Learning Local Descriptors with a CDF-Based Dynamic Soft Margin

Linguang Zhang Szymon Rusinkiewicz
Princeton University

Abstract

The triplet loss is adopted by a variety of learning tasks, such as local feature descriptor learning. However, its standard formulation with a hard margin only leverages part of the training data in each mini-batch. Moreover, the margin is often empirically chosen or determined through computationally expensive validation, and stays unchanged during the entire training session. In this work, we propose a simple yet effective method to overcome the above limitations. The core idea is to replace the hard margin with a non-parametric soft margin, which is dynamically updated. The major observation is that the difficulty of a triplet can be inferred from the cumulative distribution function of the triplets' signed distances to the decision boundary. We demonstrate through experiments on both real-valued and binary local feature descriptors that our method leads to state-of-the-art performance on popular benchmarks, while eliminating the need to determine the best margin.

1. Introduction

Efficient image matching is a fundamental problem in computer vision, robotics, and graphics. Generally, image matching is a two-step procedure consisting of extracting repeatable local keypoints using an interest point *detector*, followed by matching of *feature descriptors* corresponding to those points. Traditional pipelines use handcrafted descriptors such as SIFT [15], which has proven successful in a variety of applications. With the development of deep-learning techniques, however, recent advancements to feature descriptors have been mainly *learning*-based. These learned descriptors often achieve higher matching performance than handcrafted ones, at the same length (e.g., 128 floats, as with SIFT). Further reduction in storage and matching costs can be achieved by *binary descriptors*, which are interpreted as bit vectors and are compared using Hamming distance instead of Euclidean distance.

In this paper, we demonstrate that a common architecture (based on L2-Net [29]) and training procedure (based on HardNet [19]) can be modified to learn not only floating-point but also binary descriptors. The modified network advances the state of the art in descriptor performance, but it also highlights a frequently-encountered problem: the matching accuracy depends on hyperparameter tuning.

In particular, many well-performing learned descriptors are trained using the same loss function: a *triplet loss* that encourages the distance between a negative pair to exceed the distance between a positive pair by some *margin*. The purpose of the margin is to force the network to update its weights using the gradients computed from "harder" triplets, while excluding training samples classified as "easy" by the margin. While modern approaches improve performance by incorporating hard negative mining [3, 19] or regularization [13, 38], the effectiveness of training fundamentally depends on the setting of the margin. Because the optimal margin is problem- and often dataset-dependent, in practice the margin is either specified by hand based on an educated guess or exhaustively tuned at great computational expense.

In this work, we propose a novel triplet loss function that has three major features: 1) We use a *soft* instead of *hard* margin to fully utilize each mini-batch. 2) The soft margin *dynamically* adapts to the current state of training. 3) The method is *parameter-free*: the two goals above are accomplished without the need for any user-tunable hyperparameters. In short, as opposed to the ***Static Hard Margin*** used in the traditional triplet loss, we think of our method as an instance of a ***Dynamic Soft Margin*** strategy that could be applied to a variety of learning problems.

The traditional triplet loss makes a binary decision of whether a triplet should contribute to the gradient, using the *hard* margin as a constant threshold. We instead make the margin *soft* by using all the triplets in each mini-batch, while following the simple intuition that "difficult" triplets should receive greater weight than "easy" ones. In contrast to previous approaches that use soft margins (such as SVMs with slack variables), our formulation does not require a separate user-tunable parameter for the "softness" of the margin. The dynamic nature of our method is obtained by maintaining a *moving* Probability Density Function (PDF) of the *differ-*

ence of distances between the positive and negative pairs in each triplet. This may be thought of as the signed distance of each triplet to the decision boundary. Weights are assigned based on the integral of this PDF, in essence weighting each datapoint proportionally to the probability that it is more difficult to classify than other recently encountered datapoints. Therefore the weighting function is continuously updated as training progresses. We summarize the major contributions of this paper as follows:

- Proposing a novel loss function based on dynamic soft margin that can serve as a *drop-in replacement* for the existing triplet loss without user-tunable parameters.
- Unifying the real-valued and binary descriptor learning pipelines to adopt the same loss function for training.
- Demonstrating that our approach improves upon the state of the art for ***both*** real-valued and binary descriptor learning.

2. Related Work

There is significant previous work on local features, and we focus on descriptors that are widely used or have recently achieved state-of-the-art performance. These are generally categorized into real-valued and binary descriptors: while most existing works address *either* one or the other, we show that our dynamic soft margin approach improves *both*.

Real-valued Descriptors: Probably the most successful handcrafted feature descriptor is SIFT [15], which computes smoothed histograms using the gradient field of the image patch. Instead of computing the histogram, PCA-SIFT [11] applies principal component analysis directly to the image gradient. Recently, learning-based methods have started to demonstrate effectiveness. Simonyan et al. [27] formulate feature learning as a convex optimization problem. DeepDesc [26] uses paired image patches and adopts a Siamese network to learn a discriminative descriptor while performing hard mining to boost performance. DeepCompare [37] develops a two-stream network with one stream focusing on the central part of the image. TFeat [3] learns the descriptor using a triplet loss and applies an in-triplet hard mining method named anchor swapping. More recently, Tian et al. have proposed L2-Net [29], which adopts a deeper network and designs a new loss function requiring the true matches to have the minimal ℓ_2 distances in the batch. HardNet [19] further simplifies the idea by looking for hard negatives in each batch and achieves state-of-the-art performance using a single triplet margin loss. Instead of using the triplet margin loss as the proxy, DOAP [6] directly optimizes the ranking-based retrieval performance metric and achieves more competitive results using the same network architecture. Keller et al. [12] propose a mixed-context loss that combines triplet and Siamese loss, which performs better than using either one alone. GeoDesc [16] further leverages geometric constraints from multi-view reconstruction and demonstrates significant improvement on 3D reconstruction tasks.

Binary Descriptors: Although real-valued descriptors demonstrate good performance and applicability, they expose challenges to both storage and matching. Popular real-valued descriptors (such as SIFT) and recent learning-based descriptors use 128 floating point numbers, or 512 bytes. While the storage requirement could be aggressively reduced by quantizing the floating point values [32, 35] or applying principal component analysis (PCA) [35, 10] to reduce the length of the descriptor, comparing the shortened real-valued descriptors still requires computing a real-valued Euclidean distance. Efficient handcrafted binary descriptors ameliorate these problems by directly building a binary string using the input image patch. The metric used to evaluate the distance between two binary descriptors is the Hamming distance, which is the number of set bits after performing the XOR operation. Efficient computation of Hamming distance can exploit specialized instructions on supported hardware. Popular binary descriptors include BRIEF [4] and rotated BRIEF used by ORB [24], which typically rely on intensity comparisons using a predefined pattern. This significantly lowers the computation cost, although this also implies that these binary descriptors are less robust against drastic illumination changes. DOAP [6] demonstrates that a good binary descriptor can also be learned by optimizing the average precision on the retrieval task. Some recent real-valued descriptors can be trivially converted to binary descriptors by taking the sign of each dimension, and L2-Net has already shown promising results in generating binary descriptors by doing this. There exist more sophisticated ways to convert floating-point vectors into binary strings, such as LSH [5] or LDAHash [28].

Replacing Static Hard Margin: Instead of setting a hard margin, DeepDesc [26] back-propagates the hardest 1/8 of samples, which can be interpreted as a ***Dynamic Hard*** Margin. Person re-identification is a related topic that also uses the triplet margin. Wang et al. [34] apply two separate hard margins for positive and negative pairs. The margins are dynamically adjusted using a hand-crafted function with cross-validated, but highly unintuitive, hyper-parameters (μ = 8 and γ = 2.1). Their loss function still classifies training samples by making binary decisions, thus is yet another case of Dynamic Hard Margin. Hermans et al. [7] use the softplus function to mimic the soft margin. However, this function is fixed throughout the entire training session and we view it as an instance of ***Static Soft*** Margin. Moreover, softplus has an implicit scale hyper-parameter: $1/\beta \cdot \log(1+\exp(\beta x))$, where β controls smoothness and is 1.0 by default. Section 5.6, includes comparisons against these alternatives to demonstrate the effectiveness of our ***Dynamic Soft*** Margin.

3. Learning Local Descriptors

Our goal is to examine the effectiveness of the proposed Dynamic Soft Margin strategy, and we are motivated by the application of learning real-valued and binary local feature descriptors. As background, we first revisit the original triplet margin loss, together with the network architecture and hard-negative mining method used by state-of-the-art methods such as HardNet [19]. Next, we introduce a modified training procedure that can learn high-quality binary descriptors.

3.1. Real-Valued Descriptors

A Siamese network with two streams sharing the same deep architecture and weights is one of the natural choices for learning a descriptor. Most recent works adopt L2-Net [29] as the backbone network due to its good performance. We also use L2-Net in this work and only replace the loss function to show the effectiveness of our method. L2-Net, denoted as $\mathcal{F}(\cdot)$, takes an image patch x as input and produces a k-dimensional descriptor. Given two input image patches x and x', the distance between them in descriptor space is written as $\mathcal{D}(\mathcal{F}(x), \mathcal{F}(x'))$, where $\mathcal{D}$ is a distance metric. For real-valued descriptors, the Euclidean distance is often used as the distance metric. While evaluating the Euclidean distances among a large number of descriptors can be expensive, the computational cost can be reduced if the feature vectors are unit-length (i.e., $\|\mathcal{F}(x)\| = 1$). Since the Euclidean distance between two unit vectors can be computed using the dot product:

$$\mathcal{D}_{\text{Euclidean}}\big(\mathcal{F}(x), \mathcal{F}(x')\big) = \sqrt{2 - 2\,\mathcal{F}(x)^T \mathcal{F}(x')}, \quad (1)$$

it becomes possible to compute the pair-wise distance matrix of a batch of descriptors with a single matrix multiplication. To leverage this nice property, L2-Net normalizes the network output into a unit-length descriptor. Given N pairs of matching image patches, where each pair corresponds to a unique 3D point in the physical world, HardNet [19] computes a pair-wise distance matrix from the descriptors output by the Siamese network. The diagonal elements in the distance matrix correspond to the distances between matching pairs. HardNet mines the hardest negative for each matching pair in its row and column from the non-diagonal elements in the distance matrix. Note that this mining is a different process from the weighting of triplets by our soft margin, as described below. For more details, we refer the readers to the original paper.

If we denote the distance between a matching pair as d_{pos} and the distance between the corresponding hardest non-matching pair as d_{neg}, HardNet trains using the standard triplet margin loss with margin $\mu = 1.0$:

$$\mathcal{L}_{\text{triplet}} = \max(0, \mu + d_{\text{pos}} - d_{\text{neg}}). \quad (2)$$

The triplet margin loss simply forces the network to learn to increase the difference between d_{neg} and d_{pos} until the condition $d_{\text{neg}} - d_{\text{pos}} > \mu$ is satisfied. If μ is set sufficiently large that $d_{\text{neg}} - d_{\text{pos}} > \mu$ is never satisfied, $\frac{\partial \mathcal{L}_{\text{triplet}}}{\partial d_{\text{pos}}} = 1$ and $\frac{\partial \mathcal{L}_{\text{triplet}}}{\partial d_{\text{neg}}} = -1$, where μ is no longer relevant. Though $\mu = 1.0$ is shown to perform well for HardNet, it remains a question whether there exists a μ that works better.

3.2. Binary Descriptors

Real-valued descriptors are potentially costly to store and compute with, due to the fact that they are usually represented using 32-bit floating-point numbers. On the other hand, binary descriptors are both more compact to store and faster to compare (using Hamming distance), and hence are popular in real-time applications. Unfortunately, HardNet [19] *only* addresses real-valued descriptor learning. Below, we propose how to adapt it for binary descriptor learning, which is inherently non-differentiable.

At testing time, it is easy to convert a real-valued L2-Net output to a binary descriptor: we simply use the sign function to convert the output to a length-k vector of the values -1 and 1. This reduces the Hamming distance between two descriptors to a dot product:

$$\mathcal{D}_{\text{Hamming}}\big(\mathcal{F}(x), \mathcal{F}(x')\big) = \big(k - \mathcal{F}(x)^T \mathcal{F}(x')\big) \,/\, 2. \quad (3)$$

Therefore, as with real-valued descriptors, the pair-wise Hamming distance matrix of a list of binary descriptors can be computed using a single matrix multiplication.

The gradient of the sign function, however, is undefined at the origin and zero everywhere else. So, we need a differentiable proxy for training purposes. Therefore instead of normalizing the output of L2-Net, we use a hyperbolic tangent function (tanh) to compress the output of each element into the range $(-1, 1)$. During training, we use tanh whenever we need the Hamming distance to be differentiable, and use sign in other cases that require full binarization. For instance, the descriptors are fully binarized before computing the distance matrix, because we need to mine the hard negatives as if the current batch were being tested.

Given the mined hard negatives and the distance metric, we can still use a triplet loss to learn the binary descriptor. Selecting an optimal margin becomes even more challenging this time: the difference between d_{neg} and d_{pos} can be as large as the maximum Hamming distance, which is the descriptor length k. It is difficult to determine a proper margin without running a few training/validation sessions. In fact, we have determined that for $k = 256$ the optimal margin is the unintuitive value $\mu = 32$.

4. Dynamic Soft Margin

In this section, we discuss how we replace the triplet margin loss used by state-of-the-art descriptor learning methods

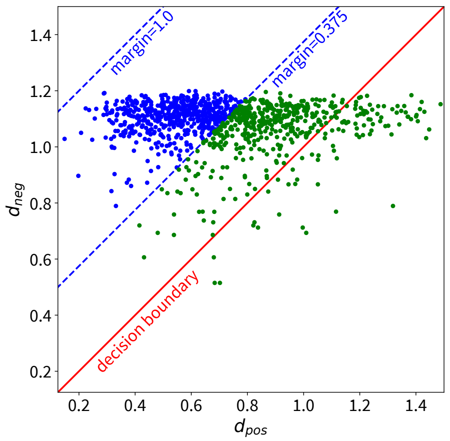

Figure 1. Scatter plot of $(d_{\text{pos}}, d_{\text{neg}})$ for triplets in one batch (1024 samples), using standard triplet margin loss. The red line is the decision boundary: d_{neg} is correctly greater than d_{pos} to its upper-left. The blue dotted lines are potential margins. The (optimal) margin of 0.375 separates points into two clusters (green and blue), where the blue cluster is considered "good enough" and is not used for back-propagation.

with our dynamic-weighted triplet loss. We first analyze the training behavior of the triplet margin loss and explain with an example why choosing a good margin is important for optimal performance. Then, we explain how our method eliminates the margin while improving performance.

4.1. Behavior of the Triplet Margin Loss

Let us first analyze how the triplet margin loss behaves when learning a real-valued descriptor. For a unit-length real-valued descriptor, the largest difference between d_{neg} and d_{pos} is 2.0 ($d_{\text{neg}} = 2.0$ and $d_{\text{pos}} = 0$). Setting the margin larger than or equal to 2.0 simply transforms the triplet margin loss into a basic triplet loss (i.e., without margin), where no triplet is affected by truncation. In practice, a margin smaller than 2.0 is expected to improve performance, since the "easiest" triplets, having larger $d_{\text{neg}} - d_{\text{pos}}$, are not involved in back-propagation. To better understand this behavior, we take a pretrained HardNet model (trained on UBC PhotoTourism - *Liberty*) and visualize all triplets in a sample batch of data by plotting $(d_{\text{pos}}, d_{\text{neg}})$ as scattered points in 2D (Figure 1).

In the case of a perfect descriptor, d_{pos} is expected to be smaller than d_{neg}. This corresponds to the region to the upper-left of the decision boundary shown in red. A straightforward approach to optimize $\mathcal{F}(\cdot)$ is to maximize the signed distance from the point $(d_{\text{pos}}, d_{\text{neg}})$ to the decision boundary, which is *equivalent* to minimizing the basic triplet loss: $\mathcal{L}_{\text{triplet}} = d_{\text{pos}} - d_{\text{neg}}$. The triplet margin loss sets a margin so that points that are sufficiently far on the correct side of the decision boundary (shown in blue, for a hypothetical margin $\mu = 0.375$) are excluded from optimization. By doing this, we force the network to focus on harder triplets (near to, or on the wrong side of, the decision boundary) without having the gradients influenced by the easy triplets. However, Figure 1 shows that a margin of 1.0, as recommended by the HardNet paper, would have almost no effect because all triplets are still within the margin.

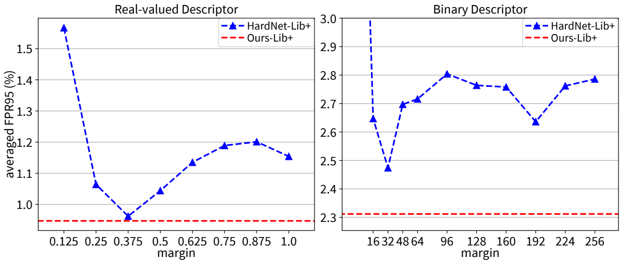

Figure 2. Varying the margin used by the triplet margin loss. The network is trained on the *Liberty* subset of UBC PhotoTourism and evaluated on the other two subsets. The left and right figures show the performance on the real-valued and binary descriptors, respectively. For our results, we keep all other configurations used by the triplet margin loss and only replace the loss function.

To investigate whether there is a "sweet-spot" for the margin, we vary it across a wide range and re-train the real-valued descriptor (always training on *Liberty* and evaluating the false positive rate at 95 percent recall – FPR95 – on the other two subsets of the UBC PhotoTourism dataset [36]). As demonstrated by the blue curve of Figure 2, left, there indeed exists a better choice of margin. As shown in Figure 1, $\mu = 0.375$ excludes a substantial number of easy triplets and lets the network focus on the harder cases. Figure 2, right, shows the corresponding graph for our new binary descriptor. Note that the shapes of the curves are different, and the optimal margin is problem-dependent. Of course, it is possible that even better margins could be found at greater computational cost by increasing the precision of the search.

We conclude that finding the best margin is a non-trivial job in practice and may require extensive validation. In the following section, we introduce our dynamic triplet weighting method, which avoids setting a hard threshold and weights the triplets in a mini-batch based on the training status of the network. Results produced by our approach are displayed as the red dotted lines in Figure 2.

4.2. Dynamic Triplet Weighting

Our approach shares the same motivation as the triplet margin loss, in that the "harder" triplets in a mini-batch are more useful for training. In other words, "easy" triplets should be suppressed in the loss function because the network's performance on these triplets is already likely to be saturated. The concept of emphasizing harder training examples is also the major reason why hard negative mining has become recognized as vital to good performance in recent learned descriptors.

The key observation of this work is that we can directly measure how hard a triplet is compared to other triplets in

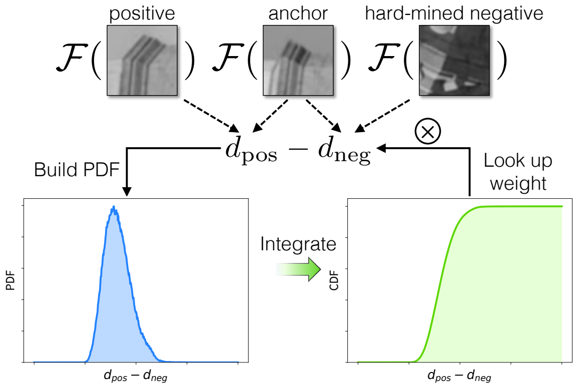

Figure 3. Our scheme for Dynamic Triplet Weighting. We compute $d_{\text{pos}} - d_{\text{neg}}$ for each hard-mined triplet, then build a moving histogram (PDF) of these values and integrate to obtain the CDF. The loss for each triplet is $d_{\text{pos}} - d_{\text{neg}}$, weighted by the corresponding value from the CDF.

the same mini-batch, by seeing how its signed distance to the decision boundary ($d_{\text{pos}} - d_{\text{neg}}$) compares to the distribution of these distances. To measure this, we would like to know the Probability Distribution Function (PDF) of signed distances, which in practice we discretize into a histogram. To make the aggregated histogram more accurate, we compute $d_{\text{pos}} - d_{\text{neg}}$ for each triplet, and then linearly allocate it into two neighboring bins in the histogram. In our implementation, since a temporally stable PDF is preferred, we maintain it as an exponentially-decaying moving histogram (with a weight of 0.1 on each new batch), similarly to other neural network modules that utilize moving averages (e.g., batch normalization [8]). An example of the PDF is shown in Figure 3, bottom left.

Given the distribution of difficulty in recent batches, the relative difficulty of a *particular* triplet corresponds to the fraction of triplets that have a lower $d_{\text{pos}} - d_{\text{neg}}$. This is just the integral of the PDF, or the Cumulative Distribution Function (CDF), shown in Figure 3, bottom right. The hardest triplet in a mini-batch results in a CDF of 1.0, while the easiest triplet corresponds to ≈ 0. More generally, a triplet with a CDF value of $k\%$ means that it is empirically “harder” than $k\%$ of triplets within recent batches.

Because these CDF values have an intuitive interpretation as difficulty, we use them directly as weights. Given a mini-batch of size N, we define our weighted triplet loss (without a hard margin) as:

$$\mathcal{L} = \frac{1}{N}\sum_i w_i \cdot (d^i_{\text{pos}} - d^i_{\text{neg}}), \tag{4}$$

$$w_i = \text{CDF}(d^i_{\text{pos}} - d^i_{\text{neg}}). \tag{5}$$

This loss function automatically rejects “easy” triplets by assigning them low weights. One may wonder how the loss function behaves when the variance of $d_{\text{pos}} - d_{\text{neg}}$ is very small, so that the CDF is close to a step function. In fact, when such a case happens, the loss function weights all triplets nearly equally and the optimization continues. This is not always possible for the original triplet margin loss because the optimization would stop when every triplet satisfies $d_{\text{neg}} - d_{\text{pos}} > \mu$ (though this scenario rarely happens in practice). The red lines in Figure 2 show that our method consistently leads to better performance than the triplet margin loss on both the real-valued and binary descriptors.

5. Experiments

We have experimented with three benchmarks: UBC PhotoTourism [36], HPatches [2], and the Oxford Affine benchmark [18]. UBC PhotoTourism is a classic patch-based dataset that is mainly evaluated on the patch verification task, which can be quickly computed and is effective for preliminary analysis of the descriptor performance. The patch verification task is often not sufficient for estimating the performance of a descriptor in practical applications where patch retrieval is a more important task. HPatches is a more comprehensive benchmark that contains a much larger collection of image patches and evaluates a descriptor on three different tasks: patch verification, image matching, and patch retrieval. The Oxford Affine benchmark contains image sequences with different types of distortion, which is useful for understanding the robustness of a descriptor when the input images are less than ideal.

5.1. Implementation

We adopt a training configuration as similar as possible to that used by previous work, to ensure that our new loss function is the major factor in the final results. For training, we use the UBC PhotoTourism dataset [36]. Each of its three subsets, known as *Liberty*, *Yosemite*, and *Notre Dame*, consists of more than 400k image patches, cropped to 64×64 and re-oriented using Difference-of-Gaussians (DoG) keypoints [15]. We train one model using each subset and test on the other two subsets. We downsample each patch to a 32×32 input, which is required by L2-Net. Each patch is then normalized by subtracting the mean pixel value and dividing by the standard deviation. Online data augmentation is achieved by random flipping and rotating the patch by 90, 180 or 270 degrees. The UBC PhotoTourism dataset assigns each patch with its 3D point ID, which is used to identify matching image patches. Each 3D point ID is associated with a list of patches that are assumed to be matching. To form a mini-batch of size N for training, we randomly select N 3D points without replacement and select two patches for each chosen 3D point.

We use Stochastic Gradient Descent (SGD), with momentum and weight decay equal to 0.9 and 10^{-4}, respectively, to optimize the network. Inspired by HardNet and DOAP, the network is trained for 50k iterations, with the learning rate linearly decaying from 0.1 to 0. The batch

Table 1. Evaluation on the UBC PhotoTourism dataset, demonstrating that both real-valued and binary descriptors trained using our method outperform the state of the art. Numbers shown are FPR95(%) – lower is better. "+" and "*" denote training with data augmentation and anchor swapping [3]. DOAP-ST+ represents the DOAP descriptor with a Spatial Transformer [9] to compensate for geometric noise.

Descriptor	Length	Train →	Notredame	Yosemite	Liberty	Yosemite	Liberty	Notredame	Mean
		Test →	Liberty		Notredame		Yosemite		
			Real-valued Descriptors						
SIFT [15]	128		29.84		22.53		27.29		26.55
DeepDesc [26]	128		10.9		4.40		5.69		7.0
TFeat-M* [3]	128		7.39	10.31	3.06	3.80	8.06	7.24	6.64
TL+GOR* [38]	128		4.80	6.45	1.95	2.38	5.40	5.15	4.36
PCW [20]	128		7.44	9.84	3.48	3.54	5.02	6.56	5.98
L2-Net+ [29]	128		2.36	4.70	0.72	1.29	2.57	1.71	2.23
CS-L2-Net+ [29]	256		1.71	3.87	0.56	1.09	2.07	1.30	1.76
HardNet+ [19]	128		1.49	2.51	0.53	0.78	1.96	1.84	1.52
DOAP+ [6]	128		1.54	2.62	0.43	0.87	2.00	**1.21**	1.45
DOAP-ST+ [6]	128		1.47	2.29	**0.39**	0.78	1.98	1.35	1.38
Ours+	128		**1.21**	**2.01**	**0.39**	**0.68**	**1.51**	1.29	**1.18**
			Binary Descriptors						
ORB [24]	256		59.15		54.57		54.96		56.23
BinBoost [31]	64		20.49	21.67	16.90	14.54	22.88	18.97	19.24
LDAHash [28]	128		49.66		51.58		52.95		51.40
DeepBit [14]	256		32.06	34.41	26.66	29.60	57.61	63.68	40.67
L2-Net+ [29]	128		7.44	10.29	3.81	4.31	8.81	7.45	7.02
CS-L2-Net+ [29]	256		4.01	6.65	1.90	2.51	5.61	4.04	4.12
DOAP+ [6]	256		3.18	4.32	1.04	1.57	4.10	3.87	3.01
DOAP-ST+ [6]	256		2.87	4.17	0.96	1.76	3.93	3.64	2.89
Ours+	256		**2.70**	**4.01**	**0.93**	**1.44**	**3.69**	**2.98**	**2.63**

size is set to 1024 for all experiments to match the publicly available implementations of HardNet and DOAP. To facilitate future research, we package our implementation as a standalone PyTorch [21] module.

5.2. UBC PhotoTourism

Each of the UBC PhotoTourism subsets includes a test split containing 100k pairs of image patches, with half of them being true matches and the rest being false matches. We adopt the commonly used false positive rate at 95% true positive recall (FPR95) to evaluate how well the proposed descriptor classifies patch pairs. We compare with a collection of existing real-valued descriptors including both handcrafted (SIFT [15] and root-SIFT [1]) and learned (DeepDesc [26], TFeat [3], GOR [38] PCW [20], L2-Net [29], HardNet [19], DOAP [6]). GeoDesc [16] is not evaluated because it is trained on a custom dataset. We also compare against existing binary descriptors including ORB [24], BinBoost [31], LDAHash [28], DeepBit [14], L2-Net [29], and DOAP [6]. The results are shown in Table 1. Our approach outperforms all existing methods under the same configuration. DOAP-ST+ uses a larger input (42×42) to augment DOAP+ with the Spatial Transformer [9], which noticeably improves the performance by correcting geometric noise. Note that our method also surpasses DOAP-ST+ in most cases even without the Spatial Transformer. Compared to HardNet, our method automatically produces better performance that otherwise would have required fine-tuning the margin, or even manually adjusting the margin at different stages of training.

5.3. HPatches

The recently-introduced HPatches benchmark of Balntas et al. [2] evaluates descriptors in a more sophisticated setting. Different amounts of geometric noise are introduced into the test image patches, which are then categorized as "Easy", "Hard" or "Tough". HPatches evaluates a descriptor on three different tasks: patch verification, image matching, and patch retrieval. For a more detailed description of the tasks, we refer the readers to their paper.

Figure 4 compares descriptors trained with the proposed method and top-performing real-valued and binary descriptors. As is common practice, learned descriptors are evaluated using a model trained on the *Liberty* subset of the UBC PhotoTourism dataset, with data augmentation. For all descriptors, we do not apply the ZCA normalization that is originally used in HPatches. HardNet does not come with a binary version and we simply take the sign to obtain HardNet-b+. It is not surprising to see that both the real-valued and binary descriptors learned using our loss func-

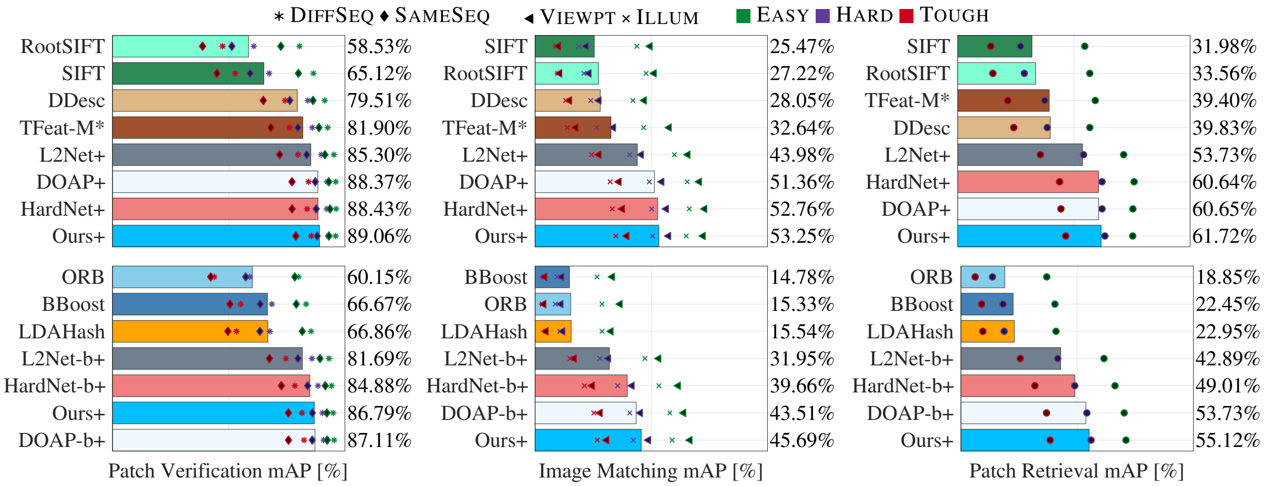

Figure 4. Evaluation on the HPatches dataset [2]. The evaluation is carried out on the "full" split of HPatches. The patch retrieval task is evaluated with the maximum amount of distractors (same setting used in the original HPatches paper). Top row: real-valued descriptor comparison. Bottom row: binary descriptor comparison. While both HardNet and DOAP perform well in easy cases, our descriptor is more robust in tough cases, leading to state-of-the-art performance overall.

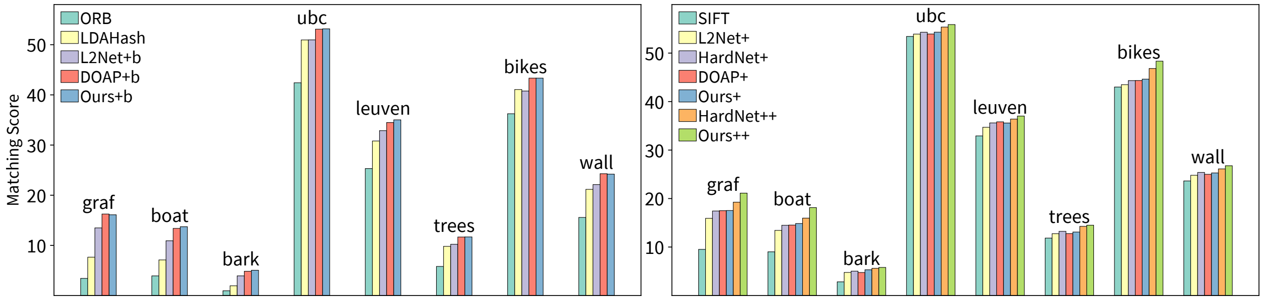

Figure 5. Evaluation on the Oxford Affine dataset, for binary (left) and real-valued (right) descriptors. All are trained on the UBC *Liberty* subset with data augmentation, except the models suffixed with "++", which are trained on the union of UBC PhotoTourism and HPatches.

tion perform well on the patch verification task, which is consistent with our observation on the UBC PhotoTourism dataset. On the more challenging image matching and patch retrieval tasks that require the descriptor to be more distinctive, our descriptors outperform all existing methods.

5.4. Image Matching on the Oxford Dataset

In real image matching scenarios, images may undergo diverse distortions including geometric transformations, blurring, illumination changes, and JPEG compression. In order to verify whether the descriptors learned with our method are vulnerable to a particular type of distortion, we further evaluate the image matching performance using the Oxford Affine Dataset [18], which contains all the above-mentioned transformations. In this dataset, homography matrices are provided to help verify correspondences. We choose the Harris-Affine detector [17] to extract keypoints from the images and crop image patches using a magnification factor of 6. We strictly follow the public evaluation protocol [18]. The matching scores are reported in Figure 5. The result shows that our descriptors can withstand various type of distortions presented in this dataset and achieve state-of-the-art results. Also note that our descriptor trained with the union of UBC PhotoTourism and HPatches outperforms HardNet trained with the same data.

Table 2. Evaluation on two image retrieval engines — VisualIndex and Hamming Query Expansion (HQE). SA: single assignment. MA: multiple assignments. +: model is trained on the UBC-*Liberty* subset with data augmentation. ++: model is trained with the union of UBC PhotoTourism and HPatches datasets.

Method	Oxford5k			Paris6k		
	Visual Index	HQE (SA)	HQE (MA)	Visual Index	HQE (SA)	HQE (MA)
RootSIFT	67.19	80.99	83.60	66.40	79.32	80.36
HardNet+	73.07	84.58	85.42	67.82	87.34	88.35
HardNet++	71.01	84.57	85.44	68.96	87.19	88.45
Ours+	72.92	84.29	85.94	67.63	87.83	88.54
Ours++	**73.80**	**85.43**	**86.42**	**71.66**	**88.52**	**89.24**

5.5. Image Retrieval

Local feature descriptors are also often deployed for image retrieval. We have evaluated our method on two image retrieval engines: VisualIndex [33] and Hamming Query

Table 3. Comparing existing alternatives to *Static Hard Margin* with our *Dynamic Soft Margin*.

Method	Dynamic	Soft	UBC	HPatches (mAP %)		
			FPR95	Verification	Matching	Retrieval
Real-valued Descriptors						
softplus [7]	✗	✓	1.20	89.00	52.84	60.72
Wang [34]	✓	✗	1.18	88.88	52.63	60.36
Hardest 1/8 [26]	✓	✗	2.19	85.82	46.74	56.51
Ours	✓	✓	**0.95**	**89.06**	**53.25**	**61.72**
Binary Descriptors						
softplus [7]	✗	✓	2.73	86.35	45.45	54.23
Wang [34]	✓	✗	2.76	86.37	45.60	54.19
Hardest 1/4 [26]	✓	✗	3.09	85.58	43.24	52.97
Ours	✓	✓	**2.31**	**86.79**	**45.69**	**55.12**

Table 4. Comparing different ways to construct the PDF.

PDF built from	UBC	HPatches (mAP %)		
	FPR95 (%)	Verification	Matching	Retrieval
Real-valued Descriptors				
d_{pos}	0.98	**89.10**	52.93	61.37
d_{neg}	1.20	88.55	53.15	60.36
Gaussian	1.07	89.05	52.95	61.38
$d_{\text{pos}} - d_{\text{neg}}$	**0.95**	89.06	**53.25**	**61.72**
Binary Descriptors				
d_{pos}	**2.30**	**86.93**	**45.81**	**55.19**
d_{neg}	3.02	85.58	45.21	53.27
Gaussian	2.33	86.72	45.68	54.86
$d_{\text{pos}} - d_{\text{neg}}$	2.31	86.79	45.69	55.12

Expansion [30], with default settings. As is common practice, we use the Oxford5k [22] and Paris6k [23] datasets for evaluation. The vocabulary is learned independently — when evaluating on the Oxford5k dataset, the vocabulary is learned with descriptors extracted from the Paris6k dataset, and vice versa. In Table 2, we report the mean average precision (mAP). Note that our method trained using all data (Ours++) consistently outperforms the HardNet counterpart.

5.6. Ablation Studies

To be consistent, we again use the models trained on the *Liberty* subset of UBC PhotoTourism in the following experiments. FPR95 is evaluated on the other two subsets.

Existing Alternatives to Static Hard Margin: In Section 1, we have discussed three previous attempts to replace the undesired static hard margin. Recall that none of them is both *dynamic* and *soft* like ours. Since these baselines are either originally proposed in a different context (e.g., person re-identification [34, 7]) or with a different learning scheme (e.g., contrastive loss with two-stage training [26]), we re-implement and adapt these methods into our pipeline to ensure a fair comparison. Table 3 shows the results on both UBC PhotoTourism and HPatches. Our dynamic-soft strategy outperforms all baseline methods. Training the binary descriptor with the hardest 1/8 of triplets [26] could not converge in our experiment, likely because 1/8 is too selective, hence we use 1/4 instead.

Different Ways to Construct the PDF: In the approach described above, we weighted samples based on a moving PDF of $d_{\text{pos}} - d_{\text{neg}}$, mainly because it is well-correlated with how "hard" a triplet is. In a more general context, we believe that any variable that effectively reflects the "hardness" of a triplet can be used to build the PDF. For example, we observe from Figure 1 that the variation of the visualized points mainly happens along the d_{pos} axis, whereas the variation along the d_{neg} axis is smaller. This implies that instead of maintaining the moving PDF of $d_{\text{pos}} - d_{\text{neg}}$, using a PDF of d_{pos} could also work well, while we would expect a PDF of d_{neg} to be less effective. We also observe that Figure 3 suggests that the PDF is approximately Gaussian, and might be summarized by its mean and variance. This indicates that we can potentially use a parametric PDF to replace the histogram representation that we currently use and save extra memory and computation. We have therefore explored simply maintaining a running mean and variance of $d_{\text{pos}} - d_{\text{neg}}$, and then weighting triplets based on the *analytic* Gaussian CDF, which may be computed in terms of the standard erf function. The comparisons are shown in Table 4, from which we observe that building the PDF from d_{neg} indeed leads to the worst performance. As expected, d_{pos} is as good an indicator as $d_{\text{pos}} - d_{\text{neg}}$, suggesting that positive samples with larger d_{pos} are more useful for training: this may be thought of as "hard *positive* mining." Approximating the PDF with a simple Gaussian distribution is also feasible, but with a small sacrifice in performance, suggesting that the actual distribution is non-Gaussian.

6. Conclusion

In this work, we observe that the traditional approach of manually setting a margin for triplet loss usually leads to sub-optimal results. The "hard" nature of the margin uses samples inefficiently, while keeping the margin constant ignores the improvement in the network during training.

We instead propose a "dynamic soft margin" strategy that automatically assigns lower weights to datapoints that are too "easy" for improving the network, at each stage of training. The key insight is that the relative "hardness" of a triplet can be inferred from the moving PDF of the difference of distances. Using the CDF computed from this PDF as a weight causes the network focus on harder triplets. Our method can be applied to both real-valued and binary descriptor learning, leading to state-of-the-art performance. Future work includes generalizing the proposed method to other similar domains where empirical margins are being used. For instance, the field of face verification and recognition [25, 7, 39] also leverages the triplet loss, and could potentially benefit from our approach.

References

[1] Relja Arandjelović and Andrew Zisserman. Three things everyone should know to improve object retrieval. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2911–2918, 2012. 6

[2] Vassileios Balntas, Karel Lenc, Andrea Vedaldi, and Krystian Mikolajczyk. HPatches: A benchmark and evaluation of handcrafted and learned local descriptors. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 5, 6, 7

[3] Vassileios Balntas, Edgar Riba, Daniel Ponsa, and Krystian Mikolajczyk. Learning local feature descriptors with triplets and shallow convolutional neural networks. In *BMVC*, pages 119.1–119.11, 2016. 1, 2, 6

[4] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. BRIEF: Binary Robust Independent Elementary Features. In *European Conference on Computer Vision (ECCV)*, pages 778–792, 2010. 2

[5] Aristides Gionis, Piotr Indyk, and Rajeev Motwani. Similarity search in high dimensions via hashing. In *VLDB*, pages 518–529, 1999. 2

[6] Kun He, Yan Lu, and Stan Sclaroff. Local descriptors optimized for average precision. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 596–605, 2018. 2, 6

[7] Alexander Hermans, Lucas Beyer, and Bastian Leibe. In defense of the triplet loss for person re-identification. *arXiv preprint arXiv:1703.07737*, 2017. 2, 8

[8] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015. 5

[9] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial transformer networks. In *Advances in Neural Information Processing Systems*, pages 2017–2025, 2015. 6

[10] Hervé Jégou, Matthijs Douze, Cordelia Schmid, and Patrick Pérez. Aggregating local descriptors into a compact image representation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3304–3311, 2010. 2

[11] Yan Ke and Rahul Sukthankar. PCA-SIFT: A more distinctive representation for local image descriptors. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2004. 2

[12] Michel Keller, Zetao Chen, Fabiola Maffra, Patrik Schmuck, and Margarita Chli. Learning deep descriptors with scale-aware triplet networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[13] 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 *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5385–5394, 2016. 1

[14] Kevin Lin, Jiwen Lu, Chu-Song Chen, and Jie Zhou. Learning compact binary descriptors with unsupervised deep neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1183–1192, 2016. 6

[15] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision (IJCV)*, 60(2):91–110, 2004. 1, 2, 5, 6

[16] 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 *European Conference on Computer Vision (ECCV)*, 2018. 2, 6

[17] Krystian Mikolajczyk and Cordelia Schmid. An affine invariant interest point detector. In *European Conference on Computer Vision (ECCV)*, pages 128–142, 2002. 7

[18] Krystian Mikolajczyk and Cordelia Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 27(10):1615–1630, 2005. 5, 7

[19] Anastasiya Mishchuk, Dmytro Mishkin, Filip Radenović, and Jiři Matas. Working hard to know your neighbor's margins: Local descriptor learning loss. In *Advances in Neural Information Processing Systems*, pages 4829–4840, 2017. 1, 2, 3, 6

[20] Arun Mukundan, Giorgos Tolias, and Ondřej Chum. Multiple-kernel local-patch descriptor. *arXiv preprint arXiv:1707.07825*, 2017. 6

[21] 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 *Advances in Neural Information Processing Systems*, 2017. 6

[22] James Philbin, Ondřej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, 2007. 8

[23] James Philbin, Ondřej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, 2008. 8

[24] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. ORB: An efficient alternative to SIFT or SURF. In *IEEE International Conference on Computer Vision (ICCV)*, pages 2564–2571, 2011. 2, 6

[25] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 815–823, 2015. 8

[26] Edgar Simo-Serra, Eduard Trulls, Luis Ferraz, Iasonas Kokkinos, Pascal Fua, and Francesc Moreno-Noguer. Discriminative learning of deep convolutional feature point descriptors. In *IEEE International Conference on Computer Vision (ICCV)*, pages 118–126, 2015. 2, 6, 8

[27] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Learning local feature descriptors using convex optimisation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 36(8):1573–1585, 2014. 2

[28] Christoph Strecha, Alex Bronstein, Michael Bronstein, and Pascal Fua. LDAHash: Improved matching with smaller descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 34(1):66–78, 2012. 2, 6

[29] Yurun Tian, Bin Fan, and Fuchao Wu. L2-Net: Deep learning of discriminative patch descriptor in Euclidean space. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1, 2, 3, 6

[30] Giorgos Tolias and Hervé Jégou. Visual query expansion with or without geometry: Refining local descriptors by feature aggregation. *Pattern Recognition*, 47(10):3466–3476, 2014. 8

[31] Tomasz Trzcinski, Mario Christoudias, Pascal Fua, and Vincent Lepetit. Boosting binary keypoint descriptors. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2874–2881, 2013. 6

[32] Tinne Tuytelaars and Cordelia Schmid. Vector quantizing feature space with a regular lattice. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1–8, 2007. 2

[33] Andrea Vedaldi. Visualindex – A simple image indexing engine in MATLAB. `https://github.com/vedaldi/visualindex`. Accessed: 2019-06-23. 7

[34] Jiayun Wang, Sanping Zhou, Jinjun Wang, and Qiqi Hou. Deep ranking model by large adaptive margin learning for person re-identification. *Pattern Recognition*, 74:241–252, 2018. 2, 8

[35] Simon Winder, Gang Hua, and Matthew Brown. Picking the best DAISY. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 178–185, 2009. 2

[36] Simon AJ Winder and Matthew Brown. Learning local image descriptors. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, 2007. 4, 5

[37] Sergey Zagoruyko and Nikos Komodakis. Learning to compare image patches via convolutional neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4353–4361, 2015. 2

[38] Xu Zhang, Felix X. Yu, Sanjiv Kumar, and Shih-Fu Chang. Learning spread-out local feature descriptors. In *IEEE International Conference on Computer Vision (ICCV)*, 2017. 1, 6

[39] Liming Zhao, Xi Li, Yueting Zhuang, and Jingdong Wang. Deeply-learned part-aligned representations for person re-identification. In *IEEE International Conference on Computer Vision (ICCV)*, pages 3239–3248, 2017. 8