DeepRecognition: An Attractive Approach for Deep Fake Synthesis

Emily Brunelli gallage1@uci.edu
Andreana Chua chuaa2@uci.edu
Adi Faintuch afaintuc@uci.edu
Brooke Ryan brooke.ryan@uci.edu

Abstract

In this project, we aim to understand state-of-the-art generative adversarial networks (GANs) and implement a neural network transfer model to classify and feed facial attributes into a GAN for Deep Fake image synthesis. Since their inception in 2014, several variations of the original GAN approach have been fine-tuned to obtain higher accuracy and synthesis of Deep Fakes based on specific facial attributes, such as StarGAN and AttGAN. These approaches tend to focus on objective, binary facial attributes, such as whether the individual is smiling, if they are wearing makeup, eyeglasses, etc., but these approaches have not explored subjective traits such as the individuals' perceived attractiveness. In this project, we seek to train a network on the popular CelebA dataset to recognize "attractive" faces, feed these images to our GAN, and explore the resultant synthesized images. One of the computational challenges to overcome in this project is the significant hardware demands that GANs demand. The objective of this project is to explore state-of-the-art methodologies in neural networks and deep learning in order to foster a holistic understanding of the subject.

1 Introduction

Since the technology was introduced in 2014 by Goodfellow et al. [1], generative adversarial networks (GANs) have become of particular interest, both to the machine learning research community and the greater public at large through the genesis of Deep Fakes. Deep Fakes are artificially generated images, often portrayed in the media through its malicious applications such as celebrity impersonation, disinformation agents, or pornography generation. In less pernicious applications, Deep Fakes are still associated with the dystopian capabilities of artificial intelligence, such as the popular mobile application FaceApp, which is used to artificially augment the users' appearance.

Permutations of the original GAN approach, such as StarGAN [2] and AttGAN [3], have been introduced with the goal of obtaining higher accuracy and synthesis of Deep Fakes based on specific facial attributes. The GAN literature has focused thus far on objective, binary facial attributes, such as whether the individual is smiling, if they are wearing makeup, eyeglasses, etc. We seek instead to explore *subjective* traits, namely an individual's perceived "attractiveness", by training a network on the popular CelebA dataset, feed images categorized as "attractive" to a GAN, and analyze the resultant synthesized images. One of the challenges to overcome in this project is the significant computational demands of GANs. While this project exposed us to state-of-the-art GAN literature, we lacked the physical hardware capabilities to produce images that even moderately resembled human faces, let alone attractive ones. Ultimately, this project provided a still-valuable learning exercise in deep learning libraries, technologies, and tools that are able to be run in the scope of a standard home computer.

UC Irvine Neural Networks and Deep Learning (CS 274C), Winter Quarter 2021

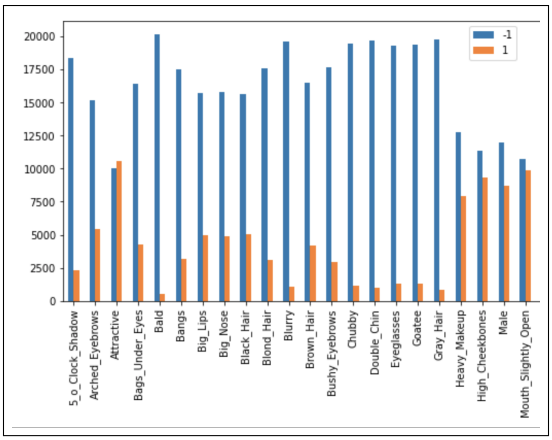

Figure 1: Distribution of attributes over the CelebA dataset.

2 Data

Our project used the CelebA dataset, introduced by Liu et al. in 2015 [4]. This dataset is well-known and widely used in machine learning, particularly in the space of problems related to GANs and facial attribute generation and recognition. The dataset contains 200,000+ images of celebrities in various lighting conditions and expressions. There are 40 attributes provided with each image, some binary objective attributes such as "big nose", "eyeglasses", "smiling", while others are subjective such as "attractive."

Our project seeks to explore and implement Deep Fake images based on the quality of "attractiveness" of the individual portrayed in the photograph. Specifically, this is represented in the "attractive" attribute in the dataset. Due to the plentiful attribute descriptions that accompany each image in this dataset, as well as the wide usage of CelebA in GAN/ Deep Fake benchmark comparisons, this dataset lent itself well to the problem exploration at hand. Furthermore, the dataset did not require the application of any data augmentation techniques.

2.1 Processing

The original dimensions of the images, 218×178 pixels, must be converted to $218 \times 178 \times 3$, representing (height, width, color channel) respectively, then further compressed to $128 \times 128 \times 3$. Min-max normalization is then performed on each image x, in which $x = \frac{x-0}{255}$, again representing the range of pixel values.

After, we partition the data into a 60/40/40 split, for training, validation, and test sets respectively.

As can be seen in Figure 1, the majority of the attributes in the dataset have an uneven distribution. While the "attractive" attribute has approximately equal representation of 1 and -1 labels, when performing experiments on other attributes, to avoid dealing with imbalanced data, balanced attributes were chosen.

3 Architecture Design and Training

3.1 Convolutional Neural Network (CNN)

To establish a baseline, we implement a convolutional neural network (CNN) from scratch to compare its performance to a transfer learning model. The hyperparameters of the model are based on those in literature presented by Simonyan and Zisserman [5]. The input layer accepts an input size of $128 \times 128 \times 3$. The first layer has 32 filters which doubles in size as a new convolutional layer is added. The filter size is 3×3, followed by one fully connected layer and then the output layer, with 1 node. Each convolutional layer has a max-pooling

layer after it, in which the filter size is 2×2, with stride 2. Furthermore, we use a padding value of "SAME" so the input size and the output size stay the same. Each layer also uses ReLU as its activation function, except for the output layer, which uses sigmoid. For the loss, we use the Binary Cross-entropy loss which is congruent with the goal of binary classification. We also use Adam Optimizer for training, since it works efficiently with the model.

The one hyperparameter we trained was the number of layers our model would have. Here, we experimented with 1-4 layers, using our validation set. After each experiment, we obtained the following results:

Num Layers	Validation Test Accuracy
1	0.7443
2	0.7580
3	0.8460
4	0.8142

From these results, we built the final model with 3 layers, since it obtains the highest accuracy on the validation test. From there, we train and test on the final model and achieve an accuracy of 0.8481.

Based on this model, we created a CNN model to do multilabel classification. The main changes we did were to convert the output layer's activation function from sigmoidal to softmax and the number of nodes from 1 to 4. We chose 4 since there were 4 attributes (Heavy_Makeup, High_Cheekbones, Male, and Mouth_Slightly_Open) that we found that had an even split between -1 and 1 labels. After training and testing, we achieved an accuracy of 0.5909.

3.2 Transfer Learning

We used ResNet50 as our pre-trained model that served as the base for our single-label classifier and a multi-label classifier that took 4 attributes. ResNet50 is a 50-layer deep convolutional neural network that was trained to classify objects into more than 1000 different categories (none of which included faces, which was the focus of our project) [6]. With transfer learning, several of the things we might hope to see are 1) better starting accuracy, 2) faster training and 3) better asymptotic accuracy. Our basic approach was to use the bottom 45 layers of ResNet50, freeze the weights, and then train on an additional 4 layers (which included 3 layers of RELU, and a final layer that was either sigmoidal or softmax). In between each of our 4 custom layers, we also made use of batch normalization and set the dropout rate to 0.5. We ran it for 12 epochs. Our improved accuracy was 0.8759 for the single-label classifier (our non-transfer learning accuracy was 0.8481), and of 0.634 for multilabel accuracy (compared to a non-transfer learning accuracy of 0.591). As expected, our training and asymptotic accuracy were better with transfer learning, as seen in the graphs below.

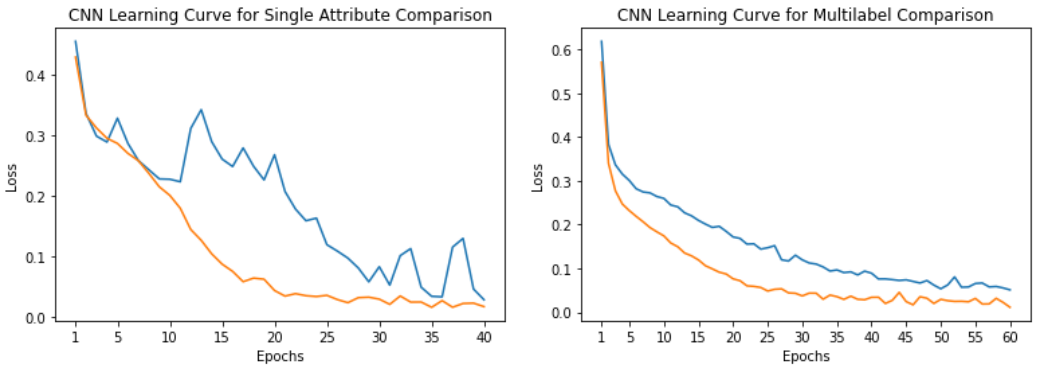

Figure 2: Comparison of CNN learning curve for single vs. multi attribute classification. Orange represents our transfer learning model while blue represents our CNN scratch model.

4 Generative Adversarial Network (GAN)

4.1 Experiments

Given the ample set of images provided from the network as outlined in the sections above, we have a training set of images deemed to be "attractive" that can be fed to the GAN. The construction of the GAN requires two components–a generator and a discriminator. The experiments were performed on Jupyter notebooks hosted on a local machine, the source code can be viewed at github.com/brookekelseyryan/DeepRecognition. The subsequent paragraphs provide a walk-through of the construction of the experiments, as well as an analysis of the parameters chosen.

We began by constructing the generator. In the `create_generator()` function, we first create an `Input` using TensorFlow. Then we used TensorFlow's `Model` class to create a neural network with 16 layers, of types `Dense`, `Conv2D`, `Conv2DTranspose`, `LeakyReLU`, and `Reshape`. For our last layer (`Conv2D`), we used the activation function `tanh`. Our `create_generator()` function returns this model.

We then constructed the discriminator. In the `create_discriminator()` function, we also created an `Input` using TensorFlow. Then we used TensorFlow's `Model` class to, again, create a neural network with 13 layers, of types `Dense`, `Conv2D`, `LeakyReLU`, `Flatten`, and `Dropout`. For our last layer (`Dense`), we used the activation function `sigmoid`. We used `RMSprop` for our optimizer, and binary crossentropy for our loss. Our `create_discriminator()` function returns this model.

The following lines of code put the pieces of the generator and discriminator together to actually make the generative adversarial model:

```
gan_input = Input(shape=(LATENT_DIM, ))
gan_output = discriminator(generator(gan_input))
gan = Model(gan_input, gan_output)
```

The main loop houses the code that synthesizes the Deep Fake images. We used a for-loop with 1000 iterations. In the loop, we first created a `latent_vectors` in the size `(batch_size, LATENT_DIM)`, where `batch_size` was equal to 16 and `LATENT_DIM` was equal to 32. We generated images by calling predict on our generator with the `latent_vectors` as the input. We then also slices out real images from our dataset in the same size as our generated images, and stored it in a variable called real. We then trained the discriminator on the combined real and generated images. Next, we called `train_on_batch` on the GAN with inputs being `latent_vectors` and a numpy array of shape `(batch_size, 1)` to store the misleading targets. Finally, if it was iteration 50, we would take our images and combine them into a mosaic of 36 fake faces. After iteration 50, this process would start over and a new set of faces would be created and improved over 50 iterations.

Figure 3: Images generated after 1000 iterations.

After we were able to get the GAN to produce Deep Fake faces, we fine-tuned our code to generate, specifically, "attractive" faces with the GAN. To do this, instead of using any real images, we only used real images with the "attractive" attribute in the CelebA dataset.

4.2 Results

Using our Adversarial Generative Network, we were able to produce DeepFake faces that appear to be human. Of course, given that our GAN was so simple, unlike those from cutting edge research, the faces generated do not look the most realistic. While they do have human

attributes, such as eyes, mouths, noses, hair, and all in relatively normal human proportions, they look more like Picasso-esque human than an actual human.

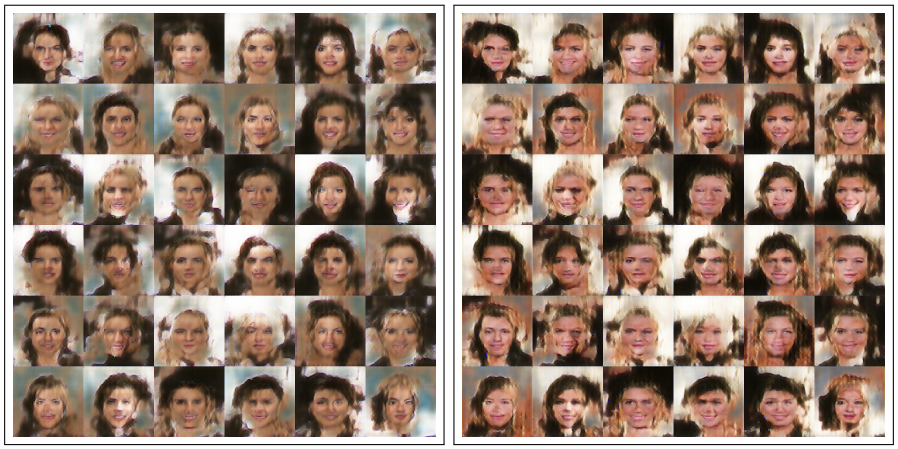

Figure 4: Two mosaics of deep fake faces generated.

As can be seen in Figure 4, we were able to generate fairly accurate human faces with our basic GAN. The Deep Fakes which were generated solely from the "attractive" images can be seen in Figure 3.

5 Conclusions and Broader Impact

In terms of the first component of our project, we found that transfer learning was much more effective when training on the image dataset, which ultimately resulted in higher accuracies for the "attractivenes" classification. The pretrained model was able to capture more patterns in the images than the model built from scratch, even when using hyperparameters from the literature on standard tasks.

For our second component, particularly given that we were using standard hardware available within a home computer, our two different GANs were able to produce Deep Fakes that did clearly represent human faces. However, because of hardware limitations that did not allow a significant amount of iterations to be run, the results were not as realistic as they could be, and additionally we could not glean much information on our hypothesis about generating interesting Deep Fakes from a pool of "attractive" images. As Deep Fakes become increasingly realistic at a sometimes alarming rate, we conclude that not only is the research behind detecting and creating Deep Fakes a figurative arms race, but it is also a race to procure the greatest amount of computing power. Ultimately, while our project lacked the capabilities to run state-of-the-art algorithms, it provided a vehicle to study the literature and theory of such models, and reaffirmed that learning the theoretical mathematical structure of a deep learning architecture is paramount, particularly when such models consume such significant computational resources that it becomes a burden to validate scientific hypotheses.

Appendix

The code for our project is hosted on github.com/brookekelseyryan/DeepRecognition.

Team Contributions

As per assignment requirements, each author has delineated their list of contributions to the project below under their corresponding paragraph. As this was a collaborative assignment, some of the contributions were worked on simultaneously by multiple group members, and some overlap is to be expected.

Emily Brunelli

- Organized and participated in regular group meetings.
- Practiced and presented presentation slides on transfer learning and model training to the class.
- Wrote transfer learning code
- Worked on transfer learning

Andreana Chua

- Organized and participated in regular group meetings.
- Practiced and presented presentation slides on convolutional neural networks and model training to the class.
- Wrote CNN scratch model code
- Worked on Data and CNN sections of report

Adi Faintuch

- Organized and participated in regular group meetings.
- Practiced and presented presentation slides on generative adversarial networks and experiment results to the class.
- Wrote the GAN code and generated mosaics of faces
- Wrote final report sections on Generative Adversarial Networks (4.1, Experiments, and 4.2 Results)

Brooke Ryan

- Researched both classic and state-of-the-art implementations of generative adversarial networks.
- Organized and participated in regular group meetings.
- Wrote final report sections on Abstract, Introduction, Dataset; edited all sections.
- Practiced and presented presentation slides on introduction and conclusion to the class.

References

[1] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.

[2] Y. Choi, M. Choi, M. Kim, J. Ha, S. Kim, and J. Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8789–8797, 2018.

[3] Z. He, W. Zuo, M. Kan, S. Shan, and X. Chen. Attgan: Facial attribute editing by only changing what you want. *IEEE Transactions on Image Processing*, 28(11):5464–5478, 2019.

[4] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[5] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.