Milestone Report 2
Capstone 2
Predicting Styles from Street Fashion Data

Maia Paddock

04/19/2019

Once the data cleaning and visualization process was complete, I moved on to setting up initial models. The three methods that I tried were PCA, t-SNE, and a deep neural network using convolution, batch normalization, and max pooling layers. PCA and t-SNE were performed on the data to see if there was a way to reduce the dimensionality of the problem. At this point in the analysis neither of these methods produced a useful output but they will be further explored once the analysis is migrated to a remote server.

Preparing the images with the data proved to be a challenge for my computer. At most, my computer was able to prepare 50,000 images, a little less than one fifth of the total images available for training. However, at most my computer was able to train on only 10,000 images. This precipitated the need to move the analysis onto a hosted virtual machine in order to train on the entire data set. I chose an image size of 150 by 100 pixels in order to keep the original scale. Operating on larger size pixel matrices than this also caused memory errors on my computer. Once I began running sample trainings, I also increased the tag occurrences being used for the data in order to get more samples of each tag. I began with all tags with less than 100 mentions in the entire dataset being included. However, once I realized that I would only be able to train on 10,000 images, I raised this limit to 1000 mentions in the dataset. Therefore, the models are multilabel classification on 21 different tags. All tags were transformed into vectors using MultiLabelBinarizer from sklearn and the binarizer was saved to transform the results back into text.

I began with a simple convolutional neural network setup in Keras, using two dimensional convolution layers, batch normalization, two dimensional max pooling, and dropout layers before the final dense layer for output. The final dense layer used a sigmoid activation, to facilitate multilabel classification. All other layers used rectified linear unit (ReLU) activation. Convolution was tested with symmetric windows between 2 and 6, with 4 performing the best as a window size. Batch size of approximately 256 and five epochs led to the best results, with any more epochs either being unnecessary as the loss and accuracy gain leveled off for the most part after 5 epochs. Four dropout layers were used to prevent overtraining. Two dimensional max pooling was used to reduce the dimensions between the layers. Different hidden layer structures were tested, with the most successful being five convolutional layers, each followed by batch normalization. Two dimensional max pooling and dropout were added after the first, third, and

fifth batch normalizations. Finally, the model was flattened and two dense layers were applied to provide the output.

At this point in the analysis, the model has a high precision but low recall. With the classification parameter that a successful prediction is a value higher than 0.1 in the output matrix, most of the tags have a precision of close to 1 but a recall closer to 0.2. Therefore, right now the model is solving the problem by predicting almost every image to have every tag. This should be solvable once more data points are added. I have performed some optimization on the model, but more will be performed once I begin running it on a remote server.