**Project Report**

**Image Classification CNN vs FNN**

**Ps. No: 40027784**

**Name: Tharun Prabhakar**

**Table of Contents**

|  |  |
| --- | --- |
| **S. No** | **Title** |
| **1** | **Problem Statement** |
| **2** | **About Dataset** |
| **3** | **Algorithms Used** |
| **4** | **Experiments/Implementation** |
| **5** | **Results** |
| **6** | **Testing** |
| **7** | **Conclusion** |
| **8** | **References** |

**Problem Statement**

We live in the era of data. With the Internet of Things (IoT) and Artificial Intelligence (AI) becoming ubiquitous technologies, we now have huge volumes of data being generated. Differing in form, data could be speech, text, image, or a mix of any of these. In the form of photos or videos, images make up for a significant share of global data creation.

Since the vast amount of image data, we obtain from cameras and sensors is unstructured, we depend on advanced techniques such as machine learning algorithms to analyse the images efficiently. Image classification is probably the most important part of digital image analysis. It uses AI-based deep learning models to analyse images with results that for specific tasks already surpass human-level accuracy (for example, in Face Recognition).

Image classification is the task of categorizing and assigning labels to groups of pixels or vectors within an image dependent on particular rules. The categorization law can be applied through one or multiple spectral or textural characterizations. This project involves the creation of image classification neural network that is been trained on CIFAR-10 dataset.

**About Datasets**

The CIFAR-10 dataset consists of 60000 32x32 colour images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images. The dataset contains labels of 10 classes Aeroplane, Automobile, Bird, Cat, Deer, Dog, Frog, Horse, Ship and Truck. These labels are numbered from 0 to 9. These classes are also mutually exclusive that is there is no link between automobile class and truck class they are treated as separate classes. Each image is blurry with of 32x32 dimension.

**Description of Dataset:**

The dataset is default dataset present in TensorFlow from which it is imported.



Once called the dataset is loaded using load dataset feature in keras.

Graphical user interface

Description automatically generated with low confidence

From the above image we can see that the images and the labels are directly loaded to the training and testing set.

A screenshot of a computer

Description automatically generated with medium confidence

The images are then divided as 50000 for training set and 10000 for testing set. The labels are in the form of 2D array.

Text

Description automatically generated

This array needs to be converted to 1D so that it’s easy for matching or identifying of images. Reshape function is used to change the dimension of the labels.

Text

Description automatically generated

Graphical user interface

Description automatically generated

Sample image has been tested and its corresponding labels are checked. The labels are encoded so requires labelling them.



The training set contains 5000 images of each class summing up to 50000 images and the testing set contains 1000 images of each class for testing. These images are in random order in both the sets, so the training does not go in order of class labels.

Text

Description automatically generated with medium confidence

Text

Description automatically generated

**Algorithms Used**

**Feed Forward Network:**

Feed Forward Network or FNN is an artificial neural network in which the nodes are connected circularly. The feed-forward model is the basic type of neural network because the input is only processed in one direction. The data always flows in one direction and never backwards/opposite. Since the flow is always moves forward hence the name feed forward network. FFN generally consists of an input layer, hidden layer, and output layer. In the input layer the features are assigned weights, and these are assigned randomly during initialisation and the weights gets updated as the model gets trained so that the output is as accurate as possible. The feature with the high weight as high importance compared to others. At each layer the features and the weights are multiplied and the summed together which is then compared with a threshold value usually 0 and the output is determined based on it. There’s also the use of bias to balance out the errors. For the determination of output is based on the activation function which determines which node of the output layer to be activated. The number of nodes in the 2 layers, the number of hidden layers all determined based on problem statement.

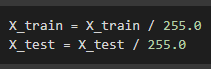
**Convolutional Neural Network:**

Convolutional neural network or CNN is the most popular neural network used in image classification problems. Convolutional neural networks are very good at picking up on patterns in the input image, such as lines, gradients, circles, or even eyes and faces. It is this property that makes convolutional neural networks so powerful for computer vision. Unlike earlier computer vision algorithms, convolutional neural networks can operate directly on a raw image and do not need any pre-processing. A convolutional neural network is a Feed Forward network, often with up to 20 or 30 layers. The power of a convolutional neural network comes from a special kind of layer called the convolutional layer. The usage of convolutional layers in a convolutional neural network mirrors the structure of the human visual cortex, where a series of layers process an incoming image and identify progressively more complex features. The CNN has 3 parts the convolution, classification, and output. It has 3 layers convolutional, pooling and fully connected layer. When a filter is applied matrix multiplication is done and result is updated to next filter. The filter moves block by block which is determined by stride. Default value of stride is 1. Padding is done so that the features are not lost on the feature map. Padding adds a layer of cells at the edges of the matrix generally the value inside cells be 0. Pooling layer is then used for dimensionality reduction. Generally max pooling method is used that selects the max value on the feature map. This is done so that chances of overfitting reduce, and time consumption of training is reduced. Before sending the data to fully connected layer flattening is done.

**Experiments and Implementation**

**Normalisation of Image Matrix:**

The images in the sets have colour ranges from 0 to 255 hence normalisation is done so that the values range from 0 to 1 and can be easily recognised by the model.



Before Normalisation:

Text

Description automatically generated

After Normalisation:

Text

Description automatically generated

**Building of Model:**

1. **FFN Model with Stochastic Gradient Descent:**

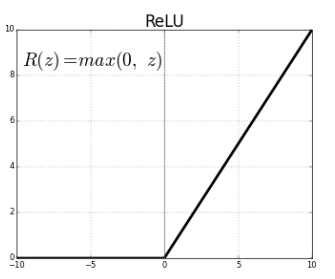
A simple 3-layer model was defined using sequential from keras. Before sending the data to the input layer flattening is done with input shape 32x32x3. Then it is fed into the input layer with 3000 nodes with activation ‘ReLu’ and sent to hidden layer with 1000 nodes with ‘ReLu’ activation and sent to output layer with 10 nodes to predict the type of class with ‘softmax’ activation. The model is then compiled using optimizer as ‘SGD’ for stochastic gradient descent and losses as ‘Sparse Categorical Crossentropy’ and metrics to be checked as ‘Accuracy’.

Text

Description automatically generated

**Stochastic Gradient Descent** is a type of gradient descent that is used mostly by deep learning algorithms as an optimizer. Gradient means slope and descent means moving downwards. The gradient descent is an optimization technique which finds the parameters such that the error term is minimum. It is an iterative method which converges to the optimum solution. It takes large steps when it is away from the solution and takes smaller steps closer to the optimal solution. The estimates of the parameter are updated at every iteration. Stochastic gradient descent instead of taking the whole datapoints it takes them at random, this reduces the time for the calculation of the optimising parameter.

**Rectified Linear Unit** Activation Function or ReLu is one of the most used activation functions



It gives the output as 0 if the value is equal to or less than 0 and gives z if the value is equal to z. It has become the default activation function for many types of neural networks because a model that uses it is easier to train and often achieves better performance.

**Softmax** is a function that turns a vector of K real values into a vector of K real values that sum to 1. The input values can be positive, negative, zero, or greater than one, but the softmax transforms them into values between 0 and 1, so that they can be interpreted as probabilities. If one of the inputs is small or negative, the softmax turns it into a small probability, and if an input is large, then it turns it into a large probability, but it will always remain between 0 and 1.

**Sparse Categorical Crossentropy** is an error in classification computed for the whole training set. It is similar Categorical Crossentropy only difference is that it’s used to when the data is integer labelled rather than one-hot encoding

The Model is then fit with train images and its labels with an epoch setting of 5.

**Epochs** means number of times the models have to be trained to get better required accuracy.

1. **FNN model with Adam Optimiser:**

Similar to previous model structure the model is built but while compiling the optimiser is given as ‘adam’ which is one of the most widely used optimiser to train and the model trained with the training set

Text

Description automatically generated

**Adaptive Movement Estimation or Adam algorithm** is an extension to stochastic gradient descent that has recently seen broader adoption for deep learning applications in computer vision and natural language processing. It is very efficient when there is huge data to be trained upon. It requires less memory and is efficient.

1. **FNN model with increased Epochs:**

The FFN is built with similar setting as the previous model, but the model is trained for higher number of epochs. The epoch is set to 10.

Text

Description automatically generated

1. **CNN model with 2 Convolutional Layers:**

The CNN model is built with 2 convolutional layers with the first layer having 32 filter with kernel size 3x3, ‘ReLu’ activation function with input shape 32x32x3.

Next a pooling layer is added with kernel 2x2 followed by a convolutional layer with 64 filters and same properties as the previous convolutional layer and pooling layer. Both the pooling layers used are Max pooling so that max value from the kernel is taken. After processing the data is then flattened using the flatten function an then it is sent to fully connected layer with 64 nodes for hidden layer and 10 nodes for predicting the class on the output layer. The model is compiled with ‘adam’ optimiser, sparse categorical crossentropy loss and accuracy as the metrics. The model is then fit to the train set with 5 epochs.

Text

Description automatically generated

Text

Description automatically generated

1. **CNN model with 2 Convolutional layers and 20 Epochs:**

The previous model is trained with 20 epochs where previously it was trained for 5 epochs.

**Text

Description automatically generated**

1. **CNN model with 3 Convolutional layers:**

In addition to the previous convolutional layers a third layer is added with 128 filters, 3x3 filter size with ‘ReLu’ activation which is followed by pooling layer with kernel size 2x2. This model is again trained with 5 epochs.

Text

Description automatically generated

1. **CNN model with 3 Convolutional layer and Increased Filters:**

The CNN 3 Convolutional layer as the previous one but the first layer has been updated with 64 layers. This model is again run for 5 Epochs

Text

Description automatically generated

1. **CNN with 3 Convolutional layers and 10 Epochs:**

The number of filters in first convolutional layer is reduced back to 32 and the model run for 10 epochs for increasing the accuracy.

Text

Description automatically generated

1. **CNN model with 15 Epochs:**

The previous network is trained with 15 Epochs.

**Text

Description automatically generated**

1. **CNN model with 20 Epochs:**

The same model is then run for 20 Epochs.

Text

Description automatically generated

1. **CNN model with 6 layers:**

This CNN model consists of 2 convolutional layers before the pooling layer adding up to 6 convolutional layers. Additionally, kernel initializer and padding options have been used.

**Kernel initializer** is used to set the value of the filter kernel with which feature extraction takes places. The option used here ‘he\_uniform’. This function is used to assign values from uniform distribution table to the kernel.

**Padding** is used to add a layer around the feature map so that there is no feature loss. The option used here is ‘same’. This option is used to preserve the dimensions of the feature map so that the output matches the input.

**Text

Description automatically generated**

**Results**

The model is fit with training set and the result predictions for the various models is obtained and accuracy score is checked.

**Accuracy Score** is a metric that is used for checking the performance of the model of classification problem. It compares with predicted value with test value and gives the score that ranges between 0 and 1. A model with high accuracy is a good model. It is ratio of true predicted values to all predicted values.

1. **FFN Model with Stochastic Gradient Descent:**

**Text

Description automatically generated**

1. **FNN model with Adam Optimiser:**

**Text

Description automatically generated**

1. **FNN model with increased Epochs:**

**Graphical user interface, text

Description automatically generated**

1. **CNN model with 2 Convolutional Layers:**

**Text, table

Description automatically generated with medium confidence**

1. **CNN model with 2 Convolutional layers and 20 Epochs:**

**Text

Description automatically generated**

1. **CNN model with 3 Convolutional layers:**

**Text

Description automatically generated**

1. **CNN model with 3 Convolutional layer and Increased Filters:**

**Text

Description automatically generated**

1. **CNN with 3 Convolutional layers and 10 Epochs:**

**Text

Description automatically generated**

1. **CNN model with 15 Epochs:**

**Graphical user interface, text

Description automatically generated**

1. **CNN model with 20 Epochs:**

**Graphical user interface, text

Description automatically generated**

1. **CNN model with 6 layers:**

**Graphical user interface, text

Description automatically generated**

**Testing**

The model after being built were tested with images selected randomly from GOOGLE and they results were stored.

Code for testing:

Text

Description automatically generated

1st Image:

Graphical user interface, application

Description automatically generated

2nd Image:

A picture containing text, cat, screenshot, domestic cat

Description automatically generated

3rd Image:

**Graphical user interface, application

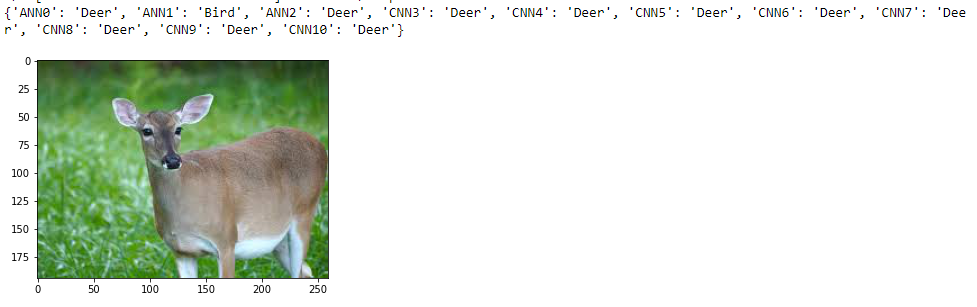
Description automatically generated**

4th Image:

Graphical user interface, application, Word

Description automatically generated

5th Image:



6th Image:

Graphical user interface, application, Word

Description automatically generated

7th Image:

Graphical user interface, application

Description automatically generated

8th Image:

Graphical user interface, application, Word

Description automatically generated

9th Image:

A picture containing text, bird, parrot, screenshot

Description automatically generated

10th Image:

Graphical user interface, application, Word

Description automatically generated

11th Image:

Graphical user interface, application, Word

Description automatically generated

12th Image:

Graphical user interface, application, Word

Description automatically generated

13th Image:

Graphical user interface, application, Word

Description automatically generated

14th Image:

A cat with its mouth open

Description automatically generated with medium confidence

**Scores of the Models**:

Code for getting the score:

Text

Description automatically generated

Scores for each model is represented below. The score is calculated based on the number of correct predictions of images out of 14 images of different class selected randomly.

Text

Description automatically generated with medium confidence

**Conclusion**

Table

Description automatically generated

For the comparison of models all the models were trained with batch size of 64, meaning the data is fed to the model in batches of 64.

The first 3 models that were built were basic type of artificial neural network that is the feed forward network. The first FFN that was built with stochastic gradient descent had an accuracy of 0.3540 in the testing set which means it had predicted almost 36 to images correctly out of 100. This indicates that the model is not able to recognize the patterns from the image to classify images into the separate classes. The second model with adam optimiser has also performed to an accuracy of 0.4736 which is providing slightly better results but as the number of epochs is increased for 3rd model, we can see that the accuracy has mild reduction to 0.4714 for the test set which means out of 100 images 47 to 48 will be correctly classified.

Next CNN models were built to check their performance with images and as we can see from the above table. The first model was built with 2 convolutional layer and run for 5 epochs this model provides an accuracy of 0.6587 for train set and 0.7044 for the test set meaning that the model is working well when compared with the previous FNN models. The next model the number of epochs was increased to 20. This resulted in overtraining of the model using the training set to get an accuracy of 0.8582 and test set with 0.6918. This indicates the overfitting of training data so the image from train set gets correct prediction compared to test set.

The next CNN models had 3 convolution layers with 128 filters in the last convolutional layers, this model was run for 5 epochs a result of 0.6591 for test set and 0.7288 for training set. In the next model the number of filters in the first convolutional layer is increased to 64. This model performed comparatively better on the test set than the previous model with an accuracy score of 0.6876 and 0.7043 on the training set. This is one of the models that has good predictions in both training and test set.

The following models have 3 layers similar to previous models with 32 filters in the first convolutional layer, but the number of epochs required to run the model was increased. First with 10 epochs a score of 0.7266 on test set and 0.8064 on the train set which indicates overfitting of model and as the number of epochs increases in the subsequent models training accuracy increases but there is not much increase in the test accuracy.

The final model that was built has 6 convolutional layer, 2 layers before pooling layer, this model was trained for 10 epochs and a score of 0.7537 for test and score of 0.9154 was obtained for the train set. The difference in the score is high indicating overfitting, but compared to rest of the models, it has better accuracy score.

The models that were built were tested with a series of images from different classes and the output of each model were recorded. The images of dogs, cats, airplanes, automobiles, birds, and ships were given at random to the models. We can see that most of the models are struggling to predict the class label correctly for Dogs and cats. This might be due to less accuracy of the models and images of cats and dogs were not sufficient enough for the models to learn the pattern of cats and dogs. All models have predicted images of classes ship, airplane, deer, and automobile better compared to other classes. At times there were misclassifications of images by the model.

The scores of the model were recorded and we can see that FNN model were able to predict a minimum of 6 images out of 14 but the CNN models were able to predict a minimum of 9 images out of 14 images. The final model with 6 convolutional layers has the highest number of predictions of 12 out of 14 images.

From this we can see that the CNN models were able to predict the classes much better than the FNN models because they have better accuracy and can reduce the number of features without losing the quality of the models.

Even though the final model has overfitting in it was able to predict most of the images accurately. From the result table and the prediction score of the models I conclude that the final model with 6 layers as best model for the classification of images. The drawback of the model is that it could not differentiate between cat and dog images accurately leading to misclassification of them at times.

To improve the Final model hyperparameter tuning could be done in order to improve the accuracy score of test set so that it can identify classes better than it is now.

**References**

1. Dataset Download: https://www.cs.toronto.edu/~kriz/cifar-10-python.tar.gz
2. [cifar10  |  TensorFlow Datasets](https://www.tensorflow.org/datasets/catalog/cifar10)
3. [Basic Introduction to Feed-Forward Network in Deep Learning (analyticsvidhya.com)](https://www.analyticsvidhya.com/blog/2022/03/basic-introduction-to-feed-forward-network-in-deep-learning/)
4. [Activation Functions in Neural Networks | by SAGAR SHARMA | Towards Data Science](https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6)
5. [Implementing feedforward neural networks with Keras and TensorFlow - PyImageSearch](https://pyimagesearch.com/2021/05/06/implementing-feedforward-neural-networks-with-keras-and-tensorflow/)
6. [An intuitive guide to Convolutional Neural Networks (freecodecamp.org)](https://www.freecodecamp.org/news/an-intuitive-guide-to-convolutional-neural-networks-260c2de0a050/)
7. [Convolutional Neural Network Definition | DeepAI](https://deepai.org/machine-learning-glossary-and-terms/convolutional-neural-network)