This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2017.2771766, IEEE Transactions on Pattern Analysis and Machine Intelligence

Learning Consensus Representation for Weak Style Classification

Shuhui Jiang, Ming Shao, *Member, IEEE*, Chengcheng Jia and Yun Fu, *Senior Member, IEEE*

Abstract—Style classification (e.g., Baroque and Gothic architecture style) is grabbing increasing attention in many fields such as fashion, architecture, and manga. Most existing methods focus on extracting discriminative features from local patches or patterns. However, the *spread out* phenomenon in style classification has not been recognized yet. It means that visually less representative images in a style class are usually very diverse and easily getting misclassified. We name them *weak style* images. Another issue when employing multiple visual features towards effective weak style classification is lack of consensus among different features. That is, weights for different visual features in the local patch should have been allocated similar values. To address these issues, we propose a Consensus Style Centralizing Auto-Encoder (CSCAE) for learning robust style features representation, especially for weak style classification. First, we propose a Style Centralizing Auto-Encoder (SCAE) which centralizes weak style features in a progressive way. Then, based on SCAE, we propose both the non-linear and linear version CSCAE which adaptively allocate weights for different features during the progressive centralization process. Consensus constraints are added based on the assumption that the weights of different features of the same patch should be similar. Specifically, the proposed linear counterpart of CSCAE motivated by the "shared weights" idea as well as group sparsity improves both efficacy and efficiency. For evaluations, we experiment extensively on fashion, manga and architecture style classification problems. In addition, we collect a new dataset—Online Shopping, for fashion style classification, which will be publicly available for vision based fashion style research. Experiments demonstrate the effectiveness of the SCAE and CSCAE on both public and newly collected datasets when compared with the most recent state-of-the-art works.

Index Terms—Style classification, deep learning, auto-encoder

1 Introduction

Style classification, such as the architectural style [1], [2], the manga style [3] and the fashion style [4] attracts increasing attention from researchers in vision and machine learning fields. Style classification is related, but essentially different from most existing classification tasks. A particular clothing type is generally made up of a diverse set of fashion styles. For example, "suit" may have both *elegant* and *renascent* fashion styles while "dress" may have both *romantic* and *young* fashion styles. Thus, automatically learning robust and discriminative representation for style classification becomes a critic research topic.

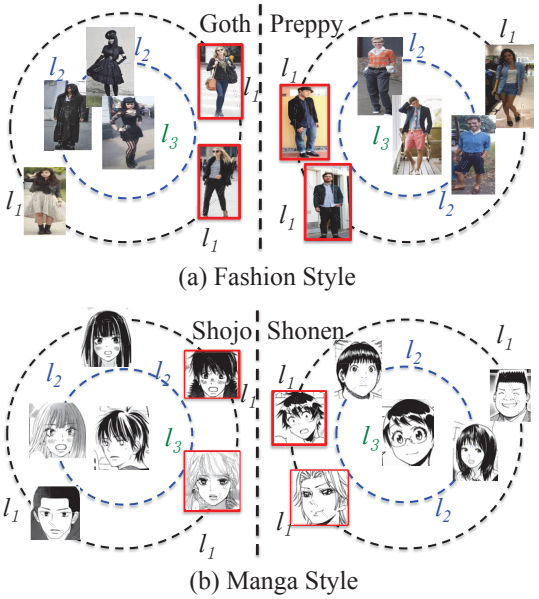
Most existing style classification methods mainly focused on extracting discriminative local patches or patterns from low-level features [2], [3], [4], [5], [6], [7]. However, none of these style classification works has discussed the *spread out* phenomenon—an important observation of style images. Style images from the same class are usually diffuse since the style is reflected by the high-level abstract concepts. Figure 1 illustrates the spread out phenomenon. Take fashion style classification of "Goth" and "Preppy" as an example. Representative images in the center of each class are assigned *strong style* level l_3, where we can easily distinguish "Goth" and "Preppy" images, while less representative images distant to the center are assigned lower style level l_1. We name them as *weak style* images. Weak style images within one class may be visually diverse, and images in two different classes may be visually similar (shown in red frames). Thus, weak style images easily get mis-classified. To better illustrate this phenomenon, in Figure 2, we visualize the distributions of two kinds of feature descriptors of manga images including both shojo and shonen classes [3]. Note that PCA is employed to reduce the dimensionality of the descriptors for visualization. In this figure, strong style data points (in blue) are dense and well separated while weak style data points (in magenta) are diffuse with a vague boundary.

Employing multiple visual descriptors for effective representations for weak style images is a possible solution; however, jointly learning weights for different visual features in representation learning is challenging. The most common pitfall is failing to consider the "consensus" between different visual descriptors, especially in local patches. For example, if one patch from the image is critical for discrimination, we expect higher weights of this patch for all the features, meaning a consistency of weights across different feature descriptors. However, such a consensus constraint is rarely considered in weak style image related problems.

To address the issues above, in this paper, we propose a Consensus Style Centralizing Auto-Encoder (CSCAE) for robust style feature extraction for weak style classifica-

- *S. Jiang, C. Jia are with the Department of Electrical and Computer Engineering, Northeastern University, Boston, MA 02115 USA (e-mail: shjiang@ece.neu.edu, cjia@ece.neu.edu)*
- *M. Shao is with the Department of Computer Information Science, University of Massachusetts Dartmouth, Dartmouth, MA 02747 USA (email: mshao@umassd.edu)*
- *Y. Fu is with the Department of Electrical and Computer Engineering,College of Engineering, and College of Computer and Information Science, Northeastern University, Boston, MA 02115 USA (e-mail: yunfu@ece.neu.edu).*
- *This work is supported in part by the NSF IIS award 1651902, NSF CNS award 1314484, ONR award N00014-12-1-1028, ONR Young Investigator Award N00014-14-1-0484, and U.S. Army Research Office Young Investigator Award W911NF-14-1-0218.*

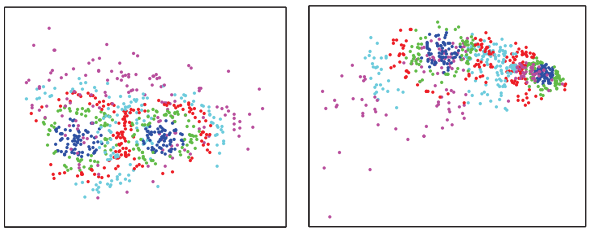
0162-8828 (c) 2017 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.



(a) Fashion Style

(b) Manga Style

Figure 1. Illustration of *weak style* phenomenon in fashion and manga style images in (a) and (b) respectively. Images in red frames on the boundary are weak style images from two different classes, but they seem visually similar. We denote l_1 to l_3 as different style levels.



(a) (b)

Figure 2. Data visualization of the "spread out" phenomenon in "shoji" and "shonen" classes of manga style. PCA is conducted to reduce the dimension of feature descriptors of "line strength" and "included an angle between lines" into 2D for visualization in (a) and (b) respectively. In both (a) and (b), five colors, blue, green, red, cyan, and magenta, are used to present the data points in different style levels from the strongest to the weakest. We could see that strong style data points are dense and weak style data points are diffuse.

tion. First, we propose a Style Centralizing Auto-Encoder (SCAE), which progressively pulls weak style images back to the class center. The inputs of SCAE are concatenated low-level features from all the local patches of an image (e.g., eyes, nose, mouth patches in a face image). As shown in Figure 3, for each Auto-Encoder (AE), the corresponding output feature is the same type of feature in the same class but one style level stronger than the input feature. When piling all the AEs together, we can progressively "convert" the weak style features to strong style features, which is represented by the hidden units of this deep structure as the smooth transition. In addition, to approach the "lack of consensus" issue among different kinds of visual features, we jointly learn weights for them by a novel AE called the Consensus SCAE (CSCAE). We propose both non-linear and linear versions of CSCAE, and explore their efficiency and effectiveness. Most importantly, on the strength of the proposed Rank-Constrained Group Sparsity Auto-Encoder (RCGSAE), the linear CSCAE outperforms the non-linear one in terms of both efficacy and efficiency. We evaluate our methods based on three applications: fashion style classification, manga style classification, and architecture style classification. We also collect a new Online Shopping dataset and compare our methods with the most recent state-of-the-art works on it. On both our newly collected and public datasets, our methods achieve encouraging results. The novelties of our paper could be summarized as:

- To the best of our knowledge, this is the first time when weak style classification problem has been identified in machine learning and vision communities as a representation learning problem;
- We propose a Style Centralizing Auto-Encoder (SCAE) to progressively pull weak style images back to the center of the class to learn robust and discriminative feature representation;
- We propose both non-linear and linear CSCAE to jointly learn multiple SCAEs for different types of features under consensus constraints. Specifically, a novel Rank-Constrained Group Sparsity Auto-Encoder (RCGSAE) has been designed as the counterpart of the non-linear CSCAE, and its fast solution yields better performance in terms of both efficacy and efficiency than its non-linear predecessor.

This paper is an extension of our previous conference work [8]. There are three major differences compared to the conference version. First, we propose a linear CSCAE to reduce the computational cost of the non-linear CSCAE yet provide a better performance. To that end, we identify an approximate model Rank-Constrained Group Sparsity Auto-Encoder (RCGSAE) which shares the motivations of the non-linear CSCAE, and then find its fast solution to achieve the acceleration purpose. To our best knowledge, this is the first time when RCGSAE is invented to approach weak style images classification in a fast manner. Second, we add a new application in the experiment—architecture style classification to extend the range of the application of our model (Table 4). Third, we add more data visualizations, discussions, and comparisons to better illustrate and explain our method (Figure 2,6–15, Table 4–9).

2 Related Work

Style classification such as the architectural style [1], [2], [5], [9], the music style [6], [7], [10], the photographic style [11], the manga style [3] and the fashion style [4], [12], [13], [14] attracts increasing attention recently. In this section, we mainly introduce the fashion, manga, and architecture style classification and their most recent state-of-the-art works.

Fashion Style: Fashion style classification essentially ties to the clothing parsing [15] and clothing recognition [16]. Bossard et al. introduced a system framework to recognize and classify people by their clothing in natural scenes [12]. They first densely extracted a number of feature descriptors like HOG in the bounding box of the upper body followed by a bag-of-words model [12]. Hipster Wars is mostly related to our work [4]. In this work, Kiapour et al. proposed an online game to collect a fashion dataset and a new style descriptor. First, a style descriptor was formed by

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2017.2771766, IEEE Transactions on Pattern Analysis and Machine Intelligence

accumulating visual features like color and texture. Then they applied mean-std pooling and concatenated all the pooled features as the final style descriptor, followed by a linear SVM for classification.

Manga Style: Recently, efforts have been made in manga style classification. Chu et al. [3] paved the way for manga style classification which discriminates mangas targeting at young boys and young girls named "shonen" and "shojo", respectively. They designed both explicit (e.g., the density of line segments) and implicit (e.g., included angles between lines) features, which are also promising in discriminating artworks produced by different artists.

Architecture Style: Another good instance of weak style in vision problems is the architectural style. Goel et al. mined characteristic features with the semantic utility from the low-level feature for different architectural styles [9]. These characteristic features are of various scales and provide an insight into what makes a particular architectural style category distinct. Van et al. created correspondences across images by a generalized spatial pyramid matching scheme [11]. They assumed that images within a category share a similar style defined by attributes such as colorfulness and lighting. Xu et al. adopted Deformable Part-based Models (DPM) to capture the morphological characteristics of basic architectural components [1].

In this paper, different from them, we propose a novel deep representation learning framework towards weak style image classification and use the three applications above to demonstrate the generality and superiority of our framework. In addition, we for the first time design a consensus mechanism to synchronize different visual features in the deep structure and develop a fast solution for efficiency while keeping competitive performance in style image classification. To our knowledge, these have never been discussed in the previous style image classification research, as they either targeted at a specific weak style problem or simply used the visual feature(s) or their concatenations without considering the correlations among them.

3 Style Centralizing Auto-Encoder

Deep Auto-Encoders (AEs) have been exploited to learn discriminative feature representation [17], [18], [19]. Conventional AEs [17] include two parts: (1) encoder, (2) decoder. An encoder attempts to map the input feature to the hidden layer representation by a linear transform and a successive nonlinear activation function $f(\cdot)$:

$$z_i = f(x_i) = \sigma(W_1 \times x_i + b_1), \tag{1}$$

where $x_i \in \mathbb{R}^D$ is the input feature, $z_i \in \mathbb{R}^d$ is the hidden layer representation, $W_1 \in \mathbb{R}^{d \times D}$ is a linear transform, $b_1 \in \mathbb{R}^d$ is the bias, and σ is the non-linear activation (e.g., sigmoid function). The decoder $g(\cdot)$ manages to map the hidden representation z_i back to the input feature x_i, namely,

$$x_i = g(z_i) = \sigma(W_2 \times z_i + b_2), \tag{2}$$

where $W_2 \in \mathbb{R}^{D \times d}$ is a linear transform. $b_2 \in \mathbb{R}^D$ is the bias.

To optimize the model parameters W_1, b_1, W_2 and b_2, we minimize the following least square error:

$$\min_{\substack{W_1,b_1 \\ W_2,b_2}} \frac{1}{2N} \sum_{i=1}^{N} \left\| x_i - g(f(x_i)) \right\|^2 + \lambda R(W_1, W_2), \tag{3}$$

where N is the number of data points, $R(W_1, W_2) = (\|W_1\|_F^2 + \|W_2\|_F^2)$ works as a regularizer, $\| \cdot \|_F^2$ is the Frobenius norm, and λ is the weight decay parameter to suppress arbitrarily large weights.

A Stacked Auto-Encoder (SAE) [20], [21] stacks multiple AEs to form a deep structure. It feeds the hidden layer of the k-th AE as the input feature to the $(k+1)$-th layer. However, in the weak style classification problem, the performance of AE or SAE degrades due to the "spread out" phenomenon. The reason is that the conventional AE or SAE runs in an unsupervised fashion to learn mid/high-level feature representation, meaning there is no guidance to lead images in the same class close and images in the different classes far away to each other. This is very similar to the conventional PCA (SAE can be seen as a multi-layer non-linear PCA). In Figure 2 where data are illustrated after PCA, weak-style classes represented by cyan and magenta are diffused and overlap with other classes, from which we can see that the mid/high-level feature representation by AE or SAE will suffer from the "spread out" phenomenon. To address these issues, we propose a novel Style Centralizing Auto-Encoder (SCAE).

We start from building local visual features as the input for SCAE. Assume that there are N images from N_c style classes, and $x_i (i \in \{1, ..., N\})$ is the feature representation of the ith image. For each image, we first divide it into several patches (e.g., eyes, nose and mouth patches in a face image). Then we extract visual features (e.g., HoG, RGB, Gabor) from each patch. For each feature descriptor (e.g., HoG), we concatenate extracted features from all the patches as one part of input features for SCAE. By concatenating all different visual features, we obtain the final input features for SCAE. In addition, each image is assigned a style level label. Intuitively, representative images of each style are usually assigned the strong style level, while less representative images are assigned the weak style level. In this paper, we use L distinct style levels denoted as $\{l_1, l_2, ..., l_k, ..., l_L\}$ from the weakest to the strongest.

Different from the conventional AE taking identical input and output features, the input and output of our SCAE are different. Illustration of the full pipeline of SCAE can be found in Figure 3. Suppose that we have $L = 4$ style levels, and the inputs of the SCAE in the first layer are the features of images in the ascent order of style level, namely, X_1, X_2, X_3 and X_4. For example, X_2 is the set of features of images in style level l_2. In the step k, we seek for the AE that handles the following mappings:

$$\{X_k, X_{k+1}, ...X_j, X_L\} \rightarrow \{X_{k+1}, X_{k+1}, ...X_j, X_L\}, \tag{4}$$

where only X_k is pulled towards stronger style level $k+1$, and others keep the same style level before and after the k-th step. In this way, the weak style level will be gradually pulled towards the strong style level, i.e., centralization, till $k = L - 1$. Thus, the $L - 1$ stacked AEs embody the proposed stacked SCAE. Note that the mappings between

0162-8828 (c) 2017 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

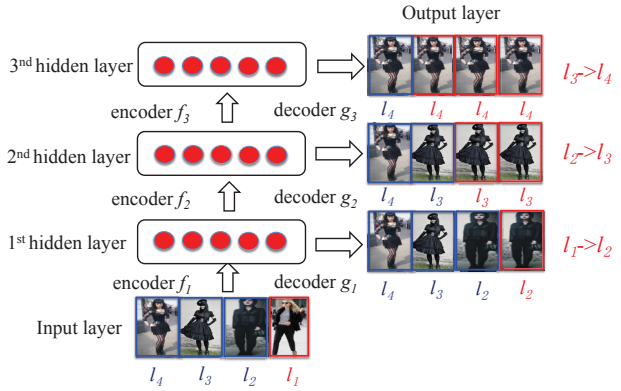


Figure 3. Illustration of the Stacked Style Centralizing Auto-Encoder (SSCAE). Example images of each level l are presented with colored frames. In the step k, samples in l_k are replaced by l_{k+1} samples' (red) nearest neighbors found in l_k. Samples in the higher level than l_k are not changed (blue).

Algorithm 1 Stacked Style Centralizing Auto-Encoder

INPUT: Non-linear Style feature X including weak style feature.

OUTPUT: Style centralizing feature Z_k, model parameters: $W_{1,k}^{(c)}$, $W_{2,k}^{(c)}$, $b_{1,k}^{(c)}$, $b_{2,k}^{(c)}$, $k \in [1, L-1]$, and $c \in \{1, ..., N_c\}$.

1: Initial $Z^{(0)} = X$.
2: **for** k=1,2,...,L-1 **do**
3: $X^{(k)}$=$Z^{(k-1)}$.
4: **for** c = 1,..., N_c **do**
5: Calculate $W_{1,k}^{(c)}$, $W_{2,k}^{(c)}$, $b_{1,k}^{(c)}$, $b_{2,k}^{(c)}$ by Eq. (6).
6: Calculate $Z_k^{(c)}$ by Eq. (1).
7: **end for**
8: Combine all $Z_k^{(c)}, c \in \{1, ..., N_c\}$ into Z_k.
9: **end for**

X_k and X_{k+1} are still unclear. To keep the style level transition smooth, for each output feature $x \in X_{k+1}$, we find its nearest neighbor in X_k in the same style class as the corresponding input to learn SCAE. The whole process is shown in Figure 3.

Next, we explain how to build the Stacked SCAE (SSCAE). Suppose that we have L style levels, in the k-th layer and style class c, the corresponding input for the output $x_{i,\xi+1}$ is given by:

$$\tilde{x}_{i,\xi}^{(c)} = \begin{cases} x_{j,\xi}^{(c)} \in u(x_{i,\xi+1}^{(c)}), & \text{if } \xi = k, \\ x_{i,\xi}^{(c)}, & \text{if } \xi = k+1, ...L, \end{cases} \tag{5}$$

where $u(x_{i,\xi+1}^{(c)})$ is the set of nearest neighbors of $x_{i,\xi+1}^{(c)}$.

As there are N_c style classes, in each layer, we first separately learn parameters and hidden layer features $Z_k^{(c)}$ of SCAE of each class, and then combine all the $Z_k^{(c)}$ together as Z_k. Mathematically, SSCAE can be formulated as: the k-th layer of SCAE for category c can be written as:

$$\min_{\substack{W_{1,k}^{(c)}, b_{1,k}^{(c)} \\ W_{2,k}^{(c)}, b_{2,k}^{(c)}}} \sum_{i,j \quad x_{j,k}^{(c)} \in u(x_{i,k+1}^{(c)})} \left\| x_{i,k+1}^{(c)} - g(f(x_{j,k}^{(c)})) \right\|^2 + \sum_{\xi=k+1}^{L} \sum_{i} \left\| x_{i,\xi}^{(c)} - g(f(x_{i,\xi}^{(c)})) \right\|^2 + \lambda R(W_{1,k}^{(c)}, W_{2,k}^{(c)}). \tag{6}$$

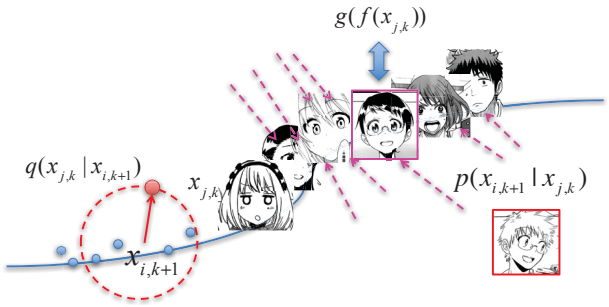


Figure 4. Manifold learning perspective of SCAE with mangas in "shonen" style.

The problem above can be solved in the similar way as the conventional AE by back propagation algorithms [22]. Similarly, the deep structure can be built in a layer-wise way, which is outlined in the Algorithm 1. The visualization of encoded feature in SSCAE in each progressive step is shown in Figure 12 in the experimental section.

Geometric Interpretation of SCAE: Recalling the mappings presented in Eq. (3), if we consider X_k as a corrupted version of X_{k+1}, our SCAE can be recognized as a Denoising Auto-Encoder (DAE) [20] that uses partially corrupted feature as the input and the clean noise free features as the output to learn robust representations [20], [23]. Thus, inspired by the geometric perspective under the manifold assumption [24], we may offer a geometric interpretation for the SCAE, analogical to that of DAE [20].

Figure 4 illustrates the manifold learning perspective of SCAE where images of shonen style mangas are shown as the examples. Suppose that the higher level (l_{k+1}) mangas lie close to a low dimensional manifold. The weak style examples are more likely being far away from the manifold than the higher level ones. Note that $x_{j,k}$ is the corrupted version of $x_{i,k+1}$ by the operator $q(X_k|X_{k+1})$, and therefore lies far away from the manifold. In SCAE, $q(X_k|X_{k+1})$ manages to find the nearest neighbor of $x_{i,k+1}^{(c)}$ in level l_k to obtain the corrupted the version of $x_{i,k+1}^{(c)}$ as $x_{j,k}^{(c)}$. During the centralizing training, similar to DAE, SCAE learns the stochastic operator $p(X_{k+1}|X_k)$ that maps the lower style level samples X_k back to a higher level. Successful centralization implies that the operator $p(\cdot)$ is able to map spread-out weak style data back to the strong style data which are close to the manifold.

4 Non-Linear Consensus Style Centralizing Auto-Encoder

Given multiple low-level visual descriptors (e.g., HOG, RGB, Gabor), existing methods treat them equally and concatenate them to formulate the final representation [3], [4]. Thus, they failed to consider the correlation between different kinds of features, i.e., consensus.

Intuitively, the weights of two features from the same patch should be similar, as they encode the same visual information, but in different ways. Taking face recognition as an example, the eyes patch should be more important than the cheek patch, as demonstrated by many face recognition works. Back to our consensus model, taking manga

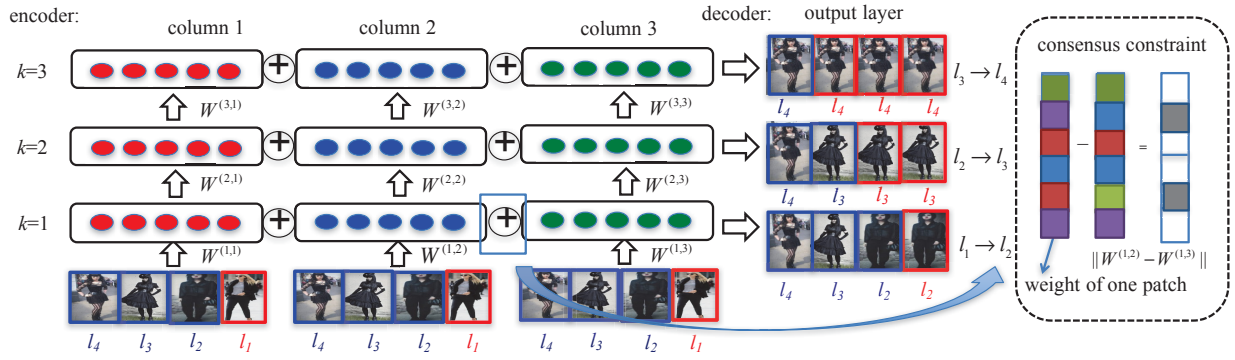


Figure 5. Illustration of Non-linear Consensus Style Centralizing Auto-Encoder (NCSCAE).

as an example, given different kinds of features used by different SCAEs, the eyes patches in different SCAEs should be equally important. To that end, we propose to add a consensus constraint through minimizing the differences of weights of the same patch from different feature descriptors. In this section, following framework of SCAE, we introduce the Non-linear Consensus Style Centralizing Auto-encoder (NCSCAE) first, to differentiate the linear version of CSCAE (LCSCAE) introduced in the later section.

To clarify, each SCAE in the CSCAE is responsible for one particular visual descriptor. Thus, similar to Section 3, for each image, we first divide it into several patches (e.g., eyes, nose and mouth patches in a face image). Then we extract low-level features (e.g., HoG, RGB, Gabor) from each patch. For each feature descriptor (e.g., HoG), we concatenate extracted features from all the patches, which serves as the input for each SCAE. Afterwards, multiple SCAEs will jointly work to learn CSCAE.

In practice, in the k-th step, we set the dimensions of hidden layers of all the SCAEs to the same value. Thus, when $k > 1$, the dimensionality of weight matrices in different SCAEs, e.g., $W^{(k,\mu)}$ and $W^{(k,\nu)}$ are same, where μ and ν index different visual features. We may expect to minimize the following value: $\|W^{(k,\mu)} - W^{(k,\nu)}\|_F$.

When $k = 1$, however, since the dimensionality of the input features may be different, the operation $\|W^{(1,\mu)} - W^{(1,\nu)}\|_F$ may not be valid anymore. Thus, we could not directly conduct the subtraction in Eq. (8). Instead, we calculate the mean value of the weight of each patch first. Then, we use the mean weight of each patch and their concatenation instead of the original weight matrix to measure the difference between weight matrices of two different kinds of feature descriptors.

Figure 5 illustrates the framework of NCSCAE. Each column (red, blue, green) represents one SCAE, as shown in Figure 3. Different feature channels (columns) are trained jointly by adding a consensus constraint, which can be found in the right part of Figure 5. Each cell represents one patch, and their colors indicate the patch weights. We use light color for small values while dark color for large values. Take column 2 and column 3 in step $k = 1$ as an example, $\|W^{(1,2)} - W^{(1,3)}\|_F$ measures the difference between the weights of the 2nd column and the 3rd column. Following the consensus assumption, the matched cells of $W^{(1,2)}$ and $W^{(1,3)}$ either both have large values, or both have small values, leading to small values in each cell of the vector $W^{(1,2)} - W^{(1,3)}$.

To incorporate the constraint above into our SCAE model, which would be equivalent finding a sparse solution for $W^{(1,2)} - W^{(1,3)}$, we propose to add a sparse constraint to the SCAE model. There are several popular ways to model the "sparsity" in the Auto-Encoder, which have been comprehensively discussed in [25]. In light of this, we use the KL-divergence to provide the stable performance by minimize the following:

$$KL(\hat{\rho}\|\rho) = \rho \log \frac{\rho}{\hat{\rho}} + (1-\rho) \log \frac{1-\rho}{1-\hat{\rho}}, \tag{7}$$

where ρ is a model parameter (usually a very small value), and $\hat{\rho}$ is the average patch weight difference over SCAEs of different feature descriptors, calculated as:

$$\hat{\rho}_k = \frac{1}{N_f \times (N_f - 1)} \sum_{\mu > \nu} \|W^{(k,\mu)} - W^{(k,\nu)}\|_F^2, \tag{8}$$

where N_f is the number of feature channels, and $\|\cdot\|_F$ indicates the matrix Frobenius norm.

The KL-divergence in Eq. (7) essentially measures the divergence between a Bernoulli random variable with the mean ρ and a Bernoulli random variable with the mean $\hat{\rho}$. By minimizing Eq. (7), we ensure that the weights for different features (different columns) on the same patch are similar from statistics perspective.

Thus, the new loss function of NCSCAE in the k-th step becomes:

$$J_k' = \sum_{\mu=1}^{N_f} J_{\mu,k} + \beta KL(\hat{\rho}_k\|\rho), \tag{9}$$

where $J_{\mu,k}$ is the loss of the SCAE of the μth feature descriptor in the k-th progressive step. β is a balance parameter. Essentially, the left part is the sum of loss function of different SCAEs in the k-th step illustrated in Eq. (6), and the right part is the regularizer for column consensus discussed above.

4.1 Solutions

Here we describe how to solve the objective function proposed in Eq. (9). Although we still use stochastic gradient descent + back propagation for solutions, we have to jointly learn multiple SCAEs for different features. Thus, the basic

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2017.2771766, IEEE Transactions on Pattern Analysis and Machine Intelligence

gradient updating rules for model parameters $W^{(k,\mu)}$, $b^{(k,\mu)}$ become:

$$W_{i,j}^{(k,\mu)} := W_{i,j}^{(k,\mu)} - \frac{\partial}{\partial W_{i,j}^{(k,\mu)}} J_k^{'}(W,b), \tag{10}$$

$$b_{i}^{(k,\mu)} := b_{i}^{(k,\mu)} - \frac{\partial}{\partial b_{i}^{(k,\mu)}} J_k^{'}(W,b), \tag{11}$$

where j and i index the input and hidden layer nodes of the k-th layer, respectively. $W_{i,j}^{(k,\mu)}$ and $b_i^{(k,\mu)}$ are the elements of weight matrix and bias vector.

The key procedure is finding the partial derivatives $\frac{\partial}{\partial W_{i,j}^{(k,\mu)}} J_k^{'}(W,b)$ and $\frac{\partial}{\partial b_i^{(k,\mu)}} J_k^{'}(W,b)$ by the back propagation algorithm. To facilitate the following derivation, we introduce an intermediate parameter $\delta_i^{(k,\mu)}$ which is the derivative of the loss function w.r.t. to the output of layer k. This is straightforward for the layer L, but not for the layer ($k < L$). Thus, we use the following formula for the deduction of the reconstruction loss $J_k(W,b)$:

$$\delta_i^{(k,\mu)} = \Big(\sum_j W_{ji}^{(k,u)} \delta_j^{(k+1,\mu)} \Big) f'(z_i^{(k,\mu)}). \tag{12}$$

In NCSCAE, since the consensus constraint $KL(\hat{\rho}\|\rho)$ is added, we obtain partial derivatives as:

$$\frac{\partial}{\partial W_{i,j}^{(k,\mu)}} J_k^{'}(W,b) = a_j^{(k,\mu)} \delta_i^{(k+1,\mu)} + \Big(-\frac{\rho}{\hat{\rho}} + \frac{1-\rho}{1-\hat{\rho}} \Big) \hat{\rho}'(W^{(k,\mu)}), \tag{13}$$

$$\frac{\partial}{\partial b_i^{(k,\mu)}} J_k^{'}(W,b) = \delta_i^{(k+1,\mu)}, \tag{14}$$

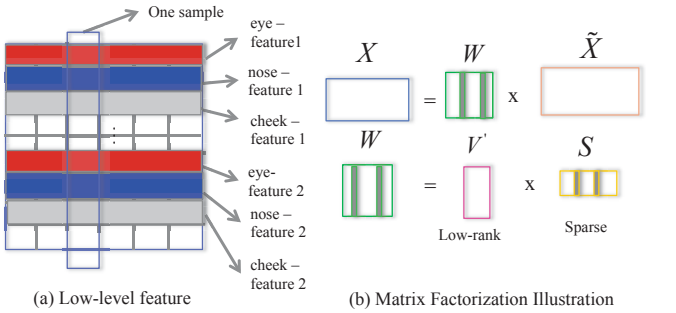
where $a_j^{(k,\mu)} = z_j^{(k-1,\mu)}$ is the j-th input node of μ-th column of the k-th layer.

We solve the problem above using the MATLAB implementation of the L-BFGS optimizer [26], [27] since it can solve the large-scale problems using the limited memory. As the weights of different kinds of feature descriptors are learned under the consensus constraints, we do not manually adjust weights. Afterwards, we use hidden layer z_i as the learned representation for the style features. To train the deep model in a more efficient way, we employ the layer-wise training procedure. All hidden layers from the deep model are stacked to formulate the final style representation.

5 Linear Consensus Style Centralizing Auto-Encoder

The downside of NCSCAE is the large computational burden given complex data (e.g., high dimensionality). Recently, the efficient Linear Denoising Auto-Encoder (LDAE) and its approximation cast a light in real world applications with competitive performance [28]. This motivates us to seek for a linear counterpart of our NCSCAE.

However, a direct linear modeling of NCSCAE following conventional methods such as [23] is infeasible and trivial due to the introduction of the additional regularizer in Eq. (7). To that end, we first find a conceptually equivalent problem for the original NCSCAE which is easy to linearize, and then find its efficient solutions. We will detail these two steps in Section 5.1 and 5.2.



(a) Low-level feature (b) Matrix Factorization Illustration

Figure 6. Illustration of the Linear Consensus Style Centralizing Auto-Encoder (LCSCAE). The illustration of the low-rank group sparsity structure of low-level features is shown in (a). The illustration of the matrix factorization in the solution of the model is shown in (b).

5.1 NCSCAE Reframing

5.1.1 Low-rank Constraint for Consensus

In CSCAE, we expect similar weights across different feature descriptors for the same patch. This will give rise to an interesting phenomenon: if we concatenate all the weight matrices of different feature descriptors together denoted as W, the rank of W should be low. The main reason is the similar values across different columns in W. Thus, instead of enforcing the consensus constraint directly on the weight matrices, we can pursue this through a low-rank matrix constraint on the concatenated weight matrix W.

To that end, we propose a Rank-Constrained Auto-Encoder model to pursue the low-rankness of the weight matrix W in the following formula:

$$\min_{W,E} \|W\|_* + \lambda \|E\|_{2,1}, \ \text{ s.t., } X = W\tilde{X} + E, \tag{15}$$

where $\tilde{X}$ is the input feature and X is the output feature of AE, similar to SCAE. $\|\cdot\|_*$ is the nuclear norm of a matrix used as the convex surrogate of the original rank constraint, $\|E\|_{2,1}$ is the matrix $\ell_{2,1}$ norm for characterizing the sparse noise E, and λ is a balancing parameter. Intuitively, the residual term E encourages sparsity, as both W and $W\tilde{X}$ are low-rank matrices. This is also well explored by many low-rank recover/representation works [29], [30]. The Eq. (15) can also be considered as a special form of the work [29] by only considering features in the column space.

Figure 6 (a) illustrates this phenomenon. Each row represents a specific feature of one patch, and different colors indicate different kinds of features. In addition, each column represents all features of one sample. It can be seen that the concatenated features for one sample is formulated by first stacking different features of the same patch, and then stacking different patches. Note that for simplicity, we use one cell to represent one kind of features of one patch. In brief, based on the definition of the matrix rank, the consensus constraint among different features induces the low-rank structure of matrix W identified in Figure 6 (a). And we may employ Eq. (15) as a conceptually equivalent problem for NCSCAE.

5.1.2 Group Sparsity Constraint for Consensus

To further consider the regularizers introduced in Eq. (3) under the new Rank Constraint Auto-Encoder framework,

0162-8828 (c) 2017 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2017.2771766, IEEE Transactions on Pattern Analysis and Machine Intelligence

we introduce an additional group sparsity constraint on W. The reasons are three-fold. First, like conventional regularizers in neural networks, it helps avoid the arbitrarily large magnitude in W. Second, it enforces the row-wise selection on W to ensure a better consensus effect together with the low-rank constraint. Third, it helps find the most discriminative representation.

Mathematically, we can achieve this by adding a matrix $\ell_{2,1}$ norm $\|W\|_{2,1} = \sum_{i=1}^{D} \|W(i)\|_2$ which is equal to the sum of the Euclidean norms of all columns of W. The ℓ_2 norm constraint is applied to each group separately (i.e., each column of W). It ensures that all elements in the same column are either approaching zero or nonzero at the same time. The ℓ_1 norm guarantees that only a few columns are nonzero. Figure 6 (a) also illustrates how the group sparsity works. If the entry in the j-th row and i-th column of X indicates an unimportant patch, all the entries from the j-th row are also less important, and vice-versa. As discussed above, these patches should have been assigned very low or zero weights to suppress the noise.

5.1.3 Rank-Constrained Group Sparsity Auto-Encoder

Considering both rank and group sparsity constraints, we finally formulate the objective function of the Rank-Constrained Group Sparsity Auto-Encoder (RCGSAE) as:

$$\hat{W}_r = \underset{\text{rank}(W)\leq r}{\arg\min} \{||X - W\tilde{X}||_F^2 + 2\lambda\|W\|_{2,1}\}, \tag{16}$$

where λ is the balancing parameter. Note that we skip the sparse error term in Eq. (15) for simplicity. Clearly, for $r = D$, we have no rank constraint in Eq. (16) which degrades to a group sparsity problem, while for $\lambda = 0$, we obtain the reduced-rank regression estimator. Thus, an appropriate rank r and λ will balance the two parts to yield better performance.

In summary, the equivalent problem above can be considered as the linear counterpart of the NCSCAE as we do not include the non-linear activation and the back-propagation process anymore. Thus, we name it as the Linear Consensus Style Centralizing Auto-Encoder (LCSCAE) and use this term for the following sections.

5.2 Efficient Solutions

It should be noted that the problem defined in Eq. (16) is non-convex and has no close form solutions for W. Thus, we propose to use an iterative algorithm to solve this in a fast manner. As shown in Figure 6 (b), we first factorize W into $W = V^{'}S$, where V is a $r \times D$ orthogonal matrix and S is a $r \times D$ matrix with the group sparse constraint [31]. Then the optimization problem of W in Eq. (16) turns out to be:

$$(\hat{S}, \hat{V}) = \underset{S\in\mathbb{R}^{r\times D}, V\in\mathbb{R}^{r\times D}}{\arg\min} \{||X - V'S\tilde{X}||_F^2 + 2\lambda||S||_{2,1}\}. \tag{17}$$

The details of the algorithm are outlined in Algorithm 2. In addition, the following theorem presents a convergence analysis for Algorithm 2 and ensures that the algorithm converges well regardless of the initial points.

Theorem 1. Given λ and an arbitrary starting point $V_{r,\lambda}^{(0)} \in \mathbb{O}^{r\times D}$, let $(S_{r,\lambda}^{(j)}, V_{r,\lambda}^{(j)})$ (j = 1, 2, . . .) be the sequence of iterates generated by Algorithm 2. Then, any accumulation point of $((S_{r,\lambda}^{(j)}, V_{r,\lambda}^{(j)})$ is a coordinatewise minimum point (and a stationary point) of F and $F(S_{r,\lambda}^{(j)}, V_{r,\lambda}^{(j)})$ converges monotonically to $F(S_{r,\lambda}^{*}, V_{r,\lambda}^{*})$ for some coordinate-wise minimum point $(S_{r,\lambda}^{*}, V_{r,\lambda}^{*})$.

Algorithm 2 Solutions for LCSCAE

INPUT: Original style feature X, corrupted style feature $\tilde{X}$, $1 \leq r \leq N \wedge D$, $\lambda \geq 0$, $V_{r,\lambda}^{(0)} \in \mathbb{O}^{n\times r}$, $j \leftarrow 0$, converged $\leftarrow$ FALSE

OUTPUT: Model parameters: W.

1: **while** not converged **do**
2: (a) $S_{r,\lambda}^{(j+1)} \leftarrow \arg\min_{S\in R^{p\times r}} \frac{1}{2}||V_{r,\lambda}^{(j)}X - S\tilde{X}||_F^2 + \lambda||S||_{2,1}$.
3: (b) Let $Q \leftarrow S_{r,\lambda}^{(j+1)}\tilde{X}X' \in \mathbb{R}^{n\times r}$ and perform SVD: $Q = U_w D_w V_w^{'}$.
4: (c) $V_{r,\lambda}^{(j+1)} \leftarrow U_m V_w^{'}$.
5: (d) $W_{r,\lambda}^{(j+1)} \leftarrow (V_{r,\lambda}^{(j+1)})' S_{r,\lambda}^{(j+1)}$.
6: (e) converged $\leftarrow |F(W_{r,\lambda}^{(j+1)}; \lambda) - F(W_{r,\lambda}^{(j)}; \lambda)| < \varepsilon$
7: (f) $j \leftarrow j + 1$
8: **end while**
9: $\hat{W}_{r,\lambda} = W_{r,\lambda}^{(j+1)}$, $\hat{S}_{r,\lambda} = S_{r,\lambda}^{(j+1)}$, $\hat{V}_{r,\lambda} = V_{r,\lambda}^{(j+1)}$.

The proof can be referred to Appendix A.7 of [31].

Discussions. To show the intuitions behind the new LCSCAE, we summarize the connections and differences between the proposed LCSCAE and NCSCAE: (1) Although LCSCAE and NCSCAE are not mathematically equivalent, they share similar motivations, and thus may achieve similar performance for weak-style classification tasks. Similar thoughts can be extended to other vision problems. (2) The group sparsity introduced in the LCSCAE dose work well and helps find out the most discriminative features by penalizing irrelevant features in group, which accounts for the better performance over NCSCAE in the experimental section. (3) The Linear Denoising Auto-Encoder (LDAE) [28] which constitutes the building block of LCSCAE, as well as the novel solution to the low-rank and group sparsity problem can approach the original problem in an efficient manner, as shown in Table 8. A more robust AE model and efficient solution in the future may further benefit our conversion. (4) Our linear model to achieve the "consensus" constraint as in NCSCAE is reasonable and functionally correct, and achieves very competitive (or better) performance in most cases in the experimental section.

5.3 Progressive LCSCAE

For the progressive LCSCAE, in the k-th step, it will increase the style level of $\tilde{X}$ from k to $k + 1$, and meanwhile keep the consensus of different features. As shown in Algorithm 3, the input of the LCSCAE is the style feature X. The output of the algorithm is the encoded feature $\mathbf{h}^{(k)}$ and the projection matrix $W^{(k)}$, $k \in [1, L - 1]$.

To initialize, we set $\mathbf{h}^{(0)} = X$. For the step k, the encoded feature $\mathbf{h}^{(k-1)}$ is regarded as the input. First, we calculate the output for $X^{(k)}$ as described in Eq. (5). Second, we optimize W by the proposed LCSCAE as Algorithm

0162-8828 (c) 2017 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2017.2771766, IEEE Transactions on Pattern Analysis and Machine Intelligence

Algorithm 3 Progressive LCSCAE

INPUT: Original style feature X. The number of style level L.

OUTPUT: Style centralizing feature $\mathbf{h}^{(k)}$, projection matrices: $W^{(k)}$, $k \in [1, L-1]$.

1: Initial $\mathbf{h}^{(0)} = X$.
2: **for** k=1,2,...,L-1 **do**
3: $X^{(k)}$=$\mathbf{h}^{(k-1)}$.
4: Calculate $\tilde{X}^{(k)}$ as Eq. (5).
5: Calculate $W^{(k)}$ by Algorithm 2.
6: Calculate encode feature by $\mathbf{h}^{(k)} = \tanh(W^{(k)}X^{(k)})$.
7: **end for**

2. The learned W achieves the properties of both group sparsity and low-rankness. Afterwards, we calculate the new features $W\tilde{X}$ followed by a non-linear function for normalization. In this work, following the suggestion in [28] we use $\tanh(\cdot)$ to achieve the nonlinearity performance. The encoded feature $\mathbf{h}^{(k)}$ is regarded as the input in the next step $k+1$. After $(L-1)$ steps, we obtain $(L-1)$ sets of weight matrices and corresponding encoded features.

6 Experiment

In this section, we focus on a series of weak style image classification problems. While existing methods only target at one specific application [1], [3], [4], our methods can handle a wide range of applications with competitive performance.

6.1 Dataset Processing

6.1.1 Fashion Style Classification Dataset

Hipster Wars [4]. Kiapour et al. collected a fashion style dataset [4] including 1,893 images of 5 fashion styles, as shown in Figure 7. They also launched an online style comparison game to collect human judgments.

Online Shopping (Our collected). We collected an Online Shopping dataset for fashion classification on daily dresses. It is collected from online shopping websites (e.g., "Nordstrom.com", "barneys.com") with more than 30,000 images. We invited 7 professionals to manually label each image to one of 12 classes according to the category definition by the fashion magazine [32]. Example images are shown in Figure 8 and the distributions of 12 classes are shown in the supplementary materials.

We also provide the style level of each image based on the human judgments. For image i, we calculate how many people labeled this image to the category j, $j \in \{1, ..., 12\}$, denoted by $\phi_i^{(j)}$. We choose the j^*-th category, where $j^* = \arg\max_j \phi_i^{(j)}$, as the ground truth and $\phi_i^{(j^*)}$ is regarded as style level for image i. To conduct the experiments for SCAE, we follow the assumption that images with higher ϕ are more centralized than images with lower ϕ. This is reasonable as a consistent labeling usually indicates a representative style image.

We summarize the low-level descriptor extraction pipeline of the Hipster Wars dataset and the Online Shopping dataset:

(a) (b) (c) (d) (e)

Figure 7. Examples for 5 categories in the Hipster Wars dataset: (a) bohemian, (b) hipster, (c) goth (d) pinup, (e) preppy.

(a) (b) (c) (d) (e) (f)

(g) (h) (i) (j) (k) (l)

Figure 8. Examples for 12 categories in the Online Shopping dataset: (a) avant-garde, (b) elegant, (c) folk, (d) leisurely, (e) modern, (f) neutral, (g) renascent, (h) romantic, (i) sexy, (j) splendid, (k) technology, (l) young.

1) Pose estimation is applied to extract key boxes of the human body [33]. Note that for Hipster Wars we use the full body bounding box, while for Online Shopping, we only use the upper body bounding box (Online Shopping images are upper body centered);
2) We extract 7 dense features for each box: RGB color value, LAB color value, HSI color value, Gabor, MR8 texture response [34], HOG descriptor, and the probability of pixels belonging to skin categories[1]. For the missing bounding boxes, we refer to the neighbor bounding boxes for compensation;
3) Finally, we split each box into 4 patches (2×2) and extract features with mean-std pooling. Then we concatenate all the pooled features from 4×26 patches as the whole body style descriptor and those from 4×18 patches as the upper body style descriptor. The dimension of the full body style descriptors of a single image is 17,264 and that of the upper body is 11,952.

6.1.2 Manga Style Classification Dataset

Chu et al. collected a shonen (boy targeting) and shojo (girl targeting) Manga dataset including 240 panels [3]. Six computational features: including angle between lines, line orientation, density of line segments, orientation of nearby lines, number of nearby lines with similar orientation and line strength, are calculated. Example shojo and shonen style panels are shown in Figure 9.

Since Manga dataset does not provide manually labeled style level information, we automatically calculate it. First, we apply a mean-shift clustering algorithm to find the peak

1. http://kr.mathworks.com/matlabcentral/fileexchange/28565-skin-detection

0162-8828 (c) 2017 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2017.2771766, IEEE Transactions on Pattern Analysis and Machine Intelligence

Figure 9. Examples for shojo style and shonen style in the Manga dataset. The first row is shoji style and the second row is shonen style.

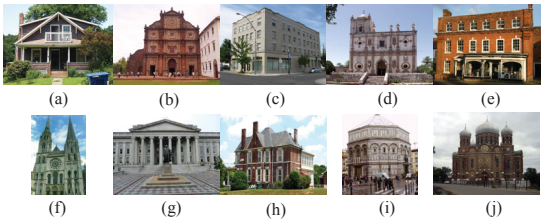


Figure 10. Examples for 10 categories in the Architecture Style dataset: (a) American craftsman, (b) Baroque architecture, (c) Chicago school architecture, (d) Colonial architecture, (e) Georgian architecture, (f) Gothic architecture, (g) Greek Revival architecture, (h) Queen Anne architecture, (i) Romanesque architecture, (j) Russian Revival architecture.

of the density of the images for each style based on the line strength feature. Line strength feature is most discriminative among 6 features measured by p-values. Images at the peak of the density are regarded the most representative ones. Then we rank the images according to the distances between the most centralized images. We denote p as the percentage of top ranked data to present the style level for each image by following [4].

6.1.3 Architecture Style Classification Dataset

Xu et al. collected an architecture style dataset containing 5,000 images [1]. It is the largest publicly available dataset for architectural style classification. The category definition is according to "Architecture_by_style" of Wikimedia[2]. Example images of ten classes are shown in Figure 10. As there is no manually labeled style level information, we apply the similar strategy used in Manga dataset to generate the style level information.

6.2 Comparison Methods

In this section, we first introduce the comparison methods including the-state-of-art works of fashion [4], [12], [13], manga [3], and architecture style classification [1]. Since there are few works for general style classification, we also investigate several general style classification methods through deep learning.

6.2.1 Fashion Style Classification

Kiapour et al. [4]: It applied mean-std pooling for 7 dense low-level features, and then concatenated them as the input of the classifier and named the concatenated features as the style descriptor.

Yamaguchi et al. [13]: It approached clothing parsing via retrieval, and considered robust style feature for retrieving similar style. They concatenated the pooling features similar to [4], followed by PCA for dimension reduction.

Bossard et al. [12]: This work focused on apparel classification with style. For style feature representation, they first learned a codebook through k-means clustering based on low-level features. Then the bag-of-words features were further processed by spatial pyramids and max-pooling.

6.2.2 Manga Style Classification

LineBased [3]: Chu and Chao designed 6 computational features derived from line segments to describe drawing styles (as described in the Dataset Section). Then they concatenated 6 features with equal weights.

6.2.3 Architecture Style Classification

Xu et al. [1]: Xu et al. adopted the Deformable Part-based Models (DPM) to capture the morphological characteristics of basic architectural components, where DPM describes an image by a multi-scale HOG feature pyramid.

MultiFea: The baseline in [1] only employed the HOG feature, but our CSCAE employed multiple features. Thus, we generate another baseline using multiple features for fair comparisons. We choose 6 low-level features according to SUN dataset [3], including: HoG, GIST, DSIFT, LAB, LBP and tinny image. First, we apply PCA on each feature. Then, we concatenate the normalized features together.

6.2.4 Deep General Style Classification

We detailed various of baseline methods:

AE: The conventional Auto-Encoder (AE) [17] is applied for learning mid/high-level features. The inputs of AE are the concatenated low-level features.

DAE: We conduct internal comparisons between SCAE and Denoising Auto-encoder (DAE) since both SCAE and DAE share the spirit of "noise". We apply the marginalized Stacked Denoising Auto-encoder (mSDA) [28], a widely applied version of DAE for evaluation. The inputs of the DAE are corrupted image features. As in [19], [23] and [28], we use the dropout noise whose corruption rate is learned by cross validation. Other settings such as the number of stacked layers and the layer size are the same as SCAE.

SCAE: We apply the proposed Style Centralizing Auto-Encoder (SCAE) for learning mid/high-level features. The inputs of SCAE are the concatenated various kinds of low-level feature descriptors, i.e., an early fusion for SCAE.

MC-SCAE: The multi-column SCAE. This is different from the proposed CSCAE as it did not consider the consensus constraint. Instead, it trains multiple SCAEs independently, one feature at a time. Then a late fusion is applied as [35].

LCAE: To demonstrate the roles of "progressive" in LC-SCAE, we generate a Linear Consensus Auto-Encoder (LCAE) as another baseline. LCAE is similar to LCSCAE except that in each progressive step, the input and output features are exactly the same.

NCSCAE [8]: It contains the full pipeline of the proposed NCSCAE in Section 4, which is our previous work in [8].

LCSCAE: This method contains the full pipeline of the proposed LCSCAE in Section 5.

2. https://commons.wikimedia.org/wiki/Category:Architecture_by_style

3. http://vision.cs.princeton.edu/projects/2010/SUN/

0162-8828 (c) 2017 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2017.2771766, IEEE Transactions on Pattern Analysis and Machine Intelligence

Table 1
Performances of fashion style classification of 10 methods on Hipster Wars dataset.

Performance	p=0.1	p=0.2	p=0.3	p=0.4	p=0.5
Kiapour et al. [4]:	77.73	62.86	53.34	37.74	34.61
Yamaguchi et al. [13]:	75.75	62.42	50.53	35.36	33.36
Bossard et al. [12]:	76.36	62.43	52.68	34.64	33.42
AE	83.76	75.73	60.33	44.42	39.62
DAE	83.89	73.58	58.83	46.87	38.33
SCAE	84.37	72.15	59.47	48.32	38.41
MC-SCAE	87.42	77.00	62.42	51.68	41.54
LCAE	87.55	76.34	63.55	50.06	41.33
NCSCAE [8] (Ours1)	89.21	75.32	**64.55**	52.88	43.77
LCSCAE (Ours2)	**90.31**	**78.42**	64.35	**54.72**	**45.31**

Table 2
Performance (%) of fashion style classification of 10 methods on Online Shopping dataset.

Performance	ϕ=7	ϕ=6	ϕ=5	ϕ=4	ϕ=3
Kiapour et al. [4]:	60.92	58.52	54.57	48.63	42.40
Yamaguchi et al. [13]:	55.00	53.96	51.73	45.38	37.91
Bossard et al. [12]:	54.58	59.43	52.47	41.39	35.42
AE	66.32	60.32	57.58	50.03	44.07
DAE	68.44	61.48	58.44	50.06	45.69
SCAE	74.33	61.93	60.72	54.54	48.89
MC-SCAE	66.15	62.53	62.54	53.28	49.52
LCAE	72.11	66.45	63.33	53.27	48.73
NCSCAE [8] (Ours1)	70.41	68.42	63.84	51.18	**50.31**
LCSCAE (Ours2)	**75.02**	**71.43**	**64.24**	**54.43**	49.41

In all the classification tasks, we apply cross-validation and a 9:1 training to test ratio. *SVM classifier* is applied in Hipster Wars and Manga datasets by following the settings in [3], [4], while *Nearest Neighbor classifier* (NN) is applied on Online Shopping and Architecture datasets. For all deep learning baselines, we use the same number of layers as our methods.

6.3 Experimental Results

6.3.1 Results on Hipster Wars Dataset (Public)

In order to show the effectiveness of different methods on weak style classification, we conduct experiments on 5 different settings. First, we rank all the images in the dataset according to the style level scores from high to low. The style level scores are provided in [4] by reliable human judgments. Following [4], we denote p as the percentage of top ranked images used to generate the sub-dataset for experiments. The default number of stacked layers are $L = 4$ with the layer size as 400. In NCSCAE, we set ρ=0.05, $\lambda = 10^{-5}$ and $\beta = 10^{-2}$, while in LCSCAE, we set λ=100, and rank r=40, based on cross-validation.

Table 1 shows the accuracy (%) of comparison methods and our methods under different style levels $p = 0.1, ..., 0.5$. First, from Table 1 we can see that results of the proposed 7 deep style representation learning methods in general performs better compared to the existing works [4], [12], [13]. Notably, the proposed SCAE is already better than Kiapour et al. [4] by 6.64%, 9.09%, 6.13%, 10.58% and 3.80% under p from 0.1 to 0.5. Second, when comparing DAE with SCAE, we could see the effectiveness of the style centralizing strategy. Third, when comparing MC-SCAE with NCSCAE, we find that NCSCAE outperforms MC-SCAE under 4 settings which demonstrates the importance of column consensus constraints. Fourth, when comparing LSCAE with LCAE, we see that LSCAE outperforms LCAE in all the settings. The only difference between LSCAE with LCAE is the style centralizing learning strategy. Fifth, we can see that LCSCAE achieves the best performance under p=0.1, 0.2, 0.4 and 0.5.

Table 3
Performance (%) of manga style classification of 8 methods on Manga dataset.

Performance	p=0.2	p=0.4	p=0.6	p=0.8	p=1.0
LineBased [3]	83.21	71.35	68.62	64.79	60.07
AE	83.61	72.52	69.32	65.18	61.28
DAE	83.67	72.75	69.32	65.86	62.86
SCAE	83.75	73.43	69.32	65.42	63.60
MC-SCAE	85.38	72.93	71.48	69.58	65.48
LCAE	85.35	76.45	72.57	67.85	65.79
NCSCAE [8] (Ours1)	86.23	76.93	73.28	68.63	67.35
LCSCAE (Ours2)	**90.70**	**80.96**	**77.97**	**77.63**	**79.90**

In addition, we conduct t-test to evaluate the significance of improvements by our methods in the settings: LCSCAE vs. NCSCAE, SCAE vs. DAE, NCSCAE vs. MC-SCAE and LCSCAE vs. LCAE in Table 5. The lower the significance level is (i.e., p value), the more confident the difference between two methods will be. Table 5 presents p values of t-test under the setting of the rightmost column in Table 1-4 which demonstrates the significance of the improvements (i.e., p$<$0.05).

6.3.2 Results on Online Shopping Dataset (Collected)

Similar to Hipster Wars dataset, we conduct experiments under 5 different settings: $\phi = 7, 6, 5, 4, 3$, where ϕ presents the maximum style level in the sub-dataset (described in the dataset processing section). Table 2 shows the accuracy (%) of 10 methods. We use NN classifier and empirically set the number of neighbors to 5. Other settings are the same as Table 1.

From these results, we can observe that the proposed NCSCAE and LCSCAE achieve the best and second best performance. NCSCAE performs better than [4] by 9.49%, 10.10%, 8.74%, 2.55% and 7.19% under ϕ from 7 to 3. And LCSCAE even outperforms NCSCAE under ϕ from 7 to 4 by 4.61%, 3.01%, 0.40% and 3.25%, showing the superiority of our methods on Online Shopping dataset. In addition, the comparisons between SCAE with DAE, NCSCAE with MC-SCAE, and LCSCAE with LCAE, demonstrate the effectiveness of the centralizing strategy and the consensus constraint.

Figure 11 visualizes the correct and incorrect classification results on Online Shopping dataset at three style centralized degrees: ϕ=6, ϕ=4 and ϕ=2. In the left sub-figure for incorrect classification results, we show the ground truth labels in blue and the estimated labels in red. The visualization reveals that even in the challenging task with weak style level ϕ=2, our method can still achieve reasonable results.

0162-8828 (c) 2017 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

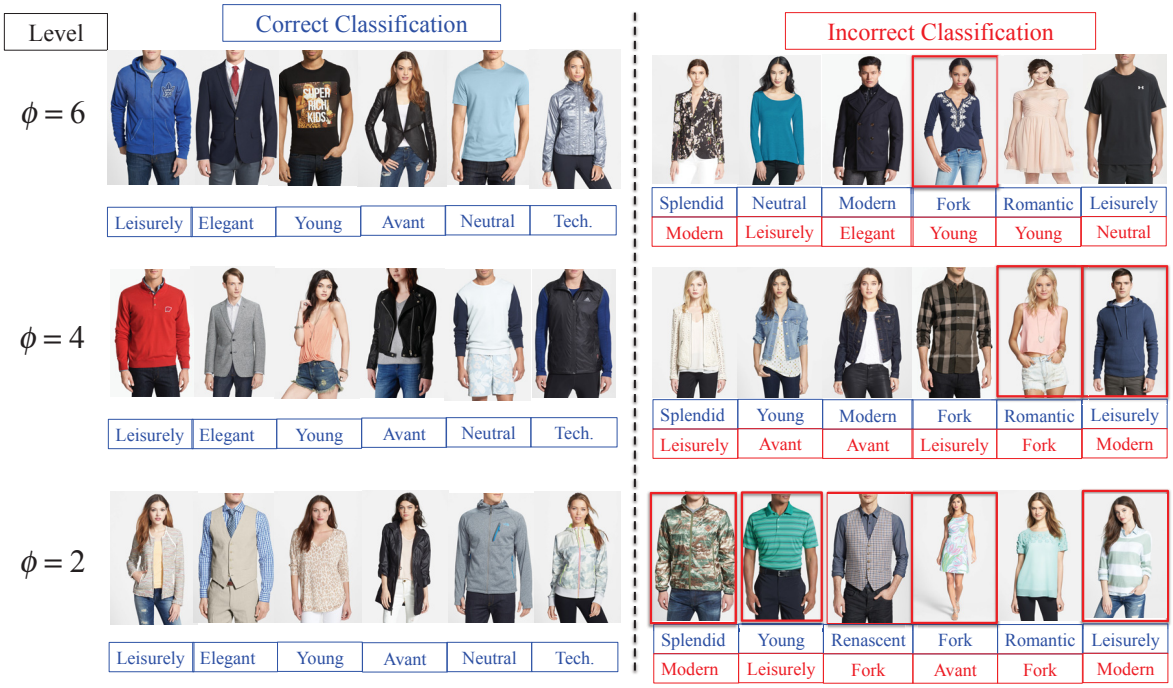


Figure 11. Visualization of the correct (left part) and incorrect (right) classification results on Online Shopping dataset under different style levels ϕ. Below images, the correct category labels (ground truth) are marked in blue (first row) and the incorrect are marked in red (second line). From both the correct and incorrect examples, we could see when ϕ decreases, it becomes difficult to distinguish a fashion style by human. For example, in the correct classification results, it is easier to recognize the "Avant" style in ϕ=6 (higher style level), but more difficult to recognize the "Avant" style in ϕ=2. For the incorrect examples, we could see that although the estimated results are not the same with the ground truth, they are still reasonable upon common sense. For example, in the first column in incorrect classification in ϕ = 6, the ground truth is "Splendid", and our estimation is "Modern". The red box on the image means the estimate result is neither the same as the ground truth, nor acceptable by the human sense.

Table 4
Performance (%) of architecture style classification on 10-class Architecture dataset by NN classifier. The best and second best results under each p are shown in bold font and underline.

Performance	p=0.2	p=0.4	p=0.6	p=0.8	p=1.0
Xu et al. [3]	40.32	35.96	32.65	33.32	31.34
MultiFea	52.78	53.00	50.29	49.93	46.79
AE	58.72	56.32	52.32	52.32	48.31
DAE	58.55	56.99	53.34	52.39	50.33
SCAE	59.61	57.00	53.27	54.28	51.76
MC-SCAE	58.16	57.93	54.27	53.21	51.23
LCAE	59.54	58.66	54.55	53.46	51.88
NCSCAE [8] (Ours1)	**60.48**	<u>58.85</u>	**55.37**	**54.97**	<u>52.84</u>
LCSCAE (Ours2)	<u>60.37</u>	**59.41**	<u>55.12</u>	<u>54.74</u>	**54.68**

6.3.3 Results of Manga Style Classification

Table 3 shows the accuracy (%) of the proposed 7 deep learning based methods and the state-of-the-art method LineBased [3] under $p = 0.2, 0.4, 0.6, 1.0$. From these results, we can observe that the performance of our methods are better than the state-of-the-arts.

NCSCAE outperforms LineBased [3] by 3.02%, 5.58%, 4.26%, 3.84% and 7.28% under p from 0.2 to 1.0. LCSCAE largely outperforms NCSCAE by 4.47%, 4.03%, 4.69%, 9.00% and 8.55%. Compared with other applications, LCSCAE achieves the largest improvement over NCSCAE on manga style classification. Compared to fashion and architecture images, LCSCAE works especially well for the face images. We believe that the face structure and different weights of patches work well with the low-rankness and group sparsity assumptions.

Table 5
p values of t-test of on four weak style datasets (with the same setting as the rightmost column in Table 1-4).

Performance	hipster	shopping	manga	arch
LCSCAE vs. NCSCAE	0.0054	0.0033	2.7×10^{-5}	0.044
SCAE vs. DAE	0.037	0.0036	0.012	0.026
NCSCAE vs. MC-SCAE	0.0068	0.013	0.0067	0.0094
LCSCAE vs. LCAE	0.0028	0.035	4.5×10^{-5}	0.0011

6.3.4 Results of Architecture Style Classification

Table 4 shows the classification accuracy on the architecture style dataset of 9 methods under style level p=0.2, 0.4, 0.6, 0.8 and 1.0. First, comparing the method in [3] with MultiFea, we can learn that the additional low-level features do contribute to the performance. Second, all the proposed deep learning methods achieve better performance than low-level features based methods. Our NCSCAE and LCSCAE achieve the best and second best under all the style levels.

6.4 Discussion on NCSCAE and LCSCAE

6.4.1 Discussion on Progressive Steps

In this section, we compare the progressive manner (SSCAE) with One-Step SCAE (OSCAE). OSCAE means we skip the progressive procedure and centralize all data in different style levels to the highest level in one step, which gives

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2017.2771766, IEEE Transactions on Pattern Analysis and Machine Intelligence

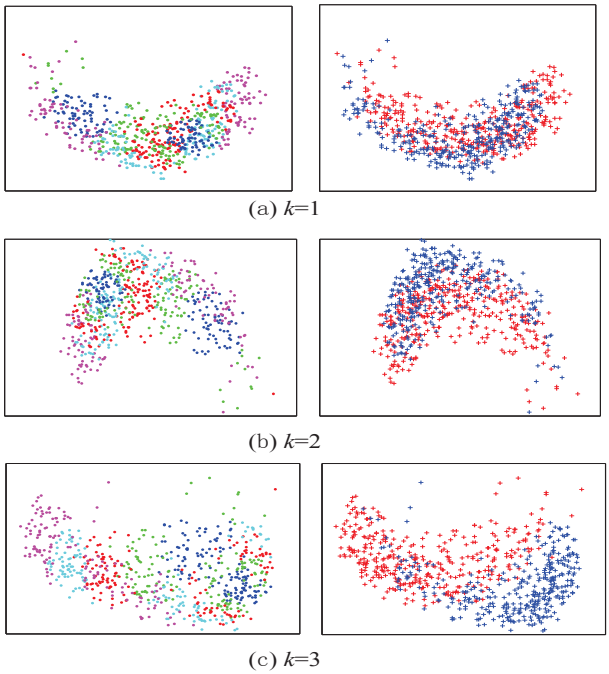


Figure 12. Visualization of the encoded features in SCAE in progressive step $k = 1, 2, 3$ on the Manga dataset. Meanings of different colors for different style levels are: Blue: level 5; Green: level 4; Red: level 3; Cyan: level 2; Magenta: level 1. Note that we apply PCA for dimensionality reduction before visualization.

Table 6
Performance (%) of OSCAE and SSCAE on Hipster Wars dataset. The best and second best results under each p are shown in bold font and underline.

Performance	p=0.1	p=0.2	p=0.3	p=0.4	p=0.5
SSCAE1	88.84	74.23	59.50	51.72	42.47
SSCAE2	89.21	75.00	62.42	**52.88**	**43.87**
SSCAE3	89.42	75.00	59.63	51.09	43.58
SSCAE4	89.21	**75.32**	**64.55**	52.82	43.77
OSCAE	**90.10**	74.71	61.24	50.50	42.40

rise to a basic supervised auto-encoder with style level as the label information. We show the performance of the non-linear version model in this section as an example.

Table 6 and Table 7 show the performance of SSCAE and OSCAE on Hipster Wars and Online Shopping datasets. For SSCAE, we show the classification results under step k =1,2,3,4 as SSCAE1 to SSCAE4. We could see that under all the style levels on two datasets, except p=0.1 in Table 6, SSCAE methods achieve the best (in bold font) and the second best (with underline) performance. It demonstrates the effectiveness of the progressive stacked SCAE. Among SSCAE1 to SSCAE4, SSCAE4 achieves two best and two second best performance on both datasets, slightly higher than SSCAE2 and SSCAE3. For OSCAE, under p=0.4, 0.5 in Table 6 and ϕ=6, 5, 4 in Table 7, it performs even worse than SSCAE1. It shows that directly mapping weak style images to the highest level by AE does not work well.

Visualization of encoded feature in SCAE: Figure 12 shows the visualization of encoded features in the progressive step $k = 1, 2, 3$ in manga style classification. Similar to Figure 2, PCA is employed to reduce the dimensionality of the descriptors. The input low-level feature is "density of line segments" proposed in [3]. In all the sub-figures, a dot represents a sample. In the right sub-figures, colors are used to distinguish *different styles*, while in left sub-figures, colors are used to distinguish *different style levels*.

From Figure 12, we could see that at step $k = 1$, in the right sub-figures, the "shoji" samples (in red) and "shonen" samples (in blue) have overlaps. In the left sub-figures, samples in the strong style level (in blue) are separated from each other. However, samples in weak style levels overlap with each other. For example, it is hard to separate the samples in cyan in two styles. During progressive steps, samples in different styles gradually separate due to the style centralization. At progressive step $k = 2, 3$, in the right sub-figures we could see the samples in red and blue gradually become separable. In the left sub-figures, we could see that during style centralizing, the weak style samples, shown in green, red, cyan and magenta move closely to the locations of blue samples in two different styles. Since the blue samples represent the strong style level and could easily be separated, the centralization process makes the weak style samples more distinguishable.

Table 7
Performance (%) of OSCAE and SSCAE on Online Shopping dataset.

Performance	ϕ=7	ϕ=6	ϕ=5	ϕ=4	ϕ=3
SSCAE1	69.25	67.72	63.34	51.35	49.72
SSCAE2	69.75	**68.93**	63.72	**51.63**	49.60
SSCAE3	69.63	68.32	63.48	51.36	**50.48**
SSCAE4	**70.41**	68.42	**63.84**	51.18	50.31
OSCAE	70.00	66.52	62.46	51.31	50.16

Table 8
Comparison of training time of NCSCAE and LCSCAE (hour)

Performance	Hipster	Shopping	Manga	Architecture
NCSCAE	17.15	10.26	6.52	8.82
LCSCAE	7.45	5.65	3.16	4.73

6.4.2 Discussion on Computational Cost

In this section, we discuss the computational cost of NCSCAE and LCSCAE. Table 8 shows the computational cost (hour) of training NCSCAE and LCSCAE on four datasets. The experiments are based on a cluster of 256G memory and 3.50 GHz Intel Xeon CPU.

NCSCAE suffers a high computational cost especially when the feature dimension is high. The computational cost of Hipster Wars dataset and Online Shopping dataset are the highest. LCSCAE uses nearly half of the running time of NCSCAE on 4 datasets. On Hipster Wars dataset, LCSCAE only spends 43% running time of NCSCAE's.

6.4.3 Discussion on LCSCAE

In this part, we discuss the impacts of rank r and the balancing parameter λ in Eq. (16) for LCSCAE.

In Figure 13 (left), we set r to 40 and discuss the impacts of λ. We could see that when $\lambda < 80$, the performance increases sharply from 0.25 to 0.52. When $\lambda \geq 80$, the

0162-8828 (c) 2017 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2017.2771766, IEEE Transactions on Pattern Analysis and Machine Intelligence

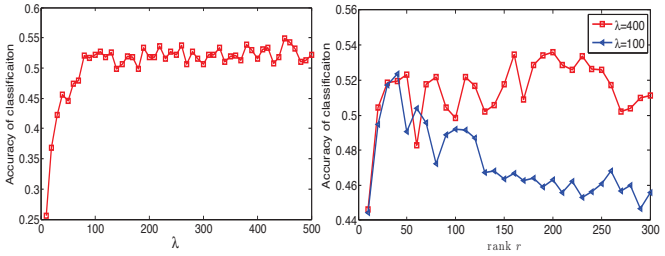


Figure 13. Discussion of LCSCAE. The left figure shows the accuracy curve on LCSCAE with λ from 10 to 500 at intervals of 10. The rank r is fixed as 40. The right figure shows accuracy curves with rank r from 10 to 300 at intervals of 10.

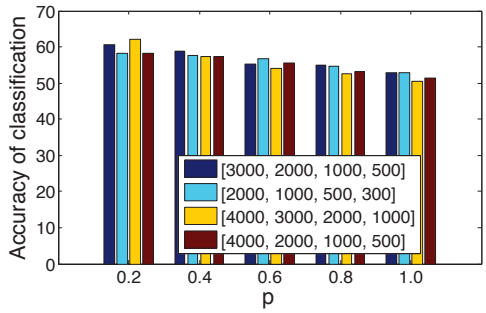


Figure 14. Accuracy on different layer sizes of NCSCAE across style levels p=0.2 to 1.0. We stack the encoded features on four hidden layers together. Different settings of layer sizes are shown in different colors. For example, [3000, 2000, 1000, 500] indicates the number of neurons of four layers as 3000, 2000, 1000 and 500 respectively.

performance is round 0.5 to 0.55. We believe a small λ will not sufficiently suppress the group sparsity term in Eq. (16), and thus fails to select the right features, and implement the consensus constraint.

In Figure 13 (right), we discuss the impacts of different settings of rank r. We could see that under both $\lambda = 400$ and $\lambda = 100$, the performance increases sharply from 0.44 to 0.52 when r increases from 10 to 40. It shows that if r is small, the majority of useful information is removed, which ruins the performance. On the other hand, if r is large, the performance is unstable given different λ. For λ=100, the performance drops when $r \geq 50$, while for λ=400, the performance degrades when $r \geq 250$. We believe that a higher rank will not implement the consensus constraint well, and thus lead to an unpredictable overall performance.

6.4.4 Discussion on Layer Size of Auto-Encoder

In this part, we discuss the impacts of the layer size of CSCAE where NCSCAE is employed as an example. Figure 14 shows the accuracy on four different settings of layer sizes. We can see that the difference of the accuracies under 4 different settings of layer size is within 2%. Although there might be infinite layer size settings, we can easily identify robust settings of the layer size for NCSCAE, as shown in Figure 14. Among four settings, [3000, 2000, 1000, 500] achieves the highest performance in average among all the style levels, and achieves the highest performance under the style level p=0.4, 0.8 and 1.0.

Table 9
Performances (%) of manga style classification of 4 style level label settings of LCSCAE.

Performance	p=0.2	p=0.4	p=0.6	p=0.8	p=1.0
Setting 0	90.70	80.96	77.97	77.63	79.90
Setting 1	81.54	71.54	66.85	66.77	63.55
Setting 2	84.57	76.54	69.32	67.18	65.28
Setting 3	86.46	76.74	72.86	71.86	70.86

6.4.5 Discussion on Style Level Label Generation

In this section, we discuss the style level label generation since style level labels are the learning targets of our framework. We show that the style level labels could be either human labeled (in Hipster Wars dataset and Online Shopping dataset) or automatically assigned (in Manga and Architecture datasets). For the human labeling way, with the recent developed crowd-sourcing Internet marketplace such as Amazon Mechanical Turk (MTurk), both the time and the cost become affordable. In Manga and Architecture datasets, we show that automatically assigning labels is also a feasible way to generate styles, although the human labeling way is more recommended.

In addition, we analyze our algorithm's sensitivity to incorrect ratings. Table 9 shows results of the classification accuracy of Manga dataset under 4 different settings of style labels, namely, Setting 0: the same setting as Table 3; Setting 1: randomly assigning style level labels for each image; Setting 2: randomly assigning the style level labels for half of the images while the other half are the same as Table 3; Setting 3: randomly increasing or reducing one or two style levels for each image.

From Table 9, we could see the influence of incorrect style labels to the algorithm. In Setting 1, with randomly assigned style labels, the performance decreases by 10% to 15 %, and sometimes even lower than the conventional AE. The incorrect style labels may enforce the strong style images towards the weak style ones by mistake. The performance of setting 2 is between 1 and 3. In setting 3, the performance degrades by 5% in most cases, similar to that of NCSCAE. It shows that with slightly perturbed labels, we could still get acceptable results.

7 Conclusion

In this paper, we proposed a Consensus Style Centralizing Auto-Encoder (CSCAE) to extract robust style feature presentation for weak style classification. The Style Centralizing Auto-Encoder (SCAE) progressively drew weak style images to the class center to increase the feature discrimination. In both non-linear and linear versions of CSCAE, the consensus constraints automatically allocated the weights for different features. Specifically, in the Linear CSCAE, we proposed a novel Rank-Constrained Group Sparsity Auto-Encoder, and a corresponding fast solution to achieve competitive performance but saving half of the training time. Extensive experimental results on fashion, manga and architecture style classification demonstrated that the proposed SCAE, NCSCAE and LCSCAE were effective for these tasks.

0162-8828 (c) 2017 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2017.2771766, IEEE Transactions on Pattern Analysis and Machine Intelligence

REFERENCES

[1] Z. Xu, D. Tao, Y. Zhang, J. Wu, and A. C. Tsoi, "Architectural style classification using multinomial latent logistic regression," in *European Conference on Computer Vision*. Springer, 2014, pp. 600–615.

[2] S. Lee, N. Maisonneuve, D. Crandall, A. A. Efros, and J. Sivic, "Linking past to present: Discovering style in two centuries of architecture," in *IEEE International Conference on Computational Photography*, 2015.

[3] W.-T. Chu and Y.-C. Chao, "Line-based drawing style description for manga classification," in *ACM International Conference on Multimedia*. ACM, 2014, pp. 781–784.

[4] M. H. Kiapour, K. Yamaguchi, A. C. Berg, and T. L. Berg, "Hipster wars: Discovering elements of fashion styles," in *European Conference on Computer Vision*. Springer, 2014, pp. 472–488.

[5] G. Shalunts, "Architectural style classification of building facade towers," in *Advances in Visual Computing*. Springer, 2015, pp. 285–294.

[6] C. Weiss and M. Muller, "Tonal complexity features for style classification of classical music," in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*. IEEE, 2015, pp. 688–692.

[7] M.-J. Wu and J.-S. R. Jang, "Combining acoustic and multilevel visual features for music genre classification," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 12, no. 1, p. 10, 2015.

[8] S. Jiang, M. Shao, C. Jia, and Y. Fu, "Consensus style centralizing auto-encoder for weak style classification," in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*. AAAI, 2016.

[9] A. Goel, M. Juneja, and C. Jawahar, "Are buildings only instances?: exploration in architectural style categories," in *Indian Conference on Computer Vision, Graphics and Image Processing*. ACM, 2012.

[10] W. Herlands, R. Der, Y. Greenberg, and S. Levin, "A machine learning approach to musically meaningful homogeneous style classification," in *AAAI Conference on Artificial Intelligence*, 2014, pp. 276–282.

[11] J. C. Van Gemert, "Exploiting photographic style for category-level image classification by generalizing the spatial pyramid," in *ACM International Conference on Multimedia Retrieval*. ACM, 2011, pp. 1–8.

[12] L. Bossard, M. Dantone, C. Leistner, C. Wengert, T. Quack, and L. Van Gool, "Apparel classification with style," in *Asian Conference on Computer Vision*. Springer, 2013, pp. 321–335.

[13] K. Yamaguchi, M. H. Kiapour, and T. L. Berg, "Paper doll parsing: Retrieving similar styles to parse clothing items," in *IEEE International Conference on Computer Vision*. IEEE, 2013, pp. 3519–3526.

[14] X. Chao, M. J. Huiskes, T. Gritti, and C. Ciuhu, "A framework for robust feature selection for real-time fashion style recommendation," in *International workshop on Interactive multimedia for consumer electronics*. ACM, 2009, pp. 35–42.

[15] K. Yamaguchi, M. H. Kiapour, L. E. Ortiz, and T. L. Berg, "Parsing clothing in fashion photographs," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 3570–3577.

[16] S. Liu, Z. Song, G. Liu, C. Xu, H. Lu, and S. Yan, "Street-to-shop: Cross-scenario clothing retrieval via parts alignment and auxiliary set," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 3330–3337.

[17] Y. Bengio, "Learning deep architectures for ai," *Foundations and trends® in Machine Learning*, vol. 2, no. 1, pp. 1–127, 2009.

[18] Z. Ding, M. Shao, and Y. Fu, "Deep low-rank coding for transfer learning," in *International Joint Conference on Artificial Intelligence*, 2015, pp. 3453–3459.

[19] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proceedings of the 25th international conference on Machine learning*. ACM, 2008, pp. 1096–1103.

[20] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *The Journal of Machine Learning Research*, vol. 11, pp. 3371–3408, 2010.

[21] M. Kan, S. Shan, H. Chang, and X. Chen, "Stacked progressive auto-encoders (spae) for face recognition across poses," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2014, pp. 1883–1890.

[22] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Cognitive modeling*, vol. 5, pp. 696–699, 1988.

[23] M. Chen, Z. Xu, K. Q. Weinberger, and F. Sha, "Marginalized stacked denoising autoencoders," in *Learning Workshop*, 2012.

[24] O. Chapelle, B. Schölkopf, A. Zien *et al.*, "Semi-supervised learning," 2006.

[25] L. Zhang and Y. Lu, "Comparison of auto-encoders with different sparsity regularizers," in *International Joint Conference on Neural Networks*. IEEE, 2015, pp. 1–5.

[26] J. Nocedal, "Updating quasi-newton matrices with limited storage," *Mathematics of computation*, vol. 35, no. 151, pp. 773–782, 1980.

[27] J. Ngiam, A. Coates, A. Lahiri, B. Prochnow, Q. V. Le, and A. Y. Ng, "On optimization methods for deep learning," in *International Conference on Machine Learning*, 2011, pp. 265–272.

[28] M. Chen, Z. Xu, K. Weinberger, and F. Sha, "Marginalized denoising autoencoders for domain adaptation," *arXiv preprint arXiv:1206.4683*, 2012.

[29] G. Liu and S. Yan, "Latent low-rank representation for subspace segmentation and feature extraction," in *2011 International Conference on Computer Vision*. IEEE, 2011, pp. 1615–1622.

[30] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, and Y. Ma, "Robust recovery of subspace structures by low-rank representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 171–184, 2013.

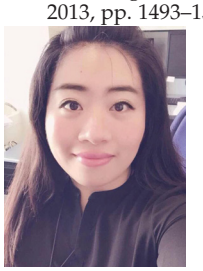
[31] F. Bunea, Y. She, M. H. Wegkamp *et al.*, "Joint variable and rank selection for parsimonious estimation of high-dimensional matrices," *The Annals of Statistics*, vol. 40, no. 5, pp. 2359–2388, 2012.

[32] Y.-C. Chang, M.-C. Chuang, S.-H. Hung, S.-J. C. Shen, and B. Chu, "A kansei study on the style image of fashion design," in *the 6th Asian Design Conference*, 2003.

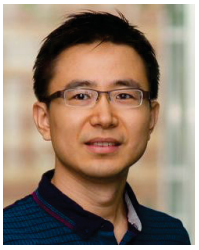
[33] Y. Yang and D. Ramanan, "Articulated pose estimation with flexible mixtures-of-parts," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2011, pp. 1385–1392.

[34] M. Varma and A. Zisserman, "A statistical approach to texture classification from single images," *International Journal of Computer Vision*, vol. 62, no. 1-2, pp. 61–81, 2005.

[35] F. Agostinelli, M. R. Anderson, and H. Lee, "Adaptive multi-column deep neural networks with application to robust image denoising," in *Advances in Neural Information Processing Systems*, 2013, pp. 1493–1501.



Shuhui Jiang received the B.S. and M.S. degrees in Xi'an Jiaotong University, Xi'an, China, in 2007 and 2011, respectively. She is now pursuing her PHD degree in School of Electrical and Computer Engineering, Northeastern University (Boston, USA). She was the recipient of the Dean's Fellowship of Northeastern University from 2014. She is interested in machine learning, multimedia and computer vision. She has served as the reviewers for IEEE journals: IEEE TNNLS etc.



Yun Fu (S'07-M'08-SM'11) received the B.Eng. degree in information engineering and the M.Eng. degree in pattern recognition and intelligence systems from Xi'an Jiaotong University, China, respectively, and the M.S. degree in statistics and the Ph.D. degree in electrical and computer engineering from the University of Illinois at Urbana-Champaign, respectively. He is an interdisciplinary faculty member affiliated with College of Engineering and the College of Computer and Information Science at Northeastern University since 2012. His research interests are Machine Learning, Computational Intelligence, Big Data Mining, Computer Vision, Pattern Recognition, and Cyber-Physical Systems. He has extensive publications in leading journals, books/book chapters and international conferences/workshops. He serves as associate editor, chairs, PC member and reviewer of many top journals and international conferences/workshops. He received seven Prestigious Young Investigator Awards from NAE, ONR, ARO, IEEE, INNS, UIUC, Grainger Foundation; seven Best Paper Awards from IEEE, IAPR, SPIE, SIAM; three major Industrial Research Awards from Google, Samsung, and Adobe, etc. He is currently an Associate Editor of the IEEE Transactions on Neural Networks and Leaning Systems (TNNLS). He is fellow of IAPR, a Lifetime Senior Member of ACM and SPIE, Lifetime Member of AAAI, OSA, and Institute of Mathematical Statistics, member of Global Young Academy (GYA), INNS and Beckman Graduate Fellow during 2007-2008.

0162-8828 (c) 2017 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.