Comparison of neural network and supervised learning for image classification

Nicholas Barros

11/17/2023

Abstract

In this paper, we compared two different methods of image classification a K nearest neighbors (KNN) and a Multi-Layer Perceptron (MLP) algorithms. On two different testing groups using images of "Pokémon" from the popular series of the same name (all rights reserved to the "Pokémon Company" and "Nintendo" who hold the rights to all images used). The experiment that we ran on the two testing groups using the KNN and MLP algorithms to find the algorithms ability to correctly identify an images class (accuracy). The experiment found that both of the algorithms preformed comparable to each other. With the KNN algorithm on average having an accuracy of 88% (average of both test groups) for both of the testing groups. And the MLP algorithm preformed with an accuracy of 88% for both of the testing groups.

Introduction

Background

Neural networks are a type of deep learning artificial intelligence algorithm that consists of three distinct layers of node: an input, hidden, and output layer. The input layer is responsible for inputting data where each node in the input layer is a different variable. The hidden layer is where different computations take place according to the input data and their associated weights.

A node's weight is determined by training the model and determines what input or nodes have a priority over each other. The output layer represents the possible outcomes of the algorithm. Neural networks are regularly used for image recognition and processing. “Image recognition is the process of identifying an object or feature of an image or video” [1]. Image recognition models utilize groups of training/ test images. These images are broken up into the programmers desired categories, for example giving the model pictures of cats, birds, and fish. Using the training images, the programmer trains the image recognition model that evaluates the testing images and identifies key features that it can use to identify the image.

The K Nearest Neighbor algorithm is used for classification of data points and is a type of supervised learning structure. The algorithm works as such: the training data is graphed or grouped together and labeled by their respected class. When a new data point is introduced, the algorithm uses an equation to determine the distance between the points. The Ellcian Distance equation is one of the most popular methods to find the distance. The existing sample with the lowest distance from the new point is designated as the nearest neighbor. The K in KNN stands for how many neighbors are considered when performing the classification function. There are two different classification functions that can be used, a “majority vote” or “weighted vote”. The majority vote takes the K closest neighbors and classifies a new point according to the class that is seen the most in the K neighbors. The Weighted vote considers an assigned weight to a data point, the higher the weight the more “important” a particular data point is. So instead of the algorithm assigning a new point its class based on the number of classes that appear as the K nearest to the new point it instead determines the class based on each class's total weight [2].

Significance

The significance of this paper and the research that accompanies it, is to provide an insight to other researchers who wish to conduct classification of grayscale images on which image classification algorithm may be best to use. I.E is a convolutional neural network image classification algorithm more effective in correctly classifying grayscale images than that of a traditional method such as the KNN algorithm. Learning which of the image recognition models performs the classification function more accurately would be an important tool. Image classification is already an important topic especially in the medical field as seen by the work of Mohsen, El-Dahsham, El-Horbaty, and Salem, who used deep learning neural networks to classify brain tumors found in magnetic resonance imaging (MRI) [3]. The result of their research resulted with about a 90% accuracy of classification while using a neural network-based image classification model.

Related work

Image recognition recently in the past few years has taken a key role in automating the classification of data. One of these examples being in 2018 where researchers combined autonomous moving underwater drones with image recognition to identify fish populations to determine the health of underwater ecosystems [5]. Another group of researchers utilized image recognition to process and classify MRI images of the brain to detect tumors [3]. The results of which indicated that after training and testing a neural network-based image recognition model to correctly classify images on average about 90% of the time [3]. Image recognition is just one example of neural networks' applications; they can also be used in image reconstruction. In one

of the related work articles researchers were able to use neural networks to apply filters to a low-resolution image taken with a regular camera and augment that image into a high-resolution gray and or color thermal image, negating the need to purchase an expensive camera that naturally produces a thermal image [6].

Introduction

In this paper, we discuss image classification using K Nearest Neighbor (KNN) and Multi-Layer Perceptron (MLP)algorithms. For an experiment we created a dataset of images, that were taken from the "Pokémon Database" [4]. The dataset that we created for this experiment contained a group of four different "Pokémon" from the popular video game series of the same name. The objective for this research was to test the effectiveness of the convolutional neural network in correctly classifying a dataset of images when compared to a traditional supervised learning algorithm like the KNN classifier. The results of the experiment that we ran to test this showed that the KNN classifier was more accurate than that of the multi-layer perceptron neural network.

Materials and Methods

All the programs that were used for this project were written using the "3.8.5" version of Python3. "Jupyter Notebook" was the chosen IDE for the project, and the Sklearn python library was used to create both the KNN and MLP image classifiers. For this experiment, we also utilized 2 different testing groups. Testing group 1 (t1) was comprised of four classes and 181 resized and gray scaled images. Testing group 2 (t2) was comprised of the same four classes and 1,274 resized and gray scaled imaged from testing group 1 as well as augmented version of that images. Descriptions of the augmentations done for testing group 2 are outlined below in step 3. An important thing to note about the data that we decided to use for the experiment's dataset is

that it is only comprised of Pokémon from the first generation of the series. The Pokémon franchise has 8 generations of Pokémon and over 1021 Pokémon in general. Pokémon from the first generation appear in other generations and because of this have more image data associated with it. Because of this we only used generation 1 Pokémon due to the increase in data compared to other generations of Pokémon.

The steps to completing the project experiment are as such (note that steps 1 – 4 can be skipped by downloading both the "t1" and "t2" zipped files from the Github): (1) Assemble and download the image data from the "Pokémon Database" [4] website, for this experiment we chose 4 different Pokémon at random (you can use any four) from generation one. (2) Compile the downloaded images into a set of folders/ groups, each Pokémon should have their own corresponding folder. (3) Use the "image augmentation" program located in the GitHub repository. The program imports images from one of the image folders previously created and resizes the image to an (80,80) pixel image as well as applying a grey scale to the image converting it from a color image to a black and white image the result of which can be seen below in **Figure 1.** For testing group 1 these are the only changes that will be made to each image. For testing group 2 we used the Pillows library in order to augment the images in several ways including; rotating the image in 4 different ways, mirroring the image as well as flipping the image upside-down. The results the image augmentation can be seen in **Figure 2**. Note that the most updated version of the code is set up to augment the original colored imaged and augment them for testing group 2. To set up the code to only apply the gray scale and resize for testing group 1 comment out the calls of the; "image_flip", "image_rotate", and the "image_mir" functions. (4) Using the provided code from the GitHub repository called "Images Rename", change the "root_dir" variable to match the location of one of the testing groups. The code will

rename all the images in that folder to make it easier to work with. Repeat this step with all the other testing groups. (5) After creating both testing group 1 and 2 we can use both the "KNN_code" and "CNN_code" programs. Both programs imports the images from one testing group and converts the images located in the selected testing group folder into an array using the to "NumPy.array" function from the NumPy python library. This array is flattened and assigned a class number. Using this array data, each program feeds the image data to the image classification libraries that is provided by SkLearn. The settings that we changed for the KNN image classification we can change the value of K by changing the value of "n_neighbors" as constants for this experiment, the "test_size" variable seen in the "KNN_code" program was set to 0.33, the "random_state" was set to 99, and the "stratify" was set to 'y'. The settings that were applied to the MLP classifier in the "CNN_code" program are; setting the test train split function to match that of the KNN classifier. As well as setting the hidden layer sizes of the classifier to (75, 20) this size for the hidden layer was selected to represent the results of the experiment based on its higher accuracy when compared to other hidden layer sizes tested as in a size of (5,2) and (20,5). (7) Read and collect the data that is obtained by each classifier for each testing group. For the KNN classifier you will need change the value of "n_neighbors" to represent the number of neighbor's taken into account for determining a class. We chose K equal to 1,4 and 7 for this experiment. The data to be collected for each testing group in each classifier should be the total accuracy of the image classifier to correctly classify the correct Pokémon/ the image's class as outputted by the program this data is represented by the generated confusion matrix in each program.

Source code for this project is available from https://github.com/NickB265/Senior-research-on-neural-networks.

Results

For the KNN classifier we compared the total accuracy for when K was set to 1, 4, and 7. In both testing groups we found that when K is set to the lowest value in this case being 1, it had the highest overall accuracy as seen in **Table 1** and **Table 2**. We can get a better understanding of how the KNN classifier made its predictions and where the accuracy was decreased by looking at the confusion matrix located in **Figure 3** and **Figure 4**. A consistent pattern that appears in the confusion matrixes is the classification of class 3 (Marowack) in that the classifier often miss-classified class 3 as class 0. Another trend in the KNN classifier data was that when the size of K increased the total accuracy decreased. This trend is highlighted in the graph in **Figure 5** whereas K increases the overall accuracy decreases. The lowest accuracy of the classifier being just over 25% when K = 26. The accuracy of the KNN classifier for both testing groups when compared to the results of the MLP classifier is slightly better at correctly identifying images. **Figure 7** and **Figure 8** below shows the confusion matrix of the resulting experiment on the MLP classifier having an accuracy of 86% for testing group 1 and an 89% accuracy for testing group 2.

Figure 1- Comparison of the original color image (Left) from the Pokémon database and the augmented gray scale image (Right) for testing group 1.

Figure 2- Comparison of the original color image (top left) and the other augmented images that were used for testing group 2.

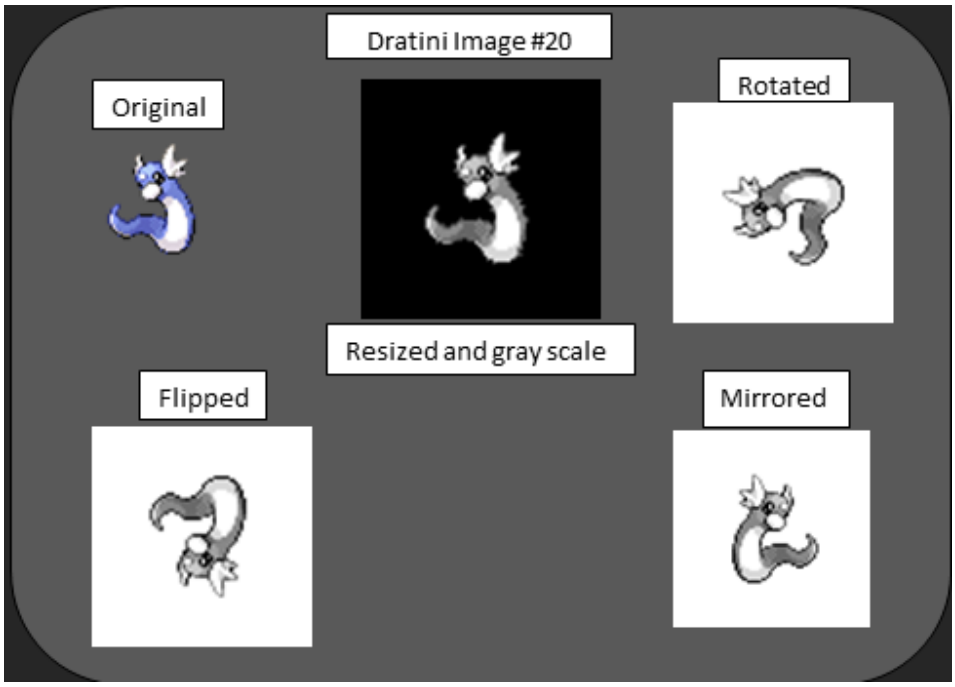

Table 1- shows the accuracy of the KNN classifier for Testing Group 1 when K is equal to 1, 4, and 7.

Testing group 1	K = 1	K = 4	K = 7
Class 0 (Bulbasaur)	92%	92%	92%
Class 1 (Dratini)	78%	50%	50%

Class 2 (Machoke)	100%	66%	73%
Class 3 (Marowak)	73%	42%	21%
Overall Average	85%	60%	55%

Table 2 shows the accuracy of the KNN classifier for Testing Group 2 when K is equal to 1, 4, and 7.

Testing group 2	K = 1	K = 4	K = 7
Class 0 (Bulbasaur)	97%	97%	94%
Class 1 (Dratini)	85%	34%	33%
Class 2 (Machoke)	95%	69%	66%
Class 3 (Marowak)	84%	36%	26%
Overall Average	90%	57%	52%

Figure 3- Confusion matrix of K = 1 to help emphasis how well the KNN classifier correctly or not correctly predicted the class of an image form Testing Group 1.

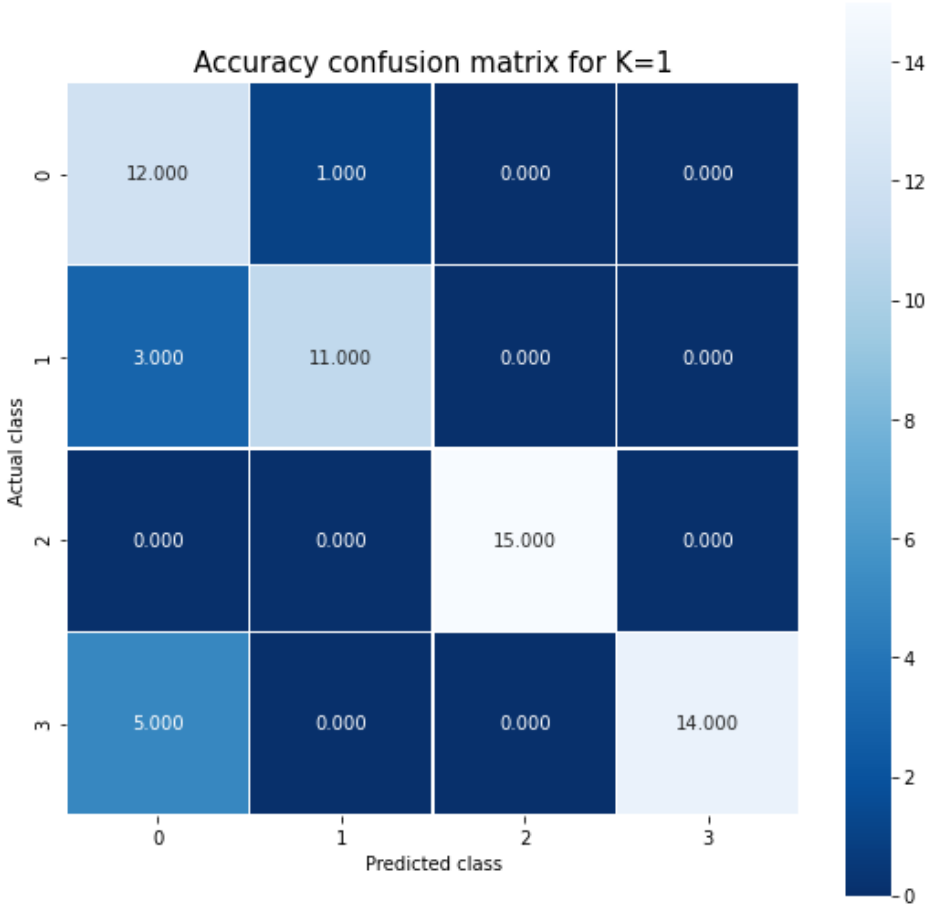

Figure 4-Confusion matrix of K = 1 to help emphasis how well the KNN classifier correctly or not correctly predicted the class of an image from Testing Group 2

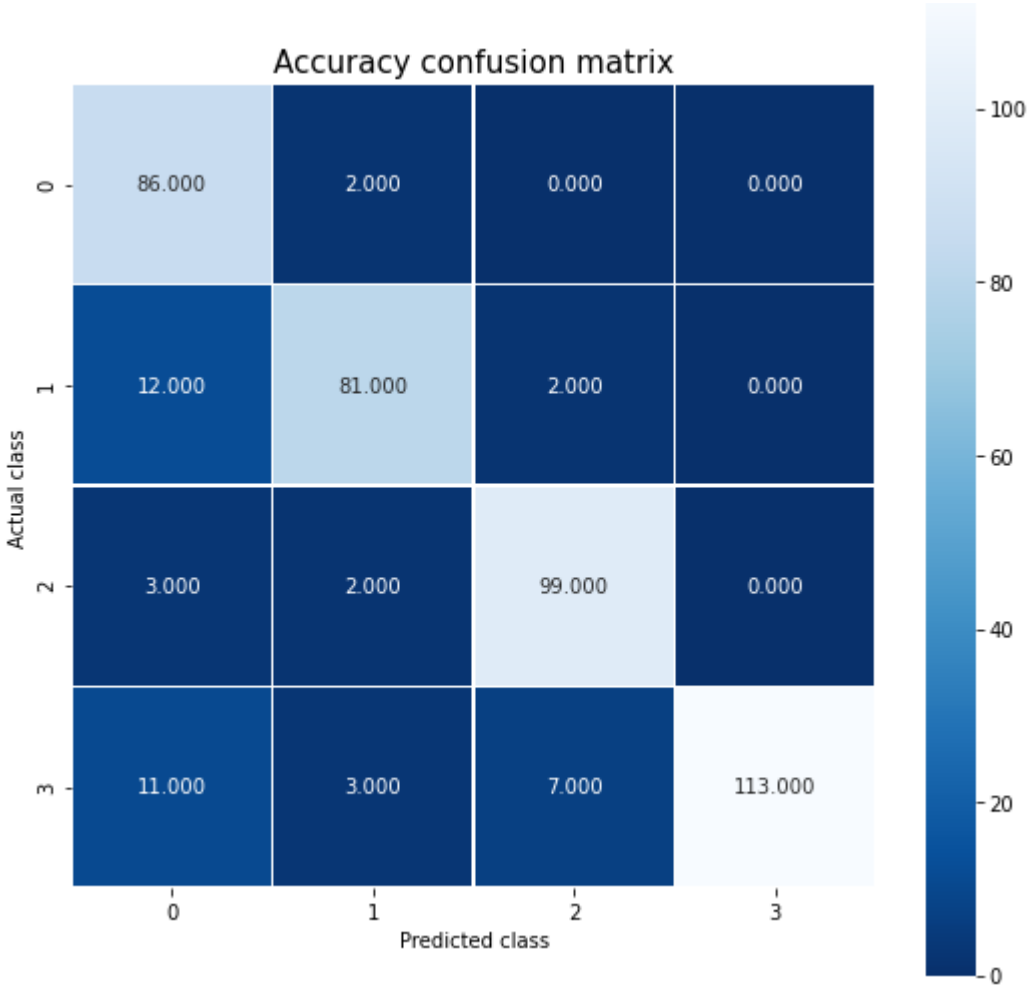
Accuracy confusion matrix
86.000
2.000
0.000
0.000
12.000
81.000
2.000
0.000
3.000
2.000
99.000
0.000
11.000
3.000
7.000
113.000
Actual class
Predicted class
0
1
2
3
100
80
60
40
20
0

Figure 5- Graph of the overall accuracy of the KNN classifier for Testing Group 1 when K is set from 1 through 30

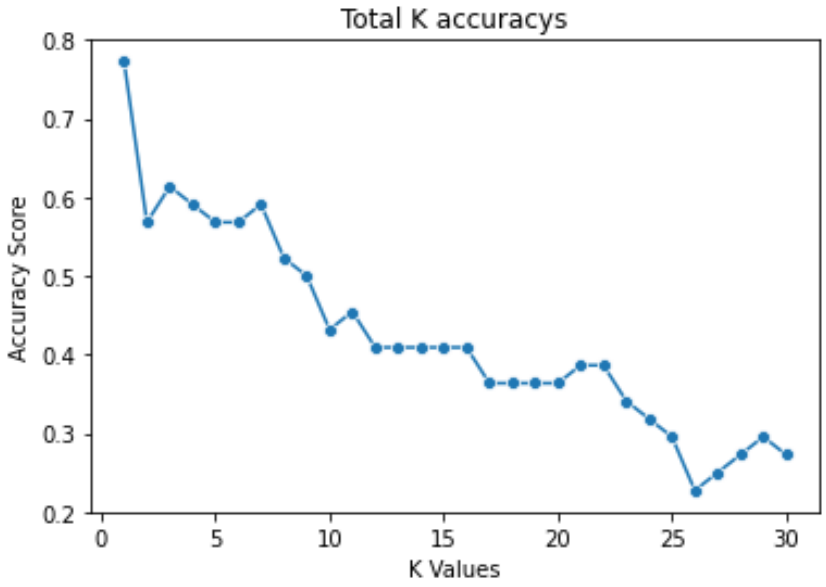

Figure 6- Graph of the overall accuracy of the KNN classifier for Testing Group 2 when K is set from 1 through 9. (This graph was set to 9 unlike figure 4 due to processing restrictions)

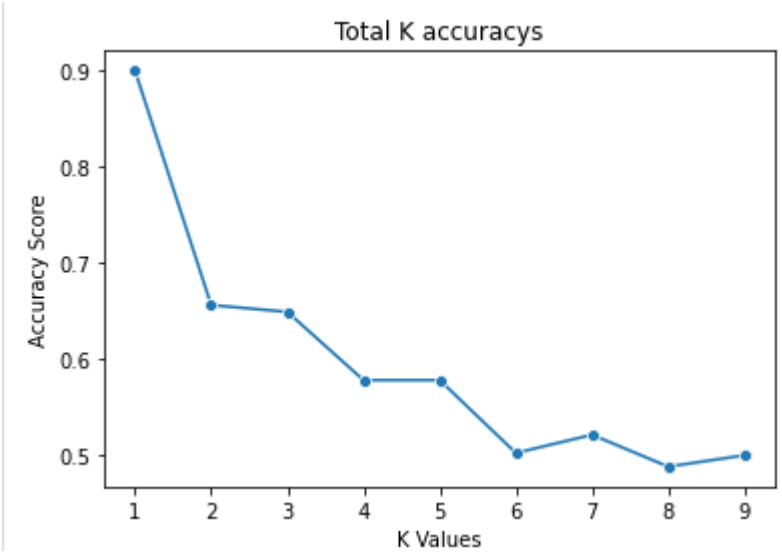

Figure 7- Confusion matrix of the MLP classifier for Testing Group 1.

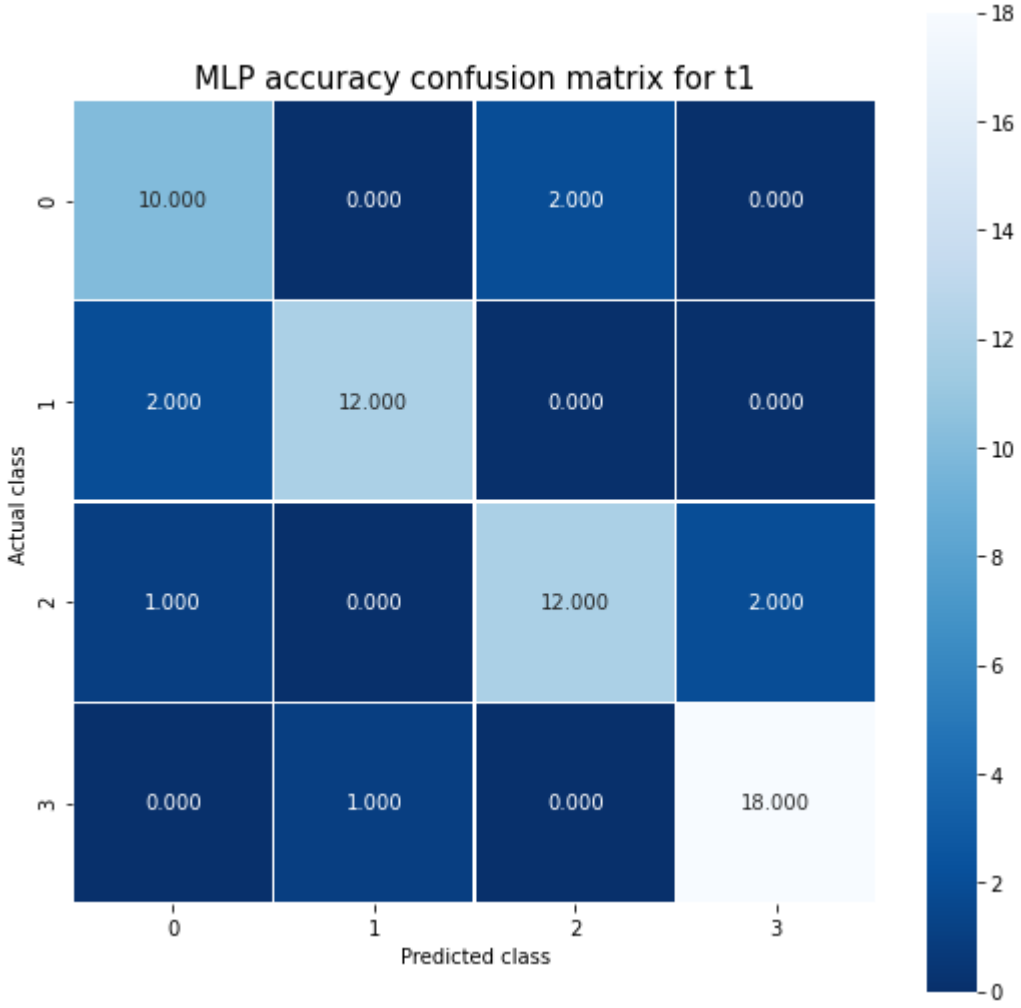

Figure 8- Confusion matrix of the MLP classifier for Testing Group 2.

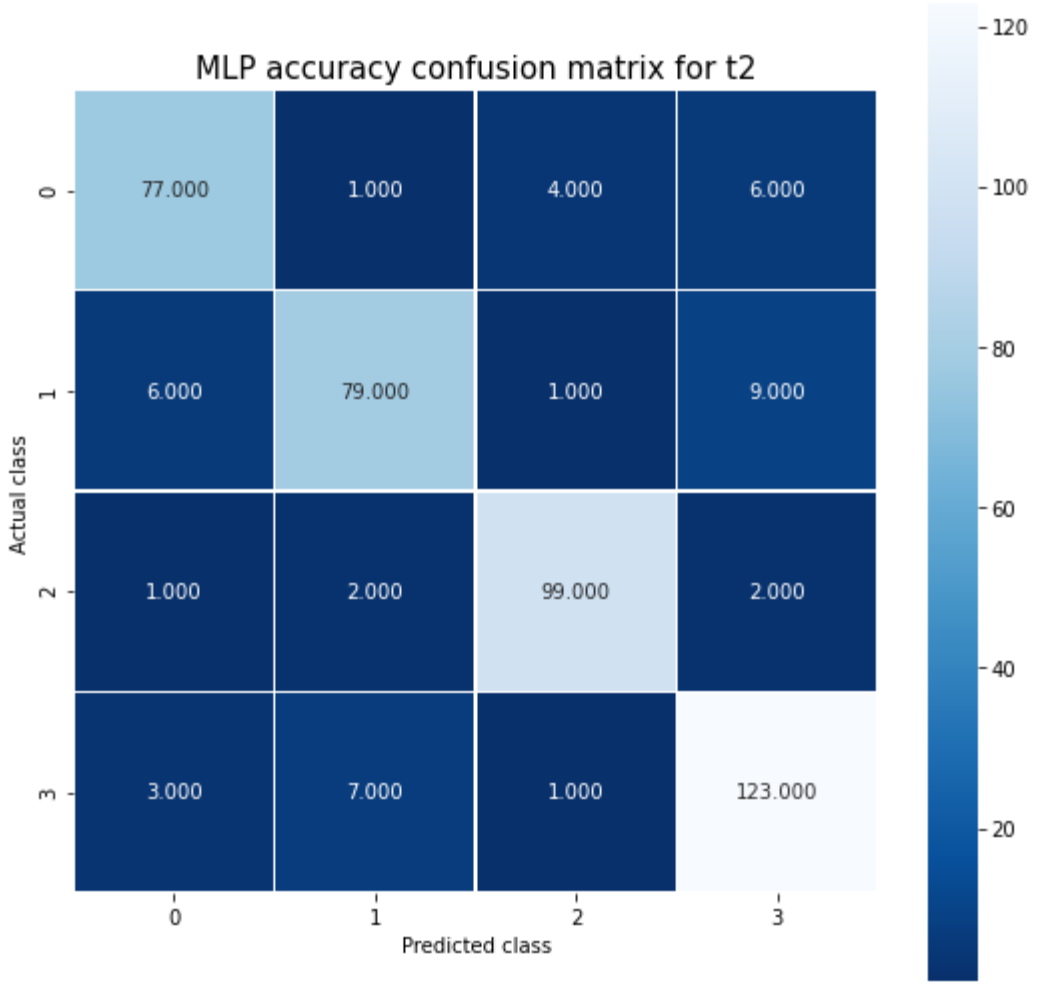

Table 3- This table shows the best result of the KNN table from table 1 and the total accuracy from the CNN classifier for testing group 1.

Testing group 1	Overall accuracy
KNN	85%
MLP	86%

Table 4- This table shows the best result of the KNN table from table 2 and the total accuracy from the CNN classifier for testing group 2.

Testing group 2	Overall accuracy
KNN	90%
MLP	89%

Discussion

Results when compared to related works

When comparing the results of the image classifier to other related work such as the work of Meng, Hirayama, and Oyanagi. Who's work included testing three different convolutional neural networks (CNN) in their ability to correctly identify four different types of fish (accuracy). The experiment that they run showed that the accuracy of all three CNN algorithms to be approximately 85% [5]. Although a CNN algorithm is slightly different than the MLP algorithm that was used in this experiment. The two algorithms are similar in structure both utilizing a neural network in order to classify data. The results of their experiment are comparable to the MLP algorithm that was used in this experiment which boasted similar results in being able to correctly classify the image class with about 88% (average of both testing groups). Similar results of both the algorithms that were used in this experiment can be seen in the work of Mohsen, El-Dahshan, El-Horbaty, and Salem. In their work of using image classifiers to classify MRI images as discussed in the related work of this paper [3]. The results of their work show that the KNN algorithm that they used generated the correct image class 95% for K = 1 and 86% for K = 3. The CNN

algorithm that was used also predicted the correct image class 96% of the time (this percentage as well as the KNN percentages are the “classification rate” provided by the authors article).

Limitations and Future work

Due to the limited time frame of the project future work includes using the Python Pillows library to optimize the preparation of the images before being resized by using the Pillows Box function to identify the “most relevant” part of the image, in this case the most relevant part would be the Pokémon itself, before resizing the image. This could help reduce the amount of background noise that may exist in an image although this may not have too much of an impact on the data that was used for this particular project because the image data we used for this project were relatively small and simple when compared to large complex images such as an image taken by a high-resolution camera. Another part of future work that can be done is increasing the number of classes for the image classifier to try to identify. At the time of the project's conception, it was planned to have multiple sets of testing groups. The first two sets of testing group, which was the data that was used for this experiment that contains a group of four classes (four different Pokémon) for the classifier to identify. The other testing groups that was planned for this project was to scale the size of the testing group to include all 151 Pokémon that make up the first generation of the Pokémon series. Speaking about the images and data of the project future work can also involve being able to classify colored images, at this time the classifiers we used for this project do not allow the use of colored images. The last part of future work that we can continue for this project is to include more image classifiers that identifies the data. For example, utilizing different libraries of the two used KNN and MLP classifiers, as seen in the related work section of this paper.

Conclusion

In conclusion, the results of the experiment done in this paper in order to provide an insight to other researchers who are conducting experiments on or with image classification. Shows that while using gray scale images of the same size shows that the use of a KNN algorithm provides similar accuracy (percent of the algorithm to correctly identify the class of an image) rates to that of the MLP algorithm that utilizes a neural network structure to classify simple small images as the ones seen in **Figures 1** and **2**.

References

[1] "What Is Image Recognition?," MathWorks. Accessed: Sep. 30, 2023. [Online]. Available: https://www.mathworks.com/discovery/image-recognition-matlab.html

[2] "What is the k-nearest neighbors algorithm? | IBM." Accessed: Nov. 17, 2023. [Online]. Available: https://www.ibm.com/topics/knn

[3] H. Mohsen, E.-S. A. El-Dahshan, E.-S. M. El-Horbaty, and A.-B. M. Salem, "Classification using deep learning neural networks for brain tumors," *Future Computing and Informatics Journal*, vol. 3, no. 1, pp. 68–71, Jun. 2018, doi: 10.1016/j.fcij.2017.12.001.

[4] "Pokémon sprites: archive of Pokémon images from every game." Accessed: Sep. 26, 2023. [Online]. Available: https://pokemondb.net/sprites

[5] Lin Meng, Takuma Hirayama, and Shigeru Oyanagi, "Underwater-Drone With Panoramic Camera for Automatic Fish Recognition Based on Deep Learning," IEEE Xplore. Accessed: Sep. 24, 2023. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/8327594

[6] G. Batchuluun, Y. W. Lee, D. T. Nguyen, T. D. Pham, and K. R. Park, "Thermal Image Reconstruction Using Deep Learning," *IEEE Access*, vol. 8, pp. 126839–126858, 2020, doi: 10.1109/ACCESS.2020.3007896.