Face-to-Painting Machine

Lingfeng ZHU lzhu88@wisc.edu
Zhuoyan XU zxu444@wisc.edu
Cecily LIU qliu273@wisc.edu

Abstract

We propose a face-to-painting machine, which can identify a person and then produce a portrait of him/her automatically based on a given painting style. On the first part of our project, we implement transfer learning of our own datasets to test the performance of different CNN architectures. We use several convolutional architectures and ResNet achieves the best result. Then, we use FaceNet methods to perform the face verification based on different CNN architectures: ResNet34, ResNet50, VGG16 and VGG19. The ResNet34 model gives the best test result on the LFW dataset, the test accuracy reaches 81.47% after only 20 epochs, with AUC 0.90. We also introduce a neural style transfer method to generate the portraits in different painting styles. We choose the pre-trained VGG19 architecture to implement such method and the generated portraits are at least qualitatively comparable to common portrait paintings. Our project can be used in many real world applications, like generating tourist attraction souvenir, building automatic painting system and so on. Our face-to-painting machine is easy to interpret, easy to train and it provides some insights into potential applications of deep learning methods. Based on our project, people can implement deep learning methods into wider fields and improve our methods to get better results.

1. Introduction

Deep Learning techniques on face images has ameliorated a lot with the algorithm development. Face analysis techniques have applied a lot in our daily lives, ranging from privacy protection, Human - Computer Interaction to public security and changes in lifestyles.

The two main areas on facial image techniques are face recognition and verification. Face recognition, which is the task of recognizing and determining the person's identity based on the face image, has been an active research topic in computer vision. It compares the features from the detected image to the ones in the repository, then pick out the one with higher similarity. Face verification is another widely used algorithm in modern world, such as self-driving cars, face login and especially when you go through customs, the machine will recognize when you scan your profile photo and face, to make sure the same person. It is more like a one to one match problem. The algorithm extracts the facial features in two different photos to obtain similarity, then compare it with the threshold we have set in order to tell us whether they are the same person.

Besides, Neural Style Transfer is another optimization method we use in this project, by which people can generate new images based on content images and style reference images. Such method can transfer style from reference images, which could be some renown painter's tyle, to our content image. As a result, the generated image could be a combination of original content and new style.

Our project is just about the applications of these techniques. First, based on the different formats and size, we perform Transfer learning to images we have scratched and collected online. We select pre-trained architectures to perform Transfer Learning over our two datasets. The next part of our project is to improve and increase the accuracy of face verification. We try out different CNN architectures VGG and ResNet convolutional architectures to perform face verification by using FaceNet techniques, then use Triplet Loss to obtain our gradient, loss function and train the model. In the final part, we perform Neural Style Transfer to generate new painting stylistic images just based on a human photo and a painting. We select the pre-trained architecture to generate our desired images.

2. Motivation

Face image analysis is an increasing popular research topic in computer vision and deep learning. It is easy and natural to formulate well-posed problems. Every year, many novel algorithms and methods spring up in Deep Learning community to improve the performance and analysis on face images applications.

The idea of first part of our project is trying to directly address the problem of low accuracy and elevate the overall analysis performance of face verification that has involved in our everyday lives. The improvement of accuracy could help us get precise identifications and verification to increase the security of our private information, more and

convenient access to smart devices and so on. Moreover, the painting-stylistic person images we generated and achieved at the end of the project could play a significant part. This model can be used to transfer one specific style from one painting to another everyday object photos, not only just for people images. Based on this feature and considering people's crazy addiction to photograph editing apps, the applications of Neural Style Transfer could be extended to fancy filter for people to try out different filter effects.

Our face-to-painting machine can be an interesting project. Based on our project, people can construct AI painters, who can draw portraits or other kinds of paintings by themselves. In addition, such project can also be used in tourist attractions to generate souvenir automatically for visitors. Besides, museums can use this method to repair damaged famous paintings or produce new ones. There are many other potential applications and such methods are really useful in many fields.

3. Related Work

3.1. FaceNet: face verification

There is several previous work about face recognition and verification such as [14] and [15] which use deep network and it is similar to our approach. They employ a complex system of multiple stages. They also use the output of deep convolutional neural network to measure the weighted chi-square of two certain images and binary SVM classifier to conduct face verification.

More specifically, Taigman et al. in [15] propose a multi-stage approach that aligns faces to a general 3D shape model. They use the deep network to conduct a face recognition task. They also came up with Siamese network where they directly optimize the L1-distance between two face features. Their best performance on LFW (97.35%) stems from an ensemble of three networks using different alignments and color channels. The predicted distances (non-linear SVM predictions based on the χ^2 kernel) of those networks are combined using a non-linear SVM.

The idea we use in this project is based on [10], which is similar to what we have discussed above. We use the feature vectors got from deep convolutional neural networks to measure the similarity between two images, and conduct a face verification based on the tiplet loss.

We use two deep convolutional architecture. The first is VGG based on [12]. The second one we use is ResNet based on paper [6]. All these architectures has high performance on image recognition.

3.2. Neural Style Transfer

In previous work, the application of Neural Style Transfer in generating Landscape paintings is given by [4], which offers a path forward to an algorithmic understanding of how humans create and perceive artistic imagery. Based on this, [7] introduces a deep-learning approach to photographic style transfer that handles a large variety of image content while faithfully transferring the reference style. This approach constrains the transformation from the input to the output to be locally affine in colorspace, and expresses this constraint as a custom fully differentiable energy term. Such method successfully suppresses distortion and yields satisfying photorealistic style transfers in a broad variety of scenarios, including transfer of the time of day, weather, season, and artistic edits[7].

In addition, [5] extends the existing method to introduce control over spatial location, color information and across spatial scale. They enable the combination of style information from multiple sources to generate new, perceptually appealing styles from existing ones. They also claim that these methods can be used to more efficiently produce large size, high-quality stylisation[5].

[3] proposes StyleBank, which is composed of multiple convolution filter banks and each filter bank explicitly represents one style, for neural image style transfer. This method is the first style transfer network that links back to traditional texton mapping methods, and hence provides new understanding on neural style transfer. This method is easy to train, runs in real-time, and produces results that qualitatively better or at least comparable to other existing methods[3].

[11] presents a new technique for transferring the painting from a head portrait onto another. This technique only requires the example painting and is not restricted to a specific style. They impose novel spatial constraints by locally transferring the color distributions of the example painting, which better captures the painting texture and maintains the integrity of facial structures[11].

4. Proposed Method

4.1. Transfer learning

The first thing we do is to use popular architecture to conduct transfer learning on our dataset. To train our own data on the CNN(especially some deep architecture) from scratch may takes a lot of time. Using some other pre-trained architecture to extract the features in our image and apply this features to our own dataset is how transfer learning works, which is shown in figure1.

In image classification or verification, we want to detect the edges and the color contrast on the image, the shape of the object, and so on. The previous layers and middle layers usually do these jobs and get feature vector of every image and applied classification models on feature vectors. As we can see in the picture, we use some pre-trained model with trained parameter to do feature extraction and applied our new classifier. Since the popular architecture's perfor-

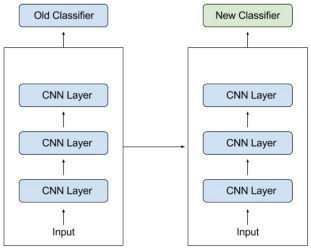


Figure 1: Transfer Learning[1]

mance, and the similarity of our data and the training data of those architectures, this approach will provide better result than training every parameter on our own dataset from scratch.

The architecture we use is VGG and ResNet with following approaches:

- VGG16: Freeze all the Conv layers + 3 FC layers, train 2 FC layers + an additional output layer.
- VGG19: Freeze all the Conv layers + 5 FC layers, train 1 FC layers + an additional output layer.
- ResNet50: Freeze all Conv layers, train the output layer.
- ResNet101: Freeze all Conv layers, train the output layer.

We change the number of nodes in last layers of all the architecture to the number of our output class.

4.2. FaceNet:Triplet loss

In this days there are many progress in the field of face verification. Face verification is not the same as well-known image classification. Face verification conducts a 1-to-1 matching to verify whether the person from two images are the same one, instead of using tons of images in each of the classes(like person) to train the model and do the classification. Face verification is widely used in customs, mobile platforms or other forms of technology.

The basic idea is based on the FaceNet paper[10]. The paper proposed a system ot new algorithm called faceNet, which learns a mapping from image to a Euclidean space as feature vectors where can measure similarity. Once get feature vectors, the following can be systematic. Feature vectors can be applied for some classification method, and can be used to measure the distance between two certain images and apply some clustering method.

4.2.1 notation

Anchor image An image chosen from a specific person.

Positive image All the images that of the same person as it in anchor image.

Negative image All the images that of the different person from the person in anchor image.

4.2.2 model architecture

Our model architecture is the same as FaceNet paper, except we use different convolutional neural network, as shown in figure2.

Figure 2: [Florian Schroff et al.2015]

The batch input first go through a deep CNN and then applied on a L_2 normalization, which result in embedding feature vectors. The embedding is represented by $f(x) \in \mathbb{R}^d$ with additional constrain $\|f(x)\|_2 = 1$. Then we use embedding feature vectors to conduct analysis.

4.2.3 Triplet loss

In this model we use triplet loss, which is motivated in [16]. We want to make sure an anchor image x_i^a of a certain person is always closer to the positive image x_i^p of the same person than it is to any negative image x_i^n of that person. As we can see in figure3

Figure 3: [Florian Schroff et al.2015]

Thus, we want:

$$\|f(x_i^a) - f(x_i^p)\|_2^2 + \alpha < \|f(x_i^a) - f(x_i^n)\|_2^2 \quad (1)$$

$$\forall (f(x_i^a), f(x_i^p), f(x_i^n)) \in \mathcal{T}$$

where $\mathcal{T}$ is a set of all possible triplets in training set with cardinality N. α is called margin that enforced between positive and negative pairs.

Then the loss can be defined as:

$$L = \sum_i^N \left[\|f(x_i^a) - f(x_i^p)\|_2^2 - \|f(x_i^a) - f(x_i^n)\|_2^2 + \alpha\right]_+$$

we define the loss to be the maximum among actual loss and 0 because we want to slow down the gradient when case is easier.

4.2.4 Triplet selection

According to FaceNet paper[10], most of the triplets will satisfy the constrain shown in equation 1, generating all possible triplets would waste computational power and result in slower convergence. Thus, it is necessary to select triplet that violate the constrain in equation 1. Given an anchor image x_i^a, we want to select:

$$\text{hard positive} : \text{argmax}_{x_i^p} \left\| f\left(x_i^a\right) - f\left(x_i^p\right) \right\|_2^2$$
$$\text{hard negative} : \text{argmin}_{x_i^n} \left\| f\left(x_i^a\right) - f\left(x_i^n\right) \right\|_2^2$$

According to the paper and our experiment, it is impossible to compute hard positive and negative through all training dataset. Additionally, this scheme will let some outliers(such as mislabelled or poorly imaged faces) dominate the result, we select hard positives and negatives within a minibatch as suggested in paper.

Following the FaceNet paper[10], we use all anchor-positive pairs in a mini-batch instead of compute the hardest positive, while still select the hard negative. Since using all anchor-positive was more stable and converge slightly faster at the beginning of the training according to the FaceNet paper.

As paper mentioned(which confirmed by our experiment), in practice, selecting the hardest negatives can result in local minimum. To mitigate this, we select x_i^n through

$$\left\| f\left(x_i^a\right) - f\left(x_i^p\right) \right\|_2^2 < \left\| f\left(x_i^a\right) - f\left(x_i^n\right) \right\|_2^2$$

The faceNet paper called these negative exemplars *semi-hard*, since they are further from the anchor than the positive exemplar, but still hard since its square distance is closed to the anchor-positive distance.

The important thing to note is we need to choose suitable triplet to ensurer fast convergence and high performance.

such as Gal Gadot, Robert Downey jr etc.

4.3. Neural Style Transfer

Given a photo of human face, our goal is to generate a corresponding portrait painting based on this photo and a well chosen style image. To do so, We perform the neural style transfer based on the given images and a well trained CNN model.

4.3.1 Notations

Pre-trained model Choose the pre-trained VGG19 model to perform a transfer learning. Figure 4 shows the construction of the chosen network[13].

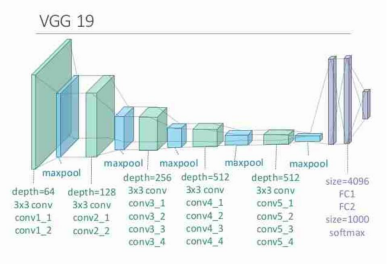


Figure 4: Construction of VGG19

Content Image Take the given photo of human face as the content image. Let C denote this content image. For example, I just choose one photo from our dataset as the content image (figure 5).

Figure 5: Content image: Gal Gadot

Generated Image We can initialize the generated image by adding some noise to the content image. Let G denote our generated image. Figure 6 is the initialized image based on figure 5.

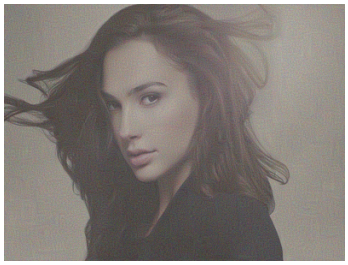


Figure 6: Initialized image: Gal Gadot

Style Image To set the style of our portrait, we can choose one of famous paintings as our style image. Let S denote this style image.

4.3.2 Cost function

One of our goals is to make the generated image G have similar content as the content image C. To do so, choose

one middle layer l which can detect some features of the input images from our network. Then, we can define the content cost function as:

$$Cost_{\text{content}}(C,G) = \frac{\sum_{\text{all entries}} \left(a^C - a^G\right)^2}{4 \times n_H \times n_W \times n_C}$$

Where n_H, n_W and n_C denotes the height, width and number of channels of layer l. a^C and a^G denote the activations of hidden layer l if we set C and G as input of our pre-trained network respectively. Here a^C and a^G are $n_H \times n_W \times n_C$ tensors.

Another goal is to make the generated image G has similar style as the style image S. To compute the style image cost, we need to calculate the style matrix first. Consider layer l as an example, if we unroll the activation tensor of layer l into a matrix V with $(n_H \times n_W)$ columns and n_C rows, then, the style matrix is defined as:

$$W^{[l]} = VV^T$$

Here W is a $n_C \times n_C$ matrix. The value W_{ij} measures the similarity of the ith filter and the jth filter of this layer l.

Similar as content cost, if we set S and G as the inputs of our network respectively, we can calculate the corresponding style matrices W^S and W^G. Then, the style image cost of layer l is defined as:

$$Cost^{[l]}_{\text{style}}(S,G) = \frac{\sum_{i=1}^{n_C}\sum_{j=1}^{n_C}\left(W^S_{ij} - W^G_{ij}\right)^2}{4 \times n_C^2 \times n_H^2 \times n_W^2}$$

To make the output better, it is common to calculate the style costs for many chosen layers and compute the weighted average style cost:

$$Cost_{\text{style}}(S,G) = \sum_l \lambda^{[l]} Cost^{[l]}_{style}(S,G)$$

Where $\lambda^{[l]}$ is the weight of layer l.

Finally, define the total cost based on the content cost and the style cost:

$$Cost(G) = \alpha Cost_{\text{content}}(C,G) + \beta Cost_{\text{style}}(S,G)$$

Here α and β are weight parameters.

5. Experiments

5.1. Dataset

5.1.1 FaceNet

For the face verification using FaceNet, we use Labeled Face in the Wild (LFW) dataset. It is a database of face photographs designed for studying the problem of unconstrained face recognition. The data set contains more than 13,000 images of faces collected from the web, for nearly 6000 people. Among of them, around 1700 people have two or more images.[2]

5.1.2 Transfer Learning and Neural Style Transfer

We use icrawler 0.6.2 package to scrape images from Google Image Search by ourselves. We perform some prerpcessing to the images and split them into train and test samples. This database contains images of 15 celebrities like Gal Gadot, Robert Downey Jr etc. There are around 7500 images in total. This dataset is mainly used in the transfer learning part and the neural style transfer part.

5.2. Transfer learning

5.2.1 Image Augmentation

In transfer learning, we use the image we scraped on Google image search, the image augmentation we apply on our image is:

- Random Rotation: Rotate the image by angle 90 degrees or 180 degrees.
- Resize: Resize the input PIL Image to the given size 256×256.
- Random Vertical Flip: Vertically flip the given PIL Image randomly with a given probability 0.5.
- Normalize: Normalize a tensor image with mean (0.485,0.456,0.406) and standard deviation (0.229,0.224,0.225) in each of RGB channel.

5.2.2 Experimental setup

To begin our experiment, we split out dataset into train dataset(80%) and test dataset(20%). There are around 6000 images in train dataset and 1500 in test dataset. We set 30 epoches and apply data transformation to each epoch. We set batch size to be 16.

Then, we use freeze the previous and medium layers of all architectures. We train the output layers of ResNet, and we and train the fully connected layers and add additional output layer to VGG. All the last layer of our model has the size of number if classes in our dataset. The approaches are detailedly illustrated in previous section.

In the optimization part, we use cross entropy to be our loss function. And we use Adam to be optimizer with learning rate 0.001.

In the model evaluation part, we compute the test accuracy and obtain the cost through each mini-batch and each epoch.

5.3. FaceNet

5.3.1 Image Augmentation

In FaceNet, we use Labeled Faces in the Wild(LFW). Still, we first apply image augmentation to our raw dataset, include:

- Random Horizontal Flip: Horizontally flip the given PIL Image randomly with a given probability 0.5.
- Random Rotation: Rotate the image by angle 90 degrees or 180 degrees.
- Random Vertical Flip: Vertically flip the given PIL Image randomly with a given probability 0.5.
- Normalize: Normalize a tensor image with mean 0.5 and standard deviation 0.5 in each of RGB channel.

5.3.2 Experimental setup

To begin our experiment, we split out dataset into train dataset(75%) and test dataset(25%). There are around 10000 images in train dataset and 3000 in test dataset. We set 20 epoches and apply data transformation to each epoch. We set batch size to be 64.

Then, we use deep convolutional architecture to get embedding feature vectors. We use two kinds of deep architectures, VGG and ResNet. For VGG and ResNet architecture, we use VGG16 and VGG19, ResNet34 and ResNet50, we use their feature extraction layers(the convolutional and maxpooling layer), and replace their average pooling layer and fully connected layer with our own embedding layer and fully connected layer.The embedding size(length of feature vector) we set is 128. In a word, we add two additional layers to our feature extraction layers, the embedding layer with size 128, and output layer with size of number of classes. When we got vectors from the output of embedding layer, we apply a L_2 normalization as we illustrate in previous section, and we also multiply the vector by 10 as suggested in [9].

In the optimization part, we set number of triplet we use to optimize to be 3000 in a mini-batch, both in training dataset and test dataset. we set the margin α to be 0.5. And we use Adam to be optimizer with learning rate 0.1, which will contribute to our model performance.

In the model evaluation part, the main method we use is Triplet loss, predict accuracy(whether the two images are from the same person) and the Receiver-operating-characteristics(ROC) and the area under ROC curve(AUC-ROC). In last case, the model perform as well as random guess has a AUC-ROC of 50%. We also calculate the triplet loss in each epoch, and in each mini-batch of each epoch.

5.4. Neural Style Transfer

In neural style transfer part, we set the generated image as the input, train the generated image instead of the parameters in the VGG19 network. The objective function is the total cost function:

$$Cost(G) = \alpha Cost_{\text{content}}(C, G) + \beta Cost_{\text{style}}(S, G)$$

We choose layer conv4_2 (see figure 4) to compute content image cost, and choose five layers (conv1_1, conv2_1, conv3_1, conv4_1 and conv5_1, of figure 4) from the network to calculate the weighted style cost. Here we set:

$$\lambda^{[l]} = 0.2, \forall\, l$$
$$\alpha = 10$$
$$\beta = 40$$

We train the model for 200 epochs for every combination of content image and style image to obtain the generated portrait paintings. Some hints of this parameter setting was from [8].

5.5. Software

Python 3.7
Jupyter notebook
Google Colab (GPU: Tesla K80)

5.6. Hardware

One Macbook Pro (2.3 GHz Intel Core i5/8GB RAM)
Two Surface Laptop (7th Gen i5/8GB RAM/256GB SSD)

6. Results and Discussion

6.1. Transfer learning

The test accuracy we get is not ideal. The test accuracy is shown like:

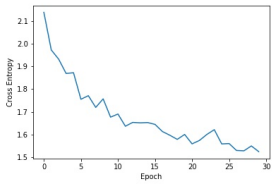
Model	Test accuracy
VGG16	38.12%
VGG19	34.26%
ResNet101	35.88%
ResNet50	40.00%

Table 1: Accuracy of different architecture for the transfer learning

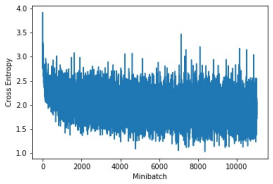
As shown in table 1. The ResNet50 got the best result while the ResNet101 fail to predict well. We dug into the cost to see the reason.

As shown in 7, the cost did not seems to converge through epoch, it takes more than 10 hours to train, the computational power seems not support. The reason might also be the pre-trained model is not suitable for our dataset. If we want better results, we need to train model from scratch, which seems not possible.

Since this dataset is collected by ourselves, it is not surprising that we cannot get very good results. Anyway, such results can at least give us some sense of the performances of different CNN architectures.



(a) ResNet101 cross entropy through epoch



(b) ResNet101 cross entropy through mini-batch

Figure 7: ResNet101 cross entropy

6.2. FaceNet

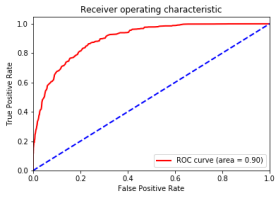
From the steps we proposed in previous section, we can get our triplet loss, accuracy and AUC-ROC of different architecture.

Architecture	Train Accuracy	Test accuracy	AUC
Resnet34	86.10%	81.47%	0.90
Resnet50	82.40%	79.87%	0.88
VGG 16	88.77%	80.22%	0.88
VGG 19	88.00%	80.03%	0.88

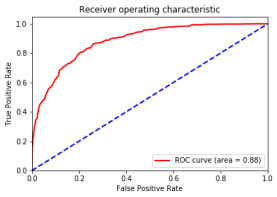
Table 2: Accuracy of different architecture, all the architecture followed by the embedding layer

As we can see in the table 2, ResNet34 got the best result of four architectures. All the architecture's train accuracy is higher test accuracy. ResNet34 got higher train and test accuracy than ResNet50, illustrate that more layers not always contribute to better results. VGG16 and VGG19 got higher train accuracy than ResNet, but their test accuracy is not highest, it seems there exist overfitting in VGG architecture.

Then we dig into the ROC curve for to better analyse the accuracy.



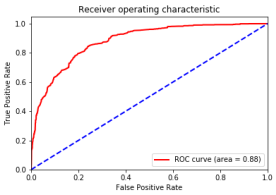
(a) ResNet34 ROC curve



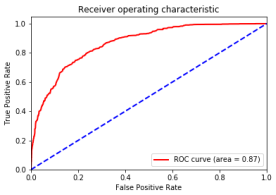
(b) ResNet50 ROC curve

Figure 8: ROC curve for ResNet

As we can see in figure 8 and figure 9, both architecture shows a good result. Still, the 90% AUC-ROC of ResNet34 beats all other architectures, in some threshold, ResNet34 can get a result of 0.2 false positive rate and 0.8 true positive rate.



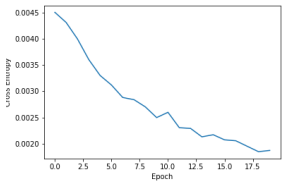
(a) VGG16 ROC curve



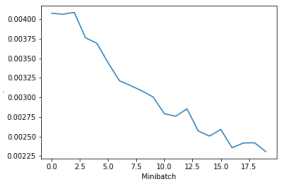
(b) VGG19 ROC curve

Figure 9: ROC curve for VGG

The triplet cost through epoch may give some ideas about overfitting and convergence.

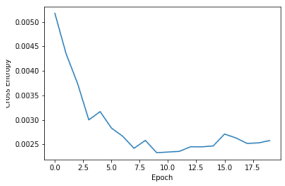


(a) ResNet34 triplet loss through epoch

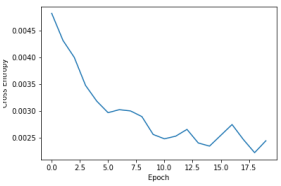


(b) ResNet50 triplet loss through epoch

Figure 10: ResNet triplet loss through epoch



(a) VGG16 triplet loss through epoch



(b) VGG19 triplet loss through epoch

Figure 11: VGG triplet loss through epoch

As we can see in figure 10 and figure 11, VGG architecture is not converged through the training, which may caused the overfitting in the performance. The ResNet34 is more stable at the end while the ResNet50 still have some fluctuations. Thus, all the result shows the ResNet34 has highest performance among all architecture, while the triplet cost function contribute to convergence and accuracy.

6.2.1 Shortcoming

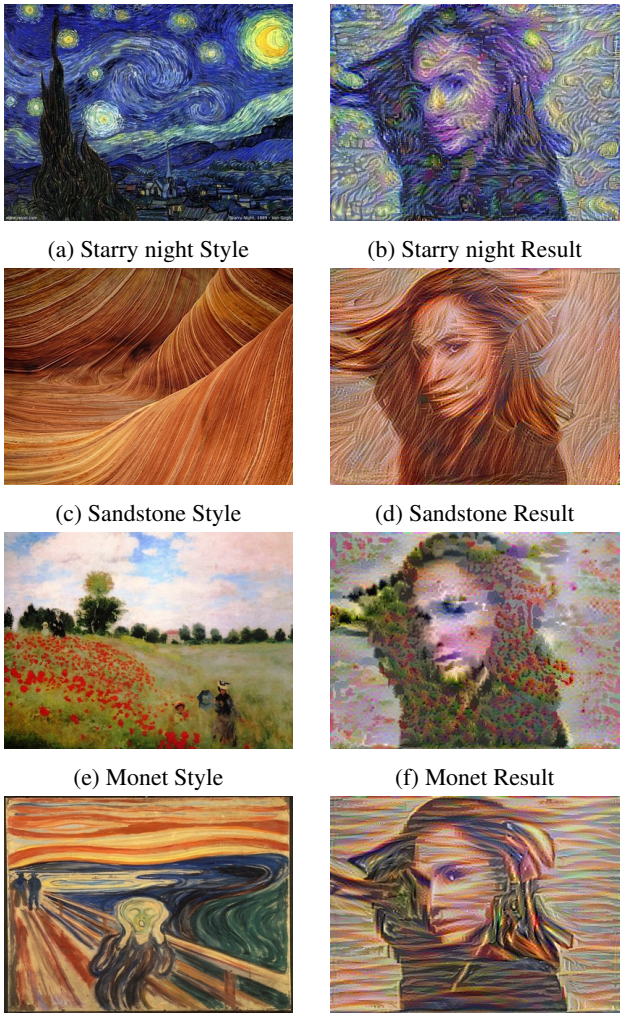
The model we built is based on ResNet architectures and VGG architectures, there is still some other architectures we could try to improve our performance. And the number of epoch we set is 20. We may get higher accuracy if we train

with more epochs. These two shortcomings comes from the limitation of time and computational power, we'll try more in our future exploration.

As for the methodology, how to select triplet plays crucial role in our model. We use the technique proposed in paper. We'll try more triplet selection methods that suitable for our dataset in further exploration.

6.3. Neural Style Transfer

From the steps we proposed in previous section, we can obtain some generated paintings having different styles. Some of them are showed in figure 12.



(a) Starry night Style (b) Starry night Result

(c) Sandstone Style (d) Sandstone Result

(e) Monet Style (f) Monet Result

(g) Skrik Style (h) Skrik Result

Figure 12: Neural Style Transfer results

6.3.1 Shortcoming

Some of the generated painting looks really good, but some paintings, like the Skrik style based result, can be improved. From previous work, we can see that the VGG19 architecture is not always the best one for some task. We can try to use other architectures to improve the results. In addition, we can also try to perform parameter tuning to find better weight parameters to obtain better paintings. Finally, running more epochs may also be helpful to improve the performance of our model.

7. Conclusions

Face recognition and verification play an important role in deep learning. In this project, we implement transfer learning, FaceNet, and Neural Style Transfer to conduct the face image analysis, which is a popular field in modern world.

We first implement transfer learning with deep convolutional architecture of VGG and ResNet on dataset we scraped on google image search. The result is not very ideal since the images we scraped contain too much noise. The best test accuracy we get is only 40% on ResNet50. In further exploration, we will dig more into data pre-processing part to get more pure face image to do transfer learning.

The next part of our project is about face verification using the FaceNet system with triplet loss. The dataset we use is Labeled Face in the Wild (LFW)[2], which is a face database designed for face recognition. The basic technique we use is based on the FaceNet paper [10]. As for the triplet loss part, we select all the anchor-positive pairs in a mini-batch and the semi-hard negative exemplars. The deep architecture we use are ResNet34, ResNet50, VGG16 and VGG 19. The ResNet34 achieve the best result with 81.47% test accuracy with AUC of 0.90. The train accuracy and test accuray of ResNet50 are both lower than that of ResNet34. This may be because the more complicated architectures do not always mean the better results, the ResNet34 might be more suitable for LFW combined with triplet loss. The VGG16 and VGG19 architecture shows over-fitting problem, and only achieve 80.22% and 80.03% accuracy. In future work, we will explore the more suitable convolutional architecture and more effective triplet selection procedure to improve our result. We will also try more training epoch if we have more time and computational power.

For the last part of our project, we introduce a neural style transfer method to generate the portraits in different painting styles. This area has been explored a lot recent years [4][5][3]. We choose the pre-trained VGG19 architecture to implement such method and the generated portraits are at least qualitatively comparable to common portrait paintings. Our face-to-painting machine is easy to interpret, easy to train and it provides some insights into potential applications of deep learning methods. To improve our results, people can try to perform hyper-parameter tuning, run more epochs and try more CNN architectures.

Based on the results listed above, we can say that the

motivation is accomplished:given a photo of human face, the machine can perform face verification to tell who he/she is, and the largest AUC is 0.90, which means the accuracy is acceptable; then, the machine can produce portrait painting based on this photo and a given style (such as Monet's style). The main idea of face verification part is from [10] and the main idea of neural style transfer is from [4]. The basic settings of parameters for neural style transfer are similar as [8]. Our project can be related to some more works in [5] and [7]. [5] extends the existing method to introduce control over spatial location and [7] introduces a deep-learning approach to photographic style transfer that handles a large variety of image content while faithfully transferring the reference style. Those methods may be helpful in improving the results of our project.

Our project can be used in many real world applications, such as AI painter, painting repair, souvenir generation and so on.

8. Acknowledgements

We are glad to acknowledge Prof. Sebastian Raschka, without whose instruction we will not be able to establish and finish this project, let alone entering this interesting world of deep learning. Last but not least, we would like to acknowledge Overleaf and Github, they make cooperation more easier than ever.

9. Contributions

Zhuoyan Xu implements the FaceNet system for the face verification. He also scrapes the data on Google image search for the transfer learning. He writes the experiment, result and conclusion part of the project.

Lingfeng Zhu implements the neural style transfer and transfer learning, he writes the abstract part, result and conclusion part and related works part and comes up with the proposed method part in our project report.

Cercily Liu helped with the data collection and data preprocessing part. She implements the data augmentation and transfer learning. She also write the introduction and motivation part for the project report.

References

[1] `https://towardsdatascience.com/transfer-learning-946518f95666/`.

[2] Lfw dataset. `http://vis-www.cs.umass.edu/lfw/`.

[3] D. Chen, L. Yuan, J. Liao, N. Yu, and G. Hua. Stylebank: An explicit representation for neural image style transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1897–1906, 2017.

[4] L. A. Gatys, A. S. Ecker, and M. Bethge. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576*, 2015.

[5] L. A. Gatys, A. S. Ecker, M. Bethge, A. Hertzmann, and E. Shechtman. Controlling perceptual factors in neural style transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3985–3993, 2017.

[6] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[7] F. Luan, S. Paris, E. Shechtman, and K. Bala. Deep photo style transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4990–4998, 2017.

[8] A. Ng. Deeplearning.ai. `https://www.deeplearning.ai/`.

[9] R. Ranjan, C. D. Castillo, and R. Chellappa. L2-constrained softmax loss for discriminative face verification. *arXiv preprint arXiv:1703.09507*, 2017.

[10] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.

[11] A. Selim, M. Elgharib, and L. Doyle. Painting style transfer for head portraits using convolutional neural networks. *ACM Transactions on Graphics (ToG)*, 35(4):129, 2016.

[12] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[13] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[14] Y. Sun, X. Wang, and X. Tang. Deeply learned face representations are sparse, selective, and robust. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2892–2900, 2015.

[15] Y. Taigman and M. Yang. Marcaurelio ranzato, and lior wolf. deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1701–1708, 2014.

[16] K. Q. Weinberger, J. Blitzer, and L. K. Saul. Distance metric learning for large margin nearest neighbor classification. In *Advances in neural information processing systems*, pages 1473–1480, 2006.