Queensland University of Technology

Queensland University of Technology

IFN680: ASSIGNMENT 2: ADVANCED TOPICS ON ARTIFCIAL INTELLIGENCE

STUDENT NAME:

DARSHEEL DESHPANDE	**N10287213**
KARTHIK KADADEVARMATH	**N10281797**
SUPRITH KANGOKAR	**N10124021**

Introduction:

This project is aimed at implementing deep neural network to predict whether the two images belong to the same class or not using image classifier techniques by following a quite generic convolutional neural network using siamese network. The project is developed using python 3.7 framework with Spyder IDE using Anaconda open source platform. The library that has been used and imported in performing the project are Tensorflow and Keras which are open source software library that are used in developing applications based on neural networks.

A neural network is defined as an assembly which is interconnected of simple processing elements, units or nodes. With the functionality based on the neurons. The networks which are processed will then be stored in the interunit connection strengths, or weights, which are obtained by the process or training the patterns (Gurney, 2014).

Siamese network is defined as the non-linear deep models which are used find ways in a to overcome the problems in learning theory due to the embedding capabilities. The Siamese network has two networks which are identical and parameterized. These networks are used for non-linear mapping when the input is images. Siamese network is a resembles conventional convolutional layers, which is advantageous as it offers an invariance for the structure of Siamese networks (Harandi, Kumar, & Nock). Due to this, we have implemented Siamese network for this Artificial intelligence assignment.

Implementation of code and function use:

- The open source artificial intelligence library called TensorFlow was for the implementation of assignment. To install this used the command **"! pip Install tensorflow".**
- The high level neural- network api "Keras was used because it allows easy and fast prototyping and also supports CNN (Convolution neural networks). To install this **"! pip install keras"** was used. Later it was called in from tensor flow for importing.
- **fashion_mnst** package was imported from **keras. datasets** for using the image in that package
- Input, Flatten, Dense, Dropout lambda package were imported from keras. layers library for developing Siamese model and Euclidean distance.
- Loading of data from fashion_mnst takes place in **load_fashionmnist_data()** function. In which data is **(x1,y1) (x2,y3) .**
- Concatenating is done dataset in variable "dataset" spilt them later for training and testing purposes
- We later normalized this data into 0 to 1 so that the image comes in general range because it is necessary for model to learn this image, the images should be in general range.
- Later the data is spilt and pairs are formed. for splitting the data it is first necessary to calculate the indexes of all train and test data labels. first index of train data is found at **"train_indexes"**

- Later indexes for test1 label are found which is 20% of train data. This is done by passing the minimum threshold.
- Finding Indexes for test2 labels are done which consist of (“dress”,” sneaker”,” bag”,” shirt”).
- Pairs are made using data and **test1_indices, test2_indices**, **test1_union_test2_indices** for making **test1_data_pairs**, **test2_data_pairs**, **test1utest2_data_pairs**.
- **makepairs_images** funcition is used which takes data and labels and forms random pairs with same or different set of images.
- After forming the pairs, the next function is regarding calculating the accuracy
- To calculate the accuracy of the model the function **model_accuracy** is used to pass in the Siamese model metrics.
- After passing the **model_accuracy** function the custom loss function is defined which will calculate the loss of difference and similarity by using contrastive loss function with the function name **custom_loss_function** with the limit set to 1and squaring the model prediction and then calculating loss on base of difference between 0 and the predicted value.
- The **L2_distance ()** function is used to calculate the Euclidean distance between 2 feature vectors of the same size by obtaining 2 feature vectors from Siamese network namely **x1_features , x2_features**. The difference if these 2 vectors are found.
- The function **model_shape()** is used in setting the output shape of the model. Then the architecture of the Siamese model is declared consisting of 6 layers which is a base network. The base network consisting of 6 layers where the first layer is incorporated with the Conv2D of keras layers which is 2D convolutional layer used for spatial convolution of the images. The second layer is enclosed with the Flatten layer which will flatten the input without affecting the batch size. The rest if the layers where been enclosed with the Dropout which will help in randomly setting a function rate of input units to 0 and update every time during training and the Dense which will densely connect the neural network.
- After declaring the architecture of the Siamese model in the previous method then the compiled model of Siamese network is being returned in the function **Build_siamese_model()** in the following steps:
    - Creating network architecture with the input shape of the model of 28*28*1.
    - Declaring keras input object as **input_1** and **input_2.**
    - Declaring two objects of the shared network’s as **shared1** and **shared2.**
    - To make the output as one model concatenating two base model instances by using lambda function.
    - After concatenating declaring one final model.
    - At last compiling the model with SGD optimizer.

After building the Siamese model the **train ()** function is implemented to train the developed model by passing parameters to the function as **model,train_data, train_labels,val_data,val_labels,batchsize,epochs.**

- The function **plt_acc_curve** does plotting for accuracy over epochs for training and validation data for training **train_acc** is implemented and for validation **val_acc** is used. Then the

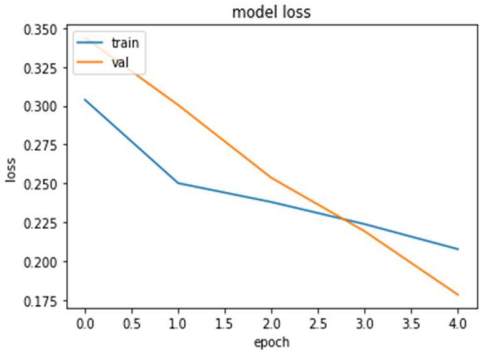
function **calculate_accuracy** calculates accuracy of model predicting with respect to ground truth label. Then **print_accuracy** function is used to print accuracy of different testing sets this accuracy is calculated based on the Siamese model.

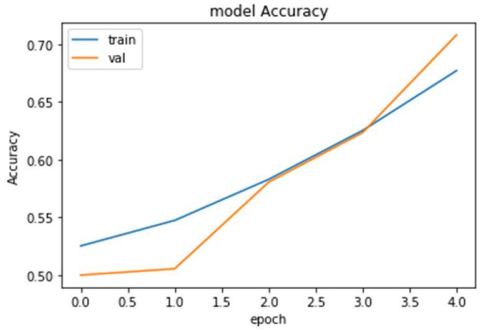
- **The draw_image_pairs function** is used to plot image pairs with their ground truth labels and predicted labels based on the conditions. The conditions for predicted label is that it will be equal to 'same' if the value of Prediction which is model prediction will be greater than 0.5 else the value in predicted label will be "differ". The condition for ground truth labels is that if the value of groundtruth table is 1 then the value in Ground truth label will be similar else it will display "differ". They are displayed in result under the name of True Label and predicted label which shows the ground truth labels and predicted labels respectively.
- The next section is of calling of functions to perform respective activity. The first function which is called in program is of load_fashion mnist which loads the datasets .The labels which are separated in train and test are being listed and spilt_and_make_pairs funcition is being called to make spilt and suitable pairs .pair of train data of labels (train_x,train_y),test1(test1_,test1_y),test2(test2_x,test2_y) ,pairs of image based on union (test1uttest2_x,test1uttest2_y). Later the siamese_model function is called to build siamese model. The training of model on train_x and validation that is test1_x takes place we have trained model for 5 epochs only. After training of model both the plot function for loss ("**plt_loss_curve**") and accuracy("**plt_acc_curve**") are called which displays the curve for loss and accuracy which has been shown in result section of this report.
- The next section in code conducts test on different sets of testing set to calculate accuracy and generalization capacity .Test are conducted on '**test1_x**' using the dataset labels as ["top","trousers","coat","sandal","ankle boot"], **'test2_x**' is done using dataset labels as "dress ["dress", "sneaker", "bag","shirt"], '**test1uttest2_x**'.pairs of images drawn with help of ground truth and predicted label of test set 1.The pair will change based on the pair number .Same thing of drawing pairs of images with the ground truth and predicted label for test set 2 and for union of test1 and test 2 is done. We can see the result in result section of report. The result displays if the pair had similar image or the image differed from each other.

RESULT:

TEST CASE 1:

For test set 1 we changed pair no =1
For test set 2 we changed pair no =1
For union of test1 and test2 dataset we changed pair no =2





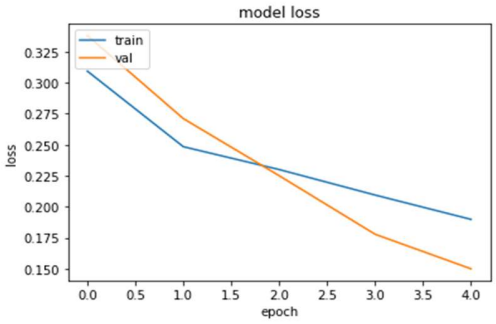

```
Accuracy on test set 1: 70.77%
Accuracy on test set 2: 60.06%
Accuracy on union of test1 and test2 dataset: 65.54%
    True Label:  differ  Predicted Label:  differ
```

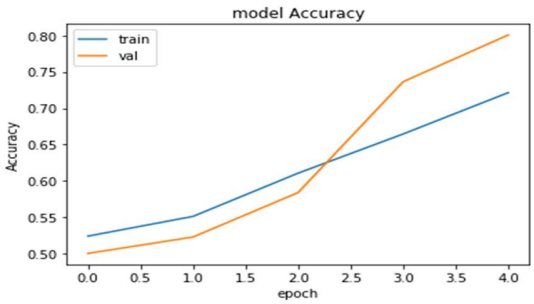
```
   True Label:  differ  Predicted Label:  differ
```

```
   True Label:  same  Predicted Label:  same
```

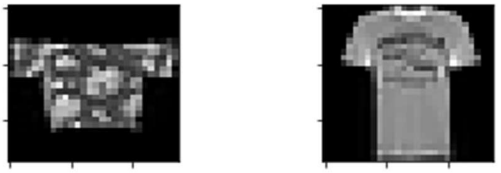
TEST CASE 2:

For test set 1 we changed pair no =2
For test set 2 we changed pair no =2
For union of test1 and test2 dataset we changed pair no =1






```
Accuracy on test set 1: 80.05%
Accuracy on test set 2: 71.55%
Accuracy on union of test1 and test2 dataset: 73.83%
    True Label:  same  Predicted Label:  same
```

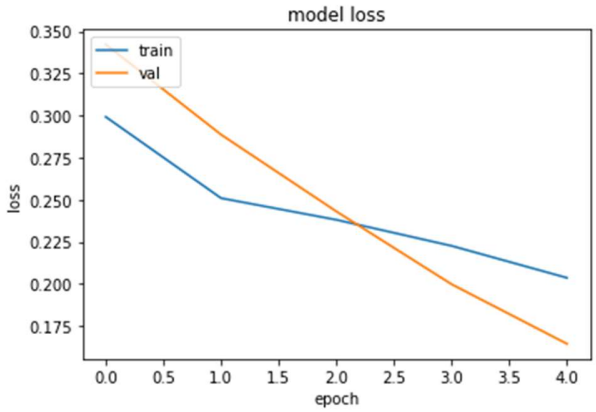


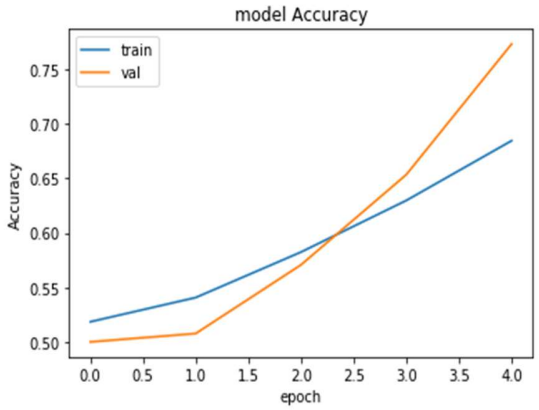
```
True Label:  same  Predicted Label:  same
```

```
True Label:  differ  Predicted Label:  differ
```

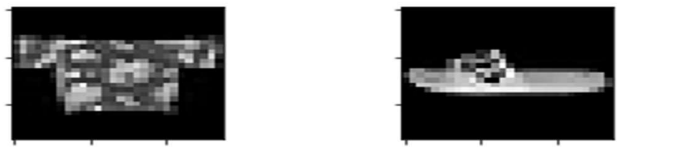
TEST CASE 3:

For test set 1 we changed pair no =3
For test set 2 we changed pair no =3
For union of test1 and test2 dataset we changed pair no =3

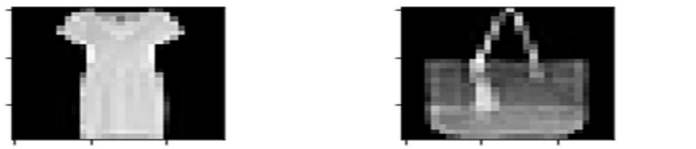




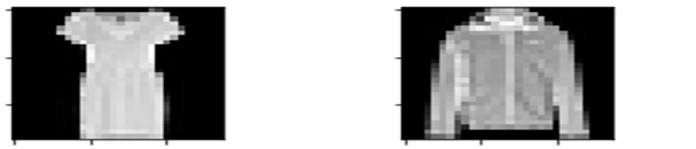

```
Accuracy on test set 1: 77.26%
Accuracy on test set 2: 64.69%
Accuracy on union of test1 and test2 dataset: 70.12%
    True Label:  differ  Predicted Label:  same
```



```
   True Label:  differ  Predicted Label:  same
```



```
   True Label:  differ  Predicted Label:  same
```



Conclusion:

The MNIST dataset has been used by initially concatenating and normalizing the dataset and then splitting the dataset into train data and test data by dividing the dataset into pairs. These pairs have been divided into 3 sets namely **Test1_x, Test2_x** and **test1uttest2_x** respectively. Then these 3 sets were tested with 3 test cases where the Graph will depict the losses and accuracy whereas the pair of images will depict whether they belong to same class or not.

References:

Gurney, K. (2014). *An introduction to neural networks*: CRC press.

Harandi, M., Kumar, S. R., & Nock, R. Siamese Networks: A Thing or Two to Know.