Exploring the Mechanisms of Feature Extraction

Niccolò Meniconi, Gokul Ramasamy, Rahavee Prabakaran

Abstract

In this project we explore the mechanisms of feature extraction of a commonly used convolutional network architecture, VGG, by training two identical networks on identical data labeled for different image attributes. By controlling these variables, we hope to uncover the ways in which a convolutional network interprets the same image when we train it on different labels, helping us better understand how the training algorithm optimizes a network for a specific task. Answers to this question can potentially uncover the mechanisms by which neural networks optimize themselves in training, and can lead to development of optimization techniques in transfer learning and model finetuning.

1. Introduction

When reading popular science articles or listening to technology-related news, one might come across "machine learning" and its contribution to solving a problem in data science or research. It's easy to understate the importance of machine learning (ML) and its relevance to modern technology: the iPhone face unlock program, automated driving, Alexa, Instagram, Spotify, and most advertising now rely on artificial intelligence in one way or another.

Machine learning (ML) is the study of computer algorithms that can improve automatically through experience and by the use of data [1]. ML algorithms are used in a wide variety of applications, such as in medicine, email filtering, speech recognition, and computer vision, where it is difficult or unfeasible to develop conventional algorithms to perform the needed tasks [2]. ML has become an acceleratingly powerful and increasingly accessible tool, two factors that have contributed to its success and widespread use.

By taking many training inputs, usually in the order of hundreds of thousands, researchers have found ways to develop algorithms first constructed in the 40s [3] that mimic the behavior of neurons and are able to recognize meaningful patterns in images for the purpose of data classification. These structures are known as neural networks (NNs). Training a NN involves using an optimization algorithm, known as back propagation, to minimize the error between a training input and its output, known as the loss function. With the advent of GPUs and the development of High-Performance Computing techniques, NNs have become prominent in the field of data science.

Even though NNs are becoming more and more popular, data scientists still regard neural networks as "black box" systems, where the mechanisms through which it learns to derive a solution are unknown. The mathematics and the algorithms that regulate neural network training

developed in the mid 80s [4] are well known and have been optimized effectively, but the nature of the training algorithm makes it hard to predict how a model will interpret data during training. This is because these algorithms involve the application of randomness on highly complex structures: the same training setup can yield two non-identical networks with non-identical performances. Understanding the paths a neural network takes while training is not necessary to train models and to deploy them in the real world. But a better understanding of the training process could not only lead to a better understanding of neural networks as a class of algorithms, but could lead to progress in the field and the development of new technologies.

Our project aims at answering how a commonly used network, VGG [5], changes via the training algorithm. We do so by training two identical VGG networks on the same input images that are labeled for different attributes in the two training instances. The dataset we use is CelebA, a large-scale face attributes dataset with more than 200K celebrity images, each with 40 attribute annotations [6]. The two VGG networks are trained to detect "Smile" vs "No Smile", and "Male" vs "Not Male" respectively. We trained the two VGG networks using PyTorch, using the same optimizer, number of epochs and training parameters.

When training the two networks we save the weights at each epoch, allowing us to perform experiments on the different weights and observe how the hidden layers change throughout the training history. More detail on our experimental setup and outcomes are found in the following sections.

2. Experimental Setup:

a. Dataset:

To compare the performance of a neural network on the same dataset labelled differently, it was essential for the dataset to have multiple labels. One such dataset is the Celeb-A dataset with 40 binary attributes. The dataset contains 162,770 training images and 19,867 testing images. Figure 2.1 represents the presence of each of the 40 attributes across the dataset images:

For our experiment, the "Male" and the "Smile" classes were chosen. Other attributes were dropped; the models were trained on "Male" vs "No Male" and "Smile" vs "No Smile" classes.

b. Model Specifications:

The model that we chose to experiment with is the VGG-19 [5]. The feature extractor part of the network is a series of conv layers followed by maxpooling layers, and the classifier is a fully connected layer with a softmax layer at the end to output the probability of the class.

For our experiment, a pre-trained VGG-19 trained on the Imagenet [7] was fine-tuned to perform the “Male” vs “No Male” and “Smile” vs “No Smile” classification. The model was trained for ten epochs with Adam Optimizer of learning rate *1e-3* and with binary cross entropy as the loss function with a batch size of 16. The model was trained on a machine equipped with RTX 3070 and 12 Gigs of VRAM.

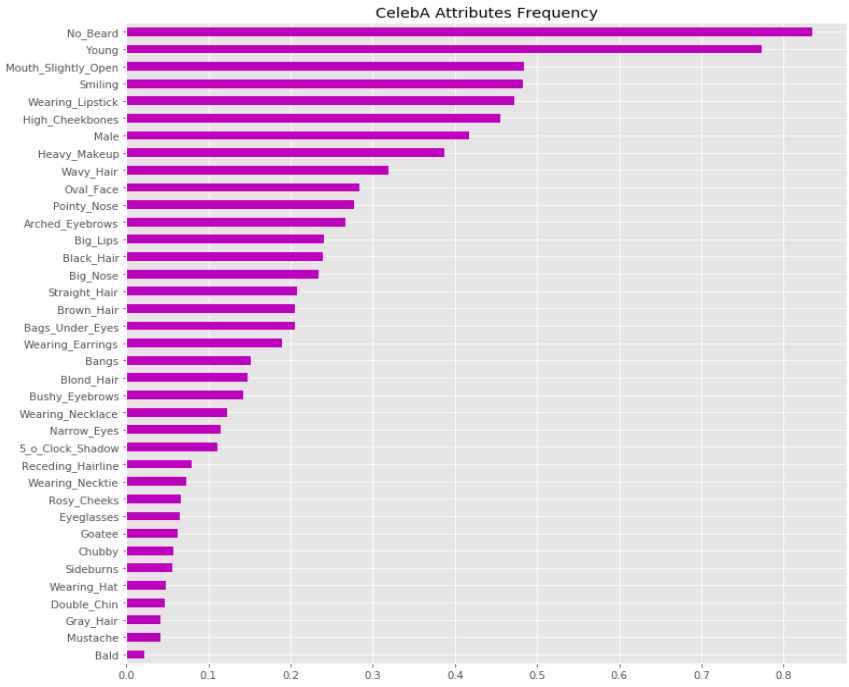


Fig 2.1: CelebA attribute distribution

3. Results:

To qualitatively assess the performance of the networks, the intermediate feature maps were visualised. The feature maps are visualised as heat maps in order to learn what features the model is looking at. Let the model trained for “Smile” vs “No Smile” be called Model L1 and let the model trained for “Male” vs “No Male” be called Model L2. The intermediate feature maps for Model L1 and Model L2 are presented in figures 3.1 and 3.2. From figures 3.1 and 3.2, it can be seen that Model L1 is looking for features around the mouth and Model L2 is looking for features around the eyes. So, it can be safely inferred that the model is looking for meaningful features and more or less looking for features that typically a human will look for.

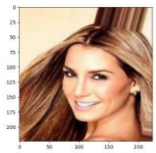


Fig 3.1: Intermediate Layer feature maps for L1

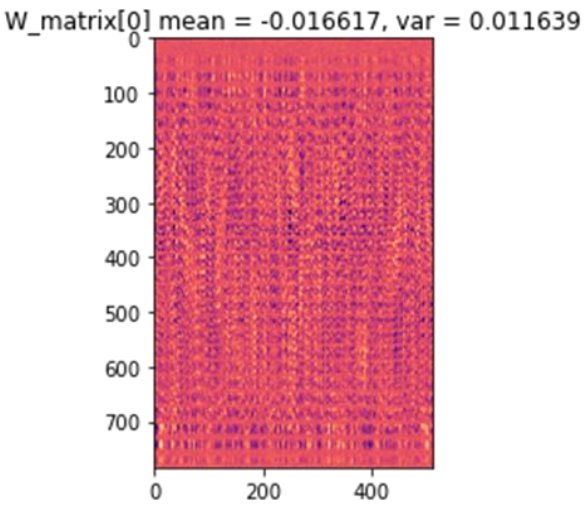
Fig 3.2: Intermediate Layer feature maps for L2

These results illustrate the differences in the way the two networks interpret the same image. The next step of the project is to observe their structural differences. Our approach is to extract the weights and bias matrices of single layers in a network structure from all the networks saved during the training process, and arrange them in an array. This will allow us to perform statistics on single weight and bias values as the model trains, and will help us determine how the two network structures are different.

We first demonstrate this approach on a simple 3-layer Fully Connected Neural Network (FCNN). We train the model on the MNIST Digit Dataset, a dataset containing 60,000 28x28 grayscale images of the 10 digits, along with a test set of 10,000 images. The network was trained on Keras for 5 epochs, with optimizer “adam”, and loss function “mean squared error”. Figures 3.3 and 3.4 illustrate the weight matrix connecting the input layer and the after training it for one epoch and five epochs respectively.

We can see how the absolute value of the mean and the variance of the overall map are increasing. If we look closely at the figures, we will also notice how dark spots (neighborhoods of negative values) and bright spots (neighborhoods of positive values) are arranged in horizontal and vertical streaks.

We then use these weight matrices, calculate the mean and the variance of single weight and bias values, and arrange them in a matrix of the same dimensions. Figures 3.5 and 3.6 illustrate the results of these calculations for the same layer



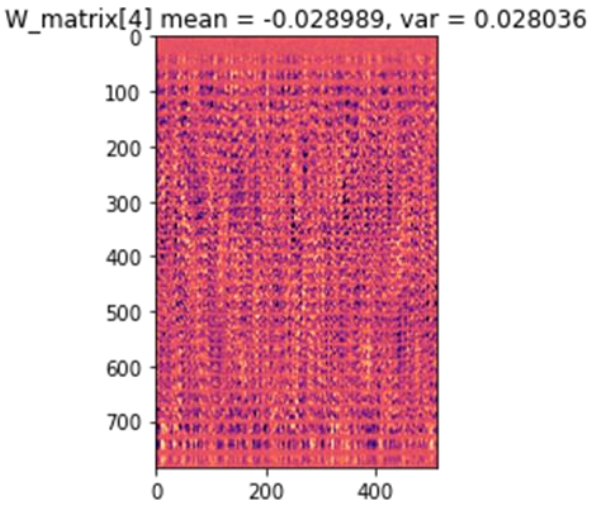
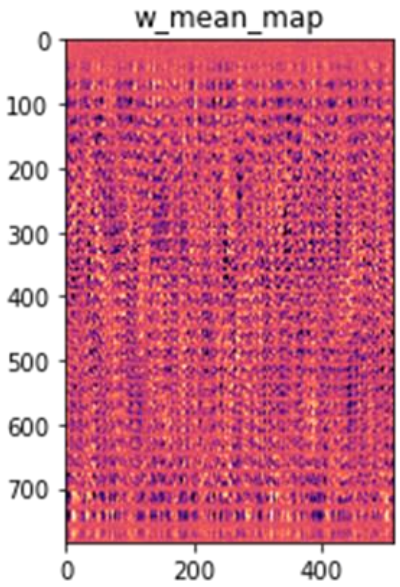


Fig 3.3: Weight matrix between first and second layer after first epoch *Fig 3.4: Weight matrix between first and second layer after fifth epoch*



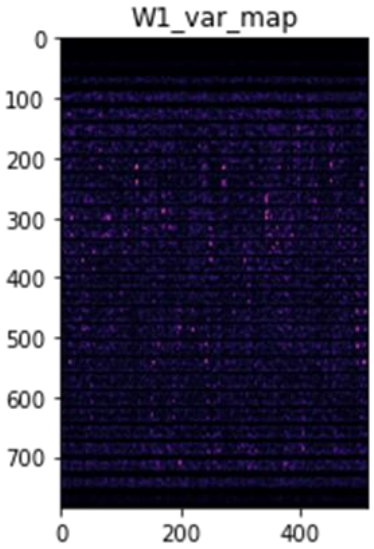
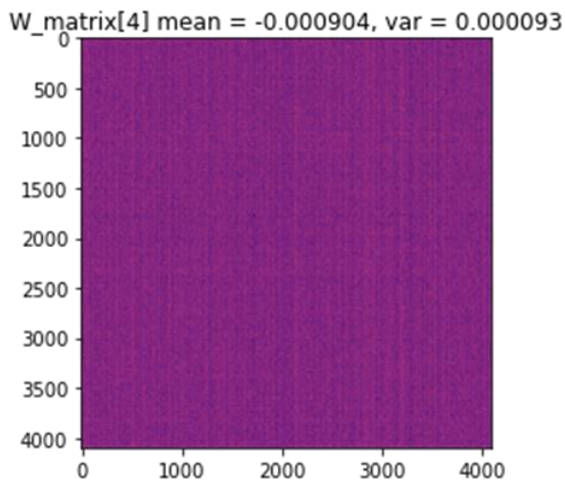


Fig 3.5: Mean value of the weights throughout training

Fig 3.6: Variance of the weights throughout training

We can observe that a similar pattern of horizontal and vertical streaks occurs in both plots. In Fig 3.6 we also notice that there are neighborhoods of neurons that change more compared to other neighborhoods throughout training, indicated by bright spots in the plot.

We attempt this approach with the two VGG networks on the 40th layer, a fully connected layer with 4096 neurons at the input and 4096 neurons at the output. Figures 3.7 and 3.8 illustrate the mean maps of the two networks, and Figures 3.9 and 3.10 illustrate the variance maps of the two networks:



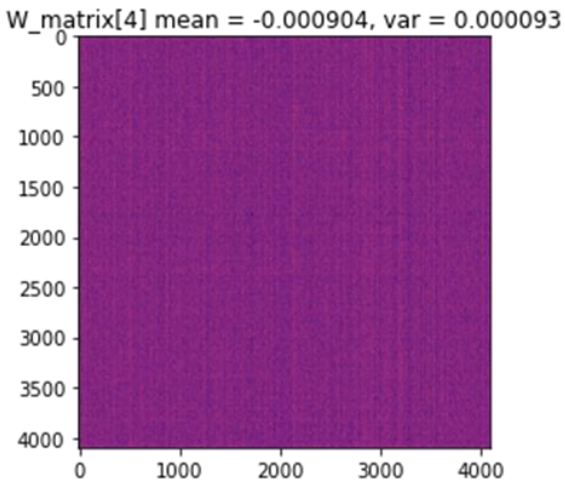
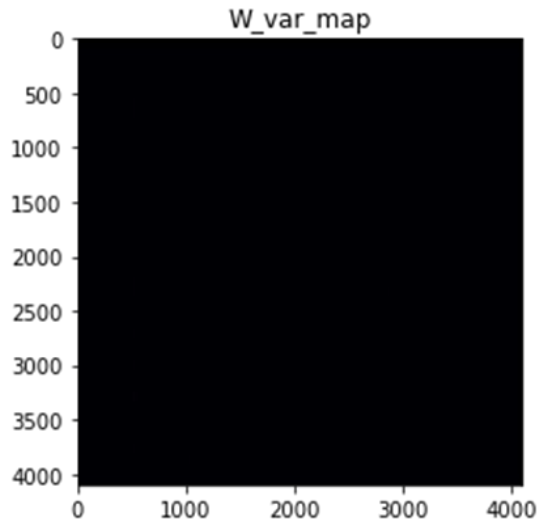


Fig 3.7: Mean values of the weights in "smile- no smile" network Fig 3.8: Mean values of the weights in "male" vs "not male" network



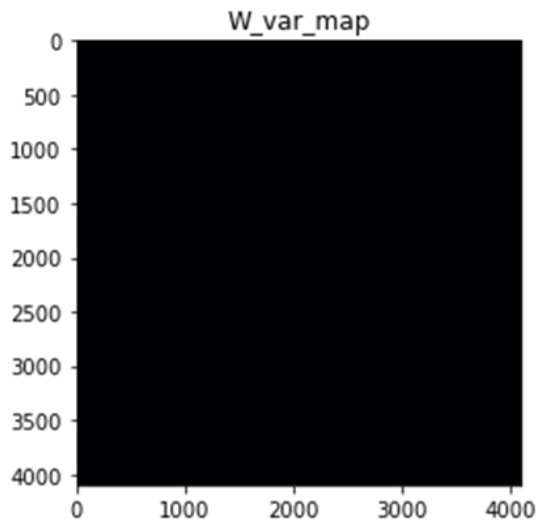


Fig 3.9: Variance values of the weights in "smile- no smile" network Fig 3.10: Variance values of the weights in "male" vs "not male" network

We can see that there are no significant differences in the structures in this layer between the two networks. This could be because mean or variance may not be appropriate metrics for quantifying the differences between two identical neural network structures. As a next step we could try the same approach for a longer training history to see if similar training patterns emerge. Mean and variance may also not be the best metrics to quantify the structural differences of the two networks.

4. Conclusion

These experiments all attempted to quantify the difference in neural network training between two identical CNN structures trained on the same dataset, CelebA, labelled differently. We

observe that the CNN network we used, VGG-19, was able to meaningfully process input images and interpret their contents in a similar way a human being would. We then proceeded to analyze how intermediate layers evolve through the training process in the two different networks. We used metrics such as mean and variance but, given the size of the network and the number of epochs we trained the network with, we did not arrive at any meaningful conclusions. We anticipate the differences in the two network structures can be more easily interpreted if we use smaller networks, run longer training sessions on the networks, or measure their progress using different statistical metrics.

References:

[1] Mitchell, Tom Michael. *Machine Learning*. New York: McGraw-Hill, 1997.

[2] Hu, Junyan, Hanlin Niu, Joaquin Carrasco, Barry Lennox, and Farshad Arvin. "Voronoi-Based Multi-Robot Autonomous Exploration in Unknown Environments via Deep Reinforcement Learning." *IEEE Transactions on Vehicular Technology* 69, no. 12 (2020): 14413–23.

[3] Palm, G. "Warren McCulloch and Walter Pitts: A Logical Calculus of the Ideas Immanent in Nervous Activity." *Brain Theory*, 1986, 229–30.

[4] Roberts, Eric. "Neural Networks - History: The 1980's to the present." https://cs.stanford.edu/people/eroberts/courses/soco/projects/neural-networks/History/history2.html.

[5] Simonyan, Karen, and Andrew Zisserman. "Very Deep Convolutional Networks for Large-Scale Image Recognition." *International Conference on Learning Representations*, 2015.

[6] Liu, Ziwei, Ping Luo, Xiaogang Wang, and Xiaoou Tang. "Deep Learning Face Attributes in the Wild." *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015.

[7] Deng, Jia, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. "ImageNet: A Large-Scale Hierarchical Image Database." *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009.