ADASAMPLE: ADAPTIVE SAMPLING OF HARD POSITIVES FOR DESCRIPTOR LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Triplet loss is commonly used in descriptor learning, where the performance heavily relies on mining triplets. Typical solution to that is first picking pairs of intra-class patches (positives) from the dataset to form batches, and then selecting in-batch negatives to construct triplets. For high-informativeness triplet collection, researchers mainly focus on mining hard negatives in the second stage, while they pay relatively less attention to constructing informative batches, *i.e.*, matching pairs are often randomly sampled from the dataset. To address this issue, we propose *AdaSample*, an adaptive and online batch sampler. Specifically, we sample positives based on their *informativeness*, and formulate our hardness-aware positive mining pipeline within a novel *maximum loss minimization* training protocol. The efficacy of the proposed method is demonstrated in several standard benchmarks, in which it results in a significant and consistent performance gain on top of the existing strong baselines. The source code will be released upon acceptance.

1 INTRODUCTION

Feature descriptors are widely used for finding correspondences (Bian et al., 2017; Yi et al., 2018) across images, and hence allow for many computer vision applications, including structure-from-motion (Agarwal et al., 2009), image retrieval (Philbin et al., 2010), and panorama stitching (Brown & Lowe, 2007). Despite rich efforts made on that, the representation power of traditional descriptors, like SIFT (Lowe, 2004), is limited. With the rapid development of deep learning, recent works (Han et al., 2015; Zagoruyko & Komodakis, 2015; Yi et al., 2016; Tian et al., 2017) show that CNN-based descriptors achieve better performance in terms of patch classification (Brown et al., 2011). Especially, HardNet (Mishchuk et al., 2017) demonstrates the generalization ability of deeply learned descriptors in multiple vision tasks.

Learning descriptors from a large image patch collection is challenging. Other than network design and optimization, sampling informative pairs or triplets of patches is also critical. Mishchuk et al. (2017) leverage the hardest-in-batch solution to construct high-quality triplets, making significant progress on the performance and generalization ability. This demonstrates the importance of triplet sampling in descriptor learning. However, current sampling strategies focus on sampling hard triplets in batch level, while batches are randomly constructed from the dataset. We propose to construct high-informativeness batches in a more principled manner to improve the performance.

The proposed method is nominated as *AdaSample*, which adaptively collects informative positive pairs from the dataset for batch construction. The methodology is developed based on informativeness analysis, where we define *informativeness* of potential samples and estimate their optimal sampling probabilities. Moreover, we propose a novel training protocol inspired by *maximum loss minimization* (Shalev-Shwartz & Wexler, 2016) to boost the generalization ability of the descriptor network. We make comprehensive evaluations and ablation studies of the proposed approach in several standard benchmarks, in which the results clearly demonstrate the superiority of our method.

2 RELATED WORK

Local Descriptor Learning. Traditional handcrafted descriptors, *e.g.* SIFT (Lowe, 2004), extract low-level texture information from image patches. Due to the low representation ability, recent research has gradually been shifted to the learned descriptors. Han et al. (2015) propose a two-stage Siamese architecture to extract features and measure patch similarity, which significantly improves

the performance and demonstrates the great potential of CNNs in descriptor learning. In (Zagoruyko & Komodakis, 2015), another Siamese network is introduced to explore different network architectures with a central-surround structure to boost the performance. Simo-Serra et al. (2015) mine hard negatives and use a shallow architecture for the pairwise similarity.

Balntas et al. (2016) use a triplet-based architecture and show its superiority against the pair-based one. Kumar et al. (2016) use a triplet network and a global loss function to separate the distribution of matching and non-matching pairs. L2-Net (Tian et al., 2017) uses n matching pairs in each batch to generate $n(n-1)$ negative samples and requires that the distance of matching pairs is minimum in each row and column. HardNet (Mishchuk et al., 2017) mines the hardest-in-batch negatives and significantly improves performance. Recently, SOSNet (Tian et al., 2019a) proposes a second order similarity regularization term, and achieves more compact patch clusters in the feature space.

The triplet mining framework can generally be decomposed into two stages, *i.e.*, batch construction from the dataset and triplet generation within the mini-batch. Previous works mainly focus on mining hard negatives in the second stage, while the first stage is often cursorily done by random sampling. Therefore, we argue that their triplet sampling solutions are still sub-optimal. To address this problem, we propose *AdaSample* to construct informative batches in the first place. Combined with the hardest-in-batch solution in Mishchuk et al. (2017) to mine negatives within the mini-batches, we formulate a complete and powerful triplet mining framework for descriptor learning.

Hard Negative Mining. Hard negative mining is widely used in object detection (Tian et al., 2019b) and other vision tasks for collecting good training examples. FaceNet (Schroff et al., 2015) proposes to sample semi-hard triplets within the batch. Wu et al. (2017) select training examples based on their relative distances. Our sampling solution differs from them in that we analyze the *informativeness* of the training data and ensure the most useful gradients for parameters update. Besides, our method adaptively adjusts the hard level of the selected training samples with training. Therefore, well-classified samples are filtered out, and the network is always fed with informative triplets with suitable hard levels. Comprehensive results demonstrate consistent performance improvement contributed by our proposed approach.

3 METHODOLOGY

3.1 PROBLEM OVERVIEW

Given a dataset that consists of N classes with each containing k patches, we decompose the triplet generation into two stages. Firstly, we select n matching pairs (positives) to form a mini-batch, where n is the batch size. This is done by proposed *AdaSample*, as introduced in Section 3.2. Secondly, we mine the hardest-in-batch negatives for each matching pair, and use the triplet loss to supervise the network training, as in Section 3.3. The overall solution is summarized in Section 3.4.

3.2 ADASAMPLE

Previous works (Tian et al., 2017; Mishchuk et al., 2017) sample positives randomly to construct batches, yielding a majority of similar matching pairs which can be easily discriminated by the network. This reduces the overall "hardness" of the triplets. Motivated by the hardest-in-batch mining strategy in Mishchuk et al. (2017), a naive solution is to select the most dissimilar matching pairs. However, this has potential issues, *i.e.*, the network may be trained with bias in favor of dissimilar matching pairs, while other cases are less-considered. We validate this solution, nominated as *Hardpos*, in experiments (Section 5.3).

A more principled solution is to sample positives based on their *informativeness*. Here we assume that more informative pairs are those contributing more to the optimization, *i.e.*, providing effective gradients for parameters update. Therefore, we quantify the *informativeness* of matching pairs by measuring their contributing gradients during training. Moreover, we employ *maximum loss minimization* (Shalev-Shwartz & Wexler, 2016) to improve the generalization ability of the learned model. We then show that the resulting gradient estimator is an *unbiased* estimator of the true gradient. In the following, we introduce our derivation and elaborate theoretical justification in Section 4.

Informativeness Based Sampling. In the end-to-end learning literature, the training data contribute to learning via gradients, so we measure the *informativeness* of training examples by analyz-

ing their resulting gradients. Generally, we consider the generic deep learning training framework, in which each data-label pair contributes independent gradients. Let $(\boldsymbol{x}_i, \boldsymbol{y}_i)$ be the i^{th} data-label pair of the training set, $f(\boldsymbol{x}; \boldsymbol{\theta})$ be the model parameterized by $\boldsymbol{\theta}$, and $\mathcal{L}(\cdot, \cdot)$ be a differentiable loss function. The goal is to find the optimal model parameters $\boldsymbol{\theta}^*$ that minimize the average loss, *i.e.*,

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \frac{1}{K} \sum_{i=1}^{K} \mathcal{L}(f(\boldsymbol{x}_i; \boldsymbol{\theta}), \boldsymbol{y}_i), \tag{1}$$

where K denotes the number of training examples. Then we proceed our idea with the following definition on the *informativeness*.

Definition 1. *The informativeness of a training example $(\boldsymbol{x}_i, \boldsymbol{y}_i)$ is quantified by how much it contributes in the training iteration t, namely,*

$$\text{info}(\boldsymbol{x}_i, \boldsymbol{y}_i) := ||\nabla_{\boldsymbol{\theta}_t} \mathcal{L}(f(\boldsymbol{x}_i; \boldsymbol{\theta}), \boldsymbol{y}_i)||_2. \tag{2}$$

At iteration t, let $P_t = \{p_1^t, \ldots, p_K^t\}$ be the sampling probabilities of each datum in the training set. More generally, we also re-weight each sample by $w_1^t, \cdots, w_K^t$. Let random variable I_t denote the sampled index at iteration t, then $I_t \sim P_t$, namely, $\mathbb{P}(I_t = i) = p_i^t$. We record the re-weighted gradient induced by training sample $(\boldsymbol{x}_i, \boldsymbol{y}_i)$ as

$$\boldsymbol{G}_i^t = w_i^t \nabla_{\boldsymbol{\theta}_t} \mathcal{L}(f(\boldsymbol{x}_i; \boldsymbol{\theta}), \boldsymbol{y}_i). \tag{3}$$

For simplicity, we omit the superscript t when no ambiguity is made. By setting $w_i = \frac{1}{K p_i}$, we can make the gradient estimator $\boldsymbol{G}_i$ an unbiased estimator of the true gradient, *i.e.*,

$$\mathbb{E}_{I_t \sim P_t} \left[\boldsymbol{G}_{I_t} \right] = \nabla_{\boldsymbol{\theta}_t} \frac{1}{K} \sum_{i=1}^{K} \mathcal{L}(f(\boldsymbol{x}_i; \boldsymbol{\theta}), \boldsymbol{y}_i). \tag{4}$$

Without loss of generality, we use stochastic gradient descent (SGD) to update model parameters:

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta_t w_{I_t} \nabla_{\boldsymbol{\theta}_t} \mathcal{L}(f(\boldsymbol{x}_{I_t}; \boldsymbol{\theta}), \boldsymbol{y}_{I_t}) = \boldsymbol{\theta}_t - \eta_t \boldsymbol{G}_{I_t}, \tag{5}$$

where η_t is the learning rate at iteration t. As the goal is to find optimal $\boldsymbol{\theta}^*$, we define the expected progress towards the optimum at each iteration as follows.

Definition 2. *At iteration t, the expected parameter rectification R_t is defined as the expected reduction of the squared distance between the parameter $\boldsymbol{\theta}$ and the optimum $\boldsymbol{\theta}^*$ after iteration t,*

$$R_t := -\mathbb{E}_{I_t \sim P_t} \left[||\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}^*||_2^2 - ||\boldsymbol{\theta}_t - \boldsymbol{\theta}^*||_2^2 \right]. \tag{6}$$

Generally, tens of thousands iterations are included in the training, so the empirical average parameter rectification will converge to the average of R_t asymptotically. By maximizing R_t, we guarantee the most progressive step towards parameters optimum at each iteration in the expectation sense. Inspired by the *greedy algorithm* (Edmonds, 1971), we aim to maximize R_t at each iteration.

It can be shown that maximizing R_t is equivalent to minimizing $tr(\text{Var}\left[\boldsymbol{G}_{I_t}\right])$, as shown in Theorem 1. Under this umbrella, we show that the optimal sampling probability is proportional to the per-sample gradient norm (a special case of Theorem 2). Therefore, the optimal sampling probability of each datum happens to be proportional to its *informativeness*. This property justifies our definition of *informativeness* as the resulting gradient norm of each training example.

However, as the neural network has multiple layers with a large number of parameters, it is computationally prohibitive to calculate the full gradient norm. Instead, we prove that the matching distance in the feature space is a good approximation to the *informativeness*[1] in Section 4.2. Concretely, for each class consisting of k patches $\{\boldsymbol{X}_i : i = 1, \ldots, k\}$, we first select a patch $\boldsymbol{X}_{i_0}$ randomly, which serves as the anchor patch, and then sample a matching patch $\boldsymbol{X}_i$ with probability

$$p_i \propto d(\boldsymbol{x}_i, \boldsymbol{x}_{i_0}), \quad for\ i \neq i_0, \tag{7}$$

where $\boldsymbol{x}_i$ is the extracted descriptor of $\boldsymbol{X}_i$, and $d(\cdot, \cdot)$ measures the discrepancy of the extracted descriptors. See specific choices of d in Section 3.4.

[1]The approximation is up to a constant factor, which is insignificant as it will be offset by the learning rate. The same reasoning applies to the approximation of gradients in **Maximum Loss Minimization** paragraph.

Maximum Loss Minimization. Minimizing the average loss may be sub-optimal because the training tends to be overwhelmed by well-classified examples that may contribute noisy gradients (Lin et al., 2017). On the contrary, well-classified examples can be adaptively filtered out by minimizing the maximum loss (Shalev-Shwartz & Wexler, 2016), which can further improve the generalization ability of models. However, directly minimizing the maximum loss may lead to insufficient usage of data and sensitivity to outliers, so we approximate the gradient of maximum loss by $\nabla_{\boldsymbol{\theta}_t} \frac{1}{K} \sum_{i=1}^{K} \mathcal{L}_i^{\alpha}$, in which α is sufficiently large. As $\boldsymbol{G}_{I_t}$ is used to update parameters, consider its expectation

$$\mathbb{E}_{I_t \sim P_t} \left[\boldsymbol{G}_{I_t} \right] = \mathbb{E}_{I_t \sim P_t} \left[w_{I_t} \nabla_{\boldsymbol{\theta}_t} \mathcal{L}_{I_t} \right] = \sum_{i=1}^{K} p_i w_i \nabla_{\boldsymbol{\theta}_t} \mathcal{L}_i. \tag{8}$$

To guarantee $\boldsymbol{G}_{I_t}$ to be an unbiased estimator of $\nabla_{\boldsymbol{\theta}_t} \frac{1}{K} \sum_{i=1}^{K} \mathcal{L}_i^{\alpha}$, it suffices to set

$$p_i w_i = \frac{\alpha}{K} \mathcal{L}_i^{\alpha - 1}, \tag{9}$$

as in this case,

$$\mathbb{E}_{I_t \sim P_t} \left[\boldsymbol{G}_{I_t} \right] = \sum_{i=1}^{K} \frac{\alpha}{K} \mathcal{L}_i^{\alpha - 1} \nabla_{\boldsymbol{\theta}_t} \mathcal{L}_i = \sum_{i=1}^{K} \frac{1}{K} \nabla_{\boldsymbol{\theta}_t} \mathcal{L}_i^{\alpha}. \tag{10}$$

Following previous reasoning, we need minimize $tr(\mathrm{Var}\left[\boldsymbol{G}_{I_t}\right])$ given the constraints in Equation 9, in order to step most progressively at each iteration. In Theorem 2, we show that the optimal sampling probability and re-weighting scalar should be given by

$$p_i \propto \mathcal{L}_i^{\alpha - 1} ||\nabla_{\boldsymbol{\theta}_t} \mathcal{L}_i||_2 \ \ and \ \ w_i \propto ||\nabla_{\boldsymbol{\theta}_t} \mathcal{L}_i||_2^{-1}. \tag{11}$$

As previously claimed, we approximate the gradient norm via the matching distance in the feature space. Besides, in our case, the loss function is hinge triplet loss (Equation 14), which is positively (or even linearly) correlated with the matching distance squared. Therefore, we use matching distance squared as an approximation of the loss. Thus, the sampling probability and re-weighting scalar are given by

$$p_i \propto d(\boldsymbol{x}_i, \boldsymbol{x}_{i_0})^{2\alpha - 1} \ \ and \ \ w_i \propto d(\boldsymbol{x}_i, \boldsymbol{x}_{i_0})^{-1}, \ \ for \ i \neq i_0. \tag{12}$$

Moreover, for better approximation, it is preferable to adjust α adaptively, *i.e.*, increase α with training. Intuitively, when easy matching pairs have already been correctly classified, we focus more on hard ones. A good indicator of the training progress is the *average loss*. As a result, instead of predefining a sufficiently large α, we set $2\alpha - 1 = \lambda / \mathcal{L}_{avg}$, where λ is a tunable hyperparameter and $\mathcal{L}_{avg}$ is the moving average of history loss. Formally, we formulate our sampling probability and re-weighting scalar as

$$p_i \propto d(\boldsymbol{x}_i, \boldsymbol{x}_{i_0})^{\frac{\lambda}{\mathcal{L}_{avg}}} \ \ and \ \ w_i \propto d(\boldsymbol{x}_i, \boldsymbol{x}_{i_0})^{-1}, \ \ for \ i \neq i_0. \tag{13}$$

The exponent increases adaptively with training, so our sampling method is named as *AdaSample*.

3.3 Triplet generation by hardest-in-batch

AdaSample focuses on the batch construction stage; nevertheless, for a complete triplet mining framework, we need to mine negatives from the mini-batch as well. Here, we adopt the hardest-in-batch strategy in Mishchuk et al. (2017). Formally, given a mini-batch of n matching pairs $\{(\tilde{\boldsymbol{X}}_i, \tilde{\boldsymbol{X}}_i^+) : i = 1, \ldots, n\}$, let $(\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{x}}_i^+)$ be the descriptors extracted from $(\tilde{\boldsymbol{X}}_i, \tilde{\boldsymbol{X}}_i^+)$ [2]. For each matching pair $(\tilde{\boldsymbol{X}}_i, \tilde{\boldsymbol{X}}_i^+)$, we select the non-matching patch which lies closest to one of the matching patches in the feature space. Then, the Hinge Triplet (HT) loss is defined as follows:

$$\begin{aligned} \mathcal{L}_i &= \max\left\{t + (d_i^{pos})^2 - (d_i^{neg})^2,\ 0\right\}, \\ d_i^{pos} &= d(\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{x}}_i^+), \\ d_i^{neg} &= \min_{j \neq i} \left\{ \min\left\{ d(\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{x}}_j),\ d(\tilde{\boldsymbol{x}}_i^+, \tilde{\boldsymbol{x}}_j^+) \right\} \right\}, \end{aligned} \tag{14}$$

where t denotes the margin. Incorporating the re-weighting scalar, we update the model parameters via the gradient estimator $\sum_{i=1}^{n} w_i \nabla_{\boldsymbol{\theta}} \mathcal{L}_i$.

[2] For clarity, $(\tilde{\boldsymbol{X}}_\diamond, \tilde{\boldsymbol{X}}_\diamond^+)$ denotes the selected matching pairs, with different pairs belonging to different classes. $\boldsymbol{X}_\diamond$ denotes a generic patch in a specific class, where $\diamond$ denotes the placeholder for the index.

3.4 DISTANCE METRIC

Euclidean distance is widely used in previous works (Tian et al., 2017; Mishchuk et al., 2017; Tian et al., 2019a). However, as descriptors lie on the unit hypersphere in 128-dimensional space (Appendix C), it is more natural to adopt the geodesic distance of the embedded manifold. Therefore, we adopt the angular distance (Deng et al., 2018) as follows:

$$d(\tilde{\boldsymbol{x}}_1, \tilde{\boldsymbol{x}}_2) = \arccos(\tilde{\boldsymbol{x}}_1 \cdot \tilde{\boldsymbol{x}}_2), \tag{15}$$

where $\cdot$ denotes inner product. We nominate our loss function as Angular Hinge Triplet (AHT) loss, which is demonstrated to result in consistent performance improvement (see Section 5.3).

Algorithm 1 summarizes the overall triplet generation framework. For each training iteration, we first randomly pick n distinct classes from the dataset and extract descriptors for patches belonging to these classes (Step 1, 2). Then, we randomly choose a patch as anchor from each selected class (Step 4), and adopt *AdaSample* to select an informative matching patch (Step 5). With the generated mini-batch, we mine hard negatives following Mishchuk et al. (2017) and compute Angular Hinge Triplet (AHT) loss (Step 7). An illustration of our sampling pipeline can be found in Appendix B.

Algorithm 1 Pipeline of *AdaSample* framework.

Require:
Dataset of N classes with each containing k matching patches;
Moving average of history loss $\mathcal{L}_{avg}$;
Hyperparameter λ;

1: Randomly select n distinct classes from the dataset without replacement;
2: Extract descriptors of the patches belonging to the selected classes;
3: **for** each selected class with k patches $\{\boldsymbol{X}_i : i = 1, \ldots, k\}$ **do**
4: Sample an anchor patch $\boldsymbol{X}_{i_0}$ randomly;
5: Sample a matching patch $\boldsymbol{X}_i$ from the remaining patches with probabilities specified by Equation 13;
6: **end for**
7: With sampled positive pairs and their descriptors $\{(\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{x}}_i^+)\}_{i=1}^n$, compute Angular Triplet Hinge loss by Equation 14;
8: Backpropagate and update model parameters via $\sum_{i=1}^n w_i \nabla_{\boldsymbol{\theta}} \mathcal{L}_i$;

4 THEORETICAL ANALYSIS

In this section, we complete the theoretical analysis of *informativeness* in Section 4.1, and prove that the matching distance can serve as a good approximation of *informativeness* in Section 4.2.

4.1 INFORMATIVENESS FORMULATION

Following notations in Section 3.2, we reformulate R_t (Equation 6), and give an equivalent condition for maximizing R_t. The same conclusion can be found in Katharopoulos & Fleuret (2018).

Theorem 1. *Let R_t, $\boldsymbol{\theta}^*$ and $\boldsymbol{G}_i$ be defined as in Equation 6, 1 and 3, respectively. Then, we have*

$$R_t = 2\eta_t(\boldsymbol{\theta}_t - \boldsymbol{\theta}^*)^T \mathbb{E}_{I_t \sim P_t}\left[\boldsymbol{G}_{I_t}\right] - \eta_t^2 \mathbb{E}_{I_t \sim P_t}\left[\boldsymbol{G}_{I_t}\right]^T \mathbb{E}_{I_t \sim P_t}\left[\boldsymbol{G}_{I_t}\right] - \eta_t^2 tr(\mathrm{Var}\left[\boldsymbol{G}_{I_t}\right]). \tag{16}$$

Due to unbiasedness (Equation 4), the first two terms in Equation 16 is fixed, so maximizing R_t is equivalent to minimizing $tr(\mathrm{Var}\left[\boldsymbol{G}_{I_t}\right])$. Theorem 2 specifies the optimal probabilities to minimize the aforementioned trace under a more general assumption.

Theorem 2. *Let $\boldsymbol{G}_i$ be defined in Equation 3 and suppose the sampled index I_t obeys distribution P_t. Then, given the constraints $p_i w_i = \frac{\alpha}{K}\mathcal{L}_i^{\alpha-1}$, $tr(\mathrm{Var}\left[\boldsymbol{G}_{I_t}\right])$ is minimized by the following optimal sampling probabilities:*

$$p_i = \frac{1}{Z}\mathcal{L}_i^{\alpha-1}||\nabla_{\boldsymbol{\theta}_t}\mathcal{L}_i||_2, \ where\ Z = \sum_{j=1}^{K}\mathcal{L}_j^{\alpha-1}||\nabla_{\boldsymbol{\theta}_t}\mathcal{L}_j||_2. \tag{17}$$

Note that in the special case of $\alpha = 1$, the constraints degrade into $p_i w_i = \frac{1}{K}$ and the optimal sampling probabilities become $p_i \propto ||\nabla_{\boldsymbol{\theta}_t}\mathcal{L}_i||_2$. The proof of the above theorems can be found in Appendix A.

4.2 APPROXIMATION OF INFORMATIVENESS

As mentioned in Section 3.2, the matching distance can serve as a good approximation of *informativeness*, up to a constant factor. We justify this here. For simplicity, we introduce some notations for a L-layer multi-layer perceptron (MLP). Let $\boldsymbol{\theta}^{(l)} \in \mathbb{R}^{M_l \times M_{l-1}}$ be the weight matrix for layer l and $g^{(l)}$ be a Lipschitz continuous activation function. Then the multi-layer perceptron can be formulated as follows:

$$\begin{aligned} \boldsymbol{x}^{(0)} &= \boldsymbol{x}, \\ \boldsymbol{h}^{(l)} &= \boldsymbol{\theta}^{(l)} \boldsymbol{x}^{(l-1)}, \quad for\ l = 1, \cdots, L, \\ \boldsymbol{x}^{(l)} &= g^{(l)}(\boldsymbol{h}^{(l)}), \quad for\ l = 1, \cdots, L, \\ f(\boldsymbol{x}; \boldsymbol{\theta}) &= \boldsymbol{x}^{(L)}, \\ \boldsymbol{\theta} &= \{\boldsymbol{\theta}^{(1)}, \ldots, \boldsymbol{\theta}^{(L)}\}. \end{aligned} \tag{18}$$

Note that although our notations describe only MLPs without bias, our analysis holds for any affine transformation followed by a Lipschitz continuous non-linearity. Therefore, our reasoning can naturally extend to convolutional neural networks (CNNs). With

$$\begin{aligned} \boldsymbol{\Gamma}_l(\boldsymbol{h}^{(l)}) &= \text{diag}\left\{ g'^{(l)}(h_1^{(l)}), \ldots, g'^{(l)}(h_{M_l}^{(l)}) \right\}, \\ \boldsymbol{\Pi}^{(l)} &= \boldsymbol{\Gamma}_l(\boldsymbol{h}^{(l)}) \boldsymbol{\theta}_{l+1}^T \cdots \boldsymbol{\Gamma}_{L-1}(\boldsymbol{h}^{(L-1)}) \boldsymbol{\theta}_L^T \boldsymbol{\Gamma}_L(h^{(L)}), \end{aligned} \tag{19}$$

we have

$$||\nabla_{\boldsymbol{\theta}_l} \mathcal{L}(f(\boldsymbol{x}; \boldsymbol{\theta}), \boldsymbol{y})||_2 = \left\| \left(\boldsymbol{\Pi}^{(l)} \nabla_{\boldsymbol{x}^{(L)}} \mathcal{L} \right) \left(\boldsymbol{x}^{(l-1)} \right)^T \right\|_2 \leq ||\boldsymbol{\Pi}^{(l)}||_2 ||\boldsymbol{x}^{(l-1)}||_2 ||\nabla_{\boldsymbol{x}^{(L)}} \mathcal{L}||_2. \tag{20}$$

Various data preprocessing, weight initialization (Glorot & Bengio, 2010; He et al., 2015), and activation normalization (Ioffe & Szegedy, 2015; Lei Ba et al., 2016; Ulyanov et al., 2016) techniques uniformise the activations of each layer across samples. Therefore, the variation of gradient norms is mostly captured by the gradient of the loss function *w.r.t.* the output of neural networks,

$$\text{info}(\boldsymbol{x}, \boldsymbol{y}) = ||\nabla_{\boldsymbol{\theta}} \mathcal{L}(f(\boldsymbol{x}; \boldsymbol{\theta}), \boldsymbol{y})||_2 \approx M \left\| \nabla_{\boldsymbol{x}^{(L)}} \mathcal{L} \right\|_2, \tag{21}$$

where M is a constant, and $M \left\| \nabla_{\boldsymbol{x}^{(L)}} \mathcal{L} \right\|_2$ serves as a precise approximation of the full gradient norm. For simplicity, we consider hinge triplet loss (Equation 14) here. Then, the gradient norm *w.r.t.* the descriptor of the matching patch is just twice the matching distance[3],

$$\left\| \nabla_{\boldsymbol{x}^{(L)}} \mathcal{L} \right\|_2 = 2 d^{pos}. \tag{22}$$

As a result, we get the conclusion that the matching distance is a good approximation to *informativeness*, up to a constant factor. Also, we empirically verify this in Section 5.3.

5 EXPERIMENTS

We first compare with the state-of-the-art methods on two standard descriptor datasets: UBC Phototour (Brown et al., 2011) and HPatches (Balntas et al., 2017). Then, we make comprehensive ablation studies. See implementation details in Appendix C and more results on the ETH SfM (Schonberger et al., 2017) dataset in Appendix D.

5.1 UBC PHOTOTOUR

UBC Phototour (Brown et al., 2011), also known as Brown dataset, consists of three subsets: *Liberty*, *Notre Dame* and *Yosemite*, with about 400K normalized 64×64 patches in each subset. Keypoints were detected by DoG detector (Lowe, 2004) and verified by 3D model. Testing set consists of 100K matching and non-matching pairs for each sequence. For evaluation, models are trained on one subset and tested on the other two. The metric is the false positive rate (FPR) at 95% true positive recall. The evaluation results are reported in Table 1.

[3]This relation holds only when the hinge triplet loss is positive. Empirically, due to the relatively large margin, the hinge loss never becomes zero.

Table 1: Patch classfication results on UBC Phototour dataset (Brown et al., 2011). The false positive rate at 95% recall is reported. + indicates data augmentation and ∗ indicates positive generation.

Train	Notredame	Yosemite	Liberty	Yosemite	Liberty	Notredame	Mean
Test	Liberty		Notredame		Yosemite		
SIFT (Lowe, 2004)	29.84		22.53		27.29		26.55
DeepDesc (Simo-Serra et al., 2015)	10.9		4.40		5.69		6.99
GeoDesc (Luo et al., 2018)	5.47		1.94		4.72		4.05
L2-Net (Tian et al., 2017)	3.64	5.29	1.15	1.62	4.43	3.30	3.24
CS-L2-Net (Tian et al., 2017)	2.55	4.24	0.87	1.39	3.81	2.84	2.61
HardNet (Mishchuk et al., 2017)	**1.47**	2.67	0.62	**0.88**	2.14	1.65	1.57
HardNet-GOR (Zhang et al., 2017)	1.72	2.89	0.63	0.91	2.10	1.59	1.64
HardNet*	1.80	2.89	0.68	0.90	1.93	1.71	1.65
AdaSample* (**Ours**)	1.64	**2.62**	**0.61**	**0.88**	**1.92**	**1.46**	**1.52**
L2-Net+ (Tian et al., 2017)	2.36	4.70	0.72	1.29	2.57	1.71	2.23
CS-L2-Net+ (Tian et al., 2017)	1.71	3.87	0.56	1.09	2.07	1.30	1.76
HardNet+ (Mishchuk et al., 2017)	1.49	2.51	0.53	0.78	1.96	1.84	1.51
HardNet-GOR+ (Zhang et al., 2017)	1.48	2.43	0.51	0.78	1.76	1.53	1.41
DOAP+ (He et al., 2018)	1.54	2.62	0.43	0.87	2.00	1.21	1.45
HardNet+*	1.35	2.28	0.43	0.69	1.64	1.15	1.26
AdaSample+* (**Ours**)	**1.23**	**2.19**	**0.41**	**0.62**	**1.46**	**1.05**	**1.16**

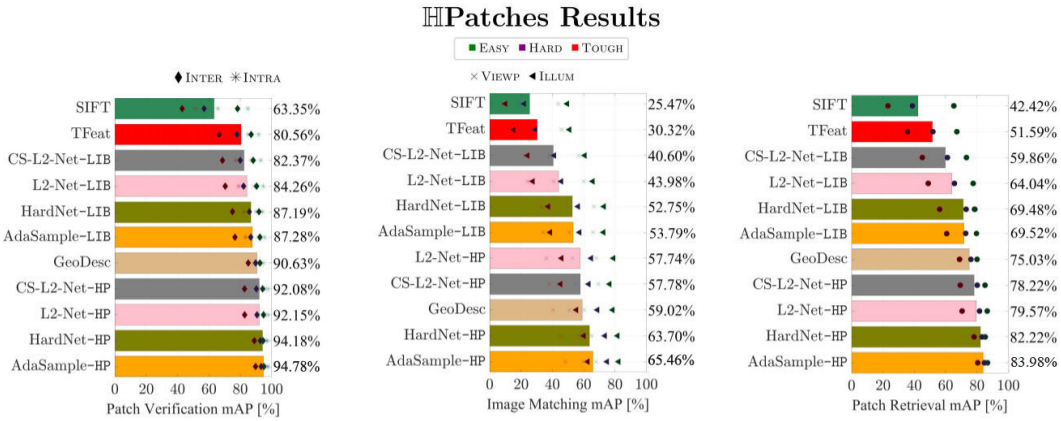

Figure 1: Evaluation results on HPatches dataset (Balntas et al., 2017). "−LIB" indicates descriptors trained on *Liberty* subset of UBC Phototour dataset (Brown et al., 2011) and "−HP" indicates descriptors trained on training set of HPatches (split 'a'). Marker color indicates the level of geometrical noises and marker type indicates the experimental setup. INTER and INTRA indicate the source of negative examples for the *verification* task. VIEWP and ILLUM indicate the sequence type for the *matching* task.

Our method outperforms other approaches by a significant margin. We randomly flip and rotate by 90 degree for data augmentation, noted by +, for all methods. Besides, for our method, we also generate positive patches by random rotation such that each class has 15 patches, noted by *. This is because there are too few patches (two or three) corresponding to one class in UBC Phototour dataset (Brown et al., 2011), which limits the capacity of our method. To analyze its effect, we also conduct it for HardNet (Mishchuk et al., 2017), and observe inferior performance, indicating that the performance improvement mainly comes from our adaptive sampling solution.

5.2 HPATCHES

HPatches (Balntas et al., 2017) consists of 116 sequences of 6 images. The dataset is split into two parts: *viewpoint* - 59 sequences with significant viewpoint change and *illumination* - 57 sequences with significant illumination change. According to the level of geometric noises, the patches can be further divided into three groups: *easy*, *hard* and *tough*. There are three evaluation tasks: *patch verification*, *image matching* and *patch retrieval*. Following standard evaluation protocols of the dataset, we show results in Figure 1. It shows that our method outperforms other methods on all three tasks when trained on either *Liberty* dataset (Brown et al., 2011) or HPatches training split.

5.3 ABLATION STUDY

Informativeness Approximation. We empirically verify the conclusion in Section 4.2 that the probability induced by matching distance is a good approximation to the one induced by *informativeness* (Figure 2(a)). Besides, the results show that the Pearson correlation is consistently greater than 0.8 with training (Figure 2(b)), which indicates these probabilities have strong correlation with each other statistically. It echoes our theoretical conclusion.

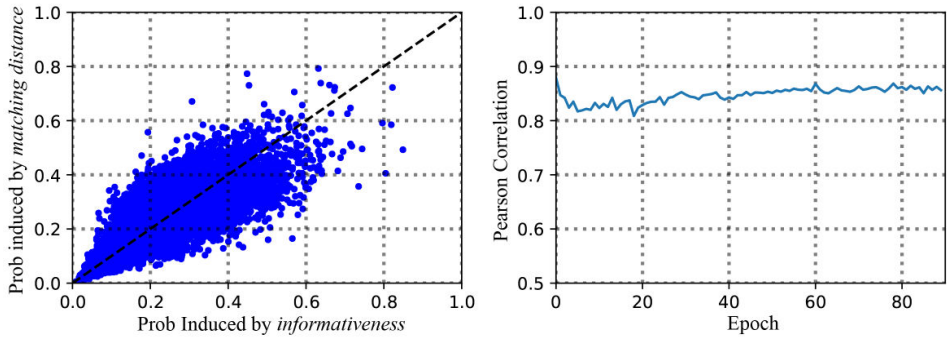

Figure 2: (a) Probabilities induced by informativeness and matching distance. (b) Pearson correlation between probabilities and training epoches.

Impact of λ. We experiment with varying λ in *AdaSample*, which controls the overall "hardness" of selected matching pairs. The larger λ is, the more likely hard matching pairs will be selected. When $\lambda = 0$, our method degrades into random sampling and overall framework becomes HardNet (Mishchuk et al., 2017), and as $\lambda \to +\infty$, the framework becomes *Hardpos*. Therefore, both HardNet and *Hardpos* are special cases of our proposed *AdaSample*. Table 2 shows the results on HPatches dataset (Balntas et al., 2017), where $\lambda = 10$ leads to the best results in most cases. It demonstrates the advantages of our balanced sampling strategy against the hardest solution.

Table 2: Abaltion study results on HPatches (Balntas et al., 2017).

Task	Verification		Matching		Retrieval	
Loss	AHT	HT	AHT	HT	AHT	HT
$\lambda = 1$	93.84	93.17	64.09	62.64	81.26	79.97
$\lambda = 2$	94.72	94.56	**66.04**	65.92	83.58	83.34
$\lambda = 5$	**94.78**	94.76	65.89	65.68	83.80	83.54
$\lambda = 10$	**94.78**	94.60	65.46	65.37	**83.98**	83.62
$\lambda = 20$	94.60	94.69	64.56	64.84	83.56	83.69
$\lambda \to +\infty$	94.42	94.51	63.81	64.02	83.41	83.29

Distance metric and training speed. As shown in Table 2, our angular hinge triplet (AHT) loss outperforms the commonly-used hinge triplet (HT) loss in most cases. Besides, our method can speedup the network convergence, as non-informative matching pairs are adaptively filtered out. See visualized learning trends in Appendix E.

6 CONCLUSION

This paper proposes *AdaSample* for descriptor learning, which adaptively samples hard positives to construct informative mini-batches during training. We demonstrate the efficacy of our method from both theoretical and empirical perspectives. Theoretically, we give a rigorous definition of *informativeness* of potential training examples. Then, we reformulate the problem and derive a tractable sampling probability expression (Equation 13) based on *informativeness*. Empirically, we enjoy consistent performance gain on top of HardNet (Mishchuk et al., 2017) baseline when evaluated on various task, including patch classification, patch verification, image matching, and patch retrieval.

REFERENCES

Sameer Agarwal, Noah Snavely, Ian Simon, Steven M Seitz, and Richard Szeliski. Building rome in a day. In *IEEE International Conference on Computer Vision (ICCV)*, pp. 72–79. IEEE, 2009.

Vassileios Balntas, Edgar Riba, Daniel Ponsa, and Krystian Mikolajczyk. Learning local feature descriptors with triplets and shallow convolutional neural networks. In *British Machine Vision Conference (BMVC)*, 2016.

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)*, pp. 5173–5182, 2017.

JiaWang Bian, Wen-Yan Lin, Yasuyuki Matsushita, Sai-Kit Yeung, Tan-Dat Nguyen, and Ming-Ming Cheng. GMS: Grid-based motion statistics for fast, ultra-robust feature correspondence. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

Matthew Brown and David G Lowe. Automatic panoramic image stitching using invariant features. *International Journal on Computer Vision (IJCV)*, 74(1):59–73, 2007.

Matthew Brown, Gang Hua, and Simon Winder. Discriminative learning of local image descriptors. *IEEE Transactions on Pattern Recognition and Machine Intelligence (PAMI)*, 33(1):43–57, 2011.

Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. ArcFace: Additive angular margin loss for deep face recognition. *arXiv preprint arXiv:1801.07698*, 2018.

Jingming Dong and Stefano Soatto. Domain-size pooling in local descriptors: DSP-SIFT. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5097–5106, 2015.

Jack Edmonds. Matroids and the greedy algorithm. *Mathematical programming*, 1(1):127–136, 1971.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256, 2010.

Xufeng Han, Thomas Leung, Yangqing Jia, Rahul Sukthankar, and Alexander C Berg. Matchnet: Unifying feature and metric learning for patch-based matching. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3279–3286, 2015.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *IEEE International Conference on Computer Vision (ICCV)*, pp. 1026–1034, 2015.

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

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

Angelos Katharopoulos and François Fleuret. Not all samples are created equal: Deep learning with importance sampling. *arXiv preprint arXiv:1803.00942*, 2018.

BG Kumar, Gustavo Carneiro, Ian Reid, et al. 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)*, pp. 5385–5394, 2016.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *IEEE International Conference on Computer Vision (ICCV)*, pp. 2980–2988, 2017.

David G Lowe. Distinctive image features from scale-invariant keypoints. *International Journal on Computer Vision (IJCV)*, 60(2):91–110, 2004.

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)*, pp. 168–183, 2018.

Anastasiia Mishchuk, Dmytro Mishkin, Filip Radenovic, and Jiri Matas. Working hard to know your neighbor's margins: Local descriptor learning loss. In *Neural Information Processing Systems (NIPS)*, pp. 4826–4837, 2017.

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. 2017.

James Philbin, Michael Isard, Josef Sivic, and Andrew Zisserman. Descriptor learning for efficient retrieval. In *European Conference on Computer Vision (ECCV)*, pp. 677–691. Springer, 2010.

Johannes L Schonberger, Hans Hardmeier, Torsten Sattler, and Marc Pollefeys. Comparative evaluation of hand-crafted and learned local features. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1482–1491, 2017.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. FaceNet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 815–823, 2015.

Shai Shalev-Shwartz and Yonatan Wexler. Minimizing the maximal loss: How and why? In *ICML*, 2016.

E. Simo-Serra, E. Trulls, L. Ferraz, I. Kokkinos, P. Fua, and F. Moreno-Noguer. Discriminative learning of deep convolutional feature point descriptors. In *IEEE International Conference on Computer Vision (ICCV)*, pp. 118–126, Dec 2015.

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)*, pp. 118–126, 2015.

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)*, pp. 661–669, 2017.

Yurun Tian, Xin Yu, Bin Fan, Fuchao Wu, Huub Heijnen, and Vassileios Balntas. SOSNet: Second order similarity regularization for local descriptor learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019a.

Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. *arXiv preprint arXiv:1904.01355*, 2019b.

Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance Normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016.

Chao-Yuan Wu, R Manmatha, Alexander J Smola, and Philipp Krahenbuhl. Sampling matters in deep embedding learning. In *IEEE International Conference on Computer Vision (ICCV)*, pp. 2840–2848, 2017.

Kwang Moo Yi, Eduard Trulls, Vincent Lepetit, and Pascal Fua. LIFT: Learned invariant feature transform. In *European Conference on Computer Vision (ECCV)*, pp. 467–483. Springer, 2016.

Kwang Moo Yi, Eduard Trulls Fortuny, Yuki Ono, Vincent Lepetit, Mathieu Salzmann, and Pascal Fua. Learning to find good correspondences. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

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

Xu Zhang, Felix X Yu, Sanjiv Kumar, and Shih-Fu Chang. Learning spread-out local feature descriptors. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4595–4603, 2017.

A PROOF OF THEOREMS

Proof of Theorem 1.

Proof. We reformulate R_t as follows,

$$\begin{aligned}
R_t &= -\mathbb{E}_{I_t\sim P_t}\left[(\boldsymbol{\theta}_{t+1}-\boldsymbol{\theta}^*)^T(\boldsymbol{\theta}_{t+1}-\boldsymbol{\theta}^*)-(\boldsymbol{\theta}_t-\boldsymbol{\theta}^*)^T(\boldsymbol{\theta}_t-\boldsymbol{\theta}^*)\right] \\
&= -\mathbb{E}_{I_t\sim P_t}\left[\boldsymbol{\theta}_{t+1}^T\boldsymbol{\theta}_{t+1}-2\boldsymbol{\theta}_{t+1}^T\boldsymbol{\theta}^*-\boldsymbol{\theta}_t^T\boldsymbol{\theta}_t+2\boldsymbol{\theta}_t^T\boldsymbol{\theta}^*\right] \\
&= -\mathbb{E}_{I_t\sim P_t}\left[(\boldsymbol{\theta}_t-\eta_t\boldsymbol{G}_{I_t})^T(\boldsymbol{\theta}_t-\eta_t\boldsymbol{G}_{I_t})+2\eta_t\boldsymbol{G}_{I_t}^T\boldsymbol{\theta}^*-\boldsymbol{\theta}_t^T\boldsymbol{\theta}_t\right] \qquad \text{(Equation 5)} \\
&= \mathbb{E}_{I_t\sim P_t}\left[2\eta_t(\boldsymbol{\theta}_t-\boldsymbol{\theta}^*)^T\boldsymbol{G}_{I_t}-\eta_t^2\boldsymbol{G}_{I_t}^T\boldsymbol{G}_{I_t}\right] \\
&= 2\eta_t(\boldsymbol{\theta}_t-\boldsymbol{\theta}^*)^T\mathbb{E}_{I_t\sim P_t}\left[\boldsymbol{G}_{I_t}\right]-\eta_t^2\mathbb{E}_{I_t\sim P_t}\left[\boldsymbol{G}_{I_t}\right]^T\mathbb{E}_{I_t\sim P_t}[\boldsymbol{G}_{I_t}]-\eta_t^2 tr(\mathrm{Var}\left[\boldsymbol{G}_{I_t}\right]). \qquad (23)
\end{aligned}$$

□

Proof of Theorem 2.

Proof. As $\boldsymbol{G}_{I_t}$ is an *unbiased* estimator of the true gradient (Equation 4), $\mathbb{E}_{I_t\sim P_t}\left[\boldsymbol{G}_{I_t}\right]$ is fixed in our case, denoted by $\boldsymbol{\mu}$ for short. By the linearity of trace and $tr(\boldsymbol{\mu}\boldsymbol{\mu}^T)=||\boldsymbol{\mu}||_2^2$,

$$\begin{aligned}
tr(\mathrm{Var}\left[\boldsymbol{G}_{I_t}\right]) &= tr(\mathbb{E}_{I_t\sim P_t}\left[(\boldsymbol{G}_{I_t}-\boldsymbol{\mu})(\boldsymbol{G}_{I_t}-\boldsymbol{\mu})^T\right]) \\
&= tr(\mathbb{E}_{I_t\sim P_t}\left[\boldsymbol{G}_{I_t}\boldsymbol{G}_{I_t}^T-\boldsymbol{\mu}\boldsymbol{\mu}^T\right]) \\
&= \mathbb{E}_{I_t\sim P_t}\left[tr(\boldsymbol{G}_{I_t}\boldsymbol{G}_{I_t}^T)\right]-tr(\boldsymbol{\mu}\boldsymbol{\mu}^T) \\
&= \mathbb{E}_{I_t\sim P_t}\left[\|\boldsymbol{G}_{I_t}\|_2^2\right]-||\boldsymbol{\mu}||_2^2 \\
&= \sum_{i=1}^K p_i w_i^2\left\|\nabla_{\boldsymbol{\theta}_t}\mathcal{L}_i\right\|_2^2-||\boldsymbol{\mu}||_2^2 \\
&= \frac{\alpha^2}{K^2}\sum_{i=1}^K\frac{\mathcal{L}_i^{2\alpha-2}\left\|\nabla_{\boldsymbol{\theta}_t}\mathcal{L}_i\right\|_2^2}{p_i}-||\boldsymbol{\mu}||_2^2. \qquad (24)
\end{aligned}$$

Mathematically, given the constraints $\sum_{i=1}^K p_i = 1$, the aforementioned harmonic mean of $\{p_1,\ldots,p_K\}$ reaches its minimum when the probabilities satisfy

$$p_i \propto \mathcal{L}_i^{\alpha-1}\left\|\nabla_{\boldsymbol{\theta}_t}\mathcal{L}_i\right\|_2. \qquad (25)$$

Divided by a normalization factor, we get the expression in Equation 17. □

B SAMPLING PIPELINE

Figure 3 illustrates our sampling pipeline and model architecture. The triplet generation consists of two stages: i) matching pairs are sampled from the dataset to construct mini-batches, and ii) hard negatives are mined from the mini-batch to form triplets. Our *AdaSample* focuses on sampling informative matching pairs for mini-batch construction, and we follow the hardest-in-batch strategy in Mishchuk et al. (2017) for the second stage. We adopt the architecture of L2-Net (Tian et al., 2017) to embed patches into 128-dimensional feature space.

C IMPLEMENTATION DETAILS

We adopt the architecture of L2-Net (Tian et al., 2017) to embed local descriptors into the unit hypersphere in 128-dimensional space. Following prior works (Tian et al., 2017; Mishchuk et al., 2017), all patches are resized to 32×32 and normalized to zero per-patch mean and unit per-patch variance. We train our model from scratch in PyTorch library (Paszke et al., 2017) using SGD optimizer with initial learning rate $\eta=10$, momentum 0.5 and weight decay 0.0001. Batch size is 1024, margin $t=1$ and $\lambda=10$ unless otherwise specified. We generate $1,000,000$ matching pairs

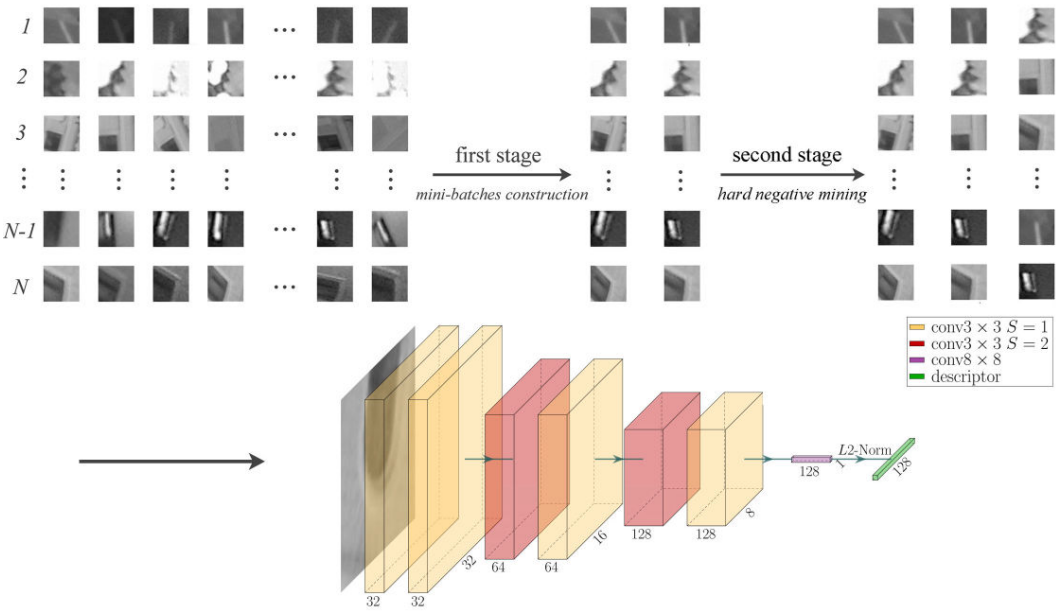

Figure 3: Training pipeline of *AdaSample* framework.

for each epoch and the total number of epochs is 90. The learning rate is divided by 10 at the end of 30, 60, 80 epochs.

We compare our method with both handcrafted and deep methods, including SIFT (Lowe, 2004), DeepDesc (Simo-Serra et al., 2015), L2-Net (Tian et al., 2017), HardNet (Mishchuk et al., 2017), HardNet with global orthogonal regularization (GOR) (Zhang et al., 2017), DOAP (He et al., 2018), and GeoDesc (Luo et al., 2018).

Note that the training set of GeoDesc is not released, and the comparisons with it is unfair. Besides, we argue that even more powerful descriptor learning framework can be established by combining our method and SOSNet, as our method focuses on data sampling and batch construction, while SOSNet on regularization. However, as the training code of SOSNet is not publicly available, we leave the efficacy comparison and system combination in future work.

D Results on ETH dataset

We evaluate the proposed descriptor on ETH SfM benchmark (Schonberger et al., 2017) to investigate its ability to reconstruct 3D models. We compare with both handcrafted algorithms (SIFT (Lowe, 2004) and DSP-SIFT (Dong & Soatto, 2015)) and deep methods (L2-Net (Tian et al., 2017) and GeoDesc (Luo et al., 2018)) as strong baselines. Following the standard protocols in Schonberger et al. (2017), we show results in Table 3, where **Reproj. Error** indicates the reconstruction accuracy, **Track Length** also indicates the quality of the reconstructed points, and others stand for reconstruction density.

It is difficult to say which one (density or accuracy) is more important in 3D scene reconstruction. Here, the results show that our method results in the best accuracy in reconstruction, and the reconstructed points have better track lengths. This is especially clear in large dataset (the last two rows). Although the reconstruction density of our approach is lower than other methods, we argue that our method is as robust as them. For example, compared with SIFT (Lowe, 2004), which is the most widely used descriptor in SfM problem, the **registered** of our method is consistently equal or higher.

Note that the density metrics should be considered with the accuracy. This is because there is no guarantee that the reconstructed points (observations, and registered images) are correct. Here, we find that other deep methods (L2-Net (Tian et al., 2017) and GeoDesc (Luo et al., 2018)) show higher

Table 3: SfM results on ETH dataset (Schonberger et al., 2017). Compared with handcrafted methods (SIFT (Lowe, 2004) and DSP-SIFT (Dong & Soatto, 2015)), other deep descriptors (L2-Net (Tian et al., 2017) and GeoDesc (Luo et al., 2018)) suffer in reconstruction accuracy (**Reproj. Error**), although they get higher density. Here we show that deep descriptors (our method) can achieve better accuracy of reconstruction against traditional methods.

		# Image	# Registered	# Sparse Points	# Observations	Track Length	Reproj. Error
Fountain	SIFT	11	11	14K	70K	4.79	0.39px
	DSP-SIFT		11	14K	71K	4.78	0.37px
	L2-Net		11	**17K**	**83K**	4.88	0.47px
	GeoDesc		11	16K	**83K**	**5.00**	0.47px
	AdaSample		11	14K	68K	4.78	**0.38px**
Herzjesu	SIFT	8	8	7.5K	31K	4.22	0.43px
	DSP-SIFT		8	7.7K	32K	4.22	0.45px
	L2-Net		8	**9.5K**	**40K**	4.24	0.51px
	GeoDesc		8	9.2K	**40K**	**4.35**	0.51px
	AdaSample		8	7.1K	30K	4.20	**0.41px**
South Building	SIFT	128	128	108K	653K	6.04	0.54px
	DSP-SIFT		128	112K	666K	5.91	0.58px
	L2-Net		128	**170K**	863K	5.07	0.63px
	GeoDesc		128	**170K**	**887K**	5.21	0.64px
	AdaSample		128	95K	603K	**6.32**	**0.52px**
Madrid Metropolis	SIFT	1344	500	116K	733K	6.32	0.60px
	DSP-SIFT		467	99K	649K	6.52	0.66px
	L2-Net		692	254K	1067K	4.20	0.69px
	GeoDesc		**809**	**306K**	**1200K**	3.91	0.66px
	AdaSample		500	75K	550K	**7.33**	**0.57px**
Gendarmenmarkt	SIFT	1463	1035	338K	1872K	5.95	0.69px
	DSP-SIFT		979	293K	1577K	5.38	0.74px
	L2-Net		1168	667K	2611K	3.91	0.73px
	GeoDesc		**1208**	**779K**	**2903K**	3.72	0.74px
	AdaSample		1051	209K	1430K	**6.83**	**0.67px**

density than traditional methods (SIFT (Lowe, 2004) and DSP-SIFT (Dong & Soatto, 2015)), but suffer in accuracy. It is indeed arguable that which type of methods are better. Here, compared with traditional methods, our method can achieve higher accuracy. This has important implications to deep learning based descriptors for high-precision 3D reconstruction.

E Comparison of Convergence Speed

Figure 4 shows the training trend of our method and HardNet (Mishchuk et al., 2017). Models are all trained and tested on HPatches (Balntas et al., 2017) using split 'a'. By incorporating *AdaSample* method, our model enjoys significantly better convergence speed.

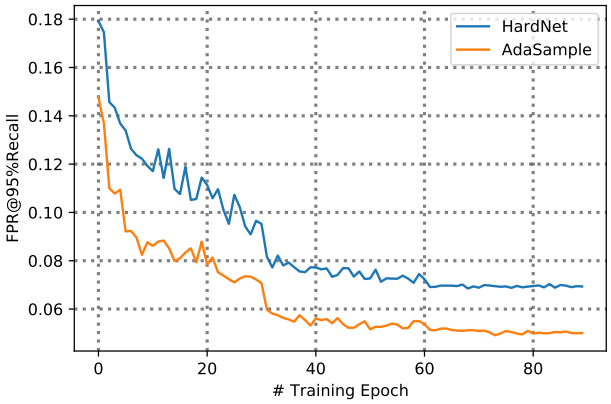

Figure 4: Traning trend of *AdaSample* and HardNet (Mishchuk et al., 2017) on HPatches (Balntas et al., 2017) dataset.