In [1]:
```
%pip install captcha
%pip install opencv-python
!apt update && apt install -y libsm6 libxext6
!pip install opencv-python-headless
!pip install opencv-contrib-python-headless
%pip install keras


import argparse
import json
import string
import os
import shutil
import uuid
from captcha.image import ImageCaptcha

!pip install "opencv-python-headless<4.3"

import itertools

import os

import numpy as np
from random import random, randint, choices

import keras
from keras.models import Sequential, Model
from keras.layers import Dense, Dropout, Activation, Flatten
from keras.layers import Conv2D, MaxPooling2D, Input
import matplotlib.pyplot as plt
import cv2
```

```
Requirement already satisfied: captcha in c:\users\karan\anaconda3\lib\site-pac
kages (0.4)
Requirement already satisfied: Pillow in c:\users\karan\anaconda3\lib\site-pack
ages (from captcha) (9.0.1)
Note: you may need to restart the kernel to use updated packages.

WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\karan\a
naconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\karan\a
naconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\karan\a
naconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\karan\a
naconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\karan\a
naconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\karan\a
naconda3\lib\site-packages)

Requirement already satisfied: opencv-python in c:\users\karan\anaconda3\lib\si
te-packages (4.6.0.66)
Requirement already satisfied: numpy>=1.14.5 in c:\users\karan\anaconda3\lib\si
te-packages (from opencv-python) (1.23.0)
Note: you may need to restart the kernel to use updated packages.
```

```
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\karan
\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\karan
\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\karan
\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\karan
\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\karan
\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\karan
\anaconda3\lib\site-packages)
'apt' is not recognized as an internal or external command,
operable program or batch file.
```

```
Requirement already satisfied: opencv-python-headless in c:\users\karan\anacond
a3\lib\site-packages (3.4.18.65)
Requirement already satisfied: numpy>=1.14.5 in c:\users\karan\anaconda3\lib\si
te-packages (from opencv-python-headless) (1.23.0)
```

```
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\karan\a
naconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\karan\a
naconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\karan\a
naconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\karan\a
naconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\karan\a
naconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\karan\a
naconda3\lib\site-packages)
```

```
Requirement already satisfied: opencv-contrib-python-headless in c:\users\karan
\anaconda3\lib\site-packages (4.6.0.66)
Requirement already satisfied: numpy>=1.19.3 in c:\users\karan\anaconda3\lib\si
te-packages (from opencv-contrib-python-headless) (1.23.0)
```

```
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\karan\a
naconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\karan\a
naconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\karan\a
naconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\karan\a
naconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\karan\a
naconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\karan\a
naconda3\lib\site-packages)
```

```
Requirement already satisfied: keras in c:\users\karan\anaconda3\lib\site-packa
ges (2.9.0)
Note: you may need to restart the kernel to use updated packages.
```

```
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\karan
\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\karan
```

```
\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\karan
\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\karan
\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\karan
\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\karan
\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\karan
\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\karan
\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\karan
\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\karan
\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\karan
\anaconda3\lib\site-packages)
WARNING: Ignoring invalid distribution -pencv-python-headless (c:\users\karan
\anaconda3\lib\site-packages)

Requirement already satisfied: opencv-python-headless<4.3 in c:\users\karan\ana
conda3\lib\site-packages (3.4.18.65)
Requirement already satisfied: numpy>=1.14.5 in c:\users\karan\anaconda3\lib\si
te-packages (from opencv-python-headless<4.3) (1.23.0)

C:\Users\KARAN\anaconda3\lib\site-packages\scipy\__init__.py:146: UserWarning:
A NumPy version >=1.16.5 and <1.23.0 is required for this version of SciPy (det
ected version 1.23.0
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
```

In [2]:

```
alphabet_all = list('qwertyupasdfghjkzxcvbnm23456789QWERTYUPKJHGFDSAZXCVBNM')
alphabet = list('qwertyupasdfghjkzxcvbnm23456789')#QWERTYUIOPLKJHGFDSAZXCVBNM')
num_alphabet = len(alphabet)
def _gen_captcha(img_dir, num_of_letters, num_of_repetition, width, height):
    if os.path.exists(img_dir):
        shutil.rmtree(img_dir)
    if not os.path.exists(img_dir):
        os.makedirs(img_dir)

    image = ImageCaptcha(width=width, height=height)

    for counter in range(num_of_repetition):
        i = choices(alphabet_all, k=5)
        captcha = ''.join(i)
        fn = os.path.join(img_dir, '%s_%s.png' % (captcha, uuid.uuid4()))
        image.write(captcha, fn)


def gen_dataset(path, num_of_repetition, num_of_letters, width, height):
    _gen_captcha(os.path.join(path, 'data'), num_of_letters, num_of_repetition, w
    print('Finished Data Generation')
```

In [3]:
```
BATCH_SIZE = 256
NUM_OF_LETTERS = 5
EPOCHS = 50
IMG_ROW, IMG_COLS = 50, 135

# Non-configs
PATH = 'C:\\Users\\KARAN\\Desktop\\10k'
DATA_PATH = os.path.join(PATH, 'train')
```

In [4]:
```
def load_data(path, test_split=0.15):
    print ('loading dataset...')
    y_train = []
    y_test = []
    x_train = []
    x_test = []

    # r=root, d=directories, f = files
    counter = 0
    for r, d, f in os.walk(path):
        for fl in f:
            if '.png' in fl:
                flr = fl.split('_')[0]
                counter += 1
                label = np.zeros((NUM_OF_LETTERS, num_alphabet))
                for i in range(NUM_OF_LETTERS):
                    label[i, alphabet.index(flr[i].lower())] = 1
#                 label = np.zeros((50, 1))
#                 for i in range(5):
#                     label[i*5+int(flr[i])] = 1

                img = cv2.imread(os.path.join(r, fl))
                img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
                img = cv2.resize(img, (int(135/2), int(50/2)), interpolation=cv2.
                img = np.reshape(img, (img.shape[0], img.shape[1], 1))

                if random() < test_split:
                    y_test.append(label)
                    x_test.append(img)
                else:
                    y_train.append(label)
                    x_train.append(img)

    print('dataset size:', counter, '(train=%d, test=%d)' % (len(y_train), len(y_
    return np.array(x_train), np.array(y_train), np.array(x_test), np.array(y_tes
```

In [5]:
```
if not os.path.exists(DATA_PATH):
    print('Generating Dataset')
    gen_dataset(DATA_PATH,100000, NUM_OF_LETTERS, IMG_COLS, IMG_ROW)
```

In [ ]:

In [6]:
```
x_train, y_train, x_test, y_test = load_data(DATA_PATH)
x_train[0]
x_train = x_train.astype('float32')
x_test = x_test.astype('float32')
x_train /= 255
x_test /= 255
```

```
loading dataset...
dataset size: 100000 (train=84946, test=15054)
```

In [ ]:
```
```

In [7]:
```
print(x_train.shape)
print(y_train.shape)
print(x_test.shape)
print(y_test.shape)
```

```
(84946, 25, 67, 1)
(84946, 5, 31)
(15054, 25, 67, 1)
(15054, 5, 31)
```

In [8]:
```
s_train = []
s_test = []
for i in range(NUM_OF_LETTERS):
    s_train.append(y_train[:, i, :])
    s_test.append(y_test[:, i, :])
```

In [9]:
```
save_dir = os.path.join(PATH, 'saved_models')
model_name = 'keras_cifar10_trained_model.h5'
```

In [10]:
```
input_layer = Input((25, 67, 1))
x = Conv2D(filters=32, kernel_size=(5, 5), padding='same', activation='relu')(inp
x = MaxPooling2D(pool_size=(2, 2))(x)

x = Conv2D(filters=48, kernel_size=(5, 5), padding='same', activation='relu')(x)
x = MaxPooling2D(pool_size=(2, 2))(x)

x = Conv2D(filters=64, kernel_size=(5, 5), padding='same', activation='relu')(x)
x = MaxPooling2D(pool_size=(2, 2))(x)

x = Dropout(0.3)(x)
x = Flatten()(x)
x = Dense(512, activation='relu')(x)
x = Dropout(0.3)(x)

out = [Dense(num_alphabet, name='digit%d' % i, activation='softmax')(x) for i in
# out = Dense(num_alphabet*5, activation='sigmoid')(x)

model = Model(inputs=input_layer, outputs=out)
```

In [11]:
```
# model_path = os.path.join(save_dir, model_name)
# model = keras.models.load_model(model_path)
```

In [12]:
```
# initiate Adam optimizer

model.compile(loss='binary_crossentropy',
              optimizer='adam',
              metrics=['accuracy'])


model.summary()
```

```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   Connected to
==================================================================================
 input_1 (InputLayer)        [(None, 25, 67, 1)]       0         []

 conv2d (Conv2D)             (None, 25, 67, 32)        832       ['input_1[0]
[0]']

 max_pooling2d (MaxPooling2D)  (None, 12, 33, 32)      0         ['conv2d[0]
[0]']

 conv2d_1 (Conv2D)           (None, 12, 33, 48)        38448     ['max_poolin
g2d[0][0]']

 max_pooling2d_1 (MaxPooling2D)  (None, 6, 16, 48)     0         ['conv2d_1
[0][0]']

 conv2d_2 (Conv2D)           (None, 6, 16, 64)         76864     ['max_poolin
g2d_1[0][0]']

 max_pooling2d_2 (MaxPooling2D)  (None, 3, 8, 64)      0         ['conv2d_2
[0][0]']

 dropout (Dropout)           (None, 3, 8, 64)          0         ['max_poolin
g2d_2[0][0]']

 flatten (Flatten)           (None, 1536)              0         ['dropout[0]
[0]']

 dense (Dense)               (None, 512)               786944    ['flatten[0]
[0]']

 dropout_1 (Dropout)         (None, 512)               0         ['dense[0]
[0]']

 digit0 (Dense)              (None, 31)                15903     ['dropout_1
[0][0]']

 digit1 (Dense)              (None, 31)                15903     ['dropout_1
[0][0]']

 digit2 (Dense)              (None, 31)                15903     ['dropout_1
[0][0]']
```

```
 digit3 (Dense)                 (None, 31)           15903       ['dropout_1
[0][0]']

 digit4 (Dense)                 (None, 31)           15903       ['dropout_1
[0][0]']

==================================================================================
====================
Total params: 982,603
Trainable params: 982,603
Non-trainable params: 0
__________________________________________________________________________________
__________________
```

In [13]:
```
hist_train_loss_digit = {i:[] for i in range(5)}
hist_test_loss_digit = {i:[] for i in range(5)}

hist_train_acc_digit = {i:[] for i in range(5)}
hist_test_acc_digit = {i:[] for i in range(5)}

hist_train_loss = []
hist_test_loss = []

hist_train_acc = []
hist_test_acc = []
```

In [14]:
```
digit_acc = [[] for _ in range(NUM_OF_LETTERS)]
val_digit_acc = [[] for _ in range(NUM_OF_LETTERS)]
loss = []
val_loss = []
```

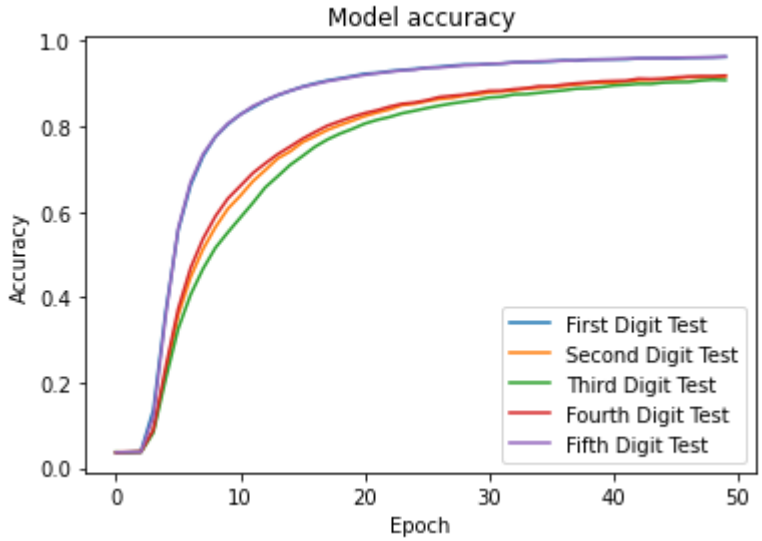
In [15]:
```
history = model.fit(x_train, s_train,
                    batch_size=BATCH_SIZE,
                    epochs=EPOCHS,
                    verbose=1,
                    validation_data=(x_test, s_test)
                   )
```

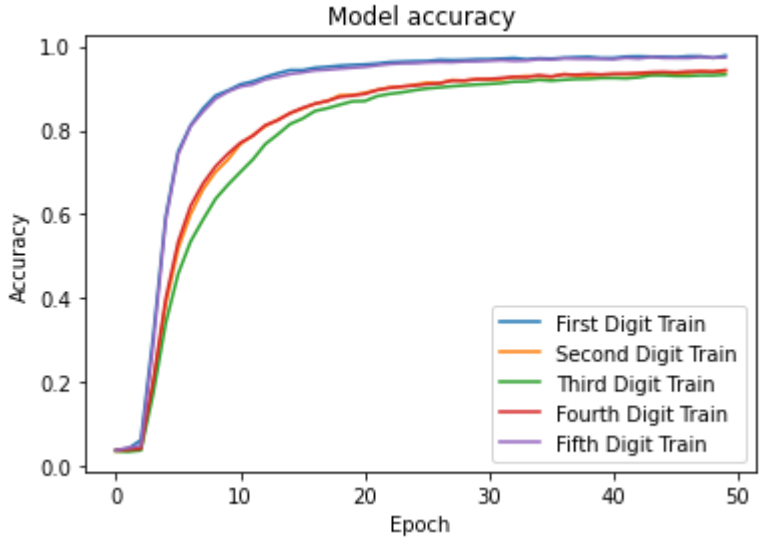
```
9404 - val_loss: 0.0755 - val_digit0_loss: 0.0072 - val_digit1_loss: 0.0191 -
val_digit2_loss: 0.0224 - val_digit3_loss: 0.0186 - val_digit4_loss: 0.0082 -
val_digit0_accuracy: 0.9672 - val_digit1_accuracy: 0.9196 - val_digit2_accura
cy: 0.9047 - val_digit3_accuracy: 0.9180 - val_digit4_accuracy: 0.9621
Epoch 29/50
332/332 [==============================] - 98s 295ms/step - loss: 0.1160 - di
git0_loss: 0.0136 - digit1_loss: 0.0285 - digit2_loss: 0.0321 - digit3_loss:
0.0279 - digit4_loss: 0.0139 - digit0_accuracy: 0.9446 - digit1_accuracy: 0.8
723 - digit2_accuracy: 0.8579 - digit3_accuracy: 0.8744 - digit4_accuracy: 0.
9431 - val_loss: 0.0746 - val_digit0_loss: 0.0073 - val_digit1_loss: 0.0189 -
val_digit2_loss: 0.0220 - val_digit3_loss: 0.0185 - val_digit4_loss: 0.0078 -
val_digit0_accuracy: 0.9677 - val_digit1_accuracy: 0.9182 - val_digit2_accura
cy: 0.9069 - val_digit3_accuracy: 0.9182 - val_digit4_accuracy: 0.9639
Epoch 30/50
332/332 [==============================] - 98s 296ms/step - loss: 0.1142 - di
git0_loss: 0.0134 - digit1_loss: 0.0281 - digit2_loss: 0.0316 - digit3_loss:
0.0273 - digit4_loss: 0.0138 - digit0_accuracy: 0.9447 - digit1_accuracy: 0.8
746 - digit2_accuracy: 0.8621 - digit3_accuracy: 0.8788 - digit4_accuracy: 0.
9440 - val_loss: 0.0706 - val_digit0_loss: 0.0066 - val_digit1_loss: 0.0178 -
```

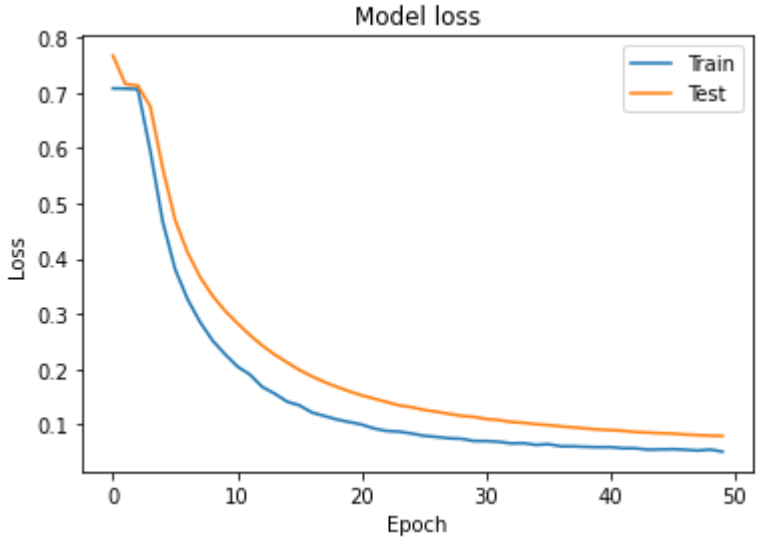
In [16]:

```
digit_acc = [[] for _ in range(NUM_OF_LETTERS)]
val_digit_acc = [[] for _ in range(NUM_OF_LETTERS)]
loss = []
val_loss = []


def plot_diagram(digit_acc_now, val_digit_acc_now, loss_now, val_loss_now):
    global digit_acc, val_digit_acc, loss, val_loss


    for i in range(NUM_OF_LETTERS):
        digit_acc[i].extend(digit_acc_now[i])
        val_digit_acc[i].extend(val_digit_acc_now[i])
    loss.extend(loss_now)
    val_loss.extend(val_loss_now)

    for i in range(NUM_OF_LETTERS):
        s = {0:'First', 1:'Second', 2:'Third', 3:'Fourth', 4:'Fifth'}[i]
        # plt.plot(val_digit_acc[i], label='%s Digit Train' % s)
        plt.plot(digit_acc[i], label='%s Digit Test' % s)

    plt.title('Model accuracy')
    plt.ylabel('Accuracy')
    plt.xlabel('Epoch')
    plt.legend()
    plt.show()

    for i in range(NUM_OF_LETTERS):
        s = {0:'First', 1:'Second', 2:'Third', 3:'Fourth', 4:'Fifth'}[i]
        plt.plot(val_digit_acc[i], label='%s Digit Train' % s)
        # plt.plot(digit_acc[i], label='%s Digit Test' % s)

    plt.title('Model accuracy')
    plt.ylabel('Accuracy')
    plt.xlabel('Epoch')
    plt.legend()
    plt.show()



    # Plot training & validation loss values

    plt.plot(val_loss, label='Train')
    plt.plot(loss, label='Test')
    plt.title('Model loss')
    plt.ylabel('Loss')
    plt.xlabel('Epoch')
    plt.legend()
    plt.show()
```

In [17]:
```
plot_diagram(

    [history.history['digit%d_accuracy' % i] for i in range(NUM_OF_LETTERS)],
    [history.history['val_digit%d_accuracy' % i] for i in range(NUM_OF_LETTERS)],
    history.history['loss'],
    history.history['val_loss'],
)
```







In [18]:
```
# Save model and weights
if not os.path.isdir(save_dir):
    os.makedirs(save_dir)
model_path = os.path.join(save_dir, model_name)
model.save(model_path)
print('Saved trained model at %s ' % model_path)
```

```
Saved trained model at C:\Users\KARAN\Desktop\10k\saved_models\keras_cifar10_tr
ained_model.h5
```

In [19]:
```
# Score trained model.
scores = model.evaluate(x_train, s_train, verbose=1)
print('Train loss:     %f' % np.mean(scores[0:5]))
acc = 1.
for i in range(5):
    acc *= scores[6+i]
print('Train accuracy: %.2f' % (acc * 100.))
```

```
2655/2655 [==============================] - 39s 15ms/step - loss: 0.0262 - dig
it0_loss: 0.0023 - digit1_loss: 0.0068 - digit2_loss: 0.0079 - digit3_loss: 0.0
068 - digit4_loss: 0.0025 - digit0_accuracy: 0.9920 - digit1_accuracy: 0.9774 -
digit2_accuracy: 0.9762 - digit3_accuracy: 0.9772 - digit4_accuracy: 0.9908
Train loss:     0.009985
Train accuracy: 91.65
```

In [20]:
```
scores = model.evaluate(x_test, s_test, verbose=1)
print('Test loss:     %f' % np.mean(scores[0:5]))
acc = 1.
for i in range(5):
    acc *= scores[6+i]
print('Test accuracy: %.2f' % (acc * 100.))
```

```
471/471 [==============================] - 7s 15ms/step - loss: 0.0513 - digit0
_loss: 0.0049 - digit1_loss: 0.0129 - digit2_loss: 0.0152 - digit3_loss: 0.0128
- digit4_loss: 0.0055 - digit0_accuracy: 0.9779 - digit1_accuracy: 0.9412 - dig
it2_accuracy: 0.9327 - digit3_accuracy: 0.9433 - digit4_accuracy: 0.9731
Test loss:     0.019435
Test accuracy: 78.80
```

In [ ]:

In [ ]:

In [ ]: