Project Report

Shuhang Zhou

**Convolutional Neural Network: Classifying Gender**

**NOTES:** In order to run our codes, you will need to have MATLAB 2017b, and Neural Network Toolbox version 11.0.

There are two folders: *“build\_nottingham\_cnn”* and *“test\_nottingham\_cnn”*

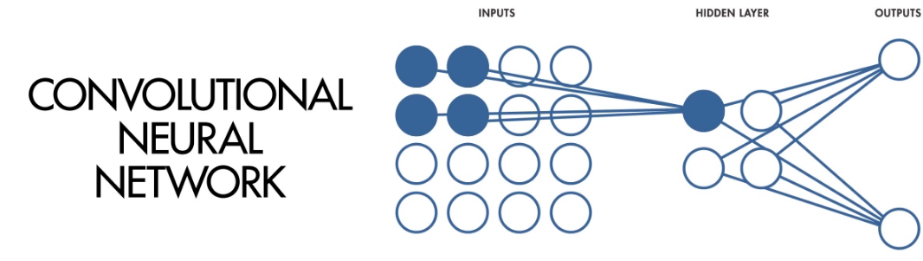
* *build\_nottingham\_cnn*: contain the files to build the CNN. (We already built the network and saved it as *net.mat*)
* *test\_nottingham\_cnn:* Run “*GenderClassificationCNN.m”* to test the performance of the network. Run “additional” to test the performance of right Cover faces.
  + I have also saved the results as .mat files
    - *additional\_result.mat*
    - *project\_result.mat*

# **Introduction to Convolutional Neural Network**

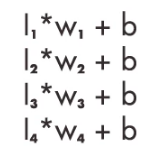
For our final project, we built a Convolutional Neural Network (CNN) to classify gender.

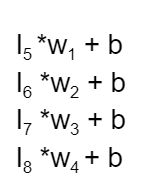
CNN is very similar to Backpropagation we learned in class. Actually, many people consider it as a “modern model of Backpropagation”. Since CNN automatically extract features from the images, it’s now widely used in Deep Learning for image recognition. Comparing to typical neural networks, CNN has three characteristics:

1. First, it simulates the **local receptive fields** in real sensory neurons. In typical neural networks, each hidden neuron is connected to all input neurons, while in CNN each hidden neuron is only connected to a subset of input neurons



1. Second, all the connections between each subset and its corresponding hidden neuron **share the same weight**. For example, the weight for the connection above is:

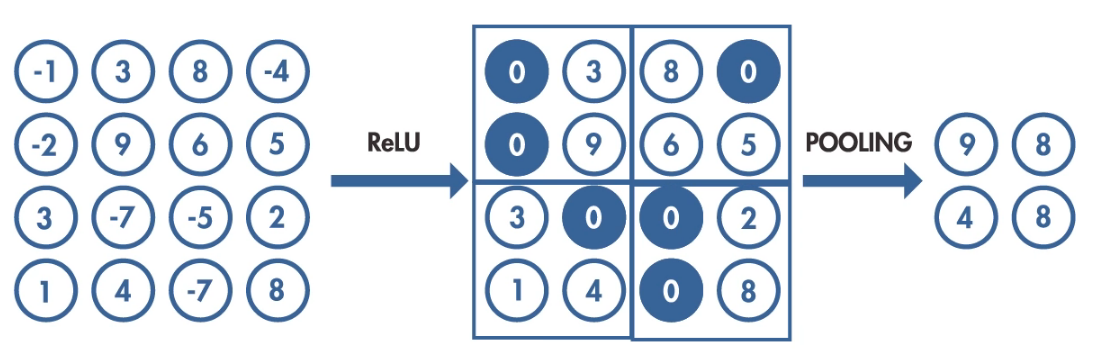


And when it comes to a different subset of input neurons, for example, for the connection of input neurons to a different hidden neuron, the connection would be: 

This characteristic enables CNN to extract features equally from all of its receptive fields.

(Note: “b” stands for bias here. Since we didn’t really talk about bias in this class, so we didn’t mention it. However, to be more precise, CNN has share weight AND --- biases!)

1. Third, it has **Activation and Pooling** features. So after each convolutional layer gets its output, negative values are set to zero by activation function ***ReLU.*** And then we use **pooling** to reduce the dimensionality of the features by only keeping the max value in each sub region. This pooling step reduces features the neural network needs to learn. The fewer features the network needs to learn, the more accurately the network performs. Therefore, as you can see, by building more hidden layers, we can further reduce the features the network needs to learn. This is exactly what we did when building the layers.



When CNN tries to extract features from images, in our model, it uses filters to slide through the images, and different filters will be detecting different features.

# **Previous Literature**

A lot of previous literatures have been done regarding Facial Recognition. According to Lawrence et al. (1997), some of the researchers used *Geometric Features* like nose width/length and mouth position to recognize faces. Some of them used the method called *Template Matching*, which is effective in images with “the same scale, orientation and, and lighting”. And there’s also a method called *Eigenfaces* which specifically using eigenvectors to recognize faces.

Lawrence et al. (1997) used CNN, combined with local image sampling and self-organizing map (SOM) neural network to recognize faces. With 200 test faces and 200 training images, they were able to achieve 94% ~ 99% accuracy. Their model can successively extract features through a hierarchical layer structure.

# **Our Project**

1. Research Question and Hypotheses

So we also wanted to do something fun with faces, so we decided to build a CNN to classify gender. Also, we are interested in how the network’s judgement would be impacted if the original trained faces are manipulated in the following way: adding noise of different level (10%, 50%, 80%, 100%), deleting upper half/lower half of face, using the mirror image, and inverting the image.

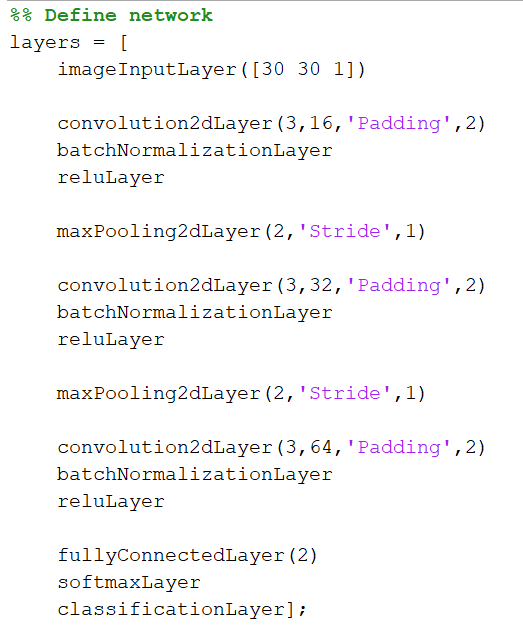
We assumed that classifying the gender for CNN would not be a very difficult task (indeed, the network we built is able to achieve almost 100% accuracy within the trained faces), so it would be interesting to see where its breaking point is. Most importantly, we want to see if the network is working well, how well it can stimulate human brain. As we know, as humans, we can easily classify gender even if the face image is with a reasonable amount of noise, presented in a mirror, or inverted. Can our network do it too? Also, by feed in these different kinds of manipulated images, we hope that we can understand some mechanics and crucial components the network uses to classify gender. (which is difficult, since in deep learning network, it’s always hard to figure out how the network is actually working).

Therefore, we have our 4 hypotheses about the network after training. First, low amount of noise (<50%) shouldn’t interfere with the judgement of the network too much but images with higher amount of noise would cause difficulty to the network. Second, the lower half will give more accurate results compared to the upper half, since many distinct features of male like mustache and thinner lips are presented in the lower half of the face. Third, it will not be very difficult for the network to classify gender from mirror images since most face features are left-right symmetrical. Fourth, inverted images will give the lowest accuracy, because CNN scan the image by sliding through it with filters, and inverted arrangement would be a dramatic change.

1. Research Procedure

For our overall research procedure, first we will build and train a CNN to classify gender using100 different gray-scaled faces (50 males, 50 females) from Nottingham Face database. All input images were cropped to 300x300 resolution and the resized to 30x30 due to the limited computation power we have. And in fact, resizing images is a very commonly used procedure! Second, we test the working network with 4 different types of manipulation and check its accuracy. In addition, we will also do a little bit of exploration by testing the generalization ability of the network with novel faces collected from Aberdeen Face Database.

Now I’m going to introduce the network architecture we built using Neural Network Toolbox from MATLAB. (Note: you’ll have to need version R2017b to run the code for building the network).



According to MATLAB, our network has 15 layers. Each *batchNormalization*, *ReLU* and *Pooling, and fullyConnectedLayer, softmaxLayer, and classificationLayer* are also considered as an individual layer. However, from a more intuitive sense, because we only have 3 convolution layers (*convolution2dLayer*), it’s more reasonable to call it a network with 3 **hidden layer sets.** In each **hidden layer set**, we have one *convolution2dLayer,* one *batchNormalizationLayer,* one *reluLayer*, and one *maxPooling2dLayer.* And all the *fullyConnectedLayer, softmaxLayer, and classificationLayer* can be considered **output layer set** which manipulate the results before it gets the output. Therefore, the overall architecture looks like this:

*Input Layer-> Convolutional Hidden Layer Set**1-> Convolutional Hidden Layer Set**2*

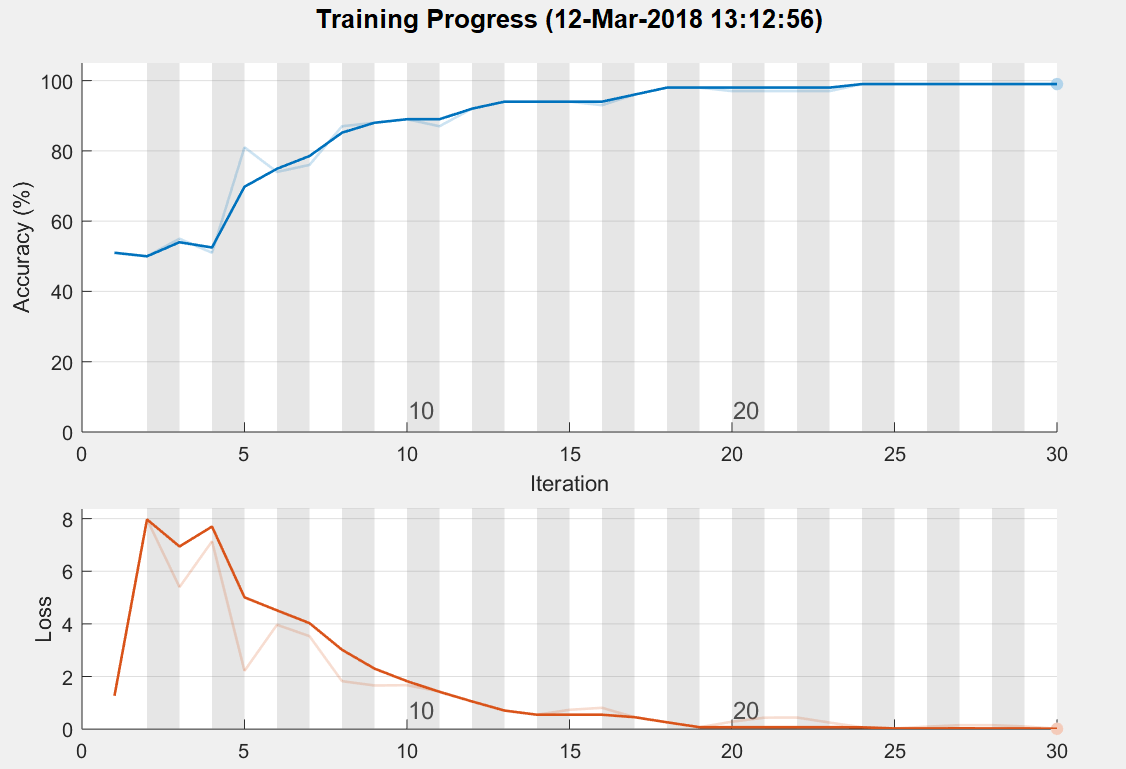
*-> Convolutional Hidden Layer Set**3-> Output Layer Set*

As mentioned in the first section “Introduction to Convolution Neural Network”, the purpose of having multiple convolutional hidden layer set is to reduce the dimensionality of the features, in other words, reduce the number of important features the network needs to learn. The fewer features the network needs to learn, the easier it is for the network to categorize things.

Here is a brief explanation of the function of each layer:

* *imageInputLayer:* input the image as matrix
* *convolution2dlayer(a, b, ‘Padding’, c):* apply convolutional structure(explained in the first section) to the input layer
  + *a:* the size of filters, axa
  + *b:* the number of filters (the number of hidden neurons), each filter detects different features
  + *‘Padding’:* sometimes the filters move out of the image, the padding fills the border of the image
  + *c:* size of padding
* *batchNormalizationLayer:* normalize the results
* *reluLayer:* explained in the first section. Nonlinear Activation function.
* *maxPooling2dLayer:* explained in the first section. Pick up the most important information in each subset receptive field
* *fullyConnectedLayer:* contains all the important features the network has picked up
* *softmaxLayer:* normalizes the output from *fullyConnectedLayer* and results a matrix of probabilities
* *classificationLayer:* makes classification based on the probabilities matrix

Note that all the building and training processes were done using Neural Network Toolbox from MATLAB. The training was done using the gradient descent method called Stochastic Gradient Descent with Momentum (SGDM), which updates the weights to minimize error. We used learning rate of 0.01, and 30 epochs. Here is what the training process looks like:



1. Results

|  |  |
| --- | --- |
| **Manipulation** | **Accuracy** |
| Training Data | 99% |
| Mirror | 85% |
| Inverted | 63% |
| Upper Half Covered | 50% |
| Lower Half Covered | 50% |
| 10% Noise | 94% |
| 50% Noise | 66% |
| 80% Noise | 53% |
| 90% Noise | 51% |
| 100% Noise | 49% |
| Novel Faces | 47.5% |

As we can see, for the noise manipulation, the accuracy progressively goes down as the amount of noise goes up. The network performs very similar to human. For the Upper/Lower half face cover, the network has only 50% accuracy. Actually, one thing really interesting is that, when the upper half is covered, the network identifies all the faces as male; when the lower half is covered, the network identifies all the faces as female. Our interpretation is that the network relies on the upper half features more heavily than the lower half when classifying females. The lower half contains more males features like mustache and thinner lips. Therefore, if the lower half is missing, the network will just identify the image as females. This is a very plausible hypothesis to us, however, it is very hard to directly prove this, since it’s very hard to look at the features deep learning network cares about. Whether it’s true or not may need further research in the future. For the mirror images, the accuracy is mostly preserved as we hypothesized, because the human face features are mostly left-right symmetrical. Lastly, for the inverted images, the network shows almost by chance accuracy. The reason is that CNN scan the images by sliding the filter step by step, so it has only analyzed the image in the original orientation. In other words, the features it’s learned to classify gender are not applicable to the inverted images, because the position of features are dramatically altered. It’s not hard to see that our network is not able to preprocess the images before analysis --- it cannot tell if the images is inverted or not. Also, for the exploration part, our network shows by chance accuracy, which means our network has very poor generalization ability. This is a common problem of neural network. For CNN, it actually has better generalization ability compared to typical neural network if the training data is large enough. The poor accuracy of novel faces is also caused by Overfitting, which means the network is trained too specific to our training data. There are several approaches to avoid overfitting, for example, adding *dropOutLayer* to the network. Although it’s just for our exploration and not the main purpose of our project, it would be interesting if one day we have the knowledge to investigate this problem.

1. Conclusion

To conclude, our network performs similarly to human in some ways, but still lower power than human brain. When presented with covered faces and images with noises, the accuracy of the network compromises just like humans. However, when presented with inverted and mirror images, the network starts to have problems while human would still classify gender without any problems. Unlike human brain, our network is only able to analyze the images in the way it’s trained. Moreover, it has poor generalization ability.

1. Future Studies/ Applications

According to the performance of our CNN, we think there are a couple of things we can try to improve the network in the future. First, adding effective image normalization procedure before analysis. As we can see from our experiments, the network is not able to tell if the pictures are manipulated or not, and that impacts the accuracy a lot. Therefore, if an effective image normalization way can be added to the network, it will improve the performance. Second, more effective features extractions. There is a lot of parameters we can adjust when defining the layers of CNN, such as, filter size, number of filters, learning rate, epochs. By adjusting these parameters, theoretically, the features the network eventually picks up should vary. If the network picks up more representative features for male and female, then it will not only improve the performance, it should also improve the network’s generalization ability.

Gender classification can be very useful in real life, such as in surveillance cameras, video games, and social media. We are looking forward to seeing more advanced and effective methods to classify gender.

# **Bonus**

After our presentation, the professor asked us to do some additional investigation regarding some of the things in our project. Our original plan was to test the hypothesis of *“the network relies on the upper half features more heavily than the lower half when classifying females. The lower half contains more males features like mustache and thinner lips. Therefore, if the lower half is missing, the network will just identify the image as females”* regarding the covering of Upper/Lower faces. However, it’s been very hard for us to come up with a direct way to prove that hypothesis. Therefore, we decide to do some additional manipulation to the data to see if there’s any interesting thing we can find.

We then manipulated the faces by covering the right side of the faces and feed these manipulated faces back to the CNN we built. The result is very interesting: we got 79% accuracy. Note that we got only 50% accuracy for both upper/lower covered faces. Why is it the case that our network can classify gender so well with right covered faces? Our interpretation is that it has to do with symmetry. Remember we did the mirror manipulation, and the accuracy is not very bad? The reason was human face features are highly left-right symmetrical. It’s plausible to infer that it’s due to the same reason why right covered faces got high accuracy: since human faces features are mostly left-right symmetrical, the network does not really need that much information about the covered right half face, after all, it already has the left half, and it’s quite enough (in other words, the left and right half face contains mostly similar information).

Again, prove hypothesis in an unsupervised network like CNN is difficult, we are not able to directly prove it with our current knowledge. However, we are proud of what we’ve done so far, and we are interested in further development of gender classification neural network in the future.

**Sources**

Lawrence, S., et al. “Face Recognition: a Convolutional Neural-Network Approach.” IEEE Transactions on Neural Networks, vol. 8,

no. 1, 1997, pp. 98–113., doi:10.1109/72.554195.

“TrainNetwork.” Create Simple Deep Learning Network for Classification - MATLAB & Simulink Example, The MathWorks, Inc.,

www.mathworks.com/help/nnet/examples/create-simple-deep-learning-network-for-classification.html#d119e2084.

MATLAB Mathworks Neural Network Toolbox Tutorial https://www.mathworks.com/help/nnet/examples/create-simple-deep-learning-network-for-classification.html

Introduction to Deep Learning: What Are Convolutional Neural Networks?

https://www.youtube.com/watch?v=ixF5WNpTzCA

Nottingham Face Dataset

http://pics.stir.ac.uk/2D\_face\_sets.htm

Aberdeen Face Dataset

http://pics.stir.ac.uk/2D\_face\_sets.htm