Customize the Features of Face Images Generated by GAN

Ruonan Feng [1] **Yi Yang** [1] **Lei Guo** [1]

Abstract

Many GAN models are dedicated to control the features of generated images such as CycleGAN and conditional GAN. We implement and improve a novel model TL-GAN for customizing the features of generated face images. First, we use NVIDIA's pg-GAN as a pre-trained network to generate face images, then design and train a multi-task CNN for predicting face feature labels on the celebA dataset from scratch. We also explore different GLM models and implement orthogonalization to discover the feature axes in latent space. The advantages of this work are that not only the features of the generated face images are continuously controllable, but also the construction of the whole structure is very efficient and fast, and it is also scalable on multiple features and multiple datasets.

1. Introduction

The original version of GAN uses a random noise as the input of the Generator to generate an extremely realistic image[1], however, the process of how the Generator deals with random noise is a closed black box. The down side is that we are unable to control the features of the generated image and to understand how each dimension of the random noise affects the features of the generated image. Controlling the features of the generated image is a very important topic: people usually need only images that meet their certain needs, rather than arbitrary images even they are realistic enough. In order to fulfill these needs, a variety of improved GAN model has been proposed, such as the successful CycleGAN[2] and StarGAN[3]. However, training a GAN is very costly. It usually takes several days to train on multiple GPUs to bring up a satisfactory model. We implement a novel and efficient model called TL-GAN, which controls the features of the generated human face images by exploring the feature axes in the latent space of the Generator, and usually takes only a few hours of training to obtain excellent results[4].

1.1. Research contributions

We improve the TL-GAN by modifying the following: (1) The original version of the CNN model simply generates a 40-dimensional vector as predicted feature labels at the output layer of the network. We improve the CNN structure by using shared multiple convolutional layers at the beginning, which will extract features of images, then add 40 sub-networks consists of fully connected layers to predict each feature. (2) We fit several potential GLM models between random noise and predicted feature labels to find the best way to represent feature axes in latent space.

2. Related Work

Variants of GAN dedicated to controlling the features of the generated images can be basically divided into two types of goals: converting styles or meeting conditions.

CycleGAN is a typical style conversion network, which presents an approach for learning to translate an image from a source domain to a target domain in the absence of paired examples[2]. unlike other transferring models using a training set of aligned image pairs, CycleGAN has a great advantage on several tasks where paired training data does not exist, including collection style transfer, object transfiguration, season transfer, photo enhancement, etc.

Pix2pix is another representative of the style conversion GAN, which investigates conditional adversarial networks as a general-purpose solution to the image-to-image translation problem[5]. These networks not only learn the mapping from an input image to output image, but also learn a loss function to train this mapping, which suggests we can achieve reasonable results without hand-engineering our loss functions either. However, the disadvantage of GANs for style conversion is that we can only convert the style of one image to the style of another, and we cannot fine-tune the style of the image. In addition, the needs to do 100 different types of conversions require to train 100 different networks, which is extremely inefficient.

StarGAN is to address limited scalability and robustness in handling more than two domains, a scalable approach

[1] Worcester Polytechnic Institute. Correspondence to: <rfeng@wpi.edu, yyang19@wpi.edu, lguo@wpi.edu>.

CS541 Special Topics Course Final Project, Worcester Polytechnic Institute. Copyright 2017 by the author(s).

called StarGAN is proposed, which can perform image-to-image translations for multiple domains using only a single model[3]. StarGAN successfully learns multi domain image translation between multiple datasets by utilizing a mask vector method that enables StarGAN to control all available domain labels.

Conditional GAN, the conditional version of GAN, can be constructed by simply feeding the condition to both the generator and discriminator. For example, conditional GAN can generate MNIST digits conditioned on class labels[6]. The advancement of conditional GAN is that images can be generated on the condition of custom features. However, when you want to add new tuneable features, you must retrain the entire model. In addition, the training data must be a single dataset containing all the feature labels, rather than multiple data sources containing different feature labels.

The model we implemented overcomes all the above shortcomings. First, TL-GAN uses pre-trained GAN to generate face images rather than train a new GAN for itself, which means that we don't have to retrain the whole GAN when adding new controllable features. Second, TL-GAN trains a CNN model to predict the feature labels of face images from any known data sources and generates corresponding feature labels for the face image generated by the pre-trained model, which usually only needs several hours of training on a single GPU. Allowing a variety of data sources also ensure the scalability of the model. Finally, by fitting a generalized linear model between the initial random noise and the feature labels of the generated face images, we made the feature adjustment continuous, which far exceeds other GANs that can only achieve discontinuous style conversions.

3. Proposed Method

In order to modeling the latent space to control image generated by GAN, we first deployed a pre-trained PG-GAN model to generate more than 8,000 random images and saved their corresponding random noises for later use. Then, we trained a multi-task CNN model using Tensorflow on CelebA dataset to discover the relation between faces and 40 different attributes (Gender, No-Beard, Young/Old, Bald, etc.). In our CNN model, we first implemented 4 convolutional layers as the shared body of our model. Then, we implemented 40 different blocks of fully-connected layers with each block designated for one attribute, and at the end of each block, sigmoid function was used to project output into a 0 to 1 range to arrive at a probabilistic result. Thus, our model has the descriptive ability for 40 different attributes. After the model was successfully trained, we inputted 8,000 + faces previously generated by the PG-GAN model, and received 8,000 + corresponding labels as outputs from CNN model. At this point, we had the noises and labels corresponding to each of 8,000 + images, and we could proceed to the final step that is to train a generalized linear model having noises as input and labels as output. Once, we obtained the weights of GLM, we can adjust the attribute by moving random noise along the feature axis.

4. Experiment

After several experiments, we have finalized the following CNN architecture:

1st conv-layer: kernel size 64*4*4, stride size 2*2

2nd conv-layer: kernel size 128*4*4, stride size 2*2

3rd conv-layer: kernel size 256*4*4, stride size 2*2

4th conv-layer: kernel size 512*4*4, stride size 2*2

After each convolutional layer we add batch normalization where takes 0.01 as epsilon and 0.9 as the decay rate. The activation function for all these layers is Leaky Relu and there is no pooling layers. The design of this CNN structure is learned from the Discriminator of DCGAN[7].

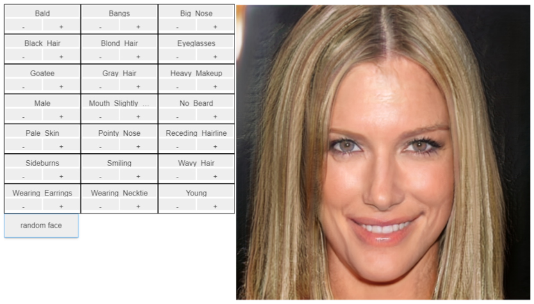
For each block of each attribute, we have 4 fully-connected layers and the numbers of neurons in each fully-connected layer are 100, 50, 30 and 1. The activation function is sigmoid function and the loss function is the sum of sigmoid cross-entropy loss from each fully-connected block plus the L2 regularization with the beta as 0.01. We use AdamOptimizer with 0.05 as the learning rate, 0.1 as the decay rate, 1000 as the decay step and the batch size is 64.

We also conducted the following experiment to arrive at our final model: (1) Added one batch normalization layer before each activation layer, and with BN layer, the model did converge faster than the original version. (2) Tested 3 different learning rate, 0.5, 0.05, and 0.005. We found with 0.05 learning rate, the training process would be more stable at the early stage and overall accuracy increased from 0.76 to 0.78. (3) After tested different kernel size of convolution layers, we found that accuracy would improve as the increasing of the kernel size.

The overall accuracy we have for our final model is 0.78, and half of 40 attributes have test accuracy over 0.85. The top three most accurately predicted attributes are Bald with 0.97, Gray hair with 0.969, and mustache with 0.962.

5. Results

We test our model to see if it can control the change of corresponding features of generated images by controlling the latent space. First, we initialized a 512-dimension random noise and generated an image. Second, we move the latent space of the corresponding feature by 1 unit and generated another image. Then, the feature on the image is expected to change by 1 unit accordingly. Since we made the latent vectors orthogonal to each other, the features of the generated images are less correlated to each other. The results of changing the noise vector for some features are shown below.



Our model makes its latent space transparent by extracting features for a pretrained GAN generator. The features of the images make corresponding changes when the noises move along the latent space.

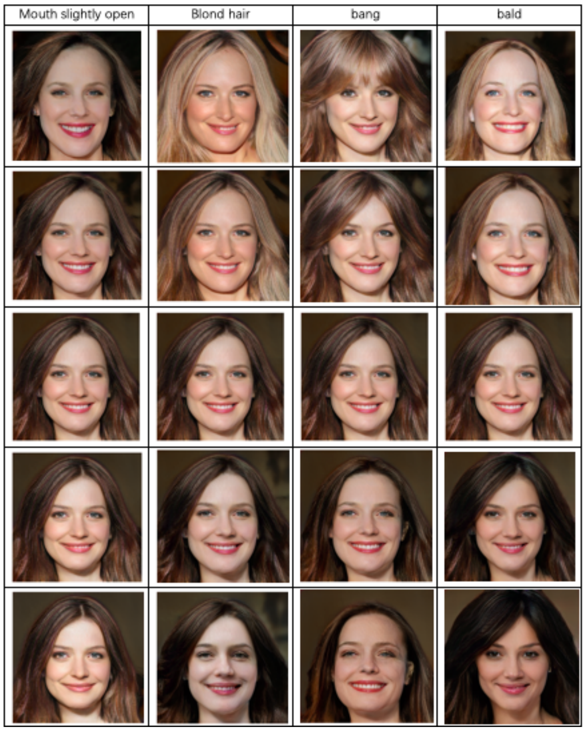
Compared to the related work like conditional GAN, StarGAN, CycleGAN and pix2pix, our model has these advantages: (1) When you want to add new tunable features to the generation process, you do not have to retrain the GAN model which can reduce a big amount of computing cost. Besides, we do not have to rely on a single dataset with all required features by adding dataset with new features to perform the training. Our model can tune multiple features based on one model instead of multiple models and it can be done efficiently. (2) Previous work can only adjust one feature from one to another while our model can tune one feature gradually among a series of discrete states so that the features can be changed continuously by users.

In summary, our model has good efficiency and flexibility. The quality of the generated images can be improved by using a better pre-trained generator without re-training the feature extraction model. Besides, the features can be added by adding more dataset with related labels into the training process efficiently.

We also built an interactive GUI for users to change the features as they like. Users can click the - or + buttons to control the direction and extend the feature changed continuously. If a user wants to fix one feature and change others, it can realized by fix the upper button of the corresponding feature.

6. Discussion

There is a shortcoming of our model: if you increase the amount of the bald feature, the generated face became older. This problem is due to the bald feature and Young feature are correlated by nature so if you change one the other will be changed correspondingly. Our model cannot deal with this problem so far since the feature variable is trained from existing dataset with feature labels. The further work will focus on the correlation of the latent vectors so that the user can change one feature and other features are fixed.



7. Conclusions and Future Work

One-third of 40 attributes did not perform well on test set. A possible improvement is based on the correlation between each attribute, similar attributes can be categorized into one training group, and we can train multiple models on each group of attributes. This approach could reduce the variance within feature space, and would ease the difficulty of multi-task learning.

References

[1] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... Bengio, Y. (2014). Generative adversarial nets. In Advances in neural information processing systems (pp. 2672-2680).

[2] Zhu, J. Y., Park, T., Isola, P., Efros, A. A. (2017). Unpaired image-to-image translation using cycle-consistent adversarial networks. In Proceedings of the IEEE international conference on computer vision (pp. 2223-2232).

[3] Choi, Y., Choi, M., Kim, M., Ha, J. W., Kim, S., Choo, J. (2018). Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 8789-8797).

[4] Shaobo G,Generating custom photo-realistic faces using AI ,https://blog.insightdatascience.com/generating-custom-photo-realistic-faces-using-ai-d170b1b59255

[5] Isola, P., Zhu, J. Y., Zhou, T., Efros, A. A. (2017). Image-to-image translation with conditional adversarial networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1125-1134).

[6] Mirza, M., Osindero, S. (2014). Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784.

[7] Radford, A., Metz, L., Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434.