The BRITISH UNIVERSITY IN EGYPT

WWC: World Wonders Classification

Report

ID	Student Name	E-mail
196280	Ashraf Adel	ashraf196280@bue.edu.eg
194233	Farah Aymen	farah194233@bue.edu.eg
192895	Hanin Monir Ismail	hanin192895@bue.edu.eg

Introduction

The proposed problem is an image classification problem where we want to differentiate an image into one of certain world landmarks (such as Burj Khalifa, Great Wall of China, etc.) and can be found here. The artificial neural network models that will be used to solve this problem are the multilayer perceptron model (MLP), the deep belief network (DBN), and the Hopfield neural network (HNN).

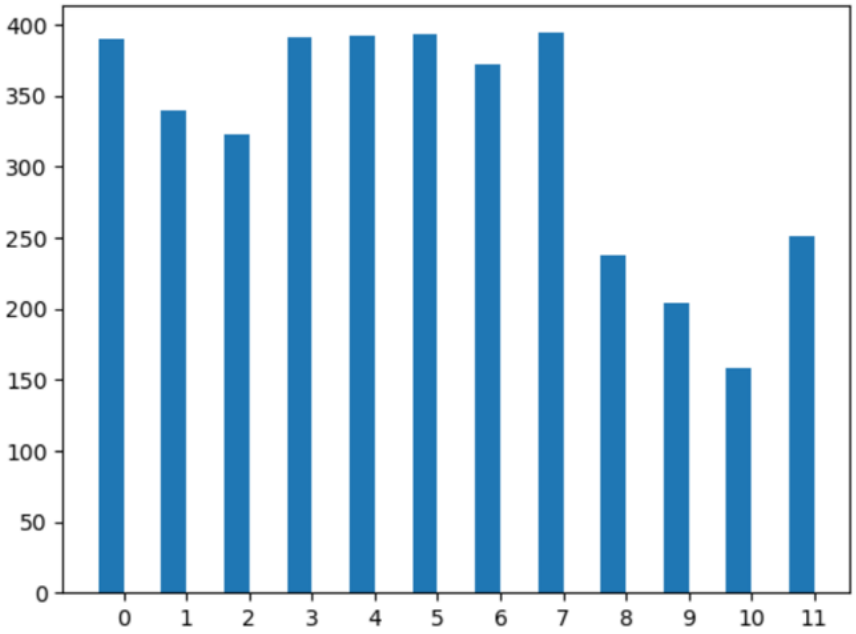
The reason for choosing these models is because they all have image classification as a common application and have been proven to obtain good results.

Dataset Description

The chosen dataset is a dataset of images of monuments of the world taken from google images; it contains 3846 images [1]. The dataset has 12 classes with an imbalance number of images for each class [1]. These classes are:

- Venezuela Angel Falls
- Taj Mahal
- Stonehenge
- Statue of Liberty
- Chichen Itza
- Christ the Redeemer
- Pyramids of Giza
- Eiffel Tower
- Great Wall of China
- Burj Khalifa
- Roman Colosseum
- Machu Picchu

Figure 1 plots the distribution of the images in the dataset over all of the classes, the classes are represented with their index number starting from 0. The key maps the index number to the class name.




```
{'burj_khalifa': 0,
 'chichen_itza': 1,
 'christ_the_reedemer': 2,
 'eiffel_tower': 3,
 'great_wall_of_china': 4,
 'machu_pichu': 5,
 'pyramids_of_giza': 6,
 'roman_colosseum': 7,
 'statue_of_liberty': 8,
 'stonehenge': 9,
 'taj_mahal': 10,
 'venezuela_angel_falls': 11}
```

Key

Figure 1: Distribution of images over classes

Dataset Pre-Processing

The main pre-processing approaches used were as follows:

- Choosing the classes with balanced distribution

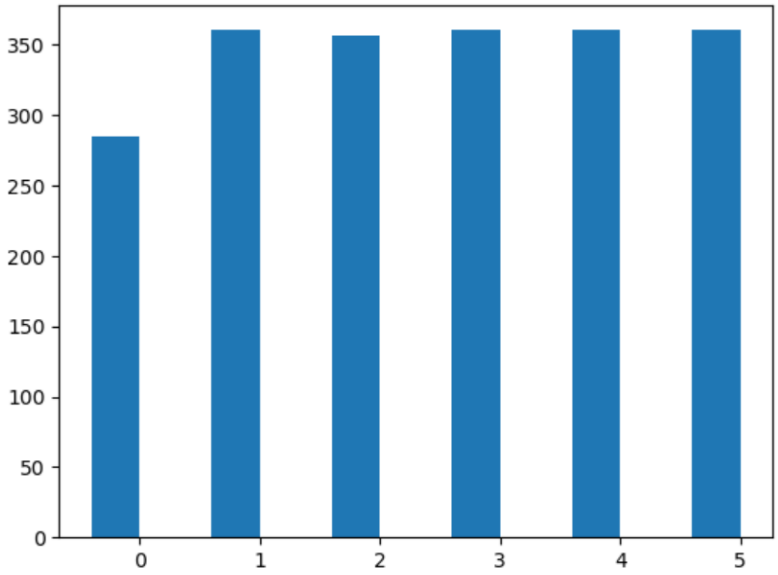


Figure 2: Balanced classes

From the bar chart the chosen balanced classes are follows:

1. Burj Khalifa
2. Eiffel Tower
3. Great wall of China
4. Machu Pichu
5. Pyramids
6. Roman Colosseum

- Removing noisy images that are irrelevant to the data

After the removal of noisy images, the classes had to be filtered based on their distribution.

Figure 3 illustrates the new distribution of the images on classes after filtering.

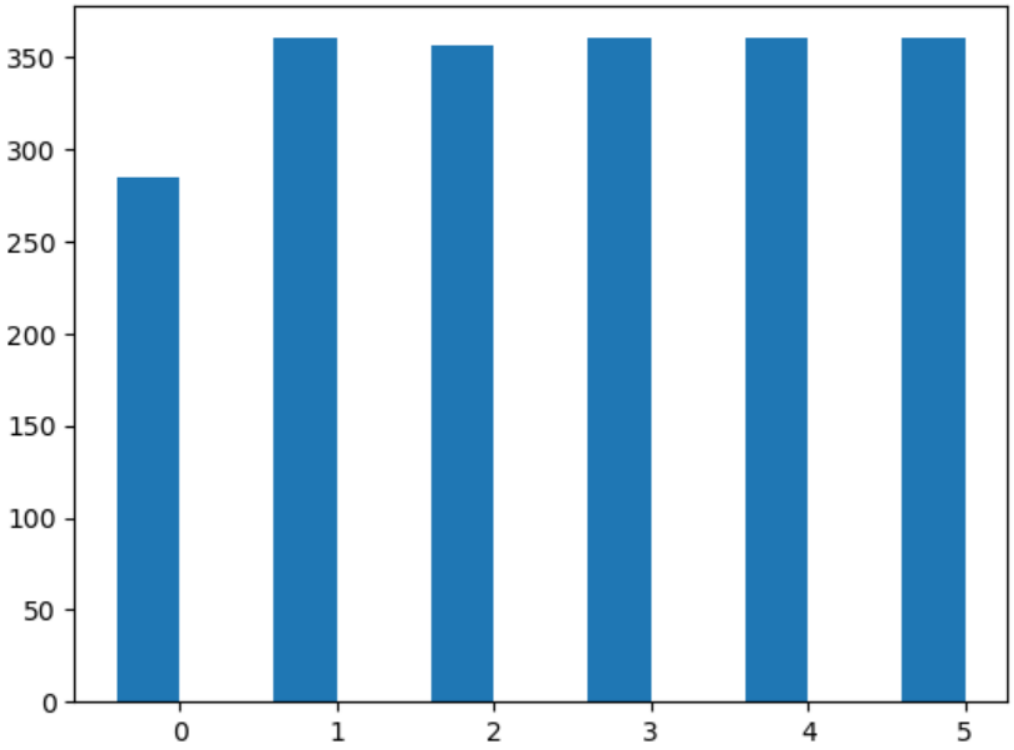


Figure 3: Redistribution of classes after filtering

Based on the redistribution of the images the final chosen classes are:

- Eiffel Tower
- Great wall of China
- Machu Pichu
- Pyramids of Giza
- Roman Colosseum

The total number of images was 1796 images over 5 classes.

- Then the data was loaded into pickle files and stored in different dimension 128x128 and 64x64
- Feature selection was performed using PCA on the images and again, image batches were stored in pickles files for 2 dimensions 64x64 and 128x128.
- Then to avoid underfitting, augmentation was performed.

Models Used

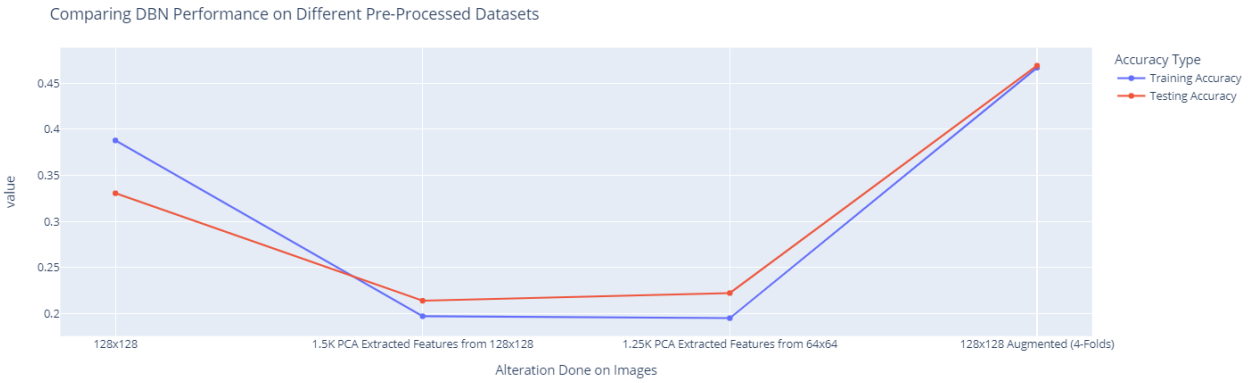
DBN

- 8 DBN models were developed and trained on different subsets of the data.
- these subsets included:
    - rescaled images
    - features extracted from images
        - 128x128
        - 64x64
    - augmented images
    - Model Hypertuning
        - changing learning rate
        - changing NN learning rate
        - changing number of hidden layers
        - changing the number of hidden neurons

Here's a summary of a DBN trained with these hyper-parameters:

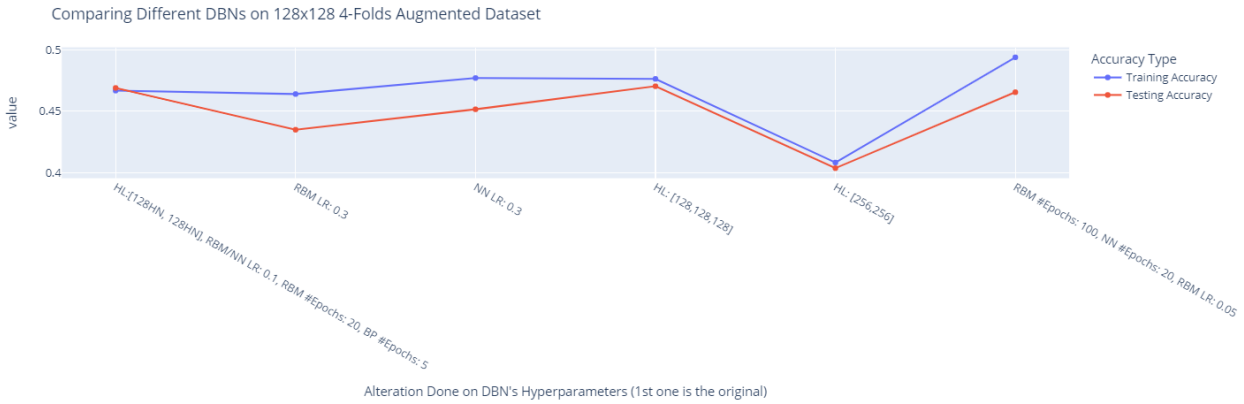
```
(hidden_layers_structure=[128, 128],
 learning_rate_rbm=0.1,
 learning_rate=0.1,
 n_epochs_rbm=20,
 n_iter_backprop=5, # loss was found to be stagnating after this value
 batch_size=32,
 activation_function='sigmoid',
 dropout_p=0.1) # low drop-out value, as model is underfitting
```

On the different pre-processed datasets:



Which clearly shows that the augmentation yields the best results for a cleaner dataset.

Moreover, here's the summary of the different hyper-tuned DBN models on the augmented dataset:



Such that:

- LR = Learning Rate
- HL = Number of Hidden Layers (RBM stacks), where the values inside the square brackets are the number of hidden neurons
- Also, if certain hyper-parameters are not mentioned in the x-axis, it means that they are the same as the first x-axis tick (i.e., same as the original hyper-parameters used in the vanilla DBN previously used)

The line plot above shows that these hyper-parameters obtained the best result:

- RBM LR = 0.05
- NN LR = 0.1
- HL = [128, 128]
- RBM #Epochs = 100
- NN #Epochs = 20

MLP

The aim is to try multiple variations of the MLP network and evaluate their performance in classifying the wonders of the world images.

MLP models on 128x128 images:

- First MLP model:
    - Model Hyperparameters:
    - Input layers: 1
    - Hidden layer: 2, 50 neurons per each
    - Activation function in the hidden layer: "relu"
    - output activation function: "sigmoid"
    - optimizer: "Adam"
    - batch size= 32
    - epochs= 100
    - Model Performance:
        - Accuracy: 0.2026
        - Loss: 0.155

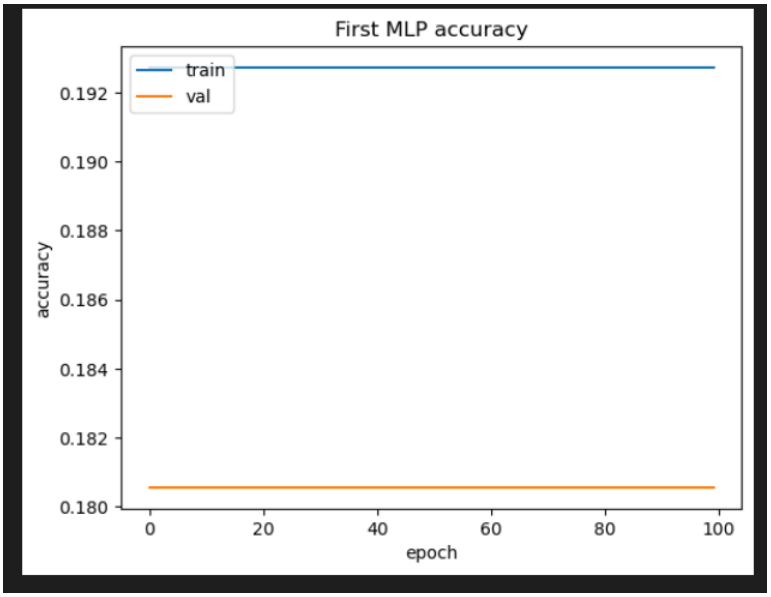
- Second MLP
    - Model Hyperparameters:
    - Input layers: 1
    - Hidden layer: 3, 100 neurons
    - Activation function in the hidden layer: "relu"
    - output activation function: "sigmoid"
    - optimizer: "Adam"
    - batch size= 32
    - epochs= 100
- Third MLP
    - Model Hyperparameters:
    - Input layers: 1
    - Hidden layer: 3, 150 neurons
    - Activation function in the hidden layer: "relu"
    - output activation function: "sigmoid"
    - optimizer: "Momentum"
    - batch size= 32
    - epochs= 100

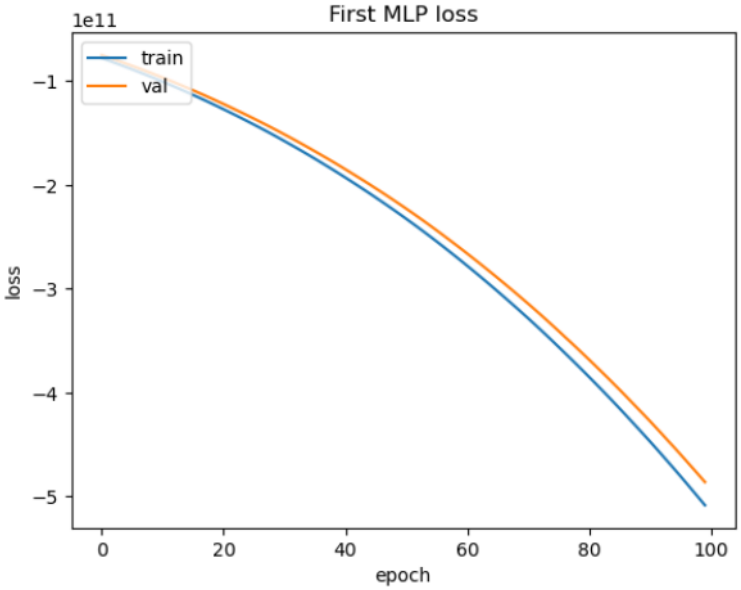
MLP models on 64x64 images:

- First MLP

- Model Hyperparameters:
- Input layers: 1
- Hidden layer: 2, 50 neurons
- Activation function in the hidden layer: "relu"
- output activation function: "sigmoid"
- optimizer: "Adam"
- batch size= 32
- epochs= 100

All the MLP models had almost the same accuracy/loss scores which were:



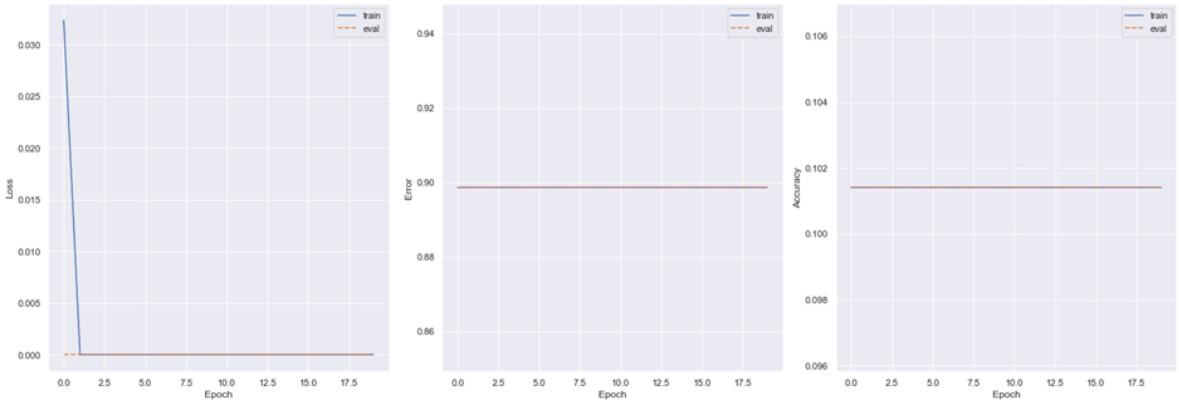


HF

According to the paper Hopfield is all you need, they claimed that they created new continuous Hopfield networks that should have higher capacity of storage and better performance. The paper claimed that they created layers that should be up for testing, but the file was encoded differently, and it was unable to be decoded. They uploaded another file where they tried MNIST on their models, along with it the file I have written trying the model on the world wonders dataset.

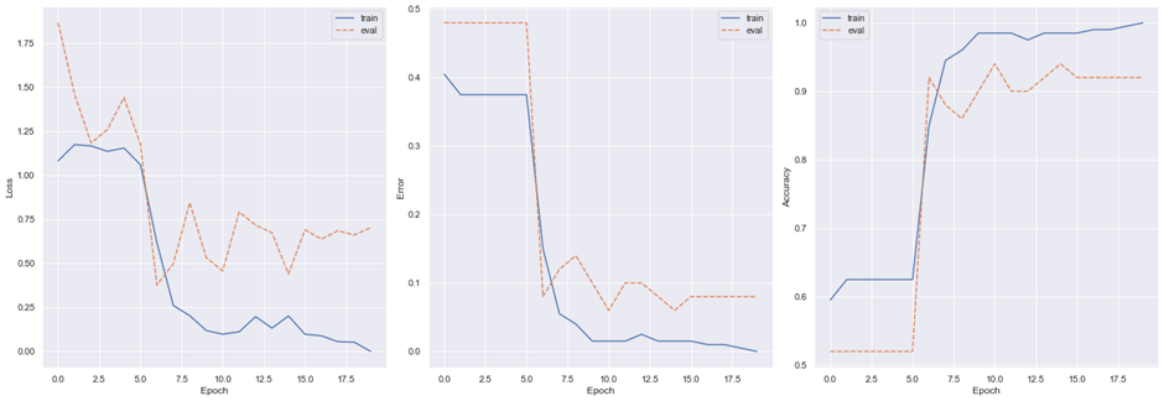
The results were not the best as the model did not accept any images higher than a resolution od 28x28 which why can be a main factor for the low results.

Note that since the network is unsupervised, I unlabelled the data by putting all the images in one file.



As seen above the results were low and it did not learn.

Regarding the MNIST dataset, it was a dataset were the MNIST numbers were loaded into bags and fed to the model.



As seen above, the accuracy reached was almost 100% but the MNIST dataset is a trivial dataset

Comparing The Models

The three models did not perform well on the dataset. The reason behind it might be that the dataset wasn't enough for training and each model had its drawbacks. For example, the Hopfield could not take capacity of better resolution for the images while the MLP is trivial on detecting the classes.

Moreover, these are accuracy results:

- DBN: 0.33 accuracy (on the original dataset)
- MLP: 0.18 accuracy (on the original dataset)
- HF: 0.1 accuracy (on the original dataset)

Which shows that DBN had the overall "best" performance out of the three models.