

MNIST Training Example

Document Author

Contents

1	Summary	1
2	Training reports	2
2.1	Model 1: ConvNet	2
2.2	Model 2: Two layer MLP	4
2.3	Model 3: Five layer MLP	6
3	Model Architectures	8
3.1	Model architecture of MLP5layers	8
3.2	Model architecture of MLP2layers	9
3.3	Model architecture of ConvNet2layers	10

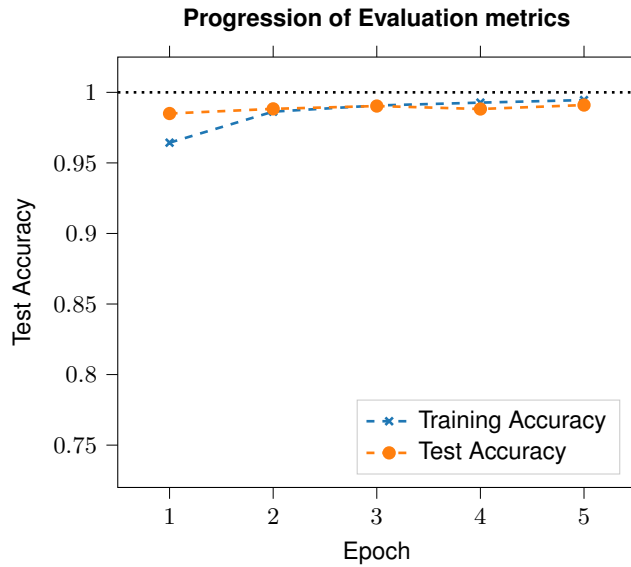
1 Summary

Nº	Study name	Model	#Parameters	#Epochs	Batch size	Test Acc.	Training Acc.
1	ConvNet	ConvNet2layers	1 199 882	5	16	99.1 %	99.45 %
2	Two layer MLP	MLP2layers	669 706	5	16	90.78 %	89.63 %
3	Five layer MLP	MLP5layers	1 457 674	5	16	91.84 %	91.38 %

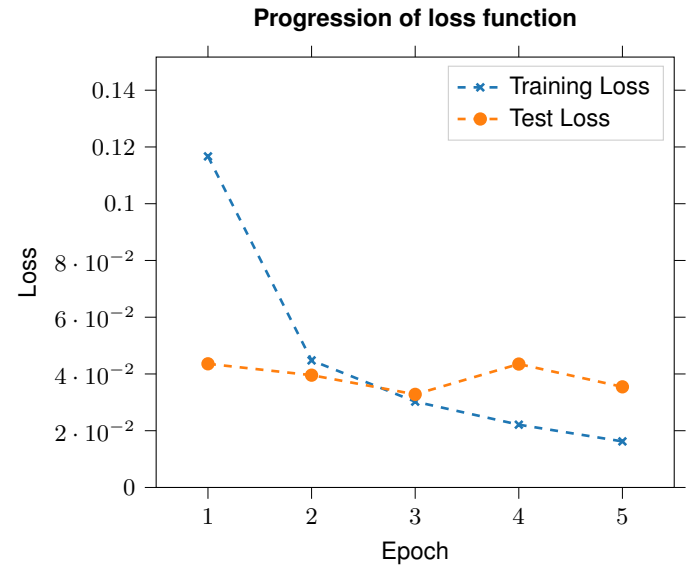
2 Training reports

2.1 Model 1: ConvNet

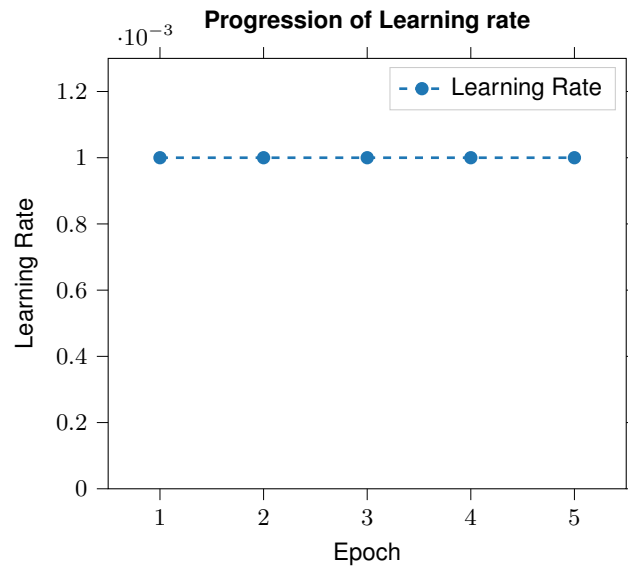
Training history See Figure 1.



(a) Accuracy learning process for study 1.



(b) Loss learning process for study 1.



(c) Learning rate per epoch for study 1.

Figure 1: Training and evaluation metrics for study 1.

Link to model: https://keras.io/examples/mnist_cnn/

Dataset

Name MNIST

Train-Test-Dev split: *Training set:* 60000, *Test set:* 10000, *Dev set:* 0,

Image size [28, 28]

Training

Number of epochs 5

Optimizer Adam (Kingma et al., 2015)

Learning Rate	0.0010000000474974513
Beta 1	0.89999999761581421
Beta 2	0.9990000128746033
Decay	0.0
Epsilon	1e-07
Amsgrad	False

Loss Categorical crossentropy

Batch size 16

Shuffle Yes

Training time 2 min 42 sec

Platform

Weights exported to path weights\ConvNet2layers_5ep_MNIST.h5

Device used GPU (GeForce GTX 1060 6GB)

CPU Intel(R) Xeon(R) CPU E3-1245 v5 @ 3.50GHz, X86_64

Python Version 3.7.2.final.0 (64 bit)

Keras Version 2.2.5 (Backend: tensorflow)

Tensorflow Version 1.14.0

Timestamp 26.09.2019 at 13:50

2.2 Model 2: Two layer MLP

Training history See Figure 2.

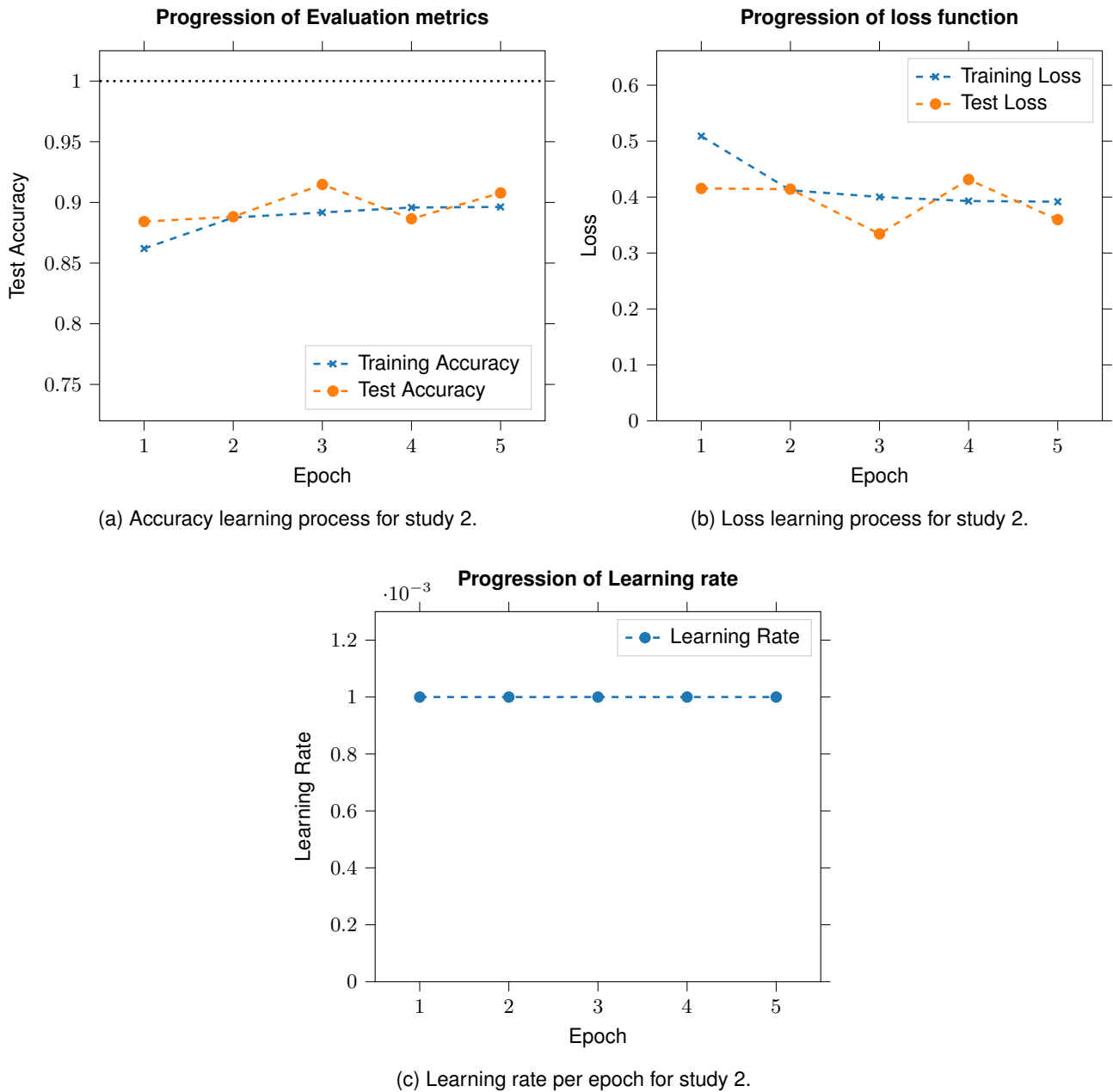


Figure 2: Training and evaluation metrics for study 2.

Link to model: https://keras.io/examples/mnist_mlp/

Dataset

Name MNIST

Train-Test-Dev split: *Training set:* 60000, *Test set:* 10000, *Dev set:* 0,

Image size [28, 28]

Training

Number of epochs 5

Optimizer RMSProp (Hinton et al. 2014)

Learning Rate 0.000100000000474974513

Rho 0.89999999761581421

Decay 0.0

Epsilon 1e-07

Loss Categorical crossentropy

Batch size 16

Shuffle Yes

Training time 1 min 51 sec

Platform

Weights exported to path weights\MLP2layers_5ep_MNIST.h5

Device used GPU (GeForce GTX 1060 6GB)

CPU Intel(R) Xeon(R) CPU E3-1245 v5 @ 3.50GHz, X86_64

Python Version 3.7.2.final.0 (64 bit)

Keras Version 2.2.5 (Backend: tensorflow)

Tensorflow Version 1.14.0

Timestamp 26.09.2019 at 13:52

2.3 Model 3: Five layer MLP

Training history See Figure 3.

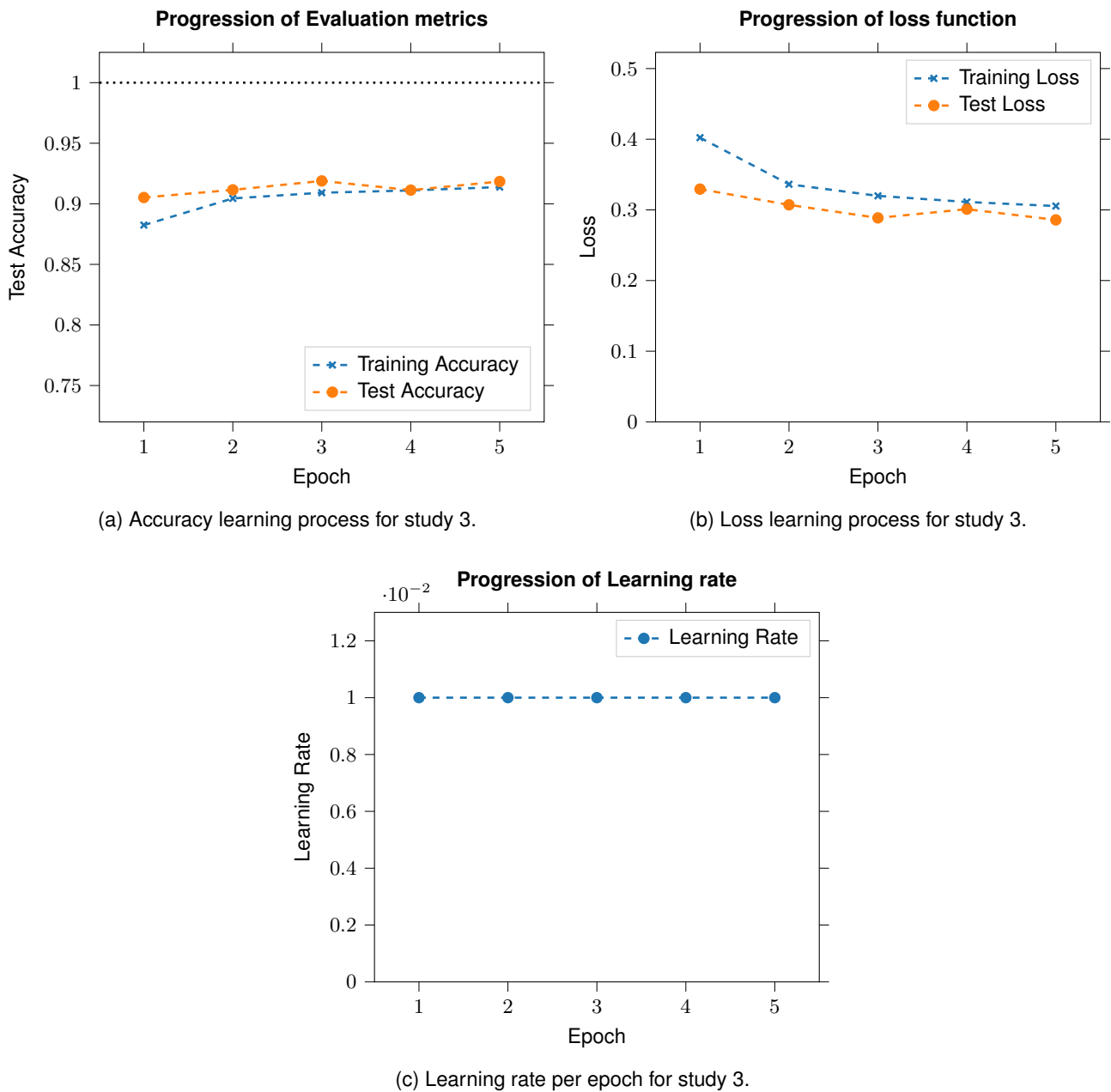


Figure 3: Training and evaluation metrics for study 3.

Dataset

Name MNIST

Train-Test-Dev split: *Training set:* 60000, *Test set:* 10000, *Dev set:* 0,

Image size [28, 28]

Training

Number of epochs 5

Optimizer Stochastic Gradient Descent

Learning Rate 0.0009999999310821295

Momentum 0.0

Decay 0.0

Nesterov False

Loss Categorical crossentropy

Batch size 16

Shuffle Yes

Training time 2 min 5 sec

Platform

Weights exported to path weights\MLP5layers_5ep_MNIST.h5

Device used GPU (GeForce GTX 1060 6GB)

CPU Intel(R) Xeon(R) CPU E3-1245 v5 @ 3.50GHz, X86_64

Python Version 3.7.2.final.0 (64 bit)

Keras Version 2.2.5 (Backend: tensorflow)

Tensorflow Version 1.14.0

Timestamp 26.09.2019 at 13:54

3 Model Architectures

3.1 Model architecture of MLP5layers

Used in study №: 3

Model summary:

Nº	Layer (Type)	Output shape	Config	#Parameters	Inbound layers
0	input_3 (InputLayer)	(28, 28, 1)		0	
1	flatten_3 (Flatten)	(784,)		0	input_3
2	dense_6 (Dense)	(512,)	#Neurons: 512 Activation: linear	401 920	flatten_3
3	dropout_5 (Dropout)	(512,)	Dropout Rate: 0.0	0	dense_6
4	dense_7 (Dense)	(512,)	#Neurons: 512 Activation: linear	262 656	dropout_5
5	dropout_6 (Dropout)	(512,)	Dropout Rate: 0.0	0	dense_7
6	dense_8 (Dense)	(512,)	#Neurons: 512 Activation: linear	262 656	dropout_6
7	dropout_7 (Dropout)	(512,)	Dropout Rate: 0.0	0	dense_8
8	dense_9 (Dense)	(512,)	#Neurons: 512 Activation: linear	262 656	dropout_7
9	dropout_8 (Dropout)	(512,)	Dropout Rate: 0.2	0	dense_9
10	dense_10 (Dense)	(512,)	#Neurons: 512 Activation: linear	262 656	dropout_8
11	dropout_9 (Dropout)	(512,)	Dropout Rate: 0.2	0	dense_10
12	dense_11 (Dense)	(10,)	#Neurons: 10 Activation: softmax	5130	dropout_9

3.2 Model architecture of MLP2layers

Used in study №: 3

Model summary:

Nº	Layer (Type)	Output shape	Config	#Parameters	Inbound layers
0	input_2 (InputLayer)	(28, 28, 1)		0	
1	flatten_2 (Flatten)	(784,)		0	input_2
2	dense_3 (Dense)	(512,)	#Neurons: 512 Activation: linear	401 920	flatten_2
3	dropout_3 (Dropout)	(512,)	Dropout Rate: 0.0	0	dense_3
4	dense_4 (Dense)	(512,)	#Neurons: 512 Activation: linear	262 656	dropout_3
5	dropout_4 (Dropout)	(512,)	Dropout Rate: 0.2	0	dense_4
6	dense_5 (Dense)	(10,)	#Neurons: 10 Activation: softmax	5130	dropout_4

3.3 Model architecture of ConvNet2layers

Used in study №: 3

Model summary:

Nº	Layer (Type)	Output shape	Config	#Parameters	Inbound layers
0	input_1 (InputLayer)	(28, 28, 1)		0	
1	conv2d_1 (Conv2D)	(26, 26, 32)	Activation: relu Kernel Size: [3, 3] Stride: [1, 1] Dilation: [1, 1] Padding: valid	320	input_1
2	conv2d_2 (Conv2D)	(24, 24, 64)	Activation: relu Kernel Size: [3, 3] Stride: [1, 1] Dilation: [1, 1] Padding: valid	18 496	conv2d_1
3	max_pooling2d_1 (MaxPooling2D)	(12, 12, 64)	Pool size: [2, 2] Strides: [2, 2] Padding: valid	0	conv2d_2
4	dropout_1 (Dropout)	(12, 12, 64)	Dropout Rate: 0.0	0	max_pooling2d_1
5	flatten_1 (Flatten)	(9216,)		0	dropout_1
6	dense_1 (Dense)	(128,)	#Neurons: 128 Activation: relu	1 179 776	flatten_1
7	dropout_2 (Dropout)	(128,)	Dropout Rate: 0.2	0	dense_1
8	dense_2 (Dense)	(10,)	#Neurons: 10 Activation: softmax	1290	dropout_2