Authors: Ze Hong Wu, Ying Jie Mei

Introduction

This report describes the work we carried out during Stage V.

This report will be structured to include the following sections:

- Part 1: Changes attempted by each team member; Results and Observations
- Part 2: Model Architecture
- Since there is no further improvement stage, there will not be a Part 3 to discuss future plans. We will write up a full report of our work after completing this Stage V report..

Part 1: Changes Tested

At the end of Stage IV, we selected a model based on a reduced version of the DenseNet architecture. The model architecture will be revisited in Part 2 of this report.

During Stage IV of this project, we tested the following changes to the DenseNet model:

Ze Hong Wu: Learning rate scheduler
Ying Jie Mei: Compression factor

Part 1.1: Work by Ze Hong Wu

Part 1.1.1: Changes Tested

I started my work by attempting to combine my work and Ying Jie's work into a hybrid model that, I initially thought, would perform better (defined as having a higher accuracy) than either model.

Table 1: Loss - Accuracy - F1 Macro values for the Stage IV models and the combined model.

Model	Based on	Changes	Loss	Accuracy	F1 Macro
Adam	Ying Jie's work	n/a	0.3813	0.8574	0.8589
Compression	Ze Hong's work	n/a	0.4658	0.8433	0.8422
Combined	Adam + Compression	Merges the two above models	0.5318	0.8143	0.8164

The combined model performed worse than its predecessors. This behavior was not expected, and I concluded that this may be caused by destructive interference between the Compression

model changes and the Adam model changes. I reviewed the code used for Stage IV that I based my Stage V work on and discovered a number of inconsistencies:

- The Adam model used a learning rate scheduler that was not mentioned in the Stage IV report. I briefly tested this scheduler during Stage III work but did not continue researching it during Stage IV.
    - When training the Combined model, I did not include a learning rate scheduler as I was under the initial expectation that the model with Adam did not have one. The Compression model did not use a scheduler.
- The Adam model also includes a transition model right before the final layers that is not present in the Compression model.
- The Compression model suffered from a missing ReLU() activation layer in the transition layers caused by a typo in naming the layer outputs.

After correcting the missing ReLU, I attempted to back-trace my work to see what might be causing this issue.

Table 2: Loss - Accuracy - F1 Macro values for the combined model and sequentially cut back versions..

Model	Based on	Changes	Loss	Accuracy	F1 Macro
Combined	Adam + Compression	Merges the two above models	0.5318	0.8143	0.8164
No-Compress	Combined	Reset compression value to 0.5	0.5311	0.8147	0.8168
Last-transition	No-compress	Re-added removed transition layer	0.5304	0.8121	0.8177

My conclusion from the back-tracing is that the cause of the discrepancy is in the learning rate schedulers. As a result of this, I started from the combined model and trained it using several different schedulers.

Table 3: Loss - Accuracy - F1 Macro values for various scheduler models

Model	Based on	Changes	Loss	Accuracy	F1 Macro
Combined	Adam + Compression	Merges the two above models	0.5318	0.8143	0.8164
Learning-1	Combined	-90% lr at 50% and 75% epochs; initial lr of 0.01	0.4581	0.8464	0.8164
Learning-2	Learning-1	-50% lr instead of -90%	0.5171	0.8304	0.8275
Learning-3	Combined	-0.0005 lr at 25%, 50%, and 75% epochs; initial lr of 0.002	0.5068	0.8384	0.8365

Learning-4	Combined	-0.001 lr per epoch; initial lr of 0.025	0.6234	0.7937	0.7933
Learning-5	Learning-1	Set compression to 0.5	0.4495	0.8455	0.8438
Learning-6	Learning-5	Used custom scheduler 1	0.5124	0.8192	0.8179
Learning-7	Learning-6	Used custom scheduler 2	0.4880	0.8326	0.8316

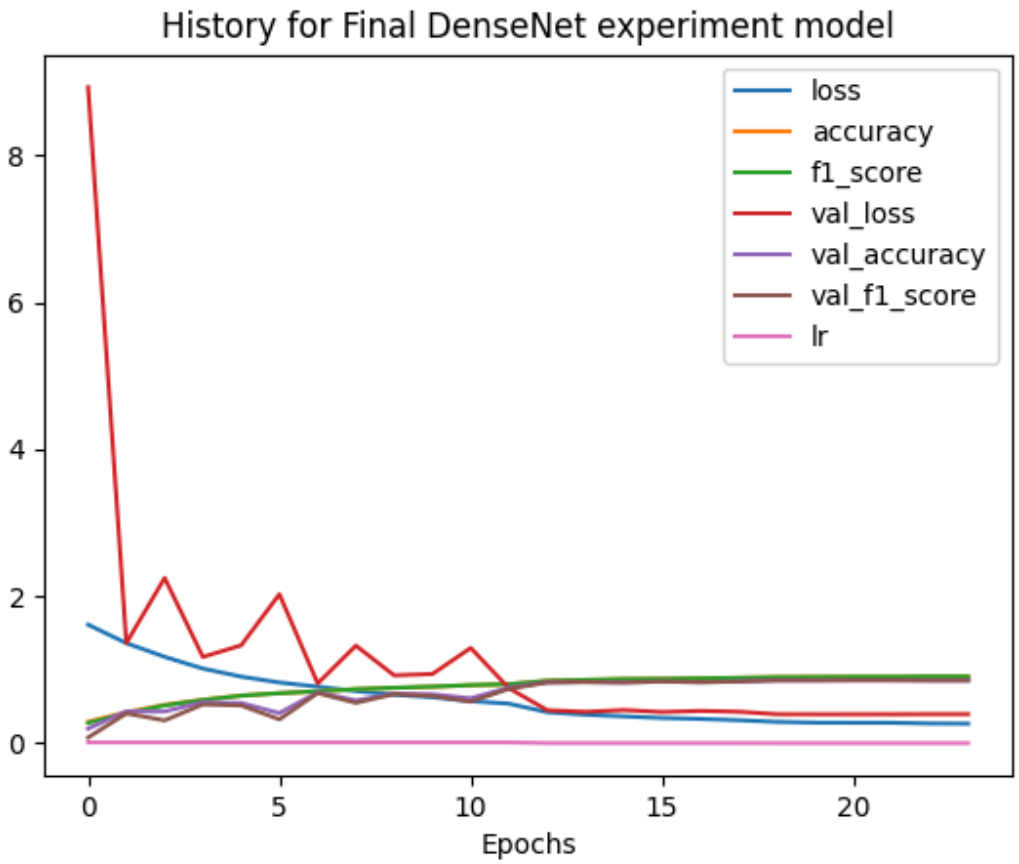
The two custom schedulers are described in the table below. My hypothesis, based on the results I saw in Ying Jie's Adam model and my Learning-1 model, was that careful reduction of the learning rate could improve model training results. I hoped that gradually reducing the learning rate instead of doing it at sharp discrete steps might lead to better accuracy.

Table 4: Custom schedulers and how they work

Scheduler	Epoch range	Behavior
1	1-6	Gradually drop from 0.01 to 0.005 in steps of 0.001
	7-12	Drop from 0.0042 to 0.0012 in steps of 0.0006
	13-18	Drop from 0.0011 to 0.0006 in steps of 0.0001
	19-24	Drop from 0.0006 to 0.0001 in steps of 0.0001
2	1-8	Drop from 0.01 to 0.00125 in steps of 0.00125
	9-16	Drop from 0.001 to 0.000125 in steps of 0.000125
	17-24	Drop from 0.0001 to 0.00001 in steps of 0.0000125

Part 1.1.2: Results and Observations

A graph for the model training history of the final chosen model is presented on the next page.



Graph 1: Training history of the Learning-5 model (which I selected as my “final” best model for comparison with Ying Jie’s results).

From this graph I observe the following:

- Training prior to epoch 12 was chaotic, with the validation loss changing chaotically, as if the steps taken during gradient descent are bouncing around the optimum point.
- Training from 12 to 18 epochs and from 18 to 24 epochs was much more consistent, with an overall downward trend.

These observations influenced my choice to implement the schedulers used in Learning-6 and Learning-7, though they did not quite perform as well as the simple reduction of Learning-5. It might be possible to achieve superior results using a carefully tailored scheduler, however the simple reduction scheduler of the Learning-5 model achieved comparably good results for much less effort.

Part 1.2: Work by Ying Jie Mei

According to the previous report, I have made the following changes:

-compression factor from 0.5 to 0.75
-compression factor from 0.5 to 1.00

Table 5

Model	Based on	Changes	Loss	Accuracy	F1 Macro
Adam	Stage IV Model	Optimizer	0.3813	0.8574	0.8589
Compression	Adam	Changes compression from 0.5 to 0.75	0.4153	0.8525	0.8552
Compression	Adam	Changes compression from 0.5 to 1.00	0.4417	0.8418	0.8443

These models trained is modified based on the stage IV model which we submitted on gradescope, by changing the compression factor from 0.5 which was what we used during stage IV of the project to 0.75 or 1.00. We want to make sure by changing the compression factor, we wouldn't be missing out if this would actually change the results leading to a better accuracy and f1 score model. The result is disappointing and it didn't lead to a better model, but instead as we increase the compression factor the model's accuracy and f1 score decreases, which is another failure. Since that didn't go well, we decide to train the final model without the compression factor.

Part 2: Model Architecture

The architecture of the model we will submit for Stage V is largely identical to the architecture of our model for Stage IV. Since our Stage V final product focuses on features that do not show up well on an architecture diagram, this part of the report will focus more on these less visible changes.

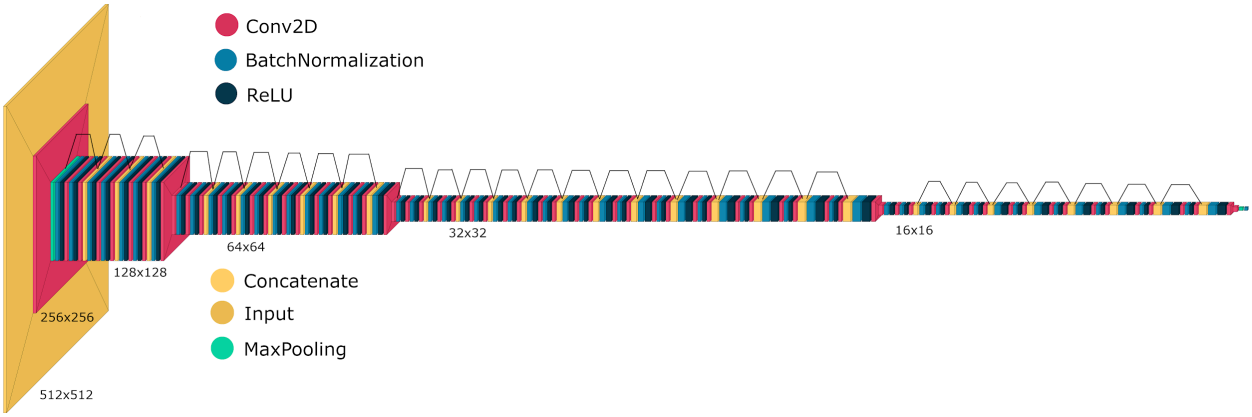


Figure 1: Architecture of the chosen model, generated using the visualkeras package for Python and edited using GIMP and Paint. Black lines connecting different layers are skip connections. A full-resolution version of this image can be found here.

We chose to continue with this model, which is largely identical to the “Learning-5” model from Ze Hong’s section but trained for 48 epochs, after comparing findings and recognizing that the compression factor changes did not seem to provide a benefit.

The chosen model is optimized using Adam with these hyperparameters:

- Initial learning rate: 0.01
- Learning rate scheduler: -90% lr value at 50% and 75% of epochs elapsed
- ClipNorm: 1.0

Additionally, the chosen model itself uses these hyperparameters:

- Growth Rate: 32 filters
- Compression Factor: 0.5
- Activation function: ReLU
- Metrics: Accuracy, F1 Macro
- Loss: Categorical Cross-entropy
- Epochs trained: 16
- Initial Learning Rate (lr): 0.1
- Learning Rate schedule: Reduce lr by 90% at 8 and 12 epochs

The architecture of this model is described in the following tables, taken from the Stage IV report and updated to reflect our changes.

Table 6: Residual Block architecture. Note that x,y,filters in the “Output shape” column represent dynamic dimensions, since the input to each residual block may have a different number of filters with different x,y dimensions.

Layers	Parameters and Notes	Output shape
Input	Residual Block begins here	x,y,filters
BatchNormalization		x,y,filters
ReLU		x,y,filters
Conv2D	128 filters, 1x1 kernels, stride 1, same padding	x,y,128
BatchNormalization		x,y,128
ReLU		x,y,128
Conv2D	32 filters, 3x3 kernels, stride 1, same padding, output=b	x,y,32
Concatenate	Skip connection, links input with output	x,y,filters+32

Residual blocks are one of the complex components of this model. They consist of a pair of BatchNorm-ReLU-Conv2D stacks, one after the other, with a skip connection that concatenates the input with the output.

Table 7: Transition Block architecture. Note that the “Output shape” column describes dynamic dimensions in the same manner as in Table 6.

Layers	Parameters and Notes	Output shape
Input	Transition layer begins here	x,y,filters
BatchNormalization		x,y,filters
ReLU		x,y,filters
Conv2D	filters/2 filters, 3x3 kernel, stride 1, same padding	x,y,filters*0.5
AveragePooling2D	2x2 pooling, stride 2, same padding	(x,y,filters)*0.5

Transition Blocks are another complex component. They consist of a BatchNormalization layer and a ReLU activation followed by a Conv2D layer and an AveragePooling2D layer. The Conv2D and AveragePooling2D layers serve to down-size the filters by an amount proportional to the Compression Factor.

Table 8: DenseNet model architecture. Note that each “Residual block [number]” in the Layers column describes a number of residual blocks connected end to end. The purpose is to convey the same information as listing each layer individually without requiring hundreds of rows of info.

Layers	Parameters and Notes	Output shape
Input		512,512,3
Conv2D	64 filters, 7x7 kernel, stride 2, same padding, relu	256,256,64
MaxPooingl2D	3x3 pooling, stride 2, same padding	128,128,64
Residual Block		128,128,96
Residual Block		128,128,128
Residual Block		128,128,160
Transition Layer		64,64,80
Residual Block x6	Filter counts after each block are: 112, 144, 176, 208, 240, 272. (x,y) dims remain at (64,64).	64,64,272
Transition Layer		32,32,136

Residual block x12	Filter counts after each block are: 168, 200, 232, 264, 296, 328, 360, 392, 424, 456, 488, 520. (x,y) dims remain at (32,32).	32,32,520
Transition Layer		16,16,260
Residual Block x8	Filter counts after each block are:290, 324, 356, 388, 420, 458, 484, 516. (x,y) dims remain at (16,16).	16,16,516
Transition Layer	Unintentionally added to the model during Stage III. It was removed in Ze Hong's tests but not in Ying Jie's tests. It will be removed in Stage V.	8,8,258
GlobalAveragePooling2D		None,258

This model, after training for 48 epochs, achieved results superior to that of Learning-5 (trained for 24 epochs only).

Table 9: Training results for the final model.

Model	Based on	Changes	Loss	Accuracy	F1 Macro
Final	Learning-5	Trained for 48 epochs	0.4210	0.8670	0.8674