**A Comparative Study between Multilayer Perceptron and Support Vector Machines in Image Classification**

Achraf Abed

Achraf.abed@city.ac.uk

**Abstract**

This paper will seek to provide an impartial comparison between two models trained to predict an MRI scan diagnosis by recognizing patterns and features in images. The models are a Multilayer Perceptron (MLP) and a Convolutional Neural Network (CNN) with backpropagation. A parameter grid was implemented to iterate over the possible hyperparameters pairings and determine the best ones by calculating validation accuracy. The final best obtained models are tested on the unseen data and compared using confusion matrices, seeing that they are most suitable in this context.

1. **Introduction**

AI (Artificial Intelligence) models have become one of the most sought-after tools in the medicine domain to assist with analyzing and predicting scan results. Mainly, because they're time consuming and some of them have recurring patterns and it would be more efficient to outsource that task to a trained AI model. For Instance, the disease known as Alzheimer's can be detected based on any changes seen on MRI scans of the brain. Moreover, AI models have proven to be very accurate, resulting in a more satisfied patient's experience and better healthcare [1].

The aim of this paper is to compare and critically evaluate two models developed to interpret an Alzheimer’s MRI scan and classify it accordingly into one of the four categories based on image data. Those two models being: a Multilayer Perceptron, and a Convolutional Neural Network with backpropagation. The various compositions of these networks will be discussed in order to establish the accurate and suitable setting for both models.

In the first section, we’ll briefly describe how both models operate and state some of their conveniences and inconveniences. Followed by a short analysis of the chosen dataset, then the next section will cover the methodology pursued when implementing the models. Finally, we discuss the results obtained and conclude with the lessons learned and any missing steps that could be added for further work.

* 1. **Convolutional Neural Networks (CNN)**

CNNs are a type of neural network that is more computer vision oriented, i.e., used to recognize emerging patterns in images or videos by implementing a sequence of filters to the input data. The latter’s purpose is to extract characteristics (such as edges, sharpness, text) also known as features maps, which are then fed to further layers for processing depending on the data’s complexity [2]. This type of network uses another method known as ‘Pooling’, the purpose of which is to reduce the size of those extracted feature maps according to their importance. It also helps train the model much quicker at a less cost [3].

* 1. **Multilayer Perceptron (MLP)**

MLP is another type of neural network, that comprises of node layers interconnected to perform a specific computation on the loaded data. The first layer which consists of the first set of nodes or ‘neurons’, selects an initial set of weights at random after receiving input. An activation function, then establishes the value of the output after these are added together in a weighted sum [4].

Same computation is done on the level of the hidden layers, until we finally reach the output layer, whose number of nodes depend on the task at hand (classification, regression…). By adjusting the weights to reduce the discrepancy between the network's output and the desired output, a process known as backpropagation is used to learn the weights in an MLP. Usually, a gradient descent optimization algorithm is used to carry out this process [1].

Image recognition, speech recognition, and natural language processing are just a few of the many applications where MLPs are an effective tool [4].

1. **Dataset**

The dataset chosen for this purpose is images collected from several websites, hospitals, and public repositories, downloaded from Kaggle datasets. The images are MRI (Magnetic Resonance Imaging) scans that have already been preprocessed and resized to 128\*128 pixels. The task involves classifying the scans into one of the four diagnoses: Mild Demented (896 images), Moderate Demented (64 images), Non-Demented (3200 images) and Very Mild Demented (2240 images).

First looking at the dataset, we see a bit of class distribution imbalance, where the Moderate Demented class is the minority with only 64 images, and a significant dominance by the non-demented class. Sampling techniques such as oversampling, under sampling or SMOTE have been avoided in this study in order to see the impact this anomaly would have on our models.

* 1. **Initial Data Analysis**

As stated before, the images chosen have been preprocessed already and resized to 128\*128, which will result in our model training time to be significantly shorter than bigger resolutions without having an impact on its performance. Moreover, these images are in grayscale format (black and white), that is one of the image augmentation techniques that contribute to improving the model’s performance [5]. In a way, that was perceived as compensation for the existing class imbalance and promising results were expected.

Below are a couple of images from each class, to give a general idea of what our data looks like.

**A close-up of the moon

Description automatically generated with low confidenceA close-up of a human brain

Description automatically generated with low confidenceA close-up of a human brain

Description automatically generated with medium confidenceA close-up of a human skull

Description automatically generated with low confidenceFigure 1:** Example images of each diagnosis

So according to the images, the brain regions that must focused on are the middle areas where we see a growth of a scale as we go from the non-demented to the moderate-demented diagnosis.

1. **Methods**

In this section, details of dataset splitting and model training, validating and testing will be covered, along with a discussion of how the models were implemented and the optimal hyperparameters were chosen.

* 1. **Methodology**

We begin by splitting the downloaded data directory into training, validating and testing portions: 80%, 10%, 10%, respectively. The first two subsets are used for model selection, and the rest is left as a holdout for the purpose of comparison.

In order to address the model selection inquiry, an iteration over a defined parameter grid was implemented for both models and training using DataLoader with the attribute shuffle set to true, so cross-validation wasn’t needed. Afterwards, validating the trained model with each pair of hyperparameters and calculating the validation accuracy. However, a k-fold stratified cross-validation could be an addition that might improve model performance due to the existing class dominance mentioned earlier, as it is a variation of k-fold cross-validation, each fold is guaranteed to contain roughly the same number of samples from each class as the original data set [6].

As for the remaining task, which is to compare the models, we decided to retrain the optimal obtained networks from the previous step on a combined data loader that consists of both the train loader and the validation loader and test the final model on the held-out test set. Unlike CNN, the MLP has an addition during the model selection phase which is early stopping that’s conditioned on the validation accuracy’s improvement. A fixed number of epochs was set for both models to ensure a fair comparison.

* 1. **Network Architecture for MLP**

Our multi-layer perceptron (MLP) has an input layer with input size of 128\*128 neurons (image dimensions), three hidden layers with the size of the hidden neurons left undetermined to run the parameter search, and an output layer with 4 neurons. The activation function used in all the hidden layers is ReLU (rectified linear unit), which helps to introduce non-linearity to the network. The output layer uses the SoftMax activation function, which normalizes the output values to probabilities that add up to 1 and can be used to make predictions and evaluate the model [1].

In terms of the forward pass, the input is first flattened into a 1D vector using view. Then, the hidden layers are applied in succession using the ReLU activation function, followed by the output layer which applies the SoftMax function.

The Backpropagation algorithm was used to iteratively adjust the weights of the network to reduce the error between the predicted output and the desired output using a training set of inputs and desired outputs. Despite it being a powerful technique, it can suffer from other problems such as vanishing gradients and overfitting, which is why various other methods, including weight initialization, batch normalization, and dropout, have been developed to enhance the performance and stability of the algorithm [1].

* 1. **Network Architecture for CNN**

As for our CNN's architecture, it consists of two convolutional layers with 16 and 32 filters respectively to perform feature extraction of affected regions to distinguish the features between the affected regions [7], both with a kernel size left undetermined and passed later as a parameter with a stride of 1, followed by max pooling layers with a kernel size of 2 and a stride of 2. The output of the second convolutional layer is flattened and passed through three fully connected layers with 120, 84, and 4 neurons respectively. ReLU activation function is applied after each layer except the output layer, which has no activation function.

The \_get\_conv\_output\_size method is used to calculate the output size of the convolutional layers based on the input size and kernel size, assuming padding is same, stride is 1, and only MaxPool2d is used for pooling.

|  |  |  |  |
| --- | --- | --- | --- |
| CNN | | | |
| Learning rate | Kernel size | Momentum | Validation acc |
| 0.001 | 5 | 0.9 | 61% |
| 0.001 | 3 | 0.9 | 57% |
| 0.003 | 5 | 0.9 | 89% |
| 0.003 | 3 | 0.9 | 77% |
| 0.01 | 5 | 0.9 | 94% |
| 0.01 | 3 | 0.9 | 96% |
| 0.1 | 5 | 0.9 | 50% |
| 0.1 | 3 | 0.9 | 69% |

1. **Results and Critical Evaluation Table 1:** CNN Parameter Search
   1. **Model Selection**

Table 1 on the right shows the results obtained from the parameter grid search conducted. First, we notice a huge variance in terms of validation accuracy just by changing the learning rate and not so much when swapping the kernel size. Where an accuracy of 96% was achieved, which also reflects the right number of filters chosen regarding the convolution layers, in our case was 16,32 for the first conv1 and second conv2 respectively. Furthermore, most sections of the MRI scan are black (at the edges) and given that the images are sized 128\*128 which is a small resolution, has made the model’s task easier at capturing the necessary feature maps for predictions [7].

|  |  |  |  |
| --- | --- | --- | --- |
| **Table 2:** MLP | | | |
| Learning rate | Hidden neurons | Momentum | Validation acc |
| 0.003 | [64,64,64] | 0.3 | 59.37% |
| 0.003 | [64,64,64] | 0.9 | 62.51% |
| 0.003 | [128,128,128] | 0.3 | 50.39% |
| 0.003 | [128,128,128] | 0.9 | 60.94% |
| 0.003 | [256,256,256] | 0.3 | 50.39% |
| 0.003 | [256,256,256] | 0.9 | 61.41% |
| 0.01 | [64,64,64] | 0.3 | 62.67% |
| 0.01 | [64,64,64] | 0.9 | 60.47% |
| 0.01 | [128,128,128] | 0.3 | 67.87% |
| 0.01 | [128,128,128] | 0.9 | 65.35% |
| 0.01 | [256,256,256] | 0.3 | 65.19% |
| 0.01 | [256,256,256] | 0.9 | 50.39% |
| 0.03 | [64,64,64] | 0.3 | 56.22% |
| 0.03 | [64,64,64] | 0.9 | 50.39% |
| 0.03 | [128,128,128] | 0.3 | 60.78% |
| 0.03 | [128,128,128] | 0.9 | 50.39% |
| 0.03 | [256,256,256] | 0.3 | 59.21% |
| 0.03 | [256,256,256] | 0.9 | 50.39% |

As for the MLP results, shown in Table 2, a much poorer performance is observed, where validation accuracies drop down to 50% in some pairings, the highest being nearly 68%. Our MLP had some different parameter sets where we altered some of the values for the learning rates and added a different value for the momentum. The latter showed a bit of an impact in some cases where it increased the accuracy by more than 10%, however the optimal learning rate remains the same as the one found in CNN.

* 1. **Algorithm Comparison**

Figure 1 below represents the confusion matrices obtained after running the best fully trained models on unseen data. Where we ended up with 99% accuracy from the CNN model, and just 65% for the MLP.

Chart

Description automatically generatedChart

Description automatically generatedAs can be seen in the confusion matrix for the MLP, commonly mis predicting the diagnoses, for instance we have a total of 150 misclassification of the class very mild demented and that could cause issues in that context, informing a patient of wrong diagnosis. We also have some cases where our model predicted very mild demented when the actual diagnosis was non demented, which again could spawn fear in the patient’s head, and unrecommended to have a model deliver. This poor performance is a clue that MLPs can handle a variety of classification tasks, but they might not be the best option for tasks involving image classification. This is because the spatial relationship between pixels in an image, which is crucial for image classification, is not taken into account by MLPs and it is important to carefully select and apply preprocessing techniques, depending on the task at hand [8]. Whereas our CNN delivered outstanding results, rarely misclassifying the diagnoses, around 10 mis predicted cases. Thanks to the pooling technique it possesses, which helps it reduce the size of the feature maps and by extension the computational complexity of the network [4].

**Figure 1:** Confusion matrices for both models

1. **Conclusion**

Effective and accurate diagnosis of Alzheimer’s is an important step to initiate effective treatment. In particular, early investigation of Alzheimer’s is expected to play an immense role in the development of cures and ultimately for the remarkable recovery of the patient. In this study, MLP and CNN predict the 4 Alzheimer’s diagnoses from MRI images. CNN provides significantly better accuracy than MLP in image classification.

The main lesson learned is that in order to proceed with MLP for the purpose of image classification, preprocessing techniques must be pursued whether that’s cropping, resizing, **data augmentation** or feature extraction. These methods will improve the performance obtained in this study drastically [8].

Moreover, the low number of epochs might have had a factor too, and the reason behind setting a low epochs number was because of the immense feature dimensionality that would have expanded the training time massively.

Maybe for future reference, it would be also ideal to have more parameters in the grid search step, and a few changes to the network’s architecture might improve the network, such as adding more hidden layers.

1. **References**

[1] Bae, J.B. *et al.* (2020) ‘Identification of Alzheimer’s disease using a convolutional neural network model based on T1-weighted magnetic resonance imaging’, *Scientific Reports*, 10(1), p. 22252. Available at: https://doi.org/10.1038/s41598-020-79243-9.

[2] Kamal, Md.S. *et al.* (2021) ‘Alzheimer’s Patient Analysis Using Image and Gene Expression Data and Explainable-AI to Present Associated Genes’, *IEEE Transactions on Instrumentation and Measurement*, 70, pp. 1–7. Available at: https://doi.org/10.1109/TIM.2021.3107056.

[3] Wu, J. (no date) ‘Introduction to Convolutional Neural Networks’.

[4] Bento, C. (2021) *Multilayer Perceptron Explained with a Real-Life Example and Python Code: Sentiment Analysis*, *Medium*. Available at: https://towardsdatascience.com/multilayer-perceptron-explained-with-a-real-life-example-and-python-code-sentiment-analysis-cb408ee93141 (Accessed: 13 April 2023).

[5] Brownlee, J. (2016) ‘Crash Course on Multi-Layer Perceptron Neural Networks’, *MachineLearningMastery.com*, 16 May. Available at: https://machinelearningmastery.com/neural-networks-crash-course/ (Accessed: 13 April 2023).

[6] JAN 26, J.N. and Read, 2020 8 Min (2020) *Why should I do pre-processing and augmentation on my computer vision datasets?*, *Roboflow Blog*. Available at: https://blog.roboflow.com/why-preprocess-augment/ (Accessed: 13 April 2023).

[7] Khalid, I.A. (2021) *Multilayer Perceptron for Image Classification*, *Medium*. Available at: https://towardsdatascience.com/multilayer-perceptron-for-image-classification-5c1f25738935 (Accessed: 14 April 2023).

[8] Kumar, N. (2019) *Visualizing Convolution Neural Networks using Pytorch*, *Medium*. Available at: https://towardsdatascience.com/visualizing-convolution-neural-networks-using-pytorch-3dfa8443e74e (Accessed: 13 April 2023).

Appendix 1 – Glossary

**Data Augmentation:** it’s a technique used to artificially expand the size of a training dataset by applying various transformations to the existing data, such as rotating, scaling, flipping, cropping, adding noise, etc.

Appendix 2 – Implementation details

Initially, I selected SVMs and CNNs as models to compare. I found the OneVsRest classifier to be suitable for my task (multi-class classification), that said, I ran into an obstacle when I was running a grid search for the model where the cell on jupyter notebooks took about 4+ hours running and still hasn’t finished, which just slowed me down and made me worry about being able to deliver the coursework on time. The code will still be there in the notebook but I wont be including it in the print-out version