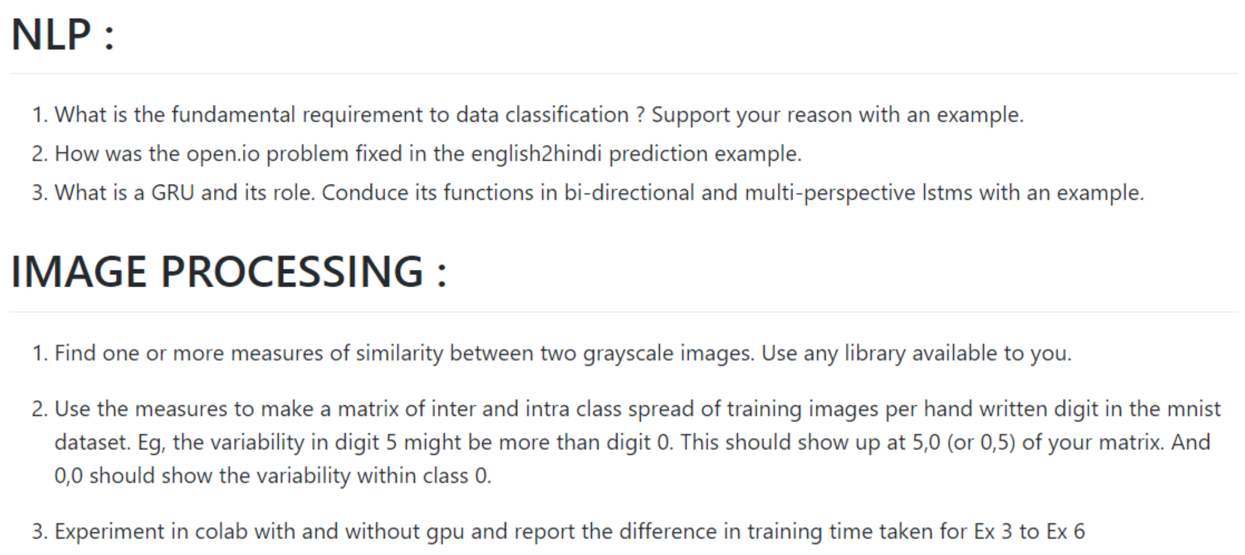
Name : Nischith Shadagopan M N   Roll number : CS18B102

Problem Statement



NLP :

1. What is the fundamental requirement to data classification ? Support your reason with an example.
2. How was the open.io problem fixed in the english2hindi prediction example.
3. What is a GRU and its role. Conduce its functions in bi-directional and multi-perspective lstms with an example.

IMAGE PROCESSING :

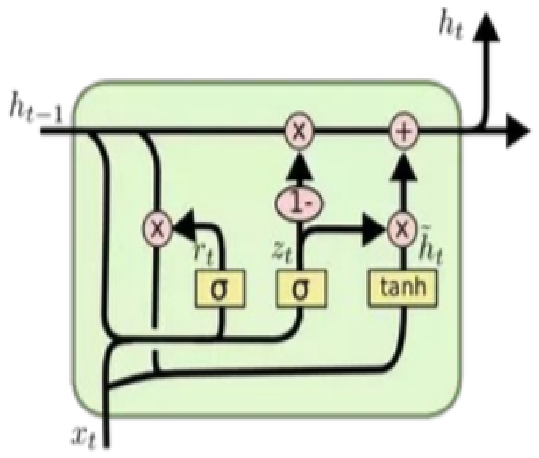
1. Find one or more measures of similarity between two grayscale images. Use any library available to you.
2. Use the measures to make a matrix of inter and intra class spread of training images per hand written digit in the mnist dataset. Eg, the variability in digit 5 might be more than digit 0. This should show up at 5,0 (or 0,5) of your matrix. And 0,0 should show the variability within class 0.
3. Experiment in colab with and without gpu and report the difference in training time taken for Ex 3 to Ex 6

NLP:

1. The fundamental requirement to data classification is relevant data and an appropriate classification model. Both of these are very important to get a good classifier. Suppose we have lots of relevant data but our model is not complex enough, then we will end up overfitting and our model might perform well on training data but will fail on test data. Similarly if we have a good/complex model but if we have less relevant data, the model will underfit and will not be able to perform well on real life data. If I had to choose one, I would choose relevant data because I believe this is a harder task than choosing the right model. Here by relevant data, I mean data that covers all kinds of different scenarios with sufficient magnitude. This is because the model can only learn patterns actually present in the data. Hence we should ensure that whatever patterns we want the model to learn is well represented in the data. We should also ensure our data is not biased as if it is biased, the model will also end up being biased.

   As an example I would like to share my experience as part of the Computer Vision and Intelligence club at IITM. As part of this club me and my friends developed an automated attendance system. Details of this can be found here : link
   -If the data is biased that is if we have many pictures of the same person as part of the training data, then our attendance system will end up being biased and will favour the person who has the most number of training pictures.
   -Also variance in training data is important. For example, if we only have training pictures taken in good lighting conditions, then our model will fail in bad lighting conditions.
2. I am unable to open the english2hindi notebook.

3. Gated Recurrent Unit is a modification to the RNN hidden layer that makes it better at remembering things from the distant past and also helps with the vanishing gradient problem. It can also be thought of as a simplified version of LSTM.The equations governing a GRU is as follows:



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$
$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$
$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

The intuition behind its working is as follows:
-The third equation calculates a new candidate to replace the existing memory cell
-The second equation calculates the relevance gate of the existing memory cell to the candidate to replace it
-The first equation calculates how much to retain from the existing memory cell and how much to add from the new candidate. Also called update gate
-The last equation calculates the new memory cell

In this way, the memory cell can remember things for a long time as the update gate takes a value close to zero for considerably negative values (due to sigmoid function) and hence most of the existing memory cell can be retained. This also solves the vanishing gradient problem.
In a sentence like:
I am from Chennai, …. I study in IIT
We need to remember the speaker is from Chennai to conclude that he studies in IIT Chennai. GRU can be used for this as a memory cell can remember that the speaker is from Chennai to make deductions later.

Bi-directional LSTM:
When we have to process sentences such as:
-Everyone loves to watch Tom and Jerry
-Everyone loves to watch Tom Hiddleston as Loki
We need information from the past as well as the present to infer information about Tom. For this we can use a Bi-directional LSTM. Bi-directional LSTM contains a forward running chain of RNNs and a backward running chain of RNNs. Hence information from the forward chain as well as the backward chain is used to infer information. One

drawback of this is that to process the information, we need to wait until all the information is received. Hence this cannot be used in real time applications.

Any one of {RNN, GRU, LSTM} can be used as a module in the bi-directional LSTM. GRU and LSTM have a better ability in remembering things longer. GRU is like a simplified version of LSTM and hence trains faster than a LSTM but at the cost of accuracy.

Multi perspective LSTM:

Multi perspective LSTMs use Bi directional LSTMs along with Multi-Perspective Context Matching. Here the goal of the Bi LSTM is to aggregate the information in the matching vector from multiple perspectives. Here GRU can be used as a module in the Bi LSTM as it is faster to train and easier to modify as compared to LSTM.

IMAGE PROCESSING:

1. We can use a variational autoencoder to find similarity between two grayscale images. We should first train the autoencoder to reproduce the input image. Once this is done the latent space representation obtained by encoding the input acts like a feature vector for the input. So, to measure similarity between two images we can measure the distance between their corresponding latent space representation.

2. I modified the code in ex3 to obtain a variational autoencoder trained on the MNIST dataset. code

Libraries included:

```python
import tensorflow as tf
from tensorflow import keras

# Helper libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import scipy.spatial.distance as dist
import scipy.cluster.hierarchy as heir
from PIL import Image
```

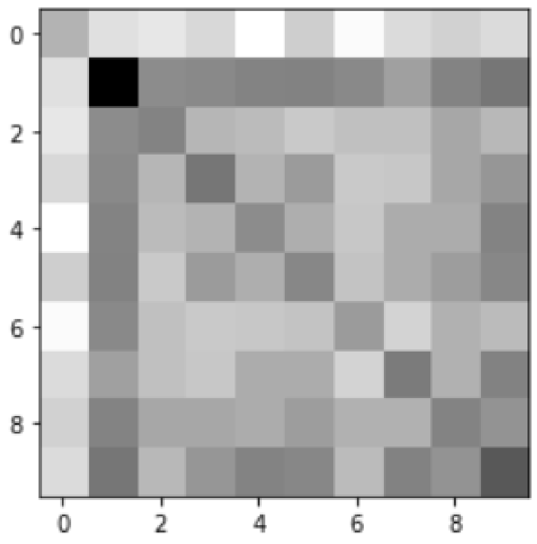
Here is the code I have added:

```
test_images_encoded = encoder.predict(test_images)
test_labels = test_labels.reshape((10000,1))
test = np.hstack((test_images_encoded, test_labels))
testd = pd.DataFrame(test)
digit = []
for i in range(10):
  testnp = testd[testd[16] == i].values
  digit.append(testnp[:, :-1])
  print(digit[i].shape)
spread = np.zeros((10, 10))
for i in range(10):
  for j in range(10):
    if i!=j:
      temp = dist.cdist(digit[i], digit[j])
    else:
      temp = dist.pdist(digit[i])
    spread[i][j] = np.sum(temp) / np.size(temp)
spread *= 255.0/spread.max()
spreadd = pd.DataFrame(spread)
print(spreadd)
plt.imshow(spread, cmap="gray")
plt.show()
```

Here are the results:

```
     0    1    2    3    4    5    6    7    8    9
0  224  242  245  239  255  235  253  240  236  240
1  242  151  208  207  205  204  207  216  205  199
2  245  208  205  225  227  233  229  229  219  226
3  239  207  225  199  224  214  233  232  219  212
4  255  205  227  224  208  222  232  221  221  205
5  235  204  233  214  222  206  231  221  215  206
6  253  207  229  233  232  231  214  237  223  227
7  240  216  229  232  221  221  237  201  223  204
8  236  205  219  219  221  215  223  223  205  211
9  240  199  226  212  205  206  227  204  211  187
```

This matrix can be visualised as an image:



Some observations:
-Diagonal element is smaller than all other elements in the corresponding row or column
-Images of the digit 1 are quite similar to the images of other digits and hence appears darker compared to other cells
-Images of digit 4 and images of digit 6 have very less similarity with images of digit 0

3.

Example number	Time taken for training with GPU	Time taken for training without GPU
3	1min 19s	12min 30s
4	3 min 8s	41min 45s
6	2min 21s	35 min 37s

There was no training involved inEx5.
As we can see here, training without GPU takes an order of magnitude greater time as compared to training with a GPU. This is expected as our models use image processing which the GPU is better at as compared to the CPU.