularization, to encourage the "spread-out" property. The global orthogonal regularization can be easily incorporated into other losses. Experiments show that the proposed regularization can improve the performance of different types of losses, especially those originally without the "spread-out" property.

3. Methodology

3.1. "Spread-out" local descriptors

The main idea of this paper is that in order for the descriptors to fully utilize the descriptor space, it should be sufficiently "spread-out" in the descriptor space. On the contrary, suppose there is part of the space where no fea-
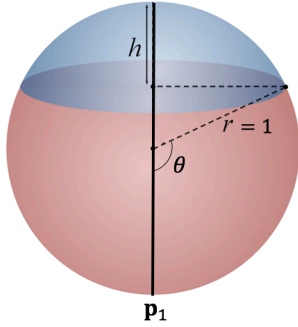
Figure 1. Valid area for $\mathbf{p}_1^T\mathbf{p}_2 \leq s = \cos\theta$. If $\mathbf{p}_2$ is on the blue spherical cap, $\mathbf{p}_1^T\mathbf{p}_2 \leq \cos\theta$, otherwise, $\mathbf{p}_1^T\mathbf{p}_2 > \cos\theta$

ture descriptor appears, the learned feature descriptor is not fully utilizing the expression power of the space.

One intuitive way to characterize "spread-out" is that: *Given a dataset, we say that the learned descriptors are spread-out if two randomly sampled non-matching descriptors are close to orthogonal with a high probability.* As an obvious example, we notice that uniform distribution has such property.

Proposition 1. *Let* $\mathbf{p}_1, \mathbf{p}_2 \in \mathbb{S}^{d-1}$ *be two points independently and uniformly sampled from the unit sphere in* d-*dimensional space. Each point is represented by a* d-*dimensional* ℓ_2 *normalized vector. The probability density of* $\mathbf{p}_1^T\mathbf{p}_2$ *satisfies*

$$
p(\mathbf{p}_1^T\mathbf{p}_2 = s) = \begin{cases} \dfrac{(1-s^2)^{\frac{d-1}{2}-1}}{B(\frac{d-1}{2}, \frac{1}{2})} & -1 \leq s \leq 1 \\ 0 & otherwise, \end{cases}
$$

where $B(a,b)$ *is the beta function.*

Proof. Since $-1 \leq \mathbf{p}_1^T\mathbf{p}_2 \leq 1$, the second equation is obvious. To show the first equation, we calculate the cumulative distribution first. Here we only consider the case when $-1 \leq s < 0$, and $0 \leq s < 1$ can be shown similarly. Without loss of generality, we fix $\mathbf{p}_1$ and also assume that $s = \cos\theta$, as shown in Figure 1. Since $\cos(\cdot)$ is a monotone function in $[0, \pi]$, $\mathbf{p}_1^T\mathbf{p}_2 \leq s$ if and only if $\mathbf{p}_2$ is located on the blue spherical cap. Since $\mathbf{p}_2$ is uniformly sampled from the sphere, the probability of $\mathbf{p}_1^T\mathbf{p}_2 \leq s$ is equal to the area of the blue spherical cap divided by the area of the whole sphere. The area of the $d-1$ dimensional spherical cap is

$$
S = \frac{1}{2}S_0 r^{d-1} I_{(2rh-h^2)/r^2}\left(\frac{d-1}{2}, \frac{1}{2}\right),
$$

where S_0 is the area of the whole sphere. $r = 1$ is the radius of the sphere. h is the height of the spherical cap: $h = r + r\cos(\theta) = r + sr$. $I_x(a, b)$ is the regularized incomplete beta function. Therefore, the cumulative distribution can be
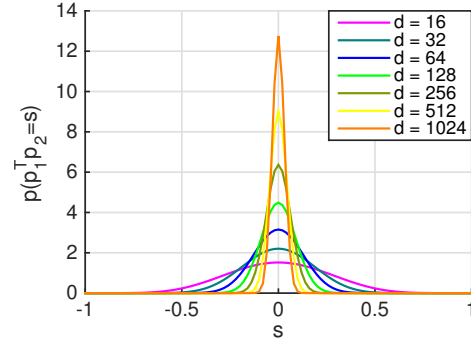

Figure 2. Probability density of inner product of two points which are independently and uniformly sampled from the unit sphere in d-dimensional space. We can see that, in high dimensional space, most pairs are close to orthogonal.

written as,

$$
P(\mathbf{p}_1^T\mathbf{p}_2 \leq s) = \frac{1}{2}I_{1-s^2}\left(\frac{d-1}{2}, \frac{1}{2}\right), \quad -1 \leq s < 0.
$$

The probability density is the derivative of the cumulative distribution. □

Figure 2 shows the probability distribution of the inner product (cosine similarity) of two points independently and uniformly sampled from the unit sphere[2]. It shows that with high probability, the two independently and uniformly sampled points are close to orthogonal.

Based on the above observation, one might hope to make the distribution of the learned descriptors matches that of the uniform distribution. This is not practical in two ways. 1) How the learned descriptors distribute depends not only on the learned model, but also on the natural distribution of the image patches (not controllable). 2) It is technically difficult to match two distributions in practice. Instead, in this paper, we propose a regularization technique inspired by the theoretic properties of the uniform distribution on unit sphere. The regularization encourages the inner product of two randomly sampled non-matching descriptors matches that of two points independently and uniformly sampled from the unit sphere in its *mean and second moment*.

The following proposition shows that for two points that are independently and uniformly sampled on the unit sphere, the mean and the second moment of their inner product are 0 and $1/d$, respectively.

Proposition 2. *Let* $\mathbf{p}_1, \mathbf{p}_2 \in \mathbb{S}^{d-1}$ *be two points independently and uniformly sampled from the unit sphere. The*

---

[2]Note that since we assume the descriptors stay on the unit sphere (ℓ_2 normalized), there is only a sign and constant difference between ℓ_2 distance and cosine similarity, and the cosine similarity equals to the inner product.

mean and the second moment of $\mathbf{p}_1^T \mathbf{p}_2$ are

$$\mathbb{E}(\mathbf{p}_1^T \mathbf{p}_2) = 0 \quad \text{and} \quad \mathbb{E}((\mathbf{p}_1^T \mathbf{p}_2)^2) = \frac{1}{d}.$$

Proof. Due to symmetry, it's easy to show that $\mathbb{E}(\mathbf{p}_1) = \mathbb{E}(\mathbf{p}_2) = \mathbf{0}$, since $\mathbf{p}_1$ and $\mathbf{p}_2$ are independent,

$$\mathbb{E}(\mathbf{p}_1^T \mathbf{p}_2) = \mathbb{E}(\mathbf{p}_1^T)\mathbb{E}(\mathbf{p}_2) = \mathbf{0}^T\mathbf{0} = 0.$$

Since both $\mathbf{p}_1$ and $\mathbf{p}_2$ are uniformly sampled from the unit sphere, when considering the second moment of the inner-product, we can fix one point and let the other point to be uniformly sampled. Without loss of generality, we choose $\mathbf{p}_1 = [1, 0, \ldots, 0]^T$ and denote $\mathbf{p}_2$ as $[p_{21}, \ldots, p_{2d}]^T$, thus,

$$(\mathbf{p}_1^T \mathbf{p}_2)^2 = p_{21} \quad \text{and} \quad \mathbb{E}((\mathbf{p}_1^T \mathbf{p}_2)^2) = \mathbb{E}(p_{21}^2).$$

Due to the symmetry of the sphere, we can have $\mathbb{E}(p_{21}^2) = \mathbb{E}(p_{22}^2) = \ldots = \mathbb{E}(p_{2d}^2)$. Since $\mathbf{p}_2$ is on the unit sphere, $\sum_{i=1}^{d} p_{2i}^2 = 1$. Thus, $\mathbb{E}(p_{21}^2) = 1/d$. $\qquad\square$

3.2. Global orthogonal regularization

We propose a regularization which tries to match the mean and second moment shown in Proposition 2. It encourages that the descriptors of random sampled non-matching pairs have similar statistical property as two points independently and uniformly sampled from the unit sphere. We call this regularization Global Orthogonal Regularization (GOR). Following the notations in Section 2, given a set of N random sampled non-matching patches $\{(\mathbf{x}_i, \mathbf{x}_i^-)\}_{i=1}^{N}$, denote the descriptor function as $f(\cdot)$. The sample mean of the inner product of the descriptors of non-matching pairs is,

$$M_1 = \frac{1}{N}\sum_{i=1}^{N} f(\mathbf{x}_i)^T f(\mathbf{x}_i^-). \tag{3}$$

The sample second moment of the inner product is,

$$M_2(f(\mathbf{x})^T f(\mathbf{x}^-)) = \frac{1}{N}\sum_{i=1}^{N} (f(\mathbf{x}_i)^T f(\mathbf{x}_i^-))^2. \tag{4}$$

The Global Orthogonal Regularization (GOR) is defined as

$$\ell_{\text{gor}} = M_1^2 + \max(0, M_2 - \frac{1}{d}), \tag{5}$$

where d is the dimension of the final output descriptor.

In (5), the first term tries to match the mean of the distributions and the second term tries to make the second moment close to $1/d$. In order to calculate the regularization term, one needs to consider all the non-matching pairs in the training set – this is impractical. In practice we use a sampled batch to estimate its value. The reason for using the hinge loss for the second term is that, in many batches, all the non-matching pairs are already very close to being

orthogonal ($M_2 < 1/d$), and there is no need to force M_2 to be $1/d$. We have tried other loss functions for the second term. ℓ_1 loss results in a similar performance, while ℓ_2 leads to slight degradation.

The proposed regularization term can be used with any loss function. Denote the original training loss as $\ell_{(\cdot)}$, the final loss can be written as,

$$\ell_{(\cdot)\text{\_gor}} = \ell_{(\cdot)} + \alpha\ell_{\text{gor}}, \tag{6}$$

where α is a tunable parameter. In the experiment section, we test combining the global orthogonal regularization with contractive loss[3] (1) [19], triplet loss (2) [1], lifted structured similarity softmax loss (LSSS) [14] and N-pair loss [21].

3.3. Non-Euclidean distance

So far, we assumed that the distance of the descriptors is based on the Euclidean distance. There are several recent works that use a decision network instead of the Euclidean distance to calculate the similarity. Han *et al.* [6] propose to use a Siamese network followed by a decision net. Zagoruyko and Komodakis [27] develop a 2-stream networks, in which one stream focuses on the central area of the patch and the other focuses on the surrounding area of the patch. Kumar *et al.* [8] propose a global loss and combine it with the 2-stream networks to achieve the state-of-the-art performance. The drawback of using a new type of distance rather than Euclidean distance is that efficient large-scale nearest neighbor search method such as locality sensitive hashing (LSH) [15] can no longer be used. In this paper, we focus on training local descriptor in the Euclidean space.

4. Implementation

In this section, we show our training framework based on triplet loss and the proposed global orthogonal regularization. The framework has three branches (as shown in Figure 3). The proposed global orthogonal regularization only considers two branches which process the non-matching pairs. Training pipeline of other losses can be achieved accordingly. For example, for the pairwise loss, we use a network with two branches (known as the Siamese network) instead of three.

Though our method is flexible in terms of the patch sizes, here we follow [6] to use patch size 64×64. Each branch in the triplet/Siamese network has the following structure: {Conv(7,7,32) - MaxP(2,2) - Conv(6,6,64) - MaxP(2,2) - Conv(5,5,128) - MaxP(2,2) - FC(128) - ℓ_2 Norm}. Conv(n, m, c) means convolutional layer with kernel size (n, m) and output channel number c. MaxP(n, m)

---

[3]For contractive loss (1), we substitute the second term in (1) with the proposed regularization, since both terms try to separate the non-matching pairs.
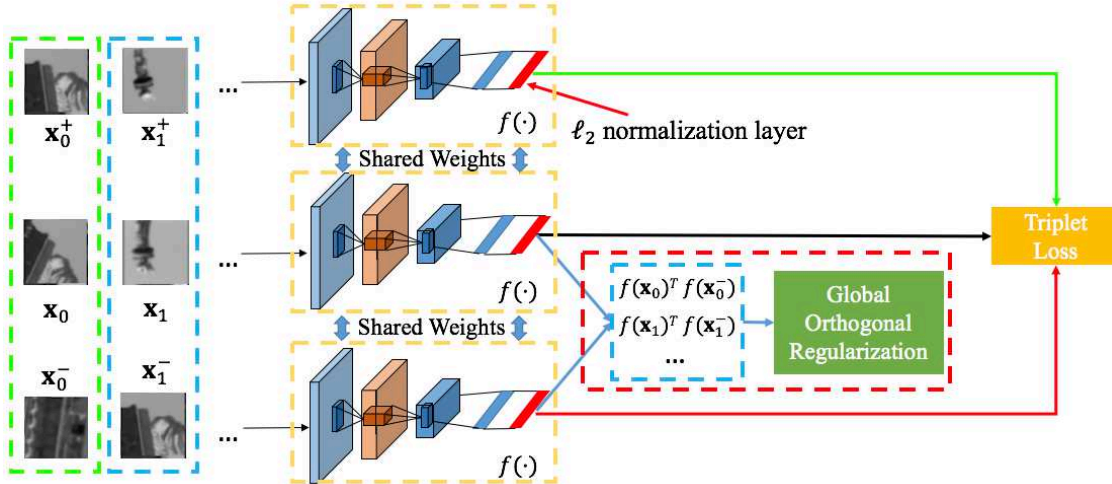
Figure 3. Local feature descriptor training pipeline with triplet loss and the proposed global orthogonal regularization (GOR). GOR can also be used with the pairwise loss. In that case, there will be two branches of the network (known as the Siamese network) instead of three.

is a max pooling layer with size $n \times n$ and stride m. FC(d) is a fully connected layer with output dimension d. ℓ_2 Norm is ℓ_2 normalization layer to guarantee each descriptor has unit norm. All the convolution layers are followed by batch normalization [7] and ReLU. Based on our implementation, when trained without the proposed global regularization, the above network structure achieves similar performance as the one proposed in [1]. The motivation of the use of the above shallow network is for efficiency and avoiding over-fitting [1, 19]. We show the experiment of the proposed method over the large-scale patch descriptor benchmark in the next section.

5. Local Descriptor Result

5.1. Dataset

We first conduct experiments on the standard local patch descriptor benchmark, UBC patch dataset [3]. The dataset contains three subsets, Yosemite, Notre Dame and Liberty. Each subset consists of more than 100k classes which include different image patches corresponding to the same 3D location obtained through a 3D reconstruction from different multi-view images. The total number of local image patches within each subset is more than 450k. Each patch has a size of 64×64 and is sampled around the output of difference of Gaussian (DOG) [10] detector. The scale and orientation of the patch is normalized by the detector. Though with normalized scale and orientation, the patch dataset still contains great variations in view points, lighting, camera conditions *etc*. We follow the evaluation protocol proposed in [3] to separate the whole dataset into six training-test combinations in which one subset is for training and the other for test.

The metric used to evaluate different methods is false positive rate at 95% true positive rate (FPR95), which is the standard metric in previous works [1, 19, 27]. The test split of each subset contains 100k patch pairs in which 50% are matching pairs and the other 50% are non-matching pairs. The test pairs are predefined in [3].

5.2. Training setting and evaluation method

For training, we randomly sample 1M triplets (for triplet network), or 1M matching pairs and 1M non-matching pairs (for Siamese network), for each training subset. No data augmentation or specially designed sampling is used. The training batch size is set to 128. We use SGD with momentum in the optimization. The learning rate starts at 0.1, with momentum 0.9. The learning rate is reduced after each epoch by a factor of 0.96. The trade-off parameter α in (6) is set to 1 (we discuss the choice of α in Section 5.3). The loss function in (5) is hinge loss. The margin for matching pair in Siamese network (ϵ^+ in (1)) is set to 0.7 and the margin of triplet network (ϵ in (2)) is set to 0.5. All are estimated via empirical cross validation. Our implementation is based on TensorFlow [5]. The training of each epoch takes about 10 minutes on a Titan X GPU. All the networks are trained with 20 epochs, and they all converge before the end of training.

We compare our method with a large set of local feature descriptors which use Euclidean distance as similarity metric. The methods include: 1) hand-crafted descriptor (SIFT [10]), conventional machine learning based descriptor (VGG-Opt [20]), 2) deep learning based descriptors learned with pairwise loss (DeepCampare$_{siam}$ [27],

Loss Type	Training Test / Descriptor	Dim	NotreDame Liberty / Yosemite		NotreDame Yosemite / Liberty		Yosemite Liberty / NotreDame		Mean
N/A	SIFT [10]	128	27.29		29.84		22.53		26.55
	VGG-Opt [20]	80	10.08	11.63	11.42	14.58	7.22	6.17	10.28
Pairwise	DeepCompare$_{siam}$[27]	256	15.89	19.91	13.24	17.25	8.38	6.01	13.45
	DeepCompare$_{2str}$[27]	512	13.02	13.24	8.79	12.84	5.58	4.54	9.67
	DeepDesc[19]	128	16.19		8.82		4.54		9.85
	CL+GOR (Ours)	128	6.88	6.99	6.46	8.33	3.73	3.40	5.97
Global	TGLoss[8]	256	9.47	10.65	9.91	13.45	5.43	3.91	8.80
Triplet	TFeat[1]	128	7.95	8.10	7.64	9.88	3.83	3.39	6.79
	TFeat+AS[1]	128	7.08	7.82	7.22	9.79	3.85	3.12	6.47
	TL+GOR (Ours)	128	**4.94**	5.74	5.47	7.13	2.58	2.28	4.69
	TL+AS+GOR (Ours)	128	5.15	**5.40**	**4.80**	**6.45**	**2.38**	**1.95**	**4.36**
Structured	N-pair[21]	128	5.53	8.29	**4.80**	7.51	3.01	2.60	5.29
	N-pair+GOR (Ours)	128	5.16	7.43	5.03	7.10	2.81	2.34	4.98

Table 1. FPR95 (%) of different methods on UBC patch dataset. TL+AS+GOR achieves the lowest FPR95 rate.
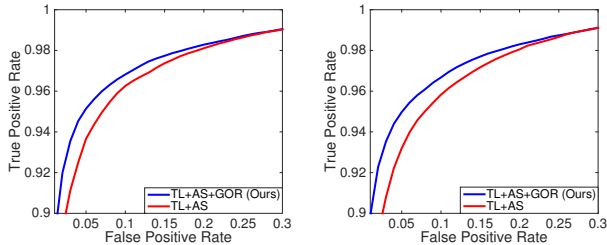
(a) Train: Notre Dame, Test: Liberty.

(b) Train: Notre Dame, Test: Yosemite.

Figure 4. ROC curves for our method and baseline method trained on the Notre Dame subset and tested on the Liberty and Yosemite subsets.

DeepCampare$_{2str}$[4] [27] and DeepDesc [19]), 3) descriptors learned with triplet loss with and without anchor swap (TFeat+AS [1] and TFeat [1]), 4) descriptors learned with global loss (TGLoss [8]), and 5) descriptors learned with structured loss (N-pair [21]).

We combine the proposed Global Orthogonal Regularization with four commonly used losses mentioned in Section 2, namely, contractive loss, triplet loss, triplet loss with anchor swap and N-pair loss [21]. Thus four variants of our method are used in evaluation: contractive loss with global orthogonal regularization (CL+GOR), triplet loss with GOR (TL+GOR), triplet loss with anchor swap [1] and GOR (TL+AS+GOR) and N-pair loss with GOR (N-pair+GOR).

5.3. Patch pair classification result

Classification error. Table 1 summarizes the performance of all the evaluated Euclidean embedding methods on UBC patch dataset. We show FPR95 on each of the six training-test combinations and also the mean over all of them.

By simply applying the proposed global orthogonal regularization, almost all the baseline methods show performance gains. Specifically, among all the pairwise loss based methods, contractive loss with the proposed GOR (CL+GOR) reduces the error of the previous best pairwise loss model (DeepCampare$_{2str}$) from 9.67 to 5.97 with a relative deduction of 38.3%. Among all the triplet loss based methods, triplet loss with the proposed GOR (TL+GOR) reduces the error of its triplet loss baseline (TFeat) from 6.79 to 4.69. For the anchor swap version (TL+AS+GOR vs. TFeat+AS), the error reduces from 6.47 to 4.36. The relative deductions are 30.9% and 32.6%, respectively. For the structured loss, the error was reduced from 5.29 to 4.98. The improvement is not as significant because the N-pair loss already has the ability to force the random non-matching pairs to be orthogonal. The second moment of the non-matching pairs trained with N-pair loss is close to $2/d$, while that of the triplet loss is close to $50/d$. Overall, TL+AS+GOR achieves the lowest FPR95 rate.

The improvement of our method can also be shown using other metrics such as the ROC curves. The ROC curves of TL+AS+GOR and TL+AS both are shown in Figure 4. Here, the training subset is Notre Dame, and the test subsets are Liberty (Figure 4(a)) and Yosemite (Figure 4(b). The result shows that the performance gain of the proposed regularization is universal at different false positive rates.

Similarity histogram. To understand how the proposed GOR affects the distribution of the similarity, the histograms of cosine similarity of the matching pairs and non-matching pairs of the models trained with/without the proposed GOR on test set are shown in Figure 5. We use the same baseline method defined above. This figure shows the setting in which the training subset is Notre Dame and the test subset is Liberty, but the observation is also general for
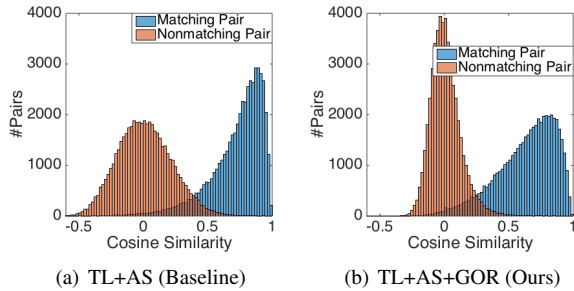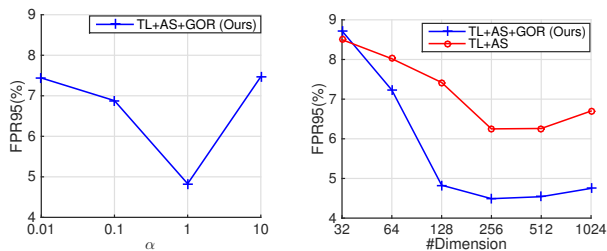
---

[4]Subscript "$2str$" means central-surround network proposed in [27].

(a) TL+AS (Baseline)　　(b) TL+AS+GOR (Ours)

Figure 5. Histogram of cosine similarity of matching pairs and non-matching pairs on "Liberty". The model is trained on "Notre Dame". When trained with GOR, the non-matching pairs are more close to being orthogonal.

(a) FPR95(%) with different values of α (#dimension= 128).

(b) FPR95(%) with different feature dimensions ($\alpha = 1$).

Figure 6. FPR95(%) with different α and embedding dimensions. α trades off the regularization term and the triplet loss. Training set: Notre Dame, Test set: Liberty.

other training/test combinations.

The histogram of the similarity of the matching and non-matching pairs of the baseline method is shown in Figure 5(a), while those of model trained with the proposed regularization is shown in Figure 5(b). The histogram in blue is for matching pairs, while the histogram in orange is for non-matching pairs. The histogram of the similarity of non-matching pairs trained with GOR has a much sharper shape than that without the proposed regularization, which means when trained with GOR, non-matching pairs are more likely to be close to orthogonal. With the proposed GOR, the empirical error (overlapped area in Figure 5(b)) decreases by 15% relatively in comparison with the baseline (Figure 5(a)).

Trade-off parameter. α in (6) controls the trade-off between the triplet loss and GOR. We use Notre Dame as training set and Liberty as test set and show the FPR95 of different models trained with different α values (from 0.01 to 10) in Figure 6(a). When $\alpha = 0$, (6) becomes standard triplet loss. When α is large, the network will enforce the descriptor of all the non-matching pairs (including "hard negatives") to be close to orthogonal.

Embedding dimension. We investigate how the proposed

GOR affects the training of descriptors of different dimensionalities. We change the output node number in final fully-connected layer from $[32, 64, 128, 256, 512, 1024]$. The result is shown in Figure 6(b). The proposed GOR achieves significant performance gain when training a high-dimensional descriptor ($d \geq 64$). The low dimensional case ($d = 32$) does not work as well. One possible reason is that the descriptors of two non-matching patches are harder to be spread-out, and forcing non-matching patches to be orthogonal may lead to error. Finally, both our method and the baseline degrade for very high dimensions. We conjecture this is due to over-fitting. One may think that when d is large, the network may not be able to force the second moment to a very small $1/d$. However, the proposed GOR is only a regularization not a hard constraint. And we can always make a trade-off by changing the value of α in (6).

5.4. Descriptor extraction efficiency

Since there are hundreds of patches in one image, the speed of descriptor extraction is also very important. The proposed GOR only affects the training stage, adding no additional cost in extraction pipeline. Based on our implementation on TensorFlow, when running a Titan X GPU, the extraction speed is about 10K patches per second, which is comparable to the conventional local descriptor extraction method like SIFT [10] and descriptor learning techniques using "shallow" structure such as TFeat and DeepDesc.

6. Extension to image-level embedding

Although GOR is proposed to learn local descriptors, the method can also be used in other applications where a feature embedding is learned. As an example, we show that it can also be used to improve the performance of image-level embedding. We compare our method to LSSS [14], which, as reviewed in Section 2.3, outperforms triplet and pairwise losses.

6.1. Dataset and evaluation metric

The image level feature embedding experiment is conducted on Stanford Online Products dataset [14]. Stanford Online Products dataset contains 120,053 product images crawled from eBay.com. There are a total of 22,634 products belonging to 12 categories. Each product is an individual class and has an average of 5.3 images. We strictly follow the same experiment setting in [14], that using 11,318 classes with a total of 59,551 images for training and another 11,316 classes with 60,502 images for test. The training and test splits have no overlap and are predefined in the dataset. We choose this dataset due to its realistic setting and rich variations within classes.

As in [14], we perform both clustering and retrieval tasks. For the clustering task, the F1 and NMI scores are used as the evaluation metrics [11]. F1 metric computes the
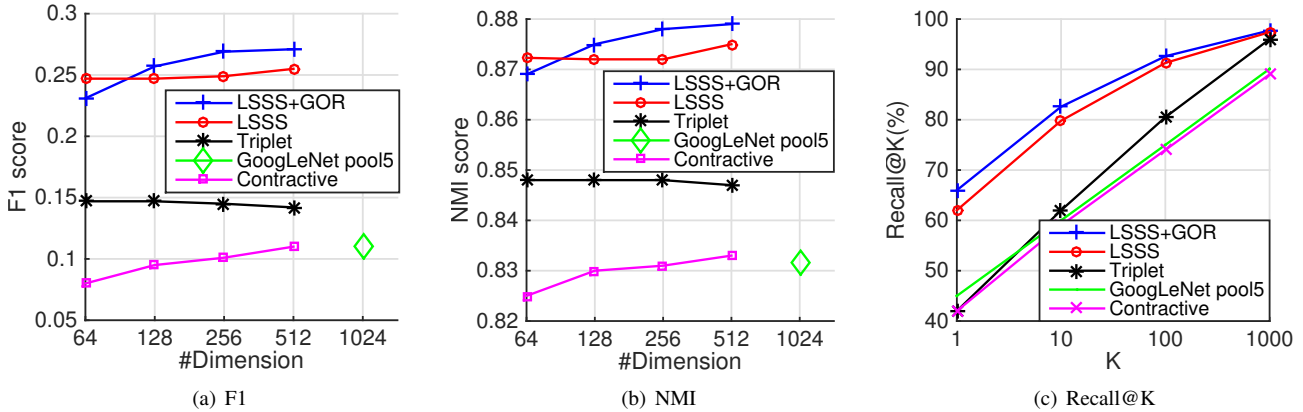
Figure 7. F1, NMI (for clustering) and Recall@K (for retrieval) scores for image-level descriptor learning using Stanford Online Product dataset.

harmonic mean of precision and recall. NMI metric equals to the mutual information divided by the average value of the entropy of clusters and the entropy of labels. For retreival task, the performance is evaluated by Recall@K score as in [14]. For each query image, we first remove the query from the test set and then retrieve its K nearest neighbors from the test set. The recall of the test image is set to 1 if *any* image in the same class with the query is retrieved and 0 otherwise.

6.2. Implementation details

The proposed GOR is embedded with the lifted structured similarity softmax loss (LSSS), which is one of the best performing losses used in learning feature embedding. The network structure follows GoogLeNet [22] up to the "pool5" layer. The final descriptor is generated by a fully connected layer. All the convolutional layers are initialized from the network pre-trained on ImageNet ILSVRC dataset [16]. All convolutional layers are fine-tuned with a learning rate that is 10 times smaller than that of the fully-connected layer. The batch size is set to 128 and the training iteration is set to 20,000.

6.3. Result

Figure 7(a) and Figure 7(b) further show the F1 score and NMI score for the clustering task with different embedding sizes. By combining the proposed GOR with LSSS, our method shows better performance especially in high-dimensional cases ($d \geq 128$). The reason is discussed in Section 5.3. Figure 7(c) shows the Recall@K score for 512 dimensional descriptor.

We also test the proposed regularization on small metric learning datasets such as Car196 and CUB-200-2011. The proposed regularization does not show clear improvement. One possible explanation is that the numbers of the classes in Car196 (196) and CUB-200-2011 (200) are

much smaller than that of UBC ($>$100k) and Stanford online dataset ($>$22k). The assumption of uniform distribution for non-matching samples is not ideal for such situations. To understand this, one can imagine an extreme case of only two classes in a high dimensional space, putting them on opposite positions of the unit sphere (instead of orthogonal) is optimal.

7. Conclusion

We proposed a regularization technique named Global Orthogonal Regularization (GOR) that makes the local feature descriptor more spread-out in the descriptor space. Inspired by the properties of uniform distribution, the regularization achieves the desired property by making the non-matching pairs close to orthogonal. We showed the proposed regularization can be easily used to improve the performance of various feature embedding losses such as the pairwise and triplet losses.

In the future, we plan to extend the proposed regularization technique to non-Euclidean distance. We also plan to apply our method to more general metric learning settings. Our prototype implementation can be downloaded from `https://github.com/ColumbiaDVMM/Spread-out_Local_Feature_Descriptor`.

References

[1] V. Balntas, E. Riba, D. Ponsa, and K. Mikolajczyk. Learning local feature descriptors with triplets and shallow convolutional neural networks. *BMVC*, 2016.

[2] H. Bay, T. Tuytelaars, and L. Van Gool. Surf: Speeded up robust features. In *ECCV*, 2006.

[3] M. Brown, G. Hua, and S. Winder. Discriminative Learning of Local Image Descriptors. *TPAMI*, 2011.

[4] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *CVPR*, 2005.

[5] M. A. *et al*. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015.

[6] X. Han, T. Leung, Y. Jia, R. Sukthankar, and A. C. Berg. MatchNet: Unifying Feature and Metric Learning for Patch-Based Matching. In *CVPR*, 2015.

[7] S. Ioffe and C. Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *ICML*, 2015.

[8] B. Kumar, G. Carneiro, I. Reid, et al. Learning local image descriptors with deep siamese and triplet convolutional networks by minimising global loss functions. In *CVPR*, 2016.

[9] J. Lin, O. Morere, V. Chandrasekhar, A. Veillard, and H. Goh. Deephash: Getting regularization, depth and fine-tuning right. *arXiv preprint arXiv:1501.04711*, 2015.

[10] D. G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *IJCV*, 2004.

[11] C. D. Manning, P. Raghavan, H. Schütze, et al. *Introduction to information retrieval*. 2008.

[12] A. Mishchuk, D. Mishkin, F. Radenovic, and J. Matas. Working hard to know your neighbor's margins: Local descriptor learning loss. *arXiv preprint*, 2017.

[13] Y. Movshovitz-Attias, A. Toshev, T. K. Leung, S. Ioffe, and S. Singh. No Fuss Distance Metric Learning using Proxies. *arXiv:1703.07464 [cs]*, 2017.

[14] H. Oh Song, Y. Xiang, S. Jegelka, and S. Savarese. Deep metric learning via lifted structured feature embedding. In *CVPR*, 2016.

[15] L. Paulevé, H. Jégou, and L. Amsaleg. Locality sensitive hashing: A comparison of hash function types and querying mechanisms. *Pattern Recognition Letters*, 2010.

[16] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 2015.

[17] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, 2015.

[18] M. Schultz and T. Joachims. Learning a distance metric from relative comparisons. In *NIPS*, 2003.

[19] E. Simo-Serra, E. Trulls, L. Ferraz, I. Kokkinos, P. Fua, and F. Moreno-Noguer. Discriminative Learning of Deep Convolutional Feature Point Descriptors. In *CVPR*, 2015.

[20] K. Simonyan, A. Vedaldi, and A. Zisserman. Learning Local Feature Descriptors Using Convex Optimisation. *TPAMI*, 2014.

[21] K. Sohn. Improved Deep Metric Learning with Multi-class N-pair Loss Objective. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *NIPS*. 2016.

[22] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *CVPR*, 2015.

[23] B. F. Y. Tian and F. Wu. L2-net: Deep learning of discriminative patch descriptor in euclidean space. In *CVPR*, 2017.

[24] E. Tola, V. Lepetit, and P. Fua. Daisy: An efficient dense descriptor applied to wide-baseline stereo. *TPAMI*, 2010.

[25] E. Ustinova and V. Lempitsky. Learning Deep Embeddings with Histogram Loss. In *NIPS*. 2016.

[26] K. M. Yi, E. Trulls, V. Lepetit, and P. Fua. LIFT: Learned Invariant Feature Transform. In *ECCV*, 2016.

[27] S. Zagoruyko and N. Komodakis. Learning to compare image patches via convolutional neural networks. In *CVPR*, 2015.