We Taught Our Computers to Identify Clothes

William Chandra, Spencer Ng, Anthony Nguyen

STAT 4354 Numerical and Statistical Computing

Dr. Shengjie Jiang

May 10th, 2024

1. Introduction

For this project, data was taken from a clothing dataset found on Kaggle. (link) The dataset features over 40,000 images totalling almost 8 gigabytes, and consists of images of both clothing and assorted items.

Our goal for the project was to train a variety of models that would be able to adequately classify these images into their respective categories of clothing, while also utilizing various techniques learned in class.

After training these models, we also hoped to test them on real-world examples — moving from traditional fast fashion to modern haute couture. We believe that these tasks, while seemingly mundane, offer important commentary on various domains, including the capabilities of artificial intelligence and the relative complexity of classification in modern fashion.

1. Data Cleaning

Our original dataset consisted of over 44,441 images with a 1800 x 2400 resolution. These images were sorted into 143 categories, with most of these categories being non-clothing items such as watches, makeup, or even basketballs, or being similar to another category, such as shoes and sport shoes. We decided to choose 23 categories and condense them into 11 categories, which cut the amount of images we had to process in half. However, this was still too many images we would be able to process.

From the original set of over 40,000 images, we selected 50 images from 11 categories that we believed best represented the variety of clothing types in the dataset. These categories, in no particular order, were: dresses, formalwear, outerwear, pants, shirts, t-shirts, shoes, shorts, skirts, socks, and underwear.

The decision to use 50 images was unfortunate, but ultimately deemed necessary due to sheer size constraints. Ultimately, we believe superior results may have been possible given a larger training dataset, and we suggest that future work could take this route.

Images were further demarcated into three separate categories: unchanged originals, grayscaled images with PCA run, and images grayscaled through singular value decomposition. The PCA methodology taken will be described in section 3.

Images that were grayscaled normally followed the traditional luminance method using the weights 0.21 R + 0.72 G + 0.07 B.

The images that were grayscaled using SVD followed the method proposed by Khudhair et. al in their paper “Color to Grayscale Image Conversion Based on Singular Value Decomposition.” A link and citation are provided below.

Briefly, the proposed method consists of loading a color image split into individual R, G, and B matrices. For each individual pixel in the image, a color vector of each R, G, and B value is created. Assorted weights are suggested based on research goals; ultimately, our team chose to add no separate weights to ensure balanced image composition.

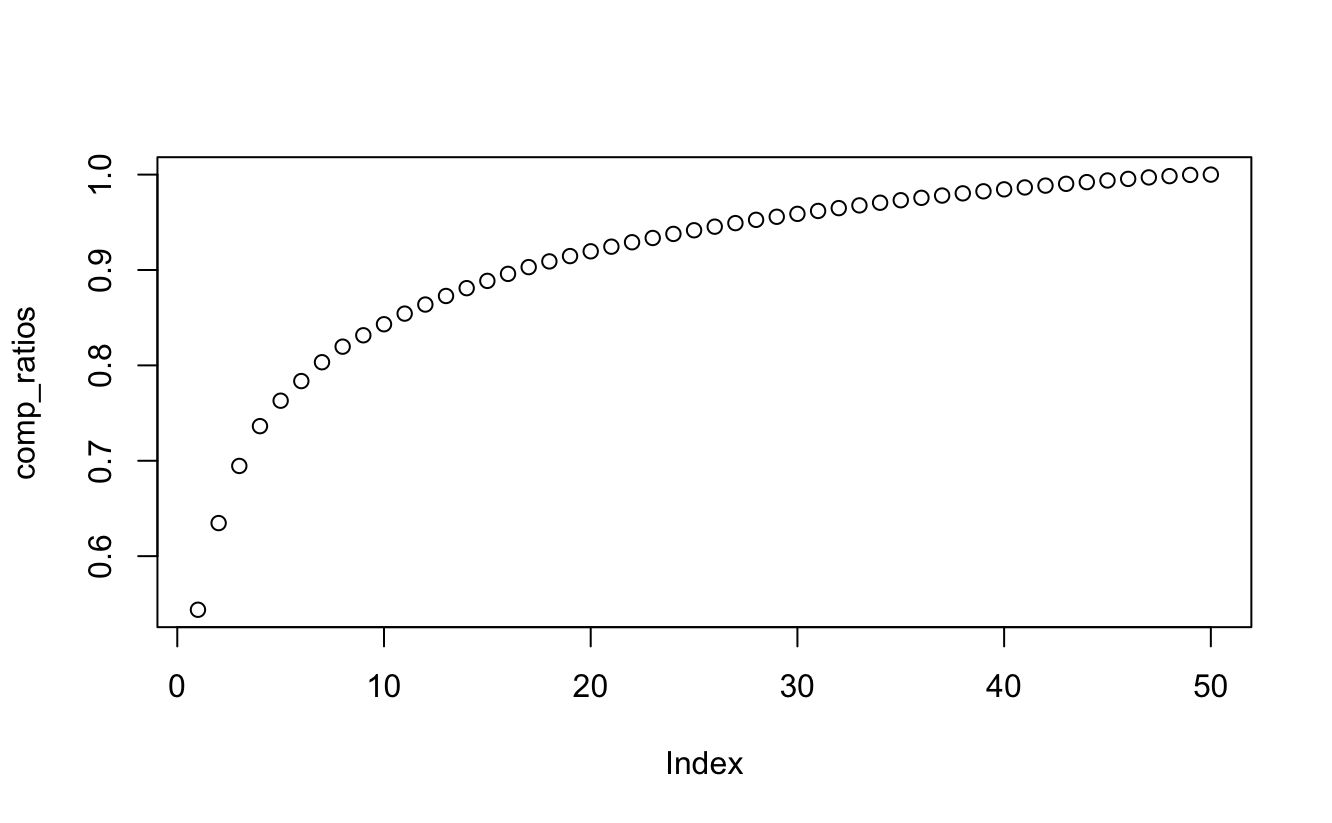
Finally, for each color vector created, the SVD function is called. The result is added into a new matrix, which depicts a grayscaled version of the original color image.

Ultimately, images greyscaled using SVD were similar, but not identical to images greyscaled traditionally — a fact that would ultimately be reflected in our later models.

1. PCA

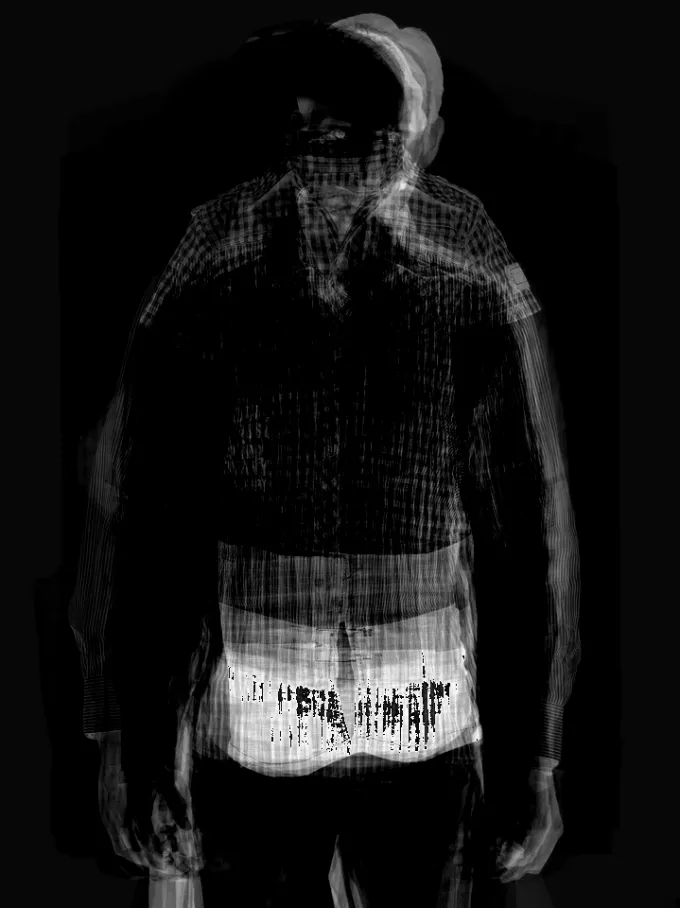
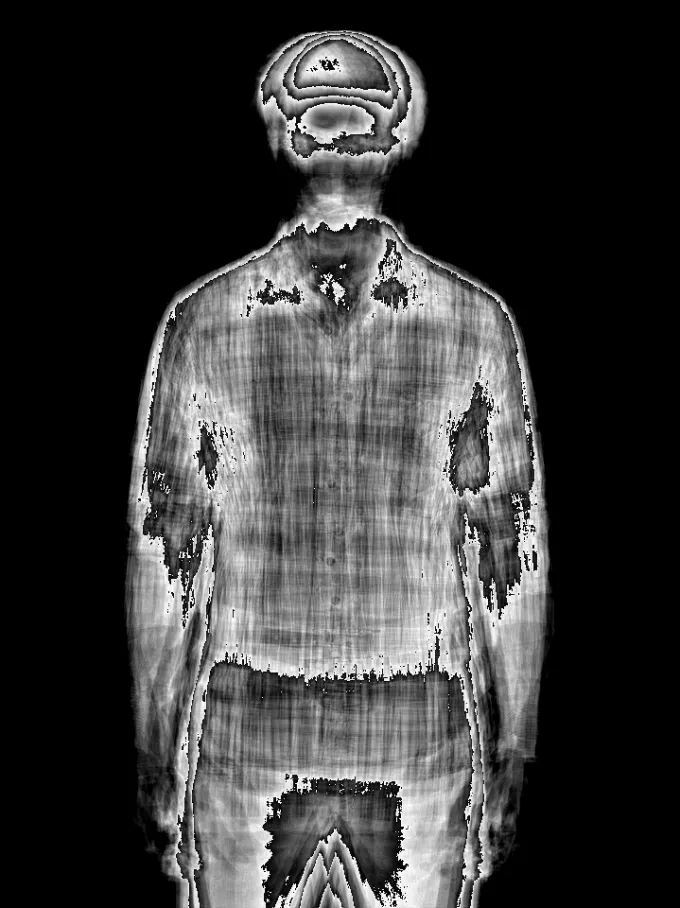
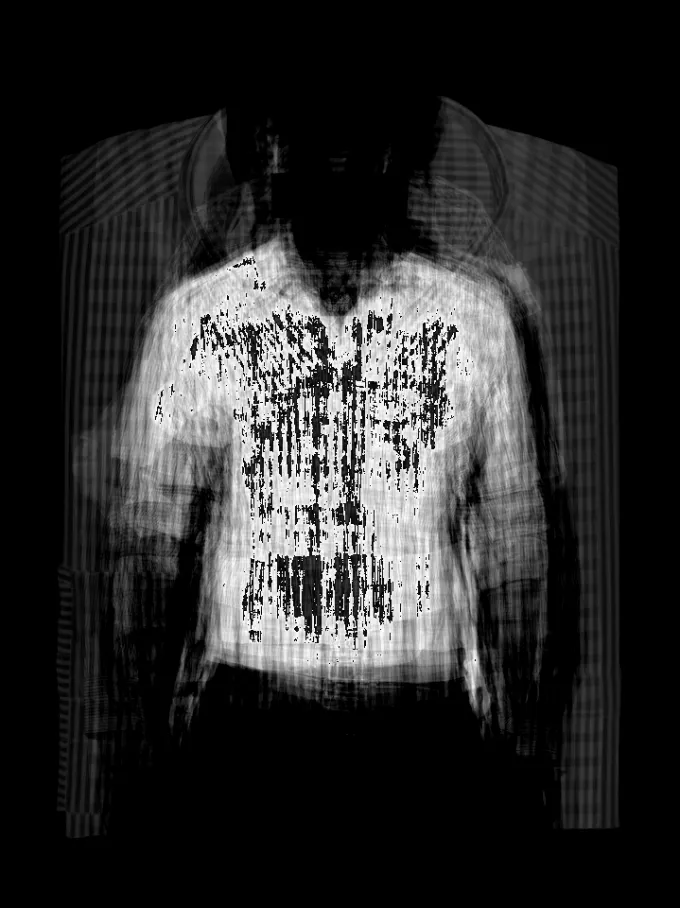
Using the black-and-white images created from SVD, we performed PCA on each set of 50 images to condense the information from them into fewer images. At first, we were unable to perform PCA due to not having enough memory to create the covariance matrix. With each image containing over 4,320,000 pixels, each covariance matrix would take over 70 TB of memory! We had to use the prcomp library, which finds one chunk of covariances at a time for the matrix instead of all together, to get the principal component vectors.

To decide how many vectors to pass on to the CNN, we analyzed the ratio of each principal component to the total variance of the set of images. Though computationally slow, we were able to obtain the proper amount of vectors and send them to the CNN model. Most of the calculated variance ratios for each number of vectors looked like this graph:

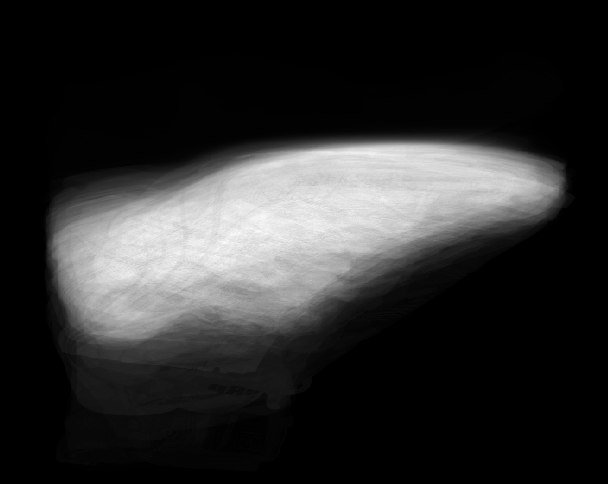
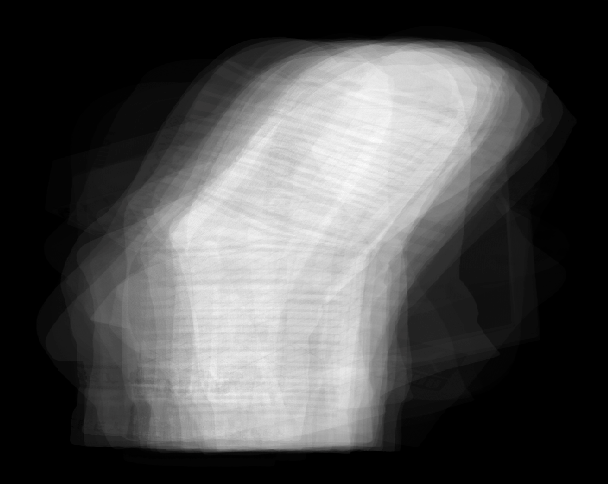
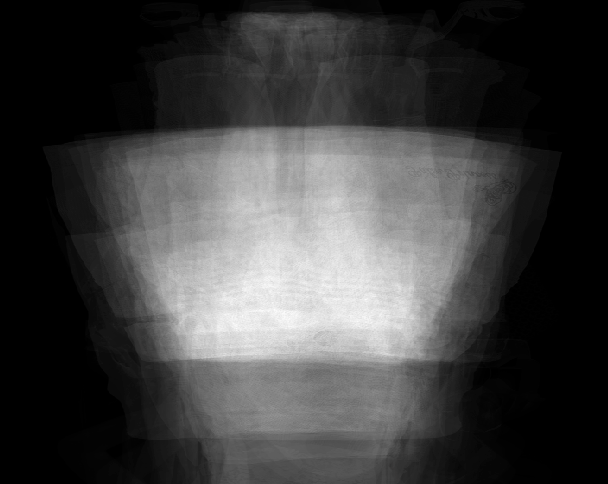


For all the categories, the first principal component accounts for about 50-60% of the total variance. At around six or seven principal components, the variance ratio starts to clearly converge, and at around 30-40 principal components, they accounted for 95% of the total variance. While six or seven principal components captured a satisfactory amount of the data, we decided to use 30-40 principal components (the 95% benchmark) to decide how many principal components to use, since we believed the lost variance would be magnified when fitting the components into the CNN model, so we wanted the principal components to cover as much of the data as possible.

Here were the first six principal components of shirts in the form of an image. While the principal components do not look pretty, there are some resemblances of a shirt. After looking at these components, we chose a threshold on the number of principal components that captured a lot of data due to the apparent loss in information.



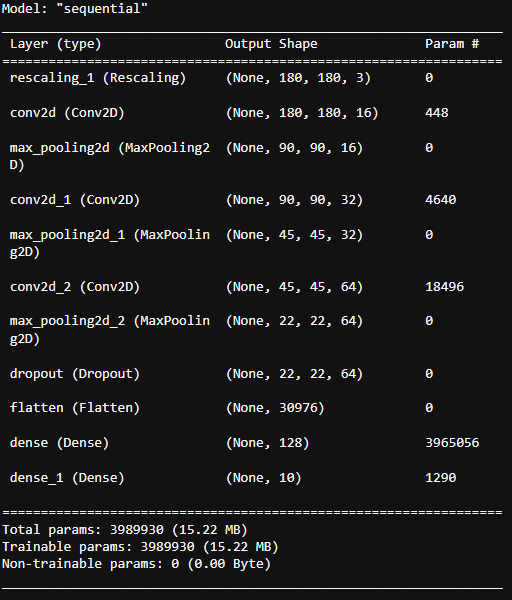
Here are the first principal components of dresses, pants, shoes, and socks. While they certainly represent their piece of clothing much better than the shirts, we decided to use the same threshold of the number of principal components as the shirts due to consistency and to play it safe.



1. Models

Three models were trained based on the three aforementioned categories. All models were convolutional neural networks with 10 parameters. The model uses convolutional, pooling, dense, etc layers. We applied dropout to all models in order to improve their accuracy and limit bias. Originally we made models that had data augmentation as well but we decided to not use them as we thought it would make the model worse. Most images of clothing aren’t really rotated and we the model would perform a lot worse due to the low the amount of images.

CNN was chosen due its popularity on image classification. Looking back on it a transfer learning model would have been much better due to our low amount of available data.

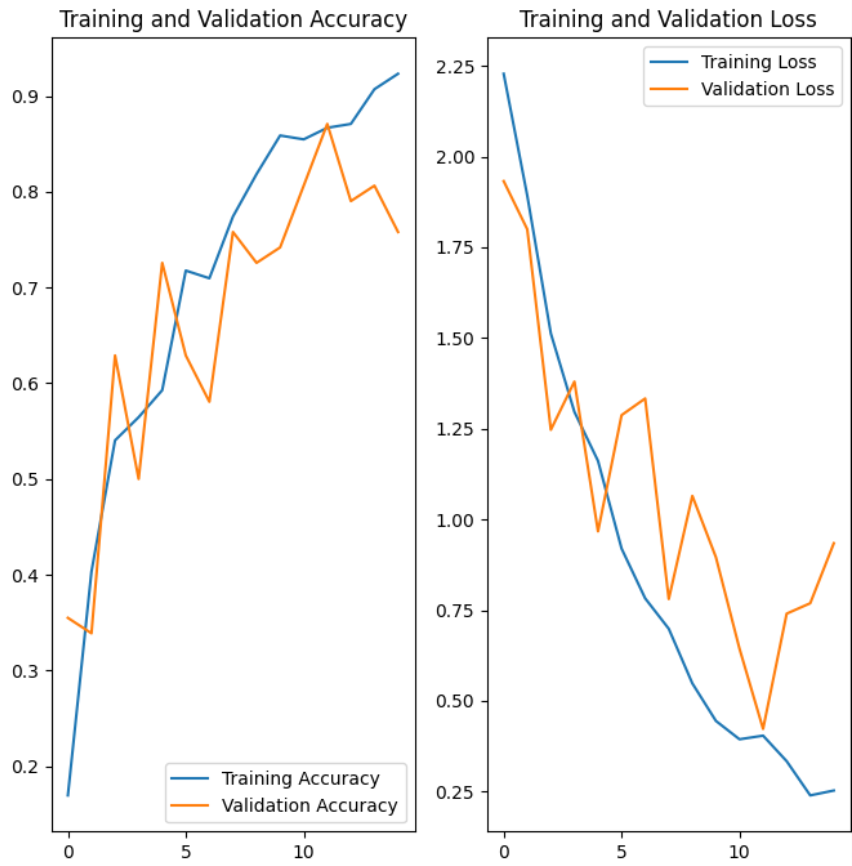
Summary of the models:  


The first model was run on the original images, sans-modification. Ultimately, this model resulted in:

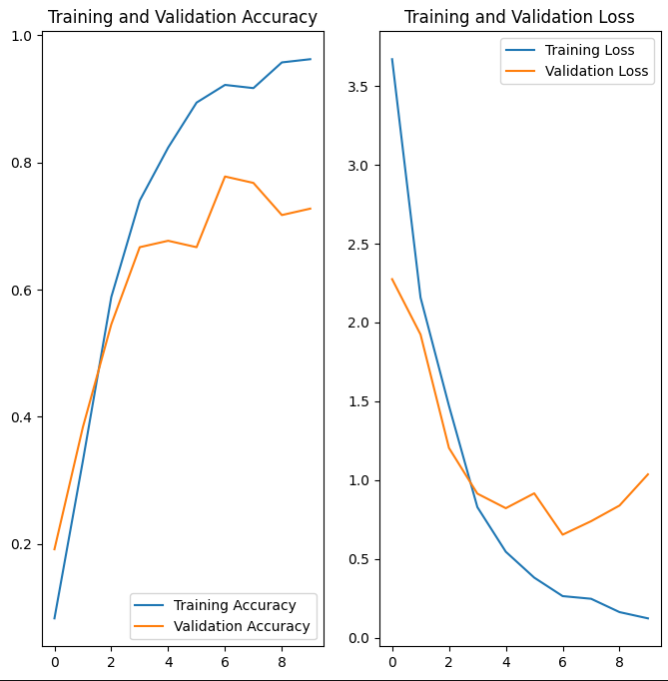


This model topped out at around 75% accuracy, however the validation loss started going up really early, indicating biasness.

The second model was run on greyscaled images with PCA run. This model largely performed better than the first, with a max accuracy of around 85% and loss of %50. However the validation lines were extremely sporadic indicating overfitting to the data.



Lastly, the third model was run on images that had been greyscaled using singular value decomposition. This resulted in a model with superior accuracy, including the best results compared to the other two. The validation loss follows the training loss curve much better and accuracy is even higher than the first model.



1. Further Tests & Results

To test the accuracy and usefulness of the above models, we chose to test them on a variety of real-world examples. Firstly, we chose three simple items of clothing found on the H&M website, consisting of a shirt, a pair of jeans, and a jacket.

These images, while largely similar in composition to our training data, proved difficult for our models. Ultimately, models 1 and 2 proved unable to classify any of the images correctly. Model 1 predicted that the images were formalwear, socks, and formalwear respectively, while model 2 predicted underwear, shirt, and shirt respectively.

Model 3, however, outperformed our expectations, correctly predicting all three of the images. While this, in large part, can partially be attributed to random chance, it underscored our confidence in model 3 as compared to the former 2.

To test model 3 further, we found a variety of images from fashion shows, mostly directly off the runway. An example of these items can be found below.



These images, for the most part, defy classification — even from a human perspective. Unsurprisingly, we found that even model 3 struggled greatly with them — only correctly predicting one of the six haute couture images (the remaining five were all predicted to be underwear.)

1. Conclusions

There is something quite silly about attempting to teach your computer to classify clothing. On one hand, this is largely because of the relative ease that we humans experience when identifying articles of clothing. And yet, on the other, it seems as if there can simply be no tangible insight gleaned from a project of this relative complexity.

Truthfully, one of the reasons we chose this project is because we believe no project should be prohibitively silly — that ultimately, true insight can be gleaned from all manner of locales.

In training, testing, and observing these models, our team realized just how difficult it is to do something as simple as identifying clothing from a first principles perspective. Indeed, while state-of-the-art solutions would likely be able to match human accuracy in this task, the fact that such a simple task would ultimately be so difficult for computers provided a glimmer of hope for a world where artificial intelligence seems to dominate so effortlessly.

Our later tests on haute couture only further cement this idea. Ultimately, AI and machine learning are wonderful tools for mankind, but we conclude that there is a barrier of argumentation that the two have not yet been able to cross.

For future work, we primarily propose models run with larger source data. We also suggest work to be done with more specific categories of clothing, and are curious to see if the same results would hold for other household objects. Overall, we believe there is great potential for this line of study and look forward to future work in the area.

# References

Z. N. Khudhair et al., "Color to Grayscale Image Conversion Based on Singular Value Decomposition," in IEEE Access, vol. 11, pp. 54629-54638, 2023, doi: 10.1109/ACCESS.2023.3279734.