BANA 290 Final Project Report

Team: Kevin Cheung, Mira Daya, Ankit Jain, Misha Khan, Kathy Yu-Hsin Lee

Project: COVID-19 Mask Detection in Images

The Problem:

a. **What is the binary outcome variable?**
   Y = 1 if a face mask is detected in the image, 0 otherwise.
b. **What are your independent variables (X variables)?**
   The independent variables are the images of people with or without masks. These images are converted to RGB arrays to allow for easier processing.

The Dataset:

a. **How many observations do you have in your dataset?**
   There are 11,760 images in our dataset which we converted to an RGB array to use for the creation of the models. Each image was standardized to the shape of: (64, 64, 3).
b. **How many observations have an outcome of zero, and how many have an outcome of one?**
   Y = 1, n = 5880
   Y = 0, n = 5880

1) Fit training set well:

a. What metric(s) are you using to evaluate your training performance (training accuracy, recall, precision, false positives, etc.)? Explain why these metrics are useful in your data context.

For training performance, we will be using accuracy as it is important in the context of our mask classification data to get the correct classifications of all cases with or without mask. If an image is classified wrongly as no mask, it may cause issues in terms of logistics for the ones using the model. If the model wrongly classifies an image as 'with mask' when there is no mask, this may have implications in terms of safety associated with the health of surrounding individuals.

b. Comparing to human level performance (or any other benchmark in your data context), how well is your model fitting the training data?

Human level performs at 100% accuracy (that is, a human should be able to determine whether a person is wearing a mask or not). Here, our model is only performing at 84.01% accuracy which means there can be room for improvement.

c. Can you improve your training performance using a bigger neural network? Report a table in which you show the training performance for a total of ten different models (with different numbers of hidden layers and/or different number of hidden units).

Model Number	**Number of Layers**	**Number of Units**	**Training Performance**
1	1	64	0.9556
2	1	128	0.5000
3	1	512	0.9412
4	1	2048	0.9551
5	1	4096	0.9598
6	2	1st layer: 4096 2nd layer: 128	0.9128
7	3	1st layer: 4096 2nd layer: 128 3rd layer: 64	**0.9688**
8	4	1st layer: 4096 2nd layer: 128 3rd layer: 64 4th layer: 32	0.9580
9	5	1st layer: 4096 2nd layer: 128 3rd layer: 64 4th layer: 32 5th layer: 16	0.9660
10	6	1st layer: 4096 2nd layer: 128 3rd layer: 64 4th layer: 32 5th layer: 16 6th layer: 8	0.9590

Yes, we achieve the highest training performance of 96.88% when adding 3 layers, where the first layer includes 4096 hidden units, 128 hidden units in the second layer, and 64 hidden units in the third layer.

d. Can you improve your training performance using a better optimization algorithm? Report a table in which you show the training performance for at least three different optimization algorithms.

Optimization Algorithm	**Training Performance**
Adam	**0.9688**
Adagrad	0.8876
Adamax	0.9420
Adadelta	0.8672

No, our training performance stayed the same at 96.88% using the Adam optimization algorithm.

e. Can you improve your training performance using a higher number of iterations (epochs)? Report a table in which you show the training performance for at least three different numbers of epochs.

Numbers of Epochs	**Training Performance**
45	0.9846
60	0.9548
75	**0.9911**

Yes, we were able to improve our training performance to 99.11% by using 75 epochs.

f. Can you improve your training performance using a better weight initialization? Report a table in which you show the training performance for at least three different weight initializations.

Weight Initializations	**Training Performance**
HeUniform	0.9891
RandomUniform	0.9861
RandomNormal	0.9892

No, our training performance did not improve by using the above different weight initializations.

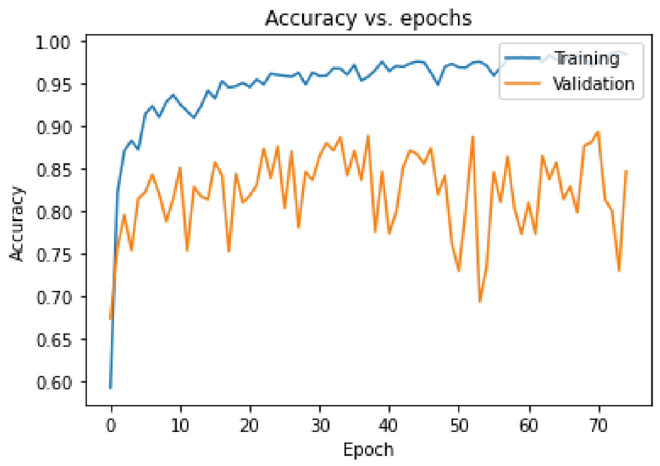
g. Explain which model is the best model fitting your training set? Use this model for the next part.

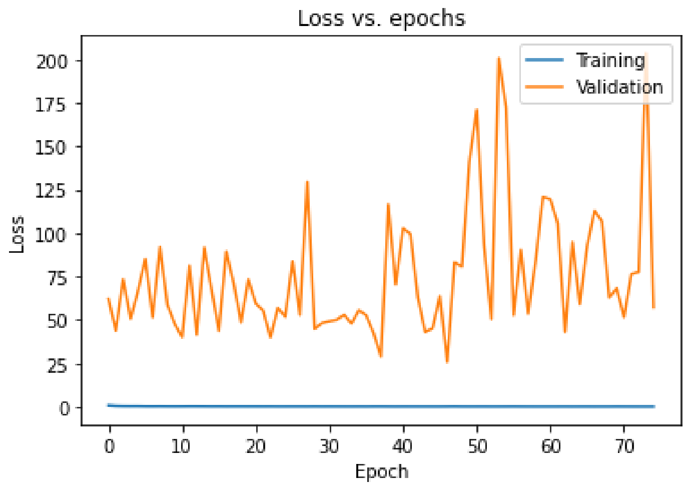
The best fitting model includes 3 hidden layers with the default weight initializer, Adam as the optimizer, and 75 epochs. This model gave us the highest accuracy of 99.11%.

2) Fit validation set well:

a. Compared to training performance, how well is your model fitting the validation data? Explain if you have an overfitting problem.

The training data had a performance of 99.11% with a loss of 0.0298. On the other hand, the validation data had a performance of 84.61% with a loss of 57.1895. This indicates an overfitting problem which we can see in the graphs below.





b. Can you improve your validation performance using L2 regularization? Report a table in which you show the validation performance for five different penalty (λ) rates.

Penalty Rates	**Validation Performance**
0.001	0.8435
0.01	**0.8865**
0.1	0.5000
0.5	0.5000
0.05	0.8406

Yes, the validation performance does improve when using L2 regularization at a penalty rate of 0.01, which returned a validation performance of 88.65%.

c. Can you improve your validation performance using dropout regularization? Report a table in which you show the validation performance for five different dropout rates.

Dropout Rates	Validation Performance
0.1	0.8861
0.2	0.8614
0.3	0.7474
0.4	0.8746
0.5	0.6722

No, dropout regularization did not help in improving the validation performance.

d. Can you improve your validation performance using a mixture of dropout regularization and L2 regularization? Report a table in which you show the validation performance for five different combinations.

Penalty Rates	Dropout Rates	Validation Performance
0.01	0.2	0.8482
0.02	0.15	0.8491
0.01	0.25	0.8784
0.02	0.2	0.8550
0.01	0.034	0.8431

No, a mixture of dropout regularization and L2 regularization did not help to improve the validation performance.

e. Can you improve your validation performance using batch-normalization? Report a table in which you show the validation performance for five different batch-normalizations (one model with default TF batch-normalization parameters and four models with customized parameters).

Batch-normalization	Validation Performance
Default	0.8002
Momentum = 0.95	0.8040

Momentum = 0.95
Epsilon = 0.005
	0.6637

Momentum = 0.95
Epsilon = 0.005
Beta Initializer Random Normal: Mean = 0, SD = 0.05
	0.6395

Momentum = 0.95
Epsilon = 0.005
Beta Initializer Random Normal: Mean = 0, SD = 0.05
Gamma Initializer Constant: Value = 0.9
	0.5000

No, the validation performance did not improve upon using batch-normalization.

f. Can you improve your validation performance using a mixture of batch-normalization and dropout regularization? Report a table in which you show the validation performance for five different combinations.

Batch-normalization	Dropout Rates	Validation Performance

Momentum = 0.95
Epsilon = 0.005
Beta Initializer Random Normal: Mean = 0, SD = 0.05
Gamma Initializer Constant: Value = 0.9
	0.034	0.5089

Momentum = 0.95
Epsilon = 0.005
Beta Initializer Random Uniform: Mean = 0, SD = 0.05
Gamma Initializer Constant: Value = 0.9
	0.2	0.6395

Momentum = 0.95 Epsilon = 0.005 Beta Initializer Random Normal: Mean = 0, SD = 0.05 Gamma Initializer Constant: Value = 0.9	0.1	0.8240
Momentum = 0.8 Epsilon = 0.005 Beta Initializer Random Normal: Mean = 0, SD = 0.05 Gamma Initializer Constant: Value = 0.9	0.05	0.7474
Momentum = 0.95 Epsilon = 0.01 Beta Initializer Random Normal: Mean = 0, SD = 0.05 Gamma Initializer Constant: Value = 0.9	0.03	0.5264

No, the validation performance did not improve upon using a mixture of batch-normalization and dropout regularization.

g. Can you improve your validation performance using a mixture of batch-normalization and dropout regularization and L2 regularization? Report a table in which you show the validation performance for five different Combinations.

Batch-normalization	**Dropout Rates**	**Penalty Rates**	**Validation Performance**
Momentum = 0.95 Epsilon = 0.005 Beta Initializer Random Normal: Mean = 0, SD = 0.05 Gamma Initializer Constant: Value = 0.9	0.01	0.034	0.7704
Momentum = 0.1 Epsilon = 0.005 Beta Initializer Random Normal: Mean = 0, SD = 0.05	0.03	0.005	0.7317

Gamma Initializer Constant: Value = 0.9			
Momentum = 0.95 Epsilon = 0.005 Beta Initializer Random Normal: Mean = 0, SD = 0.05 Gamma Initializer Constant: Value = 0.9	0.008	0.034	0.5000
Momentum = 0.95 Epsilon = 0.5 Beta Initializer Random Normal: Mean = 0, SD = 0.05 Gamma Initializer Constant: Value = 0.9	0.008	0.003	0.7168
Momentum = 0.95 Epsilon = 0.005 Beta Initializer Random Normal: Mean = 0, SD = 0.05 Gamma Initializer Constant: Value = 0.01	0.005	0.04	0.7997

No, the validation performance did not improve upon using a mixture of batch-normalization and dropout regularization and L2 regularization.

h. Explain which model is the best model fitting your validations set? Use this model for the next part.

The best model fitting our validation set is the model we had with the best training performance, along with the addition of the L2 regularization with a penalty rate of 0.01, which gave the best validation performance of 88.65%.

3) Best model:

a. Explain which model would be your final (best) model and why. Compare the training performance, validation performance and test performance.

Our best model consists of three hidden layers, with 4,096 hidden units in the first layer, 128 hidden units in the second layer, and 64 hidden units in the third layer. Within each of these 3 layers, we include L2 regularization with a penalty weight of 0.01. This 3 layer model provided the best training performance, and after we included the L2 regularization, this provided the best validation performance.

Using this model, we have the following training, validation, and testing performances:
Training performance: 90.17%
Validation performance: 87.41%
Testing performance: 94.22%