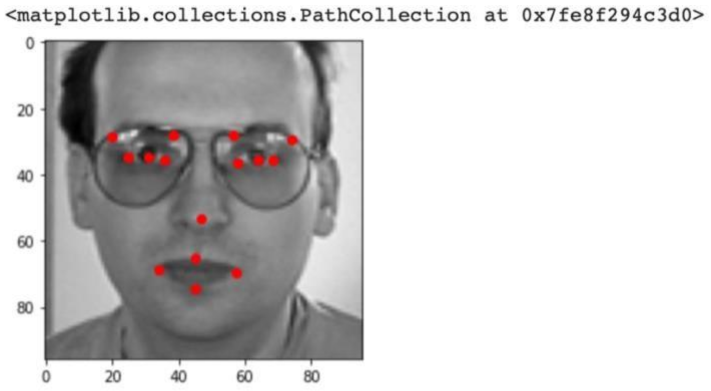
To receive the lowest mean squared error (MSE) I tried numerous architectures. I created my own neural network using convolutions, hidden layers, flattening, max pooling, and utilizing the ReLU function. After multiple attempts of customizing my neural network using one or two convolutions with varying kernel size, adjusting the number of outputs in each layer, changing the kernel size for max pooling, and using one to many dense layers, the pretrained network had the best results. With the factors batch size, learning rate, and the number of epochs being held constant for each trial, the pretrained network had the lowest MSE.

During training I used 15 epochs. I decided on this number through trial and error. After concluding on an architecture, I used 5 epochs and received an MSE of about 3-4 for the test data. As an attempt to lower the MSE further I increased the epochs to 10. I noticed that by doing so the MSE value would fluctuate after 6 epochs. It increased at 6 epochs to then decrease further at 7 epochs. This usually occurred one more time during four remaining epochs. I was hesitant of overfitting but interested to see how low I could get the MSE. With 15 epochs the MSE fluctuates a few times but has a lower value after completion. Before I would receive an MSE of about 3-4 for the testing data. Now I receive values between 2-3 and 1-2 for training data.

The only learning rate that wouldn't greatly increase the MSE during training was a learning rate of .001. I experimented using different optimizers (Adam and Stochastic Gradient Descent (STG)) with different learning rates. I expected STG to perform better but Adam had the best results.

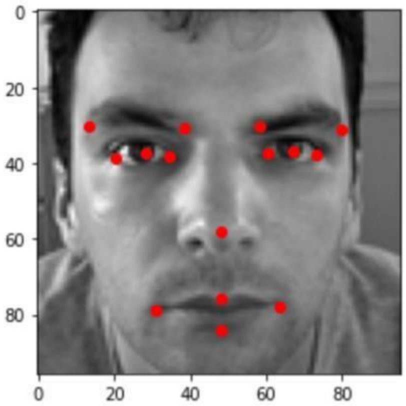
My predictions on the images from Kaggle turned out fairly well. Despite individuals wearing glasses, having their face tilted, or wearing items on their head, the key points were accurate. The blurrier an image is and the more hair they have interfering with heir face, the more error that occurred with the key pointers. I've listed some examples below.



```
<matplotlib.collections.PathCollection at 0x7fe8f294c3d0>
```

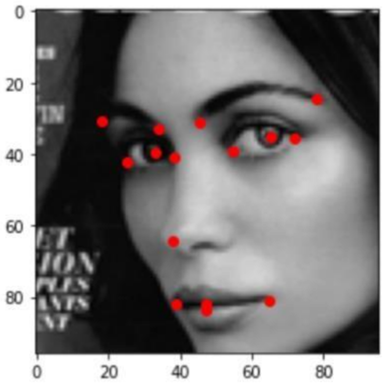
My predictions on this image were accurate despite the man wearing glasses. The eyes, eyebrows, and nose were accurately positioned.

```
<matplotlib.collections.PathCollection at 0x7fe8f2049ed0>
```



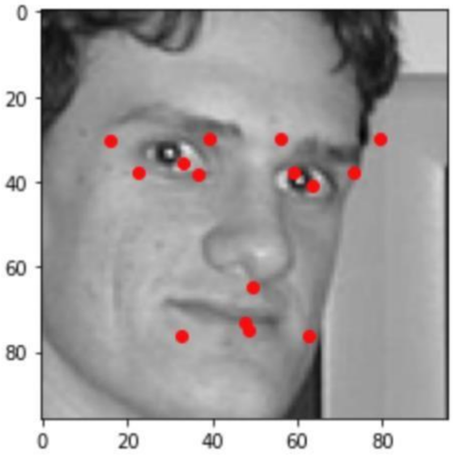
The points on this close-up picture were good. The right key pointers for the eyebrows could be better.

```
<matplotlib.collections.PathCollection at 0x7fe8f2346b10>
```



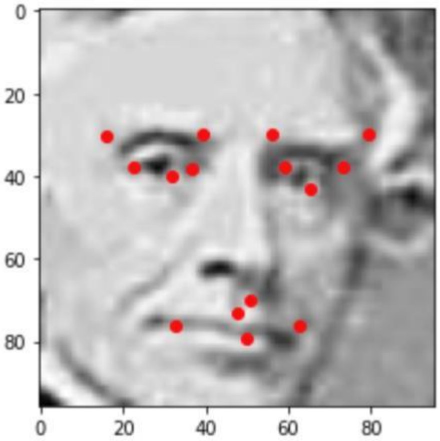
Here is an example where the woman's face is facing a different direction. Despite the complexity of the image the predictions were very accurate. The only point that is off is the right eyebrow. It was probably more of an estimate, which was accurate.

```
<matplotlib.collections.PathCollection at 0x7fe8e40d8690>
```



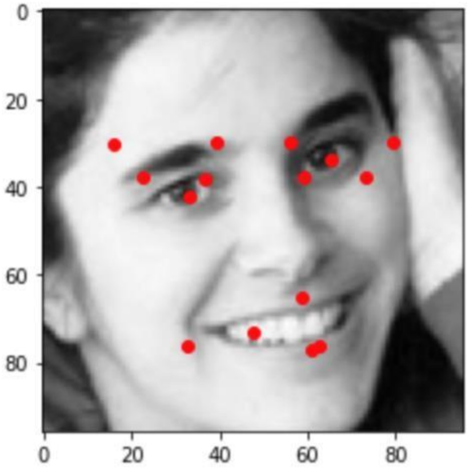
This is an example of bad predictions of an image. Every point was inaccurate. It's interesting this example had bad predictions because of the simplicity of the image. The man isn't doing anything complicated and the image isn't too blurry. There were better results on blurrier images.

```
<matplotlib.collections.PathCollection at 0x7fe575f4a8d0>
```



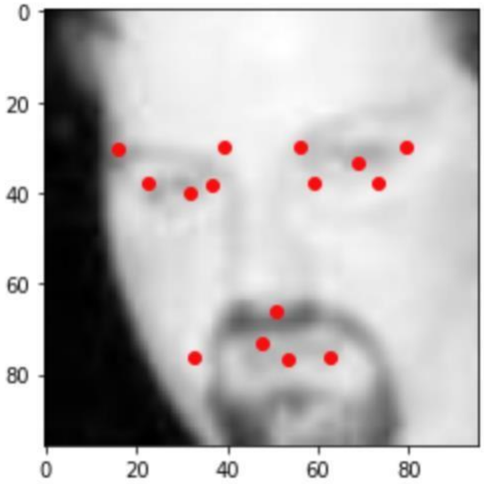
I was expecting to see worse predictions on this example because of the clarity. The image is older and blurry. More estimates had to be made and this is an example where they weren't as accurate. The eyes are close, but the nose and mouth region need more work.

```
<matplotlib.collections.PathCollection at 0x7fe8f22dabd0>
```



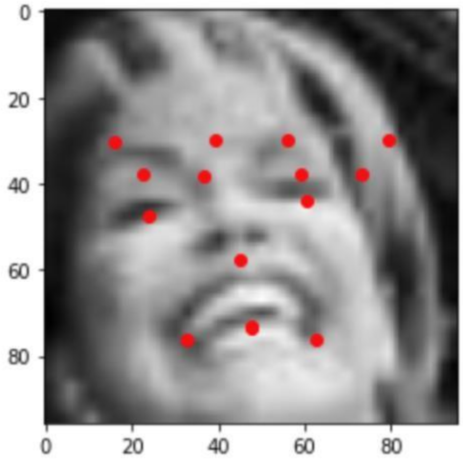
The predictions on this example are most likely off because of the hand on her face. The eyes and mouth aren't close to accurate. The nose is close but the pointer was placed on the shadow of here nose. That is a trend that I've noticed, placing the nose key pointer on the shadow of the individual's nose.

```
<matplotlib.collections.PathCollection at 0x7fe575c5c410>
```



The example above was a mix of good and bad predictions on a blurry image. The right eyes key pointers are on point while the left is good but could be better positioned. The nose pointer was placed either on the tip of the nose or just below, possibly mistaking the nose with the shadow of his nose. The mouth region is very inaccurate.

```
<matplotlib.collections.PathCollection at 0x7fe575648410>
```



My last example was chosen because of the complexity of the image. The woman is looking up with her eyes closed and the image is blurry. Besides the location of the center of her eyes, the key point predictions were off. It placed here right eyebrow key point in her hair and placed the nose on the tip of here lip, probably because of the tilt.

Conclusion: Overall my predictions were accurate. A minority of images had bad predictions. I noticed that more error occurred when the person closed their eyes, tilted their face, had hair on their face, opened their mouth, or the image was too blurry. I was shocked to see facial hair fair well; the key pointers didn’t have much difficulty determining where the nose or corners of the mouth were. Having glasses didn’t interfere with the predictions either. If the glasses were transparent the key pointers were placed in the right position. If the picture includes hands or other objects on their face the accuracy of predictions decreased dramatically. That can be seen in my examples.

Kaggle link: https://www.kaggle.com/c/facial-keypoints-detection