Deep Multi-Patch Aggregation Network for Image Style, Aesthetics, and Quality Estimation

Xin Lu* Zhe Lin† Xiaohui Shen† Radomír Měch† James Z. Wang*
*The Pennsylvania State University, University Park, Pennsylvania
†Adobe Research, San Jose, California
{xinlu, jwang}@psu.edu, {zlin, xshen, rmech}@adobe.com

Abstract

This paper investigates problems of image style, aesthetics, and quality estimation, which require fine-grained details from high-resolution images, utilizing deep neural network training approach. Existing deep convolutional neural networks mostly extracted one patch such as a down-sized crop from each image as a training example. However, one patch may not always well represent the entire image, which may cause ambiguity during training. We propose a deep multi-patch aggregation network training approach, which allows us to train models using multiple patches generated from one image. We achieve this by constructing multiple, shared columns in the neural network and feeding multiple patches to each of the columns. More importantly, we propose two novel network layers (statistics and sorting) to support aggregation of those patches. The proposed deep multi-patch aggregation network integrates shared feature learning and aggregation function learning into a unified framework. We demonstrate the effectiveness of the deep multi-patch aggregation network on the three problems, i.e., image style recognition, aesthetic quality categorization, and image quality estimation. Our models trained using the proposed networks significantly outperformed the state of the art in all three applications.

1. Introduction

Problems of image styles, aesthetics, and quality estimation have been actively investigated over the past decade [17, 24, 26, 18], with the goal of endow computers with the capability of perceiving aesthetics, style, and visual quality as human vision systems. Potential usage of methods developed for these three tasks could be foreseen towards wide applications from intelligent computer systems to real-time, mobile applications. Unlike tasks of image classification and object detection, the key of these problems is to capture both the holistic information and fine-grained high resolution details, as presented in [17] and [24], respectively.

Deep convolutional neural network has demonstrated effectiveness for various image classification tasks, but most of the work ignored fine-grained high resolution details in images. Such fine-grained details has been shown highly useful in applications such as image quality estimation [17], image aesthetics categorization, and image style classification [24]. Learning fine-grained details is challenging, as that information locates in original, relatively high resolution images (*e.g.*, 1024×768, 2560×1920). Deep convolutional neural networks are often trained with $256 \times 256 \times 3$ inputs (for color images), and training deep networks with large-size input dimensions requires much longer training time and a significantly larger network structure, training dataset, and hardware memory.

To learn fine-grained details using deep network training approaches, previous studies [24, 17] represented each image with one randomly cropped patch, and paired the patch with the label of the image as one training example. Such approach generates ambiguity in training examples as aesthetics/style/quality attributes in one patch may not well represent the fine-grained information in the entire image.

To address this issue, we formulate the learning problem by representing an input image with a small set or *bag* of patches cropped from it and associating the set with the image's training label, and propose novel deep neural network architectures to solve the problem. Instances in a bag are ***orderless*** and the central idea is to perform aggregation of the instances. In this work, we propose a deep multi-patch aggregation network architecture (DMA-Net) to support fine-grained details learning utilizing multiple patches cropped from one image.

Designing an optimal network structure that supports both feature learning and aggregation function learning simultaneously is nontrivial. In this paper, we propose two novel layers: ***statistics layer*** and ***sorting layer*** to enable aggregation of multiple input sources. The statistics layer leverages common statistical functions to let the output be independent of the input order, and the sorting layer leverages a sorting function to achieve the same goal. Building upon the two novel layers, we develop two different aggregation structures embedded in deep neural networks to support deep multi-patch aggregation network training. We demonstrate the effectiveness of the models trained with the proposed neural network architectures in three applications: image style classification, aesthetic quality categorization, and image quality estimation.

Our main contributions are three-fold.

- We introduce novel neural network architectures to support learning from multiple patches. In particular, we propose two novel network layers and their aggregation strategies to support multi-patch aggregation.
- We apply our proposed neural network-based approach to three vision applications that greatly depend on fine-grained details and demonstrate significant improvement over the state of the art.
- By leveraging both the holistic information of the image and the extracted fine-grained details using DMA-Net, we further boost the performance in image style classification and image aesthetics categorization.

2. Related Work

2.1. Deep Neural Networks

The success made by the deep neural network approach for image classification [21] has inspired many follow-on studies on deep learning and their applications to vision. We review the studies that are closely related to our work.

Recent work [9, 30, 17, 22] focused on adapting deep neural network training to various vision applications. They mostly were able to show some improvement with slight modification of the network structures (*e.g.*, adding a layer or adding a column) or changing the training strategy (*e.g.*, fine-tuning). Beside the useful techniques such as ReLU, dropout, and data augmentation introduced in [21], we notice two key ideas that have led to promising results in classification problems:

(i) **Multiple Image Resolutions.** Different vision applications require information from different image resolution. In image classification, deep convolutional neural network (CNN) achieved great success by training on $256 \times 256 \times 3$ images. However, in image quality estimation, image aesthetics, and image style classification, training deep neural networks on relatively high-resolution images helps improve the performance significantly [24, 17].

When using imagenet feature as a generic image descriptor for recognition tasks, researchers found that aggregating descriptors from different image resolutions helps boost the classification performance. In [27, 20, 15], researchers computed ImageNet features (*i.e.*, features extracted by the neural network trained in the ImageNet Challenge [21]) from multi-scale image pyramid for object recognition, scene recognition, and object detection. In [10], Gong *et al.* improved the geometric invariance of imagenet activations by pooling features extracted from multi-resolution patches. In addition, recent study has demonstrated the significance of maintaining the aspect ratio of images through spatial pyramid pooling [12] in object detection.

In [24, 17], a single randomly cropped patch was used to represent the entire image. In [27, 20, 15, 10], the neural networks are trained on small downsized images and applied to multi-resolution images in the testing stage.

In this paper, the key idea is to represent the original high-resolution input image using multiple patches and construct the deep multi-patch aggregation network to directly learn from the bags of multiple patches.

(ii) **Multi-Column Neural Network.** Multi-column neural network [5, 1, 24] has been demonstrated as an efficient approach to improve performance of single-column neural networks in various classification problems. Motivated by part-based approaches (*e.g.*, [7, 16, 29, 3]), recent studies attempt to train multiple convolutional neural networks on aligned parts. Zhang *et al.* trained pose-normalized CNNs on semantically aligned part patches, whose learned features are associated with certain parts under specific poses [39]. A similar approach has been applied to fine-grained category detection in [38]. Bourdev *et al.* applied trained CNN to extract features on poselet patches [4]. In addition to aligned patches, multi-column neural networks were trained with heterogenous inputs in [24].

In multi-column neural networks, one could also constrain the multi-column structures to share weights and aggregate multi-column outputs using max-pooling. Wei *et al.* probed the image multi-labeling problem using such a network structure [32]. Whereas our work follows a similar strategy of constraining the multi-column structures to share weights, our work is different in three aspects: 1) Wei *et al.* implicitly assume that each cropped patch is associated with one of the image labels in a multi-labeled image classification problem, while we take the entire set of multiple patches as one training sample in the proposed deep framework. 2) Wei *et al.* applied the max-pooling only on the final labeling layer of activations. In contrast, we take multiple randomly cropped patches as inputs and conduct patch aggregation in an intermediate layer through the two proposed patch aggregation structures, which better utilize the interactions of patch features; 3) The activation aggregation in the earlier work is hand-designed to be max pooling, whereas we propose two novel network layers to learn the aggregation from training data.

2.2. Multiple Instance Learning

Our work is related to the multiple-instance learning (MIL) paradigm, under which the training set consists of bags of instances and the goal is to train a bag classifier. In binary classification settings, a standard assumption in MIL states that a bag is labeled as positive with at least one positive instance, and a bag is labeled as negative if all instances in the bag are negative. According to the assumption, probability measures such as Noisy-OR [13, 25] and ISR [19] are designed for multiple instance aggregation, following which various cost functions were proposed to learn a function that maps the multiple-instances as a whole to the label. Recent studies on MIL targeted manifold assumptions on feature distributions of instances in a bag [2], instance selection for MIL [8], and vision applications, such as object detection [37, 23], content-based image retrieval [36], multi-class image classification [40], and very high-resolution satellite imagery [31].

Recent studies have attempted to unify deep learning and the MIL framework for many applications. In [35] and [14], deep learning features were incorporated into a MIL framework, and used to perform medical image analysis and object detection, respectively. In [28], the MIL-based training of CNN was discussed and applied to object detection. In [34], the learned deep features were unified in the MIL framework and applied to image classification and auto-annotation.

Although our work is related to MIL [6, 33, 11, 40, 37, 23, 36, 19, 8, 2, 31], the proposed network architecture is a generalized deep learning framework, which is not limited to using $\max$ as the aggregation function.

3. Convolutional Neural Network

Before introducing our Deep Multi-Patch Aggregation Networks, we first review the deep convolutional neural network (CNN) training approach [21] that uses a single image patch as input. As a supervised learning approach, CNN is commonly adopted to learn a function $f : \mathcal{X} \to \mathcal{Y}$, from a collection of training examples $\{(x_n, y_n)\}_{n\in[1,N]}$, where N is the size of the training set, x_n is the image, and y_n is the associated label. During each training epoch, CNN takes one patch p_i extracted from each x_n as input, where p_n has a similar size with x_n (*e.g.*, cropping $224 \times 224 \times 3$ patches from $256 \times 256 \times 3$ images). Suppose we use the output from the second last layer as the feature $\mathbf{d}_n$ of patch p_n, the training of the last layer is done by maximizing the following log likelihood function:

$$l(\mathbf{W}) = \sum_{n=1}^{N} \sum_{c\in\mathcal{Y}} \mathbb{I}(y_n = c) \log p(y_n = c \mid \mathbf{d}_n, \mathbf{w}_c) \ \ , \quad (1)$$

where N is the number of training images, $\mathbf{W} = \{\mathbf{w}_c\}_{c\in\mathcal{Y}}$ is the set of model parameters, and $\mathbb{I}(x) = 1$ iff x is true. The probability $p(y_n = c \mid \mathbf{d}_n, \mathbf{w}_c)$ is expressed as

$$p(y_n = c \mid \mathbf{d}_n, \mathbf{w}_c) = \frac{\exp\left(\mathbf{w}_c^T \mathbf{d}_n\right)}{\sum_{c'\in\mathcal{Y}} \exp\left(\mathbf{w}_{c'}^T \mathbf{d}_n\right)} \ \ . \quad (2)$$

To efficiently train the deep networks, the size of x_n cannot be too large (*e.g.*, a typical size is $256 \times 256 \times 3$). Given fixed settings in the convolutional layers (*e.g.*, number of layers, filter size, and pooling patch size), larger-size training images result in larger weight matrix in the fully-connected layers. Therefore, increasing the input size of CNN would require significantly more training data, larger hardware memory, and longer training duration. Real-world high-resolution images are typically transformed to smaller size through global functions such as warping [24], center-crop [21] and padding [24]. To predict the class label of an image, the same global transform has to be performed first.

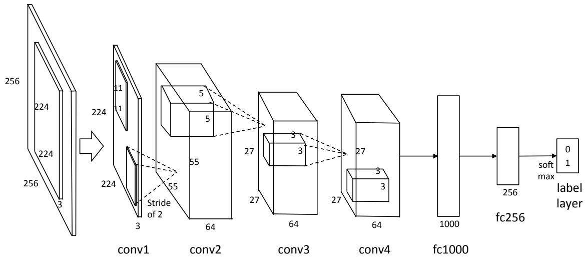

Figure 1. Convolutional Neural Network (CNN).

The problem of training CNN on downsized input images is that the network fails to encode fine-grained information existing in original-resolution images, which results in information loss. Alternatively, prior work [24, 17] replaced each image x_n with a randomly cropped patch p_n of x_n during training, and adopted multi-view test, *i.e.*, CNN takes patches of images (and their horizontal flips) as input and aggregates the final output. As discussed in Section 1, a single patch is likely not informative enough or, even worse, ambiguous for any learning method to train a reliable model.

In all the aforementioned situations, CNNs are learned based on a training set where each patch has an associated class label. We refer to such network as Convolutional Neural Network (CNN), as illustrated in Figure 1.

4. Deep Multi-Patch Aggregation Network

We represent each image with a bag of instances (*e.g.*, patches) and associate the set with the image's label. The training data become $\{P_n, y_n\}_{n\in[1,N]}$ where $P_n = \{p_{nm}\}_{m\in[1,M]}$ is the set of M patches cropped from each image. Our objective is to estimate a function $f : \mathcal{Z} \to \mathcal{Y}$, where $\mathcal{Z}$ is the domain of bags. The mapping function f sequentially performs three steps: extracting features of individual patches in a bag, aggregating the features and predicting the label of the bag.

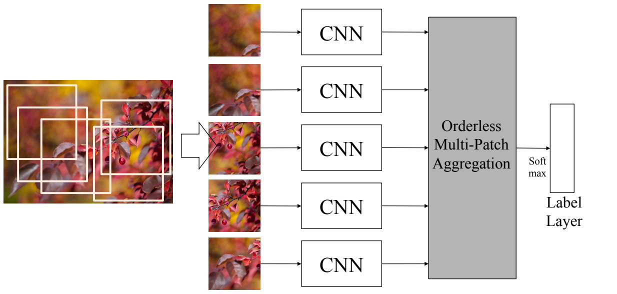

Figure 2. Deep Multi-Patch Aggregation Network (DMA-Net). DMA-Net aligns magnitude of individual patch outputs using shared CNNs[1] and conducts aggregation on orderless patch outputs.

We propose a novel deep neural network, Deep Multi-Patch Aggregation Network (DMA-Net), which conducts feature extraction and aggregation function learning jointly. Figure 2 shows the structure of DMA-Net, which contains two main parts: a set of CNNs[1] which extract features from multiple input patches, and an orderless aggregation structure which combines the output features from the CNNs.

When designing the network structure, two issues need to be considered. First, in order for the network to learn aggregation, features returned from different CNNs should be comparable. To satisfy this requirement, we force all the CNN columns to share the same weights.

Second, our network does not make any assumption about the order of the patches in a bag. There are two strategies to deal with this "orderless" constraint. The first and straightforward strategy is to take commonly adopted statistical functions, where the output is independent of input orders, *e.g.*, $\min$, $\max$, median and mean. We refer to these functions as orderless statistical functions. To increase the learning power of our network, an alternative strategy is to first arrange outputs from the shared CNNs in a certain order and learn aggregation functions on the ordered inputs. Here, we adopt both strategies and propose two different structures for multi-patch aggregation: the *Statistics Aggregation Structure* and the *Fully-Connected Sorting Aggregation Structure*. In the first structure, we use orderless statistical functions to perform aggregation. In the latter structure, we introduce a sorting layer, which aligns values from each dimension of patch features, and then aggregation function is applied to the sorted values.

In the remainder of this section, we will describe the two aggregation structures in detail. We first introduce some notations. Let $D_n = \{\mathbf{d}_m^n\}_{m\in[1,M]}$ be the set of patch features of bag P output by the shared CNNs, where $\mathbf{d}_m^n$ is a K-dimensional vector. For simplicity, we will omit the index n in the following. Denote by T_k the set of values of the k-th component of all $\mathbf{d}_m \in D$, *i.e.*, $T_k = \{d_{mk}\}_{m\in[1,M]}$. We use $\oplus$ as the vector concatenation operator which produces a column vector.

4.1. Statistics Aggregation Structure

The core and first component of this structure is the statistics layer, which is comprised of a collection of orderless statistical functions, *i.e.*, $\mathcal{S} = \{S_u\}_{u\in[1,U]}$. Each S_u computes a certain orderless statistics of the set of patch features D returned by the shared CNNs. In our experiments, we have $\mathcal{S} = \{\min, \max, \mathrm{median}, \mathrm{mean}\}$. The outputs of the functions in $\mathcal{S}$ are concatenated and then aggregated by a fully-connected layer to produce a V_{stat}-dimensional feature vector. The whole structure can be expressed as a function $g : \{D\} \to \mathbb{R}^{V_{\text{stat}}}$:

$$g(D) = \mathbf{W} \times \left(\oplus_{u=1}^{U} \oplus_{k=1}^{K} S_u(T_k)\right) \ , \tag{3}$$

where $\mathbf{W} \in \mathbb{R}^{V_{\text{stat}} \times UK}$ is the parameters of the fully-connected layer. The left of Figure 3 shows an example of Statistics Aggregation Structure with $M = 5$ and $K = 3$.

In the forward propagation stage, the output o_j of each neuron j at the statistics layer can be expressed as

$$o_j = \sum_{m=1}^{M} \sum_{k=1}^{K} r_{mk\to j} o'_{mk} \ . \tag{4}$$

Here $r_{mk\to j}$ can be considered as the "contribution" of the neuron of d_{mk} to the neuron j at the statistics layer. Denote by δ_j the error propagated to neuron j at the statistics layer, the error δ'_{mk} backpropagated to the neuron of d_{mk} is computed as

$$\delta'_{mk} = \sum_{j} r_{mk\to j} \delta_j \ . \tag{5}$$

4.2. Fully-Connected Sorting Aggregation

The learning capacity of the Statistics Aggregation Structure is limited by the predefined statistical functions used in the statistics layer. To further generalize the feature aggregation function, we propose this structure to aggregate the patch features at each dimension and among different feature dimensions.

As the fully-connected layer assumes that the input is ordered, before $\{T_k\}_{k=1}^{K}$ can be fed into a fully-connected layer, we need to define an order on the elements in each of the $T_k \in \{T_k\}_{k=1}^{K}$. A straightforward way is "order by value". Let be τ a sorting function, *i.e.*, $\tau(T_k) = (d_{(1)k}, d_{(2)k}, \ldots, d_{(M)k})$ where $d_{(1)k} \le d_{(2)k} \le, \ldots, \le d_{(M)k}$. We define the layer that implements τ as the sorting layer. We then concatenate $\{\tau(T_k)\}_{k=1}^{K}$ into a single vector, which is then fed into a fully-connected layer:

$$g(D) = \mathbf{W} \times \oplus_{k=1}^{K} \tau(T_k) \ , \tag{6}$$

[1] The CNN refers to the convolutional neural networks drawn in Figure 1 from the input layer to the fc256 layer.

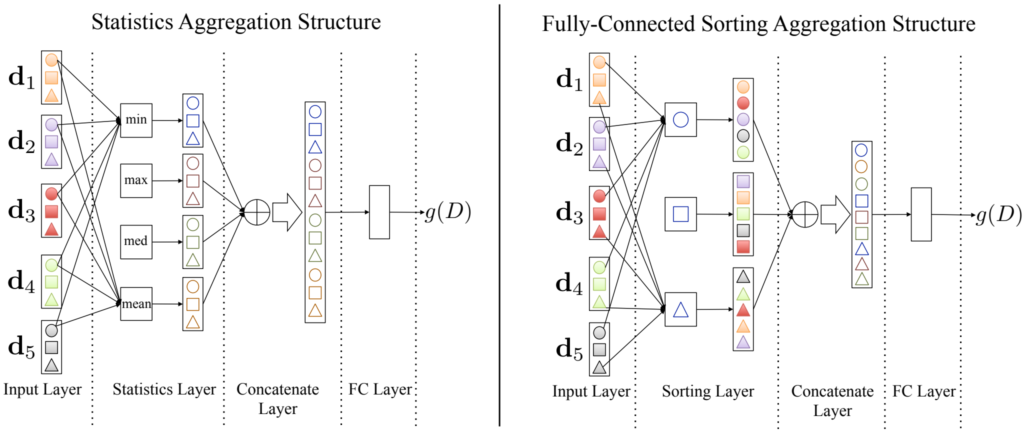

Figure 3. Examples of two multi-patch aggregation structures in DMA-Net. Left: Statistics Aggregation Structure. Right: Fully-Connected Sorting Aggregation. The input is a bag of $M = 5$ patch features $D = \{\mathbf{d}_1, \mathbf{d}_2, \dots, \mathbf{d}_5\}$ where the feature dimension is $K = 3$. Not all the connections are shown between input and statistics/sorting layers for clarity. The figure is best viewed in color.

where $\mathbf{W} \in \mathbb{R}^{V_s \times MK}$ is the parameters of the fully-connected layer.

As we can express the output of the sorting layer as the weighted sum of the outputs from the previous layer, back-propagation can be performed as Equation 5. We show an example of this structure in Figure 3 (right).

4.3. Training with Image Details

The patch size is fixed to be $256 \times 256 \times 3$. Training a DMA-Net consists of two steps. First, CNN is trained[2], then the weights of CNN in the DMA-Nets are initialized by the weights of the learned CNN, with which we intend to accelerate weight initialization in DMA-Net training. The number of patches in a bag is set to be 5. During DMA-Net training, we randomly extracted 5 patches from each original-size image in each training epoch, and feed them to the five CNNs with shared weights in parallel. The initial learning rate is 0.001 for all layers, and is annealed by 0.1 every time the training loss plateaus. We used weight decay of $1e - 5$ and momentum of 0.9. We used the convnet[3] package to train CNN and implemented the two new layers and the aggregation structures.

5. Experimental Results

We evaluate the proposed DMA-Net[4] on three applications, style classification, aesthetics categorization, and quality estimation, using real-world photos. This section introduces the baselines that we compared with, the dataset we utilized, and our experimental settings, and reports experimental results.

5.1. Settings

We denote the DMA-Net using Statistics Aggregation Structure (Section 4.1) as **Ours-DMA-Net**$_{\text{stat}}$ and Fully-Connected Sorting Aggregation Structure (Section 4.2) as **Ours-DMA-Net**$_{\text{fc}}$. To evaluate the proposed approach, the DMA-Net (**Ours-DMA-Net**$_{\text{stat}}$ and **Ours-DMA-Net**$_{\text{fc}}$) were compared with several baseline methods:

(i) **CNN** performs single-column CNN training and testing. One randomly cropped patch from each image was used as training, and the label of the patch used for training is the label of the entire image. In **SPP**[5], the entire image was used as training data, incorporating the SPP [12] layer in the CNN structure.

(ii) **DMA-Net**$_{\textbf{ave}}$ and **DMA-Net**$_{\textbf{max}}$ train deep multi-patch aggregation network using standard patch pooling scheme. **DMA-Net**$_{\textbf{ave}}$ performs average pooling and **DMA-Net**$_{\textbf{max}}$ performs max pooling. No aggregation structures were used in DMA-Net$_{\text{ave}}$ and DMA-Net$_{\text{max}}$.

Since our DMA-Nets rely on fine-grained details of the image by cropping multiple patches, one may argue that the global view of the entire image would also be useful for these tasks. Given limited training data in each specific task, a simple solution for this is to leverage pre-trained models with external data (*e.g.*, ImageNet features). To this end, we integrate our approach with ImageNet features (**ImageNet Fusion**): In **Alexnet-FTune** [21], we fine-tuned the Alexnet [21] by resizing all the training images to $256 \times 256 \times 3$ as training data and fine-tuning its last layer to fit image style labels[6]. In **Ours-DMA-Net-ImgFu**, we averaged the prediction results of **DMA-Net** and **Alexnet-**

[2] We randomly cropped $256 \times 256 \times 3$ patches from the original image as training examples.

[3] http://code.google.com/p/cuda-convnet/

[4] In all DMA-Nets, we used random sampling to generate multiple patches for network training. Sampling patches from four corners and the center did not result in performance improvement.

[5] We applied the code provided by [12] at: https://github.com/ShaoqingRen/SPP_net.

[6] Fine-tuning all the layers produces worse results than fine-tuning the last layer only due to the limited number of training data

Table 1. Summary of baselines and our approaches

Without external Data	Baselines	(a) Single patch network training	CNN SPP
		(b) Deep multi-patch aggregation network training with naive aggregation	DMA-Net$_{\text{ave}}$ DMA-Net$_{\text{max}}$
	Our approaches	(c) Deep multi-patch aggregation network training	Ours-DMA-Net$_{\text{stat}}$ Ours-DMA-Net$_{\text{fc}}$
With external data	Baselines	(d) ImageNet fine-tune	Alexnet-FTune
	Our approaches	(e) Deep multi-patch aggregation network training with ImageNet fusion	Ours-DMA-Net-ImgFu$_{\text{stat}}$ Ours-DMA-Net-ImgFu$_{\text{fc}}$

FTune. We summarize all baseline approaches and our approaches in Table 1.

To ensure the fair comparison between the proposed approach and the baselines, we made the following experimental settings.

In **training**, all networks were initialized with the same learning rate and all networks were fully-trained (*i.e.*, network training was stopped at the point when training error stops dropping.). The **network structure** of **CNN** and each column of **DMA-Net$_{\text{ave}}$**, **DMA-Net$_{\text{max}}$**, and the proposed **DMA-Net** (**Ours-DMA-Net$_{\text{stat}}$** and **Ours-DMA-Net$_{\text{fc}}$**) share exactly the same network architecture up to the two fully-connected layers in individual CNNs (fc1000 and fc256). In **DMA-Net$_{\text{ave}}$**, **DMA-Net$_{\text{max}}$**, and the proposed **Ours-DMA-Net**, 5 patches were used per image.

In **testing**, we used the same collection of patches among all baseline approaches and the proposed Ours-DMA-Net$_{\text{stat}}$ and Ours-DMA-Net$_{\text{fc}}$. For each image, 250 patches are randomly sampled offline, and divided into 50 groups, 5 patches per group. For **CNN**, 250-time evaluation is performed on those 250 patches per image, and we averaged the result on each patch as the final result. In **DMA-Net$_{\text{ave}}$**, **DMA-Net$_{\text{max}}$**, the **Ours-DMA-Net$_{\text{stat}}$**, and the **Ours-DMA-Net$_{\text{fc}}$**, for each image, each of the patch group was evaluated at a time, and the evaluation was repeated 50 times using each of the 50 patch groups. The final result was the averaged result of all patch groups. In **SPP** and **Alexnet-FTune**, the entire image was used for testing. We conducted 250-view test and averaged the results.

We used the architecture in Figure 1 because it was demonstrated effective for aesthetics and style classification [24]. The CNN includes four convolutional layers, and two pooling and normalization layers following the first and second convolutional layers, and two fully-connected layer of 1000 and 256 neurons respectively. The first and forth convolutional layers have 64 filters. The second and third convolutional layers have 32 filters in image style classification and image quality estimation and 64 filters in image aesthetics.

5.2. The Datasets

We test our approach on three datasets:

(i) **Image Style Dataset**: We used the AVA Style dataset introduced in [26] to evaluate classification of 14 different photographic style labels. The 14 style classes include: complementary colors, duotones, HDR, image grain, light on white, long exposure, macro, motion blur, negative images, rule of thirds, shallow DOF, silhouettes, soft focus, and vanishing point. The publishers of the dataset provide a train/test split ($11,000$ for training and $2,573$ for testing). **Average Precision (AP)** and **Mean average precision (mAP)** are the evaluation metric.

(ii) **Image Aesthetics Dataset**: The AVA aesthetics dataset [26] includes $250,000$ images, where each image has about 200 aesthetic ratings ranging from one to ten. We follow the experimental settings in [26], and use the same collection of training data and testing data: $230,000$ images for training and $20,000$ images for testing[7]. Training images are divided into two categories, *i.e.*, low-aesthetic images and high-aesthetic images, based on the same criteria as in [26]. **Overall Accuracy** is the evaluation metric.

(iii) **Image Quality Dataset**: The problem of image quality estimation on real-world photos is different from the conventional problem of no-reference image quality assessment [17]. In no-reference image quality assessment, the corruption is synthetically added and uniformly distributed in the entire image. In real-world photos, the situation is more complicated. This motivates us to study image quality estimation on real-world photos. This problem is also different from image aesthetic categorization because in real-world photos the variation of image quality is much larger than that of professional photographs, which results in a more complex problem that fires in concert with both the traditional problems of quality estimation and aesthetics.

We collected $6,478$ real-world high resolution color photos (*e.g.*, 1024×768 or 2560×1920) from the Internet, and manually labeled them as high quality or low quality in aspect of lighting, color and quality. Low-quality pho-

[7] We have $19,930$ test images as some images are no longer existing on the Web.

Table 2. Image style classification on the AVA style dataset

Methods	AP	mAP
CNN	56.99%	56.83%
SPP	44.56%	47.04%
DMA-Net$_{\mathrm{ave}}$	54.7%	56.9%
DMA-Net$_{\mathrm{max}}$	55.71%	57.11%
Ours-DMA-Net$_{\mathrm{stat}}$	62.83%	59.19%
Ours-DMA-Net$_{\mathrm{fc}}$	62.46%	60.01%
Alexnet-FTune	59.09%	58.02%
Ours-DMA-Net-ImgFu$_{\mathrm{stat}}$	69.74%	63.95%
Ours-DMA-Net-ImgFu$_{\mathrm{fc}}$	69.78%	64.07%
Murray *et al.* [26]	n/a	53.85%
Karayev *et al.* [18]	n/a	58.1%
Lu *et al.* [24]	56.93%	56.81%

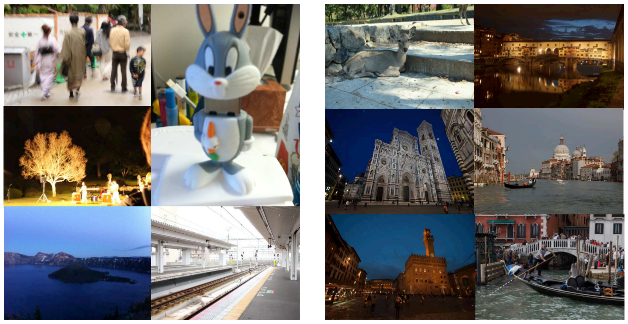

Figure 4. Examples in the real-world photo quality dataset. Left: low-quality photos. Right: high-quality photos.

tos include images with poor lighting (*e.g.*, over-exposure and under-exposure), lacking color combination, and inferior sharpness (*e.g.*, blur and strong noise). We obtained 2,793 negative and 3,685 positive photos. We show example photos in Figure 4. We randomly selected 500 negative photos and 500 positive ones for testing and used the rest of photos for training. Our dataset is available for others. **Overall Accuracy** is the evaluation metric.

5.3. Results

5.3.1 Style Classification

As shown in Table 2, results of the proposed DMA-Net approach (Ours-DMA-Net$_{\mathrm{stat}}$ and Ours-DMA-Net$_{\mathrm{fc}}$) all outperforms the single-patch network training approach (CNN and SPP) in terms of both AP and mAP. The reason that SPP did not perform well may because the parameters in the SPP layer is optimized for the image classification task and for the ImageNet architecture. It may also be limited by the scarceness of training examples when using the entire image as training data. Meanwhile, the results in the table show that the multi-patch aggregation network using simpler pooling strategies of average (DMA-Net$_{\mathrm{ave}}$) and max (DMA-Net$_{\mathrm{max}}$) performs much worse than the proposed DMA-Net training approach. Such results indicate that training network on multiple patches generates better predicting performance than network training on single patch.

The DMA-Net approach alone has improved the best performance on the AVA style dataset for image style classification (mAP: 58.1%[18], AP: 56.93%[24]). We have also noticed that ours-DMA-Net without using external data performs better than Alexnet-FTune using external data and [18] that utilized the ImageNet feature. It indicates that the fine-grained information captured by the proposed multiple patch training strategy is very useful for image style classification, and the global view itself without the fine-grained information is not sufficient for such classification task. By integrating the global view (*i.e.*, ImageNet features), the performance of the DMA-Net (Ours-DMA-Net-ImgFu) is further boosted by a large margin, as it effectively integrates both the global and fine-grained information of the images.

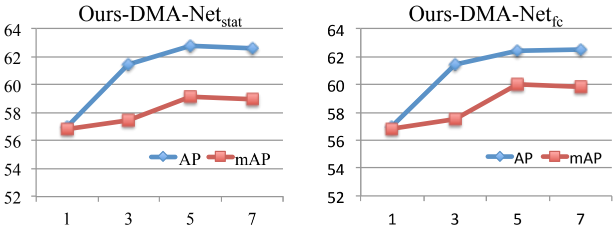

Figure 5. Performance of DMA-Net using different number of patches (Left: Ours-DMA-Net$_{\mathrm{stat}}$, Right: Ours-DMA-Net$_{\mathrm{fc}}$). mAP is denoted by red and AP is denoted by blue. Ours-DMA-Net$_{\mathrm{stat}}$ and Ours-DMA-Net$_{\mathrm{fc}}$ were trained and evaluated using 3, 5, and 7 patches, respectively.

To evaluate the performance of the proposed DMA-Net using different numbers of patches, we trained and evaluated Ours-DMA-Net$_{\mathrm{stat}}$ and Ours-DMA-Net$_{\mathrm{fc}}$ with three, five, and seven patches, respectively. We present the performance of AP and mAP in Figure 5. As shown in the Figure, both the performances converge in general as the number of patches increasing. As more number of patches requires larger GPU memory and increases training duration, we used five patches in all the following experiments.

To examine how the proposed architecture contributes to the performance, we took the Ours-DMA-Net$_{\mathrm{stat}}$ as an example and compared the performance of each individual aggregation function (*i.e.*, min, max, median and mean). We took max as an example to present this analytical process (named as max_{ft}). Using the well-trained Ours-DMA-Net$_{\mathrm{stat}}$, we disabled the aggregation functions of min, median and mean, and we fine-tuned the remaining layers for 10 epochs. The same procedure was adopted to achieve min_{ft}, median_{ft} and mean_{ft}. The mAP produced by min_{ft}, max_{ft}, median_{ft} and mean_{ft} are 54.96%, 56.69%, 54.29%, 54.21%, respectively. The results indicate that

Table 3. Image aesthetics categorization on the AVA dataset

Methods	Accuracy
CNN	72.32%
SPP	72.85%
DMA-Net$_{\text{ave}}$	73.1%
DMA-Net$_{\text{max}}$	73.9%
Ours-DMA-Net$_{\text{stat}}$	74.44%
Ours-DMA-Net$_{\text{fc}}$	74.46%
Alexnet-FTune	72.3%
Ours-DMA-Net-ImgFu$_{\text{stat}}$	75.41%
Ours-DMA-Net-ImgFu$_{\text{fc}}$	75.4%
Murray *et al.* [26]	68%

Table 4. Image quality estimation on real-world photos

Methods	Accuracy
CNN	83.7%
SPP	79%
DMA-Net$_{\text{ave}}$	84.7%
DMA-Net$_{\text{max}}$	85.2%
Ours-DMA-Net$_{\text{stat}}$	88.3%
Ours-DMA-Net$_{\text{fc}}$	89.2%
Alexnet-FTune	82.1%
Ours-DMA-Net-ImgFu$_{\text{stat}}$	88.3%
Ours-DMA-Net-ImgFu$_{\text{fc}}$	86.8%

when training Ours-DMA-Net$_{\text{stat}}$, the aggregation function of $\max$ in the statistics layer contributes most among the four and $\mathrm{average}$ contributes the least.

5.3.2 Image Aesthetics Categorization

Table 3 reports results on the AVA dataset for image aesthetics categorization. Following the same trends with image style classification, the proposed DMA-Net approach outperforms CNN[8], SPP, DMA-Net$_{\text{ave}}$ and DMA-Net$_{\text{max}}$. Such results further confirmed our conclusion made in the image style classification that training network on multiple patches generates better prediction performance than network training on a single patch.

ImageNet fusion approach Ours-DMA-Net-ImgFu slightly boosts the performance of Ours-DMA-Net, and significantly performs better than Alexnet-FTune. Such results show that both the global information and fine-grained information are useful for image aesthetics categorization, and the proposed DMA-Net approach captures the fine-grained information in compensate to the global view of images. The proposed DMA-Net also outperforms recent studies of image aesthetics on the AVA dataset [26].

5.3.3 Quality Estimation on Real-World Photos

The results of image quality estimation are presented in Table 4, where several conclusions can be drawn: (i) Multi-patch aggregation network training improves the single-patch network training results: DMA-Net$_{\text{max}}$ improves the CNN by 1.5%, DMA-Net$_{\text{ave}}$ improves the CNN by 1%, and Ours-DMA-Net improves the CNN by $5-6\%$. (ii) The proposed Ours-DMA-Net performs better than DMA-Net$_{\text{ave}}$ and DMA-Net$_{\text{max}}$. (iii) Interestingly, ImageNet feature does not help image quality estimation, as Alexnet-FTune performs worse than CNN, and significantly worse than Ours-DMA-Net, while Ours-DMA-Net-ImgFu performs slightly worse than Ours-DMA-Net without ImageNet feature fusion. Such results indicate that fine-grained information is much more useful than the global view of an image in determining the quality of a real-world photo.

5.4. Computational Efficiency

In a single mini-batch, the computing time for the sorting layer and the statistics layer are negligible in both forward and backward propagation. Bottleneck for training is at the convolutional layers and fully-connected layers.

The training time highly depends on the number of training images and the network architecture. For instance, in quality estimation, with 5,478 high-resolution images and proposed DMA-Net architecture, it took 3-4 days to train from scratch on NVidia Tesla K40. In testing, for instance, for quality estimation, evaluating an image (5 crops per image) using the DMA-Net takes about 8.25 ms on NVidia Tesla K40.

6. Conclusions

This paper proposes novel deep neural network architectures to learn fine-grained details from multiple patches. With the proposed network architecture, multi-patch aggregation functions can be learned as part of neural network training. In particular, we developed two novel network layers (statistics and sorting) and their aggregation strategies to support orderless path aggregation. We evaluated and demonstrated the effectiveness of the proposed networks in image style, aesthetics, and quality estimation on real-world photos. Meanwhile, the proposed deep multiple patch aggregation network model can be directly applied to many other computer vision tasks, such as object category recognition, image retrieval, and scene classification, which we leave as our future work.

References

[1] F. Agostinelli, M. Anderson, and H. Lee. Adaptive multi-column deep neural networks with application to robust image denoising. In *NIPS*, pages 1493–1501. 2013.

[8] CNN for image aesthetics was presented in the [24]. In [24], the results were averaged on 50 patches per image during testing. To ensure the fair comparison, the results we presented were averaged on 50 patches per image during testing (as discussed in Section 5.1).

[2] B. Babenko, N. Varma, P. Dollr, and S. Belongie. Multiple instance learning with manifold bags. In *ICML*, pages 81–88, 2011.

[3] L. Bourdev, S. Maji, and J. Malik. Describing people: A poselet-based approach to attribute classification. In *ICCV*, pages 1543–1550, 2011.

[4] L. Bourdev, F. Yang, and R. Fergus. Deep poselets for human detection. In *arXiv:1407.0717v1*, 2014.

[5] D. Ciresan, U. Meier, and J. Schmidhuber. Multi-column deep neural networks for image classification. In *CVPR*, pages 3642–3649, 2012.

[6] T. Dietterich, R. H. Lathrop, and T. Lozano-Prez. Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89(1-2):31–71, 1997.

[7] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *TPAMI*, 32(9):1627–1645, 2010.

[8] Z. Fu, A. Robles-Kelly, and J. Zhou. MILIS: Multiple instance learning with instance selection. *TPAMI*, 33(5):958–977, 2011.

[9] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, pages 580–587, 2014.

[10] Y. Gong, L. Wang, R. Guo, and S. Lazebnik. Multi-scale orderless pooling of deep convolutional activation features. In *ECCV*, pages 392–407, 2014.

[11] T. Grtner, P. Flach, A. Kowalczyk, and A. Smola. Multi-instance kernels. In *ICML*, pages 179–186, 2002.

[12] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *ECCV*, pages 346–361, 2014.

[13] D. Heckerman. A tractable inference algorithm for diagnosing multiple diseases. In *UAI*, pages 163–171, 1989.

[14] J. Hoffman, D. Pathak, T. Darrell, and K. Saenko. Detector discovery in the wild: Joint multiple instance and representation learning. In *arXiv:1412.1135v1*, 2014.

[15] F. Iandola, M. Moskewicz, S. Karayev, R. Girshick, T. Darrell, and K. Keutzer. DenseNet: Implementing efficient Convnet descriptor pyramids. Technical report, University of California, Berkeley, arXiv:1404.1869v1, 2014.

[16] M. Juneja, A. Vedaldi, C. Jawahar, and A. Zisserman. Blocks that shout: Distinctive parts for scene classification. In *CVPR*, pages 923–930, 2013.

[17] L. Kang, P. Ye, Y. Li, and D. Doermann. Convolutional neural networks for no-reference image quality assessment. In *CVPR*, pages 1733–1740, 2014.

[18] S. Karayev, A. Hertzmann, H. Winnermoller, A. Agarwala, and T. Darrel. Recognizing image style. In *BMVC*, 2014.

[19] J. Keeler, D. Rumelhart, and W. Leow. Integrated segmentation and recognition of hand-printed numerals. In *NIPS*, pages 557–563. 1991.

[20] M. Koskela and J. Laaksonen. Convolutional network features for scene recognition. In *ACM MM*, pages 1169–1172, 2014.

[21] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1106–1114, 2012.

[22] S. Li, Z. Q. Liu, and A. B. Chan. Heterogeneous multi-task learning for human pose estimation with deep convolutional neural network. In *arXiv:1406.3474v1*, 2014.

[23] Z. Lin, G. Hua, and L. Davis. Multiple instance feature for robust part-based object detection. In *CVPR*, pages 405–412, 2009.

[24] X. Lu, Z. Lin, H. Jin, J. Yang, and J. Wang. RAPID: Rating pictorial aesthetics using deep learning. In *ACM MM*, pages 457–466, 2014.

[25] O. Maron and T. Lozano-Prez. A framework for multiple-instance learning. In *NIPS*, pages 570–576. 1998.

[26] N. Murray, L. Marchesotti, and F. Perronnin. AVA: A large-scale database for aesthetic visual analysis. In *CVPR*, pages 2408–2415, 2012.

[27] M. Oquab, L. Bottou, I. Laptev, and J. Sivic. Weakly supervised object recognition with convolutional neural networks. Technical Report HAL-01015140, INRIA, 2014.

[28] G. Papandreou, I. Kokkinos, and P. Savalle. Modeling local and global deformations in deep learning: Epitomic convolution, multiple instance learning, and sliding window detection. In *CVPR*, 2015.

[29] J. Sun and J. Ponce. Learning discriminative part detectors for image classification and cosegmentation. In *ICCV*, pages 3400–3407, 2013.

[30] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf. DeepFace: Closing the gap to human-level performance in face verification. In *CVPR*, pages 1701–1708, 2014.

[31] R. R. Vatsavai. Gaussian multiple instance learning approach for mapping the slums of the world using very high resolution imagery. In *SIGKDD*, pages 1419–1426, 2013.

[32] Y. Wei, W. Xia, J. Huang, B. Ni, J. Dong, Y. Zhao, and S. Yan. CNN: Single-label to multi-label. In *arXiv:1406.5726v3*, 2014.

[33] N. Weidmann, E. Frank, and B. Pfahringer. A two-level learning method for generalized multi-instance problems. In *ECML*, pages 468–479, 2003.

[34] J. Wu, Y. Yu, C. Huang, and K. Yu. Deep multiple instance learning for image classification and auto-annotation. In *CVPR*, 2015.

[35] Y. Xu, T. Mo, Q. Feng, P. Zhong, M. Lai, and E.-C. Chang. Deep learning of feature representation with multiple instance learning for medical image analysis. In *ICASSP*, 2014.

[36] C. Yang and T. Lozano-Perez. Image database retrieval with multiple-instance learning techniques. In *International Conference on Data Engineering*, pages 233–243, 2000.

[37] C. Zhang, J. Platt, and P. Viola. Multiple instance boosting for object detection. In *NIPS*, pages 1417–1424. 2005.

[38] N. Zhang, J. Donahue, R. Girshick, and T. Darrell. Part-based R-CNNs for fine-grained category detection. In *ECCV*, pages 834–849, 2014.

[39] N. Zhang, M. Paluri, M. Ranzato, T. Darrell, and L. Bourdev. PANDA: Pose aligned networks for deep attribute modeling. In *CVPR*, pages 1637–1644, 2014.

[40] Z. Zhou and M. Zhang. Multi-instatnce multi-label learning with application to scene classification. In *NIPS*, pages 1609–1616, 2006.