Mining Discriminative Triplets of Patches for Fine-Grained Classification

Yaming Wang[†], Jonghyun Choi[‡], Vlad I. Morariu[†], Larry S. Davis[†]
[†]University of Maryland, College Park, MD 20742, USA
[‡]Comcast Labs DC, Washington, DC 20005, USA
{wym, jhchoi, morariu, lsd}@umiacs.umd.edu

Abstract

Fine-grained classification involves distinguishing between similar sub-categories based on subtle differences in highly localized regions; therefore, accurate localization of discriminative regions remains a major challenge. We describe a patch-based framework to address this problem. We introduce triplets of patches with geometric constraints to improve the accuracy of patch localization, and automatically mine discriminative geometrically-constrained triplets for classification. The resulting approach only requires object bounding boxes. Its effectiveness is demonstrated using four publicly available fine-grained datasets, on which it outperforms or achieves comparable performance to the state-of-the-art in classification.

1. Introduction

The task of fine-grained classification is to recognize sub-ordinate categories belonging to the same super-ordinate category [42, 39, 32, 25]. The major challenge is that fine-grained objects share similar overall appearance and only have subtle differences in highly localized regions. To effectively and accurately find these discriminative regions, some previous approaches utilize humans-in-the-loop [4, 7, 40], or require semantic part annotations [30, 2, 1, 3, 12, 47, 48] or 3D models [29, 25]. These methods are effective, but they require extra keypoint/part/3D annotations from humans, which are often expensive to obtain. On the other hand, recent research on discriminative mid-level visual elements mining [9, 36, 8, 21, 27] automatically finds discriminative patches or regions from a huge pool and uses the responses of those discriminative elements as a mid-level representation for classification. However, this approach has mainly been applied to scene classification and not typically to fine-grained classification. This is probably due to the fact that the discriminative patches needed for fine-grained categories need to be more accurately localized than for scene classification.

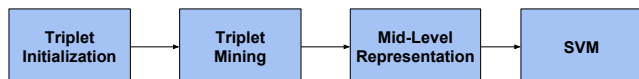To avoid the cost of extra annotations, we propose a

Figure 1. Summary of our discriminative triplet mining framework. Candidate triplets are initialized from sets of neighboring images and selected by how discriminative they are across the training set. The mid-level representation consists of the maximum responses of the selected triplets with geometric constraints, which is fed into a linear SVM for classification.

patch-based approach for fine-grained classification that overcomes the difficulties of previous discriminative mid-level approaches. Our approach requires only object bounding box annotations and belongs to the category of weakly-annotated fine-grained classification [44, 45, 16, 34, 5, 43, 28, 24].

Two issues need to be addressed. The first issue is accurately localizing discriminative patches without requiring extra annotations. Localizing a single patch based only on its appearance remains challenging due to noisy backgrounds, ambiguous repetitive patterns and pose variations. Instead, we localize *triplets* of patches with the help of geometric constraints. Previously, triple or higher-order constraints have been used in image matching and registration [10, 22, 46], but they have not been integrated into a patch-based classification framework. Our triplets consist of three appearance descriptors and two simple, but efficient, geometric constraints, which can be used to remove accidental detections that would be encountered if patches were localized individually. One attractive property of triplets over simpler models (*e.g.*, pairs) is that they can be used to model rich geometric relationships while providing additional invariance (*e.g.*, to rotation) or robustness (*e.g.*, to perspective changes).

The second issue is automatically discovering discriminative geometrically-constrained triplets from the huge pool of all possible triplets of patches. The key insight is that fine-grained objects share similar overall appearance. Therefore, if we retrieve the nearest neighbors of a training image, we obtain samples from different classes with al-

most the same pose, from which potentially discriminative regions can be found. Similar ideas have been adopted in [15, 16, 18, 23], but they aggregate results obtained from local neighborhood processing without further analysis across the whole dataset. In contrast, our discriminative triplet mining framework uses sets of overall similar images to propose potential triplets, and only selects good ones by measuring their discriminativeness using the whole training set or a large portion of it.

We evaluate our approach on four publicly available fine-grained datasets and obtain comparable results to the state-of-the-art without expensive annotation.

2. Triplets of Patches with Geometric Constraints

In this section, we discuss two geometric constraints encoded within triplets of patches and describe a triplet detector incorporating these constraints.

Suppose we have three patch templates T_A, T_B and T_C, with their centers located at points A, B and C, respectively. Each template T_i ($i \in \{A, B, C\}$) can be a feature vector extracted from a single patch or an averaged feature vector of patches from the corresponding locations of several positive samples. Given an image, let A', B' and C' be three patches possibly corresponding to A, B and C, respectively.

2.1. Order Constraint and Shape Constraint

The order constraint encodes the ordering of the three patches (Figure 2). For triplet $\{A, B, C\}$, consider the two vectors $\overrightarrow{AB}$ and $\overrightarrow{AC}$. Treating them as three dimensional vectors with the third dimension being 0, it follows that

$$\overrightarrow{AB} \times \overrightarrow{AC} = (0, 0, Z_{ABC}). \quad (1)$$

Let

$$G_{ABC} = \text{sign}(Z_{ABC}), \quad (2)$$

which indicates whether the three patches are arranged clockwise ($G_{ABC} = 1$) or counterclockwise ($G_{ABC} = -1$), as can be seen in Figure 2. There is a side-test interpretation of this constraint [14]. If we fix two patches, say B and C, $G_{ABC} = 1$ means that A lies on the left side of the line passing between B and C, while $G_{ABC} = -1$ indicates A is on the right side. Therefore, a simple penalty function between $\{A, B, C\}$ and $\{A', B', C'\}$ based on the order constraint can be defined as

$$p_o(G_{ABC}, G_{A'B'C'}) = 1 - \eta_o \mathbf{l}(G_{A'B'C'} \neq G_{ABC}), \quad (3)$$

where $0 \leq \eta_o \leq 1$ controls how important the order constraint is, and the indicator function is defined as

$$\mathbf{l}(G_{A'B'C'} \neq G_{ABC}) = \begin{cases} 1, & G_{A'B'C'} \neq G_{ABC} \\ 0, & G_{A'B'C'} = G_{ABC}. \end{cases} \quad (4)$$
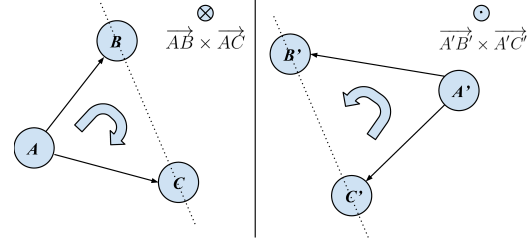

Figure 2. Visualization of the order constraint. Top: Patch A, B and C are arranged in clockwise order. The direction of $\overrightarrow{AB} \times \overrightarrow{AC}$ points into the paper, and $G_{ABC} = 1$. Bottom: A', B' and C' are arranged in counterclockwise order. The direction of the cross product points out of the paper, and $G_{A'B'C'} = -1$.
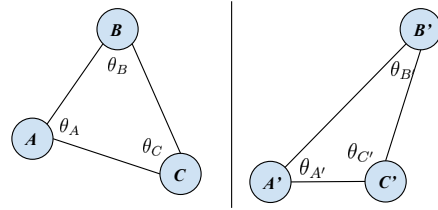

Figure 3. Visualization of the shape constraint. The constraint compares the three angles of two triplets.

Intuitively, p_o penalizes by η_o when $\{A', B', C'\}$ violates the order constraint defined by $\{A, B, C\}$.

The shape constraint measures the shape of the triangle defined by three patch centers (Figure 3). Let $\Theta_{ABC} = \{\theta_A, \theta_B, \theta_C\}$ denote the angles of triangle ABC, and $\Theta_{A'B'C'} = \{\theta_{A'}, \theta_{B'}, \theta_{C'}\}$ denote the angles of triangle $A'B'C'$, as displayed in Figure 3. We define a shape penalty function by comparing corresponding angles as

$$\begin{aligned} &p_s(\Theta_{A'B'C'}, \Theta_{ABC}) \\ &= 1 - \eta_s \frac{\sum_{i \in \{A,B,C\}} |\cos(\theta_i) - \cos(\theta_{i'})|}{6}, \end{aligned} \quad (5)$$

where $\eta_s \in [0, 1]$ controls how important the shape constraint is. The denominator 6 in Eq. (5) ensures that $0 \leq p_s \leq 1$, since $|\cos(\theta_i) - \cos(\theta_{i'})| \leq 2$. $\{\cos(\theta_i)\}$ and $\{\cos(\theta_{i'})\}$ can be easily computed from inner products. The motivation for introducing this second constraint is that we use the relatively loose order constraint to perform coarse verification and use the shape constraint for finer adjustments. As will be demonstrated in Section 4.1, the two constraints contain complementary information.

2.2. Triplet Detector

Our triplet detector consists of three appearance models and the two geometric constraints. We construct three linear weights $\{w_A, w_B, w_C\}$ from the patch templates $\{T_A, T_B, T_C\}$, and our triplet detector becomes

$$\mathbf{T} = \{w_A, w_B, w_C, G_{ABC}, \Theta_{ABC}\}. \quad (6)$$

For any triplet $\{A', B', C'\}$ with features $\{T_{A'}, T_{B'}, T_{C'}\}$ and geometric parameters $\{G_{A'B'C'}, \Theta_{A'B'C'}\}$, its detection score is defined as

$$S_{A'B'C'} = (S_A(T'_A) + S_B(T'_B) + S_C(T'_C)) \cdot p_o \cdot p_s, \quad (7)$$

where p_o, p_s are defined by Eq. (3), Eq. (5) respectively, and the appearance scores are defined as

$$S_i(T_{i'}) = w_i^{\mathrm{T}} T_{i'}, \quad i \in \{A, B, C\}. \quad (8)$$

To make triplet detection practical, three technical details must be addressed. The first is how to efficiently obtain the maximum response of a triplet detector in an image. In principle, we could examine all possible triplets from the combinations of all possible patches and simply compute

$$\{A^*, B^*, C^*\} = \underset{\{A', B', C'\}}{\mathrm{argmax}} \ S_{A'B'C'}. \quad (9)$$

However, this is too expensive since the number of all possible triplets will be $O(N^3)$ for N patches. Instead, we adopt a greedy approach. We first find the top K non-overlapping detections for each appearance detector independently. Then we evaluate the K^3 possible triplets and select the one with the maximum score defined by Eq. (7).

The second technical detail is how to obtain the linear weights w_i from a patch template T_i. For efficiency, we use the LDA model introduced by [20]

$$w_i = \Sigma^{-1} (T_i - \mu), \quad (10)$$

where μ is the mean of features from all patches in the dataset, and Σ is the corresponding covariance matrix. The LDA model is efficient since it constructs the model for negative patches (μ and Σ) only once.

Our triplet detector is able to handle moderate pose variations. However, if we flip an image, both the appearance (*e.g.* the dominant direction of an edge) and the order of the patches will change. We address this issue by applying the triplet detector to the image and its mirror, generating two detection scores, and choose the larger score as the response. This simple technique proves effective in practice.

3. Discriminative Triplets for Fine-Grained Classification

In this section, we describe how to automatically mine discriminative triplets with the geometric constraints and generate mid-level representations for classification with the mined triplets. We present the overview of our framework in Figure 1. In the triplet initialization stage, we use a nearest-neighbor approach to propose potential triplets, taking advantage of the fact that instances of fine-grained objects share similar overall appearance. Then we verify the discriminativeness of the candidate triplets using the whole

Figure 4. Some examples of a set of neighboring images. The query image is highlighted with a red box. Since fine-grained objects share similar overall appearance, the neighborhood consists of samples from different classes with the same pose.

training set or a large portion of it, and select discriminative ones according to an entropy-based measure. For classification, we concatenate the maximum responses of the selected discriminative triplets to construct mid-level image representations. The key to our approach is proposing candidate triplets locally and selecting discriminative ones globally, avoiding the insufficient data problems of other nearest-neighbor based fine-grained approaches.

3.1. Triplet Initialization

To reduce the computational burden of triplet mining, we initialize candidate triplets using potentially discriminative patches in a nearest-neighbor fashion. The overview of the procedure is displayed in Figure 5.

Construct Neighborhood. For a seed training image I_0 with class label c_0, we extract features [6] of the whole image X_0 and use it to retrieve the nearest neighbors from the training set. Since fine-grained objects have similar overall appearance, the resulting set of images consists of training images from different classes with almost the same pose (Figure 4). This first step results in a set of roughly aligned images, so that potentially discriminative regions can be found by comparing corresponding regions across the images. We refer to the set consisting of a seed training image and its nearest neighbors as a neighborhood.

Find Candidate Regions. We regard each neighborhood as a stack of aligned images with their class labels and locate potentially discriminative regions. Consider the set of patches at the same location of each image. For each location (x, y), let $F_i(x, y)$ be the features extracted from the patch in the ith image and let c_i be its label. Denote the set of observed class labels as C. The discriminative score at (x, y) is simply defined as the ratio of between-class variation and in-class variation, *i.e.*

$$d(x, y) = \frac{\sum_{c \in C} \left\| \overline{F_c}(x, y) - \overline{F}(x, y) \right\|^2}{\sum_{c \in C} \sum_{c_i = c} \left\| F_i(x, y) - \overline{F_c}(x, y) \right\|^2}, \quad (11)$$

where $\overline{F}(x, y)$ is the averaged feature of all patches at location (x, y), and $\overline{F_c}(x, y)$ is the average of patches from class c. We compute discriminative scores $d(x, y)$ for patch locations in a sliding window fashion, with patch size 64×64 and stride 8, and choose the patch locations with top scores. To ensure the diversity of the regions, non-maximum suppression is used to allow only a small amount of overlap.
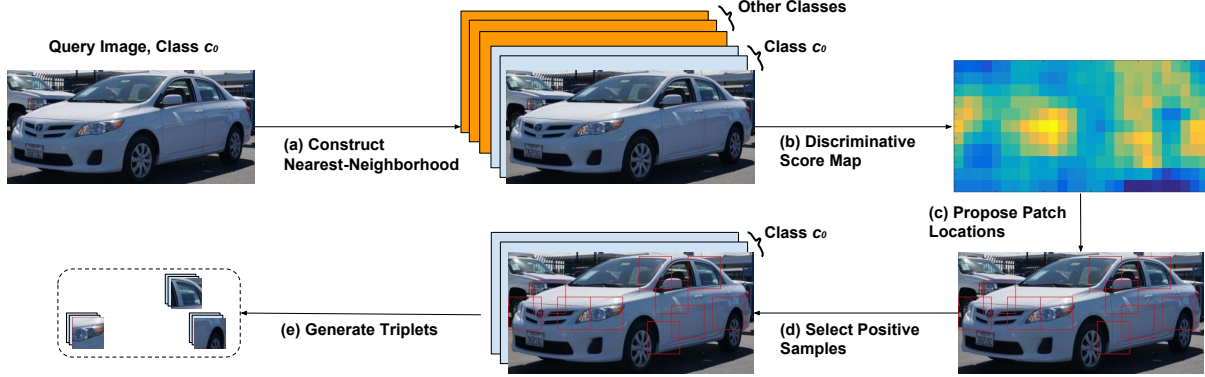
Figure 5. Visualization of the triplet initialization stage. **(a)** A seed image from class c_0 is used to construct its nearest-neighbor set including itself. **(b)** Discriminative score map is generated from the stack of neighboring images. **(c)** Patch locations with top discriminative scores are selected by non-maximum suppression. **(d)** Images from positive class (class c_0) are selected to generate triplets. **(e)** Triplets are generated from positive samples at locations proposed in (c).

Propose Candidate Triplets. For a neighborhood generated from the seed image with class label c_0, candidate triplets are proposed as follows. We first select all positive samples with label c_0 in the neighborhood. For each positive sample, we extract features from patches at the discriminative patch locations obtained in the last step. Then for patch location i, the patch template T_i is obtained by averaging the features of all the positive samples. We propose candidate triplets by selecting all possible combinations of three patch locations. To avoid duplicate triplets, we rank three patch locations by their discriminative scores Eq. (11). We construct triplet detectors with geometric constraints from these patch triplets as discussed in Section 2.2.

In practice, in each neighborhood we find the top 6 discriminative locations and propose $\binom{6}{3} = 20$ candidate triplets. By considering all the neighborhoods for every class, we obtain the pool of candidate triplets.

3.2. Discriminative Triplets Mining by Entropy Scores

Candidate triplets are constructed from potentially discriminative regions measured by Eq. (11). However, this measure is computed only within a small neighborhood and might be noisy due to lack of data. On the other hand, recent approaches to scene understanding [9, 36, 8, 21] mine discriminative patches using a large portion of the training data and obtain very good results. Consequently, we select discriminative triplets by evaluating each candidate triplet on the broader dataset.

For each triplet detector obtained in Section 3.1, we detect triplets in each training image and obtain the maximum detection score as discussed in Section 2.2, *i.e.*, find the top K detections for each appearance detector, consider all K^3 triplets, and choose the one with maximum geometrically-penalized score Eq. (7). We obtain the top detections within

the training set along with their corresponding class labels. If a triplet is discriminative across the training set, the top detections are expected to arise from only one or a few classes. Therefore, if we calculate the entropy of the class distribution over the top detections, the entropy should be low. Let $p(c|\mathbf{T})$ denote the probability of top detections coming from class c for triplet $\mathbf{T}$. Then

$$H(c|\mathbf{T}) = \sum_c p(c|\mathbf{T}) \log p(c|\mathbf{T}) \qquad (12)$$

is an entropy-based measure that has been effectively used by [21, 13] for patch selection. We calculate this measure for all candidate triplets and choose the ones with the lowest entropy to form the set of discriminative triplets.

3.3. Mid-Level Image Representations for Classification

Finally, we use the maximum responses of the mined discriminative triplet detectors to construct a mid-level representation for an image. The dimension of the mid-level representation equals the number of triplet detectors, with each detection score occupying one dimension in the image-level descriptor. The mid-level representation is used as the input for a linear SVM to produce the classification result. We refer to the image-level descriptor as a *Bag of Triplets* (BoT).

4. Experiments

4.1. Triplet Localization with Geometric Constraints

We first design a simple experiment to demonstrate that the geometric constraints described in Section 2 can actually improve patch localization, assuming we already have a good patch set. To achieve this goal, we use the FG3DCar dataset provided by [29]. This dataset consists of 300 car
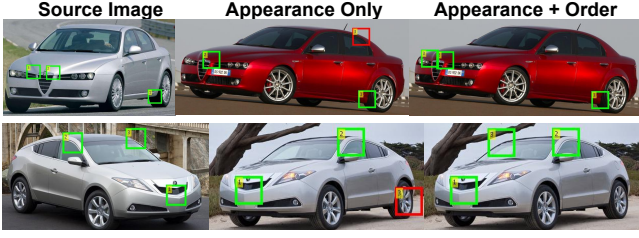
Figure 6. Visualization of the effects of the two geometric constraints. Red boxes are incorrectly localized patches. The order constraint roughly checks the geometric arrangement of three patches and can eliminate incidental false detections which happen to have high appearance score; the shape constraints enforce finer adjustments on patch locations than the order constraint.

images from 30 classes, with each image annotated with 64 ground-truth landmark points. Cars in this dataset have large pose variations, which makes triplet localization difficult.

Experimental Settings. The experiment is designed as follows. For each image, we construct a good set of 64 patches by extracting the ones located at the annotated landmarks. We repeatedly and randomly select two images from the same class and obtain two corresponding sets of 64 patches with their locations. Then we randomly select three patches from one image (denoted as Image 1) to construct a triplet detector in Eq. (6) and attempt to find the corresponding triplet in the pool of 64 patches of the other (denoted as Image 2). During the experiment, patches are represented by the Fisher Vector features provided by [29].

The following four methods are evaluated on this task. The decision procedure of the three methods with gemetric constraints is discussed in Section 2.2.

- Appearance Only (Baseline): Independently apply each patch detector, and choose the detection with the highest score. The detected triplet consists of the three top individual detections. In this method, only the appearance features of the three patches are used.

- Order Constraint: Use the appearance and the order constraint by setting $\eta_s = 0$ in Eq. (5), such that $p_s = 1$ (no shape penalty) always holds in Eq. (7).

- Shape Constraint: Use the appearance and the shape constraint by setting $\eta_o = 0$ in Eq. (3), such that $p_o = 1$ (no order penalty) always holds in Eq. (7).

- Combined: Use both geometric constraints. In practice, we set $\eta_o = 0.5$ and $\eta_s = 1$.

Due to large pose changes, several landmark locations are highly overlapped with each other in some images. Therefore, during evaluation, each patch is regarded as correctly localized if the detected patch is (i) the same as the ground truth corresponding patch; (ii) highly overlapped with the ground truth corresponding patch, with
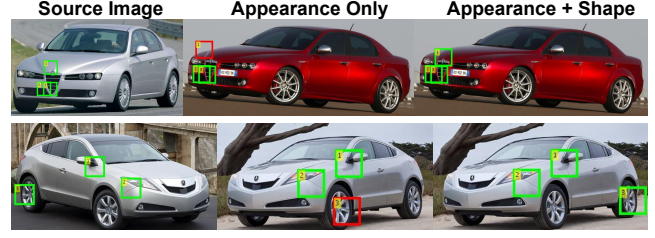
Method	Localization Accuracy (%)	Improvement Over Baseline (%)
Appearance Only	24.9	-
Order Constraint	27.7	11.2
Shape Constraint	34.4	38.2
Combined	35.3	41.9

Table 1. Triplets localization test result on FG3DCar dataset. The localization accuracy and relative improvement over baseline (Appearance Only method) are demonstrated.

overlap/union ratio greater than 50%. Each triplet is regarded as successfully localized if all of its three patches are correctly localized. We randomly select 1000 image pairs, and for each pair we randomly test 1000 triplets. The accuracy of triplet localization is the percentage of successfully localized triplets over all the 1 million triplets evaluated.

Result and Analysis. We demonstrate triplet localization accuracy and relative improvement over baseline in Table 1. Even though we have a human-annotated pool of patches, localization is challenging with appearance only, since the pose variations are large and the appearance detector is learnt with only one positive sample. As we add geometric constraints, we obtain cumulative improvement over the baseline. Typical examples indicating the effects of the two constraints are displayed in Figure 6. The order constraint, which is relatively loose, tends to roughly check the geometric arrangement of three patches. It can eliminate the patches which happen to have a very high appearance score. On the other hand, the shape constraint enforces fine adjustment, which is complementary to the order constraint. With the two geometric constraints combined, the improvement is significant.

4.2. Fine-Grained Classification

We demonstrate fine-grained classification results on three standard fine-grained car datasets. No extra annotation beyond object bounding boxes is used throughout the experiments. When comparing results, we refer to our approach as *Bag of Triplets* (BoT).

4.2.1 14-Class BMVC Cars Dataset Results

Dataset. The fine-grained car dataset provided by [37] (denoted as BMVC-14) consists of 1904 images of cars from 14 classes. [37] has split the data into 50% *train*, 25% *val* and 25% *test*. We follow this setting for evaluation.

Experimental Settings. The implementation details of our discriminative triplet mining approach are briefly stated as follows. Each image is cropped to its bounding box and resized such that the width is 500 (aspect ratio maintained). The patch size is set to be 64×64 and HOG features are extracted to represent the patches for fair comparison to preivously reported results. In the triplet initialization stage, for each seed image we construct the neighborhood of size 20 including itself. As mentioned in Section 3.1, for each neighborhood we propose the top 6 discriminative patch locations and propose $\binom{6}{3} = 20$ triplets for mining. In the triplet mining stage, we obtain the top detections across the whole training set and calculate entropy measure Eq. (12). Then we select 300 discriminative triplets per class. Finally, the mid-level representation has dimension $14 \times 300 = 4200$, which is fed into the linear SVM implemented by LIBLINEAR [11].

We test the following two cases:

- Without Geometric Constraints (Without Geo): In the discriminative mining and mid-level representation construction stages, we adopt the "Appearance Only" triplet detection strategy described in Section 4.1.

- With Geometric Constraints (With Geo): Each time we use a triplet detector, we use Eq. (7) and related techniques in Section 2.2 to incorporate the two constraints. By comparing the two cases, we quantitatively test the effectiveness of geometric constraints.

Results and Analysis. We compare our results with previous work, citing the results from [29], which has provided a summary of previously published results on BMVC-14. It includes several baseline methods such as LLC [41] and PHOW [38] with codebook size 2048, Fisher Vector (FV) [33] with 256 Gaussian Mixture Model (GMM) components, as well as structDPM [37] and BB-3D-G [25] specifically designed for the task. Among these methods, BB-3D-G [25] used extra 3D models, while others only used ground truth bounding boxes as we did. The results are summarized in Table 2. Our method without geometric constraints outperforms all three baseline methods. When geometric constraints are further added, our approach not only outperforms the best reported result using only bounding boxes with a noticeable margin, but it also outperforms the BB-3D-G method which uses extra 3D model fitting. It is worth mentioning that our method with triplet geometric constraints outperforms the DPM-based method [37], which incorporates root-part pair-wise constraints.

Method	Accuracy (%)
LLC [41]	84.5
PHOW [38]	89.0
FV [33]	93.9
structDPM [37]	93.5
BB-3D-G [25]	94.5
BoT (HOG Without Geo)	94.1
BoT (HOG With Geo)	96.6

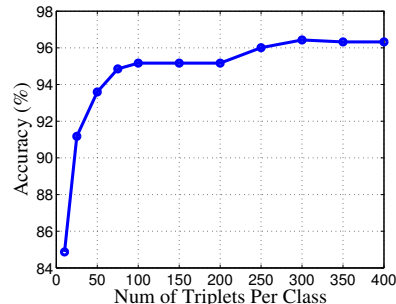Table 2. Results on BMVC-14 dataset.

Figure 7. Classification accuracy with respect to the number of triplets per class.

We plot the performance as a function of the number of discriminative triplets per class (with geometric constraints) in Figure 7. When we use only 10 triplets/class, the performance of 84.9% already outperforms the baseline LLC [41], suggesting that the mined discriminative triplets are highly informative. As we increase the number of triplets per class, performance more or less saturates after 100 triplets/class, although the best performance is at 300 triplets/class. Therefore, when we deal with large-scale datasets such as the Stanford Cars dataset below, we can use a smaller number of triplets to construct lower-dimensional mid-level descriptors without much loss in performance. As the number of triplets/class exceeds 300, the performance decreases, suggesting that the remaining triplets, which rank low by our criteria, do not add discriminatively useful information.

We further visualize the most discriminative triplet measured by the entropy score Eq. (12) for all 14 classes in Figure 8. The triplets in the figure accurately localize the subtle discriminative regions, which are highly consistent with human perception, such as the distinctive side vent grill of Chevrolet Corvette (the second image in the second row of Figure 8, see Figure 8 for more details). This empirically explains why our triplets are highly informative.

4.2.2 196-Class Stanford Cars Dataset Results

Dataset. The Stanford Cars Dataset [25] contains 16,185 car images from 196 classes (denoted as Cars-196). The

Figure 8. Visualization of the most discriminative triplet (measured by Eq. (12)) for each class in BMVC-14 proposed by our method. The triplets accurately capture the subtle discriminative information of each class, which is highly consistent with human perception. For instance, for the **first image in the first row**, the triplet captures the curvy nature of Volkswagen Beetle such as rounded hood; for the **first image in the second row**, the triplet focuses on the rear cargo of the pick-up truck, since Ford F-Series is the only pick-up in the dataset; for the **last image in the first row**, the triplet highlights the frontal face of Jeep Wrangler.

data split provided by [25] is 8,144 training images and 8,041 testing images, where each class has been roughly split 50-50. We follow this setting in our experiment.

Experimental Settings. Our method focuses on generating effective mid-level representations and is independent from the choice of low-level features. In this experiment, in order to fairly compare to both the traditional methods without using extra data/annotation and the more recent ones which finetune ImageNet pre-trained Convolutional Neural Networks (CNN), we evaluate our approach using both HOG and CNN features as the low-level representations of the patches. When extracting CNN features, we directly use the off-the-shelf ImageNet pre-trained CNN model as a general feature extractor without any finetuning. For fair comparison, we adopt the popular 16-layer VGGNet-16 [35] as the network architecture, and extract features from $pool_4$ layer, which is the max-pooled output of its 10th convolutional layer.

Also, we adapt our approach slightly to handle such a large-scale dataset. Instead of traversing the whole dataset, when retrieving nearest-neighbors for a seed image with class label c_0, we regard class c_0 as the positive class, randomly select 29 other classes as negative classes, and retrieve nearest-neighbors within the training images from these 30 classes; when finding top detections for a triplet from class c_0, we use the training images from class c_0 and 14 randomly selected negative classes. Additionally, since we have empirically determined that the discriminative triplets are informative in Figure 7, we select 150 discriminative triplets per class.

Except for the settings described above, other parameters remain the same as those in Section 4.2.1.

Results and Analysis. Our baselines include LLC [41] as HOG-based baseline and AlexNet [26] as CNN baselines. For CNN, we cite the result of training an AlexNet from scratch on Cars-196 without extra data (AlexNet From Scratch) [23] and the result of finetuning an ImageNet pretrained AlexNet on Cars-196 (AlexNet Finetuned) [43]. We also compare with previously published results including BB-3D-G [25], ELLF [23], FT-HAR-CNN [43], Bilinear

CNN (B-CNN) [28] and the method with the highest reported accuracy so far [24]. It is worth mentioning that the last three approaches [43, 28, 24] are CNN-based, where [43] is AlexNet based, [28] is VGGNet-16 based, and [24] is 19-layer VGGNet-19 based. [43] finetunes a CNN with the help of another 10,000 images of cars without finegrained labels; the best result of [28] is achieved by finetuning a two-stream CNN architecture; [24] integrates segmentation, graph-based alignment, finetuned R-CNN [17] and SVM to produce its best result.

The results are displayed in Table 3. Even though we adopted relatively "economical" settings, our method behaves stably and operates at the state-of-the-art performance. When using HOG as low-level patch representation, our approach not only greatly outperforms the HOG-based baseline (LLC) (by more than 15%), but it even outperforms the CNN baseline of finetuned AlexNet by a fairly noticeable margin (more than 2%) – a significant achievement since we are only using HOG and geometric constraints without any extra data or annotations. When using off-the-shelf CNN features, our method with geometric constraints outperforms B-CNN [28] which uses two streams of VGGNet-16, and obtained quite comparable results to the state-of-the-art [24]. Furthermore, our method does not perform finetuning and depends on the strength of our discriminative triplet mining itself, which is supported by the results, rather than the learning capability of CNNs.

To intuitively demonstrate the effectiveness of the geometric constraints, we plot the image-level BoT descriptors from a few classes in the second and third columns of Figure 9. For each class we plot the averaged BoT descriptor across all *test* samples from that class. Figure 9 shows that after introducing the geometric constraints, the BoT descriptor becomes more peaked at the corresponding class, since the geometric constraints help learn more discriminative triplets which generate more peaky responses, as well as penalizing those incorrect detections from other classes which happen to have high appearance scores (which can be clearly seen from the second row of Figure 9). This discriminative capability is achieved during test time, showing
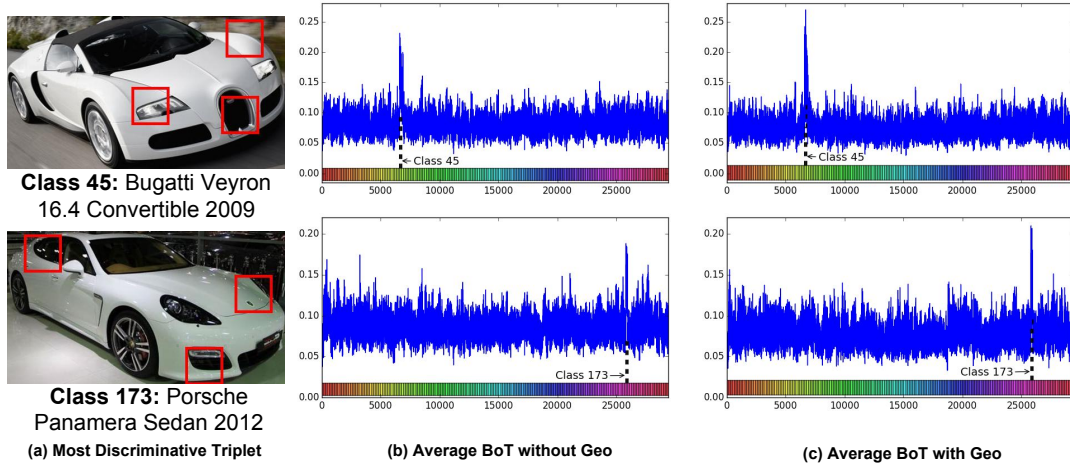
Figure 9. (a) Visualization of the most discriminative triplets of two example classes in Cars-196. (b)(c) The averaged image-level BoT descriptor across all test samples in the corresponding class. Each dimension in the BoT is generated by the response of a mined triplet. The color bars are used to describe the dimensions correponding to the responses of triplets from different classes.

Method	Accuracy (%)
LLC*[41]	69.5
BB-3D-G [25]	67.6
ELLF* [23]	73.9
AlexNet From Scratch [23]	70.5
AlexNet Finetuned [43]	83.1
FT-HAR-CNN [43]	86.3
B-CNN [28]	91.3
Best Result in [24]	92.8
BoT(HOG Without Geo)*	84.6
BoT(HOG With Geo)*	85.7
BoT(CNN Without Geo)	91.2
BoT(CNN With Geo)	92.5

Table 3. Results on Cars-196 dataset. Items with "*" indicate that no extra annotations/data are involved.

that our approach generalizes very well.

Finally, we visualize the most discriminative triplet measured by Eq. (12) in the first column of Figure 9. Similar to Figure 8, our approach captures the subtle difference of fine-grained categories and accurately localizes the discriminative regions, which are highly interpretable by humans. For example, it highlights the distinctive air grill and rounded fender of Bugatti Veyron, and the classical headlight and tail of Porsche.

4.2.3 100-Class FGVC-Aircraft Dataset Results

Finally, to demonstrate that our approach is effective in multiple fine-grained domains, we briefly present our results on FGVC-Aircraft dataset [31], which contains 10,000 images from 100 classes of aircrafts and is of similar scale to the

Method	Accuracy (%)
Symbiotic [5]	75.9
Fine-tuned AlexNet [19]	78.9
Fisher Vector [19]	81.5
B-CNN [28]	84.1
BoT (CNN without Geo)	86.7
BoT (CNN with Geo)	88.4

Table 4. Results on FGVC-Aircraft dataset.

Cars-196 dataset (16185 images from 196 classes). For fair comparison, we use the standard train/test split provided by the dataset provider [31] and the parameter settings of our approach are *exactly the same* as those in Section 4.2.2. We report our results in Table 4. Our approach using CNN features (without fine-tuning) outperforms state-of-the-art (VGGNet-16 based) [28] by a noticeable margin. The results suggest that our approach performs well in various fine-grained domains.

5. Conclusion

We proposed a mid-level patch-based approach for fine-grained classification. We first introduce triplets of patches with two geometric constraints to improve localizing patches, and automatically mine discriminative triplets to construct mid-level representations for fine-grained classification. Experimental results demonstrated that our discriminative triplets mining framework performs very well on both mid-scale and large-scale fine-grained classification datasets, and outperforms or obtains comparable results to the state-of-the-art.

References

[1] T. Berg and P. N. Belhumeur. How do you tell a blackbird from a crow? In *ICCV*, 2013. 1

[2] T. Berg and P. N. Belhumeur. POOF: part-based one-vs.-one features for fine-grained categorization, face verification, and attribute estimation. In *CVPR*, 2013. 1

[3] T. Berg, J. Liu, S. W. Lee, M. L. Alexander, D. W. Jacobs, and P. N. Belhumeur. Birdsnap: Large-scale fine-grained visual categorization of birds. In *CVPR*, 2014. 1

[4] S. Branson, C. Wah, F. Schroff, B. Babenko, P. Welinder, P. Perona, and S. Belongie. Visual recognition with humans in the loop. In *ECCV*, 2010. 1

[5] Y. Chai, V. S. Lempitsky, and A. Zisserman. Symbiotic segmentation and part localization for fine-grained categorization. In *ICCV*, 2013. 1, 8

[6] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005. 3

[7] J. Deng, J. Krause, and L. Fei-Fei. Fine-grained crowdsourcing for fine-grained recognition. In *CVPR*, 2013. 1

[8] C. Doersch, A. Gupta, and A. A. Efros. Mid-level visual element discovery as discriminative mode seeking. In *NIPS*, 2013. 1, 4

[9] C. Doersch, S. Singh, A. Gupta, J. Sivic, and A. A. Efros. What makes paris look like paris? *ACM Trans. Graph.*, 31(4):101, 2012. 1, 4

[10] O. Duchenne, F. R. Bach, I. Kweon, and J. Ponce. A tensor-based algorithm for high-order graph matching. In *CVPR*, 2009. 1

[11] R. Fan, K. Chang, C. Hsieh, X. Wang, and C. Lin. LIB-LINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008. 6

[12] R. Farrell, O. Oza, N. Zhang, V. I. Morariu, T. Darrell, and L. S. Davis. Birdlets: Subordinate categorization using volumetric primitives and pose-normalized appearance. In *ICCV*, 2011. 1

[13] B. Fernando, É. Fromont, and T. Tuytelaars. Mining mid-level features for image classification. *International Journal of Computer Vision*, 108(3):186–203, 2014. 4

[14] V. Ferrari, T. Tuytelaars, and L. J. V. Gool. Wide-baseline multiple-view correspondences. In *CVPR*, 2003. 2

[15] A. Freytag, E. Rodner, and J. Denzler. Birds of a feather flock together - local learning of mid-level representations for fine-grained recognition. In *ECCV Workshop on Parts and Attributes*, 2014. 2

[16] E. Gavves, B. Fernando, C. G. M. Snoek, A. W. M. Smeulders, and T. Tuytelaars. Fine-grained categorization by alignments. In *ICCV*, 2013. 1, 2

[17] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014. 7

[18] C. Göring, E. Rodner, A. Freytag, and J. Denzler. Nonparametric part transfer for fine-grained recognition. In *CVPR*, 2014. 2

[19] P. H. Gosselin, N. Murray, H. Jégou, and F. Perronnin. Revisiting the fisher vector for fine-grained classification. *Pattern Recognition Letters*, 49:92–98, 2014. 8

[20] B. Hariharan, J. Malik, and D. Ramanan. Discriminative decorrelation for clustering and classification. In *ECCV*, 2012. 3

[21] M. Juneja, A. Vedaldi, C. V. Jawahar, and A. Zisserman. Blocks that shout: Distinctive parts for scene classification. In *CVPR*, 2013. 1, 4

[22] P. Kohli, M. P. Kumar, and P. H. S. Torr. P^3 & beyond: Solving energies with higher order cliques. In *CVPR*, 2007. 1

[23] J. Krause, T. Gebru, J. Deng, L. Li, and L. Fei-Fei. Learning features and parts for fine-grained recognition. In *ICPR*, 2014. 2, 7, 8

[24] J. Krause, H. Jin, J. Yang, and F. Li. Fine-grained recognition without part annotations. In *CVPR*, 2015. 1, 7, 8

[25] J. Krause, M. Stark, J. Deng, and L. Fei-Fei. 3d object representation for fine-grained categorization. In *International IEEE Workshop on 3D Representation and Recognition*, 2013. 1, 6, 7, 8

[26] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 7

[27] Y. Li, L. Liu, C. Shen, and A. van den Hengel. Mid-level deep pattern mining. In *CVPR*, 2015. 1

[28] T. Lin, A. RoyChowdhury, and S. Maji. Bilinear CNN models for fine-grained visual recognition. In *ICCV*, 2015. 1, 7, 8

[29] Y. Lin, V. I. Morariu, W. H. Hsu, and L. S. Davis. Jointly optimizing 3d model fitting and fine-grained classification. In *ECCV*, 2014. 1, 4, 5, 6

[30] J. Liu, A. Kanazawa, D. W. Jacobs, and P. N. Belhumeur. Dog breed classification using part localization. In *ECCV*, 2012. 1

[31] S. Maji, E. Rahtu, J. Kannala, M. B. Blaschko, and A. Vedaldi. Fine-grained visual classification of aircraft. *CoRR*, abs/1306.5151, 2013. 8

[32] O. M. Parkhi, A. Vedaldi, A. Zisserman, and C. V. Jawahar. Cats and dogs. In *CVPR*, 2012. 1

[33] F. Perronnin, J. Sánchez, and T. Mensink. Improving the fisher kernel for large-scale image classification. In *ECCV*, 2010. 6

[34] J. Pu, Y. Jiang, J. Wang, and X. Xue. Which looks like which: Exploring inter-class relationships in fine-grained visual categorization. In *ECCV*, 2014. 1

[35] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014. 7

[36] S. Singh, A. Gupta, and A. A. Efros. Unsupervised discovery of mid-level discriminative patches. In *ECCV*, 2012. 1, 4

[37] M. Stark, J. Krause, B. Pepik, D. Meger, J. J. Little, B. Schiele, and D. Koller. Fine-grained categorization for 3d scene understanding. In *BMVC*, 2012. 6

[38] A. Vedaldi and B. Fulkerson. Vlfeat: an open and portable library of computer vision algorithms. In *Proceedings of the 18th International Conference on Multimedia 2010, Firenze, Italy, October 25-29, 2010*, pages 1469–1472, 2010. 6

[39] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The caltech-ucsd birds 200-2011 dataset. In *Technical Report CNS-TR-2011-001, Caltech,*, 2011. 1

[40] C. Wah, G. V. Horn, S. Branson, S. Maji, P. Perona, and S. Belongie. Similarity comparisons for interactive fine-grained categorization. In *CVPR*, 2014. 1

[41] J. Wang, J. Yang, K. Yu, F. Lv, T. S. Huang, and Y. Gong. Locality-constrained linear coding for image classification. In *CVPR*, 2010. 6, 7, 8

[42] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona. Caltech-ucsd birds 200. In *Technical Report CNS-TR-2010-001, Caltech,*, 2010. 1

[43] S. Xie, T. Yang, X. Wang, and Y. Lin. Hyper-class augmented and regularized deep learning for fine-grained image classification. In *CVPR*, 2015. 1, 7, 8

[44] S. Yang, L. Bo, J. Wang, and L. G. Shapiro. Unsupervised template learning for fine-grained object recognition. In *NIPS*, 2012. 1

[45] B. Yao, G. R. Bradski, and L. Fei-Fei. A codebook-free and annotation-free approach for fine-grained image categorization. In *CVPR*, 2012. 1

[46] R. Zass and A. Shashua. Probabilistic graph and hypergraph matching. In *CVPR*, 2008. 1

[47] N. Zhang, R. Farrell, and T. Darrell. Pose pooling kernels for sub-category recognition. In *CVPR*, 2012. 1

[48] N. Zhang, R. Farrell, F. N. Iandola, and T. Darrell. Deformable part descriptors for fine-grained recognition and attribute prediction. In *ICCV*, 2013. 1