An Adversarial Approach to Hard Triplet Generation

Yiru Zhao[1,2⋆], Zhongming Jin[2], Guo-jun Qi[3,2],
Hongtao Lu[1⋆⋆], Xian-sheng Hua[2⋆⋆]

[1]Key Laboratory of Shanghai Education Commission for Intelligent Interaction and Cognitive Engineering, Shanghai Jiao Tong University
[2]Alibaba Damo Academy, Alibaba Group
[3]Laboratory for MAchine Perception and LEarning, University of Central Florida
{yiru.zhao,htlu}@sjtu.edu.cn, guojun.qi@ucf.edu,
{zhongming.jinzm,xiansheng.hxs}@alibaba-inc.com,

Abstract. While deep neural networks have demonstrated competitive results for many visual recognition and image retrieval tasks, the major challenge lies in distinguishing similar images from different categories (i.e., hard negative examples) while clustering images with large variations from the same category (i.e., hard positive examples). The current state-of-the-art is to *mine* the most hard triplet examples from the mini-batch to train the network. However, mining-based methods tend to look into these triplets that are hard in terms of the current estimated network, rather than deliberately generating those hard triplets that really matter in globally optimizing the network. For this purpose, we propose an adversarial network for Hard Triplet Generation (HTG) to optimize the network ability in distinguishing similar examples of different categories as well as grouping varied examples of the same categories. We evaluate our method on the real-world challenging datasets, such as CUB200-2011, CARS196, DeepFashion and VehicleID datasets, and show that our method outperforms the state-of-the-art methods significantly.

Keywords: image retrieval, hard examples, adversarial nets

1 Introduction

Deep metric learning is of great practical importance and has shown promising results in many tasks, such as image retrieval [1–4], face recognition [5–7], person re-identification [8–10], etc. In spite of various forms of deep metric learning in different tasks, it shares a common goal of learning an optimal image representation that pulls semantically similar images close to each other while pushing dissimilar ones apart in a learned feature space.

⋆ This work was done when the author was visiting Alibaba as a research intern.
⋆⋆ Corresponding author.

Deep metric learning often considers images in triplets as its training units inside a mini-batch. A triplet contains a query along with a relevant and an irrelevant example. Then a deep metric learning algorithm seeks to push the relevant (irrelevant) example towards (away from) the query in the underlying embedding space. It is obvious that randomly choosing triplets can be very inefficient to train a deep embedding network, as not all triplets are equally informative [3]. Some triplets contain harder examples that cannot be well handled by the current embedding network, where the irrelevant example is closer to the query than the relevant counterpart. Using these harder triplets for training can not only result in faster convergence of the learning algorithm [4], but also better improve the global structure of the embedding space by learning sharper boundaries between relevant and irrelevant examples for given queries [2]. This leads to several recent works to *mine* hard examples for training [11, 12].

However, mining-based methods aim to find those triplets from existing training examples that are hard in terms of the current network. This is essentially a greedy algorithm, which could make the trained feature embedding network vulnerable to a bad local optimum [3, 1]. In this paper, we seek an approach that can deliberately generate hard triplets to optimize the network globally, instead of using a greedy strategy to explore existing samples only for the current network. The objective of generating hard triplets should also be integrated with the objective of learning an optimal feature embedding network for different tasks. For this purpose, we propose an adversarial learning algorithm, where a hard triplet generator and an embedding network are jointly optimized in an adversarial fashion to mutually benefit each other. Our overarching goal is to learn an optimal embedding of images that is adequate in distinguishing between the relevant and the irrelevant examples even for the most challenging queries in the hardest triplets.

2 Related Works

2.1 Metric Learning

The goal of distance metric learning is to learn an embedding representation such that similar samples are mapped to nearby points on a manifold and dissimilar ones are mapped apart from each other [13]. Thanks to the success of deep neural networks(DNNs) [14–16], deep metric learning has shown great superiority on many visual recognition tasks [1, 3, 5, 6, 8, 17, 18].

In a standard deep metric learning network, a DNN model f is trained to embed an input image x into a new representation $f(x)$ by minimizing triplet loss [19, 20] that showed better performance than contrastive loss [21]. However, triplet loss is quite sensitive to a proper selection of triplets and often suffers from slow convergence and poor local optima. To address this, [22] proposed a coupled clusters loss to make the training more stable and achieved higher accuracy. [1] proposed a lifted structured embedding where each positive pair compared its distance against those of all negative pairs weighted by the margin

constraint. [4] presented multi-class N-pairs loss to improve the triplet loss by pushing away multiple negative examples simultaneously at each iteration.

Indeed, not all triplets are equally informative to train a model. Hence mining hard triplet examples plays a very important role to effectively train deep metric networks [12, 11]. The mining-based method is often performed by sampling hard triplets from existing training examples in a mini-batch. These hard triplets reflect the cases that cannot be well handled by the current model. Thus it is essentially a greedy algorithm that could be vulnerable to bad local minimum [1]. [3] presented a family of models with different levels of complexities in a cascaded manner to mine hard examples adaptively. These methods select hard examples only based on the absolute distances and are sensitive to the threshold set manually.

Inspired by the development of generative adversarial networks (GANs) [23], we propose to learn an adversarial network that can deliberately generate hard triplets in a principled fashion to improve the feature embedding network, instead of greedily mining from existing training examples.

2.2 Generative Adversarial Networks

Recently, generative adversarial networks (GANs) [23] have shown very promising results for many generative tasks such as image generation [24, 25] and translation [26–28]. More important, the adversarial training and its ability of modeling data distributions have been utilized to improve many discriminative tasks as well.

For example, [29] combined neural network classifiers with an adversarial generative model to regularize a discriminatively trained classifiers, which yields classification performance compared with state-of-the-art results for semi-supervised learning. [30] proposed Perceptual GANs that generate super-resolution representations for small objects to boost detection accuracy by leveraging iteratively updated generator networks and discriminator networks. [31] presented a Lipschitz regularized GAN to explore the margin between different classes of examples and their generated counterparts in both supervised and semi-supervised settings, and it is extended to a loss-sensitive learning framework by pull-backing original full-dimensional data onto a latent manifold representation to explore the distribution of both labeled and unlabeled samples [32]. [33] instead localized GAN-based parameterization of data manifolds so that the Laplace-Beltrami operator for semi-supervised learning can be accurately formalized without resorting to Graph Laplacian approximation. [34] learned an adversarial network by generating examples with occlusions and deformations to challenge original object detectors. Such an adversarial learning strategy significantly boosts detection performance. In contrast, in this paper, we propose a generative network in the feature embedding space to produce challenging triplets by pulling negative pairs closer and pushing positive pairs apart. Through these generated hard triplets, we wish to boost the performance of the associated feature embedding network so that it can correctly retrieve relevant examples even in adversarial cases.

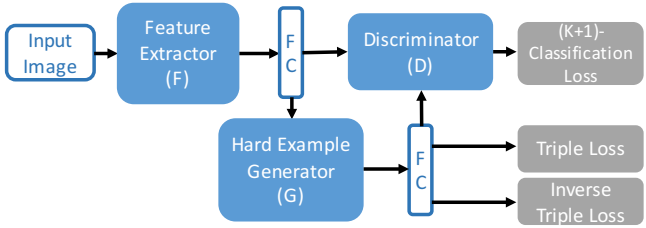

Fig. 1. Architecture of the adversarial training method.

3 Hard Triplet Generation

Our goal is to learn a feature embedding network to extract features from images. The obtained features ought to be resilient against inter-class similarity (i.e., hard negative examples) as well as intra-class variance (i.e., hard positive examples). In contrast to existing mining-based methods that merely rely on existing examples, we present an alternative approach by generating hard triplets to challenge the ability of feature embedding network correctly distinguishing relevant examples from irrelevant counterparts in these triplets. The architecture of the proposed method is shown in Figure 1.

Formally, we denote the feature embedding network by F, whose output for an input image x is $F(x) \in \mathbb{R}^L$. Given a triplet $< a, p, n >$, $< a, p >$ is a positive (relevant) pair and $< a, n >$ is a negative (irrelevant) pair. The original objective function to train F is to minimize the following triplet loss,

$$\mathcal{L}'_{F,tri} = [d(F(a), F(p)) - d(F(a), F(n)) + m]_+ \tag{1}$$

where $d(x_1, x_2) = \| \frac{x_1}{\|x_1\|} - \frac{x_2}{\|x_2\|} \|^2$ is the squared Euclidean distance between two $l2$-normalized feature vectors, and $[\cdot]_+ \triangleq \max(\cdot, 0)$ takes the positive component of its input. Then, the network is trained to find an embedding where the distance between the negative pair ought to be larger than the distance between the positive pair by at least a margin m.

3.1 Adversarial Triplet Generator G

Now let us consider a hard example generator G that generates a new adversarial sample $G(F(x)) \in \mathbb{R}^L$ by manipulating the feature representation $F(x)$ of an input x. Specifically, G learns to produce challenging triplets of examples by pushing apart the vectors from the same category while pulling closer the vectors from different categories.

Formally, We can minimize the following adversarial triplet loss to train G,

$$\mathcal{L}_{G,tri} = [d(G(F(a)), G(F(n))) - d(G(F(a)), G(F(p))) + m]_+ \tag{2}$$

Then, given a fixed G, the objective function to train F in our method becomes

$$\mathcal{L}_{F,tri} = [d(G(F(a)), G(F(p))) - d(G(F(a)), G(F(n))) + m]_+ \tag{3}$$

Clearly, $\mathcal{L}_{F,tri}$ and $\mathcal{L}_{G,tri}$ constitute an adversarial loss pair. Compared with the original training loss (1), F is trained through hard triplets of examples generated by G by pulling the positive pair closer and pushing the negative pair apart to meet the margin m.

3.2 Multi-category Discriminator D

However, the above adversarial mechanism alone is insufficient to train a reliable G because it could arbitrarily manipulate the feature representation F with no suitable constraints. For example, G could simply output random vectors to achieve a lower value of $\mathcal{L}_{G,tri}$, which was useless for training a better F. Thus, to properly constrain the triplet generator G, we require its output features should not change the label of its input features $F(x)$.

Consider a discriminator D. Given a feature vector, D categories it into $(K+1)$ categories, where the first K categories represent real classes of examples and the last one denotes a *fake* class. For a triplet $< a, p, n >$ and their labels $< l_a, l_p, l_n >$, we have $l_a = l_p$ for the positive pair and $l_a \neq l_n$ for the negative pair. Then, we minimize the following loss function to train D

$$\mathcal{L}_D = \mathcal{L}_{D,real} + \beta \mathcal{L}_{D,fake} \tag{4}$$

Here, the first term enforces D to correctly classify the feature vectors in the triplet,

$$\mathcal{L}_{D,real} = \frac{1}{3}(\mathcal{L}_{sm}(D(F(a)), l_a) + \mathcal{L}_{sm}(D(F(p)), l_p) + \mathcal{L}_{sm}(D(F(n)), l_n)) \tag{5}$$

where $\mathcal{L}_{sm}$ denotes the softmax loss. Meanwhile, the second term enables D to distinguish generated features from real ones,

$$\begin{aligned} \mathcal{L}_{D,fake} = \frac{1}{3}(&\mathcal{L}_{sm}(D(G(F(a))), l_{fake}) \\ &+\mathcal{L}_{sm}(D(G(F(p))), l_{fake}) + \mathcal{L}_{sm}(D(G(F(n))), l_{fake})) \end{aligned} \tag{6}$$

where l_{fake} represents the fake class.

Once we have D, as aforementioned, G ought to preserve the label of its input features. Thus we have the following loss to enforce this label preservation assumption,

$$\begin{aligned} \mathcal{L}_{G,cls} = \frac{1}{3}(&\mathcal{L}_{sm}(D(G(F(a))), l_a) \\ &+\mathcal{L}_{sm}(D(G(F(p))), l_p) + \mathcal{L}_{sm}(D(G(F(n))), l_n)) \end{aligned} \tag{7}$$

Putting together with (2), now we will minimize the following loss to train the hard triplet generator G,

$$\mathcal{L}_G = \mathcal{L}_{G,tri} + \gamma \mathcal{L}_{G,cls} \tag{8}$$

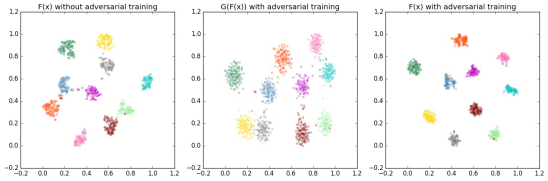

Fig. 2. Visualization of feature embedding on MNIST dataset.

3.3 Summary

Figure 1 illustrates the architecture of the proposed adversarial method. In summary, the proposed method contains an alternate optimization problem to train

i) the feature embedding network F by minimizing the triplet loss (3) combined with the classification loss (5):

$$\mathcal{L}_{F,tri} + \mu\mathcal{L}_{D,real} \tag{9}$$

where the classification loss ensures the learned features F can correctly classify different categories of real examples.

ii) the discriminator D in (4), and

iii) the hard triplet generator G in (8).

It is not hard to see that G and F form a pair of adversarial players who compete to learn feature representation resilient against hard triplets. On the other hand, D and G is another adversarial pair playing the similar role as the discriminator and generator in the classic GAN, except that G is trained to preserve labels of given examples.

Fig 2 visualizes the feature space with and without adversarial training on MNIST dataset by t-SNE [35]. It demonstrates that G pushes $F(x)$ away from the middle of the cluster, while F learns a tighter distribution of feature with the effect of G. More details will be further discussed in the next section.

4 Algorithm Details

We will discuss more details about our method in this section. First, a basic model that jointly minimizes the softmax loss and the triplet loss is presented. Then the adversarial triplet generator is added to the basic model. Finally, we also elaborate on the network details.

4.1 Basic Model

The basic model of learning the feature embedding network is illustrated in Figure 3. The output (embedded) feature layer is followed by one fully-connected layer for the softmax loss and one l_2-normalization layer for the similarity loss.

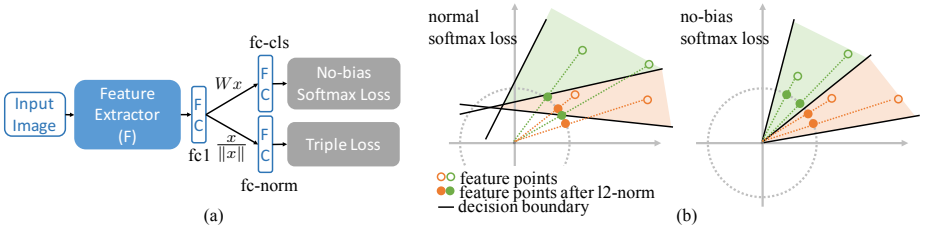

Fig. 3. (a) Architecture of the basic model with joint no-bias softmax loss and triplet loss. (b) Illustration of the feature space in 2-dimensional with conventional softmax loss and no-bias softmax loss. Hollow dots denote the feature points and filled dots denote the points projected to the unit circle after l2-norm. The solid line represent the decision boundary of softmax. The embedded feature points with no-bias softmax loss have longer inter-class distance than those with conventional softmax loss.

The softmax loss was combined with similarity loss in several previous works [36, 6, 37], but the relationship between softmax loss and similarity loss has not been well studied.

In the feature embedding space, all the data points of the same class ought to be grouped together on the unit hyper sphere after l_2-normalization. The decision boundaries between different classes divide the feature space for K classes and this global structure helps accelerate convergence and achieve optimal results. However, the conventional softmax loss is not naturally compatible with distance-based similarity loss. See the left of Figure 3(b) for example. The decision boundaries by minimizing conventional softmax loss may not pass through the origin due to the existence of a bias b. Thus, the data points from different classes may overlap each other after l_2-normalization. This results in a short inter-class distance that impacts the performance of feature embedding. Hence, we propose to use softmax loss without a bias. As shown in the right of Figure 3(b), all the decision boundaries from such no-bias softmax loss pass through the origin and the decision area for a class is cone-shaped with its vertex located at the origin. Thus, a class of examples have a separate projection on the unit hyper sphere, which ensures a long inter-class distance between examples from different classes.

Given a training tuple $< a, p, n, l >$, where the anchor image a is labeled as class l, the no-bias softmax loss is defined as

$$\mathcal{L}_{F,cls} = -\log \frac{e^{W_l F(a)}}{\sum_{k=1}^{K} e^{W_k F(a)}} \tag{10}$$

where $F(\cdot)$ denotes the output feature of a CNN model. Then the network F is trained by minimizing $\mathcal{L}_F = \mathcal{L}_{F,cls} + \lambda \mathcal{L}'_{F,tri}$ by stochastic gradient descent, and λ is the weight to control the trade-off between the no-bias softmax loss and the original triplet loss (1).

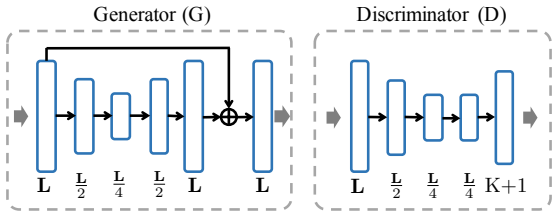

Fig. 4. Architecture of the Generator(G) and Discriminator(D).

4.2 Adversarial Training

The above basic model trains a benchmark feature extractor F, and here we will show that it can benefit from the hard triplets generated by an adversarial generator. In the basic model, F is trained by randomly sampling triplets from a training set without considering their hardness. Now we attempt to train a hard triplet generator at the feature level.

As shown in the left dotted box in Figure 4, the generator denoted by G takes an input feature vector $F(x) \in \mathbb{R}^L$ output from F and produces a generated feature vector of the same dimension. The generator G consists of 4 fully-connected layers, first two of which reduce the input feature dimension by half as a bottleneck and then the last two layers redouble it to the original dimension. Each layer is followed with BatchNormalization [38] and ReLU [39], and the output vector is calculated by performing an element-wise addition between the output fully-connected layer and the input vector. Such residual structure enforces G to learn the offset of the input vector, which yields faster convergence of G in the training phase. In this fashion, G only needs to learn how to move a feature vector in the feature space instead of generating features from scratch.

On the other hand, the discriminator D also takes an input feature vector $F(x) \in \mathbb{R}^L$ from F and classifies it into one of $K+1$ classes. D also has four fully-connected layers, where the first three are followed with BatchNormalization and ReLU and the last one is a softmax layer.

We use mini-batch SGD with the learning rate α to train the networks F, D, and G step by step with the loss functions introduced in Section 3.

i) Update the feature embedding network F with $\mathcal{L}_F = \mathcal{L}_{F,tri} + \mu\mathcal{L}_{D,real}$, where F is trained by ensuring the distance of a positive pair should be smaller than that of the negative pair by at least a margin m in a hard triplet generated by G; meanwhile, all feature vectors of F should be correctly classified by D.

ii) Update the discriminator D with $\mathcal{L}_D = \mathcal{L}_{D,real} + \beta\mathcal{L}_{D,fake}$, where D is learned by distinguishing real vectors F from the fake counterparts generated by G, and $\mathcal{L}_{D,real}$ is also used to train D to classify the labeled examples.

iii) Update the hard triplet generator G with $\mathcal{L}_G = \mathcal{L}_{G,tri} + \gamma\mathcal{L}_{G,cls}$, where G is trained to produce hard triplets that can challenge F while it is regularized by preserving the labels of training examples.

4.3 Harder Triplet Generation from Local Details

Furthermore, we attempt to build a more powerful extractor F allowing the HTG to create harder triplets from fine-grained local details, thus a visual recogntion model can be challenged with harder triplet examples to become more robust.

Indeed, local features play a critical role in many fine-grained visual recognition tasks. Typical deep neural networks designed for image classification are good at extracting high-level global features, but the features of local details are often missing. This could limit the HTG in exploring local details to create harder triplets. For example, without local details, the HTG could not generate such hard triplets that could force a model to focus on most discriminative parts such as logos, lights and sunroof in identifying different cars.

To address this problem, we introduce keypoint maps for training HTG by focusing on local details. For example ResNet-18 consists of four sequential convolution blocks, and the output of the fully-connected (FC) layer following the last convolution block is used as the global feature f_{global}. The output feature map of a convolution block-l is denoted by $X^l \in \mathbb{R}^{C \times W \times H}$. Then we add a local branch called keypoint block, which has an architecture similar to a convolution block, to localize the distribution of keypoints that could focus on the most discriminative parts to create harder triplets. The high-level semantic feature maps are sparse and we assume that each channel of a keypoint layer corresponds to a particular type of keypoint, thus we apply channel-wise softmax on the output feature map of the keypoint layer to estimate the density of a keypoint over different image locations:

$$M^l_{cij} = \frac{e^{P^l_{cij}}}{\sum_{w=1}^{W} \sum_{h=1}^{H} e^{P^l_{cwh}}} \tag{11}$$

with P^l_{cij} being the output feature of channel c at (i, j) in keypoint block-l.

This softmax output is used as a channel-wise keypoint mask, which allows us to perform element-wise product of X^l and M^l. The resulting local feature f^l of block-l is calculated by a channel-wise summation over locations: $f^l_c = \sum_{i=1}^{W} \sum_{j=1}^{H} X^l_{cij} M^l_{cij}$.

In experiments, we will extract such local features at block-3 and block-4, which are then concatenated with the global feature to form the final output feature $f_{out} = [f_{global}; f^3; f^4]$.

5 Experiments

We evaluate the proposed method on four real-world datasets in two performance metrics, i.e., Recall@K [40] and mAP. For the network architecture, we choose ResNet-18 [41] pre-trained on ImageNet ILSVRC-2012 [42]. The keypoint blocks described in Section 4.3 are initialized with the same weights as the ResNet-18 convolution blocks. We use the same hyper parameters in all experiments without tuning them. Input images are first resized to 256×256 and cropped

at 224×224. For the data augmentation, we use random crops with random horizontal mirroring for training and a single center crop for testing.

For training basic model in Section 4.1, mini-batch size is 128, triplet margin is $m = 0.1$, and $\lambda = 1$. The learning rate α starts from 0.01 and is divided by 10 every 5 epochs to train the model for 15 epochs.

For training adversarial triplet generator in Section 4.2, mini-batch size is 64, and $\mu = \beta = \gamma = 1$. The generator network is trained for 10 epochs with the learning rate α being initialized to 0.001 and divided by 10 every 5 epochs.

5.1 Datasets

We use four datasets in our experiments which are commonly used in many fine-grained visual recognition tasks. We follow the standard experiment protocol to split train and test sets for a fair comparison with existing methods.

- **CUB200** [43] dataset has 200 classes of birds with 11,788 images, in which the first 100 classes (5,864 images) are used for training and the rest 100 classes (5,924 images) are used for testing. Both query and gallery sets are from the test set.
- **CARS196** [44] dataset has 196 classes of cars with 16,185 images, where the first 98 classes (8,054 images) are used for training and the rest 98 classes (8,131 images) are used for testing. Also, both query and gallery sets are from the test set.
- **In-Shop Clothes Retrieval** is one of the three benchmarks used in [45], which has 7,982 classes of clothes with 52,712 images. Among them, 3,997 classes (25,882 images) are used for training and the other 3,985 classes (28,760 images) are used for testing. The test images are partitioned to a query set and a gallery set. The query set contains 14,218 images and the gallery set has 12,612 images.
- **VehicleID** [22] dataset contains 221,763 images of 26,267 vehicles, where the training set contains 110,178 images of 13,134 vehicles and the test set contains 111,585 images of 13,133 vehicles. It is more challenging than CARS196 because different identities of vehicles are considered as different classes, even though they share the same car model. Following the protocol in [22], there are three test sets of different sizes. The smallest test set contains 7,332 images of 800 vehicles. The medium test set contains 12,995 images of 1600 vehicles. The largest test set contains 20,038 images of 2,400 vehicles.

5.2 Generation-based Method v.s. Mining-based Method

We start by demonstrating the improvements of our proposed HTG method. We focus on the CARS196 dataset and evaluate models with different settings on training strategy. The network architecture is ResNet-18 without any additional branch. Results are presented in Table 1. We compare random sampling and online hard example mining (OHEM) [12] to our hard triplet generation(HTG). For

training OHEM, we sample 32 identities with 4 images for each identity to form a mini-batch, and mine the nearest negative sample and longest positive sample for each anchor data to constitute a triplet. The experimental results show that OHEM improves the recall scores by learning from hard triplets and our HTG method outperforms OHEM further with an absolute 2.4% improvement on Recall@1 score. The results of A/D, B/E and C/F prove that removing the bias of softmax loss offers notable boost on varied training strategies, which means the no-bias softmax loss is more compatible with distance-based similarity loss.

Table 1. Recall@K(%) on CARS196 with different model settings

model	training	softmax	1	2	4	8
A	random	bias	65.4	76.5	84.7	91.0
B	OHEM	bias	67.1	78.1	86.2	91.7
C	HTG	bias	69.3	79.2	86.7	92.0
D	random	no-bias	66.6	77.0	85.2	91.3
E	OHEM	no-bias	68.2	78.7	86.5	92.0
F	HTG	no-bias	**70.6**	**79.9**	**87.3**	**92.9**

5.3 Hard Triplet Generation from Local Attentions

In the last section, we showed the HTG with global ResNet-18 features outperformed the other compared methods. We also consider to add a local keypoint branch to maximize the ability of the HTG in generating hard triplets. This could allow the HTG to explore local details so that harder triplets with fine-grained details can be produced to further imporve the recognition accuracy. Below we will demonstrate more competitive performance can be achieved by the HTG with various local attentions models.

Table 2. Recall@1(%) score on CARS196

	vanilla	K-Branch	ResAttention	KPM
random	66.6	71.9	72.3	72.7
OHEM	68.2	72.2	72.6	72.5
HTG	70.6	72.7	73.9	76.5

To this end, we compare the global vanilla model without any modification, K-branch [10] and ResAttention [46] with our keypoint maps (KPM). These methods are all based on ResNet-18 backbone for fair comparation. K-branch designs 8 branch to detect discriminative regions and align local parts. ResAttention extracts local feature with attention-aware masks. All the architectures are

Fig. 5. Visualization of keypoint maps on VehicleID, CUB200 and In-Shop Clothes datasets respectively.

tested with three training strategies and the results are shown in Table 2. The attention models outperform the base model with different training strategies.

It is worth nothing that on attention models, OHEM does not significantly outperform random sampling of triplets as on the vanilla model. We attribute this performance degeneration to the bad local optimum of OHEM. On the contrary, the proposed HTG successfully boosts the performance on attention models, especially on KPM model. This demonstrates the competency of HTG in exploring local details to generate harder triplet examples for training more competitive recognition models.

In order to demonstrate that KPM is able to find keypoints, we intuitively illustrate detected keypoints by taking the maximum response over C channels at different locations:

$$V_{i,j} = \max_{c \in 1,\cdots,C} \{M_{c,i,j}\} \tag{12}$$

The keypoint maps are superimposed on their input images in Figure 5. On VehicleID dataset, there are high responses on logo, lights, sunroof and antenna on the car roof. These are the most discriminative parts to classify car models. To tell the differences between distinct vehicles of the same model, the model further localizes more customized landmarks, such as the stickers on the car window and the small objects put on the car dashboard. It shows that this model is able to find these subtle keypoints effectively.

Local features play an important role in fine-grained image recognition tasks. Although the local branch of keypoint blocks does not need any manual annotations, it is expected to localize discriminative parts by learning from the generated hard triplets to distinguish subtle differences between fine-grained image classes.

On CUB200 dataset, the detected keypoints often locate in eyes, beaks, wings and tails, which are useful to classify different species of birds. On In-Shop Clothes dataset, collar ends, sleeve ends and hem parts are detected. These ex-

amples demonstrate that the proposed model trained from the generated hard triplets successfully localizes the most discriminative part in classifying or identifying images at a very fine-grained level.

5.4 Comparison with the State-of-the-art Methods

On the CUB200 and CARS196 datasets, the proposed model is compared with five state-of-the-art methods. LiftedStruct [1] uses an algorithm taking full advantage of the training batches by lifting the vector of pairwise distances within the batch to the matrix of pairwise distance. StructuredCluster [2] uses structured prediction that is aware of the global structure of an embedding space. SmartMining [47] proposes an adaptive controller that automatically adjusts the smart mining hyper-parameters and speeds up the convergence. N-Pair [4] proposes multi-class N-pair loss to leverage more than one negative examples for each anchor. HDC [3] ensembles a set of models with increasing complexities in a cascaded manner to mine hard samples at different levels. Table 3 shows the Recall@K results on these two datasets. Our proposed method successfully boosts the Recall@K upon the existing methods and gets an absolute 5.9% improvement on Recall@1 compared with HDC on CUB200 dataset, and 2.8% improvement on CARS196 dataset.

Table 3. Recall@K(%) on CUB200 and CARS196

dataset	CUB200				CARS196			
Recall@K	1	2	4	8	1	2	4	8
LiftedStruct [1]	43.6	56.6	68.6	79.6	53.0	65.7	76.0	84.3
StructuredCluster [2]	48.2	61.4	71.8	81.9	58.1	70.6	80.3	87.8
SmartMining [47]	49.8	62.3	74.1	83.3	64.7	76.2	84.2	90.2
N-Pair [4]	51.0	63.3	74.3	83.2	71.1	79.7	86.5	91.6
HDC [3]	53.6	65.7	77.0	85.6	73.7	83.2	89.5	93.8
Ours	**59.5**	**71.8**	**81.3**	**88.2**	**76.5**	**84.7**	**90.4**	**94.0**

On In-Shop Clothes dataset, FashionNet [45] is a novel deep model that learns clothing features by jointly predicting landmark locations and massive attributes. HDC is also included in comparison on this dataset. Table 4 reports the results. Compared methods suffer from the problem with a large number of classes and limited images in each class. The proposed method significantly improves the Recall@1 score from 62.1% to 80.3%. It is worth noting that the proposed method does not use any manually annotated landmarks and attributes. FashionNet did not report the numeric results on this task, and the results of FashionNet are referred to [3].

On VehicleID dataset, Mixed Diff + CCL [22] uses coupled cluster loss and the mixed difference network structure. Table 5 reports the results on VehicleID. This dataset contains many hard negative examples that are different

Table 4. Recall@K(%) on In-Shop Clothes Retrieval, FN is short for FashionNet.

K	1	10	20	30	40
FN+Joints [45]	41.0	64.0	68.0	71.0	73.0
FN+Poselets [45]	42.0	65.0	70.0	72.0	72.0
FN [45]	53.0	73.0	76.0	77.0	79.0
HDC [3]	62.1	84.9	89.0	91.2	92.3
Ours	**80.3**	**93.9**	**95.8**	**96.6**	**97.1**

vehicles with the same model, and it is an ideal example showing the advantage of learning from hard triplets. Compared with HDC, our model achieves an absolute 7.7%/7.5%/9.0% improvement on Small/Medium/Large test set in mAP, respectively. The results on these four datasets show the proposed method outperforms the existing state-of-the-art methods.

Table 5. mAP(%) on VehicleID

Method	Small	Medium	Large
VGG+Triple Loss [22]	44.4	39.1	37.1
VGG+CCL [22]	49.2	44.8	38.6
Mixed Diff + CCL [22]	54.6	48.1	45.5
HDC [3]	65.5	63.1	57.5
Ours	**73.2**	**70.6**	**66.5**

6 Conclusion

In this paper, we propose a novel algorithm, Hard Triplet Generation via Adversarial training for learning an optimal embedding of images. A feature extractor is pushed to distinguish relevant examples from irrelevant ones even for the most challenging queries in the generated hard triplets. This generation-based method avoids the problem of being trapped to a bad local optimum by a greedy mining-based method. Experimental results on four real-world datasets demonstrate the advantage of the proposed model over the compared state-of-the-art methods.

Acknowledgements. This paper is partially supported by NSFC (No. 61772330, 61533012, 61472075), the Basic Research Project of Innovation Action Plan (16JC1402800), the Major Basic Research Program (15JC1400103) of Shanghai Science and Technology Committee, NSF under award #1704309 and IARPA under grant #D17PC00345.

References

1. Oh Song, H., Xiang, Y., Jegelka, S., Savarese, S.: Deep metric learning via lifted structured feature embedding. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2016) 4004–4012
2. Song, H.O., Jegelka, S., Rathod, V., Murphy, K.: Learnable structured clustering framework for deep metric learning. IEEE Conference on Computer Vision and Pattern Recognition (2017)
3. Yuan, Y., Yang, K., Zhang, C.: Hard-aware deeply cascaded embedding. ICCV (2017)
4. Sohn, K.: Improved deep metric learning with multi-class n-pair loss objective. In: Advances in Neural Information Processing Systems. (2016) 1857–1865
5. Schroff, F., Kalenichenko, D., Philbin, J.: Facenet: A unified embedding for face recognition and clustering. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2015) 815–823
6. Sun, Y., Chen, Y., Wang, X., Tang, X.: Deep learning face representation by joint identification-verification. In: Advances in neural information processing systems. (2014) 1988–1996
7. Wen, Y., Zhang, K., Li, Z., Qiao, Y.: A discriminative feature learning approach for deep face recognition. In: European Conference on Computer Vision, Springer (2016) 499–515
8. Cheng, D., Gong, Y., Zhou, S., Wang, J., Zheng, N.: Person re-identification by multi-channel parts-based cnn with improved triplet loss function. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2016) 1335–1344
9. Shi, H., Yang, Y., Zhu, X., Liao, S., Lei, Z., Zheng, W., Li, S.Z.: Embedding deep metric for person re-identification: A study against large variations. In: European Conference on Computer Vision, Springer (2016) 732–748
10. Zhao, L., Li, X., Wang, J., Zhuang, Y.: Deeply-learned part-aligned representations for person re-identification. IEEE Conference on Computer Vision and Pattern Recognition (2017)
11. Cui, Y., Zhou, F., Lin, Y., Belongie, S.: Fine-grained categorization and dataset bootstrapping using deep metric learning with humans in the loop. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2016) 1153–1162
12. Simo-Serra, E., Trulls, E., Ferraz, L., Kokkinos, I., Fua, P., Moreno-Noguer, F.: Discriminative learning of deep convolutional feature point descriptors. In: Proceedings of the IEEE International Conference on Computer Vision. (2015) 118–126
13. Xing, E.P., Jordan, M.I., Russell, S.J., Ng, A.Y.: Distance metric learning with application to clustering with side-information. In: Advances in neural information processing systems. (2003) 521–528
14. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems. (2012) 1097–1105
15. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition. (2015) 1–9
16. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)

17. Hu, J., Lu, J., Tan, Y.P.: Discriminative deep metric learning for face verification in the wild. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2014) 1875–1882
18. Yi, D., Lei, Z., Liao, S., Li, S.Z.: Deep metric learning for person re-identification. In: Pattern Recognition (ICPR), 2014 22nd International Conference on, IEEE (2014) 34–39
19. Schultz, M., Joachims, T.: Learning a distance metric from relative comparisons. In: Advances in neural information processing systems. (2004) 41–48
20. Weinberger, K.Q., Blitzer, J., Saul, L.K.: Distance metric learning for large margin nearest neighbor classification. In: Advances in neural information processing systems. (2006) 1473–1480
21. Hadsell, R., Chopra, S., LeCun, Y.: Dimensionality reduction by learning an invariant mapping. In: Computer vision and pattern recognition, 2006 IEEE computer society conference on. Volume 2., IEEE (2006) 1735–1742
22. Liu, H., Tian, Y., Yang, Y., Pang, L., Huang, T.: Deep relative distance learning: Tell the difference between similar vehicles. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2016) 2167–2175
23. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Advances in neural information processing systems. (2014) 2672–2680
24. Denton, E.L., Chintala, S., Fergus, R., et al.: Deep generative image models using a laplacian pyramid of adversarial networks. In: Advances in neural information processing systems. (2015) 1486–1494
25. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434 (2015)
26. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. IEEE Conference on Computer Vision and Pattern Recognition (2017)
27. Zhu, J.Y., Park, T., Isola, P., Efros, A.A.: Unpaired image-to-image translation using cycle-consistent adversarial networks. In: IEEE International Conference on Computer Vision. (2017)
28. Yi, Z., Zhang, H., Tan, P., Gong, M.: Dualgan: Unsupervised dual learning for image-to-image translation. In: IEEE International Conference on Computer Vision. (2017)
29. Springenberg, J.T.: Unsupervised and semi-supervised learning with categorical generative adversarial networks. In: International Conference on Learning Representations (ICLR). (2016)
30. Li, J., Liang, X., Wei, Y., Xu, T., Feng, J., Yan, S.: Perceptual generative adversarial networks for small object detection. IEEE Conference on Computer Vision and Pattern Recognition (2017)
31. Qi, G.J.: Loss-sensitive generative adversarial networks on lipschitz densities. arXiv preprint arXiv:1701.06264 (2017)
32. Edraki, M., Qi, G.J.: Generalized loss-sensitive adversarial learning with manifold margins. In: Proceedings of European Conference on Computer Vision (ECCV 2018). (2018)
33. Qi, G.J., Zhang, L., Hu, H., Edraki, M., Wang, J., Hua, X.S.: Global versus localized generative adversarial nets. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (2018)

34. Wang, X., Shrivastava, A., Gupta, A.: A-fast-rcnn: Hard positive generation via adversary for object detection. IEEE Conference on Computer Vision and Pattern Recognition (2017)
35. Maaten, L.v.d., Hinton, G.: Visualizing data using t-sne. Journal of machine learning research **9**(Nov) (2008) 2579–2605
36. Zhang, X., Zhou, F., Lin, Y., Zhang, S.: Embedding label structures for fine-grained feature representation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2016) 1114–1123
37. Parkhi, O.M., Vedaldi, A., Zisserman, A., et al.: Deep face recognition. In: BMVC. Volume 1. (2015) 6
38. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: International Conference on Machine Learning. (2015) 448–456
39. Maas, A.L., Hannun, A.Y., Ng, A.Y.: Rectifier nonlinearities improve neural network acoustic models. In: Proc. ICML. Volume 30. (2013)
40. Jegou, H., Douze, M., Schmid, C.: Product quantization for nearest neighbor search. IEEE transactions on pattern analysis and machine intelligence **33**(1) (2011) 117–128
41. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. (2016) 770–778
42. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. International Journal of Computer Vision **115**(3) (2015) 211–252
43. Wah, C., Branson, S., Welinder, P., Perona, P., Belongie, S.: The caltech-ucsd birds-200-2011 dataset. (2011)
44. Krause, J., Stark, M., Deng, J., Fei-Fei, L.: 3d object representations for fine-grained categorization. In: Proceedings of the IEEE International Conference on Computer Vision Workshops. (2013) 554–561
45. Liu, Z., Luo, P., Qiu, S., Wang, X., Tang, X.: Deepfashion: Powering robust clothes recognition and retrieval with rich annotations. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2016) 1096–1104
46. Wang, F., Jiang, M., Qian, C., Yang, S., Li, C., Zhang, H., Wang, X., Tang, X.: Residual attention network for image classification. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2017) 3156–3164
47. Harwood, B., Kumar, B., Carneiro, G., Reid, I., Drummond, T., et al.: Smart mining for deep metric learning. In: IEEE International Conference on Computer Vision. (2017)