Deep Learning

Lab Session 2 - 3 Hours

Convolutional Neural Network (CNN) for Handwritten Digits Recognition

Student 1: EL MAATAOUI OUSSAMA **Student 2:** EL KHAMAR Zakaria

The aim of this session is to practice with Convolutional Neural Networks. Answers and experiments should be made by groups of one or two students. Each group should fill and run appropriate notebook cells.

Once you have completed all of the code implementations and successfully answered each question above, you may finalize your work by exporting the iPython Notebook as an pdf document using print as PDF (Ctrl+P). Do not forget to run all your cells before generating your final report and do not forget to include the names of all participants in the group. The lab session should be completed by May 29th 2017.

Send you pdf file to benoit.huet@eurecom.fr and olfa.ben-ahmed@eurecom.fr using **[DeepLearning_lab2]** as Subject of your email.

Introduction

In the last Lab Session, you built a Multilayer Perceptron for recognizing hand-written digits from the MNIST data-set. The best achieved accuracy on testing data was about 97%. Can you do better than these results using a deep CNN ? In this Lab Session, you will build, train and optimize in TensorFlow one of the early Convolutional Neural Networks: **LeNet-5** to go to more than 99% of accuracy.

Load MNIST Data in TensorFlow

Run the cell above to load the MNIST data that comes with TensorFlow. You will use this data in **Section 1** and **Section 2**.

In [2]:

```
import tensorflow as tf
from tensorflow.examples.tutorials.mnist import input_data
mnist = input_data.read_data_sets("MNIST_data/", one_hot=True)
X_train, y_train           = mnist.train.images, mnist.train.labels
X_validation, y_validation = mnist.validation.images, mnist.validation.labels
X_test, y_test             = mnist.test.images, mnist.test.labels
print("Image Shape: {}".format(X_train[0].shape))
print("Training Set:   {} samples".format(len(X_train)))
print("Validation Set: {} samples".format(len(X_validation)))
print("Test Set:       {} samples".format(len(X_test)))
```

```
Extracting MNIST_data/train-images-idx3-ubyte.gz
Extracting MNIST_data/train-labels-idx1-ubyte.gz
Extracting MNIST_data/t10k-images-idx3-ubyte.gz
Extracting MNIST_data/t10k-labels-idx1-ubyte.gz
Image Shape: (784,)
```

```
Training Set:   55000 samples
Validation Set: 5000 samples
Test Set:       10000 samples
```

Section 1 : My First Model in TensorFlow

Before starting with CNN, let's train and test in TensorFlow the example : **y=softmax(Wx+b)** seen in the DeepLearing course last week.

This model reaches an accuracy of about 92 %. You will also learn how to launch the tensorBoard https://www.tensorflow.org/get_started/summaries_and_tensorboard to visualize the computation graph, statistics and learning curves.

Part 1 : Read carefully the code in the cell below. Run it to perform training.

In [3]:

```
from __future__ import print_function
import tensorflow as tf


#STEP 1

# Parameters
learning_rate = 0.01
training_epochs = 100
batch_size = 128
display_step = 1
logs_path = 'log_files/'  # useful for tensorboard

# tf Graph Input:  mnist data image of shape 28*28=784
x = tf.placeholder(tf.float32, [None, 784], name='InputData')
# 0-9 digits recognition,  10 classes
y = tf.placeholder(tf.float32, [None, 10], name='LabelData')

# Set model weights
W = tf.Variable(tf.zeros([784, 10]), name='Weights')
b = tf.Variable(tf.zeros([10]), name='Bias')

# Construct model and encapsulating all ops into scopes, making Tensorboard's Graph visualization more c
onvenient
with tf.name_scope('Model'):
    # Model
    pred = tf.nn.softmax(tf.matmul(x, W) + b) # Softmax
with tf.name_scope('Loss'):
    # Minimize error using cross entropy
    cost = tf.reduce_mean(-tf.reduce_sum(y*tf.log(pred), reduction_indices=1))
with tf.name_scope('SGD'):
    # Gradient Descent
    optimizer = tf.train.GradientDescentOptimizer(learning_rate).minimize(cost)
with tf.name_scope('Accuracy'):
    # Accuracy
    acc = tf.equal(tf.argmax(pred, 1), tf.argmax(y, 1))
    acc = tf.reduce_mean(tf.cast(acc, tf.float32))

# Initializing the variables
init = tf.global_variables_initializer()
# Create a summary to monitor cost tensor
tf.summary.scalar("Loss", cost)
# Create a summary to monitor accuracy tensor
```

```
tf.summary.scalar("Accuracy", acc)
# Merge all summaries into a single op
merged_summary_op = tf.summary.merge_all()


#STEP 2


# Launch the graph for training
with tf.Session() as sess:
    sess.run(init)
    # op to write logs to Tensorboard
    summary_writer = tf.summary.FileWriter(logs_path, graph=tf.get_default_graph())
    # Training cycle
    for epoch in range(training_epochs):
        avg_cost = 0.
        total_batch = int(mnist.train.num_examples/batch_size)
        # Loop over all batches
        for i in range(total_batch):
            batch_xs, batch_ys = mnist.train.next_batch(batch_size)
            # Run optimization op (backprop), cost op (to get loss value)
            # and summary nodes
            _, c, summary = sess.run([optimizer, cost, merged_summary_op],
                                     feed_dict={x: batch_xs, y: batch_ys})
            # Write logs at every iteration
            summary_writer.add_summary(summary, epoch * total_batch + i)
            # Compute average loss
            avg_cost += c / total_batch
        # Display logs per epoch step
        if (epoch+1) % display_step == 0:
            print("Epoch: ", '%02d' % (epoch+1), "  =====> Loss=", "{:.9f}".format(avg_cost))

    print("Optimization Finished!")

    # Test model
    # Calculate accuracy
    print("Accuracy:", acc.eval({x: mnist.test.images, y: mnist.test.labels}))
```

```
Epoch:  01   =====> Loss= 1.286880517
Epoch:  02   =====> Loss= 0.732130182
Epoch:  03   =====> Loss= 0.600188281
Epoch:  04   =====> Loss= 0.536868138
Epoch:  05   =====> Loss= 0.497761437
Epoch:  06   =====> Loss= 0.471160888
Epoch:  07   =====> Loss= 0.451120950
Epoch:  08   =====> Loss= 0.435958526
Epoch:  09   =====> Loss= 0.423563886
Epoch:  10   =====> Loss= 0.413208945
Epoch:  11   =====> Loss= 0.404384853
Epoch:  12   =====> Loss= 0.397065376
Epoch:  13   =====> Loss= 0.390438711
Epoch:  14   =====> Loss= 0.384245008
Epoch:  15   =====> Loss= 0.379302852
Epoch:  16   =====> Loss= 0.374581395
Epoch:  17   =====> Loss= 0.370026611
Epoch:  18   =====> Loss= 0.366134630
Epoch:  19   =====> Loss= 0.363024412
Epoch:  20   =====> Loss= 0.359554454
Epoch:  21   =====> Loss= 0.356713935
Epoch:  22   =====> Loss= 0.353803091
Epoch:  23   =====> Loss= 0.351250163
Epoch:  24   =====> Loss= 0.348754776
Epoch:  25   =====> Loss= 0.346152383
Epoch:  26   =====> Loss= 0.344285993
Epoch:  27   =====> Loss= 0.342232766
```

Epoch: 27 =====> Loss= 0.342252766
Epoch: 28 =====> Loss= 0.340291154
Epoch: 29 =====> Loss= 0.338551447
Epoch: 30 =====> Loss= 0.336681009
Epoch: 31 =====> Loss= 0.335104630
Epoch: 32 =====> Loss= 0.333318973
Epoch: 33 =====> Loss= 0.332122456
Epoch: 34 =====> Loss= 0.330621399
Epoch: 35 =====> Loss= 0.329232193
Epoch: 36 =====> Loss= 0.327746604
Epoch: 37 =====> Loss= 0.326606188
Epoch: 38 =====> Loss= 0.325466239
Epoch: 39 =====> Loss= 0.324251288
Epoch: 40 =====> Loss= 0.323196035
Epoch: 41 =====> Loss= 0.322047231
Epoch: 42 =====> Loss= 0.320889451
Epoch: 43 =====> Loss= 0.320052969
Epoch: 44 =====> Loss= 0.319119280
Epoch: 45 =====> Loss= 0.318031952
Epoch: 46 =====> Loss= 0.317180339
Epoch: 47 =====> Loss= 0.316070622
Epoch: 48 =====> Loss= 0.315036609
Epoch: 49 =====> Loss= 0.314505193
Epoch: 50 =====> Loss= 0.313559846
Epoch: 51 =====> Loss= 0.313004764
Epoch: 52 =====> Loss= 0.312009802
Epoch: 53 =====> Loss= 0.311367977
Epoch: 54 =====> Loss= 0.310596779
Epoch: 55 =====> Loss= 0.310297175
Epoch: 56 =====> Loss= 0.309299224
Epoch: 57 =====> Loss= 0.308769881
Epoch: 58 =====> Loss= 0.308101477
Epoch: 59 =====> Loss= 0.307491954
Epoch: 60 =====> Loss= 0.306709714
Epoch: 61 =====> Loss= 0.305945171
Epoch: 62 =====> Loss= 0.305437322
Epoch: 63 =====> Loss= 0.304671963
Epoch: 64 =====> Loss= 0.304189998
Epoch: 65 =====> Loss= 0.303699724
Epoch: 66 =====> Loss= 0.303161961
Epoch: 67 =====> Loss= 0.302680239
Epoch: 68 =====> Loss= 0.302326534
Epoch: 69 =====> Loss= 0.301751988
Epoch: 70 =====> Loss= 0.301280748
Epoch: 71 =====> Loss= 0.301040544
Epoch: 72 =====> Loss= 0.300013314
Epoch: 73 =====> Loss= 0.299895031
Epoch: 74 =====> Loss= 0.299330392
Epoch: 75 =====> Loss= 0.298652754
Epoch: 76 =====> Loss= 0.298564738
Epoch: 77 =====> Loss= 0.298117534
Epoch: 78 =====> Loss= 0.297861958
Epoch: 79 =====> Loss= 0.296834854
Epoch: 80 =====> Loss= 0.296939363
Epoch: 81 =====> Loss= 0.296352997
Epoch: 82 =====> Loss= 0.296028742
Epoch: 83 =====> Loss= 0.295795181
Epoch: 84 =====> Loss= 0.295343559
Epoch: 85 =====> Loss= 0.294933017
Epoch: 86 =====> Loss= 0.294695835
Epoch: 87 =====> Loss= 0.294009037
Epoch: 88 =====> Loss= 0.293841000
Epoch: 89 =====> Loss= 0.293728300
Epoch: 90 =====> Loss= 0.292952279
Epoch: 91 =====> Loss= 0.292944078

```
Epoch:  92   =====> Loss= 0.292389964
Epoch:  93   =====> Loss= 0.291990771
Epoch:  94   =====> Loss= 0.291567540
Epoch:  95   =====> Loss= 0.291503989
Epoch:  96   =====> Loss= 0.291220591
Epoch:  97   =====> Loss= 0.290986427
Epoch:  98   =====> Loss= 0.290613253
Epoch:  99   =====> Loss= 0.289965497
Epoch:  100   =====> Loss= 0.289691290
Optimization Finished!
Accuracy: 0.9203
```

Part 2 : Using Tensorboard, we can now visualize the created graph, giving you an overview of your architecture and how all of the major components are connected. You can also see and analyse the learning curves.

To launch tensorBoard:

- Go to the **TP2** folder,
- Open a Terminal and run the command line **"tensorboard --logdir= log_files/"**, it will generate an http link ,ex http://666.6.6.6:6006,
- Copy this link into your web browser

Enjoy It !!

Section 2 : The 99% MNIST Challenge !

Part 1 : LeNet5 implementation

One you are now familar with **tensorFlow** and **tensorBoard**, you are in this section to build, train and test the baseline LeNet-5 model for the MNIST digits recognition problem.

In more advanced step you will make some optimizations to get more than 99% of accuracy. The best model can get to over 99.7% accuracy!

For more information, have a look at this list of results :
http://rodrigob.github.io/are_we_there_yet/build/classification_datasets_results.html

<img src="lenet.png",width="800" height="600" align="center">
Figure 1: Lenet 5

The LeNet architecture accepts a 32x32xC image as input, where C is the number of color channels. Since MNIST images are grayscale, C is 1 in this case.

Layer 1: Convolutional. The output shape should be 28x28x6 **Activation.** sigmoid **Pooling.** The output shape should be 14x14x6.

Layer 2: Convolutional. The output shape should be 10x10x16. **Activation.** sigmoid **Pooling.** The output shape should be 5x5x16.

Flatten. Flatten the output shape of the final pooling layer such that it's 1D instead of 3D. You may need to use *_flatten* from tensorflow.contrib.layers import flatten

Layer 3: Fully Connected. This should have 120 outputs. **Activation.** sigmoid

Layer 4: Fully Connected. This should have 84 outputs. **Activation.** sigmoid

Layer 5: Fully Connected. This should have 10 outputs.**Activation.** softmax

Question 2.1.1 Implement the Neural Network architecture described above. For that, your will use classes and functions from https://www.tensorflow.org/api_docs/python/tf/nn.

We give you some helper functions for weigths and bias initilization. Also you can refer to section 1.

In [4]:

```
#Helper functions  for weigths and bias initilization



def weight_variable(shape):
  initial = tf.truncated_normal(shape, stddev=0.1)
  return tf.Variable(initial)

def bias_variable(shape):
  initial = tf.constant(0.1, shape=shape)
  return tf.Variable(initial)
```

In [4]:

```
from tensorflow.contrib.layers import flatten

def LeNet5_Model(x):

    # weights and biais initialization
    W_conv1 = tf.Variable(tf.truncated_normal(shape=(5, 5, 1, 6), mean = 0, stddev = 0.1))
    b_conv1  = bias_variable([6])

    # Convolution Layer1
    conv1   = tf.nn.conv2d(x, W_conv1, strides=[1, 1, 1, 1], padding='VALID') + b_conv1

    # Activation.
    conv1 = tf.nn.sigmoid(conv1)

    # Max Pooling
    conv1 = tf.nn.max_pool(conv1, ksize=[1, 2, 2, 1], strides=[1, 2, 2, 1], padding='VALID')

    # Convolution Layer2
    W_conv2 = tf.Variable(tf.truncated_normal(shape=(5, 5, 6, 16), mean = 0, stddev = 0.1))
    b_conv2 = bias_variable([16])
    conv2   = tf.nn.conv2d(conv1, W_conv2 , strides=[1, 1, 1, 1], padding='VALID') + b_conv2

    # Activation.
    conv2 = tf.nn.sigmoid(conv2)

    # Max Pooling
    conv2 = tf.nn.max_pool(conv2, ksize=[1, 2, 2, 1], strides=[1, 2, 2, 1], padding='VALID')

    flatten_conv2   = flatten(conv2)

    # Fully Connected1
    W_fully1= tf.Variable(tf.truncated_normal(shape=(400, 120), mean = 0, stddev = 0.1))
    b_fully1 = tf.Variable(tf.zeros(120))
    fully1   = tf.matmul(flatten_conv2, W_fully1) + b_fully1

    # Activation.
    fully1   = tf.nn.sigmoid(fully1)

    # Fully Connected2
```

```
    W_fully2 = tf.Variable(tf.truncated_normal(shape=(120, 84), mean = 0, stddev = 0.1))
    b_fully2  = bias_variable([84])
    fully2   = tf.matmul(fully1, W_fully2) + b_fully2

    # Activation.
    fully2   = tf.nn.sigmoid(fully2)

    # Fully Connected3
    W_fully3 = tf.Variable(tf.truncated_normal(shape=(84, 10), mean = 0, stddev = 0.1))
    b_fully3  = bias_variable([10])
    fully3 = tf.matmul(fully2, W_fully3) + b_fully3
    # the softmax will be applied internally in what follows with the softmax_cross_entropy_with_logits
    return fully3
```

Question 2.1.2. Calculate the number of parameters of this model

Convolutionnal layer 1 :

Weights : 5x5x1x6 = 150

biais : 6

Convolutionnal layer 2:

Weights : 5x5x6x16 = 2400

biais : 16

Fully connected layer :

Weights : 5x5x16x120 = 48000

biais : 120

Fully connected layer 2 :

Weights :120x84 = 10080

biais : 84

Fully connected layer 3 :

Weights :84x10 = 840

biais : 10

In [6]:

```
print ('the total number of the parameters of this model is',150+2400+48000+10080+840+10+84+120+16+6
)
```

```
the total number of the parameters of this model is 61706
```

Question 2.1.3. Start the training with the parameters cited below:

Learning rate =0.1
Loss Fucntion : Cross entropy
Optimisateur: SGD
Number of training iterations= 10000
The batch size =128

In [224]:

```
# test
# Parameters
```

```
import numpy as np
learning_rate = 0.1
training_epochs = 100
batch_size = 128
logs_path = "TP2/MNIST_99_Challenge_Figures"
x = tf.placeholder(tf.float32, [None, 28, 28, 1], name='InputData')
# 0-9 digits recognition,  10 classes
y = tf.placeholder(tf.float32, [None, 10], name='LabelData')

x_adapted = tf.map_fn(lambda x1: tf.image.pad_to_bounding_box(x1, np.random.randint(4), np.random.ran
dint(4), 32, 32), x)
model = LeNet5_Model(x_adapted)

cross_entropy = tf.nn.softmax_cross_entropy_with_logits(logits=model, labels=y)
loss_operation = tf.reduce_mean(cross_entropy)
optimizer = tf.train.GradientDescentOptimizer(learning_rate = learning_rate)
training_operation = optimizer.minimize(loss_operation)
```

Question 2.1.4. Implement the evaluation function for accuracy computation

In [15]:

```
correct_prediction = tf.equal(tf.argmax(model, 1), tf.argmax(y, 1))
accuracy_operation = tf.reduce_mean(tf.cast(correct_prediction, tf.float32))
saver = tf.train.Saver()

def evaluate(X_data, Y_data, sess):
    X_data = X_data.reshape([-1,28,28,1])
    num_examples = len(X_data)
    total_accuracy = 0

    for i in range(0, num_examples, batch_size):
        batch_xs, batch_ys = X_data[i:i+batch_size], Y_data[i:i+batch_size]

        accuracy = sess.run(accuracy_operation, feed_dict={x: batch_xs, y: batch_ys})
        total_accuracy += (accuracy * len(batch_xs))

    return total_accuracy / num_examples
```

Question 2.1.5. Implement training pipeline and run the training data through it to train the model.

- Before each epoch, shuffle the training set.
- Print the loss per mini batch and the training/validation accuracy per epoch. (Display results every 100 epochs)
- Save the model after training
- Print after training the final testing accuracy

In [16]:

```
from sklearn.utils import shuffle
def train(X_train,y_train,X_validation,y_validation,X_test,y_test,loss_operation = 'Cross_entropy',optimizer =
'SGD',batch_size =128,learning_rate = 0.1):
    with tf.Session() as sess:
        sess.run(tf.global_variables_initializer())
        tf.summary.scalar("training_accuracy", accuracy_operation)
        tf.summary.scalar("Validation_accuracy", accuracy_operation)
        summary_writer = tf.summary.FileWriter(logs_path, graph=tf.get_default_graph())
        merged_summary_op = tf.summary.merge_all()
        num_examples = len(X_train)

        print("Start Training!")
```

```
    for i in range(training_epochs):
        X_train, y_train = shuffle(X_train, y_train)
        for offset in range(0, num_examples, batch_size):
            end = offset + batch_size
            batch_x, batch_y = X_train[offset:end], y_train[offset:end]
            batch_x = batch_x.reshape([-1,28,28,1])


            sess.run(training_operation, feed_dict={x: batch_x, y: batch_y})

        validation_accuracy = evaluate(X_validation, y_validation, sess)

        training_accuracy = evaluate(X_train, y_train, sess)
        print("EPOCH {} :".format(i+1))
        print("Training Accuracy = {:.3f}".format(training_accuracy))
        print("Validation Accuracy = {:.3f}".format(validation_accuracy))

    print("Optimization Finished!")

    # Test model
    # Calculate accuracy
    test_accuracy = evaluate(X_test, y_test, sess)
    print("Test Accuracy = {:.3f}".format(test_accuracy))

    print ("Training Finished!")


    saver.save(sess, 'lenet5V')
    print("Model saved")
```

In [223]:

```
train(X_train,y_train,X_validation,y_validation,X_test,y_test)
```

```
Start Training!
EPOCH 1 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 2 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 3 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 4 :
Training Accuracy = 0.103
Validation Accuracy = 0.099
EPOCH 5 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 6 :
Training Accuracy = 0.104
Validation Accuracy = 0.110
EPOCH 7 :
Training Accuracy = 0.099
Validation Accuracy = 0.099
EPOCH 8 :
Training Accuracy = 0.104
Validation Accuracy = 0.110
EPOCH 9 :
Training Accuracy = 0.145
Validation Accuracy = 0.132
EPOCH 10 :
Training Accuracy = 0.303
Validation Accuracy = 0.311
```

Validation Accuracy = 0.911
EPOCH 11 :
Training Accuracy = 0.668
Validation Accuracy = 0.674
EPOCH 12 :
Training Accuracy = 0.791
Validation Accuracy = 0.803
EPOCH 13 :
Training Accuracy = 0.857
Validation Accuracy = 0.863
EPOCH 14 :
Training Accuracy = 0.893
Validation Accuracy = 0.899
EPOCH 15 :
Training Accuracy = 0.910
Validation Accuracy = 0.919
EPOCH 16 :
Training Accuracy = 0.922
Validation Accuracy = 0.930
EPOCH 17 :
Training Accuracy = 0.931
Validation Accuracy = 0.939
EPOCH 18 :
Training Accuracy = 0.940
Validation Accuracy = 0.949
EPOCH 19 :
Training Accuracy = 0.944
Validation Accuracy = 0.956
EPOCH 20 :
Training Accuracy = 0.949
Validation Accuracy = 0.958
EPOCH 21 :
Training Accuracy = 0.954
Validation Accuracy = 0.960
EPOCH 22 :
Training Accuracy = 0.958
Validation Accuracy = 0.963
EPOCH 23 :
Training Accuracy = 0.961
Validation Accuracy = 0.967
EPOCH 24 :
Training Accuracy = 0.963
Validation Accuracy = 0.968
EPOCH 25 :
Training Accuracy = 0.965
Validation Accuracy = 0.969
EPOCH 26 :
Training Accuracy = 0.967
Validation Accuracy = 0.971
EPOCH 27 :
Training Accuracy = 0.968
Validation Accuracy = 0.973
EPOCH 28 :
Training Accuracy = 0.970
Validation Accuracy = 0.975
EPOCH 29 :
Training Accuracy = 0.971
Validation Accuracy = 0.976
EPOCH 30 :
Training Accuracy = 0.973
Validation Accuracy = 0.975
EPOCH 31 :
Training Accuracy = 0.973
Validation Accuracy = 0.978
EPOCH 32 :

Training Accuracy = 0.974
Validation Accuracy = 0.978
EPOCH 33 :
Training Accuracy = 0.975
Validation Accuracy = 0.977
EPOCH 34 :
Training Accuracy = 0.975
Validation Accuracy = 0.979
EPOCH 35 :
Training Accuracy = 0.977
Validation Accuracy = 0.979
EPOCH 36 :
Training Accuracy = 0.978
Validation Accuracy = 0.981
EPOCH 37 :
Training Accuracy = 0.979
Validation Accuracy = 0.979
EPOCH 38 :
Training Accuracy = 0.978
Validation Accuracy = 0.980
EPOCH 39 :
Training Accuracy = 0.979
Validation Accuracy = 0.981
EPOCH 40 :
Training Accuracy = 0.980
Validation Accuracy = 0.981
EPOCH 41 :
Training Accuracy = 0.979
Validation Accuracy = 0.980
EPOCH 42 :
Training Accuracy = 0.981
Validation Accuracy = 0.981
EPOCH 43 :
Training Accuracy = 0.981
Validation Accuracy = 0.982
EPOCH 44 :
Training Accuracy = 0.982
Validation Accuracy = 0.981
EPOCH 45 :
Training Accuracy = 0.982
Validation Accuracy = 0.984
EPOCH 46 :
Training Accuracy = 0.983
Validation Accuracy = 0.983
EPOCH 47 :
Training Accuracy = 0.983
Validation Accuracy = 0.984
EPOCH 48 :
Training Accuracy = 0.982
Validation Accuracy = 0.982
EPOCH 49 :
Training Accuracy = 0.984
Validation Accuracy = 0.983
EPOCH 50 :
Training Accuracy = 0.984
Validation Accuracy = 0.985
EPOCH 51 :
Training Accuracy = 0.983
Validation Accuracy = 0.984
EPOCH 52 :
Training Accuracy = 0.985
Validation Accuracy = 0.985
EPOCH 53 :
Training Accuracy = 0.985

Validation Accuracy = 0.985
EPOCH 54 :
Training Accuracy = 0.985
Validation Accuracy = 0.985
EPOCH 55 :
Training Accuracy = 0.985
Validation Accuracy = 0.985
EPOCH 56 :
Training Accuracy = 0.985
Validation Accuracy = 0.986
EPOCH 57 :
Training Accuracy = 0.985
Validation Accuracy = 0.985
EPOCH 58 :
Training Accuracy = 0.986
Validation Accuracy = 0.986
EPOCH 59 :
Training Accuracy = 0.987
Validation Accuracy = 0.985
EPOCH 60 :
Training Accuracy = 0.987
Validation Accuracy = 0.985
EPOCH 61 :
Training Accuracy = 0.986
Validation Accuracy = 0.984
EPOCH 62 :
Training Accuracy = 0.987
Validation Accuracy = 0.986
EPOCH 63 :
Training Accuracy = 0.987
Validation Accuracy = 0.986
EPOCH 64 :
Training Accuracy = 0.987
Validation Accuracy = 0.986
EPOCH 65 :
Training Accuracy = 0.988
Validation Accuracy = 0.986
EPOCH 66 :
Training Accuracy = 0.988
Validation Accuracy = 0.986
EPOCH 67 :
Training Accuracy = 0.987
Validation Accuracy = 0.986
EPOCH 68 :
Training Accuracy = 0.988
Validation Accuracy = 0.986
EPOCH 69 :
Training Accuracy = 0.989
Validation Accuracy = 0.987
EPOCH 70 :
Training Accuracy = 0.988
Validation Accuracy = 0.986
EPOCH 71 :
Training Accuracy = 0.988
Validation Accuracy = 0.986
EPOCH 72 :
Training Accuracy = 0.989
Validation Accuracy = 0.986
EPOCH 73 :
Training Accuracy = 0.989
Validation Accuracy = 0.986
EPOCH 74 :
Training Accuracy = 0.989
Validation Accuracy = 0.987
EPOCH 75 :

Training Accuracy = 0.989
Validation Accuracy = 0.987
EPOCH 76 :
Training Accuracy = 0.989
Validation Accuracy = 0.987
EPOCH 77 :
Training Accuracy = 0.988
Validation Accuracy = 0.985
EPOCH 78 :
Training Accuracy = 0.989
Validation Accuracy = 0.986
EPOCH 79 :
Training Accuracy = 0.990
Validation Accuracy = 0.987
EPOCH 80 :
Training Accuracy = 0.990
Validation Accuracy = 0.987
EPOCH 81 :
Training Accuracy = 0.989
Validation Accuracy = 0.988
EPOCH 82 :
Training Accuracy = 0.990
Validation Accuracy = 0.987
EPOCH 83 :
Training Accuracy = 0.990
Validation Accuracy = 0.988
EPOCH 84 :
Training Accuracy = 0.990
Validation Accuracy = 0.987
EPOCH 85 :
Training Accuracy = 0.990
Validation Accuracy = 0.987
EPOCH 86 :
Training Accuracy = 0.991
Validation Accuracy = 0.989
EPOCH 87 :
Training Accuracy = 0.991
Validation Accuracy = 0.988
EPOCH 88 :
Training Accuracy = 0.991
Validation Accuracy = 0.987
EPOCH 89 :
Training Accuracy = 0.991
Validation Accuracy = 0.988
EPOCH 90 :
Training Accuracy = 0.991
Validation Accuracy = 0.987
EPOCH 91 :
Training Accuracy = 0.991
Validation Accuracy = 0.988
EPOCH 92 :
Training Accuracy = 0.991
Validation Accuracy = 0.987
EPOCH 93 :
Training Accuracy = 0.992
Validation Accuracy = 0.988
EPOCH 94 :
Training Accuracy = 0.991
Validation Accuracy = 0.987
EPOCH 95 :
Training Accuracy = 0.992
Validation Accuracy = 0.989
EPOCH 96 :
Training Accuracy = 0.992

```
Validation Accuracy = 0.988
EPOCH 97 :
Training Accuracy = 0.992
Validation Accuracy = 0.988
EPOCH 98 :
Training Accuracy = 0.992
Validation Accuracy = 0.988
EPOCH 99 :
Training Accuracy = 0.992
Validation Accuracy = 0.989
EPOCH 100 :
Training Accuracy = 0.991
Validation Accuracy = 0.989
Optimization Finished!
Test Accuracy = 0.986
Training Finished!
Model saved
```

Question 2.1.6 : Use tensorBoard to visualise and save the LeNet5 Graph and all learning curves. Save all obtained figures in the folder **"TP2/MNIST_99_Challenge_Figures"**

In [31]:

```
#  insert your obtained figure here
saver = tf.train.Saver()
save_path = saver.save(sess, "TP2/MNIST_99_Challenge_Figures")
print ("Model saved in file: ", save_path)
```

Part 2 : LeNET 5 Optimization

Question 2.2.1 Change the sigmoid function with a Relu :

- Retrain your network with SGD and AdamOptimizer and then fill the table above :

Optimizer	Gradient Descent	AdamOptimizer
Training epochs	200	200
learning_rate	0.1	0.1
Validation Accuracy	0.991	0.110
Testing Accuracy	0.990	0.103
Training Time	6281.1995 seconds	6256.7613 seconds

- Try with different learning rates for each Optimizer (0.0001 and 0.001) and different Batch sizes (50 and 128) for 20000 Epochs.
- For each optimizer, plot (on the same curve) the **testing accuracies** function to **(learning rate, batch size)**
- Did you reach the 99% accuracy ? What are the optimal parametres that gave you the best results?

Replacing the sigmoid with a relu and retraining with SGD and AdamOptimizer

In [18]:

```
# Replace the sigmoid with a relu
from tensorflow.contrib.layers import flatten
```

```
def LeNet5_Model(x):

    # weights and biais initialization
    W_conv1 = tf.Variable(tf.truncated_normal(shape=(5, 5, 1, 6), mean = 0, stddev = 0.1))
    b_conv1  = bias_variable([6])

    # Convolution Layer1
    conv1   = tf.nn.conv2d(x, W_conv1, strides=[1, 1, 1, 1], padding='VALID') + b_conv1

    # Activation.
    conv1 = tf.nn.relu(conv1)

    # Max Pooling
    conv1 = tf.nn.max_pool(conv1, ksize=[1, 2, 2, 1], strides=[1, 2, 2, 1], padding='VALID')

    # Convolution Layer2
    W_conv2 = tf.Variable(tf.truncated_normal(shape=(5, 5, 6, 16), mean = 0, stddev = 0.1))
    b_conv2 = bias_variable([16])
    conv2   = tf.nn.conv2d(conv1, W_conv2 , strides=[1, 1, 1, 1], padding='VALID') + b_conv2

    # Activation.
    conv2 = tf.nn.relu(conv2)

    # Max Pooling
    conv2 = tf.nn.max_pool(conv2, ksize=[1, 2, 2, 1], strides=[1, 2, 2, 1], padding='VALID')

    flatten_conv2   = flatten(conv2)

    # Fully Connected1
    W_fully1= tf.Variable(tf.truncated_normal(shape=(400, 120), mean = 0, stddev = 0.1))
    b_fully1 = tf.Variable(tf.zeros(120))
    fully1   = tf.matmul(flatten_conv2, W_fully1) + b_fully1

    # Activation.
    fully1   = tf.nn.relu(fully1)

    # Fully Connected2
    W_fully2 = tf.Variable(tf.truncated_normal(shape=(120, 84), mean = 0, stddev = 0.1))
    b_fully2  = bias_variable([84])
    fully2   = tf.matmul(fully1, W_fully2) + b_fully2

    # Activation.
    fully2   = tf.nn.relu(fully2)

    # Fully Connected3
    W_fully3 = tf.Variable(tf.truncated_normal(shape=(84, 10), mean = 0, stddev = 0.1))
    b_fully3  = bias_variable([10])
    fully3 = tf.matmul(fully2, W_fully3) + b_fully3
    # the softmax will be applied internally in what follows with the softmax_cross_entropy_with_logits
    return fully3
```

SGD, learning rate = 0.1 , Batch size : 128, Training epochs : 200

In [19]:

```
# SGD, learning rate = 0.1 , Batch size : 128, Training epochs : 200

# Parameters
import numpy as np
```

```
learning_rate = 0.1
training_epochs = 200
batch_size = 128
logs_path = "TP2/MNIST_99_Challenge_Figures"
x = tf.placeholder(tf.float32, [None, 28, 28, 1], name='InputData')
# 0-9 digits recognition,  10 classes
y = tf.placeholder(tf.float32, [None, 10], name='LabelData')

x_adapted = tf.map_fn(lambda x1: tf.image.pad_to_bounding_box(x1, np.random.randint(4), np.random.ran
dint(4), 32, 32), x)
model = LeNet5_Model(x_adapted)

cross_entropy = tf.nn.softmax_cross_entropy_with_logits(logits=model, labels=y)
loss_operation = tf.reduce_mean(cross_entropy)
optimizer = tf.train.GradientDescentOptimizer(learning_rate = learning_rate)
training_operation = optimizer.minimize(loss_operation)
```

In [22]:

```
import time
timeInit = time.time()
train(X_train,y_train,X_validation,y_validation,X_test,y_test)
print('training time :',time.time()- timeInit,'seconds')
```

```
Start Training!
EPOCH 1 :
Training Accuracy = 0.967
Validation Accuracy = 0.971
EPOCH 2 :
Training Accuracy = 0.977
Validation Accuracy = 0.977
EPOCH 3 :
Training Accuracy = 0.981
Validation Accuracy = 0.980
EPOCH 4 :
Training Accuracy = 0.986
Validation Accuracy = 0.983
EPOCH 5 :
Training Accuracy = 0.989
Validation Accuracy = 0.986
EPOCH 6 :
Training Accuracy = 0.986
Validation Accuracy = 0.984
EPOCH 7 :
Training Accuracy = 0.990
Validation Accuracy = 0.986
EPOCH 8 :
Training Accuracy = 0.992
Validation Accuracy = 0.987
EPOCH 9 :
Training Accuracy = 0.994
Validation Accuracy = 0.989
EPOCH 10 :
Training Accuracy = 0.994
Validation Accuracy = 0.989
EPOCH 11 :
Training Accuracy = 0.996
Validation Accuracy = 0.990
EPOCH 12 :
Training Accuracy = 0.995
Validation Accuracy = 0.989
EPOCH 13 :
Training Accuracy = 0.997
Validation Accuracy = 0.991
EPOCH 14 :
```

Training Accuracy = 0.997
Validation Accuracy = 0.989
EPOCH 15 :
Training Accuracy = 0.997
Validation Accuracy = 0.990
EPOCH 16 :
Training Accuracy = 0.996
Validation Accuracy = 0.988
EPOCH 17 :
Training Accuracy = 0.997
Validation Accuracy = 0.988
EPOCH 18 :
Training Accuracy = 0.997
Validation Accuracy = 0.990
EPOCH 19 :
Training Accuracy = 0.998
Validation Accuracy = 0.990
EPOCH 20 :
Training Accuracy = 0.997
Validation Accuracy = 0.988
EPOCH 21 :
Training Accuracy = 0.999
Validation Accuracy = 0.991
EPOCH 22 :
Training Accuracy = 0.998
Validation Accuracy = 0.987
EPOCH 23 :
Training Accuracy = 0.999
Validation Accuracy = 0.991
EPOCH 24 :
Training Accuracy = 0.999
Validation Accuracy = 0.989
EPOCH 25 :
Training Accuracy = 0.998
Validation Accuracy = 0.991
EPOCH 26 :
Training Accuracy = 0.999
Validation Accuracy = 0.990
EPOCH 27 :
Training Accuracy = 0.998
Validation Accuracy = 0.987
EPOCH 28 :
Training Accuracy = 1.000
Validation Accuracy = 0.990
EPOCH 29 :
Training Accuracy = 1.000
Validation Accuracy = 0.990
EPOCH 30 :
Training Accuracy = 0.999
Validation Accuracy = 0.990
EPOCH 31 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 32 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 33 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 34 :
Training Accuracy = 1.000
Validation Accuracy = 0.990
EPOCH 35 :
Training Accuracy = 1.000

Validation Accuracy = 0.990
EPOCH 36 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 37 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 38 :
Training Accuracy = 1.000
Validation Accuracy = 0.990
EPOCH 39 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 40 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 41 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 42 :
Training Accuracy = 1.000
Validation Accuracy = 0.990
EPOCH 43 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 44 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 45 :
Training Accuracy = 1.000
Validation Accuracy = 0.990
EPOCH 46 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 47 :
Training Accuracy = 1.000
Validation Accuracy = 0.990
EPOCH 48 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 49 :
Training Accuracy = 1.000
Validation Accuracy = 0.990
EPOCH 50 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 51 :
Training Accuracy = 1.000
Validation Accuracy = 0.990
EPOCH 52 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 53 :
Training Accuracy = 1.000
Validation Accuracy = 0.990
EPOCH 54 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 55 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 56 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 57 :

Training Accuracy = 1.000
Validation Accuracy = 0.990
EPOCH 58 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 59 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 60 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 61 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 62 :
Training Accuracy = 1.000
Validation Accuracy = 0.990
EPOCH 63 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 64 :
Training Accuracy = 1.000
Validation Accuracy = 0.990
EPOCH 65 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 66 :
Training Accuracy = 1.000
Validation Accuracy = 0.990
EPOCH 67 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 68 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 69 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 70 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 71 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 72 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 73 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 74 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 75 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 76 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 77 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 78 :
Training Accuracy = 1.000

Validation Accuracy = 0.990
EPOCH 79 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 80 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 81 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 82 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 83 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 84 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 85 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 86 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 87 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 88 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 89 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 90 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 91 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 92 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 93 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 94 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 95 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 96 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 97 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 98 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 99 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 100 :

Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 101 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 102 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 103 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 104 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 105 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 106 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 107 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 108 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 109 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 110 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 111 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 112 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 113 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 114 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 115 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 116 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 117 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 118 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 119 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 120 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 121 :
Training Accuracy = 1.000

Validation Accuracy = 0.991
EPOCH 122 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 123 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 124 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 125 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 126 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 127 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 128 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 129 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 130 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 131 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 132 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 133 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 134 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 135 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 136 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 137 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 138 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 139 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 140 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 141 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 142 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 143 :

Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 144 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 145 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 146 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 147 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 148 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 149 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 150 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 151 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 152 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 153 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 154 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 155 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 156 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 157 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 158 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 159 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 160 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 161 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 162 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 163 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 164 :
Training Accuracy = 1.000

Validation Accuracy = 0.991
EPOCH 165 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 166 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 167 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 168 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 169 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 170 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 171 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 172 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 173 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 174 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 175 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 176 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 177 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 178 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 179 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 180 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 181 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 182 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 183 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 184 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 185 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 186 :

```
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 187 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 188 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 189 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 190 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 191 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 192 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 193 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 194 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 195 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 196 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 197 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 198 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 199 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 200 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
Optimization Finished!
Test Accuracy = 0.990
Training Finished!
Model saved
training time : 6281.19958901 seconds
```

AdamOptimiser, learning rate = 0.1 , Batch size : 128, Training epochs : 200

In [23]:

```
# AdamOptimiser, learning rate = 0.1 , Batch size : 128, Training epochs : 200

# Parameters
import numpy as np
learning_rate = 0.1
training_epochs = 200
```

```
batch_size = 128
logs_path = "TP2/MNIST_99_Challenge_Figures"
x = tf.placeholder(tf.float32, [None, 28, 28, 1], name='InputData')
# 0-9 digits recognition,  10 classes
y = tf.placeholder(tf.float32, [None, 10], name='LabelData')

x_adapted = tf.map_fn(lambda x1: tf.image.pad_to_bounding_box(x1, np.random.randint(4), np.random.ran
dint(4), 32, 32), x)
model = LeNet5_Model(x_adapted)

cross_entropy = tf.nn.softmax_cross_entropy_with_logits(logits=model, labels=y)
loss_operation = tf.reduce_mean(cross_entropy)
optimizer = tf.train.AdamOptimizer(learning_rate=0.1)
training_operation = optimizer.minimize(loss_operation)
```

In [26]:

```
import time
timeInit = time.time()
train(X_train,y_train,X_validation,y_validation,X_test,y_test)
print('training time :',time.time()- timeInit,'seconds')
```

```
Start Training!
EPOCH 1 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 2 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 3 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 4 :
Training Accuracy = 0.104
Validation Accuracy = 0.110
EPOCH 5 :
Training Accuracy = 0.099
Validation Accuracy = 0.096
EPOCH 6 :
Training Accuracy = 0.104
Validation Accuracy = 0.110
EPOCH 7 :
Training Accuracy = 0.099
Validation Accuracy = 0.099
EPOCH 8 :
Training Accuracy = 0.104
Validation Accuracy = 0.110
EPOCH 9 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 10 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 11 :
Training Accuracy = 0.103
Validation Accuracy = 0.099
EPOCH 12 :
Training Accuracy = 0.099
Validation Accuracy = 0.099
EPOCH 13 :
Training Accuracy = 0.099
Validation Accuracy = 0.098
EPOCH 14 :
Training Accuracy = 0.099
Validation Accuracy = 0.096
```

EPOCH 15 :
Training Accuracy = 0.098
Validation Accuracy = 0.100
EPOCH 16 :
Training Accuracy = 0.098
Validation Accuracy = 0.100
EPOCH 17 :
Training Accuracy = 0.103
Validation Accuracy = 0.099
EPOCH 18 :
Training Accuracy = 0.099
Validation Accuracy = 0.096
EPOCH 19 :
Training Accuracy = 0.103
Validation Accuracy = 0.099
EPOCH 20 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 21 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 22 :
Training Accuracy = 0.096
Validation Accuracy = 0.107
EPOCH 23 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 24 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 25 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 26 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 27 :
Training Accuracy = 0.098
Validation Accuracy = 0.092
EPOCH 28 :
Training Accuracy = 0.103
Validation Accuracy = 0.099
EPOCH 29 :
Training Accuracy = 0.099
Validation Accuracy = 0.098
EPOCH 30 :
Training Accuracy = 0.098
Validation Accuracy = 0.100
EPOCH 31 :
Training Accuracy = 0.099
Validation Accuracy = 0.098
EPOCH 32 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 33 :
Training Accuracy = 0.099
Validation Accuracy = 0.099
EPOCH 34 :
Training Accuracy = 0.099
Validation Accuracy = 0.096
EPOCH 35 :
Training Accuracy = 0.098
Validation Accuracy = 0.100
EPOCH 36 :

Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 37 :
Training Accuracy = 0.099
Validation Accuracy = 0.099
EPOCH 38 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 39 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 40 :
Training Accuracy = 0.098
Validation Accuracy = 0.100
EPOCH 41 :
Training Accuracy = 0.103
Validation Accuracy = 0.099
EPOCH 42 :
Training Accuracy = 0.103
Validation Accuracy = 0.099
EPOCH 43 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 44 :
Training Accuracy = 0.096
Validation Accuracy = 0.107
EPOCH 45 :
Training Accuracy = 0.098
Validation Accuracy = 0.100
EPOCH 46 :
Training Accuracy = 0.103
Validation Accuracy = 0.099
EPOCH 47 :
Training Accuracy = 0.099
Validation Accuracy = 0.096
EPOCH 48 :
Training Accuracy = 0.098
Validation Accuracy = 0.100
EPOCH 49 :
Training Accuracy = 0.099
Validation Accuracy = 0.096
EPOCH 50 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 51 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 52 :
Training Accuracy = 0.098
Validation Accuracy = 0.100
EPOCH 53 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 54 :
Training Accuracy = 0.098
Validation Accuracy = 0.100
EPOCH 55 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 56 :
Training Accuracy = 0.104
Validation Accuracy = 0.110
EPOCH 57 :
Training Accuracy = 0.112
Validation Accuracy = 0.113

EPOCH 58 :
Training Accuracy = 0.098
Validation Accuracy = 0.092
EPOCH 59 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 60 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 61 :
Training Accuracy = 0.104
Validation Accuracy = 0.110
EPOCH 62 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 63 :
Training Accuracy = 0.104
Validation Accuracy = 0.110
EPOCH 64 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 65 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 66 :
Training Accuracy = 0.103
Validation Accuracy = 0.099
EPOCH 67 :
Training Accuracy = 0.099
Validation Accuracy = 0.096
EPOCH 68 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 69 :
Training Accuracy = 0.096
Validation Accuracy = 0.107
EPOCH 70 :
Training Accuracy = 0.099
Validation Accuracy = 0.096
EPOCH 71 :
Training Accuracy = 0.099
Validation Accuracy = 0.099
EPOCH 72 :
Training Accuracy = 0.099
Validation Accuracy = 0.096
EPOCH 73 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 74 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 75 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 76 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 77 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 78 :
Training Accuracy = 0.099
Validation Accuracy = 0.098
EPOCH 79 :

Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 80 :
Training Accuracy = 0.103
Validation Accuracy = 0.099
EPOCH 81 :
Training Accuracy = 0.103
Validation Accuracy = 0.099
EPOCH 82 :
Training Accuracy = 0.098
Validation Accuracy = 0.100
EPOCH 83 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 84 :
Training Accuracy = 0.099
Validation Accuracy = 0.099
EPOCH 85 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 86 :
Training Accuracy = 0.103
Validation Accuracy = 0.099
EPOCH 87 :
Training Accuracy = 0.098
Validation Accuracy = 0.100
EPOCH 88 :
Training Accuracy = 0.098
Validation Accuracy = 0.100
EPOCH 89 :
Training Accuracy = 0.104
Validation Accuracy = 0.110
EPOCH 90 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 91 :
Training Accuracy = 0.099
Validation Accuracy = 0.099
EPOCH 92 :
Training Accuracy = 0.104
Validation Accuracy = 0.110
EPOCH 93 :
Training Accuracy = 0.103
Validation Accuracy = 0.099
EPOCH 94 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 95 :
Training Accuracy = 0.098
Validation Accuracy = 0.100
EPOCH 96 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 97 :
Training Accuracy = 0.099
Validation Accuracy = 0.096
EPOCH 98 :
Training Accuracy = 0.099
Validation Accuracy = 0.098
EPOCH 99 :
Training Accuracy = 0.091
Validation Accuracy = 0.087
EPOCH 100 :
Training Accuracy = 0.099
Validation Accuracy = 0.098

EPOCH 101 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 102 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 103 :
Training Accuracy = 0.103
Validation Accuracy = 0.099
EPOCH 104 :
Training Accuracy = 0.099
Validation Accuracy = 0.096
EPOCH 105 :
Training Accuracy = 0.096
Validation Accuracy = 0.107
EPOCH 106 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 107 :
Training Accuracy = 0.099
Validation Accuracy = 0.096
EPOCH 108 :
Training Accuracy = 0.099
Validation Accuracy = 0.096
EPOCH 109 :
Training Accuracy = 0.103
Validation Accuracy = 0.099
EPOCH 110 :
Training Accuracy = 0.099
Validation Accuracy = 0.099
EPOCH 111 :
Training Accuracy = 0.091
Validation Accuracy = 0.087
EPOCH 112 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 113 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 114 :
Training Accuracy = 0.098
Validation Accuracy = 0.092
EPOCH 115 :
Training Accuracy = 0.091
Validation Accuracy = 0.087
EPOCH 116 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 117 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 118 :
Training Accuracy = 0.099
Validation Accuracy = 0.098
EPOCH 119 :
Training Accuracy = 0.099
Validation Accuracy = 0.096
EPOCH 120 :
Training Accuracy = 0.099
Validation Accuracy = 0.096
EPOCH 121 :
Training Accuracy = 0.098
Validation Accuracy = 0.092
EPOCH 122 :

Training Accuracy = 0.103
Validation Accuracy = 0.099
EPOCH 123 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 124 :
Training Accuracy = 0.098
Validation Accuracy = 0.092
EPOCH 125 :
Training Accuracy = 0.098
Validation Accuracy = 0.100
EPOCH 126 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 127 :
Training Accuracy = 0.104
Validation Accuracy = 0.110
EPOCH 128 :
Training Accuracy = 0.099
Validation Accuracy = 0.098
EPOCH 129 :
Training Accuracy = 0.103
Validation Accuracy = 0.099
EPOCH 130 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 131 :
Training Accuracy = 0.099
Validation Accuracy = 0.096
EPOCH 132 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 133 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 134 :
Training Accuracy = 0.098
Validation Accuracy = 0.100
EPOCH 135 :
Training Accuracy = 0.099
Validation Accuracy = 0.099
EPOCH 136 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 137 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 138 :
Training Accuracy = 0.104
Validation Accuracy = 0.110
EPOCH 139 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 140 :
Training Accuracy = 0.103
Validation Accuracy = 0.099
EPOCH 141 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 142 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 143 :
Training Accuracy = 0.099
Validation Accuracy = 0.096

EPOCH 144 :
Training Accuracy = 0.098
Validation Accuracy = 0.100
EPOCH 145 :
Training Accuracy = 0.104
Validation Accuracy = 0.110
EPOCH 146 :
Training Accuracy = 0.104
Validation Accuracy = 0.110
EPOCH 147 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 148 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 149 :
Training Accuracy = 0.099
Validation Accuracy = 0.096
EPOCH 150 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 151 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 152 :
Training Accuracy = 0.099
Validation Accuracy = 0.099
EPOCH 153 :
Training Accuracy = 0.099
Validation Accuracy = 0.098
EPOCH 154 :
Training Accuracy = 0.099
Validation Accuracy = 0.098
EPOCH 155 :
Training Accuracy = 0.104
Validation Accuracy = 0.110
EPOCH 156 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 157 :
Training Accuracy = 0.104
Validation Accuracy = 0.110
EPOCH 158 :
Training Accuracy = 0.099
Validation Accuracy = 0.096
EPOCH 159 :
Training Accuracy = 0.099
Validation Accuracy = 0.099
EPOCH 160 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 161 :
Training Accuracy = 0.104
Validation Accuracy = 0.110
EPOCH 162 :
Training Accuracy = 0.099
Validation Accuracy = 0.098
EPOCH 163 :
Training Accuracy = 0.103
Validation Accuracy = 0.099
EPOCH 164 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 165 :

Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 166 :
Training Accuracy = 0.104
Validation Accuracy = 0.110
EPOCH 167 :
Training Accuracy = 0.098
Validation Accuracy = 0.092
EPOCH 168 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 169 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 170 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 171 :
Training Accuracy = 0.099
Validation Accuracy = 0.099
EPOCH 172 :
Training Accuracy = 0.098
Validation Accuracy = 0.092
EPOCH 173 :
Training Accuracy = 0.103
Validation Accuracy = 0.099
EPOCH 174 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 175 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 176 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 177 :
Training Accuracy = 0.099
Validation Accuracy = 0.096
EPOCH 178 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 179 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 180 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 181 :
Training Accuracy = 0.096
Validation Accuracy = 0.107
EPOCH 182 :
Training Accuracy = 0.104
Validation Accuracy = 0.110
EPOCH 183 :
Training Accuracy = 0.099
Validation Accuracy = 0.096
EPOCH 184 :
Training Accuracy = 0.099
Validation Accuracy = 0.099
EPOCH 185 :
Training Accuracy = 0.104
Validation Accuracy = 0.110
EPOCH 186 :
Training Accuracy = 0.104
Validation Accuracy = 0.110

```
EPOCH 187 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 188 :
Training Accuracy = 0.096
Validation Accuracy = 0.107
EPOCH 189 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 190 :
Training Accuracy = 0.098
Validation Accuracy = 0.092
EPOCH 191 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 192 :
Training Accuracy = 0.099
Validation Accuracy = 0.098
EPOCH 193 :
Training Accuracy = 0.104
Validation Accuracy = 0.110
EPOCH 194 :
Training Accuracy = 0.099
Validation Accuracy = 0.096
EPOCH 195 :
Training Accuracy = 0.091
Validation Accuracy = 0.087
EPOCH 196 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 197 :
Training Accuracy = 0.112
Validation Accuracy = 0.113
EPOCH 198 :
Training Accuracy = 0.099
Validation Accuracy = 0.099
EPOCH 199 :
Training Accuracy = 0.099
Validation Accuracy = 0.096
EPOCH 200 :
Training Accuracy = 0.104
Validation Accuracy = 0.110
Optimization Finished!
Test Accuracy = 0.103
Training Finished!
Model saved
training time : 6256.76132393 seconds
```

Trying each optimizer with different learning rates and different Batch sizes

AdamOptimiser, learning rate = 0.0001 , Batch size : 50, Training epochs : 100

In [27]:

```
# AdamOptimiser, learning rate = 0.0001 , Batch size : 50, Training epochs : 100

# Parameters
```

```
import numpy as np
learning_rate = 0.0001
training_epochs = 100
batch_size = 50
logs_path = "TP2/MNIST_99_Challenge_Figures"
x = tf.placeholder(tf.float32, [None, 28, 28, 1], name='InputData')
# 0-9 digits recognition,  10 classes
y = tf.placeholder(tf.float32, [None, 10], name='LabelData')

x_adapted = tf.map_fn(lambda x1: tf.image.pad_to_bounding_box(x1, np.random.randint(4), np.random.ran
dint(4), 32, 32), x)
model = LeNet5_Model(x_adapted)

cross_entropy = tf.nn.softmax_cross_entropy_with_logits(logits=model, labels=y)
loss_operation = tf.reduce_mean(cross_entropy)
optimizer = tf.train.AdamOptimizer(learning_rate=0.0001)
training_operation = optimizer.minimize(loss_operation)
```

In [30]:

```
import time
timeInit = time.time()
train(X_train,y_train,X_validation,y_validation,X_test,y_test)
print('training time :',time.time()- timeInit,'seconds')
```

```
Start Training!
EPOCH 1 :
Training Accuracy = 0.884
Validation Accuracy = 0.887
EPOCH 2 :
Training Accuracy = 0.920
Validation Accuracy = 0.924
EPOCH 3 :
Training Accuracy = 0.938
Validation Accuracy = 0.943
EPOCH 4 :
Training Accuracy = 0.949
Validation Accuracy = 0.952
EPOCH 5 :
Training Accuracy = 0.957
Validation Accuracy = 0.961
EPOCH 6 :
Training Accuracy = 0.962
Validation Accuracy = 0.965
EPOCH 7 :
Training Accuracy = 0.965
Validation Accuracy = 0.968
EPOCH 8 :
Training Accuracy = 0.968
Validation Accuracy = 0.972
EPOCH 9 :
Training Accuracy = 0.972
Validation Accuracy = 0.971
EPOCH 10 :
Training Accuracy = 0.974
Validation Accuracy = 0.974
EPOCH 11 :
Training Accuracy = 0.974
Validation Accuracy = 0.974
EPOCH 12 :
Training Accuracy = 0.976
Validation Accuracy = 0.975
EPOCH 13 :
Training Accuracy = 0.978
```

Validation Accuracy = 0.977
EPOCH 14 :
Training Accuracy = 0.979
Validation Accuracy = 0.977
EPOCH 15 :
Training Accuracy = 0.977
Validation Accuracy = 0.977
EPOCH 16 :
Training Accuracy = 0.979
Validation Accuracy = 0.979
EPOCH 17 :
Training Accuracy = 0.981
Validation Accuracy = 0.979
EPOCH 18 :
Training Accuracy = 0.982
Validation Accuracy = 0.980
EPOCH 19 :
Training Accuracy = 0.983
Validation Accuracy = 0.982
EPOCH 20 :
Training Accuracy = 0.983
Validation Accuracy = 0.981
EPOCH 21 :
Training Accuracy = 0.985
Validation Accuracy = 0.982
EPOCH 22 :
Training Accuracy = 0.984
Validation Accuracy = 0.981
EPOCH 23 :
Training Accuracy = 0.983
Validation Accuracy = 0.981
EPOCH 24 :
Training Accuracy = 0.986
Validation Accuracy = 0.983
EPOCH 25 :
Training Accuracy = 0.987
Validation Accuracy = 0.985
EPOCH 26 :
Training Accuracy = 0.986
Validation Accuracy = 0.983
EPOCH 27 :
Training Accuracy = 0.987
Validation Accuracy = 0.984
EPOCH 28 :
Training Accuracy = 0.984
Validation Accuracy = 0.984
EPOCH 29 :
Training Accuracy = 0.988
Validation Accuracy = 0.983
EPOCH 30 :
Training Accuracy = 0.989
Validation Accuracy = 0.985
EPOCH 31 :
Training Accuracy = 0.987
Validation Accuracy = 0.982
EPOCH 32 :
Training Accuracy = 0.989
Validation Accuracy = 0.984
EPOCH 33 :
Training Accuracy = 0.988
Validation Accuracy = 0.984
EPOCH 34 :
Training Accuracy = 0.989
Validation Accuracy = 0.985
EPOCH 35 :

Training Accuracy = 0.989
Validation Accuracy = 0.983
EPOCH 36 :
Training Accuracy = 0.990
Validation Accuracy = 0.986
EPOCH 37 :
Training Accuracy = 0.991
Validation Accuracy = 0.985
EPOCH 38 :
Training Accuracy = 0.990
Validation Accuracy = 0.986
EPOCH 39 :
Training Accuracy = 0.990
Validation Accuracy = 0.985
EPOCH 40 :
Training Accuracy = 0.992
Validation Accuracy = 0.986
EPOCH 41 :
Training Accuracy = 0.992
Validation Accuracy = 0.985
EPOCH 42 :
Training Accuracy = 0.991
Validation Accuracy = 0.985
EPOCH 43 :
Training Accuracy = 0.992
Validation Accuracy = 0.984
EPOCH 44 :
Training Accuracy = 0.993
Validation Accuracy = 0.984
EPOCH 45 :
Training Accuracy = 0.993
Validation Accuracy = 0.986
EPOCH 46 :
Training Accuracy = 0.993
Validation Accuracy = 0.985
EPOCH 47 :
Training Accuracy = 0.994
Validation Accuracy = 0.986
EPOCH 48 :
Training Accuracy = 0.993
Validation Accuracy = 0.987
EPOCH 49 :
Training Accuracy = 0.992
Validation Accuracy = 0.987
EPOCH 50 :
Training Accuracy = 0.993
Validation Accuracy = 0.989
EPOCH 51 :
Training Accuracy = 0.994
Validation Accuracy = 0.987
EPOCH 52 :
Training Accuracy = 0.994
Validation Accuracy = 0.987
EPOCH 53 :
Training Accuracy = 0.995
Validation Accuracy = 0.987
EPOCH 54 :
Training Accuracy = 0.994
Validation Accuracy = 0.986
EPOCH 55 :
Training Accuracy = 0.995
Validation Accuracy = 0.988
EPOCH 56 :
Training Accuracy = 0.995

Validation Accuracy = 0.986
EPOCH 57 :
Training Accuracy = 0.994
Validation Accuracy = 0.988
EPOCH 58 :
Training Accuracy = 0.995
Validation Accuracy = 0.987
EPOCH 59 :
Training Accuracy = 0.996
Validation Accuracy = 0.988
EPOCH 60 :
Training Accuracy = 0.995
Validation Accuracy = 0.988
EPOCH 61 :
Training Accuracy = 0.996
Validation Accuracy = 0.987
EPOCH 62 :
Training Accuracy = 0.991
Validation Accuracy = 0.984
EPOCH 63 :
Training Accuracy = 0.995
Validation Accuracy = 0.988
EPOCH 64 :
Training Accuracy = 0.996
Validation Accuracy = 0.987
EPOCH 65 :
Training Accuracy = 0.996
Validation Accuracy = 0.987
EPOCH 66 :
Training Accuracy = 0.997
Validation Accuracy = 0.988
EPOCH 67 :
Training Accuracy = 0.997
Validation Accuracy = 0.989
EPOCH 68 :
Training Accuracy = 0.998
Validation Accuracy = 0.988
EPOCH 69 :
Training Accuracy = 0.996
Validation Accuracy = 0.988
EPOCH 70 :
Training Accuracy = 0.995
Validation Accuracy = 0.987
EPOCH 71 :
Training Accuracy = 0.997
Validation Accuracy = 0.989
EPOCH 72 :
Training Accuracy = 0.998
Validation Accuracy = 0.988
EPOCH 73 :
Training Accuracy = 0.998
Validation Accuracy = 0.989
EPOCH 74 :
Training Accuracy = 0.997
Validation Accuracy = 0.988
EPOCH 75 :
Training Accuracy = 0.997
Validation Accuracy = 0.987
EPOCH 76 :
Training Accuracy = 0.998
Validation Accuracy = 0.988
EPOCH 77 :
Training Accuracy = 0.998
Validation Accuracy = 0.988
EPOCH 78 :

Training Accuracy = 0.998
Validation Accuracy = 0.988
EPOCH 79 :
Training Accuracy = 0.998
Validation Accuracy = 0.986
EPOCH 80 :
Training Accuracy = 0.998
Validation Accuracy = 0.988
EPOCH 81 :
Training Accuracy = 0.996
Validation Accuracy = 0.986
EPOCH 82 :
Training Accuracy = 0.998
Validation Accuracy = 0.989
EPOCH 83 :
Training Accuracy = 0.998
Validation Accuracy = 0.988
EPOCH 84 :
Training Accuracy = 0.998
Validation Accuracy = 0.986
EPOCH 85 :
Training Accuracy = 0.998
Validation Accuracy = 0.986
EPOCH 86 :
Training Accuracy = 0.999
Validation Accuracy = 0.989
EPOCH 87 :
Training Accuracy = 0.999
Validation Accuracy = 0.988
EPOCH 88 :
Training Accuracy = 0.999
Validation Accuracy = 0.989
EPOCH 89 :
Training Accuracy = 0.999
Validation Accuracy = 0.988
EPOCH 90 :
Training Accuracy = 0.999
Validation Accuracy = 0.987
EPOCH 91 :
Training Accuracy = 0.998
Validation Accuracy = 0.986
EPOCH 92 :
Training Accuracy = 0.999
Validation Accuracy = 0.989
EPOCH 93 :
Training Accuracy = 0.998
Validation Accuracy = 0.989
EPOCH 94 :
Training Accuracy = 0.997
Validation Accuracy = 0.987
EPOCH 95 :
Training Accuracy = 0.999
Validation Accuracy = 0.987
EPOCH 96 :
Training Accuracy = 0.999
Validation Accuracy = 0.988
EPOCH 97 :
Training Accuracy = 0.999
Validation Accuracy = 0.990
EPOCH 98 :
Training Accuracy = 0.999
Validation Accuracy = 0.989
EPOCH 99 :
Training Accuracy = 0.999

```
Validation Accuracy = 0.989
EPOCH 100 :
Training Accuracy = 0.998
Validation Accuracy = 0.988
Optimization Finished!
Test Accuracy = 0.988
Training Finished!
Model saved
training time : 3217.85242987 seconds
```

AdamOptimiser, learning rate = 0.0001 , Batch size : 128, Training epochs : 100

In [31]:

```
# AdamOptimiser, learning rate = 0.0001 , Batch size : 128, Training epochs : 100

# Parameters
import numpy as np
learning_rate = 0.0001
training_epochs = 100
batch_size = 128
logs_path = "TP2/MNIST_99_Challenge_Figures"
x = tf.placeholder(tf.float32, [None, 28, 28, 1], name='InputData')
# 0-9 digits recognition,  10 classes
y = tf.placeholder(tf.float32, [None, 10], name='LabelData')

x_adapted = tf.map_fn(lambda x1: tf.image.pad_to_bounding_box(x1, np.random.randint(4), np.random.ran
dint(4), 32, 32), x)
model = LeNet5_Model(x_adapted)

cross_entropy = tf.nn.softmax_cross_entropy_with_logits(logits=model, labels=y)
loss_operation = tf.reduce_mean(cross_entropy)
optimizer = tf.train.AdamOptimizer(learning_rate=0.0001)
training_operation = optimizer.minimize(loss_operation)
```

In [34]:

```
import time
timeInit = time.time()
train(X_train,y_train,X_validation,y_validation,X_test,y_test)
print('training time :',time.time()- timeInit,'seconds')
```

```
Start Training!
EPOCH 1 :
Training Accuracy = 0.891
Validation Accuracy = 0.892
EPOCH 2 :
Training Accuracy = 0.919
Validation Accuracy = 0.921
EPOCH 3 :
Training Accuracy = 0.936
Validation Accuracy = 0.940
EPOCH 4 :
Training Accuracy = 0.950
Validation Accuracy = 0.954
EPOCH 5 :
Training Accuracy = 0.953
Validation Accuracy = 0.958
EPOCH 6 :
Training Accuracy = 0.961
Validation Accuracy = 0.968
```

EPOCH 7 :
Training Accuracy = 0.966
Validation Accuracy = 0.971
EPOCH 8 :
Training Accuracy = 0.968
Validation Accuracy = 0.974
EPOCH 9 :
Training Accuracy = 0.970
Validation Accuracy = 0.973
EPOCH 10 :
Training Accuracy = 0.974
Validation Accuracy = 0.977
EPOCH 11 :
Training Accuracy = 0.975
Validation Accuracy = 0.977
EPOCH 12 :
Training Accuracy = 0.977
Validation Accuracy = 0.978
EPOCH 13 :
Training Accuracy = 0.979
Validation Accuracy = 0.980
EPOCH 14 :
Training Accuracy = 0.980
Validation Accuracy = 0.982
EPOCH 15 :
Training Accuracy = 0.981
Validation Accuracy = 0.980
EPOCH 16 :
Training Accuracy = 0.983
Validation Accuracy = 0.981
EPOCH 17 :
Training Accuracy = 0.982
Validation Accuracy = 0.981
EPOCH 18 :
Training Accuracy = 0.983
Validation Accuracy = 0.982
EPOCH 19 :
Training Accuracy = 0.984
Validation Accuracy = 0.981
EPOCH 20 :
Training Accuracy = 0.986
Validation Accuracy = 0.984
EPOCH 21 :
Training Accuracy = 0.986
Validation Accuracy = 0.983
EPOCH 22 :
Training Accuracy = 0.987
Validation Accuracy = 0.983
EPOCH 23 :
Training Accuracy = 0.986
Validation Accuracy = 0.983
EPOCH 24 :
Training Accuracy = 0.988
Validation Accuracy = 0.983
EPOCH 25 :
Training Accuracy = 0.988
Validation Accuracy = 0.984
EPOCH 26 :
Training Accuracy = 0.988
Validation Accuracy = 0.984
EPOCH 27 :
Training Accuracy = 0.990
Validation Accuracy = 0.985
EPOCH 28 :

Training Accuracy = 0.990
Validation Accuracy = 0.985
EPOCH 29 :
Training Accuracy = 0.989
Validation Accuracy = 0.985
EPOCH 30 :
Training Accuracy = 0.990
Validation Accuracy = 0.986
EPOCH 31 :
Training Accuracy = 0.991
Validation Accuracy = 0.986
EPOCH 32 :
Training Accuracy = 0.991
Validation Accuracy = 0.985
EPOCH 33 :
Training Accuracy = 0.991
Validation Accuracy = 0.984
EPOCH 34 :
Training Accuracy = 0.991
Validation Accuracy = 0.985
EPOCH 35 :
Training Accuracy = 0.991
Validation Accuracy = 0.986
EPOCH 36 :
Training Accuracy = 0.992
Validation Accuracy = 0.987
EPOCH 37 :
Training Accuracy = 0.992
Validation Accuracy = 0.985
EPOCH 38 :
Training Accuracy = 0.993
Validation Accuracy = 0.987
EPOCH 39 :
Training Accuracy = 0.994
Validation Accuracy = 0.987
EPOCH 40 :
Training Accuracy = 0.993
Validation Accuracy = 0.986
EPOCH 41 :
Training Accuracy = 0.994
Validation Accuracy = 0.987
EPOCH 42 :
Training Accuracy = 0.994
Validation Accuracy = 0.987
EPOCH 43 :
Training Accuracy = 0.994
Validation Accuracy = 0.987
EPOCH 44 :
Training Accuracy = 0.995
Validation Accuracy = 0.987
EPOCH 45 :
Training Accuracy = 0.994
Validation Accuracy = 0.986
EPOCH 46 :
Training Accuracy = 0.996
Validation Accuracy = 0.987
EPOCH 47 :
Training Accuracy = 0.994
Validation Accuracy = 0.987
EPOCH 48 :
Training Accuracy = 0.994
Validation Accuracy = 0.987
EPOCH 49 :
Training Accuracy = 0.994
Validation Accuracy = 0.986

EPOCH 50 :
Training Accuracy = 0.994
Validation Accuracy = 0.987
EPOCH 51 :
Training Accuracy = 0.996
Validation Accuracy = 0.988
EPOCH 52 :
Training Accuracy = 0.996
Validation Accuracy = 0.987
EPOCH 53 :
Training Accuracy = 0.996
Validation Accuracy = 0.987
EPOCH 54 :
Training Accuracy = 0.996
Validation Accuracy = 0.987
EPOCH 55 :
Training Accuracy = 0.996
Validation Accuracy = 0.986
EPOCH 56 :
Training Accuracy = 0.996
Validation Accuracy = 0.988
EPOCH 57 :
Training Accuracy = 0.997
Validation Accuracy = 0.987
EPOCH 58 :
Training Accuracy = 0.997
Validation Accuracy = 0.987
EPOCH 59 :
Training Accuracy = 0.997
Validation Accuracy = 0.987
EPOCH 60 :
Training Accuracy = 0.998
Validation Accuracy = 0.989
EPOCH 61 :
Training Accuracy = 0.997
Validation Accuracy = 0.988
EPOCH 62 :
Training Accuracy = 0.997
Validation Accuracy = 0.987
EPOCH 63 :
Training Accuracy = 0.998
Validation Accuracy = 0.987
EPOCH 64 :
Training Accuracy = 0.997
Validation Accuracy = 0.986
EPOCH 65 :
Training Accuracy = 0.998
Validation Accuracy = 0.989
EPOCH 66 :
Training Accuracy = 0.998
Validation Accuracy = 0.986
EPOCH 67 :
Training Accuracy = 0.998
Validation Accuracy = 0.987
EPOCH 68 :
Training Accuracy = 0.999
Validation Accuracy = 0.988
EPOCH 69 :
Training Accuracy = 0.998
Validation Accuracy = 0.988
EPOCH 70 :
Training Accuracy = 0.998
Validation Accuracy = 0.988
EPOCH 71 :

Training Accuracy = 0.998
Validation Accuracy = 0.987
EPOCH 72 :
Training Accuracy = 0.998
Validation Accuracy = 0.988
EPOCH 73 :
Training Accuracy = 0.998
Validation Accuracy = 0.988
EPOCH 74 :
Training Accuracy = 0.999
Validation Accuracy = 0.988
EPOCH 75 :
Training Accuracy = 0.998
Validation Accuracy = 0.988
EPOCH 76 :
Training Accuracy = 0.996
Validation Accuracy = 0.987
EPOCH 77 :
Training Accuracy = 0.998
Validation Accuracy = 0.988
EPOCH 78 :
Training Accuracy = 0.999
Validation Accuracy = 0.987
EPOCH 79 :
Training Accuracy = 0.999
Validation Accuracy = 0.989
EPOCH 80 :
Training Accuracy = 0.998
Validation Accuracy = 0.986
EPOCH 81 :
Training Accuracy = 0.999
Validation Accuracy = 0.987
EPOCH 82 :
Training Accuracy = 0.999
Validation Accuracy = 0.988
EPOCH 83 :
Training Accuracy = 0.997
Validation Accuracy = 0.987
EPOCH 84 :
Training Accuracy = 0.998
Validation Accuracy = 0.986
EPOCH 85 :
Training Accuracy = 0.999
Validation Accuracy = 0.988
EPOCH 86 :
Training Accuracy = 0.999
Validation Accuracy = 0.987
EPOCH 87 :
Training Accuracy = 0.999
Validation Accuracy = 0.987
EPOCH 88 :
Training Accuracy = 0.999
Validation Accuracy = 0.988
EPOCH 89 :
Training Accuracy = 0.999
Validation Accuracy = 0.988
EPOCH 90 :
Training Accuracy = 0.999
Validation Accuracy = 0.988
EPOCH 91 :
Training Accuracy = 0.999
Validation Accuracy = 0.989
EPOCH 92 :
Training Accuracy = 0.999
Validation Accuracy = 0.989

```
EPOCH 93 :
Training Accuracy = 0.999
Validation Accuracy = 0.989
EPOCH 94 :
Training Accuracy = 0.999
Validation Accuracy = 0.987
EPOCH 95 :
Training Accuracy = 0.999
Validation Accuracy = 0.989
EPOCH 96 :
Training Accuracy = 0.999
Validation Accuracy = 0.988
EPOCH 97 :
Training Accuracy = 0.999
Validation Accuracy = 0.988
EPOCH 98 :
Training Accuracy = 0.999
Validation Accuracy = 0.989
EPOCH 99 :
Training Accuracy = 0.999
Validation Accuracy = 0.988
EPOCH 100 :
Training Accuracy = 1.000
Validation Accuracy = 0.989
Optimization Finished!
Test Accuracy = 0.988
Training Finished!
Model saved
training time : 3201.62925601 seconds
```

AdamOptimiser, learning rate = 0.001 , Batch size : 128, Training epochs : 100

In [35]:

```
# AdamOptimiser, learning rate = 0.001 , Batch size : 128, Training epochs : 100

# Parameters
import numpy as np
learning_rate = 0.001
training_epochs = 100
batch_size = 128
logs_path = "TP2/MNIST_99_Challenge_Figures"
x = tf.placeholder(tf.float32, [None, 28, 28, 1], name='InputData')
# 0-9 digits recognition,  10 classes
y = tf.placeholder(tf.float32, [None, 10], name='LabelData')

x_adapted = tf.map_fn(lambda x1: tf.image.pad_to_bounding_box(x1, np.random.randint(4), np.random.ran
dint(4), 32, 32), x)
model = LeNet5_Model(x_adapted)

cross_entropy = tf.nn.softmax_cross_entropy_with_logits(logits=model, labels=y)
loss_operation = tf.reduce_mean(cross_entropy)
optimizer = tf.train.AdamOptimizer(learning_rate=0.001)
training_operation = optimizer.minimize(loss_operation)
```

In [38]:

```
import time
timeInit = time.time()
train(X_train,y_train,X_validation,y_validation,X_test,y_test)
```

```
print('training time :',time.time()- timeInit, 'seconds')
```

```
Start Training!
EPOCH 1 :
Training Accuracy = 0.969
Validation Accuracy = 0.970
EPOCH 2 :
Training Accuracy = 0.982
Validation Accuracy = 0.980
EPOCH 3 :
Training Accuracy = 0.984
Validation Accuracy = 0.983
EPOCH 4 :
Training Accuracy = 0.987
Validation Accuracy = 0.982
EPOCH 5 :
Training Accuracy = 0.991
Validation Accuracy = 0.986
EPOCH 6 :
Training Accuracy = 0.991
Validation Accuracy = 0.986
EPOCH 7 :
Training Accuracy = 0.993
Validation Accuracy = 0.987
EPOCH 8 :
Training Accuracy = 0.994
Validation Accuracy = 0.987
EPOCH 9 :
Training Accuracy = 0.994
Validation Accuracy = 0.986
EPOCH 10 :
Training Accuracy = 0.996
Validation Accuracy = 0.990
EPOCH 11 :
Training Accuracy = 0.997
Validation Accuracy = 0.989
EPOCH 12 :
Training Accuracy = 0.995
Validation Accuracy = 0.987
EPOCH 13 :
Training Accuracy = 0.996
Validation Accuracy = 0.988
EPOCH 14 :
Training Accuracy = 0.997
Validation Accuracy = 0.989
EPOCH 15 :
Training Accuracy = 0.997
Validation Accuracy = 0.989
EPOCH 16 :
Training Accuracy = 0.999
Validation Accuracy = 0.992
EPOCH 17 :
Training Accuracy = 0.999
Validation Accuracy = 0.990
EPOCH 18 :
Training Accuracy = 0.998
Validation Accuracy = 0.990
EPOCH 19 :
Training Accuracy = 0.996
Validation Accuracy = 0.988
EPOCH 20 :
Training Accuracy = 0.998
Validation Accuracy = 0.989
EPOCH 21 :
Training Accuracy = 0.999
```

Validation Accuracy = 0.991
EPOCH 22 :
Training Accuracy = 0.999
Validation Accuracy = 0.990
EPOCH 23 :
Training Accuracy = 0.998
Validation Accuracy = 0.988
EPOCH 24 :
Training Accuracy = 0.998
Validation Accuracy = 0.990
EPOCH 25 :
Training Accuracy = 0.999
Validation Accuracy = 0.991
EPOCH 26 :
Training Accuracy = 0.997
Validation Accuracy = 0.987
EPOCH 27 :
Training Accuracy = 0.999
Validation Accuracy = 0.990
EPOCH 28 :
Training Accuracy = 0.999
Validation Accuracy = 0.991
EPOCH 29 :
Training Accuracy = 0.999
Validation Accuracy = 0.990
EPOCH 30 :
Training Accuracy = 1.000
Validation Accuracy = 0.989
EPOCH 31 :
Training Accuracy = 0.999
Validation Accuracy = 0.988
EPOCH 32 :
Training Accuracy = 0.999
Validation Accuracy = 0.989
EPOCH 33 :
Training Accuracy = 0.999
Validation Accuracy = 0.991
EPOCH 34 :
Training Accuracy = 0.999
Validation Accuracy = 0.990
EPOCH 35 :
Training Accuracy = 0.998
Validation Accuracy = 0.990
EPOCH 36 :
Training Accuracy = 0.999
Validation Accuracy = 0.991
EPOCH 37 :
Training Accuracy = 0.999
Validation Accuracy = 0.987
EPOCH 38 :
Training Accuracy = 0.999
Validation Accuracy = 0.991
EPOCH 39 :
Training Accuracy = 0.999
Validation Accuracy = 0.990
EPOCH 40 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 41 :
Training Accuracy = 0.998
Validation Accuracy = 0.989
EPOCH 42 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 43 :

EPOCH 43 :
Training Accuracy = 0.999
Validation Accuracy = 0.990
EPOCH 44 :
Training Accuracy = 0.999
Validation Accuracy = 0.989
EPOCH 45 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 46 :
Training Accuracy = 0.996
Validation Accuracy = 0.988
EPOCH 47 :
Training Accuracy = 1.000
Validation Accuracy = 0.990
EPOCH 48 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 49 :
Training Accuracy = 0.999
Validation Accuracy = 0.989
EPOCH 50 :
Training Accuracy = 1.000
Validation Accuracy = 0.989
EPOCH 51 :
Training Accuracy = 0.999
Validation Accuracy = 0.989
EPOCH 52 :
Training Accuracy = 0.999
Validation Accuracy = 0.991
EPOCH 53 :
Training Accuracy = 0.999
Validation Accuracy = 0.990
EPOCH 54 :
Training Accuracy = 1.000
Validation Accuracy = 0.990
EPOCH 55 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 56 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 57 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 58 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 59 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 60 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 61 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 62 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 63 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 64 :
Training Accuracy = 1.000

Validation Accuracy = 0.992
EPOCH 65 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 66 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 67 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 68 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 69 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 70 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 71 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 72 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 73 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 74 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 75 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 76 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 77 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 78 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 79 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 80 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 81 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 82 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 83 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 84 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 85 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 86 :

```
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 87 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 88 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 89 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 90 :
Training Accuracy = 0.998
Validation Accuracy = 0.990
EPOCH 91 :
Training Accuracy = 1.000
Validation Accuracy = 0.990
EPOCH 92 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 93 :
Training Accuracy = 0.999
Validation Accuracy = 0.988
EPOCH 94 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 95 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 96 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 97 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 98 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 99 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 100 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
Optimization Finished!
Test Accuracy = 0.991
Training Finished!
Model saved
training time : 3208.06095695 seconds
```

AdamOptimiser, learning rate = 0.001 , Batch size : 50, Training epochs : 100

In [39]:

```
# AdamOptimiser, learning rate = 0.001 , Batch size : 50, Training epochs : 100

# Parameters
import numpy as np
learning_rate = 0.001
training_epochs = 100
```

```
batch_size = 50
logs_path = "TP2/MNIST_99_Challenge_Figures"
x = tf.placeholder(tf.float32, [None, 28, 28, 1], name='InputData')
# 0-9 digits recognition,  10 classes
y = tf.placeholder(tf.float32, [None, 10], name='LabelData')

x_adapted = tf.map_fn(lambda x1: tf.image.pad_to_bounding_box(x1, np.random.randint(4), np.random.ran
dint(4), 32, 32), x)
model = LeNet5_Model(x_adapted)

cross_entropy = tf.nn.softmax_cross_entropy_with_logits(logits=model, labels=y)
loss_operation = tf.reduce_mean(cross_entropy)
optimizer = tf.train.AdamOptimizer(learning_rate = 0.001)
training_operation = optimizer.minimize(loss_operation)
```

In [42]:

```
import time
timeInit = time.time()
train(X_train,y_train,X_validation,y_validation,X_test,y_test)
print('training time :',time.time()- timeInit,'seconds')
```

```
Start Training!
EPOCH 1 :
Training Accuracy = 0.968
Validation Accuracy = 0.969
EPOCH 2 :
Training Accuracy = 0.981
Validation Accuracy = 0.983
EPOCH 3 :
Training Accuracy = 0.985
Validation Accuracy = 0.984
EPOCH 4 :
Training Accuracy = 0.987
Validation Accuracy = 0.985
EPOCH 5 :
Training Accuracy = 0.989
Validation Accuracy = 0.988
EPOCH 6 :
Training Accuracy = 0.991
Validation Accuracy = 0.987
EPOCH 7 :
Training Accuracy = 0.991
Validation Accuracy = 0.986
EPOCH 8 :
Training Accuracy = 0.994
Validation Accuracy = 0.988
EPOCH 9 :
Training Accuracy = 0.995
Validation Accuracy = 0.985
EPOCH 10 :
Training Accuracy = 0.996
Validation Accuracy = 0.989
EPOCH 11 :
Training Accuracy = 0.997
Validation Accuracy = 0.990
EPOCH 12 :
Training Accuracy = 0.994
Validation Accuracy = 0.988
EPOCH 13 :
Training Accuracy = 0.994
Validation Accuracy = 0.985
EPOCH 14 :
Training Accuracy = 0.995
Validation Accuracy = 0.987
```

EPOCH 15 :
Training Accuracy = 0.998
Validation Accuracy = 0.991
EPOCH 16 :
Training Accuracy = 0.997
Validation Accuracy = 0.988
EPOCH 17 :
Training Accuracy = 0.998
Validation Accuracy = 0.990
EPOCH 18 :
Training Accuracy = 0.998
Validation Accuracy = 0.990
EPOCH 19 :
Training Accuracy = 0.994
Validation Accuracy = 0.986
EPOCH 20 :
Training Accuracy = 0.998
Validation Accuracy = 0.990
EPOCH 21 :
Training Accuracy = 0.997
Validation Accuracy = 0.987
EPOCH 22 :
Training Accuracy = 0.998
Validation Accuracy = 0.989
EPOCH 23 :
Training Accuracy = 0.999
Validation Accuracy = 0.989
EPOCH 24 :
Training Accuracy = 0.999
Validation Accuracy = 0.990
EPOCH 25 :
Training Accuracy = 0.999
Validation Accuracy = 0.990
EPOCH 26 :
Training Accuracy = 0.999
Validation Accuracy = 0.990
EPOCH 27 :
Training Accuracy = 0.998
Validation Accuracy = 0.989
EPOCH 28 :
Training Accuracy = 0.999
Validation Accuracy = 0.990
EPOCH 29 :
Training Accuracy = 0.999
Validation Accuracy = 0.989
EPOCH 30 :
Training Accuracy = 0.999
Validation Accuracy = 0.990
EPOCH 31 :
Training Accuracy = 0.997
Validation Accuracy = 0.987
EPOCH 32 :
Training Accuracy = 0.998
Validation Accuracy = 0.989
EPOCH 33 :
Training Accuracy = 0.999
Validation Accuracy = 0.990
EPOCH 34 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 35 :
Training Accuracy = 0.999
Validation Accuracy = 0.990
EPOCH 36 :

Training Accuracy = 0.999
Validation Accuracy = 0.992
EPOCH 37 :
Training Accuracy = 0.998
Validation Accuracy = 0.987
EPOCH 38 :
Training Accuracy = 0.998
Validation Accuracy = 0.987
EPOCH 39 :
Training Accuracy = 0.999
Validation Accuracy = 0.990
EPOCH 40 :
Training Accuracy = 0.998
Validation Accuracy = 0.991
EPOCH 41 :
Training Accuracy = 0.999
Validation Accuracy = 0.991
EPOCH 42 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 43 :
Training Accuracy = 0.999
Validation Accuracy = 0.991
EPOCH 44 :
Training Accuracy = 0.998
Validation Accuracy = 0.989
EPOCH 45 :
Training Accuracy = 0.999
Validation Accuracy = 0.991
EPOCH 46 :
Training Accuracy = 1.000
Validation Accuracy = 0.989
EPOCH 47 :
Training Accuracy = 0.998
Validation Accuracy = 0.989
EPOCH 48 :
Training Accuracy = 0.999
Validation Accuracy = 0.990
EPOCH 49 :
Training Accuracy = 0.999
Validation Accuracy = 0.988
EPOCH 50 :
Training Accuracy = 1.000
Validation Accuracy = 0.990
EPOCH 51 :
Training Accuracy = 0.998
Validation Accuracy = 0.991
EPOCH 52 :
Training Accuracy = 0.999
Validation Accuracy = 0.990
EPOCH 53 :
Training Accuracy = 0.999
Validation Accuracy = 0.989
EPOCH 54 :
Training Accuracy = 0.999
Validation Accuracy = 0.990
EPOCH 55 :
Training Accuracy = 0.999
Validation Accuracy = 0.989
EPOCH 56 :
Training Accuracy = 0.999
Validation Accuracy = 0.992
EPOCH 57 :
Training Accuracy = 1.000
Validation Accuracy = 0.990

EPOCH 58 :
Training Accuracy = 1.000
Validation Accuracy = 0.993
EPOCH 59 :
Training Accuracy = 0.999
Validation Accuracy = 0.989
EPOCH 60 :
Training Accuracy = 0.997
Validation Accuracy = 0.987
EPOCH 61 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 62 :
Training Accuracy = 0.998
Validation Accuracy = 0.988
EPOCH 63 :
Training Accuracy = 1.000
Validation Accuracy = 0.989
EPOCH 64 :
Training Accuracy = 1.000
Validation Accuracy = 0.989
EPOCH 65 :
Training Accuracy = 0.999
Validation Accuracy = 0.990
EPOCH 66 :
Training Accuracy = 0.999
Validation Accuracy = 0.990
EPOCH 67 :
Training Accuracy = 0.999
Validation Accuracy = 0.988
EPOCH 68 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 69 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 70 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 71 :
Training Accuracy = 1.000
Validation Accuracy = 0.989
EPOCH 72 :
Training Accuracy = 0.998
Validation Accuracy = 0.988
EPOCH 73 :
Training Accuracy = 0.999
Validation Accuracy = 0.989
EPOCH 74 :
Training Accuracy = 1.000
Validation Accuracy = 0.989
EPOCH 75 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 76 :
Training Accuracy = 1.000
Validation Accuracy = 0.990
EPOCH 77 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 78 :
Training Accuracy = 0.998
Validation Accuracy = 0.988
EPOCH 79 :

Training Accuracy = 0.999
Validation Accuracy = 0.990
EPOCH 80 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 81 :
Training Accuracy = 0.999
Validation Accuracy = 0.988
EPOCH 82 :
Training Accuracy = 1.000
Validation Accuracy = 0.990
EPOCH 83 :
Training Accuracy = 1.000
Validation Accuracy = 0.990
EPOCH 84 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 85 :
Training Accuracy = 1.000
Validation Accuracy = 0.991
EPOCH 86 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 87 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 88 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 89 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 90 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 91 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 92 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 93 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 94 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 95 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 96 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 97 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 98 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 99 :
Training Accuracy = 1.000
Validation Accuracy = 0.992
EPOCH 100 :
Training Accuracy = 1.000
Validation Accuracy = 0.992

```
Optimization Finished!
Test Accuracy = 0.992
Training Finished!
Model saved
training time : 3187.14975691 seconds
```

SGD, learning rate = 0.0001 , Batch size : 128, Training epochs : 100

In [43]:

```
# SGD, learning rate = 0.0001 , Batch size : 128, Training epochs : 100

# Parameters
import numpy as np
learning_rate = 0.0001
training_epochs = 100
batch_size = 128
logs_path = "TP2/MNIST_99_Challenge_Figures"
x = tf.placeholder(tf.float32, [None, 28, 28, 1], name='InputData')
# 0-9 digits recognition,  10 classes
y = tf.placeholder(tf.float32, [None, 10], name='LabelData')

x_adapted = tf.map_fn(lambda x1: tf.image.pad_to_bounding_box(x1, np.random.randint(4), np.random.ran
dint(4), 32, 32), x)
model = LeNet5_Model(x_adapted)

cross_entropy = tf.nn.softmax_cross_entropy_with_logits(logits=model, labels=y)
loss_operation = tf.reduce_mean(cross_entropy)
optimizer = tf.train.GradientDescentOptimizer(learning_rate = learning_rate)
training_operation = optimizer.minimize(loss_operation)
```

In [46]:

```
import time
timeInit = time.time()
train(X_train,y_train,X_validation,y_validation,X_test,y_test)
print('training time :',time.time()- timeInit,'seconds')
```

```
Start Training!
EPOCH 1 :
Training Accuracy = 0.133
Validation Accuracy = 0.126
EPOCH 2 :
Training Accuracy = 0.137
Validation Accuracy = 0.130
EPOCH 3 :
Training Accuracy = 0.140
Validation Accuracy = 0.133
EPOCH 4 :
Training Accuracy = 0.142
Validation Accuracy = 0.136
EPOCH 5 :
Training Accuracy = 0.144
Validation Accuracy = 0.137
EPOCH 6 :
Training Accuracy = 0.146
Validation Accuracy = 0.139
EPOCH 7 :
Training Accuracy = 0.147
Validation Accuracy = 0.140
EPOCH 8 :
```

EPOCH 8 :
Training Accuracy = 0.149
Validation Accuracy = 0.142
EPOCH 9 :
Training Accuracy = 0.150
Validation Accuracy = 0.145
EPOCH 10 :
Training Accuracy = 0.153
Validation Accuracy = 0.147
EPOCH 11 :
Training Accuracy = 0.156
Validation Accuracy = 0.150
EPOCH 12 :
Training Accuracy = 0.159
Validation Accuracy = 0.155
EPOCH 13 :
Training Accuracy = 0.163
Validation Accuracy = 0.157
EPOCH 14 :
Training Accuracy = 0.168
Validation Accuracy = 0.161
EPOCH 15 :
Training Accuracy = 0.173
Validation Accuracy = 0.166
EPOCH 16 :
Training Accuracy = 0.178
Validation Accuracy = 0.169
EPOCH 17 :
Training Accuracy = 0.184
Validation Accuracy = 0.176
EPOCH 18 :
Training Accuracy = 0.189
Validation Accuracy = 0.181
EPOCH 19 :
Training Accuracy = 0.195
Validation Accuracy = 0.186
EPOCH 20 :
Training Accuracy = 0.204
Validation Accuracy = 0.193
EPOCH 21 :
Training Accuracy = 0.214
Validation Accuracy = 0.203
EPOCH 22 :
Training Accuracy = 0.228
Validation Accuracy = 0.216
EPOCH 23 :
Training Accuracy = 0.243
Validation Accuracy = 0.228
EPOCH 24 :
Training Accuracy = 0.258
Validation Accuracy = 0.245
EPOCH 25 :
Training Accuracy = 0.273
Validation Accuracy = 0.260
EPOCH 26 :
Training Accuracy = 0.287
Validation Accuracy = 0.277
EPOCH 27 :
Training Accuracy = 0.302
Validation Accuracy = 0.291
EPOCH 28 :
Training Accuracy = 0.316
Validation Accuracy = 0.305
EPOCH 29 :
Training Accuracy = 0.331

Validation Accuracy = 0.320
EPOCH 30 :
Training Accuracy = 0.346
Validation Accuracy = 0.334
EPOCH 31 :
Training Accuracy = 0.363
Validation Accuracy = 0.352
EPOCH 32 :
Training Accuracy = 0.380
Validation Accuracy = 0.365
EPOCH 33 :
Training Accuracy = 0.399
Validation Accuracy = 0.386
EPOCH 34 :
Training Accuracy = 0.417
Validation Accuracy = 0.406
EPOCH 35 :
Training Accuracy = 0.437
Validation Accuracy = 0.428
EPOCH 36 :
Training Accuracy = 0.457
Validation Accuracy = 0.451
EPOCH 37 :
Training Accuracy = 0.479
Validation Accuracy = 0.475
EPOCH 38 :
Training Accuracy = 0.501
Validation Accuracy = 0.497
EPOCH 39 :
Training Accuracy = 0.523
Validation Accuracy = 0.522
EPOCH 40 :
Training Accuracy = 0.545
Validation Accuracy = 0.544
EPOCH 41 :
Training Accuracy = 0.565
Validation Accuracy = 0.563
EPOCH 42 :
Training Accuracy = 0.586
Validation Accuracy = 0.582
EPOCH 43 :
Training Accuracy = 0.604
Validation Accuracy = 0.602
EPOCH 44 :
Training Accuracy = 0.622
Validation Accuracy = 0.622
EPOCH 45 :
Training Accuracy = 0.639
Validation Accuracy = 0.635
EPOCH 46 :
Training Accuracy = 0.653
Validation Accuracy = 0.652
EPOCH 47 :
Training Accuracy = 0.666
Validation Accuracy = 0.669
EPOCH 48 :
Training Accuracy = 0.678
Validation Accuracy = 0.682
EPOCH 49 :
Training Accuracy = 0.690
Validation Accuracy = 0.696
EPOCH 50 :
Training Accuracy = 0.701
Validation Accuracy = 0.709

EPOCH 51 :
Training Accuracy = 0.711
Validation Accuracy = 0.719
EPOCH 52 :
Training Accuracy = 0.721
Validation Accuracy = 0.729
EPOCH 53 :
Training Accuracy = 0.730
Validation Accuracy = 0.739
EPOCH 54 :
Training Accuracy = 0.742
Validation Accuracy = 0.748
EPOCH 55 :
Training Accuracy = 0.750
Validation Accuracy = 0.757
EPOCH 56 :
Training Accuracy = 0.758
Validation Accuracy = 0.766
EPOCH 57 :
Training Accuracy = 0.767
Validation Accuracy = 0.773
EPOCH 58 :
Training Accuracy = 0.775
Validation Accuracy = 0.779
EPOCH 59 :
Training Accuracy = 0.784
Validation Accuracy = 0.785
EPOCH 60 :
Training Accuracy = 0.789
Validation Accuracy = 0.791
EPOCH 61 :
Training Accuracy = 0.797
Validation Accuracy = 0.798
EPOCH 62 :
Training Accuracy = 0.804
Validation Accuracy = 0.803
EPOCH 63 :
Training Accuracy = 0.809
Validation Accuracy = 0.810
EPOCH 64 :
Training Accuracy = 0.815
Validation Accuracy = 0.817
EPOCH 65 :
Training Accuracy = 0.820
Validation Accuracy = 0.823
EPOCH 66 :
Training Accuracy = 0.824
Validation Accuracy = 0.826
EPOCH 67 :
Training Accuracy = 0.827
Validation Accuracy = 0.830
EPOCH 68 :
Training Accuracy = 0.832
Validation Accuracy = 0.834
EPOCH 69 :
Training Accuracy = 0.836
Validation Accuracy = 0.838
EPOCH 70 :
Training Accuracy = 0.839
Validation Accuracy = 0.842
EPOCH 71 :
Training Accuracy = 0.841
Validation Accuracy = 0.844
EPOCH 72 :
Training Accuracy = 0.844

Validation Accuracy = 0.847
EPOCH 73 :
Training Accuracy = 0.847
Validation Accuracy = 0.849
EPOCH 74 :
Training Accuracy = 0.850
Validation Accuracy = 0.854
EPOCH 75 :
Training Accuracy = 0.852
Validation Accuracy = 0.856
EPOCH 76 :
Training Accuracy = 0.855
Validation Accuracy = 0.859
EPOCH 77 :
Training Accuracy = 0.858
Validation Accuracy = 0.862
EPOCH 78 :
Training Accuracy = 0.860
Validation Accuracy = 0.864
EPOCH 79 :
Training Accuracy = 0.862
Validation Accuracy = 0.866
EPOCH 80 :
Training Accuracy = 0.863
Validation Accuracy = 0.866
EPOCH 81 :
Training Accuracy = 0.865
Validation Accuracy = 0.868
EPOCH 82 :
Training Accuracy = 0.867
Validation Accuracy = 0.870
EPOCH 83 :
Training Accuracy = 0.868
Validation Accuracy = 0.871
EPOCH 84 :
Training Accuracy = 0.869
Validation Accuracy = 0.873
EPOCH 85 :
Training Accuracy = 0.871
Validation Accuracy = 0.874
EPOCH 86 :
Training Accuracy = 0.872
Validation Accuracy = 0.875
EPOCH 87 :
Training Accuracy = 0.873
Validation Accuracy = 0.876
EPOCH 88 :
Training Accuracy = 0.875
Validation Accuracy = 0.878
EPOCH 89 :
Training Accuracy = 0.876
Validation Accuracy = 0.879
EPOCH 90 :
Training Accuracy = 0.877
Validation Accuracy = 0.880
EPOCH 91 :
Training Accuracy = 0.878
Validation Accuracy = 0.883
EPOCH 92 :
Training Accuracy = 0.879
Validation Accuracy = 0.884
EPOCH 93 :
Training Accuracy = 0.879
Validation Accuracy = 0.886

```
EPOCH 94 :
Training Accuracy = 0.880
Validation Accuracy = 0.885
EPOCH 95 :
Training Accuracy = 0.881
Validation Accuracy = 0.887
EPOCH 96 :
Training Accuracy = 0.882
Validation Accuracy = 0.888
EPOCH 97 :
Training Accuracy = 0.882
Validation Accuracy = 0.889
EPOCH 98 :
Training Accuracy = 0.884
Validation Accuracy = 0.890
EPOCH 99 :
Training Accuracy = 0.884
Validation Accuracy = 0.890
EPOCH 100 :
Training Accuracy = 0.885
Validation Accuracy = 0.889
Optimization Finished!
Test Accuracy = 0.890
Training Finished!
Model saved
training time : 3129.83763409 seconds
```

SGD, learning rate = 0.0001 , Batch size : 50, Training epochs : 100

In [47]:

```
# SGD, learning rate = 0.0001 , Batch size : 50, Training epochs : 100

# Parameters
import numpy as np
learning_rate = 0.0001
training_epochs = 100
batch_size = 50
logs_path = "TP2/MNIST_99_Challenge_Figures"
x = tf.placeholder(tf.float32, [None, 28, 28, 1], name='InputData')
# 0-9 digits recognition,  10 classes
y = tf.placeholder(tf.float32, [None, 10], name='LabelData')

x_adapted = tf.map_fn(lambda x1: tf.image.pad_to_bounding_box(x1, np.random.randint(4), np.random.ran
dint(4), 32, 32), x)
model = LeNet5_Model(x_adapted)

cross_entropy = tf.nn.softmax_cross_entropy_with_logits(logits=model, labels=y)
loss_operation = tf.reduce_mean(cross_entropy)
optimizer = tf.train.GradientDescentOptimizer(learning_rate = learning_rate)
training_operation = optimizer.minimize(loss_operation)
```

In [50]:

```
import time
timeInit = time.time()
train(X_train,y_train,X_validation,y_validation,X_test,y_test)
print('training time :',time.time()- timeInit,'seconds')
```

```
Start Training!
EPOCH 1 :
```

Training Accuracy = 0.123
Validation Accuracy = 0.128
EPOCH 2 :
Training Accuracy = 0.134
Validation Accuracy = 0.137
EPOCH 3 :
Training Accuracy = 0.144
Validation Accuracy = 0.153
EPOCH 4 :
Training Accuracy = 0.153
Validation Accuracy = 0.166
EPOCH 5 :
Training Accuracy = 0.160
Validation Accuracy = 0.168
EPOCH 6 :
Training Accuracy = 0.167
Validation Accuracy = 0.175
EPOCH 7 :
Training Accuracy = 0.174
Validation Accuracy = 0.180
EPOCH 8 :
Training Accuracy = 0.182
Validation Accuracy = 0.188
EPOCH 9 :
Training Accuracy = 0.191
Validation Accuracy = 0.198
EPOCH 10 :
Training Accuracy = 0.202
Validation Accuracy = 0.206
EPOCH 11 :
Training Accuracy = 0.213
Validation Accuracy = 0.217
EPOCH 12 :
Training Accuracy = 0.225
Validation Accuracy = 0.227
EPOCH 13 :
Training Accuracy = 0.237
Validation Accuracy = 0.242
EPOCH 14 :
Training Accuracy = 0.251
Validation Accuracy = 0.256
EPOCH 15 :
Training Accuracy = 0.267
Validation Accuracy = 0.272
EPOCH 16 :
Training Accuracy = 0.284
Validation Accuracy = 0.288
EPOCH 17 :
Training Accuracy = 0.301
Validation Accuracy = 0.302
EPOCH 18 :
Training Accuracy = 0.317
Validation Accuracy = 0.318
EPOCH 19 :
Training Accuracy = 0.331
Validation Accuracy = 0.333
EPOCH 20 :
Training Accuracy = 0.344
Validation Accuracy = 0.348
EPOCH 21 :
Training Accuracy = 0.358
Validation Accuracy = 0.358
EPOCH 22 :
Training Accuracy = 0.369
Validation Accuracy = 0.368

EPOCH 23 :
Training Accuracy = 0.380
Validation Accuracy = 0.379
EPOCH 24 :
Training Accuracy = 0.390
Validation Accuracy = 0.391
EPOCH 25 :
Training Accuracy = 0.400
Validation Accuracy = 0.400
EPOCH 26 :
Training Accuracy = 0.408
Validation Accuracy = 0.408
EPOCH 27 :
Training Accuracy = 0.418
Validation Accuracy = 0.418
EPOCH 28 :
Training Accuracy = 0.428
Validation Accuracy = 0.429
EPOCH 29 :
Training Accuracy = 0.440
Validation Accuracy = 0.444
EPOCH 30 :
Training Accuracy = 0.454
Validation Accuracy = 0.457
EPOCH 31 :
Training Accuracy = 0.468
Validation Accuracy = 0.474
EPOCH 32 :
Training Accuracy = 0.482
Validation Accuracy = 0.491
EPOCH 33 :
Training Accuracy = 0.496
Validation Accuracy = 0.506
EPOCH 34 :
Training Accuracy = 0.510
Validation Accuracy = 0.520
EPOCH 35 :
Training Accuracy = 0.523
Validation Accuracy = 0.534
EPOCH 36 :
Training Accuracy = 0.536
Validation Accuracy = 0.547
EPOCH 37 :
Training Accuracy = 0.547
Validation Accuracy = 0.558
EPOCH 38 :
Training Accuracy = 0.559
Validation Accuracy = 0.569
EPOCH 39 :
Training Accuracy = 0.570
Validation Accuracy = 0.582
EPOCH 40 :
Training Accuracy = 0.582
Validation Accuracy = 0.594
EPOCH 41 :
Training Accuracy = 0.593
Validation Accuracy = 0.605
EPOCH 42 :
Training Accuracy = 0.603
Validation Accuracy = 0.614
EPOCH 43 :
Training Accuracy = 0.614
Validation Accuracy = 0.623
EPOCH 44 :

Training Accuracy = 0.625
Validation Accuracy = 0.634
EPOCH 45 :
Training Accuracy = 0.635
Validation Accuracy = 0.643
EPOCH 46 :
Training Accuracy = 0.645
Validation Accuracy = 0.653
EPOCH 47 :
Training Accuracy = 0.656
Validation Accuracy = 0.663
EPOCH 48 :
Training Accuracy = 0.665
Validation Accuracy = 0.672
EPOCH 49 :
Training Accuracy = 0.675
Validation Accuracy = 0.679
EPOCH 50 :
Training Accuracy = 0.686
Validation Accuracy = 0.693
EPOCH 51 :
Training Accuracy = 0.698
Validation Accuracy = 0.703
EPOCH 52 :
Training Accuracy = 0.710
Validation Accuracy = 0.716
EPOCH 53 :
Training Accuracy = 0.720
Validation Accuracy = 0.727
EPOCH 54 :
Training Accuracy = 0.730
Validation Accuracy = 0.735
EPOCH 55 :
Training Accuracy = 0.739
Validation Accuracy = 0.745
EPOCH 56 :
Training Accuracy = 0.747
Validation Accuracy = 0.753
EPOCH 57 :
Training Accuracy = 0.756
Validation Accuracy = 0.763
EPOCH 58 :
Training Accuracy = 0.764
Validation Accuracy = 0.770
EPOCH 59 :
Training Accuracy = 0.771
Validation Accuracy = 0.778
EPOCH 60 :
Training Accuracy = 0.777
Validation Accuracy = 0.785
EPOCH 61 :
Training Accuracy = 0.783
Validation Accuracy = 0.789
EPOCH 62 :
Training Accuracy = 0.789
Validation Accuracy = 0.794
EPOCH 63 :
Training Accuracy = 0.795
Validation Accuracy = 0.801
EPOCH 64 :
Training Accuracy = 0.800
Validation Accuracy = 0.806
EPOCH 65 :
Training Accuracy = 0.806
Validation Accuracy = 0.810

Validation Accuracy = 0.810
EPOCH 66 :
Training Accuracy = 0.810
Validation Accuracy = 0.815
EPOCH 67 :
Training Accuracy = 0.814
Validation Accuracy = 0.817
EPOCH 68 :
Training Accuracy = 0.818
Validation Accuracy = 0.820
EPOCH 69 :
Training Accuracy = 0.822
Validation Accuracy = 0.825
EPOCH 70 :
Training Accuracy = 0.826
Validation Accuracy = 0.829
EPOCH 71 :
Training Accuracy = 0.828
Validation Accuracy = 0.831
EPOCH 72 :
Training Accuracy = 0.832
Validation Accuracy = 0.834
EPOCH 73 :
Training Accuracy = 0.836
Validation Accuracy = 0.840
EPOCH 74 :
Training Accuracy = 0.839
Validation Accuracy = 0.841
EPOCH 75 :
Training Accuracy = 0.841
Validation Accuracy = 0.844
EPOCH 76 :
Training Accuracy = 0.845
Validation Accuracy = 0.846
EPOCH 77 :
Training Accuracy = 0.847
Validation Accuracy = 0.850
EPOCH 78 :
Training Accuracy = 0.850
Validation Accuracy = 0.851
EPOCH 79 :
Training Accuracy = 0.852
Validation Accuracy = 0.856
EPOCH 80 :
Training Accuracy = 0.855
Validation Accuracy = 0.859
EPOCH 81 :
Training Accuracy = 0.857
Validation Accuracy = 0.860
EPOCH 82 :
Training Accuracy = 0.859
Validation Accuracy = 0.862
EPOCH 83 :
Training Accuracy = 0.861
Validation Accuracy = 0.863
EPOCH 84 :
Training Accuracy = 0.863
Validation Accuracy = 0.864
EPOCH 85 :
Training Accuracy = 0.865
Validation Accuracy = 0.867
EPOCH 86 :
Training Accuracy = 0.867
Validation Accuracy = 0.868
EPOCH 87 :

```
Training Accuracy = 0.869
Validation Accuracy = 0.870
EPOCH 88 :
Training Accuracy = 0.870
Validation Accuracy = 0.871
EPOCH 89 :
Training Accuracy = 0.872
Validation Accuracy = 0.873
EPOCH 90 :
Training Accuracy = 0.873
Validation Accuracy = 0.873
EPOCH 91 :
Training Accuracy = 0.875
Validation Accuracy = 0.875
EPOCH 92 :
Training Accuracy = 0.876
Validation Accuracy = 0.876
EPOCH 93 :
Training Accuracy = 0.877
Validation Accuracy = 0.878
EPOCH 94 :
Training Accuracy = 0.879
Validation Accuracy = 0.878
EPOCH 95 :
Training Accuracy = 0.880
Validation Accuracy = 0.881
EPOCH 96 :
Training Accuracy = 0.881
Validation Accuracy = 0.882
EPOCH 97 :
Training Accuracy = 0.882
Validation Accuracy = 0.883
EPOCH 98 :
Training Accuracy = 0.883
Validation Accuracy = 0.884
EPOCH 99 :
Training Accuracy = 0.884
Validation Accuracy = 0.885
EPOCH 100 :
Training Accuracy = 0.884
Validation Accuracy = 0.887
Optimization Finished!
Test Accuracy = 0.891
Training Finished!
Model saved
training time : 3187.04807782 seconds
```

SGD, learning rate = 0.001 , Batch size : 128, Training epochs : 100

In [51]:

```
# SGD, learning rate = 0.001 , Batch size : 128, Training epochs : 100

# Parameters
import numpy as np
learning_rate = 0.001
training_epochs = 100
batch_size = 128
logs_path = "TP2/MNIST_99_Challenge_Figures"
x = tf.placeholder(tf.float32, [None, 28, 28, 1], name='InputData')
# 0-9 digits recognition,  10 classes
```

```
y = tf.placeholder(tf.float32, [None, 10], name='LabelData')

x_adapted = tf.map_fn(lambda x1: tf.image.pad_to_bounding_box(x1, np.random.randint(4), np.random.ran
dint(4), 32, 32), x)
model = LeNet5_Model(x_adapted)

cross_entropy = tf.nn.softmax_cross_entropy_with_logits(logits=model, labels=y)
loss_operation = tf.reduce_mean(cross_entropy)
optimizer = tf.train.GradientDescentOptimizer(learning_rate = learning_rate)
training_operation = optimizer.minimize(loss_operation)
```

In [54]:

```
import time
timeInit = time.time()
train(X_train,y_train,X_validation,y_validation,X_test,y_test)
print('training time :',time.time()- timeInit,'seconds')
```

```
Start Training!
EPOCH 1 :
Training Accuracy = 0.256
Validation Accuracy = 0.248
EPOCH 2 :
Training Accuracy = 0.415
Validation Accuracy = 0.410
EPOCH 3 :
Training Accuracy = 0.539
Validation Accuracy = 0.538
EPOCH 4 :
Training Accuracy = 0.653
Validation Accuracy = 0.659
EPOCH 5 :
Training Accuracy = 0.754
Validation Accuracy = 0.756
EPOCH 6 :
Training Accuracy = 0.812
Validation Accuracy = 0.818
EPOCH 7 :
Training Accuracy = 0.843
Validation Accuracy = 0.852
EPOCH 8 :
Training Accuracy = 0.864
Validation Accuracy = 0.870
EPOCH 9 :
Training Accuracy = 0.877
Validation Accuracy = 0.886
EPOCH 10 :
Training Accuracy = 0.886
Validation Accuracy = 0.893
EPOCH 11 :
Training Accuracy = 0.892
Validation Accuracy = 0.898
EPOCH 12 :
Training Accuracy = 0.896
Validation Accuracy = 0.900
EPOCH 13 :
Training Accuracy = 0.902
Validation Accuracy = 0.906
EPOCH 14 :
Training Accuracy = 0.905
Validation Accuracy = 0.909
EPOCH 15 :
Training Accuracy = 0.910
Validation Accuracy = 0.916
```

EPOCH 16 :
Training Accuracy = 0.912
Validation Accuracy = 0.917
EPOCH 17 :
Training Accuracy = 0.916
Validation Accuracy = 0.922
EPOCH 18 :
Training Accuracy = 0.918
Validation Accuracy = 0.924
EPOCH 19 :
Training Accuracy = 0.921
Validation Accuracy = 0.926
EPOCH 20 :
Training Accuracy = 0.922
Validation Accuracy = 0.927
EPOCH 21 :
Training Accuracy = 0.924
Validation Accuracy = 0.930
EPOCH 22 :
Training Accuracy = 0.927
Validation Accuracy = 0.932
EPOCH 23 :
Training Accuracy = 0.929
Validation Accuracy = 0.936
EPOCH 24 :
Training Accuracy = 0.931
Validation Accuracy = 0.937
EPOCH 25 :
Training Accuracy = 0.933
Validation Accuracy = 0.940
EPOCH 26 :
Training Accuracy = 0.935
Validation Accuracy = 0.941
EPOCH 27 :
Training Accuracy = 0.936
Validation Accuracy = 0.941
EPOCH 28 :
Training Accuracy = 0.937
Validation Accuracy = 0.943
EPOCH 29 :
Training Accuracy = 0.939
Validation Accuracy = 0.946
EPOCH 30 :
Training Accuracy = 0.939
Validation Accuracy = 0.949
EPOCH 31 :
Training Accuracy = 0.941
Validation Accuracy = 0.948
EPOCH 32 :
Training Accuracy = 0.942
Validation Accuracy = 0.951
EPOCH 33 :
Training Accuracy = 0.944
Validation Accuracy = 0.951
EPOCH 34 :
Training Accuracy = 0.945
Validation Accuracy = 0.951
EPOCH 35 :
Training Accuracy = 0.946
Validation Accuracy = 0.953
EPOCH 36 :
Training Accuracy = 0.946
Validation Accuracy = 0.954
EPOCH 37 :
Training Accuracy = 0.948

Validation Accuracy = 0.955
EPOCH 38 :
Training Accuracy = 0.947
Validation Accuracy = 0.952
EPOCH 39 :
Training Accuracy = 0.950
Validation Accuracy = 0.958
EPOCH 40 :
Training Accuracy = 0.951
Validation Accuracy = 0.958
EPOCH 41 :
Training Accuracy = 0.951
Validation Accuracy = 0.960
EPOCH 42 :
Training Accuracy = 0.952
Validation Accuracy = 0.961
EPOCH 43 :
Training Accuracy = 0.953
Validation Accuracy = 0.959
EPOCH 44 :
Training Accuracy = 0.953
Validation Accuracy = 0.960
EPOCH 45 :
Training Accuracy = 0.954
Validation Accuracy = 0.962
EPOCH 46 :
Training Accuracy = 0.955
Validation Accuracy = 0.960
EPOCH 47 :
Training Accuracy = 0.955
Validation Accuracy = 0.963
EPOCH 48 :
Training Accuracy = 0.957
Validation Accuracy = 0.962
EPOCH 49 :
Training Accuracy = 0.957
Validation Accuracy = 0.963
EPOCH 50 :
Training Accuracy = 0.957
Validation Accuracy = 0.962
EPOCH 51 :
Training Accuracy = 0.959
Validation Accuracy = 0.963
EPOCH 52 :
Training Accuracy = 0.959
Validation Accuracy = 0.963
EPOCH 53 :
Training Accuracy = 0.959
Validation Accuracy = 0.966
EPOCH 54 :
Training Accuracy = 0.960
Validation Accuracy = 0.966
EPOCH 55 :
Training Accuracy = 0.961
Validation Accuracy = 0.965
EPOCH 56 :
Training Accuracy = 0.961
Validation Accuracy = 0.968
EPOCH 57 :
Training Accuracy = 0.961
Validation Accuracy = 0.964
EPOCH 58 :
Training Accuracy = 0.963
Validation Accuracy = 0.969

EPOCH 59 :
Training Accuracy = 0.962
Validation Accuracy = 0.965
EPOCH 60 :
Training Accuracy = 0.963
Validation Accuracy = 0.968
EPOCH 61 :
Training Accuracy = 0.964
Validation Accuracy = 0.970
EPOCH 62 :
Training Accuracy = 0.964
Validation Accuracy = 0.971
EPOCH 63 :
Training Accuracy = 0.965
Validation Accuracy = 0.970
EPOCH 64 :
Training Accuracy = 0.964
Validation Accuracy = 0.968
EPOCH 65 :
Training Accuracy = 0.965
Validation Accuracy = 0.970
EPOCH 66 :
Training Accuracy = 0.965
Validation Accuracy = 0.971
EPOCH 67 :
Training Accuracy = 0.966
Validation Accuracy = 0.970
EPOCH 68 :
Training Accuracy = 0.966
Validation Accuracy = 0.970
EPOCH 69 :
Training Accuracy = 0.967
Validation Accuracy = 0.973
EPOCH 70 :
Training Accuracy = 0.967
Validation Accuracy = 0.970
EPOCH 71 :
Training Accuracy = 0.967
Validation Accuracy = 0.973
EPOCH 72 :
Training Accuracy = 0.967
Validation Accuracy = 0.972
EPOCH 73 :
Training Accuracy = 0.968
Validation Accuracy = 0.972
EPOCH 74 :
Training Accuracy = 0.969
Validation Accuracy = 0.973
EPOCH 75 :
Training Accuracy = 0.969
Validation Accuracy = 0.973
EPOCH 76 :
Training Accuracy = 0.968
Validation Accuracy = 0.972
EPOCH 77 :
Training Accuracy = 0.970
Validation Accuracy = 0.975
EPOCH 78 :
Training Accuracy = 0.969
Validation Accuracy = 0.974
EPOCH 79 :
Training Accuracy = 0.970
Validation Accuracy = 0.973
EPOCH 80 :
Training Accuracy = 0.970

Validation Accuracy = 0.975
EPOCH 81 :
Training Accuracy = 0.970
Validation Accuracy = 0.974
EPOCH 82 :
Training Accuracy = 0.970
Validation Accuracy = 0.974
EPOCH 83 :
Training Accuracy = 0.971
Validation Accuracy = 0.976
EPOCH 84 :
Training Accuracy = 0.971
Validation Accuracy = 0.974
EPOCH 85 :
Training Accuracy = 0.971
Validation Accuracy = 0.976
EPOCH 86 :
Training Accuracy = 0.972
Validation Accuracy = 0.976
EPOCH 87 :
Training Accuracy = 0.971
Validation Accuracy = 0.976
EPOCH 88 :
Training Accuracy = 0.972
Validation Accuracy = 0.977
EPOCH 89 :
Training Accuracy = 0.972
Validation Accuracy = 0.977
EPOCH 90 :
Training Accuracy = 0.972
Validation Accuracy = 0.976
EPOCH 91 :
Training Accuracy = 0.971
Validation Accuracy = 0.977
EPOCH 92 :
Training Accuracy = 0.973
Validation Accuracy = 0.977
EPOCH 93 :
Training Accuracy = 0.973
Validation Accuracy = 0.976
EPOCH 94 :
Training Accuracy = 0.973
Validation Accuracy = 0.978
EPOCH 95 :
Training Accuracy = 0.973
Validation Accuracy = 0.978
EPOCH 96 :
Training Accuracy = 0.973
Validation Accuracy = 0.976
EPOCH 97 :
Training Accuracy = 0.974
Validation Accuracy = 0.978
EPOCH 98 :
Training Accuracy = 0.973
Validation Accuracy = 0.977
EPOCH 99 :
Training Accuracy = 0.974
Validation Accuracy = 0.978
EPOCH 100 :
Training Accuracy = 0.974
Validation Accuracy = 0.978
Optimization Finished!
Test Accuracy = 0.977
Training Finished!

```
Model saved
training time : 3127.34981704 seconds
```

SGD, learning rate = 0.001 , Batch size : 50, Training epochs : 100

In [55]:

```
# SGD, learning rate = 0.001 , Batch size : 50, Training epochs : 100

# Parameters
import numpy as np
learning_rate = 0.001
training_epochs = 100
batch_size = 50
logs_path = "TP2/MNIST_99_Challenge_Figures"
x = tf.placeholder(tf.float32, [None, 28, 28, 1], name='InputData')
# 0-9 digits recognition,  10 classes
y = tf.placeholder(tf.float32, [None, 10], name='LabelData')

x_adapted = tf.map_fn(lambda x1: tf.image.pad_to_bounding_box(x1, np.random.randint(4), np.random.ran
dint(4), 32, 32), x)
model = LeNet5_Model(x_adapted)

cross_entropy = tf.nn.softmax_cross_entropy_with_logits(logits=model, labels=y)
loss_operation = tf.reduce_mean(cross_entropy)
optimizer = tf.train.GradientDescentOptimizer(learning_rate = learning_rate)
training_operation = optimizer.minimize(loss_operation)
```

In [58]:

```
import time
timeInit = time.time()
train(X_train,y_train,X_validation,y_validation,X_test,y_test)
print('training time :',time.time()- timeInit,'seconds')
```

```
Start Training!
EPOCH 1 :
Training Accuracy = 0.103
Validation Accuracy = 0.101
EPOCH 2 :
Training Accuracy = 0.279
Validation Accuracy = 0.277
EPOCH 3 :
Training Accuracy = 0.434
Validation Accuracy = 0.418
EPOCH 4 :
Training Accuracy = 0.537
Validation Accuracy = 0.528
EPOCH 5 :
Training Accuracy = 0.658
Validation Accuracy = 0.651
EPOCH 6 :
Training Accuracy = 0.752
Validation Accuracy = 0.747
EPOCH 7 :
Training Accuracy = 0.805
Validation Accuracy = 0.801
EPOCH 8 :
Training Accuracy = 0.840
Validation Accuracy = 0.843
EPOCH 9 :
```

Training Accuracy = 0.861
Validation Accuracy = 0.869
EPOCH 10 :
Training Accuracy = 0.873
Validation Accuracy = 0.879
EPOCH 11 :
Training Accuracy = 0.885
Validation Accuracy = 0.888
EPOCH 12 :
Training Accuracy = 0.891
Validation Accuracy = 0.895
EPOCH 13 :
Training Accuracy = 0.897
Validation Accuracy = 0.901
EPOCH 14 :
Training Accuracy = 0.902
Validation Accuracy = 0.906
EPOCH 15 :
Training Accuracy = 0.906
Validation Accuracy = 0.911
EPOCH 16 :
Training Accuracy = 0.909
Validation Accuracy = 0.915
EPOCH 17 :
Training Accuracy = 0.912
Validation Accuracy = 0.919
EPOCH 18 :
Training Accuracy = 0.916
Validation Accuracy = 0.921
EPOCH 19 :
Training Accuracy = 0.918
Validation Accuracy = 0.923
EPOCH 20 :
Training Accuracy = 0.921
Validation Accuracy = 0.926
EPOCH 21 :
Training Accuracy = 0.924
Validation Accuracy = 0.928
EPOCH 22 :
Training Accuracy = 0.925
Validation Accuracy = 0.929
EPOCH 23 :
Training Accuracy = 0.927
Validation Accuracy = 0.933
EPOCH 24 :
Training Accuracy = 0.929
Validation Accuracy = 0.933
EPOCH 25 :
Training Accuracy = 0.931
Validation Accuracy = 0.934
EPOCH 26 :
Training Accuracy = 0.933
Validation Accuracy = 0.937
EPOCH 27 :
Training Accuracy = 0.934
Validation Accuracy = 0.939
EPOCH 28 :
Training Accuracy = 0.935
Validation Accuracy = 0.940
EPOCH 29 :
Training Accuracy = 0.938
Validation Accuracy = 0.942
EPOCH 30 :
Training Accuracy = 0.939
Validation Accuracy = 0.943

Validation Accuracy = 0.943
EPOCH 31 :
Training Accuracy = 0.940
Validation Accuracy = 0.944
EPOCH 32 :
Training Accuracy = 0.941
Validation Accuracy = 0.945
EPOCH 33 :
Training Accuracy = 0.943
Validation Accuracy = 0.950
EPOCH 34 :
Training Accuracy = 0.943
Validation Accuracy = 0.947
EPOCH 35 :
Training Accuracy = 0.945
Validation Accuracy = 0.948
EPOCH 36 :
Training Accuracy = 0.946
Validation Accuracy = 0.951
EPOCH 37 :
Training Accuracy = 0.947
Validation Accuracy = 0.951
EPOCH 38 :
Training Accuracy = 0.948
Validation Accuracy = 0.951
EPOCH 39 :
Training Accuracy = 0.948
Validation Accuracy = 0.951
EPOCH 40 :
Training Accuracy = 0.950
Validation Accuracy = 0.955
EPOCH 41 :
Training Accuracy = 0.951
Validation Accuracy = 0.954
EPOCH 42 :
Training Accuracy = 0.951
Validation Accuracy = 0.957
EPOCH 43 :
Training Accuracy = 0.952
Validation Accuracy = 0.956
EPOCH 44 :
Training Accuracy = 0.953
Validation Accuracy = 0.955
EPOCH 45 :
Training Accuracy = 0.954
Validation Accuracy = 0.956
EPOCH 46 :
Training Accuracy = 0.955
Validation Accuracy = 0.958
EPOCH 47 :
Training Accuracy = 0.955
Validation Accuracy = 0.958
EPOCH 48 :
Training Accuracy = 0.955
Validation Accuracy = 0.958
EPOCH 49 :
Training Accuracy = 0.957
Validation Accuracy = 0.958
EPOCH 50 :
Training Accuracy = 0.956
Validation Accuracy = 0.959
EPOCH 51 :
Training Accuracy = 0.958
Validation Accuracy = 0.962
EPOCH 52 :

Training Accuracy = 0.958
Validation Accuracy = 0.960
EPOCH 53 :
Training Accuracy = 0.959
Validation Accuracy = 0.960
EPOCH 54 :
Training Accuracy = 0.959
Validation Accuracy = 0.962
EPOCH 55 :
Training Accuracy = 0.960
Validation Accuracy = 0.962
EPOCH 56 :
Training Accuracy = 0.960
Validation Accuracy = 0.963
EPOCH 57 :
Training Accuracy = 0.961
Validation Accuracy = 0.961
EPOCH 58 :
Training Accuracy = 0.961
Validation Accuracy = 0.964
EPOCH 59 :
Training Accuracy = 0.962
Validation Accuracy = 0.965
EPOCH 60 :
Training Accuracy = 0.963
Validation Accuracy = 0.964
EPOCH 61 :
Training Accuracy = 0.962
Validation Accuracy = 0.964
EPOCH 62 :
Training Accuracy = 0.962
Validation Accuracy = 0.965
EPOCH 63 :
Training Accuracy = 0.963
Validation Accuracy = 0.965
EPOCH 64 :
Training Accuracy = 0.964
Validation Accuracy = 0.966
EPOCH 65 :
Training Accuracy = 0.964
Validation Accuracy = 0.967
EPOCH 66 :
Training Accuracy = 0.965
Validation Accuracy = 0.966
EPOCH 67 :
Training Accuracy = 0.965
Validation Accuracy = 0.968
EPOCH 68 :
Training Accuracy = 0.965
Validation Accuracy = 0.968
EPOCH 69 :
Training Accuracy = 0.966
Validation Accuracy = 0.967
EPOCH 70 :
Training Accuracy = 0.966
Validation Accuracy = 0.968
EPOCH 71 :
Training Accuracy = 0.966
Validation Accuracy = 0.969
EPOCH 72 :
Training Accuracy = 0.967
Validation Accuracy = 0.968
EPOCH 73 :
Training Accuracy = 0.967
Validation Accuracy = 0.970

Validation Accuracy = 0.970
EPOCH 74 :
Training Accuracy = 0.967
Validation Accuracy = 0.969
EPOCH 75 :
Training Accuracy = 0.968
Validation Accuracy = 0.971
EPOCH 76 :
Training Accuracy = 0.968
Validation Accuracy = 0.971
EPOCH 77 :
Training Accuracy = 0.968
Validation Accuracy = 0.970
EPOCH 78 :
Training Accuracy = 0.968
Validation Accuracy = 0.971
EPOCH 79 :
Training Accuracy = 0.969
Validation Accuracy = 0.971
EPOCH 80 :
Training Accuracy = 0.969
Validation Accuracy = 0.971
EPOCH 81 :
Training Accuracy = 0.969
Validation Accuracy = 0.971
EPOCH 82 :
Training Accuracy = 0.970
Validation Accuracy = 0.972
EPOCH 83 :
Training Accuracy = 0.970
Validation Accuracy = 0.971
EPOCH 84 :
Training Accuracy = 0.969
Validation Accuracy = 0.971
EPOCH 85 :
Training Accuracy = 0.970
Validation Accuracy = 0.972
EPOCH 86 :
Training Accuracy = 0.970
Validation Accuracy = 0.973
EPOCH 87 :
Training Accuracy = 0.971
Validation Accuracy = 0.974
EPOCH 88 :
Training Accuracy = 0.971
Validation Accuracy = 0.973
EPOCH 89 :
Training Accuracy = 0.971
Validation Accuracy = 0.972
EPOCH 90 :
Training Accuracy = 0.971
Validation Accuracy = 0.973
EPOCH 91 :
Training Accuracy = 0.971
Validation Accuracy = 0.971
EPOCH 92 :
Training Accuracy = 0.972
Validation Accuracy = 0.973
EPOCH 93 :
Training Accuracy = 0.972
Validation Accuracy = 0.973
EPOCH 94 :
Training Accuracy = 0.972
Validation Accuracy = 0.973
EPOCH 95 :

```
Training Accuracy = 0.972
Validation Accuracy = 0.972
EPOCH 96 :
Training Accuracy = 0.973
Validation Accuracy = 0.973
EPOCH 97 :
Training Accuracy = 0.972
Validation Accuracy = 0.973
EPOCH 98 :
Training Accuracy = 0.973
Validation Accuracy = 0.973
EPOCH 99 :
Training Accuracy = 0.972
Validation Accuracy = 0.973
EPOCH 100 :
Training Accuracy = 0.973
Validation Accuracy = 0.974
Optimization Finished!
Test Accuracy = 0.973
Training Finished!
Model saved
training time : 3202.92427611 seconds
```

Let's sum up all the experiments results and parameters in the following board :

Experiment	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
Optimizer	Gradient Descent	Gradient Descent	Adam Optimizer	Adam Optimizer	Adam Optimizer	Adam Optimizer	Adam Optimizer	Gradient Descent	G D
Training epochs	100	200	200	100	100	100	100	100	1
Batch size	128	128	128	50	128	128	50	128	5
learning_rate	0.1	0.1	0.1	0.0001	0.0001	0.001	0.001	0.0001	0
Validation Accuracy	0.989	0.991	0.110	0.988	0.989	0.991	0.992	0.889	0
Testing Accuracy	0.986	0.990	0.103	0.988	0.988	0.991	0.992	0.890	0
Training Time		6281.1995 seconds	6256.7613 seconds	3217.8524 seconds	3201.6292 seconds	3208.0609 seconds	3187.1497 seconds	3129.8376 seconds	3 s

In [102]:

```
X= [1,2,3,4,5] # [(0.1,128),(0.0001,128),(0.0001,50),(0.001,128),(0.001,50)]
Y=[0.986,0.890,0.891,0.977,0.973]
plt.plot(X,Y,hold= True,color = 'blue')
```

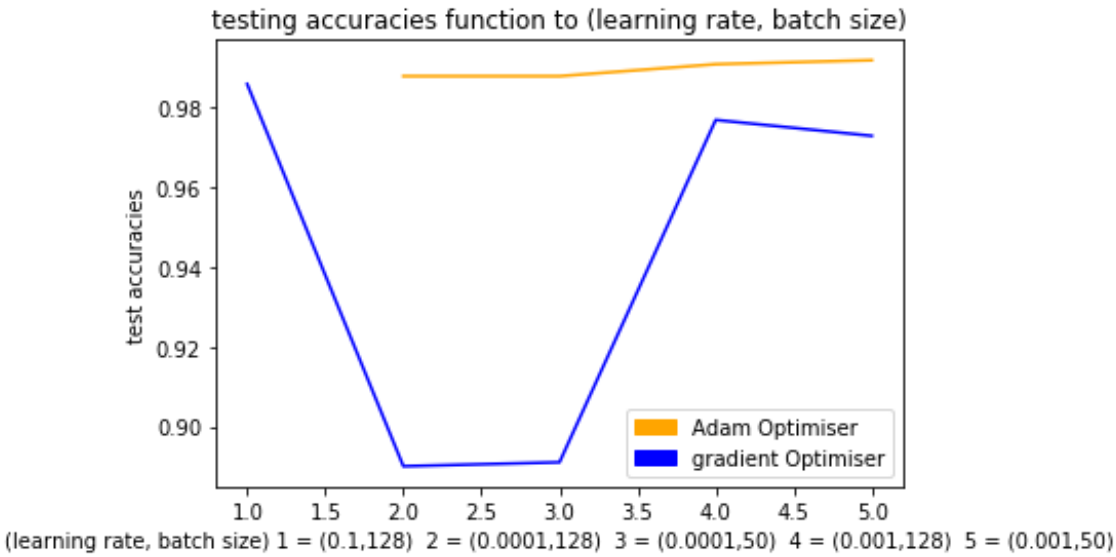
Out[102]:

```
[<matplotlib.lines.Line2D at 0x7ff607e33950>]
```

In [103]:

```
import matplotlib.pyplot as plt
import matplotlib.patches as mpatches
```

```
 # [(0.0001,128),(0.0001,50),(0.001,128),(0.001,50)]
X_2= [2,3,4,5]
Z=[0.988,0.988,0.991,0.992]
plt.plot(X_2,Z ,hold= True,color = 'orange')
orange_patch =mpatches.Patch(color = 'orange', label='Adam Optimiser')
blue_patch = mpatches.Patch(color='blue', label='gradient Optimiser')
plt.legend(handles=[orange_patch,blue_patch], loc = 4)
plt.title('testing accuracies function to (learning rate, batch size)')
plt.xlabel('(learning rate, batch size) 1 = (0.1,128)  2 = (0.0001,128)  3 = (0.0001,50)  4 = (0.001,128)  5 = (0.
001,50)')
plt.ylabel('test accuracies')
plt.show()
```



We have exluded the value of accuracy of Adam optimizer for parameters (0.1,128) in the plot because it is too low and it impacts the visualization of the other test accuracies variations between 0.9 and 0.99

Answer Yes, We've got more than 99% of accuracy. The maximum test accuracy we've got is **99.2%** and it has been obtain with **Adam Optimizer**, using a **learning rate of 0.001** and a **Batch size of 50.**

Question 2.2.2 What about applying a dropout layer on the Fully conntected layer and then retraining the model with the best Optimizer and parameters(Learning rate and Batsh size) obtained in *Question 2.2.1* ? (probability to keep units=0.75). For this stage ensure that the keep prob is set to 1.0 to evaluate the performance of the network including all nodes.

In [138]:

```
from tensorflow.contrib.layers import flatten
from sklearn.utils import shuffle
import time

def LeNet5_Dropout(data):
    # Reshape the image into a 4D tensor
    image = tf.reshape(data, [-1,28,28,1])

    # layer 1
    W_conv1 = weight_variable([5,5,1,6])
```

```
    b_conv1 = bias_variable([6])

    conv1 = tf.nn.conv2d(image, W_conv1, strides=[1, 1, 1, 1], padding='SAME') + b_conv1
    conv1 = tf.nn.relu(conv1)

    # Pulling
    conv1 = tf.nn.max_pool(conv1, ksize=[1, 2, 2, 1],strides=[1, 2, 2, 1], padding='SAME')
    #layer 2
    W_conv2 = weight_variable([5,5,6,16])
    b_conv2 = bias_variable([16])
    conv2 = tf.nn.conv2d(conv1,W_conv2, strides=[1, 1, 1, 1], padding='VALID') + b_conv2
    conv2 = tf.nn.relu(conv2)

    # Pulling
    conv2 = tf.nn.max_pool(conv2, ksize=[1, 2, 2, 1],strides=[1, 2, 2, 1], padding='VALID')
    # Flatten
    conv2 = flatten(conv2)


    # fully connected 1
    W_fully1 = weight_variable([400, 120])
    b_fully1 = bias_variable([120])
    fully1 = tf.nn.relu(tf.matmul(conv2, W_fully1) +b_fully1)

    # fully connected 2
    W_fully2 = weight_variable([120, 84])
    b_fully2 = bias_variable([84])

    fully2 = tf.nn.relu(tf.matmul(fully1, W_fully2) +  b_fully2)

    # Dropout layer
    drop = tf.nn.dropout(fully2, keep_prob)
    W_fully3 = weight_variable([84,10])
    b_fully3 = bias_variable([10])
    # fully connected 3
    with tf.name_scope('Model'):

        fully3 = tf.nn.softmax(tf.matmul(drop, W_fully3) + b_fully3)

    return fully3

def evaluate(model, X, Y, batch_size,proba):
    nb_samples = len(X)
    accuracy = 0
    sess = tf.get_default_session()
    for i in range(0, nb_samples, batch_size):
        end = i+batch_size
        batch_xs, batch_ys = X[i:end], Y[i:end]
        batch_accuracy = sess.run(acc, feed_dict={x: batch_xs, y: batch_ys, keep_prob:proba})
        accuracy += batch_accuracy*len(batch_xs)/nb_samples
    return accuracy


def train(model, cost, optimizer, X_train, y_train, file_name, training_epochs = 100,
          batch_size = 128, display_step = 10, logs_path = 'log_files/'):
    # Get starting time
    timeInit = time.time()
    # Initializing the variables
    init = tf.global_variables_initializer()
    # Create a summary to monitor cost tensor
    tf.summary.scalar("Loss", cost)
    # Merge all summaries into a single op
    merged_summary_op = tf.summary.merge_all()
    # Initialize saver
```

```
    saver = tf.train.Saver()

    print ("Start Training!")

    # Launch the graph for training
    with tf.Session() as sess:
        sess.run(init)
        # op to write logs to Tensorboard
        summary_writer = tf.summary.FileWriter(logs_path, graph=tf.get_default_graph())
        # Training cycle
        for epoch in range(training_epochs):
            # Shuffle the training set
            X_train, y_train = shuffle(X_train, y_train)
            avg_cost = 0.
            nb_samples = len(X_train)
            # Loop over all batches
            for i in range(0, nb_samples, batch_size):
                end = i + batch_size
                batch_xs, batch_ys = X_train[i:end], y_train[i:end]
                # Run optimization op (backprop), cost op (to get loss value)
                # and summary nodes
                sess.run(optimizer, feed_dict={x: batch_xs, y: batch_ys, keep_prob: 0.75})
                # Compute average loss
                avg_cost += c / (nb_samples/batch_size)
            # Display logs per epoch step
            if (epoch+1) % display_step == 0:
                print ('Epoch : ', (epoch+1))
                print ("Trainig Accuracy:", evaluate(model, X_train,y_train, batch_size,1.0))
                print(" Validation Accuracy:",evaluate(model,X_validation, y_validation, batch_size,1.0))
        print ("Training Finished!")
        # Calculate final accuracy
        print ("Test_Accuracy:", evaluate(model, X_test, y_test, batch_size,1.0))

    time_tot = time.time()-timeInit


# tf Graph Input:  mnist data image of shape 28*28=784
x = tf.placeholder(tf.float32, [None, 784], name='InputData')
# 0-9 digits recognition,  10 classes
y = tf.placeholder(tf.float32, [None, 10], name='LabelData')
# Keep probability for the dropout layer
keep_prob = tf.placeholder(tf.float32)


with tf.name_scope('Model'):
    # Model
    model = LeNet5_Dropout(x)
with tf.name_scope('Loss'):
    # Cross Entropy
    cost = tf.reduce_mean(-tf.reduce_sum(y*tf.log(model), reduction_indices=1))

acc = tf.equal(tf.argmax(model, 1), tf.argmax(y, 1))
acc = tf.reduce_mean(tf.cast(acc, tf.float32))

learning_rate = 0.001
batch_size = 50

optimizer = tf.train.AdamOptimizer(learning_rate).minimize(cost)

# AO LR=0.001 BS=128 with Dropout

train(model, cost, optimizer, X_train, y_train, 'dropout' ,training_epochs = 100, batch_size = batch_size)
```

```
Start Training!
```

```
Epoch :  10
Trainig Accuracy: 0.974345456308
 Validation Accuracy: 0.97340000391
Epoch :  20
Trainig Accuracy: 0.983236367648
 Validation Accuracy: 0.985200006962
Epoch :  30
Trainig Accuracy: 0.986363640319
 Validation Accuracy: 0.985600004196
Epoch :  40
Trainig Accuracy: 0.990781822313
 Validation Accuracy: 0.986000006795
Epoch :  50
Trainig Accuracy: 0.990036367449
 Validation Accuracy: 0.984400001168
Epoch :  60
Trainig Accuracy: 0.991018185778
 Validation Accuracy: 0.987200003862
Epoch :  70
Trainig Accuracy: 0.993436367945
 Validation Accuracy: 0.985400006771
Epoch :  80
Trainig Accuracy: 0.994672730619
 Validation Accuracy: 0.988200002313
Epoch :  90
Trainig Accuracy: 0.994600003362
 Validation Accuracy: 0.987200006247
Epoch :  100
Trainig Accuracy: 0.995800002868
 Validation Accuracy: 0.989200007319
Training Finished!
Test_Accuracy: 0.988200003207
```

Answer We have obtained a good accuracy using a dropout layer, but there isn't a real improvment comparing to what we've got before. So maybe the drop out layer can be more useful in some other cases where overfitting is more likely to happen.