Stylized Adversarial AutoEncoder for Image Generation

Yiru Zhao*
Shanghai Jiao Tong University
Alibaba Group
yiru.zhao@sjtu.edu.cn

Bing Deng
Alibaba Group
dengbing.db@alibaba-inc.com

Jianqiang Huang
Alibaba Group
jianqiang.hjq@alibaba-inc.com

Hongtao Lu†
Shanghai Jiao Tong University
htlu@sjtu.edu.cn

Xian-Sheng Hua‡
Alibaba Group
huaxiansheng@gmail.com

ABSTRACT

In this paper, we propose an autoencoder-based generative adversarial network (GAN) for automatic image generation, which is called "stylized adversarial autoencoder". Different from existing generative autoencoders which typically impose a prior distribution over the latent vector, the proposed approach splits the latent variable into two components: style feature and content feature, both encoded from real images. The split of the latent vector enables us adjusting the content and the style of the generated image arbitrarily by choosing different exemplary images. In addition, a multiclass classifier is adopted in the GAN network as the discriminator, which makes the generated images more realistic. We performed experiments on hand-writing digits, scene text and face datasets, in which the stylized adversarial autoencoder achieves superior results for image generation as well as remarkably improves the corresponding supervised recognition task.

CCS CONCEPTS

• **Computing methodologies** → **Computer vision**; *Machine learning approaches*;

KEYWORDS

Image Generation, Generative Adversarial Network, AutoEncoder

1 INTRODUCTION

Generative natural image modeling is a fundamental research problem in computer vision and machine learning areas. Early works [22, 23, 34] are more focused on the statistical principles of generative networks modeling, but the corresponding results are limited to certain particular patterns due to the lack of effective feature representations. Deep neural networks have shown prominent advantages in learning representations and have been proven highly effective in discriminative vision tasks, such as image classification [19] and object detection [8], nourishing a series of deep generative models with Bayesian inference [17, 18] or adversarial training [5, 9].

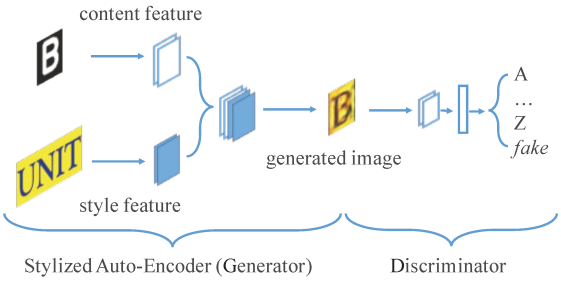

Figure 1: Illustration of our model, which extract features from content image and style image respectively, then merge them to decode target image. A multiclass discriminator forces generated image more realistic.

It is shown that regularized neural networks generally work better than unconstrained networks in practice [1]. Frequently applied regularization forms include L1-norm LASSO, L2-norm ridge regression, as well as some modern techniques such as dropout [33]. In particular, for autoencoder neural networks, quite a few regularization methods have been proposed recently [18, 21, 26]. However, all these regularizations impose a prior over the latent variable (also called hidden code) distribution, where Gaussian distribution is often employed. It works well for relatively simple generative tasks, *e.g.* , modeling grayscale digital images, but is not suitable for generating complex images, such as color alphanumeric images or human face, because the real distribution of the latent variable can be neither completely observed nor easily modeled.

As shown in Figure 1, in this paper, we propose a novel generative model, called *Stylized Adversarial AutoEncoder*(SAAE) that uses an adversarial way to train a stylized autoencoder. Unlike existing autoencoders, we divide the latent vector into two parts, one is related to the image content and the other is related to the image style. Both content and style features are encoded from exemplary images, without prior assumptions on the distribution of the latent variable. The target image with given content and style can be decoded from the combined latent variable, which means we are enabled to adjust the output image by choosing different exemplary content and/or style images. In addition, inspired by the method in [9, 21, 26], we adopt an adversarial criterion in the GAN training

*This work was done when the author was visiting Alibaba as a research intern.
†Corresponding author.
‡Corresponding author.

MM'17, October 23–27, 2017, Mountain View, CA, USA.
 ISBN 978-1-4503-4906-2/17/10…$15.00
DOI: https://doi.org/10.1145/3123266.3123450

stage. Instead of using a typical binary classifier [9], a multiclass classifier is proposed as the discriminator in our GAN network, which has a higher capacity to model the variations of the generated images when discriminating true and faked images. Moreover, the GAN training is known to be very challenging to converge due to its contradictory min-max objective [36], hence we develop an effective empirical 3-step training algorithm to improve the convergence of the proposed GAN network.

The main contributions of this work can be summarized as follows:

- We propose a novel deep autoencoder network, which encodes content and style features separately from two exemplary images and decode new images from those two features.
- A multiclass classifier is applied as the discriminator, which better models the variations of generated images and effectively enforces the generative network to generate more realistic result.
- We develop a 3-step training strategy to escort the convergence of the proposed stylized adversarial autoencoder.

2 RELATED WORK

Related works of the proposed stylized adversarial autoencoder can be summarized into three categories: autoencoder, generative adversarial network, and style transfer learning. We will explain the connections and differences between SAAE and these methods in the corresponding aspects.

2.1 Autoencoders

In the early stage, autoencoders are designed for improving data compression efficiency. It can be regarded as a way to find a minimum-length description for given data, in which the encoding weights produce a code for a particular sample and the decoding weights embody a generative model which turns this code back into an approximation of the original sample [12, 14]. In fact, this kind of autoencoder often ends up with learning a low-dimensional representation of the original data, which is very similar to PCA. Autoencoder was also proven useful in initializing network weights and make it possible to train a deep neural network [13]. Sparse Autoencoder [28] imposes a sparsity constraint on the hidden codes, then the autoencoder can capture certain intrinsic visual features in images, such as edges and corners. However, these types of autoencoders are designed for image reconstruction rather than generating a new image.

In order to make an autoencoder a truly generative network, variational autoencoder(VAE) [18] imposes a Gaussian distribution on the hidden codes in the training stage, then one can generate image from a VAE by sampling data from the Gaussian prior distribution and feeding the data into the VAE's decoder. This corresponding regularization is realized by minimizing the KL-divergence between the inferred distribution and the imposed prior. The KL-divergence can be easily obtained in a closed form when Gaussian prior is chosen, however, VAE is difficult to be used when there does not exist a simple closed form solution for the KL-divergence. Accordingly, adversarial autoencoders (AAE) [26] were proposed, which perform variational inference by matching the aggregated posterior of the hidden codes vector with an arbitrary prior distribution using adversarial training. Though imposing an arbitrary prior on the latent variable solves the former difficulty, it will not work well in real word, because the real manifold of natural images in the latent space are not easy to be completely observed or reasonably modeled. Therefore, we propose to learn the latent variable from observed data without any prior distribution assumption. As to be detailed, this design also enables us to regulate the generated outputs of the proposed model, while is not possible for most previous autoencoders.

2.2 Generative adversarial networks

In recent years, generative adversarial networks (GANs) [5, 9, 29] have been introduced for generating desired images. The GAN framework [9] establishes a min-max adversarial "game" between two neural networks: a generative model G and a discriminative model D. The generator uses a function $G(z)$ to generate data with samples z from a prior $p_z(z)$, aiming at capturing the corresponding true data distribution, while the discriminator $D(x)$ attempts to determine whether a point x in the data space is a sample from generated images(negative samples) or a sample from the true data distribution $p_{data}(x)$ (positive samples, which we are trying to model). $G(z)$ is trained to confuse $D(x)$, that is, to generate samples that $D(x)$ is difficult to differentiate whether they are from true data distribution or not. The formulation to solve this game can be summarized as:

$$\begin{aligned}\min_G \max_D V(D,G) =& E_{x\sim p_{data}(x)}[\log D(x)] \\ &+E_{z\sim p_z(z)}[\log(1-D(G(z)))]\end{aligned} \tag{1}$$

The images generated by earlier GAN networks suffer from being noisy and incomprehensible [5, 10]. A Laplacian pyramid extension to this approach [5] showed higher generation quality. A recurrent network method [10] and a deconvolution network method [29] have also shown promising results in generating natural images. However, the intrinsic semantic structure of the training data is not well exploited in existing GAN networks. Typically, G utilizes D's error gradient to update its parameters, and the loss function of G is based on a binary classifier, that is, we only differentiate whether the generated sample is 'faked' or 'real'. However, the training images come from various semantic categories, therefore, they are difficult to be clustered into one 'real' class. Hence we employ a multiclass classifier as the discriminator, in which we determine whether the generated image is faked or belongs to a particular predefined category.

2.3 Style transfer learning

Our work is partially inspired by style transfer learning [7, 35], an interesting subfield recently raised in computer vision and machine learning areas. The goal of style transfer learning is to simultaneously match the visual style of a first image, captured by certain low-level statistics, and the visual content of a second image, revealed by higher-level statistics [35]. [7] employs neural representations to separate and recombine the content and style of the images. [35] trains compact feed-forward convolutional networks to generate multiple samples of the same texture but arbitrary size and to transfer artistic style from a given image to any other image.

What they have in common is that CNN features of specific layers in the network are considered as content or style representation, and the features are also used to measure the mismatch between the original image and generated image. In a similar way, we extract content and style features from two exemplary images, and then combine them to transfer to the target image. The difference between our method and [7] is that we generate images in a single-pass feed-forward way while [7] needs iterative online optimizing. Furthermore, we only need to train one model for all different style images, instead of training a specified model for each style image as in [35].

3 STYLIZED ADVERSARIAL AUTOENCODER

For convenience, we use textual character image generation (for example, scene text generation, etc.), as the background application to introduce our algorithm, though we will show more applications (such as face generation) in the experiments. Our goal is to generate images from two exemplary images, content image c and style image s, by defining and training a neural network. For character image generation, a content image is a synthesized letter image without any style or texture or background, such as A to Z, 0 to 9; a style image is an exemplary image, for example, a real word image.

As aforementioned, we split the latent variable that reveals the prior distribution of the real-world data into two parts, style feature and content feature. Content feature will be derived (through a convolutional network) from the content image, and style feature will be from the style image.

3.1 The generator

The generative network consists of two encoders(Enc_c and Enc_s) and one decoder (Dec). Enc_c encodes the content image to the content latent representation or feature z_c, and Enc_s encodes the style image to the style latent representation or feature z_s:

$$z_c = Enc_c(c),\ z_s = Enc_s(s) \tag{2}$$

Dec decodes the combined latent representations and produces the output image $\hat{x}$:

$$\hat{x} = Dec(z_c, z_s) \tag{3}$$

For convenience, we use generator G to represent the combination of Enc_c, Enc_s and Dec:

$$\hat{x} = G(s, c) \tag{4}$$

The reconstruction error is denoted by L2-loss:

$$L_{rec} = \|x - \hat{x}\| \tag{5}$$

where x is the ground truth image.

3.2 The discriminator

The output of the discriminator in existing GANs [5, 9, 10, 26] is the probability $y = Dis(x) \in [0, 1]$, which indicates probability of x being a real-world image. Training the discriminator D is to minimize the binary cross entropy:

$$L_{dis} = -\log(Dis(x)) - \log(1 - Dis(G(z))) \tag{6}$$

The goal of generator G is to produce images that D is not able to differentiate from real world images, that is, to maximize L_{dis}. As aforementioned, in existing GAN networks, a binary classifier is applied in D to determine whether the image is a real image or a generated image. However, putting all real images into one big positive class failed to leverage the intrinsic semantic structure of the training images. Therefore, we propose to use a multi-class classifier as the discriminator, in which the classifier will determine whether the input is a generated image or a particular real image category(*e.g.* , the specific character).

In our model, the output of $D(x)$ is a $(k+1)$-dimensional vector, where k is the number of image classes, indicating the probabilities of x belonging to each corresponding class(k real classes plus one class for generated/fake images). The ground truth label of a real image is denoted by $l_{real}(l_x) = [l_x^{(1)}, \cdots, l_x^{(k)}, 0]$ and the label for fake images is $l_{fake} = [0, \cdots, 0, 1]$. Note that l_x is one-hot k-dimensional category vector of each real image x, in which $l_x^{(i)} = 1$ if x belongs to the i-th class. Then the discriminator error of D is derived as:

$$L_{dis,D} = H(Dis(x),\ l_{real}(l_x)) + H(Dis(G(s, c)),\ l_{fake}) \tag{7}$$

where $H(p, q)$ denotes the sum of binary cross entropy of each elements in vector p, q:

$$H(p, q) = \sum_{i=1}^{k+1} -q^{(i)} \log p^{(i)} - (1 - q^{(i)}) \log(1 - p^{(i)}) \tag{8}$$

Cross entropy is a typical measurement that measures the similarity between two distributions. As aforementioned, G's goal is to generate images that are difficult to be determined as faked images by D. Therefore, the discriminator error of G can be denoted by:

$$L_{dis,G} = H(Dis(G(s, c)),\ l_{real}(l_x)) \tag{9}$$

3.3 Network architecture

Convolutional neural networks (CNNs) have shown significant advantages in feature representation [8, 19] and image generation [5, 6, 29], Our proposed SAAE network is based on CNN architecture,as illustrated by Figure 2.

In fact, the proposed generative network, which we formally refer as $G(s, c)$, consists of two streams of feature extraction networks followed by a generation network. Content feature extractor and style feature extractor both have three convolutional layers without down-sampling, which preserve the detail information of the exemplary images as much as possible. The input style image and content image may have different shape. For example, when generating scene text character images, the content image is an image of one character and the style image is an image of one word or multiple characters. After three convolutional layers, style feature map is reshaped to style feature vector by a fully connected layer. In order to combine with the content feature map decoded from the content image, the style feature vector is reshaped back to a feature map, which has the same shape with the content feature map. Content feature extraction network does not have any fully connected layers as the spatial information of the content image needs to be preserved. We merge the content feature map and the style feature map at the channel dimension, which means the combined feature map has half channels from the content feature and another half from the style feature. Afterwards, the generation network decodes the combined feature map to a target character image with three convolutional layers.

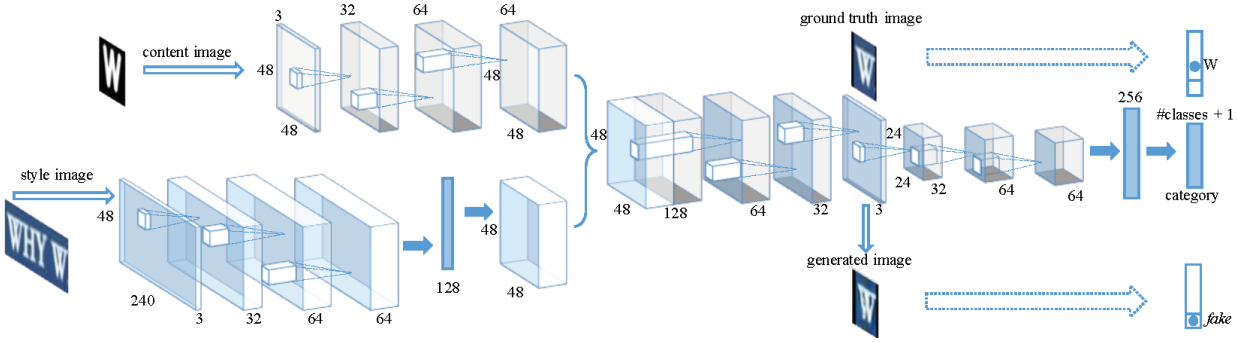

Figure 2: Architecture of the network.

The discriminative network is a generic CNN classifier consisting of three convolutional layers, one 2×2 max-pooling layer following the first convolutional layer and two fully connected layers following the last. The output layer of discriminator is a $(k+1)$-dimensional vector, indicating the probabilities of input image belonging to each class (k classes of real images and one class for faked images).

We apply batch normalization [16] to each of the convolutional layers, which accelerates convergence at the training stage. Leaky ReLU [25] is used at each layer except the last one, in which sigmoid is applied to cater each output into $[0, 1]$ (as a probability).

3.4 Training strategy

Generative adversarial networks are known to be challenging to converge [31, 36] due to their contradictory min-max objective. Another difficulty is to balance the losses from the generator and discriminator. We need to make sure that neither the generator nor the discriminator becomes too strong so that the other will be suppressed. If the discriminator is too strong and classifies all images correctly, the error gradient will be poor and the generator will not be able to learn from it. Conversely, if we allow the generator to win easily, it usually exploits a non-meaningful weakness in the discriminator (*e.g.* by coloring the entire image blue or some repeated texture), which is not desirable.

To solve this problem, inspired by step-training used in [30], we propose a 3-step training strategy to optimize our model. We denote the network parameters by θ_G, θ_D and the total number training epoch is $4n$. In $[0, n]$ epochs, we only optimize G by minimizing the reconstruction error:

$$\theta_G \overset{+}{\leftarrow} -\nabla_{\theta_G}(L_{rec}) \tag{10}$$

D is not trained at this stage because G is too weak thus the images generated by it are something like noises, which could be easily distinguished from real images. So that D cannot learn anything meaningful and may be initialized poorly. When G is able to output meaningful images, we begin to optimize D simultaneously in $[n+1, 2n]$ epochs:

$$\begin{aligned} \theta_G &\overset{+}{\leftarrow} -\nabla_{\theta_G}(L_{rec}) \\ \theta_D &\overset{+}{\leftarrow} -\nabla_{\theta_D}(L_{dis,D}) \end{aligned} \tag{11}$$

After another n epochs, D should responses meaningful error gradient that G is able to learn from it. So we begin the adversarial training stage in the second half $[2n+1, 4n]$ epochs:

$$\begin{aligned} \theta_G &\overset{+}{\leftarrow} -\nabla_{\theta_G}(L_{rec} + L_{dis,G}) \\ \theta_D &\overset{+}{\leftarrow} -\nabla_{\theta_D}(L_{dis,D}) \end{aligned} \tag{12}$$

It can be seen as that we train G by reconstruction error first and then add discriminator error to fine-tune it. This 3-step optimizing strategy helps us get stable training results.

4 EXPERIMENTS

We evaluate our approach using 4 different methods: computation of log-likelihood on MNIST dataset to measure the ability of the SAAE model to capture the data distribution; showing the visual attribute transferring on face generation; evaluating the SAAE model on scene text dataset; generating training data for supervised recognition task.

4.1 Log-likelihood analysis

Enlightened by the evaluation procedure in [9, 26], in this subsection, we evaluate the performance of the SAAE as a generative model to capture the data distribution by comparing the log-likelihood of this model to a set of hold-out images on the real-valued MNIST dataset. The basic idea of this experiment is: firstly, we train the SAAE on the training set of the MNIST dataset and $10,000$ images are then generated through the trained autoencoder. After that, we estimate the distribution of these generated data and calculate the log-likelihood of the MNIST test set (that is, a set of hold-out real-world images) over this estimated distribution. The bigger the log-likelihood, the closer the generated images are to the real world images.

To be more specific, we trained a SAAE on the generally-used training set of MNIST dataset. As shown in Figure 2 the input of SAAE actually has two parts: style image and content image. For this particular case, the content image c is a computer-print digital image, which has black background, white font color and a standard look font (*e.g.* SimSun). The size of content images and target images is 28×28. For the style image s, we randomly choose a fixed number of digit images from the training set (say, 7 we used), and arrange them horizontally to form a 196×28 image, from

Figure 3: Visualization of samples generated from our stylized adversarial autoencoder. The last column shows the closest training images in pixel-wise Euclidean distance to those in the second-to-last column, in order to demonstrate that the model has not memorized the training set.

which the generator will learn the style features(such as strokes and thickness). The ground truth image x is one of the randomly selected 7 images and have the same label with c. For each fixed c, we choose different style images s, and SAAE will generate a variety of images with the same content label while differing in styles.

We estimate the probability of the test set data under p_G by fitting a Gaussian Parzen window to 10,000 samples generated from the model and compute the log-likelihood of the MINST test dataset over this distribution. The free-parameter σ of the Parzen window is obtained by cross-validation. This procedure [4] for estimating likelihood is not an optimal benchmark due to the poor performance of Parzen estimators in high dimensional spaces, however, it is the best method available to our knowledge [9, 31].

Table 1 compares the log-likelihood results of the stylized adversarial autoencoder (SAAE) to six state-of-the-art methods. Our method shows the best performance under this criterion, outperforming AAE [26] by about 89 points. In order to demonstrate the advantage of the multiclass discriminator against a normal binary discriminator, we train two models, one with binary D and the other with multiclass D respectively (denoted by "SAAE-binary" and "SAAE" in Table 1). The log-likelihood results show that multiclass D has a better performance. Note that our SAAE-binary outperforms existing methods even using a binary D, which reveals that the latent representation in our method is sampled from real images instead of a manually-set distribution assumption(which is used in existing approaches).

Following previous methods [9, 26], we show a number of samples drawn from the trained SAAE generator in Figure 3. The last column shows the closest training images (in terms of pixel-wise Euclidean distance) to those generated images in the second-to-last column, in order to demonstrate that the SAAE model has not memorized the training set.

Table 1: Log-likelihood of test data on MNIST dataset. Higher values are better. The last two rows are our method with binary or multiclass discriminator. The reported numbers are the mean log-likelihood of samples on test set, with the standard error of the mean computed across examples.

Models	Log-likelihood
DBN [11]	138 ± 2
Stacked CAE [3]	121 ± 1.6
Deep GSN [2]	214 ± 1.1
GAN [9]	225 ± 2
GMMN + AE [24]	282 ± 2
AAE [26]	340 ± 2
SAAE-binary	402 ± 2.2
SAAE	**429 ± 2.5**

4.2 Attribute-conditioned Face Generation

We evaluate our model for face image generation on Labeled Faces in the Wild(LFW)[15] dataset. Followed the preprocessing in [36], we aligned the face images and rescaled the center region to 64×64. For each image in the training set, we blurred the image with Gaussian filter of kernel size 21×21, which only preserves background color and lineament, and used it as the content image. We applied the pre-trained model provided by [20] as the Enc_s to extract the 73 dimensional attribute score vector as the style feature vector, which describes different aspects of facial appearance such as age, gender, or facial expression, following previous method [36]. The SAAE model was trained to generate the clear face image given the blurred content image and attribute vector extracted from the style image.

Followed the evaluation procedure in [36], we generate various images with interpolated attributes by gradually increasing the values along each attribute dimension. To be more specific, we modify the value of one attribute dimension by interpolating between the minimum and maximum value. Then, we generated images by the modified attribute vector while keeping the content image fixed. As we can see in Figure 7, generated samples are visually consistent with attribute transferring. For example, by changing attributes like "eyewear", the global appearance is well preserved but the difference appears in the eye region.

In order to demonstrate the effect of the discriminator loss in SAAE, we add a hyperparameter α in G loss so that $L_G = L_{rec} + \alpha L_{dis,G}$. As shown in Figure 7, when α is small (0.05 we set), the SAAE model prefers to generate smoothed faces due to the dominant L_2 reconstruction loss, which are similar with the result in [36]. With a high weight($\alpha = 0.5$), the discriminator loss imposes the SAAE model to describe more details in face images, such as the texture of hair and wrinkle of skin, and makes the generated faces more realistic.

4.3 Model samples

In this subsection we evaluate our SAAE model on the IIIT 5k-word (IIIT5K) dataset and Chinese car license plate(PLATE) dataset.

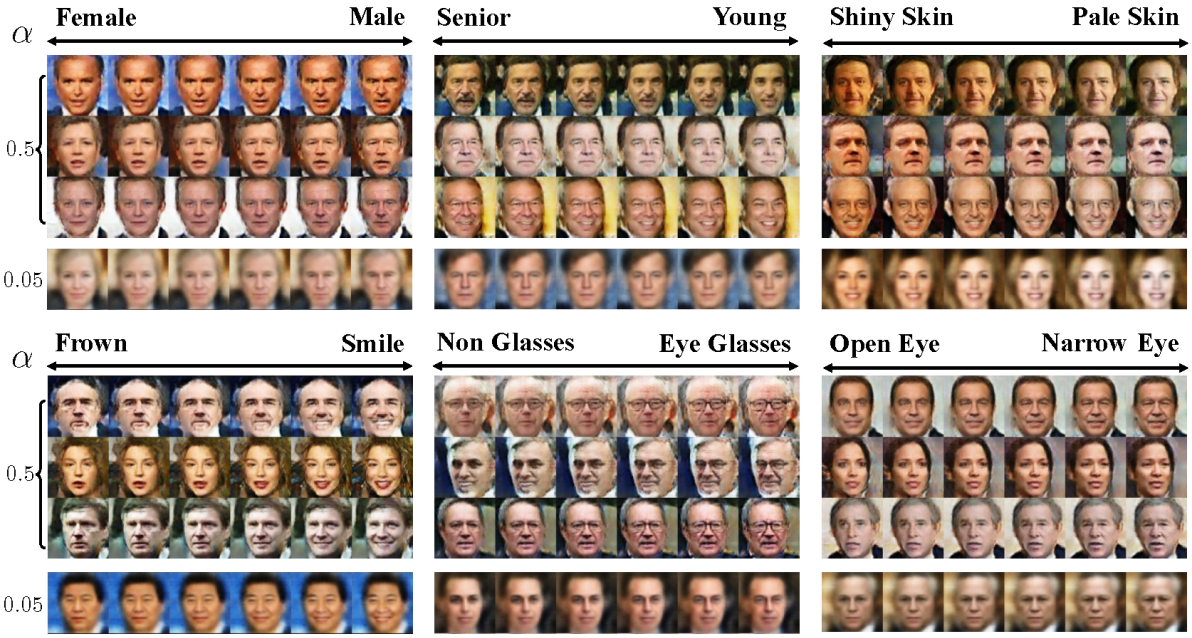

Figure 4: Attribute-conditioned image generation organized into six groups(gender, age, complexion, expression, eye wear and eye size). α denotes the weight of $L_{dis,G}$ in G loss.

Figure 5: Training data and model samples of SAAE and DCGAN. Top row: IIIT5K dataset. Bottom row: PLATE dataset.

IIIT5K [27] contains 5,000 cropped word images collected from the Internet, 2,000 of which used for training and the rest for testing. Here we model our SAAE network on the training set, which contains 9,678 characters in total. We crop each character out from the word images with the bounding box provided in dataset. In order to uniform the input shape, we transform the character images into squares by zero-padding and then resize to 48×48 pixel as the ground truth images. The reason of not resizing the images directly without zero-padding, which is a widely used method in CNNs, is that the aspect ratio is an important attribute of character style and should be preserved. For each ground truth character image x, the corresponding computer-print character image is used as the content image c, with black background, white font color and SimSun font, similar to what we have mentioned in Section 4.1. The word image, from which the ground truth character image x is cropped out, is used as the style image s(with the size of 240×48). Each word image is first resized to 48 pixels in height while keeping the aspect ratio, and then crop(randomly) or pad (in a replicating way) the image into 240 pixels in width.

PLATE dataset contains 13,345 Chinese car license plate images we collected and labeled. Each license plate has one Chinese character followed by 7 digits of alphanumeric character. All the alphabets are in uppercase. We crop each character out and resize it to 30×64 as the ground truth image x. The corresponding content image c is also a white-on-black character, but we use the standard font of Chinese car license plate instead. The license plate image itself is used as the style image s, resized to 192×64.

After training a SAAE model on each dataset, we use the model to generate samples by traversing all content images for each style image. Figure 5 shows samples randomly selected from our model and DCGAN [29] model, as well as training data for comparison. The SAAE samples appear to be much more character-like and have clearer edges and cleaner backgrounds.

It is noted that all SAAE samples are meaningful characters while some DCGAN samples appear meaningless. We owe this improvement to the well-defined content input of our model, which enables us to control the content of the output of the generator.Some DCGAN outputs look similar though they are generated from different samples in the latent distribution, due to the instability caused

Figure 6: Generative samples of one given style image. Top row: IIIT5K dataset. Bottom row: PLATE dataset. For each set of generative samples, the style image is shown on top left, marked by red rectangle. For PLATE dataset, we hid the first Chinese character of style images for privacy reasons

by binary discriminator. In such case, the generator is not able to learning category information from the binary discriminator, so that it turns to generate some particular images to confuse the discriminator.

In order to visualize the stylization property of our stylized adversarial autoencoder, we show several sets of generative samples in Figure 6, both on IIIT5K and PLATE dataset. In each set, we choose one exemplary style image and traverse all content images and labels. The result demonstrates that SAAE model can transfer the character style from exemplary style images to the content images. On IIIT5K dataset, SAAE model extracts not only the features of font color and background color, but also the features of font-size, font-weight, aspect ratio, even some stroke style of particular images. On PLATE dataset, some detail features such as character tilt and license plate boundary are captured by the model and then decode to the generative samples. Although our training set does not cover all the Chinese characters used in car license plate, SAAE model still can reconstruct the characters that never appear in the training set, with given exemplary content images.

An interesting finding is that, if we input an all-zero content vector and an all-black content image into the model, it will generate a pure background image without any character on it, as shown in the last of each set of PLATE samples in Figure 6. This finding enables us to generate arbitrary arrangement of characters by generating a background image first and then embed generated character images

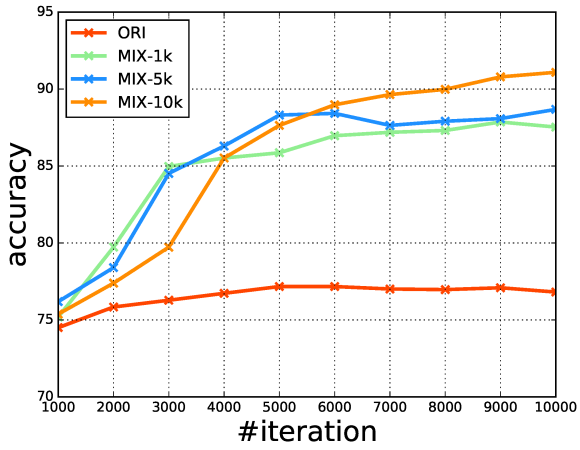

Figure 7: Recognition accuracy with respect to iteration count on different training set.

on it. We use this method to generate training data in Section 4.4 and show that it improves the supervised recognition task.

4.4 Data generation for supervised learning

Deep Neural Networks(DNNs) have shown significant superiority in supervised l earning, but it relies on massive labeled training data. A deep model is easily over-fitting on a small number of training data. In this section we use SAAE model to generate training data for recognizing Chinese car license plate.

We collect a double row plate dataset (DR-PLATE) consist of 1738 double row plate images, each contains two rows of characters and can not be well recognized by a model trained from normal single row plate images. We hold out 900 images as the testing set, while the rest 838 images are too few to train a deep optical character recognition (OCR) network. Therefore, we apply a SAAE model trained on DR-PLATE dataset to generate training data by the following steps. Firstly, choose a training sample in DR-PLATE as the exemplary style image. Then generate a background image and 7 randomly-chosen character images, following the method introduced in Section 4.3. Thirdly, embed the character images on the background image by the position defined in car license plate standard. Scaling and rotation augmentation are then applied on the generated license plates. By means of this method, we can generate plenty of training data without any manual annotation.

We evaluate the data generation quality by measuring the recognition accuracy on DR-PLATE. We use the CNN model introduced in [32], followed by 7 fully connected layers to predict 7 digits of character respectively. The mini-batch size is 32 and we use fixed learning rate of 0.001. We train the recognition network on 4 sets of training data: the original training data in DR-PLATE(ORI), mixed dataset of original data and $1k$, $5k$, $10k$ generated data respectively (MIX-1k, MIX-5k, MIX-10k). The recognition accuracy on testing data is recorded every $1k$ iterations and the result is plotted in Figure 7. We improve the recognition accuracy from 77.17% to 91.09% by generating $10k$ more training data. Figure 7 shows that with

more generated data mixed into the training set, the model converges more slowly, but the classification accuracy become better and better. This result demonstrates that our SAAE model improves supervised learning by generating data.

5 CONCLUSION

In this paper, we proposed the stylized adversarial autoencoder, which is an autoencoder-based generative adversarial network for image generation. The split of the latent vector enables us adjusting the content and the style of the generated image arbitrarily by choosing different exemplary images, and the multiclass discriminator utilizes the inter class variation and contributes to generating more realistic. We demonstrated the superior results of SAAE model in hand-writing digits, scene text and face images generation as well as the improvement in supervised recognition task. Further work will be focused on optimizing the network structure for higher generation quality. Extending this framework to other application(such as semi-supervised feature learning) would be interesting as well.

ACKNOWLEDGMENTS

This paper is partially supported by NSFC (No.61272247, 61533012, 61472075), the 863 National High Technology Research and Development Program of China (SS2015AA020501), the Basic Research Project of "Innovation Action Plan" (16JC1402800) and the Major Basic Research Program (15JC1400103) of Shanghai Science and Technology Committee.

REFERENCES

[1] Yoshua Bengio, Ian J Goodfellow, and Aaron Courville. 2015. Deep learning. *An MIT Press book in preparation. Draft chapters available at http://www. iro. umontreal. ca/ bengioy/dlbook* (2015).
[2] Yoshua Bengio, Eric Laufer, Guillaume Alain, and Jason Yosinski. 2014. Deep Generative Stochastic Networks Trainable by Backprop. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*. 226–234.
[3] Yoshua Bengio, Grégoire Mesnil, Yann Dauphin, and Salah Rifai. 2013. Better Mixing via Deep Representations.. In *ICML (1)*. 552–560.
[4] Olivier Breuleux, Yoshua Bengio, and Pascal Vincent. 2011. Quickly generating representative samples from an rbm-derived process. *Neural Computation* 23, 8 (2011), 2058–2073.
[5] Emily L Denton, Soumith Chintala, Rob Fergus, and others. 2015. Deep Generative Image Models using aï£ij Laplacian Pyramid of Adversarial Networks. In *Advances in neural information processing systems*. 1486–1494.
[6] Alexey Dosovitskiy, Jost Tobias Springenberg, and Thomas Brox. 2015. Learning to generate chairs with convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1538–1546.
[7] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. 2015. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576* (2015).
[8] Ross Girshick. 2015. Fast r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision*. 1440–1448.
[9] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in Neural Information Processing Systems*. 2672–2680.
[10] Karol Gregor, Ivo Danihelka, Alex Graves, Danilo Rezende, and Daan Wierstra. 2015. DRAW: A Recurrent Neural Network For Image Generation. In *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*. 1462–1471.
[11] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. 2006. A fast learning algorithm for deep belief nets. *Neural computation* 18, 7 (2006), 1527–1554.
[12] Geoffrey E Hinton, Michael Revow, and Peter Dayan. 1995. Recognizing handwritten digits using mixtures of linear models. *Advances in neural information processing systems* (1995), 1015–1022.
[13] Geoffrey E Hinton and Ruslan R Salakhutdinov. 2006. Reducing the dimensionality of data with neural networks. *Science* 313, 5786 (2006), 504–507.
[14] Geoffrey E Hinton and Richard S Zemel. 1994. Autoencoders, minimum description length, and Helmholtz free energy. *Advances in neural information processing systems* (1994), 3–3.
[15] Gary B. Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. 2007. *Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments*. Technical Report 07-49. University of Massachusetts, Amherst.
[16] Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167* (2015).
[17] Diederik P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. 2014. Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems*. 3581–3589.
[18] Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013).
[19] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. 1097–1105.
[20] Neeraj Kumar, Alexander C Berg, Peter N Belhumeur, and Shree K Nayar. 2009. Attribute and simile classifiers for face verification. In *Computer Vision, 2009 IEEE 12th International Conference on*. IEEE, 365–372.
[21] Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, and Ole Winther. 2015. Autoencoding beyond pixels using a learned similarity metric. *arXiv preprint arXiv:1512.09300* (2015).
[22] Nicolas Le Roux, Nicolas Heess, Jamie Shotton, and John Winn. 2011. Learning a generative model of images by factoring appearance and shape. *Neural Computation* 23, 3 (2011), 593–650.
[23] Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y Ng. 2009. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th annual international conference on machine learning*. ACM, 609–616.
[24] Yujia Li, Kevin Swersky, and Richard Zemel. 2015. Generative moment matching networks. In *International Conference on Machine Learning*. 1718–1727.
[25] Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. 2013. Rectifier nonlinearities improve neural network acoustic models. In *Proc. ICML*, Vol. 30.
[26] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, and Ian Goodfellow. 2015. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644* (2015).
[27] Anand Mishra, Karteek Alahari, and CV Jawahar. 2012. Scene text recognition using higher order language priors. In *BMVC 2012-23rd British Machine Vision Conference*. BMVA.
[28] Andrew Ng. 2011. Sparse autoencoder. *CS294A Lecture notes* 72 (2011), 1–19.
[29] Alec Radford, Luke Metz, and Soumith Chintala. 2015. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434* (2015).
[30] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*. 91–99.
[31] Eder Santana, Matthew Emigh, and Jose C Principe. 2016. Information Theoretic-Learning Auto-Encoder. *arXiv preprint arXiv:1603.06653* (2016).
[32] Baoguang Shi, Xiang Bai, and Cong Yao. 2015. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *arXiv preprint arXiv:1507.05717* (2015).
[33] Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research* 15, 1 (2014), 1929–1958.
[34] Zhuowen Tu. 2007. Learning generative models via discriminative approaches. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 1–8.
[35] Dmitry Ulyanov, Vadim Lebedev, Andrea Vedaldi, and Victor Lempitsky. 2016. Texture Networks: Feed-forward Synthesis of Textures and Stylized Images. *arXiv preprint arXiv:1603.03417* (2016).
[36] Xinchen Yan, Jimei Yang, Kihyuk Sohn, and Honglak Lee. 2016. Attribute2image: Conditional image generation from visual attributes. In *European Conference on Computer Vision*. Springer, 776–791.