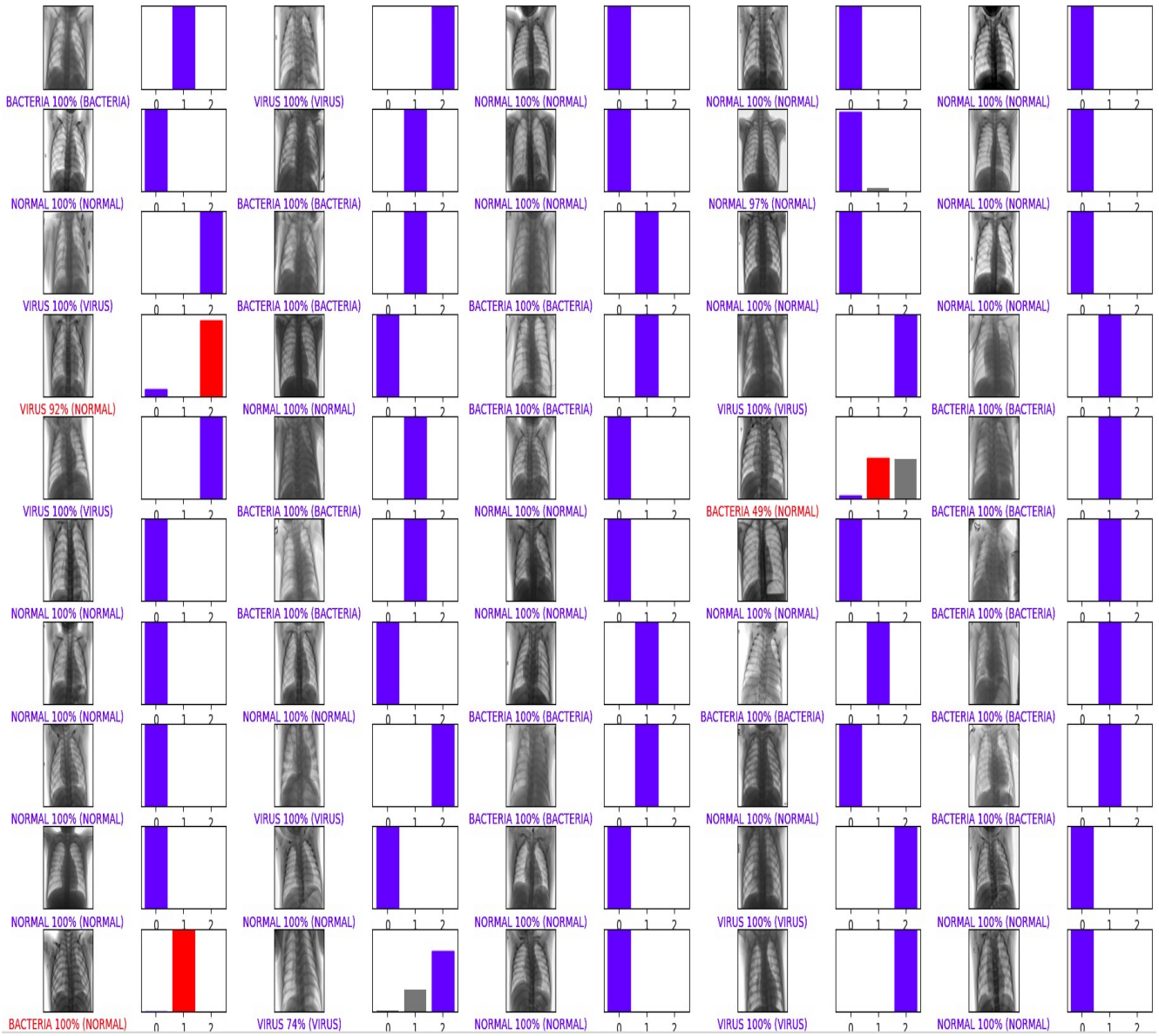
Medical Images Recognition

Here is our results of our model and A.I trained about the medical images recognition.
This study deals with the recognition of medical imaging of lung in 3 differents states:

Normal, Bacteria, Virus



Tested Models

In this part, we gonna to present the differents models tested and the graphic results with the accuracy and loss value for each one.
This work aims to illustrate the different stages of our study in order to find which model is the most efficient

The data presented are composed to : the model specification, the loss and accuracy graphic

evolution in train and test, the loss and accuracy final values in train and test.

These tests are split in 6 states : Activation, Data size, Dropout, Epoch number, RMS optimizer, Data augmentation, Layers, Data balancing, Batch size

1. Activation

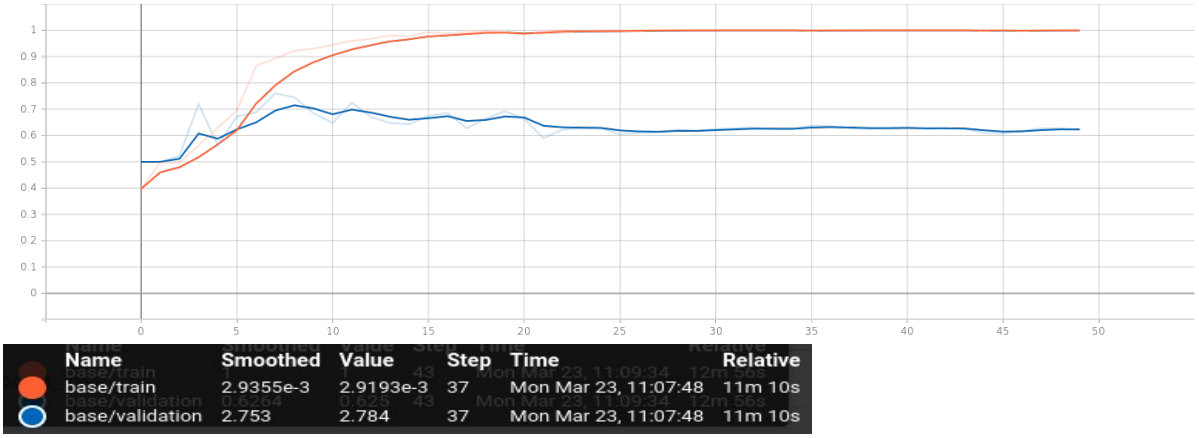
The purpose of these tests is to vary the optimization variable between “Sigmoid”, “Softmax” and “Relu” over a certain number of layers

Activation “Relu” for all layers

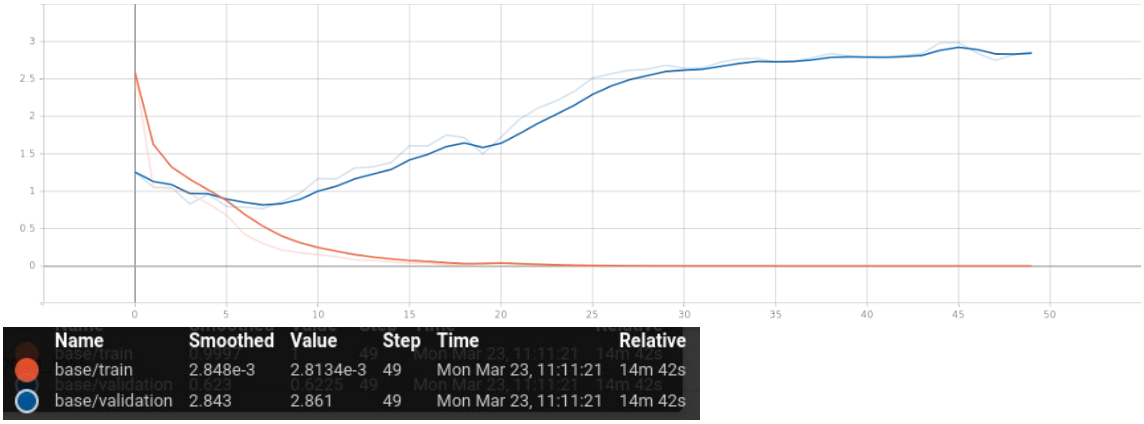
id	Model									
	$Couche_1$	$Couche_2$	$Couche_3$	$Couche_4$	$Couche_5$	$Couche_6$	$Couche_7$	$Couche_8$	$Couche_9$	Couche_10
Base	Convolution 2D : 32, 3 Activation: relu	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : relu	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : relu	Flatten	Dense: 64 Activation: relu	Dense: 3

Compilation			Fit			Data					
Optimizer	Loss	Metrics	Epoch	Final loss	Final accuracy	Batch size	Height	Width	Depth	Train images	Test images
Adam	SparseCategoricalCrossentropy	Accuracy	50	0,0028	1	400	224	224	1	398	400

- Accuracy



- Loss

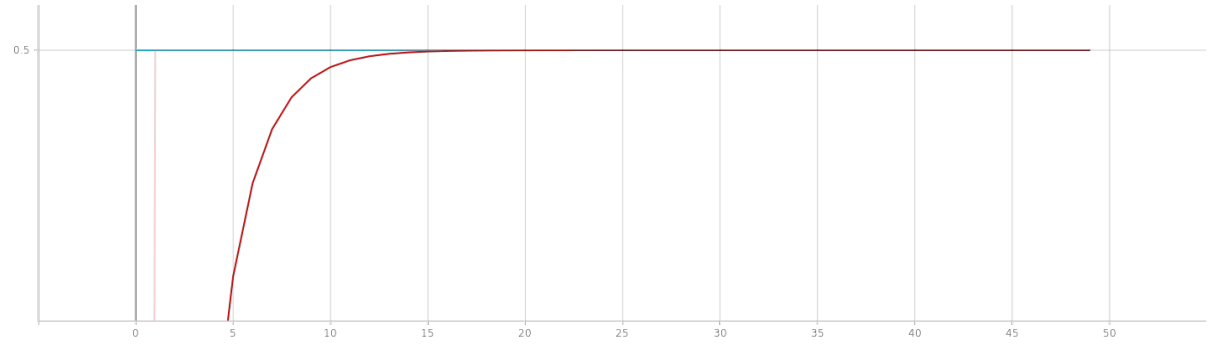


Activation “Sigmoid” for all layers

id	Model									
	$Couche_1$	$Couche_2$	$Couche_3$	$Couche_4$	$Couche_5$	$Couche_6$	$Couche_7$	$Couche_8$	$Couche_9$	Couche_10
Base	Convolution 2D : 32, 3 Activation: sigmoid	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : sigmoid	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : sigmoid	Flatten	Dense: 64 Activation: sigmoid	Dense: 3

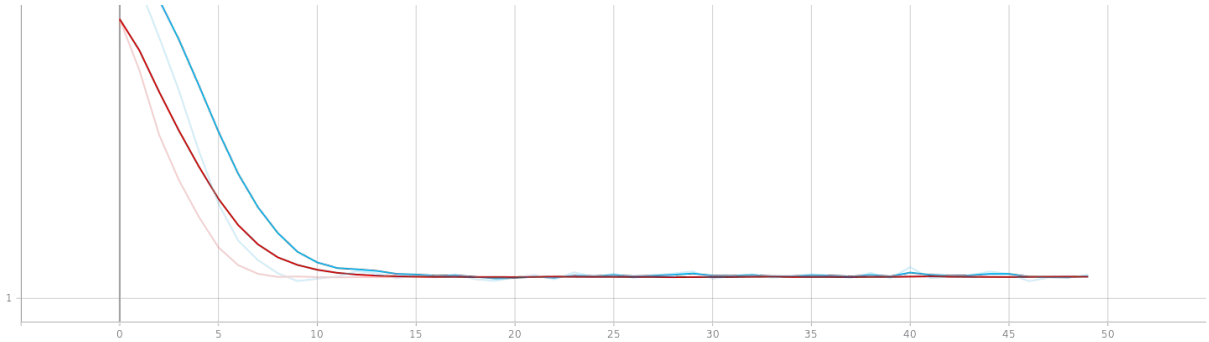
Compilation			Fit			Data					
Optimizer	Loss	Metrics	Epoch	Final loss	Final accuracy	Batch size	Height	Width	Depth	Train images	Test images
Adam	SparseCategoricalCrossentropy	Accuracy	50	1,041	0,5	400	224	224	1	398	400

- Accuracy



Name	Smoothed	Value	Step	Time	Relative
maxime_all_layers_sigmoid/train	0.5	0.5	48	Mon Mar 23, 12:33:46	15m 14s
maxime_all_layers_sigmoid/validation	0.5	0.5	48	Mon Mar 23, 12:33:46	15m 14s

- Loss



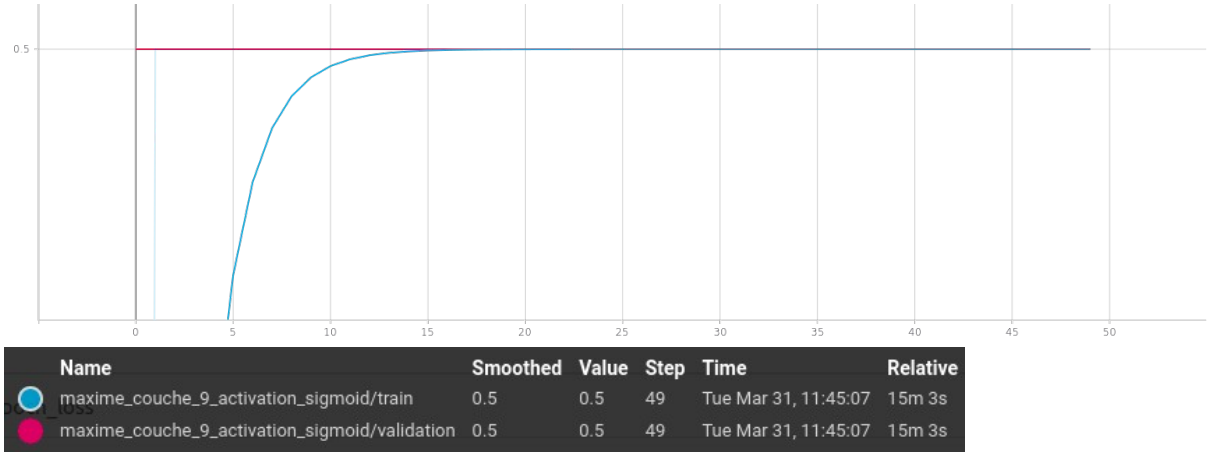
Name	Smoothed	Value	Step	Time	Relative
maxime_all_layers_sigmoid/train	1.041	1.041	49	Mon Mar 23, 12:34:05	15m 33s
maxime_all_layers_sigmoid/validation	1.042	1.045	49	Mon Mar 23, 12:34:05	15m 33s

Activation “Sigmoid” for layer 9

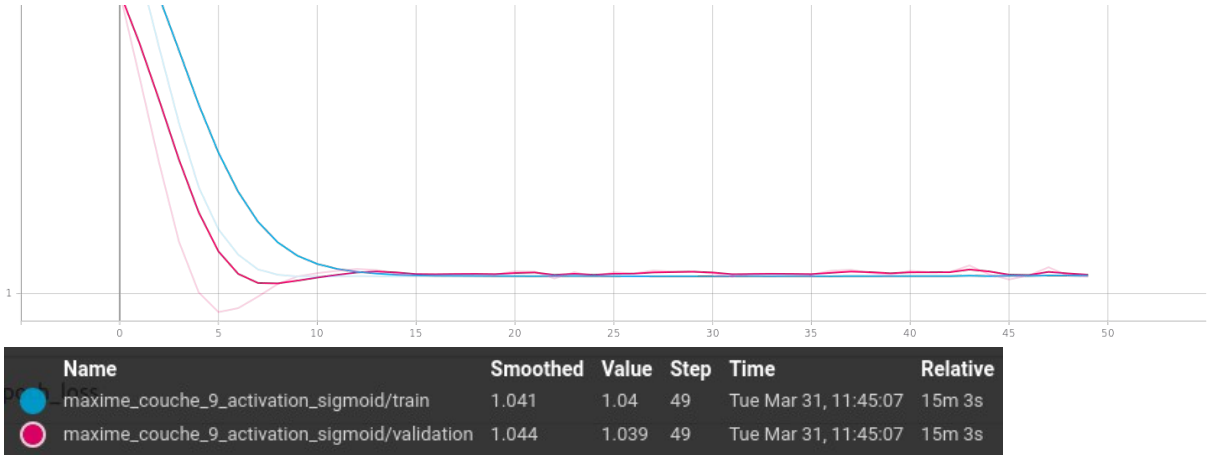
id	Model									
	$Couche_1$	$Couche_2$	$Couche_3$	$Couche_4$	$Couche_5$	$Couche_6$	$Couche_7$	$Couche_8$	$Couche_9$	Couche_10
Base	Convolution 2D : 32, 3 Activation: relu	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : relu	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : relu	Flatten	Dense: 64 Activation: sigmoid	Dense: 3

Compilation			Fit			Data					
Optimizer	Loss	Metrics	Epoch	Final loss	Final accuracy	Batch size	Height	Width	Depth	Train images	Test images
Adam	SparseCategoricalCrossentropy	Accuracy	50	1,4	0,5	400	224	224	1	398	400

- Accuracy



- Loss

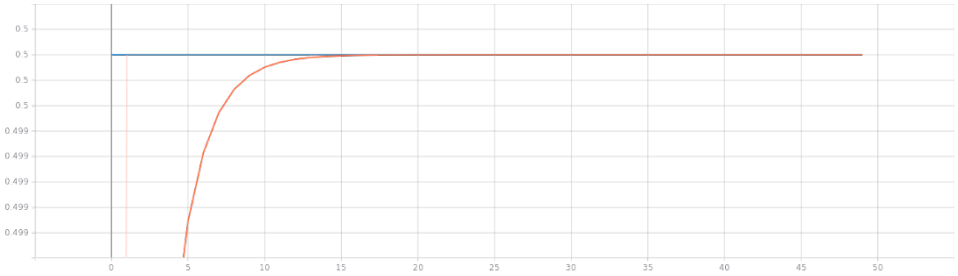


Activation “Softmax” for all layers

id	Model									
	$Couche_1$	$Couche_2$	$Couche_3$	$Couche_4$	$Couche_5$	$Couche_6$	$Couche_7$	$Couche_8$	$Couche_9$	Couche_10
Base	Convolution 2D : 32, 3 Activation: softmax	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : softmax	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : softmax	Flatten	Dense: 64 Activation: softmax	Dense: 3

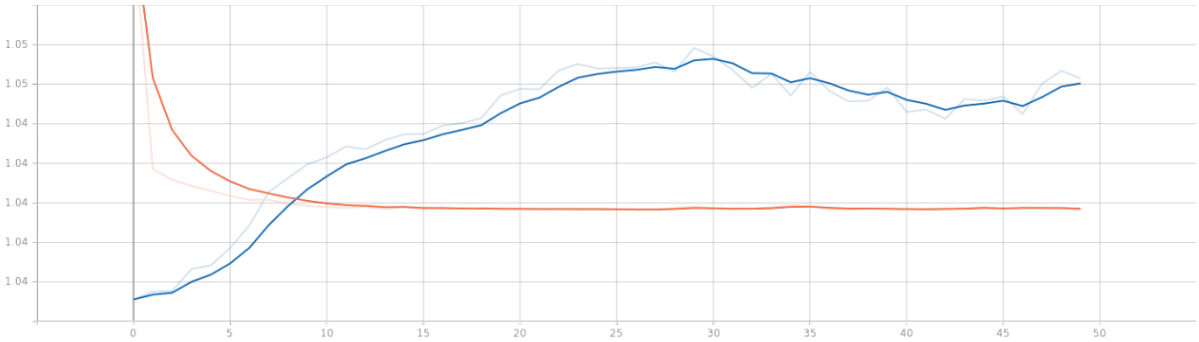
Compilation			Fit			Data					
Optimizer	Loss	Metrics	Epoch	Final loss	Final accuracy	Batch size	Height	Width	Depth	Train images	Test images
Adam	SparseCategoricalCrossentropy	Accuracy	50	1,0397	0,5	400	224	224	1	398	400

- Accuracy



Name	Smoothed	Value	Step	Time	Relative
jean_all_layers_softmax/train	0.5	0.5	33	Sun Apr 12, 11:58:36	17m 25s
jean_all_layers_softmax/validation	0.5	0.5	33	Sun Apr 12, 11:58:36	17m 25s

- Loss



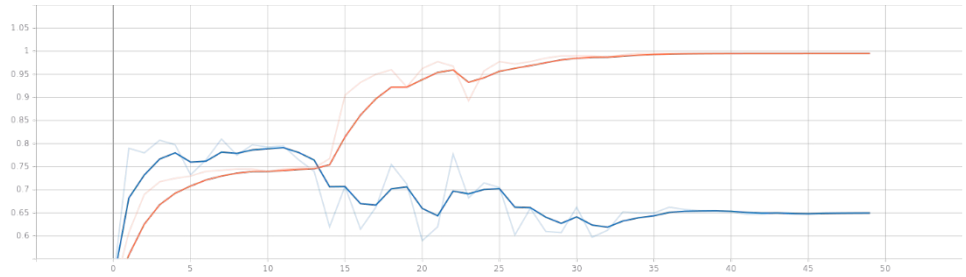
Name	Smoothed	Value	Step	Time	Relative
jean_all_layers_softmax/train	1.04	1.04	49	Sun Apr 12, 12:07:02	25m 51s
jean_all_layers_softmax/validation	1.046	1.046	49	Sun Apr 12, 12:07:02	25m 51s

Activation “Softmax” for layer 10

id	Model									
	$Couche_1$	$Couche_2$	$Couche_3$	$Couche_4$	$Couche_5$	$Couche_6$	$Couche_7$	$Couche_8$	$Couche_9$	Couche_10
Base	Convolution 2D : 32, 3 Activation: relu	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : relu	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : relu	Flatten	Dense: 64 Activation: relu	Dense: 3 Activation : softmax

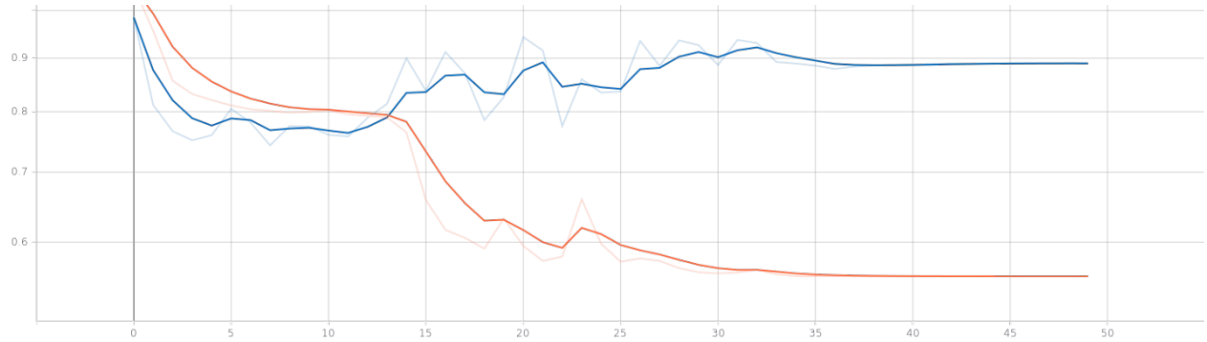
Compilation			Fit			Data					
Optimizer	Loss	Metrics	Epoch	Final loss	Final accuracy	Batch size	Height	Width	Depth	Train images	Test images
Adam	SparseCategoricalCrossentropy	Accuracy	50	0,5565	0,9950	400	224	224	1	398	400

- Accuracy



Name	Smoothed	Value	Step	Time	Relative
maxime_couche_10_softmax_activation/train	0.995	0.995	49	Tue Mar 31, 17:27:20	15m 23s
maxime_couche_10_softmax_activation/validation	0.6498	0.65	49	Tue Mar 31, 17:27:20	15m 23s

- Loss



Name	Smoothed	Value	Step	Time	Relative
maxime_couche_10_softmax_activation/train	0.5565	0.5565	49	Tue Mar 31, 17:27:20	15m 23s
maxime_couche_10_softmax_activation/validation	0.8899	0.8899	49	Tue Mar 31, 17:27:20	15m 23s

In conclusion we can see the activation by ”Relu” and ”Softmax” for layer 10 allow an optimisation of the validation accuracy. The others tests just show us a stagnation for our accuracy.

2. Data Size

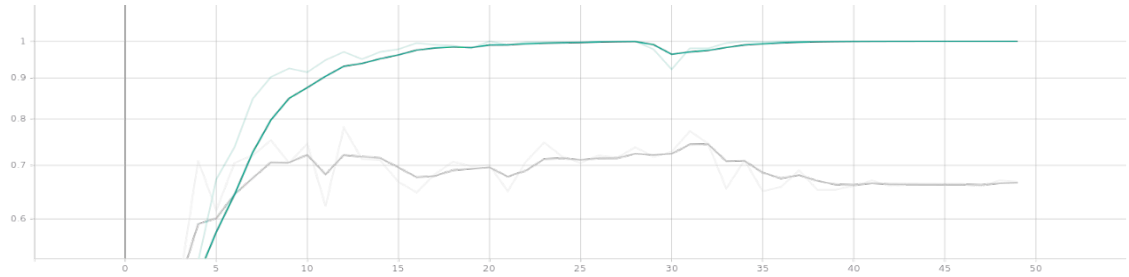
The purpose of these tests is to vary the data size and watch the evolution of our results.

Data Size 150

id	Model									
	$Couche_1$	$Couche_2$	$Couche_3$	$Couche_4$	$Couche_5$	$Couche_6$	$Couche_7$	$Couche_8$	$Couche_9$	Couche_10
Base	Convolution 2D : 32, 3 Activation: relu	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : relu	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : relu	Flatten	Dense: 64 Activation: relu	Dense: 3

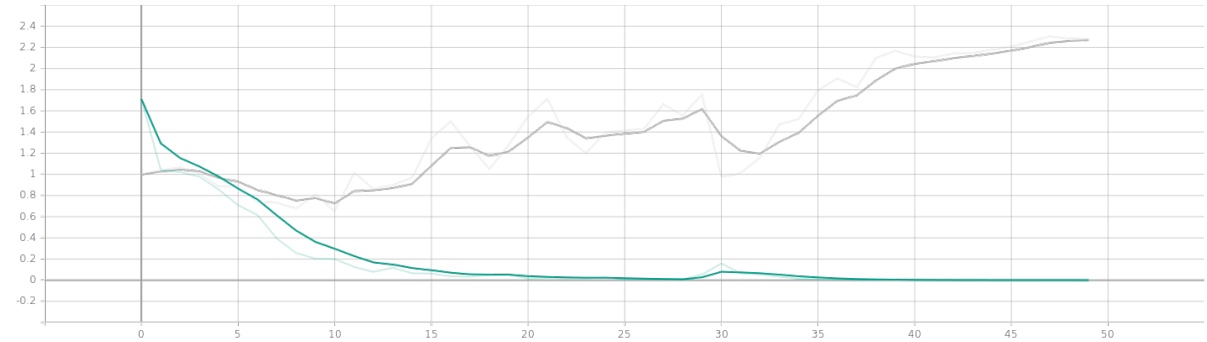
Compilation			Fit			Data					
Optimizer	Loss	Metrics	Epoch	Final loss	Final accuracy	Batch size	Height	Width	Depth	Train images	Test images
Adam	SparseCategoricalCrossentropy	Accuracy	50	0,00037	1	400	150	150	1	398	400

- Accuracy



Name	Smoothed	Value	Step	Time	Relative
jean_data_size_150/train	1	1	49	Tue Mar 31, 11:45:02	7m 3s
jean_data_size_150/validation	0.666	0.6675	49	Tue Mar 31, 11:45:02	7m 3s

- Loss



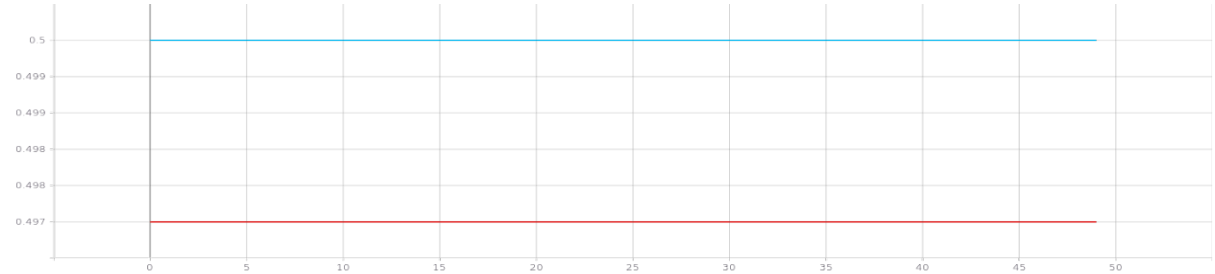
Name	Smoothed	Value	Step	Time	Relative
jean_data_size_150/train	4.8247e-4	3.7604e-4	49	Tue Mar 31, 11:45:02	7m 3s
jean_data_size_150/validation	2.269	2.284	49	Tue Mar 31, 11:45:02	7m 3s

Data Size 150 with “Sigmoid” activation

id	Model									
	$Couche_1$	$Couche_2$	$Couche_3$	$Couche_4$	$Couche_5$	$Couche_6$	$Couche_7$	$Couche_8$	$Couche_9$	Couche_10
Base	Convolution 2D : 32, 3 Activation: sigmoid	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : sigmoid	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : sigmoid	Flatten	Dense: 64 Activation: sigmoid	Dense: 3

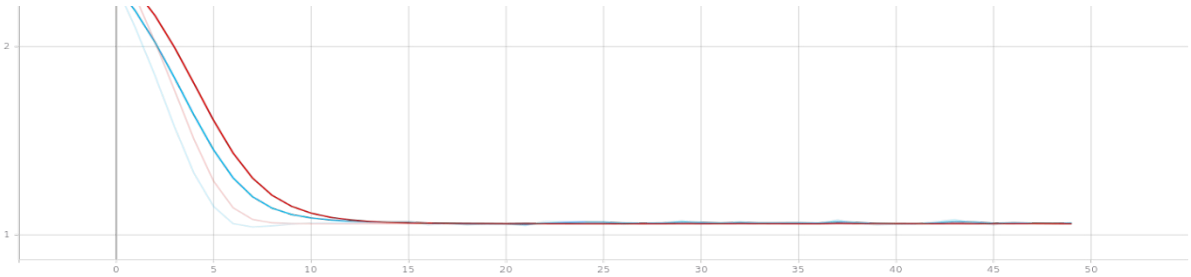
Compilation			Fit			Data					
Optimizer	Loss	Metrics	Epoch	Final loss	Final accuracy	Batch size	Height	Width	Depth	Train images	Test images
Adam	SparseCategoricalCrossentropy	Accuracy	50	1,042	0,49	400	150	150	1	398	400

- Accuracy



Name	Smoothed	Value	Step	Time	Relative
jean_data_size_150_sigmoid/train	0.4975	0.4975	49	Tue Mar 31, 13:02:37	7m 11s
jean_data_size_150_sigmoid/validation	0.5	0.5	49	Tue Mar 31, 13:02:37	7m 11s

- Loss



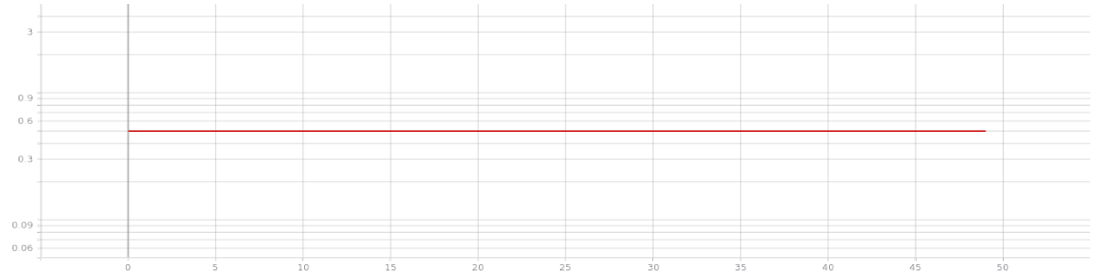
Name	Smoothed	Value	Step	Time	Relative
jean_data_size_150_sigmoid/train	1.042	1.042	49	Tue Mar 31, 13:02:37	7m 11s
jean_data_size_150_sigmoid/validation	1.044	1.045	49	Tue Mar 31, 13:02:37	7m 11s

Data Size 150 with “Softmax” activation

id	Model									
	$Couche_1$	$Couche_2$	$Couche_3$	$Couche_4$	$Couche_5$	$Couche_6$	$Couche_7$	$Couche_8$	$Couche_9$	Couche_10
Base	Convolution 2D : 32, 3 Activation: softmax	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : softmax	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : softmax	Flatten	Dense: 64 Activation: softmax	Dense: 3

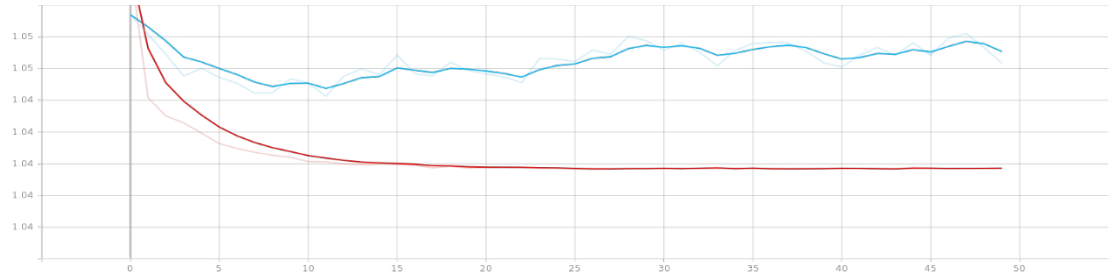
Compilation			Fit			Data					
Optimizer	Loss	Metrics	Epoch	Final loss	Final accuracy	Batch size	Height	Width	Depth	Train images	Test images
Adam	SparseCategoricalCrossentropy	Accuracy	50	1,0463	0,5	400	150	150	1	398	400

- Accuracy



Name	Smoothed	Value	Step	Time	Relative
jean_data_size_150_softmax/train	0.5	0.5	49	Sun Apr 12, 12:21:14	11m 23s
jean_data_size_150_softmax/validation	0.5	0.5	49	Sun Apr 12, 12:21:14	11m 23s

- Loss



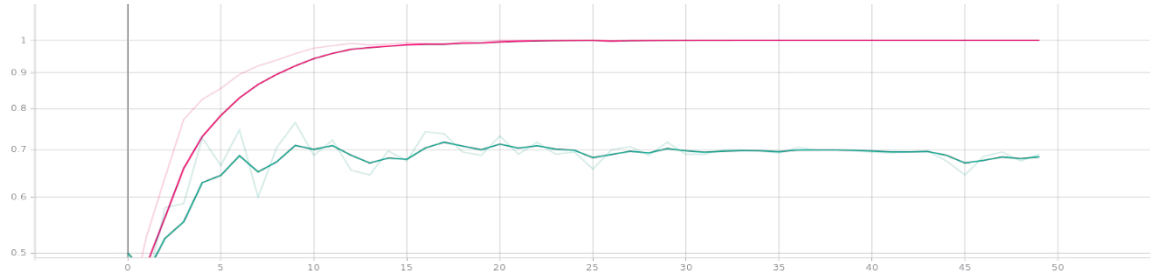
Name	Smoothed	Value	Step	Time	Relative
jean_data_size_150_softmax/train	0.5	0.5	49	Sun Apr 12, 12:21:14	11m 23s
jean_data_size_150_softmax/validation	0.5	0.5	49	Sun Apr 12, 12:21:14	11m 23s

Data Size 200

id	Model									
	$Couche_1$	$Couche_2$	$Couche_3$	$Couche_4$	$Couche_5$	$Couche_6$	$Couche_7$	$Couche_8$	$Couche_9$	Couche_10
Base	Convolution 2D : 32, 3 Activation: relu	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : relu	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : relu	Flatten	Dense: 64 Activation: relu	Dense: 3

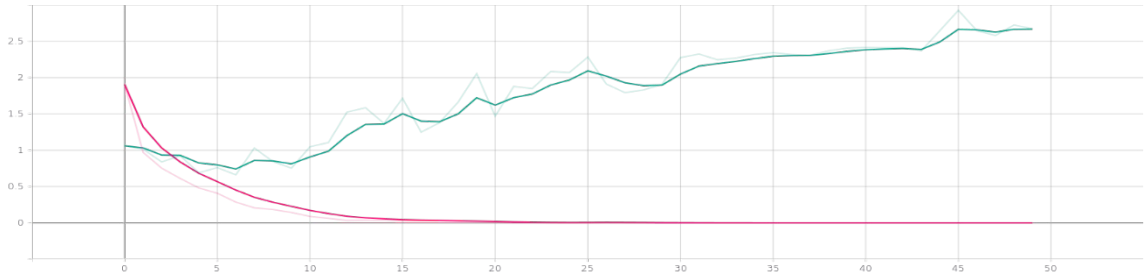
Compilation			Fit			Data					
Optimizer	Loss	Metrics	Epoch	Final loss	Final accuracy	Batch size	Height	Width	Depth	Train images	Test images
Adam	SparseCategoricalCrossentropy	Accuracy	50	0,000083	1	400	200	200	1	398	400

- Accuracy



Name	Smoothed	Value	Step	Time	Relative
jean_data_size_200/train	1	1	49	Tue Mar 31, 14:28:25	15m 12s
jean_data_size_200/validation	0.6842	0.69	49	Tue Mar 31, 14:28:25	15m 12s

- Loss



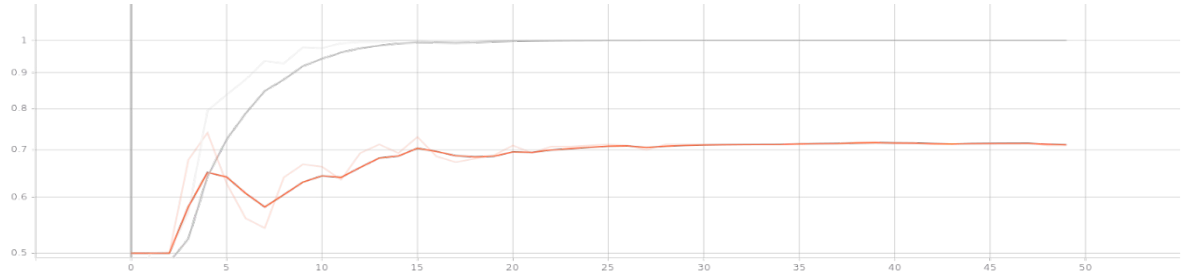
Name	Smoothed	Value	Step	Time	Relative
jean_data_size_200/train	1.1606e-4	8.3417e-5	49	Tue Mar 31, 14:28:25	15m 12s
jean_data_size_200/validation	2.668	2.673	49	Tue Mar 31, 14:28:25	15m 12s

Data Size 250

id	Model									
	$Couche_1$	$Couche_2$	$Couche_3$	$Couche_4$	$Couche_5$	$Couche_6$	$Couche_7$	$Couche_8$	$Couche_9$	Couche_10
Base	Convolution 2D : 32, 3 Activation: relu	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : relu	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : relu	Flatten	Dense: 64 Activation: relu	Dense: 3

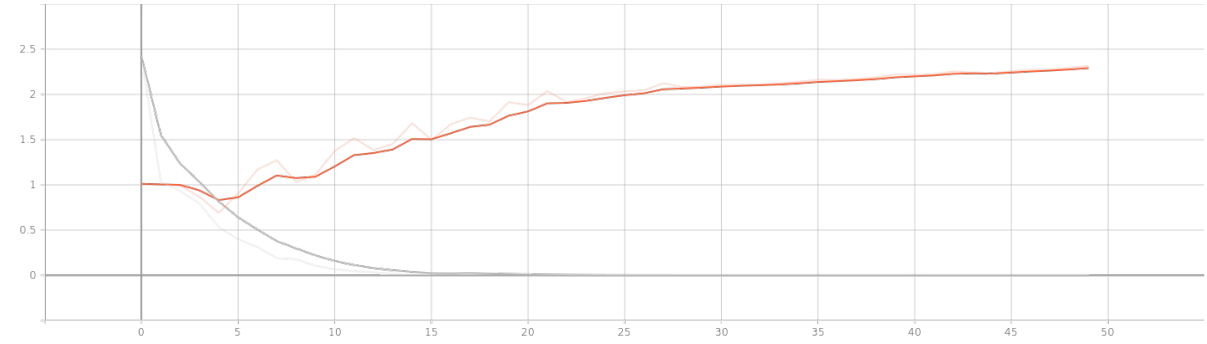
Compilation			Fit			Data					
Optimizer	Loss	Metrics	Epoch	Final loss	Final accuracy	Batch size	Height	Width	Depth	Train images	Test images
Adam	SparseCategoricalCrossentropy	Accuracy	50	0,000076	1	400	250	250	1	398	400

- Accuracy



Name	Smoothed	Value	Step	Time	Relative
jean_data_size_250/train	1	1	49	Tue Mar 31, 14:56:21	22m 29s
jean_data_size_250/validation	0.7117	0.71	49	Tue Mar 31, 14:56:21	22m 29s

- Loss



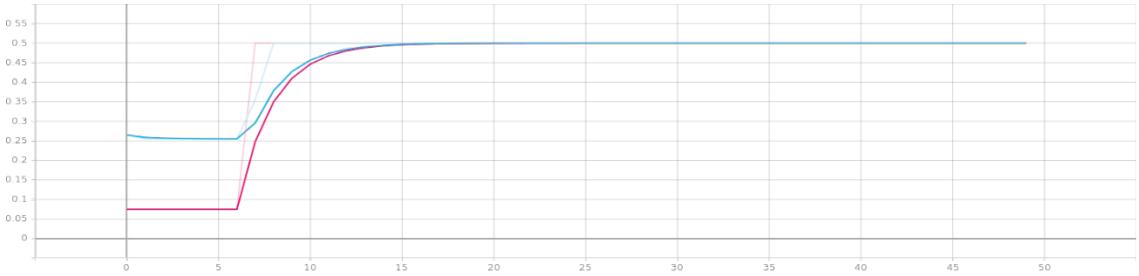
Name	Smoothed	Value	Step	Time	Relative
jean_data_size_250/train	8.0085e-5	7.6441e-5	49	Tue Mar 31, 14:56:21	22m 29s
jean_data_size_250/validation	2.292	2.316	49	Tue Mar 31, 14:56:21	22m 29s

Data Size 250 with “Sigmoid” activation

id	Model									
	$Couche_1$	$Couche_2$	$Couche_3$	$Couche_4$	$Couche_5$	$Couche_6$	$Couche_7$	$Couche_8$	$Couche_9$	Couche_10
Base	Convolution 2D : 32, 3 Activation: sigmoid	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : sigmoid	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : sigmoid	Flatten	Dense: 64 Activation: sigmoid	Dense: 3

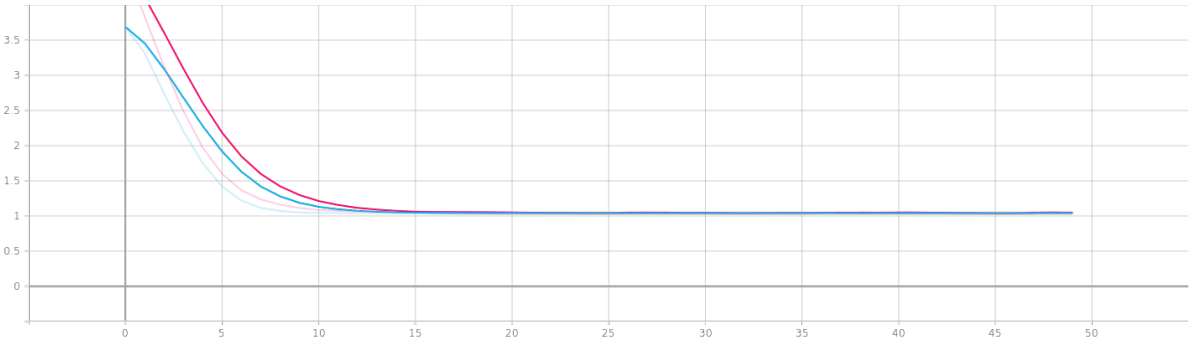
Compilation			Fit			Data					
Optimizer	Loss	Metrics	Epoch	Final loss	Final accuracy	Batch size	Height	Width	Depth	Train images	Test images
Adam	SparseCategoricalCrossentropy	Accuracy	50	1,0407	0,5	400	250	250	1	398	400

- Accuracy



Name	Smoothed	Value	Step	Time	Relative
jean_data_size_250_sigmoid/train	0.5	0.5	49	Tue Mar 31, 16:34:57	19m 14s
jean_data_size_250_sigmoid/validation	0.5	0.5	49	Tue Mar 31, 16:34:57	19m 14s

- Loss



Name	Smoothed	Value	Step	Time	Relative
jean_data_size_250_sigmoid/train	1.041	1.041	49	Tue Mar 31, 16:34:57	19m 14s
jean_data_size_250_sigmoid/validation	1.049	1.047	49	Tue Mar 31, 16:34:57	19m 14s

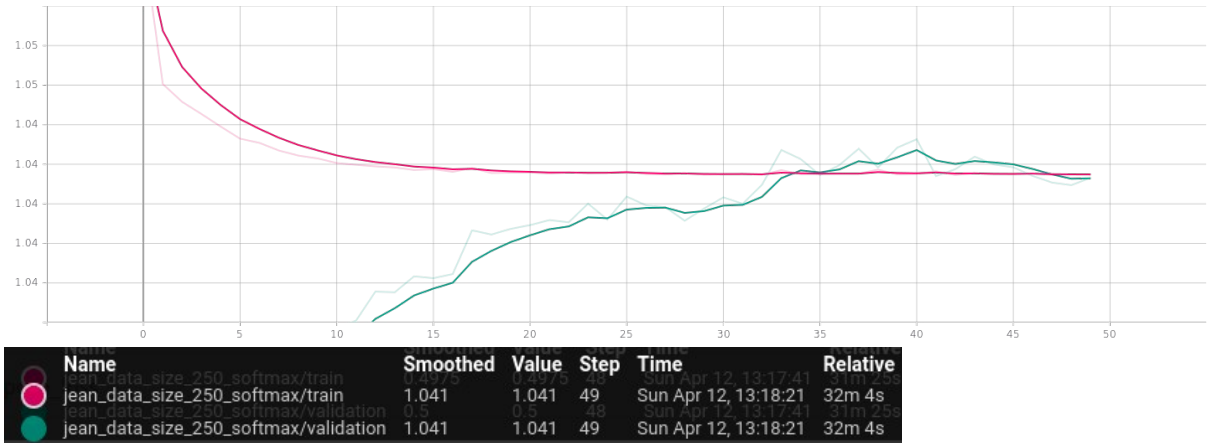
Data Size 250 with “Softmax” activation

id	Model									
	$Couche_1$	$Couche_2$	$Couche_3$	$Couche_4$	$Couche_5$	$Couche_6$	$Couche_7$	$Couche_8$	$Couche_9$	Couche_10
Base	Convolution 2D : 32, 3 Activation: softmax	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : softmax	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : softmax	Flatten	Dense: 64 Activation: softmax	Dense: 3

Compilation			Fit			Data					
Optimizer	Loss	Metrics	Epoch	Final loss	Final accuracy	Batch size	Height	Width	Depth	Train images	Test images
Adam	SparseCategoricalCrossentropy	Accuracy	50	1,0415	0,4975	400	250	250	1	398	400

- Accuracy

- Loss

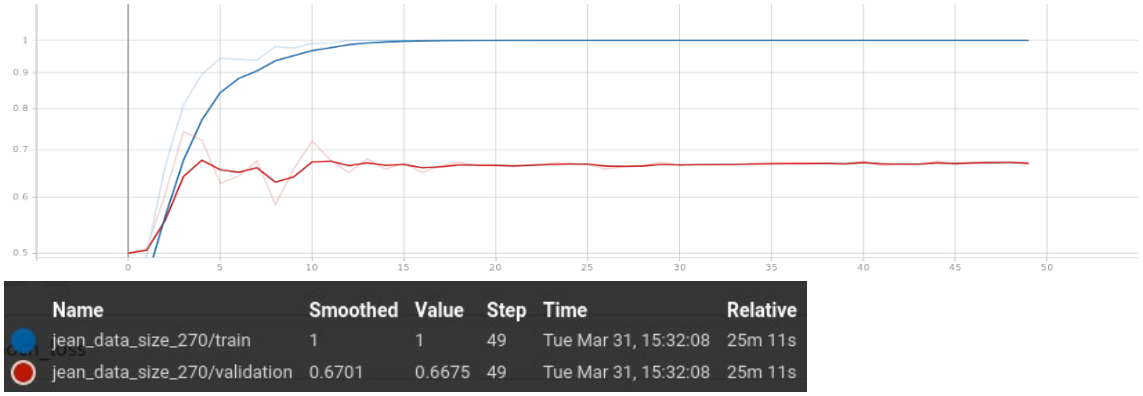


Data Size 270

id	Model									
	$Couche_1$	$Couche_2$	$Couche_3$	$Couche_4$	$Couche_5$	$Couche_6$	$Couche_7$	$Couche_8$	$Couche_9$	Couche_10
Base	Convolution 2D : 32, 3 Activation: relu	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : relu	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : relu	Flatten	Dense: 64 Activation: relu	Dense: 3

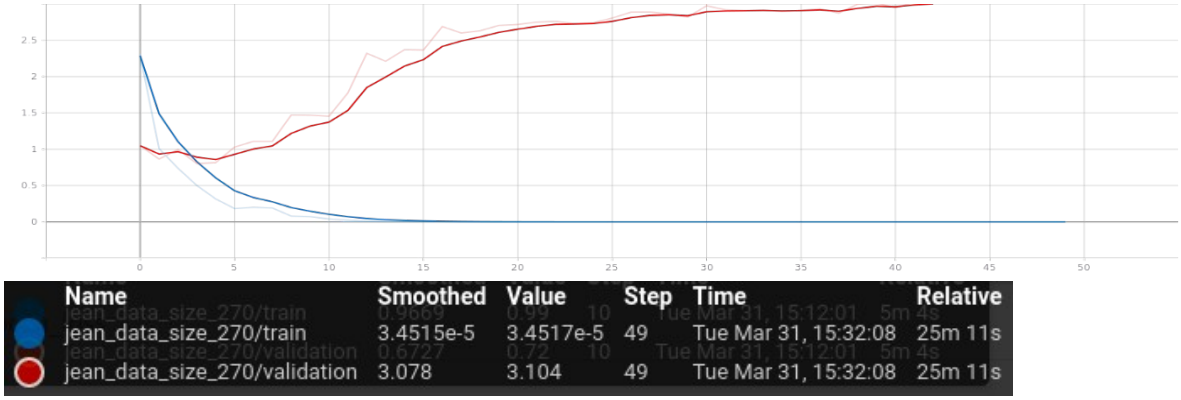
Compilation			Fit			Data					
Optimizer	Loss	Metrics	Epoch	Final loss	Final accuracy	Batch size	Height	Width	Depth	Train images	Test images
Adam	SparseCategoricalCrossentropy	Accuracy	50	0,0000345	1	400	270	270	1	398	400

- Accuracy



Name	Smoothed	Value	Step	Time	Relative
jean_data_size_270/train	1	1	49	Tue Mar 31, 15:32:08	25m 11s
jean_data_size_270/validation	0.6701	0.6675	49	Tue Mar 31, 15:32:08	25m 11s

- Loss



Name	Smoothed	Value	Step	Time	Relative
jean_data_size_270/train	3.4515e-5	3.4517e-5	49	Tue Mar 31, 15:32:08	25m 11s
jean_data_size_270/validation	3.078	3.104	49	Tue Mar 31, 15:32:08	25m 11s

In conclusion we can a data size of 250 allow the best optimisation of validation accuracy.

3. Dropout

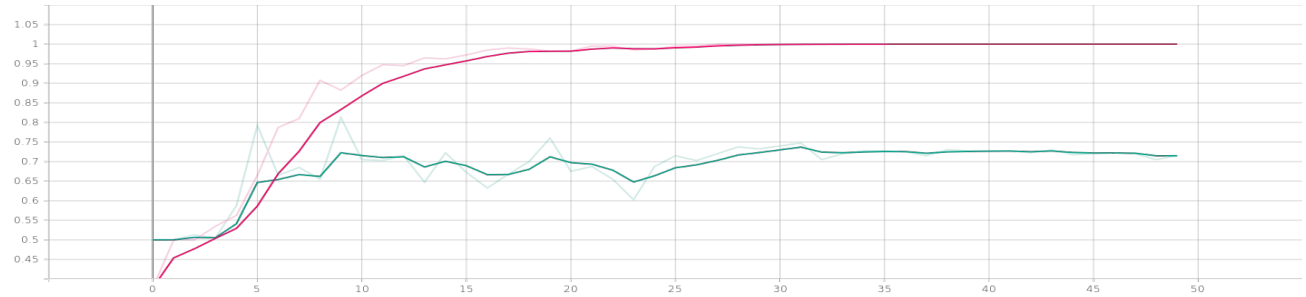
The purpose of these tests is to apply a dropout with a certain value in our model.

Dropout of 0.4

id	Model									
	$Couche_1$	$Couche_2$	$Couche_3$	$Couche_4$	$Couche_5$	$Couche_6$	$Couche_7$	$Couche_8$	$Couche_9$	Couche_10
Base	Convolution 2D : 32, 3 Activation: relu	Max pooling 2D: 2	Drop out: 0,4	Convolution 2D: 64, 3 Activation : relu	Max pooling 2D: 2	Drop out: 0,4	Convolution 2D: 64, 3 Activation : relu	Flatten	Dense: 64 Activation: relu	Dense: 3

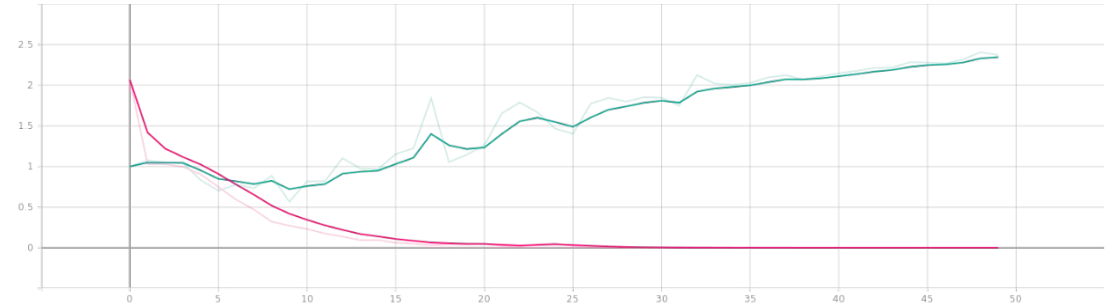
Compilation			Fit			Data					
Optimizer	Loss	Metrics	Epoch	Final loss	Final accuracy	Batch size	Height	Width	Depth	Train images	Test images
Adam	SparseCategoricalCrossentropy	Accuracy	50	0,0001	1	400	224	224	1	398	400

- Accuracy



Name	Smoothed	Value	Step	Time	Relative
maxime_dropout_0.4/train	0.9925	0.995	26	Mon Mar 23, 14:35:18	7m 59s
maxime_dropout_0.4/validation	0.6915	0.7025	26	Mon Mar 23, 14:35:18	7m 59s

- Loss



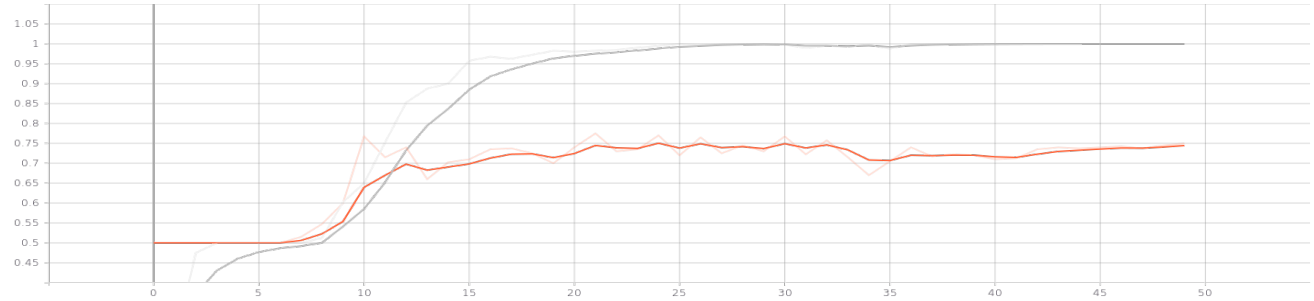
Name	Smoothed	Value	Step	Time	Relative
maxime_dropout_0.4/train	1.4958e-4	1.2531e-4	49	Mon Mar 23, 14:42:19	15m 0s
maxime_dropout_0.4/validation	2.347	2.371	49	Mon Mar 23, 14:42:19	15m 0s

Dropout of 0.6

id	Model									
	$Couche_1$	$Couche_2$	$Couche_3$	$Couche_4$	$Couche_5$	$Couche_6$	$Couche_7$	$Couche_8$	$Couche_9$	Couche_10
Base	Convolution 2D : 32, 3 Activation: relu	Max pooling 2D: 2	Drop out: 0,6	Convolution 2D: 64, 3 Activation : relu	Max pooling 2D: 2	Drop out: 0,6	Convolution 2D: 64, 3 Activation : relu	Flatten	Dense: 64 Activation: relu	Dense: 3

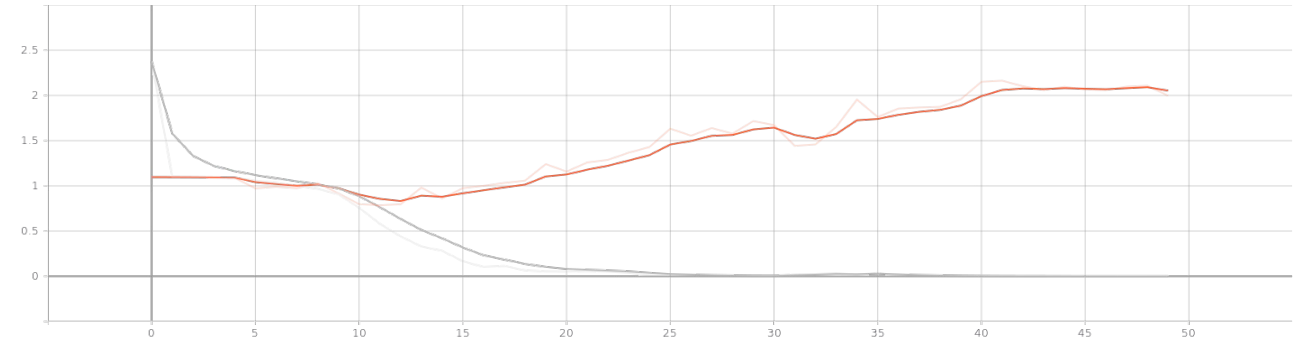
Compilation			Fit			Data					
Optimizer	Loss	Metrics	Epoch	Final loss	Final accuracy	Batch size	Height	Width	Depth	Train images	Test images
Adam	SparseCategoricalCrossentropy	Accuracy	50	0,0015	1	400	224	224	1	398	400

- Accuracy



Name	Smoothed	Value	Step	Time	Relative
maxime_dropout_0.6/train	1	1	49	Mon Mar 23, 15:05:06	15m 5s
maxime_dropout_0.6/validation	0.7445	0.75	49	Mon Mar 23, 15:05:06	15m 5s

- Loss



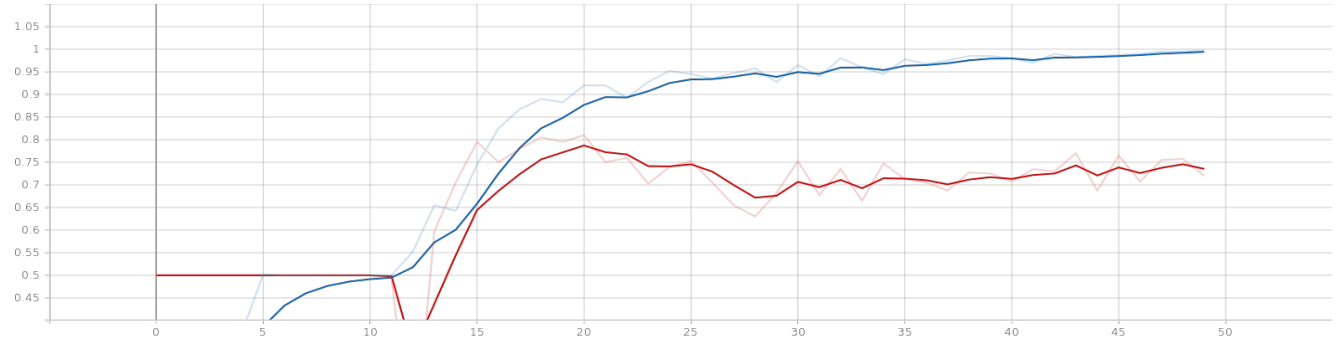
Name	Smoothed	Value	Step	Time	Relative
maxime_dropout_0.6/train	1	1	49	Mon Mar 23, 15:05:06	15m 5s
maxime_dropout_0.6/validation	0.7445	0.75	49	Mon Mar 23, 15:05:06	15m 5s

Dropout of 0.8

id	Model									
	$Couche_1$	$Couche_2$	$Couche_3$	$Couche_4$	$Couche_5$	$Couche_6$	$Couche_7$	$Couche_8$	$Couche_9$	Couche_10
Base	Convolution 2D : 32, 3 Activation: relu	Max pooling 2D: 2	Drop out: 0,8	Convolution 2D: 64, 3 Activation : relu	Max pooling 2D: 2	Drop out: 0,8	Convolution 2D: 64, 3 Activation : relu	Flatten	Dense: 64 Activation: relu	Dense: 3

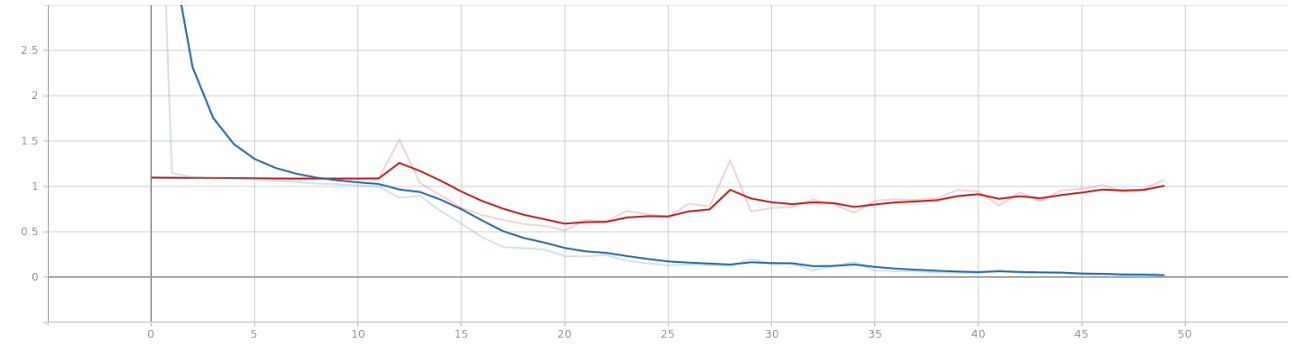
Compilation			Fit			Data					
Optimizer	Loss	Metrics	Epoch	Final loss	Final accuracy	Batch size	Height	Width	Depth	Train images	Test images
Adam	SparseCategoricalCrossentropy	Accuracy	50	0,0139	0,9975	400	224	224	1	398	400

- Accuracy



Name	Smoothed	Value	Step	Time	Relative
maxime_dropout_0.8/train	0.9943	0.9975	49	Mon Mar 23, 15:46:22	14m 36s
maxime_dropout_0.8/validation	0.7354	0.72	49	Mon Mar 23, 15:46:22	14m 36s

- Loss



Name	Smoothed	Value	Step	Time	Relative
maxime_dropout_0.8/train	0.02198	0.01389	49	Mon Mar 23, 15:46:22	14m 36s
maxime_dropout_0.8/validation	1.007	1.076	49	Mon Mar 23, 15:46:22	14m 36s

We can see that the application of a Drop Out slightly improve the accuracy. But this value stagnate near to 0,7%.

4. Epochs

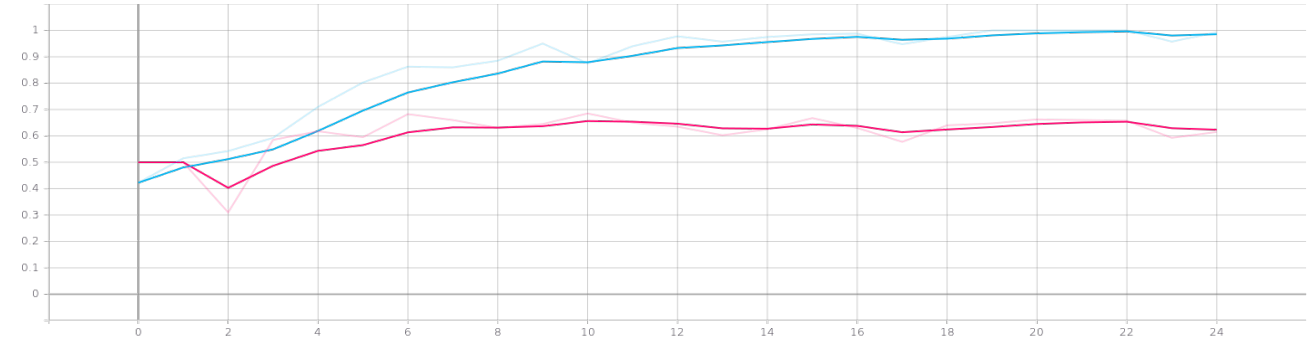
The purpose of these tests is to vary the number of epochs for the training of our model.

Epoch to 25

id	Model									
	$Couche_1$	$Couche_2$	$Couche_3$	$Couche_4$	$Couche_5$	$Couche_6$	$Couche_7$	$Couche_8$	$Couche_9$	Couche_10
Base	Convolution 2D : 32, 3 Activation: relu	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : relu	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : relu	Flatten	Dense: 64 Activation: relu	Dense: 3

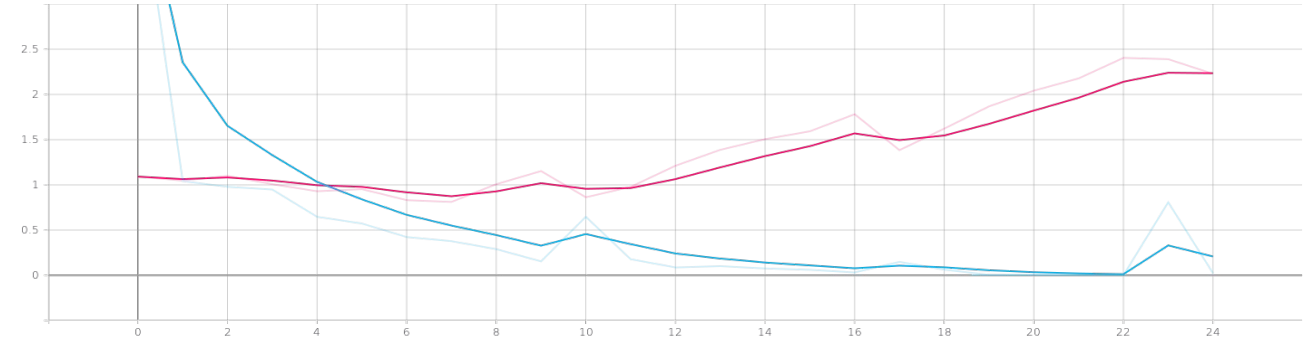
Compilation			Fit			Data					
Optimizer	Loss	Metrics	Epoch	Final loss	Final accuracy	Batch size	Height	Width	Depth	Train images	Test images
Adam	SparseCategoricalCrossentropy	Accuracy	25	0,0234	0,9925	400	224	224	1	398	400

- Accuracy



Name	Smoothed	Value	Step	Time	Relative
jean_epochs_25/train	0.9853	0.9925	24	Tue Apr 14, 17:06:12	10m 13s
jean_epochs_25/validation	0.6235	0.615	24	Tue Apr 14, 17:06:12	10m 13s

- Loss



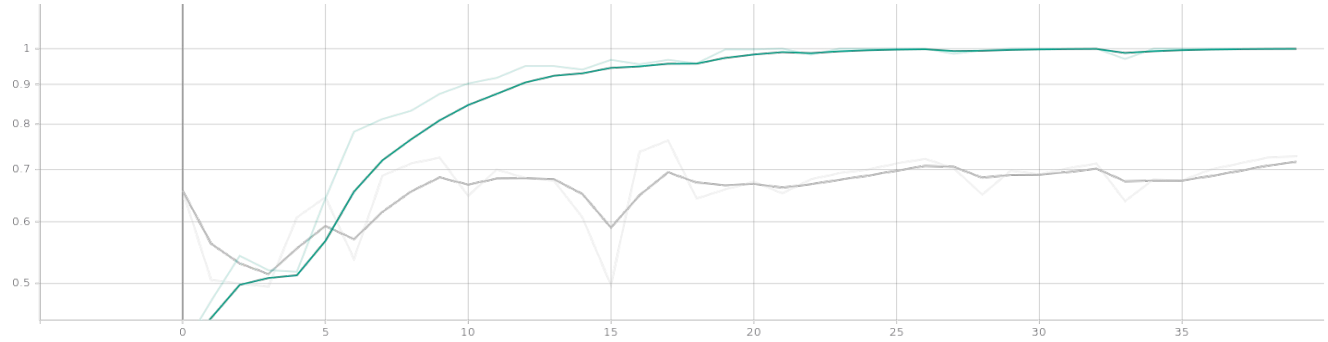
Name	Smoothed	Value	Step	Time	Relative
jean_epochs_25/train	0.8363	0.885	8	Tue Apr 14, 16:59:23	3m 23s
jean_epochs_25/validation	0.6314	0.63	8	Tue Apr 14, 16:59:23	3m 23s

Epoch to 40

id	Model									
	$Couche_1$	$Couche_2$	$Couche_3$	$Couche_4$	$Couche_5$	$Couche_6$	$Couche_7$	$Couche_8$	$Couche_9$	Couche_10
Base	Convolution 2D : 32, 3 Activation: relu	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : relu	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : relu	Flatten	Dense: 64 Activation: relu	Dense: 3

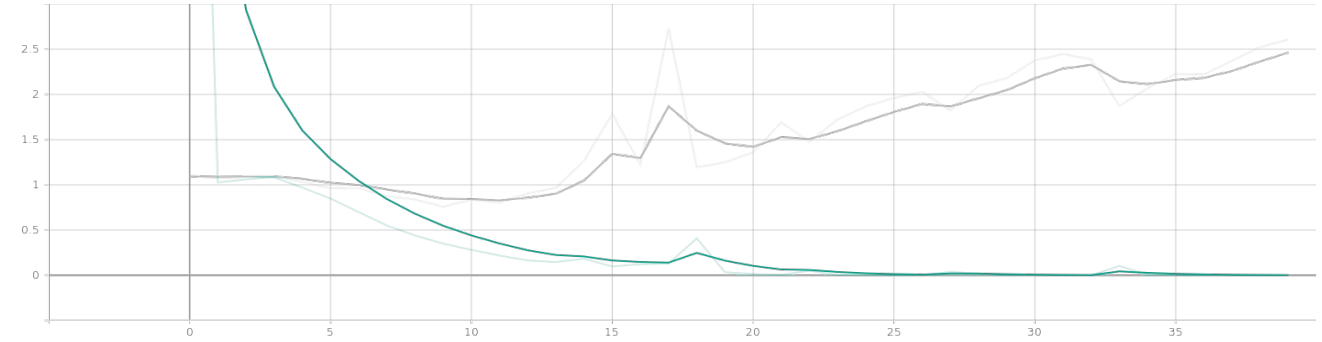
Compilation			Fit			Data					
Optimizer	Loss	Metrics	Epoch	Final loss	Final accuracy	Batch size	Height	Width	Depth	Train images	Test images
Adam	SparseCategoricalCrossentropy	Accuracy	40	0,0000136	1	400	224	224	1	398	400

- Accuracy



Name	Smoothed	Value	Step	Time	Relative
jean_epochs_40/train	0.9994	1	39	Tue Apr 14, 17:23:16	11m 35s
jean_epochs_40/validation	0.7159	0.7275	39	Tue Apr 14, 17:23:16	11m 35s

- Loss



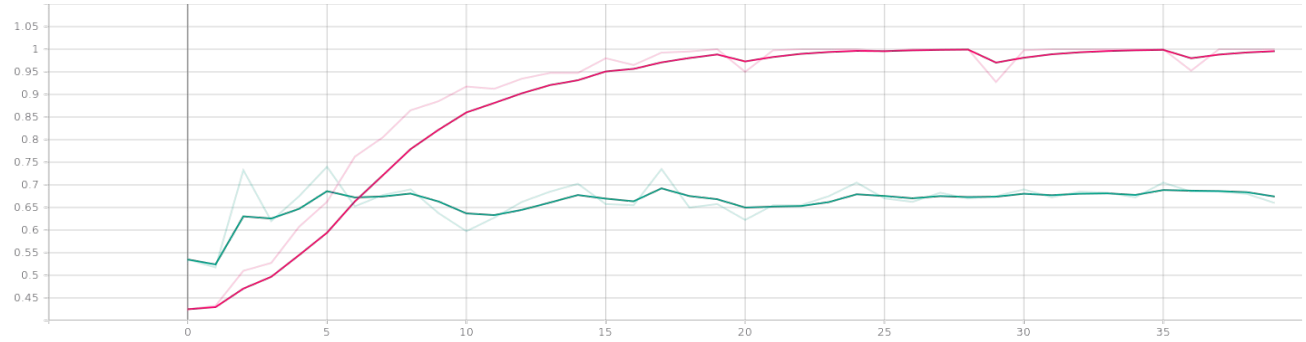
Name	Smoothed	Value	Step	Time	Relative
jean_epochs_40/train	2.2797e-3	1.3365e-5	39	Tue Apr 14, 17:23:16	11m 35s
jean_epochs_40/validation	2.463	2.61	39	Tue Apr 14, 17:23:16	11m 35s

Epoch to 40 and data size to 250

id	Model									
	$Couche_1$	$Couche_2$	$Couche_3$	$Couche_4$	$Couche_5$	$Couche_6$	$Couche_7$	$Couche_8$	$Couche_9$	Couche_10
Base	Convolution 2D : 32, 3 Activation: relu	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : relu	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : relu	Flatten	Dense: 64 Activation: relu	Dense: 3

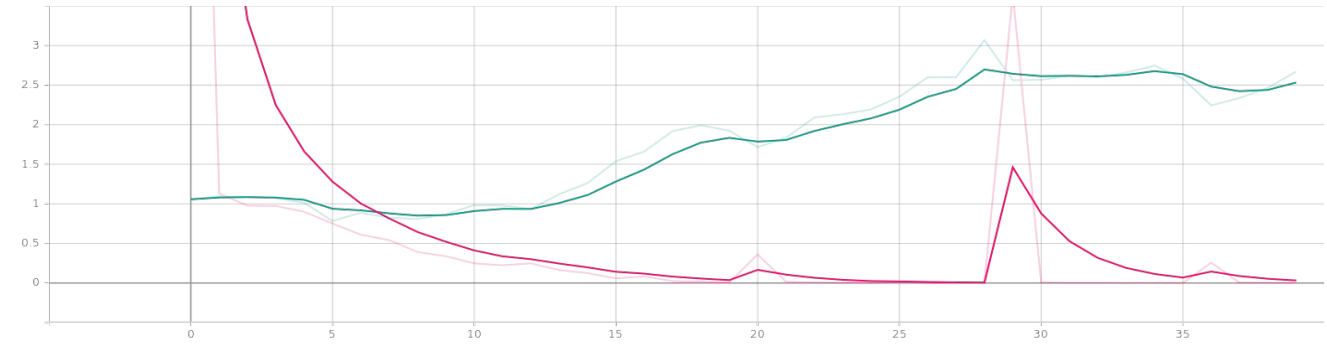
Compilation			Fit			Data					
Optimizer	Loss	Metrics	Epoch	Final loss	Final accuracy	Batch size	Height	Width	Depth	Train images	Test images
Adam	SparseCategoricalCrossentropy	Accuracy	40	0,000527	1	400	250	250	1	398	400

- Accuracy



Name	Smoothed	Value	Step	Time	Relative
jean_epochs_40_data_size_250/train	0.9957	1	39	Tue Apr 14, 18:43:13	14m 26s
jean_epochs_40_data_size_250/validation	0.6743	0.66	39	Tue Apr 14, 18:43:13	14m 26s

- Loss



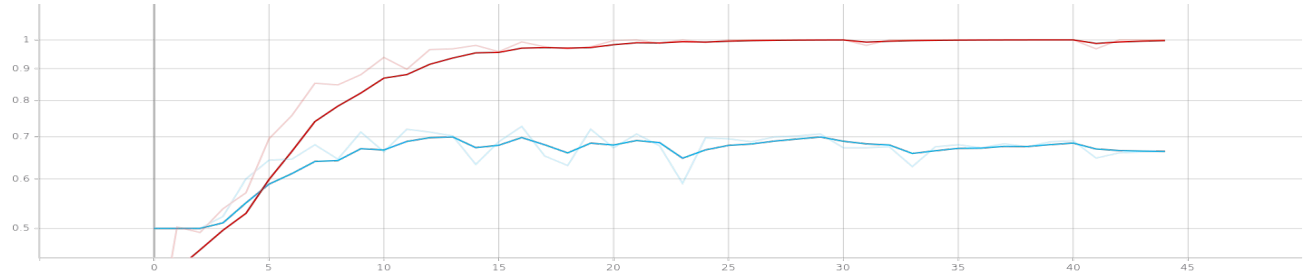
Name	Smoothed	Value	Step	Time	Relative
jean_epochs_40_data_size_250/train	0.03201	5.2788e-4	39	Tue Apr 14, 18:43:13	14m 26s
jean_epochs_40_data_size_250/validation	2.533	2.672	39	Tue Apr 14, 18:43:13	14m 26s

Epoch to 45

id	Model									
	$Couche_1$	$Couche_2$	$Couche_3$	$Couche_4$	$Couche_5$	$Couche_6$	$Couche_7$	$Couche_8$	$Couche_9$	Couche_10
Base	Convolution 2D : 32, 3 Activation: relu	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : relu	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : relu	Flatten	Dense: 64 Activation: relu	Dense: 3

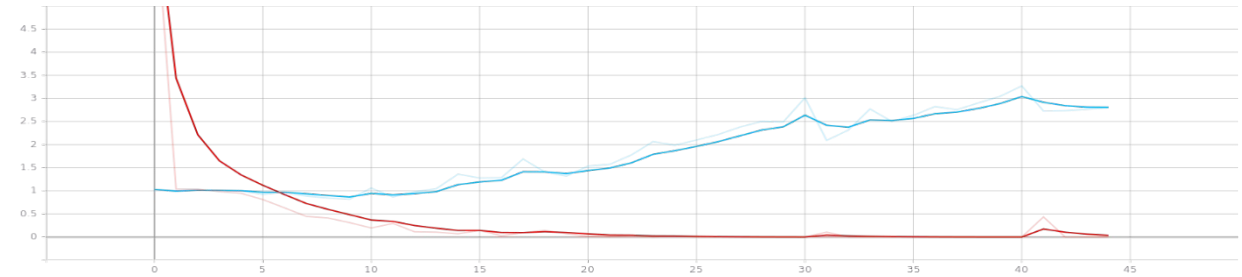
Compilation			Fit			Data					
Optimizer	Loss	Metrics	Epoch	Final loss	Final accuracy	Batch size	Height	Width	Depth	Train images	Test images
Adam	SparseCategoricalCrossentropy	Accuracy	45	0,00009	1	400	224	224	1	398	400

- Accuracy



Name	Smoothed	Value	Step	Time	Relative
jean_epochs_45/train	0.9972	1	44	Tue Apr 14, 18:21:35	12m 54s
jean_epochs_45/validation	0.6637	0.6625	44	Tue Apr 14, 18:21:35	12m 54s

- Loss



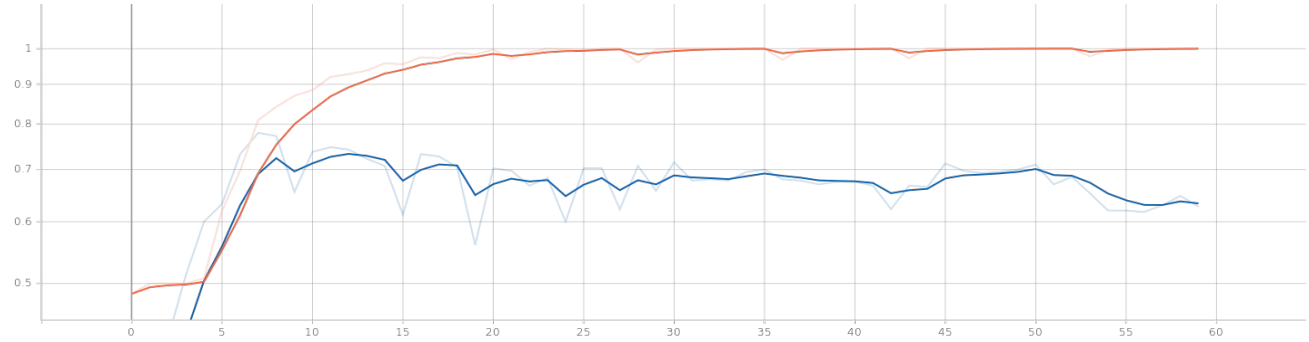
Name	Smoothed	Value	Step	Time	Relative
jean_epochs_45/train	0.03795	9.0005e-5	44	Tue Apr 14, 18:21:35	12m 54s
jean_epochs_45/validation	2.808	2.802	44	Tue Apr 14, 18:21:35	12m 54s

Epoch to 60

id	Model									
	$Couche_1$	$Couche_2$	$Couche_3$	$Couche_4$	$Couche_5$	$Couche_6$	$Couche_7$	$Couche_8$	$Couche_9$	Couche_10
Base	Convolution 2D : 32, 3 Activation: relu	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : relu	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : relu	Flatten	Dense: 64 Activation: relu	Dense: 3

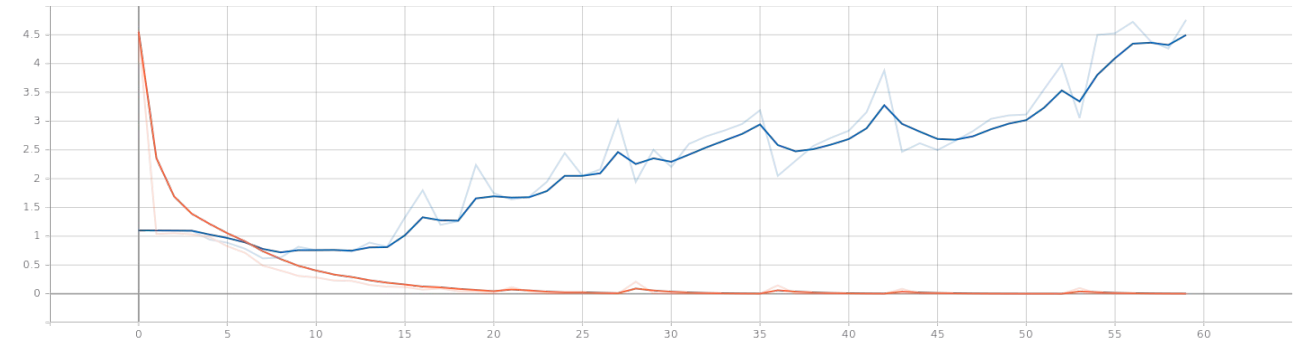
Compilation			Fit			Data					
Optimizer	Loss	Metrics	Epoch	Final loss	Final accuracy	Batch size	Height	Width	Depth	Train images	Test images
Adam	SparseCategoricalCrossentropy	Accuracy	60	0,0000071	1	400	224	224	1	398	400

- Accuracy



Name	Smoothed	Value	Step	Time	Relative
jean_epochs_60/train	0.9995	1	59	Tue Apr 14, 17:57:04	17m 47s
jean_epochs_60/validation	0.6333	0.6275	59	Tue Apr 14, 17:57:04	17m 47s

- Loss



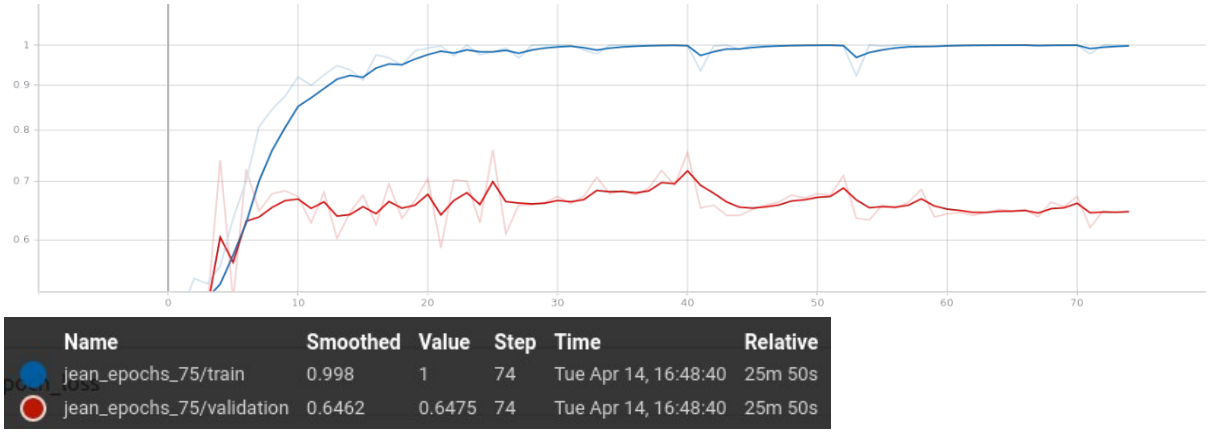
Name	Smoothed	Value	Step	Time	Relative
jean_epochs_60/train	2.1263e-3	7.1021e-6	59	Tue Apr 14, 17:57:04	17m 47s
jean_epochs_60/validation	4.496	4.758	59	Tue Apr 14, 17:57:04	17m 47s

Epoch to 75

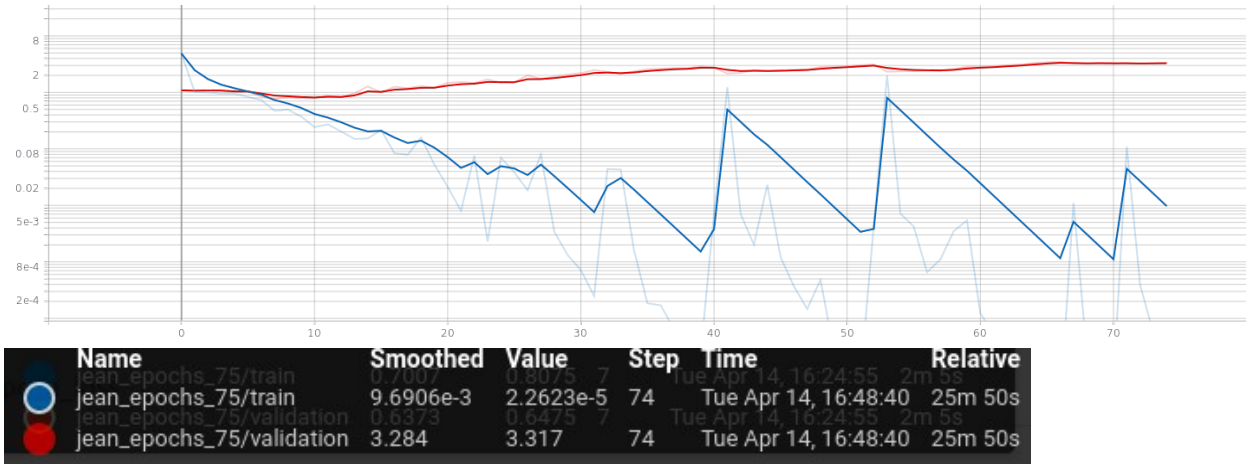
id	Model									
	$Couche_1$	$Couche_2$	$Couche_3$	$Couche_4$	$Couche_5$	$Couche_6$	$Couche_7$	$Couche_8$	$Couche_9$	Couche_10
Base	Convolution 2D : 32, 3 Activation: relu	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : relu	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : relu	Flatten	Dense: 64 Activation: relu	Dense: 3

Compilation			Fit			Data					
Optimizer	Loss	Metrics	Epoch	Final loss	Final accuracy	Batch size	Height	Width	Depth	Train images	Test images
Adam	SparseCategoricalCrossentropy	Accuracy	75	0,000026	1	400	224	224	1	398	400

- Accuracy



- Loss

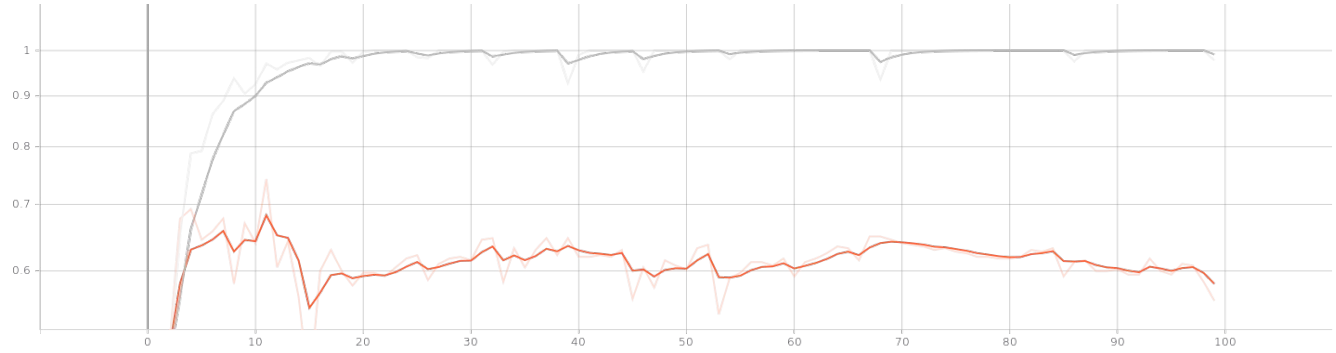


Epoch to 100

id	Model									
	$Couche_1$	$Couche_2$	$Couche_3$	$Couche_4$	$Couche_5$	$Couche_6$	$Couche_7$	$Couche_8$	$Couche_9$	Couche_10
Base	Convolution 2D : 32, 3 Activation: relu	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : relu	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : relu	Flatten	Dense: 64 Activation: relu	Dense: 3

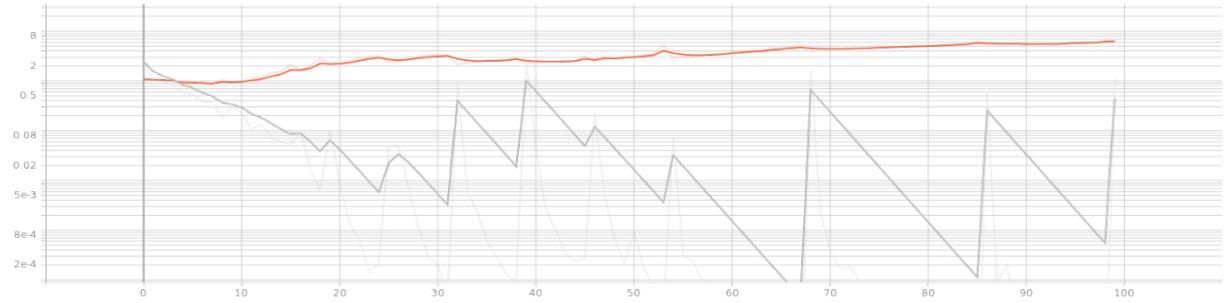
Compilation			Fit			Data					
Optimizer	Loss	Metrics	Epoch	Final loss	Final accuracy	Batch size	Height	Width	Depth	Train images	Test images
Adam	SparseCategoricalCrossentropy	Accuracy	100	1,1289	0,9775	400	224	224	1	398	400

- Accuracy



Name	Smoothed	Value	Step	Time	Relative
jean_epochs_100/train	0.995	1	55	Tue Apr 14, 16:02:24	16m 21s
jean_epochs_100/validation	0.5935	0.5975	55	Tue Apr 14, 16:02:24	16m 21s

- Loss



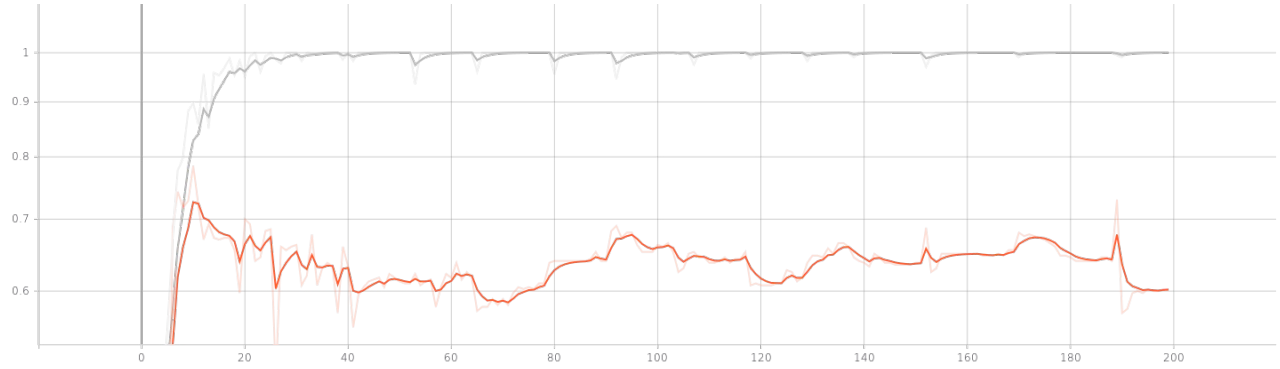
Name	Smoothed	Value	Step	Time	Relative
jean_epochs_100/train	0.991	0.9775	99	Tue Apr 14, 16:15:14	29m 12s
jean_epochs_100/validation	0.5824	0.56	99	Tue Apr 14, 16:15:14	29m 12s

Epoch to 200

id	Model									
	$Couche_1$	$Couche_2$	$Couche_3$	$Couche_4$	$Couche_5$	$Couche_6$	$Couche_7$	$Couche_8$	$Couche_9$	Couche_10
Base	Convolution 2D : 32, 3 Activation: relu	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : relu	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : relu	Flatten	Dense: 64 Activation: relu	Dense: 3

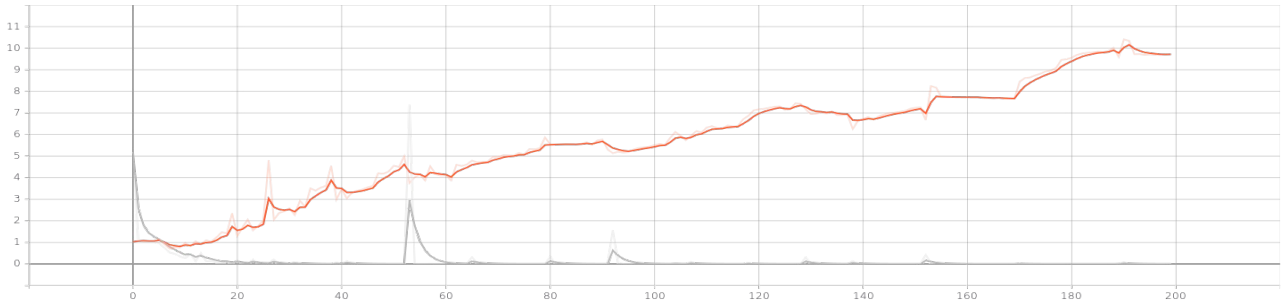
Compilation			Fit			Data					
Optimizer	Loss	Metrics	Epoch	Final loss	Final accuracy	Batch size	Height	Width	Depth	Train images	Test images
Adam	SparseCategoricalCrossentropy	Accuracy	200	0,000000724	1	400	224	224	1	398	400

- Accuracy



Name	Smoothed	Value	Step	Time	Relative
jean_epochs_200/train	0.9999	1	199	Tue Apr 14, 20:06:12	59m 6s
jean_epochs_200/validation	0.6018	0.6025	199	Tue Apr 14, 20:06:12	59m 6s

- Loss



Name	Smoothed	Value	Step	Time	Relative
jean_epochs_200/train	3.9531e-4	2.0265e-7	199	Tue Apr 14, 20:06:12	59m 6s
jean_epochs_200/validation	9.716	9.719	199	Tue Apr 14, 20:06:12	59m 6s

The results of this study is that the evolution of the accuracy with differents number of epoch is not very convincing even with the application of a change in data size.
The best results of validation accuracy is for the test with 40 Epochs, but no test excedd 0.71 of validation accuracy.

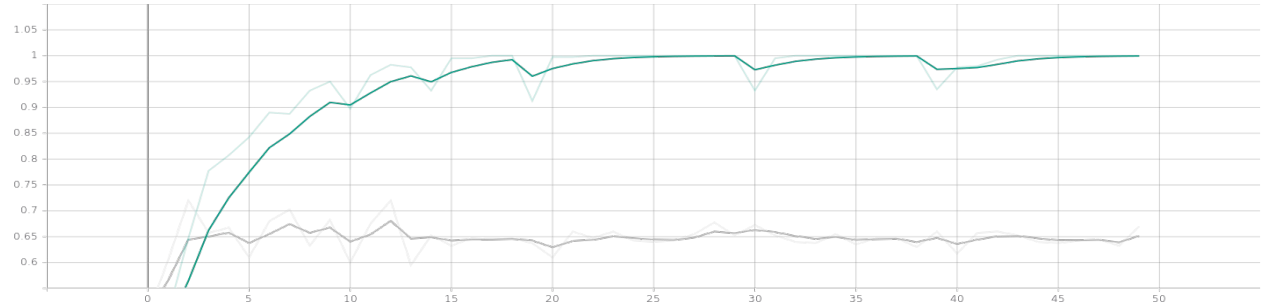
5. RMS Optimizer

The purpose of this test is to change the optimizer variable to “RMSprop” and look the evolutoin for the accuracy and loss during the train and the validation.

id	Model									
	$Couche_1$	$Couche_2$	$Couche_3$	$Couche_4$	$Couche_5$	$Couche_6$	$Couche_7$	$Couche_8$	$Couche_9$	Couche_10
Base	Convolution 2D : 32, 3 Activation: relu	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : relu	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : relu	Flatten	Dense: 64 Activation: relu	Dense: 3

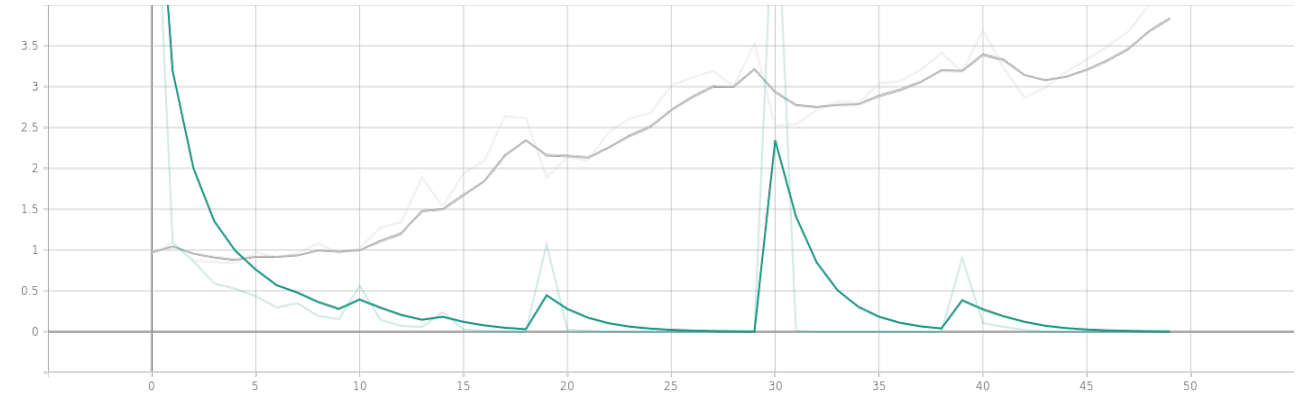
Compilation			Fit			Data					
Optimizer	Loss	Metrics	Epoch	Final loss	Final accuracy	Batch size	Height	Width	Depth	Train images	Test images
RMSprop	Sparse Categorical Cross entropy	Accuracy	50	0,000013	1	400	224	224	1	398	400

- Accuracy



Name	Smoothed	Value	Step	Time	Relative
maxime_RMSprop_optimizer/train	0.9995	1	49	Tue Mar 31, 12:40:48	15m 38s
maxime_RMSprop_optimizer/validation	0.6515	0.67	49	Tue Mar 31, 12:40:48	15m 38s

- Loss



Name	Smoothed	Value	Step	Time	Relative
maxime_RMSprop_optimizer/train	3.4745e-3	1.2861e-5	49	Tue Mar 31, 12:40:48	15m 38s
maxime_RMSprop_optimizer/validation	3.836	4.072	49	Tue Mar 31, 12:40:48	15m 38s

The conclusion of this test is the optimizer by RSM allow to reach an accuracy of 0.61 in validation by an increase of loss up to 4,072.

6. Data Augmentation

The purpose of this test is to apply some visual modification to the data and look the evolution of the results.

Here our complete model for the data augmentation.

id	Model								
	$Couche_1$	$Couche_2$	$Couche_3$	$Couche_4$	$Couche_5$	$Couche_6$	$Couche_7$	$Couche_8$	$Couche_9$
Base + Data_Augment	Convolution 2D : 32, 3 Activation : relu	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : relu	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : relu	Flatten	Dense: 64 Activation: relu

Compilation			Fit			Data					
Optimizer	Loss	Metrics	Epoch	Final loss	Final accuracy	Batch size	Height	Width	Depth	Train images	Test images
Adam	SparseCategoricalCrossentropy	Accuracy	50	0.1127	0.9575	400	224	224	1	398	400

Results: Final Accuracy = 0,6675; Final Loss = 13,0216

Batch to 1000

id	Model									
	$Couche_1$	$Couche_2$	$Couche_3$	$Couche_4$	$Couche_5$	$Couche_6$	$Couche_7$	$Couche_8$	$Couche_9$	Couche_10
Base	Convolution 2D : 32, 3 Activation: relu	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : relu	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : relu	Flatten	Dense: 64 Activation: relu	Dense: 3

Compilation			Fit			Data					
Optimizer	Loss	Metrics	Epoch	Final loss	Final accuracy	Batch size	Height	Width	Depth	Train images	Test images
Adam	SparseCategoricalCrossentropy	Accuracy	50	1.9577e-05	1	1000	224	224	1	398	400

Results: Final Accuracy = 0,7100; Final Loss = 2,6245

Data Augmentation with Bach to 1000

id	Model									
	$Couche_1$	$Couche_2$	$Couche_3$	$Couche_4$	$Couche_5$	$Couche_6$	$Couche_7$	$Couche_8$	$Couche_9$	Couche_10
Base	Convolution 2D : 32, 3 Activation: relu	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : relu	Max pooling 2D: 2	Drop out: 0,2	Convolution 2D: 64, 3 Activation : relu	Flatten	Dense: 64 Activation: relu	Dense: 3

Compilation			Fit			Data					
Optimizer	Loss	Metrics	Epoch	Final loss	Final accuracy	Batch size	Height	Width	Depth	Train images	Test images
Adam	SparseCategoricalCrossentropy	Accuracy	50	0,0671	0,9575	1000	224	224	1	398	400

Results: Final accuracy = 0,6480; Final Loss = 15,6329

In conclusion the data augmentation on our images doesn't allow a real augmentation of the accuracy. The percentage of loss undergoes a huge increase. This solution is not viable due to the low number of images used.

7. Layers

The purpose of this test is to change the layers dimension and look the consequence on the accuracy.

id	Model													
	Couche$_1$	Couche$_2$	Couche$_3$	Couche$_4$	Couche$_5$	Couche$_6$	Couche$_7$	Couche$_8$	Couche$_9$	Couche_10	Couche_11	Couche_12	Couche_13	Couche_14
4 Layers to 1x1x160	Convolution 2D : 20, 3 Activation: relu	Max pooling 2D: 2	Dropout: 0,2	Convolution 2D: 40, 7 Activation : relu	Max pooling 2D: 3	Dropout: 0,2	Convolution 2D: 80, 7 Activation : relu	Max pooling 2D: 5	Dropout: 0,2	Convolution 2D: 160, 4 Activation : relu	Max pooling 2D: 2	Max pooling 2D: 2	Dense: 1024 Activation: relu	Dense: 3

Compilation			Fit			Data					
Optimizer	Loss	Metrics	Epoch	Final loss	Final accuracy	Batch size	Height	Width	Depth	Train images	Test images
Adam	SparseCategoricalCrossentropy	Accuracy	50	1.3373e-3	1	400	200	200	1	398	400

Results: Final Accuracy = 0,73; Final Loss = 2,326

We can note a stagnation for the final accuracy and a sloght increase for the final accuracy.

8. Data balancing : train and test images numbers

Let's talk about data balancing, as the more you have images to feed the model, the more he has a chance to be accurate.

That's why, for example, with the Mnist dataset, you have 60 000 training examples and 10 000 test examples.

We can notice that for the previous example , there are more training images than test one.

If we merge all the images, that make a dataset of 70 000.

So 85% is take as training and 15% as test.

So the way to change these proportions is called data balancing.

Here's the base images separations :

Train images => 398

Test images => 400

Validation images => 16

As you can see, there's a strange proportion between them.

So we decide to merge all of these images into one directory of 814 images (separated into subdirectories for labels = PNEUMONIA, VIRUS, NORMAL).

And start to proceed data balancing to look how the proportion between train and test can influence accuracy.

As we have betters results with 0,6 dropout, we used it for these tests.

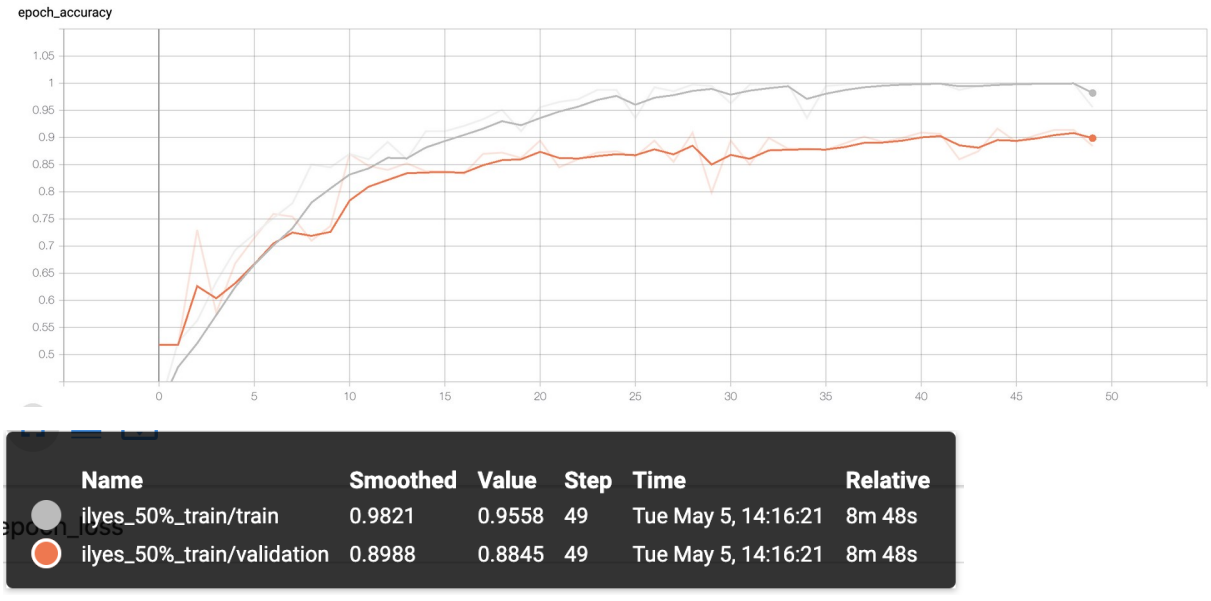
50% of the training images and a Drop Out of 0,6

id	Model									
	$Couche_1$	$Couche_2$	$Couche_3$	$Couche_4$	$Couche_5$	$Couche_6$	$Couche_7$	$Couche_8$	$Couche_9$	Couche_10
70% for training + Drop out 0,6	Convolution 2D : 32, 3 Activation : relu	Max pooling 2D: 2	Drop out: 0,6	Convolution 2D: 64, 3 Activation : relu	Max pooling 2D: 2	Drop out: 0,6	Convolution 2D: 64, 3 Activation : relu	Flatten	Dense: 64 Activation: relu	Dense: 3

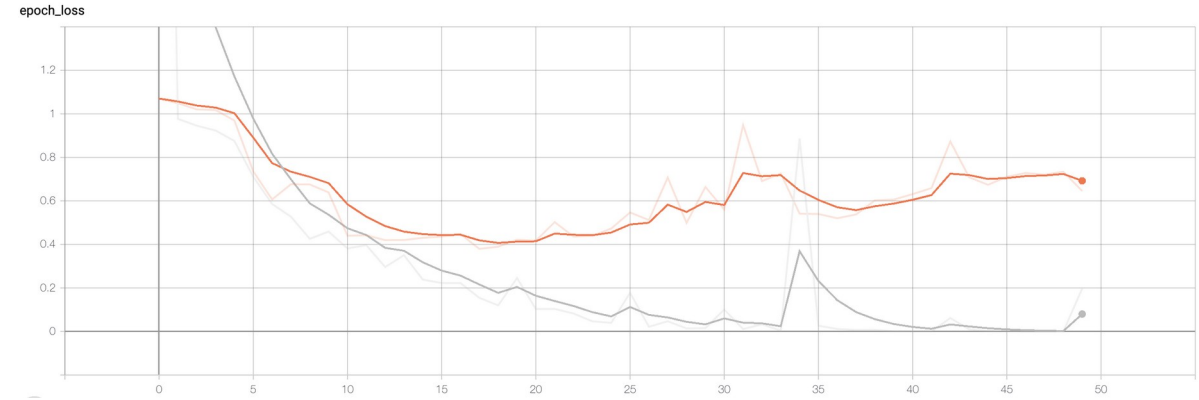
Compilation			Fit			Data					
Optimizer	Loss	Metrics	Epoch	Final loss	Final accuracy	Batch size	Height	Width	Depth	Train images	Test images
Adam	Sparse Categorical Crossentropy	Accuracy	50	0.1971	0.9558	407	224	224	1	407	407

Results: Final Accuracy = 0,8845; Final Loss = 0.6453

- Accuracy



- Loss



Name	Smoothed	Value	Step	Time	Relative
ilyes_50%_train/train	0.08014	0.1971	49	Tue May 5, 14:16:21	8m 48s
ilyes_50%_train/validation	0.6923	0.6453	49	Tue May 5, 14:16:21	8m 48s

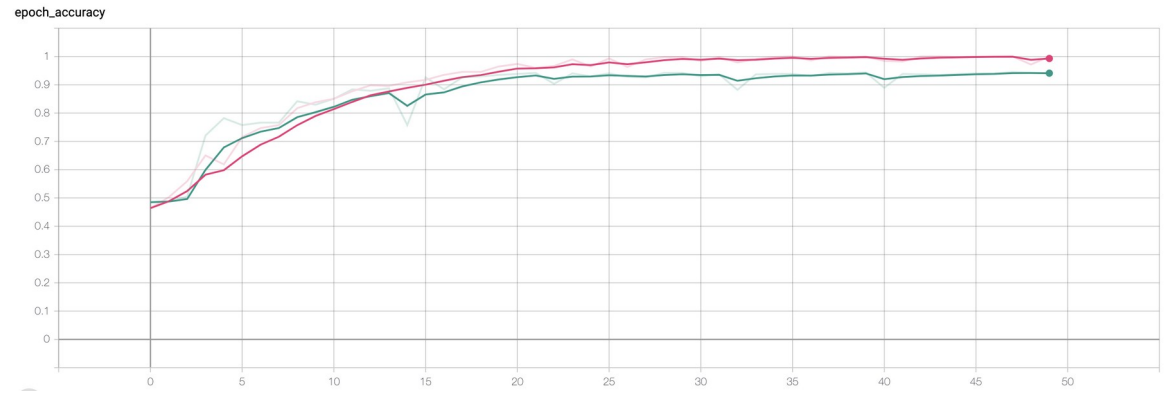
70% of the training images and a Drop Out of 0,6

id	Model									
	$Couche_1$	$Couche_2$	$Couche_3$	$Couche_4$	$Couche_5$	$Couche_6$	$Couche_7$	$Couche_8$	$Couche_9$	Couche_10
Base	Convolution 2D : 32, 3 Activation: relu	Max pooling 2D: 2	Drop out: 0,6	Convolution 2D: 64, 3 Activation : relu	Max pooling 2D: 2	Drop out: 0,6	Convolution 2D: 64, 3 Activation : relu	Flatten	Dense: 64 Activation: relu	Dense: 3

Compilation			Fit			Data					
Optimizer	Loss	Metrics	Epoch	Final loss	Final accuracy	Batch size	Height	Width	Depth	Train images	Test images
Adam	SparseCategoricalCrossentropy	Accuracy	50	5.5155e-4	1	569	224	224	1	569	244

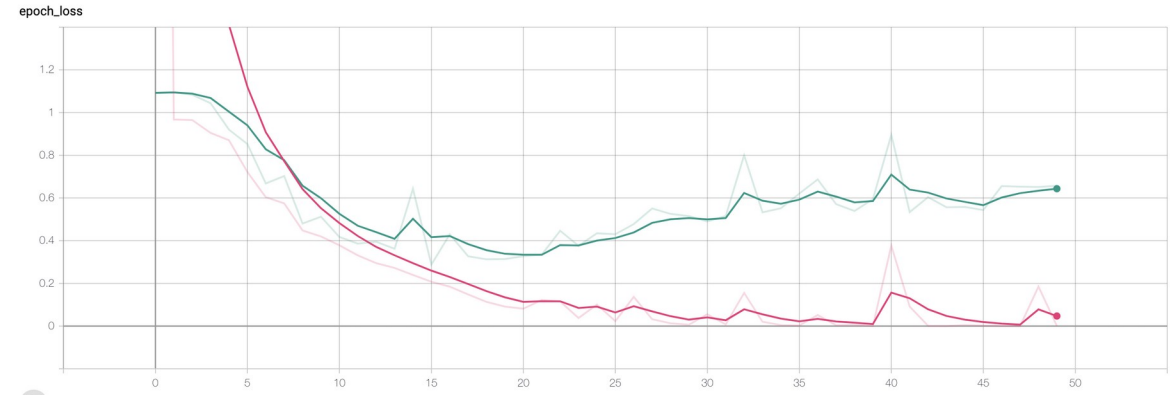
Results: Final Accuracy = 0,9402; Final Loss = 0,6575

- Accuracy



Name	Smoothed	Value	Step	Time	Relative
ilyes_70%_train/train	0.993	1	49	Tue May 5, 12:14:30	12m 2s
ilyes_70%_train/validation	0.941	0.9402	49	Tue May 5, 12:14:30	12m 2s

- Loss



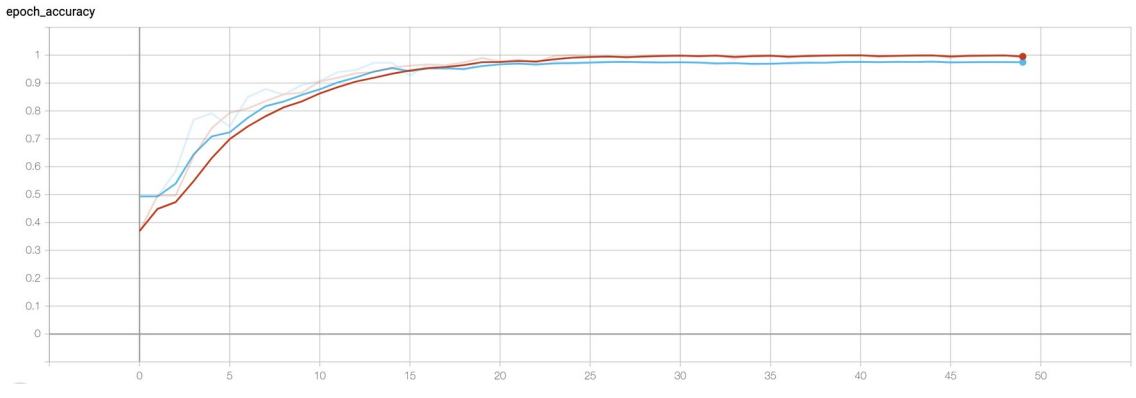
Name	Smoothed	Value	Step	Time	Relative
ilyes_70%_train/train	0.04717	5.5155e-4	49	Tue May 5, 12:14:30	12m 2s
ilyes_70%_train/validation	0.6434	0.6575	49	Tue May 5, 12:14:30	12m 2s

85% of the training images and a Drop Out of 0,6

id	Model									
	$Couche_1$	$Couche_2$	$Couche_3$	$Couche_4$	$Couche_5$	$Couche_6$	$Couche_7$	$Couche_8$	$Couche_9$	Couche_10
Base	Convolution 2D : 32, 3 Activation: relu	Max pooling 2D: 2	Drop out: 0,6	Convolution 2D: 64, 3 Activation : relu	Max pooling 2D: 2	Drop out: 0,6	Convolution 2D: 64, 3 Activation : relu	Flatten	Dense: 64 Activation: relu	Dense: 3
Compilation	Fit	Data								
Optimizer	Loss	Metrics	Epoch	Final loss	Final accuracy	Batch size	Height	Width	Depth	Train images	Test images
Adam	SparseCategoricalCrossentropy	Accuracy	50	0.112	0.9899	691	224	224	1	691	122

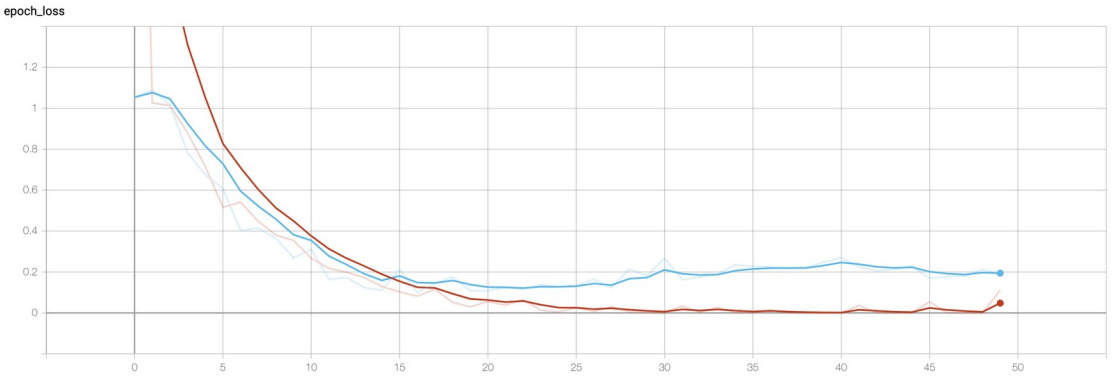
Results: Final Accuracy = 0,974; Final Loss = 0,1897

- Accuracy



Name	Smoothed	Value	Step	Time	Relative
ilyes_85%_train/train	0.9954	0.9899	49	Tue May 5, 11:40:24	14m 27s
ilyes_85%_train/validation	0.9747	0.974	49	Tue May 5, 11:40:24	14m 27s

- Loss



Name	Smoothed	Value	Step	Time	Relative
ilyes_85%_train/train	0.04798	0.112	49	Tue May 5, 11:40:24	14m 27s
ilyes_85%_train/validation	0.1944	0.1897	49	Tue May 5, 11:40:24	14m 27s

To conclude , as the training images increase, the validation_accuracy too.

But when you increase the percentage of training images , there's less images for test so the val_accuracy may have more fluctuation and tend to be less accurate to the real val_accuracy.

So as we have a pretty small dataset , I think that the good proportion to take is 70% / 30% like the minist one.

With this setup we reach 0,94 which is pretty good.

9. Batch Size

The purpose of this final study is to vary the size of the batch and look the impact on the final results.

Batch Size 32 and Drop Out to 0,6

id	Model										Compilation		
	Couche $_1$	Couche $_2$	Couche $_3$	Couche $_4$	Couche $_5$	Couche $_6$	Couche $_7$	Couche $_8$	Couche $_9$	Couche10	Optimizer	Loss	Metrics
BATCH SIZE 32 + Dropout 0,6	Convolution 2D : 32, 3 Activation: relu	Max pooling 2D: 2	Dropout: 0,6	Convolution 2D: 64, 3 Activation : relu	Max pooling 2D: 2	Dropout: 0,6	Convolution 2D: 64, 3 Activation : relu	Flatten	Dense : 64 Activation: relu	Dense : 3	Adam	Sparse Categorical Cross entropy	Accuracy

Fit			Data					
Epoch	Final loss	Final accuracy	Batch size	Height	Width	Depth	Train images	Test images
50	0.0925	0.9750	32	224	224	1	398	400

Results: Final Accuracy = 0,5725; Final Loss = 4,3970

Batch Size 64 and Drop Out to 0,6

id	Model									
	$Couche_1$	$Couche_2$	$Couche_3$	$Couche_4$	$Couche_5$	$Couche_6$	$Couche_7$	$Couche_8$	$Couche_9$	Couche_10
Base	Convolution 2D : 32, 3 Activation: relu	Max pooling 2D: 2	Drop out: 0,6	Convolution 2D: 64, 3 Activation : relu	Max pooling 2D: 2	Drop out: 0,6	Convolution 2D: 64, 3 Activation : relu	Flatten	Dense: 64 Activation: relu	Dense: 3

Compilation			Fit			Data					
Optimizer	Loss	Metrics	Epoch	Final loss	Final accuracy	Batch size	Height	Width	Depth	Train images	Test images
Adam	SparseCategoricalCrossentropy	Accuracy	50	0,6679	0,9688	64	224	224	1	398	400

Results: Final Accuracy = 0,6406; Final Loss = 0,9624

Batch Size 128 and Drop Out to 0,6

id	Model									
	$Couche_1$	$Couche_2$	$Couche_3$	$Couche_4$	$Couche_5$	$Couche_6$	$Couche_7$	$Couche_8$	$Couche_9$	Couche_10
Base	Convolution 2D : 32, 3 Activation: relu	Max pooling 2D: 2	Drop out: 0,6	Convolution 2D: 64, 3 Activation : relu	Max pooling 2D: 2	Drop out: 0,6	Convolution 2D: 64, 3 Activation : relu	Flatten	Dense: 64 Activation: relu	Dense: 3

Compilation			Fit			Data					
Optimizer	Loss	Metrics	Epoch	Final loss	Final accuracy	Batch size	Height	Width	Depth	Train images	Test images
Adam	SparseCategoricalCrossentropy	Accuracy	50	0,0125	1	128	224	224	1	398	400

Results: Final Accuracy = 0,6641; Final Loss = 1,5429

Batch Size 256 and Drop Out to 0,6

id	Model									
	$Couche_1$	$Couche_2$	$Couche_3$	$Couche_4$	$Couche_5$	$Couche_6$	$Couche_7$	$Couche_8$	$Couche_9$	Couche_10
Base	Convolution 2D : 32, 3 Activation: relu	Max pooling 2D: 2	Drop out: 0,6	Convolution 2D: 64, 3 Activation : relu	Max pooling 2D: 2	Drop out: 0,6	Convolution 2D: 64, 3 Activation : relu	Flatten	Dense: 64 Activation: relu	Dense: 3

Compilation			Fit			Data					
Optimizer	Loss	Metrics	Epoch	Final loss	Final accuracy	Batch size	Height	Width	Depth	Train images	Test images
Adam	SparseCategoricalCrossentropy	Accuracy	50	1,7265	0,8750	256	224	224	1	398	400

Results: Final Accuracy = 0,6641; Final Loss = 1,6982

Batch Size 512 and Drop Out to 0,6

id	Model									
	$Couche_1$	$Couche_2$	$Couche_3$	$Couche_4$	$Couche_5$	$Couche_6$	$Couche_7$	$Couche_8$	$Couche_9$	Couche_10
Base	Convolution 2D : 32, 3 Activation: relu	Max pooling 2D: 2	Drop out: 0,6	Convolution 2D: 64, 3 Activation : relu	Max pooling 2D: 2	Drop out: 0,6	Convolution 2D: 64, 3 Activation : relu	Flatten	Dense: 64 Activation: relu	Dense: 3

Compilation			Fit			Data					
Optimizer	Loss	Metrics	Epoch	Final loss	Final accuracy	Batch size	Height	Width	Depth	Train images	Test images
Adam	SparseCategoricalCrossentropy	Accuracy	50	6.0806e-05	1	512	224	224	1	398	400

Results: Final Accuracy = 0,6816; Final Loss = 2,8601

Batch Size 1024 and Drop Out to 0,6

id	Model									
	$Couche_1$	$Couche_2$	$Couche_3$	$Couche_4$	$Couche_5$	$Couche_6$	$Couche_7$	$Couche_8$	$Couche_9$	Couche_10
Base	Convolution 2D : 32, 3 Activation: relu	Max pooling 2D: 2	Drop out: 0,6	Convolution 2D: 64, 3 Activation : relu	Max pooling 2D: 2	Drop out: 0,6	Convolution 2D: 64, 3 Activation : relu	Flatten	Dense: 64 Activation: relu	Dense: 3

Compilation			Fit			Data					
Optimizer	Loss	Metrics	Epoch	Final loss	Final accuracy	Batch size	Height	Width	Depth	Train images	Test images
Adam	SparseCategoricalCrossentropy	Accuracy	50	1.8626e-09	1	1024	224	224	1	398	400

Results: Final Accuracy = 0,6836; Final Loss = 6, 3964

In conclusion, we can see a growth of the final loss with the increase of the batch size.
However, the final accuracy doesn't exceed 0,66.

With some research we found that we misunderstood the definition for batch size.

We thought that a batch size is just the number of data took from the dataset to feed the model.

But thanks to this article : https://towardsdatascience.com/epoch-vs-iterations-vs-batch-size-4dfb9c7ce9c9

all tensorflow weird words seems clear now.

Epochs

> One Epoch is when an ENTIRE dataset is passed forward and backward through the neural network only ONCE.

But What is a Batch?

As I said, you can't pass the entire dataset into the neural net at once. So, you **divide dataset into Number of Batches or sets or parts.**

Batch Size

> Total number of training examples present in a single batch.

Iterations

To get the iterations you just need to know multiplication tables or have a calculator. 😃

> Iterations is the number of batches needed to complete one epoch.

So with these definitions we can now really conclude on batch size results.

As previously, the number of batch size was 1, the batch size was the number of images give to the model. So that was normal that the accuracy increase when the batch size get closer to the dataset images number.

Because the more different images you give to the model, the more he will be accurate.

But the problem is that you must update the Batch size to fit with the number of images.

But as the batch size was not made for that , here's the correct way to do this :

> We can divide the dataset of 2000 examples into batches of 500 then it will take 4 iterations to complete 1 epoch.

For this example , the dataset size is 2000 and the batch_size = 500.

So you must make 4 iterations per epochs to pass trough the whole dataset,

To do that, you must fill the steps_per_epoch variable with this value in the model.fit function.

But you 'll say why make this thing difficult while it works ?

Because the batch size was made for optimization.

Since the whole dataset is too big to feed to the computer at once, we divide it in several smaller batches.

As we had a lot of things to cover, we didn't have time to implement this correct method.

But don't worry, it works fine too.