Fruit Classification based on various models and activation functions

Feiyun Yan, Olivia Yang, Siyan Wang, Yixuan Wu

University of Wisconsin Madison

Abstract

In this project, we implemented Multilayer perceptron (MLP) and Convolutional Neural Networks (CNN) for classification on a dataset composed of 10 classes of fruits and investigated the performances of MLP and CNN. We found that CNN outperformed MLP on this dataset. Additionally, we explored how three different activation functions perform on CNN: rectified linear unit (ReLU), hyperbolic tangent (tanh), and sigmoid, and made comparisons on test accuracy, time-consuming, and vanishing gradient problems, respectively. ReLu is proven to have better overall performance than the other two methods, as it proves the accuracy by more than 4%. Our findings demonstrate the effectiveness of using CNN for fruit classification and the importance of selecting an appropriate activation function which is ReLU in our project.

Keywords: Deep Learning, Multilayer perceptron, Convolutional Neural Networks, Activation function

Contents

1. Introduction

In this project, we implemented various deep learning models, including MultiLayer Perceptron (MLP) and Convolutional Neural Network (CNN) for image classification, and compared their performances on the specific dataset. We chose a dataset of fruit classification, which includes 10 classes of different types of fruits as a specific scenario to explore. Then we investigated the performance of different models with different hyperparameters and paid additional attention to exploring the functionality of different active functions in CNNs.

To investigate the performances of various models, especially for a concrete problem, we applied these MLP and CNN models to the specific fruit classification dataset and made comparisons on performances.

As a powerful neural network architecture, Multilayer Perceptron (MLP) is widely used for classification tasks in various domains. Composed of multiple layers of interconnected artificial neurons, MLP leverages feedforward propagation to process input data and produce class predictions. In MLP, each neuron applies a non-linear activation function to transform its weighted sum of inputs, which enables complex nonlinear to map from input spaces to output spaces. It is trained through backpropagation, where the error signal is propagated backward, adjusting the connection weights iteratively to minimize the classification loss. Generally, MLP has become a popular strategy for classification problems in various disciplines due to its versatility and capability of handling high-dimensional data and capturing intricate decision boundaries.(Murtagh, 1991)

Another fundamental class of deep learning models is the Convolutional Neural Network (CNN), which is also a type of prevalent solution to image classification tasks. Depending on the unique architecture, CNN exhibits exceptional performance in automatically learning and extracting relevant features from raw image data. CNN is featured with a series of convolutional layers followed by non-linear activation functions and pooling operations, enabling it to efficiently capture spatial hierarchies and local patterns within images. Then these representations are fed into fully connected layers for classification, supporting its remarkable accuracy in discriminating between different classes. Overall, CNN has emerged as a state-of-the-art solution for image classification, revolutionizing the field of computer vision and advancing various real-world applications.(Guo et al., 2017)

The project is designed to implement various MLP and CNN models to the fruit classification dataset and make comparisons of their performances. In addition, it is aimed to explore the functionality of different hyperparameters in both MLP and CNN models, such as the number of hidden layers and batch size. In particular, we have specific interests and pay additional attention to activation functions in CNN models. We inspected the performances of CNNs with different active functions: rectified linear unit (ReLU), hyperbolic tangent (tanh), and sigmoid functions, and made comparisons on their test accuracy, time-consuming, and vanishing gradient problems respectively.

2. Related Work

Recognizing diseases and assessing the quality of fruits pose significant challenges in the agricultural sector, and automating their identification is crucial for time-saving and preventing financial losses. The manual task of visually inspecting and classifying fruits in crops can be arduous, wasting valuable time that could be utilized more efficiently. In this way, the fruit classification problem has become a branch of image classification tasks for a long time.

One prevalent solution to fruit classification tasks is based on a six-layer CNN, which contains convolution layers, pooling layers and fully connected layers as its structure. It demonstrates 91.44% accuracy with a dataset with 1800 images and 9 fruit types. Chen et al. (2017) use the ReLU function as the active function in the convolutional layer to get rid of the gradient vanishing problem and choose the softmax function to be the active function in the output layer (fully connected layer), instead of the sigmoid function.(Chen et al., 2017)

Another state-of-the-art CNN approach relevant to our project is a 13-layer deep CNN method designed for fruit category classification. This method shows an average accuracy of 94.94% for the specific classification dataset with 18 fruit types and 200 images for each fruit type. By observation, Zhang et al. (2019) choose four combined layers as the structure of the CNN for optimal performance. (Zhang et al., 2019)

Moreover, there exists another novel algorithm Fruit-CNN, showing a test accuracy of 99.6% in real-world images with multiple visual variations. The suggested framework offers efficient training and testing of fruit images, surpassing existing deep learning models in terms of time. Additionally, the model's capability to handle a greater number of images from different classes further contributes to its fast training process, making it a promising solution in the field.(Kumar et al., 2021)

Compared to these state-of-art frameworks, the main objective of our project is not to find "the best" model with the highest accuracy for our fruit classification dataset but to investigate differences various hyperparameters could make in MLPs and CNNs, especially active functions in CNNs.

3. Proposed Method

3.1. Multilayer Perceptron

We first used a Multilayer Perceptron (MLP) as a structure for classification prediction. MLP has a multilayer structure consisting of many layers of neurons, with each layer connected to the next layer of neurons. Since each hidden layer contains multiple neurons, it can extract higher-level features of the data more efficiently. It uses a backpropagation algorithm to tune the parameters of the model and optimize the predictive power of the model by performing training.

Multi-layer Perceptron belongs to the category of feed-forward algorithms because the input is combined with the initial weights in the weighted sum and is influenced by the activation function. (Jain, 2021) Therefore, the algorithm needs to learn to minimize the weights of the cost function, which is done by backpropagation to iteratively adjust the weights in the network.

The linear model is not the most accurate because it will increase the output of the model when there are too many features, so we use a non-linear activation function for each hidden layer. Too many hidden layers can lead to overfitting problems, so we added a dropout function to the model. The output of each linear layer of our model is passed through the ReLU activation function, which is used to introduce nonlinearity into the model. In the three hidden layers, the dropout function is applied with the probability of 0.5, 0.3, and 0.2 during training. Specifically, the use of ReLU activation function is used to alleviate the problem of gradient disappearance, and the model is more sparse and easy to train. ReLu provides a simple nonlinear transformation. When our element is x, the function of ReLu is defined as the maximum value of x and 0:

$$ReLu(x) = \max(x, 0)$$

In our project, we adopted softmax as the final activation function. Define our data as $Z = z_1, z_2, ...z_n$ for $i = 1, ...n$ for n sample, where Z_i represents the i_{th} element from the dataset Z, the softmax of the i_{th} element as the following:

$$Softmax(z_i) = \frac{exp(z_i)}{\sum_j exp(z_j)}$$

Authors of the book "Deep Learning" mention that "Softmax units naturally represent a probability distribution over a discrete variable with k possible values, so they may be used as a kind of switch." (Goodfellow et al., 2016) The Softmax function normalizes the original data of the previous layer and converts it into a value between (0, 1) to perform classification tasks.

3.2. Convolutional neural network (CNN)

Another method we tried is CNN, also known as a convolutional neural network. Convolutional Neural Network is a type of multilayer perceptron used in deep learning models, and one of the known functions is to analyze visual images. It was used in speech recognition and document reading in the early 1990s and reads more than 10% of all checks in the United States. (LeCun et al., 2015) As Turkey Istanbul mentions in the article

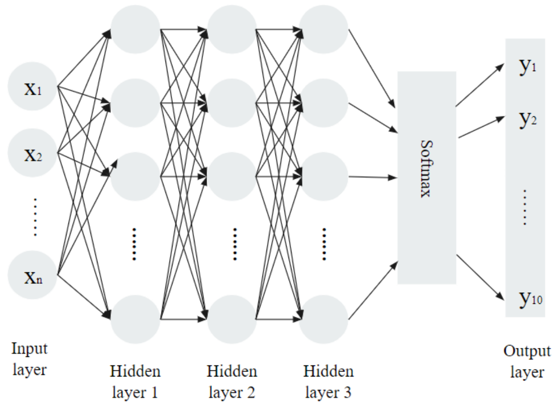

Figure 1: The structure of Multilayer Perceptron

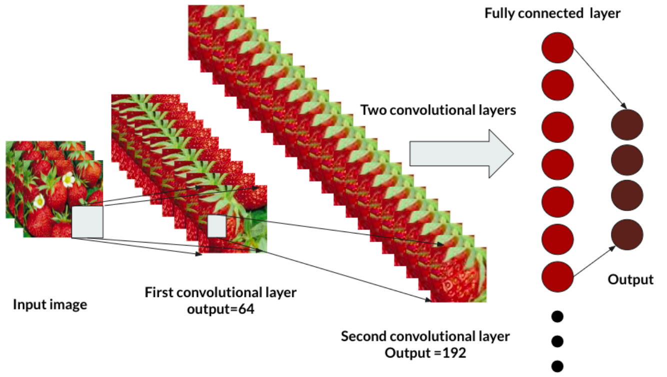

Figure 2: The structure of CNN

"Understanding of a Convolutional Neural Network", CNN has made breakthroughs in many fields related to pattern recognition in the past decade. It contains neural networks with convolutional calculations and deep structures.CNN is a useful tool for image classification, and it can perform better than classical multilayer perceptrons. Our convolutional neural network is based on the AlexNet model(ima, 2012). The structure of the Convolutional neural network can be briefly divided into two parts, that is the construct of the feature extractor and classifier. We will introduce the core part of our CNN model, the feature extractor, in this part. And the classifier introduction will be included in the experiment part. As we can see from Figure 2, the feature extractor contains 4 convolutional layers, The first layer has 3 channel features, which is due to the fact that we worked on RGB images. And the output feature is 64. Then the following 3 layers flatten the image data, from 64 to 256. For the choice of activation function, we remain the design of AlexNet. In the first three layers, we use Leaky ReLU, and regular ReLU activation in the last layer.

$$Leaky\ ReLU(x) = \begin{cases} ax, & \text{if } x < 0 \\ x, & \text{otherwise} \end{cases}$$

Maxpool layers are used in the first layer and second layers to reduce the spatial size of feature maps to reduce the workload of the neural network, making the process more efficient. Maxpool layers are used to retain the important information of those pictures while reducing the spatial size. In the same opinion, we use the average pooling layer to resize the feature map to a fixed size, which is 6 times 6, at the end of the feature extractor.

3.3. Three kinds of activation functions

We introduce three kinds of activation functions we implemented in our Convolutional Neural Network (CNN) model here and will discuss our results, comparisons on activation functions, and potential problems caused by them in the Results and Discussion section. Hyperparameter settings are referred to in the CNN part of the Experiments section. All three activation functions can be found in the PyTorch package in Python.

- Sigmoid, which can be represented mathematically as:

$$f(x) = \frac{1}{1+e^{-x}}$$

The function takes any input with real values in the range of 0 and 1. Larger inputs will produce an output closer to 1.0, and smaller inputs will produce an output closer to 0.0. It's commonly used to predict the probability of an output (Baheti, 2021) but may not be a computationally efficient function and its outputs are not zero-centered.

- Hyperbolic tangent (tanh), which can be represented mathematically as:

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

The function is similar to the Sigmoid activation function but takes a larger range of input which is from -1.0 to 1.0. Instead of tending to approach 0.0 when inputs decrease in magnitude, it will be closer to -1.0 (Baheti, 2021). It produces zero-centered outputs but is a little more complex when computing.

- Rectified Linear Unit (ReLU), which can be represented mathematically as:

$$f(x) = \max(0, x)$$

The function does not define a range for input and also has a much higher time efficiency compared to Sigmoid and Tanh activation functions. It will turn negative and zero inputs to 0 and keep positive inputs as original (Baheti, 2021).

4. Experiments

For all methods, we utilized separate training datasets to train MLP and CNN (with three different activation functions) and testing datasets to evaluate model performances. We conducted multiple iterations of adjustments and selected the optimal combination of hyperparameters. All results will be reported based on the test accuracy.

4.1. Data processing

The dataset we used in our project is fruit classification from Kaggle (Karim, 2018).in order to test the performance of our models on color image classification. We have ten classes of fruits with each of them having approximately 460 images for the training dataset and 100 images for the testing dataset, respectively. The fruit images are of various sizes in RGB format. In order to identify the data more conveniently, we replace these ten fruits' names with numbers from 0 to 9.

4.2. Multilayer Perceptron

There are a total of 4500 pictures in this set of data, and they are sorted according to the classification of fruits. In order to prevent the model from taking the order of pictures as a rule during the learning process, we randomly shuffle the pictures in the training set. Since the complexity of color images, we adopt 3 hidden layers. After resizing the image to 200 x 200 pixels, we turn the 200x200 fruit image into 40,000 pixel values as input (there are corresponding 40,000 nodes in the input layer). As we can see from Figure 1, there are 750 nodes in the first hidden layer, 500 nodes in the second hidden layer, 250 nodes in the third hidden layer, and 10 nodes in the final output layer, which correspond to ten different fruits). There is an example

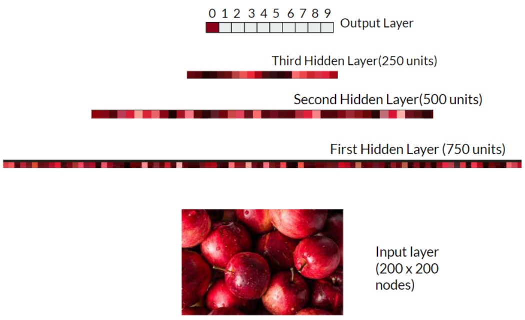

Figure 3: Example image with MLP

of images going through the Multilayer Perception with hidden layers in Figure 3. We set the batch size to 128 in each iteration and then set the mean value of the transform training set and test set to 0.5, and the standard deviation to 0.5. After several adjustments, we choose the final learning rate of 0.03. Multilayer perceptron was trained with 100 epochs, and the total time of processing is 34.75 minutes.

4.3. CNN experiment

For the classifier, the batch size is set as 256. The classifier contains several fully connected layers, with 2000 output neurons. Two dropout layers are set to avoid overfitting. During dropout, a neuron is dropped from the Neural Network with a probability of 0.5. When a neuron is dropped, it does not contribute to forward propagation or backward propagation. Actually, the dropout has an excellent performance in the neural network. We set the number of epochs to be 50, and it spent 62.84min to finish the training on our computer(2017 MacBookAir).

5. Results and Discussion

In this section, we demonstrate detailed results we obtained from the MLP model, CNN model, Sigmoid, Tanh, and ReLU activation functions, respectively. In addition, we will discuss the limitations of our models, possible issues taken by activation functions, and future extensions we may implement to improve our project.

5.1. MLP Results and Discussion

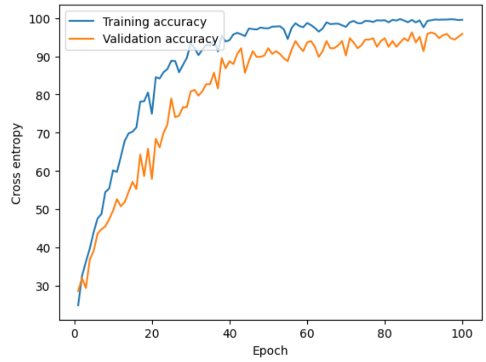

Figure 4: The training accuracy and validation accuracy of the Multilayer Perceptron Model

As we can see from Figure 4, the accuracy of the training set is 99.55%, and the accuracy of the validation is 95.86%. However, the accuracy of the test set is 67.28% after 100 iterations. Although we adopted the dropout function and validation methods to avoid overfitting, our accuracy rate is still less than 70%. In reality, the parameter overhead of a multilayer perceptron with fully connected layers is huge, especially when we have multiple hidden layers and multiple units. Since MLP cannot effectively process a large number of input features, including high resolution and a large number of pixels, when recognizing color images, it leads to an inaccurate rate when processing the image.

5.2. CNN results and discussion

The accuracy of classification using CNN is better than MLP's. Firstly, there is almost no overfitting in the training period. The training accuracy remains the same as the validation accuracy over the whole training period. We set the number of epochs to be 50, and it spent 62.84 mins to finish the training on our computer(2017 MacBookAir). At the end of our training period, the training accuracy is 98.39 percent and the validation accuracy is 98.87 percent. Figure 5 shows the training accuracy and validation accuracy, and Figure 6 shows the training loss and validation loss.

There is some backwardness in our CNN model. Although the minibatch loss keeps on reducing, there is a huge oscillation throughout the whole training period, which should be dampened by some other method, such as adaptive Learning Rate and momentum. Finally, the test accuracy is 73.74%. The dataset contains many kinds of pictures of 10 kinds of fruit, some of

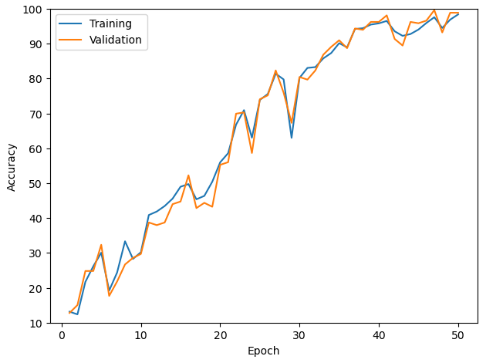

Figure 5: Training accuracy and validation accuracy in training period

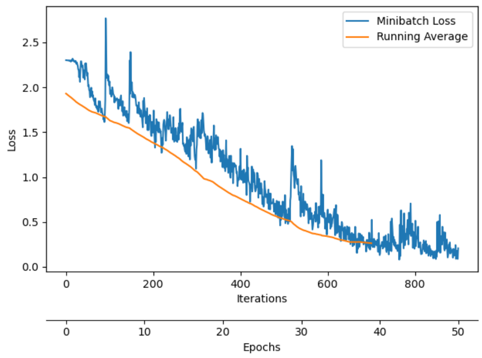

Figure 6: Minibatch loss with CNN

them is quite difficult to be recognized even by human eyes, and we believe the result has been the best.

Here is a visualization of our result in Figure 7. The image that should be classified as apple has been collected as orange, but we think that is quite reasonable, cause both apple and orange appear in this image. Another misclassification is the pineapple is classified as a Banana. That pictures can confuse people, as the raw pineapple and banana look almost the same.

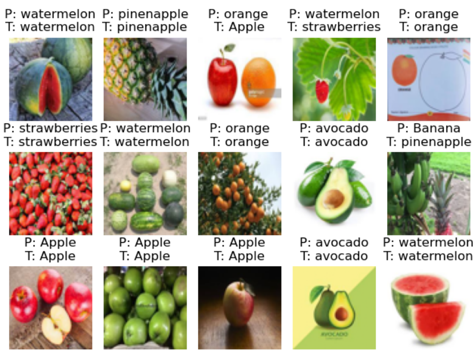

Figure 7: Example of classification with CNN

The Figure 8 is a confusion matrix of our work. The color of each cell shows the percentage of the right classification in our test data set. We find that pineapple is the best-classified fruit, while mango is the most easily misclassified fruit. We conclude that is because raw mango looks like raw bananas, especially when they hanging on the tree, not being picked yet. But most of those fruits have been collected to the right category, which means our CNN model did a great job.

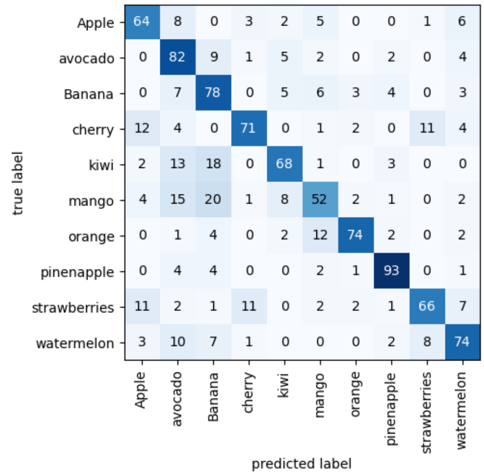

Figure 8: Confusion matrix with CNN result

5.3. Activation Functions Results and Discussions

We will discuss how different activation functions affect model performance, time consumed, and limitations that may cause errors or slow down the training process. Basic results and further comparisons are provided in Table 1 and Table 2, respectively.

	Sigmoid	Tanh	ReLU
Training Accuracy	89.61%	94.36%	97.81%
Testing Accuracy	57.33%	69.74%	73.74%
Time consumed(min)	209.01	161.37	157.58

Table 1: Results of three activation functions

As shown in Table 1, we found ReLU, compared with another two activation functions, has the highest testing accuracy and shortest time consumed to compute. We summarize the results in Table 1 and do further comparisons on activation functions in Table 2 and try to dig in more deeply for reasons of shorter computation time and possible concerns during the process.

	Sigmoid	Tanh	ReLU
Time Consumed	3rd	2nd	1st
Overall Accuracy	3rd	2nd	1st
Vanishing Gradient Problem	YES	YES	NO
Zero-centered function	NO	YES	NO

Table 2: Further comparisons of activation functions

Vanishing Gradient Problem is a common issue that appeared in the training process of neural networks due to Sigmoid and Tanh may squeeze inputs from a large range into a much smaller span between -1.0 and 1.0 or 0 and 1. As a result, even if the input has a significant change in magnitude, the output always has a small change which will lead to a small derivative, then the weights in the neural network cannot get updated and the model cannot learn efficiently or stop learning at all (Wang, 2019).

The zero-centered function is another important factor influencing the learning and convergence of a model. The reason why Sigmoid, which is not a zero-centered function, has a much slower process of training might be it always produces positive outputs which cause gradients of a neuron in one layer will be all positive or all negative. This issue sometimes will be a carrier for the model to converge. Tanh, instead produces inputs between the range of -1.0 and 1.0, and does have zero-centered outputs which can help the model have faster convergence (Shaw, 2022).

To sum up the information in Table 1 and Table 2, we can say that Sigmoid has the vanishing gradient problem and is not a zero-centered function, Tanh also has the vanishing gradient problem but as a zero-centered function, it has higher computational efficiency, ReLU solves the problem of vanishing problem and keeps its time efficiency as the highest one because of its simple calculation. Therefore, we chose ReLu as our activation function in this project.

5.4. Future Work

Although ReLU provides strong performance in our project, it still has a potential problem called the Dying ReLU Problem which might be caused by the inactivity of all inputs. In other words, there are "no gradient flows" in the model (Himanshu, 2019). The problem does not appear in our project but might be found when we want to apply our model to another dataset. As a result, we can try to solve this problem with experiments of more kinds of activation functions, such as LeakyReLu which can change inputs smaller or equal to 0 to transformed numbers instead of 0 and therefore increases the range of ReLU.

6. Conclusions

By training and testing these two types of models on the 10-class fruit classification dataset, we compare their performances and conclude that the CNN model is a better approach to handling complex image datasets given higher testing accuracy of 73.74% compared to the MLP method with testing accuracy of 67.28% in this case.

Additionally, during the process of adjusting hyperparameters, we find that adjustment of the learning rate and batch size could influence the convergence rate and the stability of the training process, but may also lead to overfitting at the same time.

Furthermore, ReLU is validated to help CNN achieve the best performances, according to the correspondingly highest testing accuracy of 73.74%. Among the comparisons of three active functions in CNN: tanh, sigmoid, and ReLU functions, the CNN with the ReLU function as the active function illustrates the best performance in this case, referring to its highest overall accuracy in testing set and evasion from the risk of vanishing gradient problem.

7. Contributions

Feiyun Yan implemented data preparation and the Multilayer Perceptron models, and create visualizations of the method, and the corresponding sections in the report. Olivia Yang tests the performances of different activation functions in our project and writes up corresponding sections of activation functions in the report. Siyan Wang implemented the CNN models, and create visualizations of the method and the corresponding sections in the report. Yixuan Wu reads related literature and works, contrasts different models, and writes up corresponding sections in the report.

References

, 2012. Imagenet large scale visual recognition challenge 2012 (ilsvrc2012). URL: `https://image-net.org/challenges/LSVRC/2012/results.html`.

Baheti, P., 2021. Activation functions in neural networks [12 types & use cases]. URL: `https://www.v7labs.com/blog/neural-networks-activation-functions`.

Chen, Y.P.P., Chen, Y.H., Chen, W.C., 2017. Fruit classification based on six layer convolutional neural network, in: 2017 8th IEEE International Conference on Awareness Science and Technology (iCAST), IEEE. pp. 197–201. URL: `https://ieeexplore.ieee.org/abstract/document/8286904`, doi:10.1109/ICAwST.2017.8286904.

Goodfellow, I., Bengio, Y., Courville, A., 2016. Deep Learning. MIT Press.

Guo, T., Dong, J., Li, H., Gao, Y., 2017. Simple convolutional neural network on image classification, in: 2017 IEEE 2nd International Conference on Big Data Analysis (ICBDA), IEEE. pp. 721–724.

Himanshu, S., 2019. Activation functions: Sigmoid, tanh, relu, leaky relu, prelu, elu, threshold relu and softmax basics for neural networks and deep learning. International Journal of Computer Science and Mobile Computing 8, 1–9.

Jain, V., 2021. Multilayer perceptron explained with a real life example and python code (sentiment analysis). URL: `https://towardsdatascience.com/multilayer-perceptron-explained-with-a-real-life-example-and-python-code-sentiment-analysis-cb408ee93141`.

Karim, A., 2018. Fruit classification (dataset. URL: `https://www.kaggle.com/datasets/karimabdulnabi/fruit-classification10-class/code?resource=download`.

Kumar, A., Joshi, R.C., Dutta, M.K., Jonak, M., Burget, R., 2021. Fruit-cnn: An efficient deep learning-based fruit classification and quality assessment for precision agriculture, in: 2021 13th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), pp. 60–65. doi:10.1109/ICUMT54235.2021.9631643.

LeCun, Y., Bengio, Y., Hinton, G., 2015. Deep learning. Nature 521, 436–444.

Murtagh, F., 1991. Multilayer perceptrons for classification and regression. Neurocomputing 2, 183–197.

Shaw, S., 2022. Activation functions compared with experiments. Neural Networks 152, 1–12.

Wang, C., 2019. The vanishing gradient problem. Journal of Machine Learning Research 20, 1–15.

Zhang, Y.D., Dong, Z., Chen, X., Jia, W., Du, S., Muhammad, K., Wang, S.H., 2019. Image based fruit category classification by 13-layer deep convolutional neural network and data augmentation. Multimedia Tools and Applications 78, 3613–3632. URL: `https://doi.org/10.1007/s11042-017-5243-3`, doi:10.1007/s11042-017-5243-3.