Silver Exercises

Deep Learning

6600

6600 ~ 80% Deep Learning main exercises marks

FFNN [500]

Consider a neural network with the following structure:

- Input layer with 3 neurons (including the bias neuron).
- One hidden layer with 2 neurons.
- One output layer with 1 neuron.

The weights and biases for the network are given as:

Input to hidden layer weights:

W_hidden

$$\begin{bmatrix} 0.2 & 0.4 \\ 0.3 & 0.5 \\ 0.5 & 0.1 \end{bmatrix}$$

Hidden layer biases:

b_hidden

$$\begin{bmatrix} 0.1 \\ 0.2 \end{bmatrix}$$

Hidden to output layer weights:

W_Output

$$\begin{bmatrix} 0.6 \\ 0.7 \end{bmatrix}$$

Output layer bias

b_Output

$$\begin{bmatrix} 0.3 \end{bmatrix}$$

Given an input vector

x

$$\begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

Perform the following tasks:

- Compute the input to the hidden layer neurons.
- Apply the sigmoid activation function to compute the output of the hidden layer.
- Compute the input to the output layer neuron.
- Apply the sigmoid activation function to compute the output of the network.

Back Propagation [600]

Consider a neural network with the following structure:

- Input layer with 2 neurons (excluding bias).
- One hidden layer with 2 neurons.
- One output layer with 1 neuron.

The initial weights and biases for the network are given as:

Input to hidden layer weights:

W_hidden

$$\begin{bmatrix} 0.15 & 0.25 \\ 0.20 & 0.30 \end{bmatrix}$$

Hidden layer biases:

b_hidden

$$\begin{bmatrix} 0.35 \\ 0.35 \end{bmatrix}$$

Hidden to output layer weights:

W_Output

$$\begin{bmatrix} 0.40 \\ 0.45 \end{bmatrix}$$

Output layer bias

b_Output

$$\begin{bmatrix} 0.6 \end{bmatrix}$$

Given an input vector

x

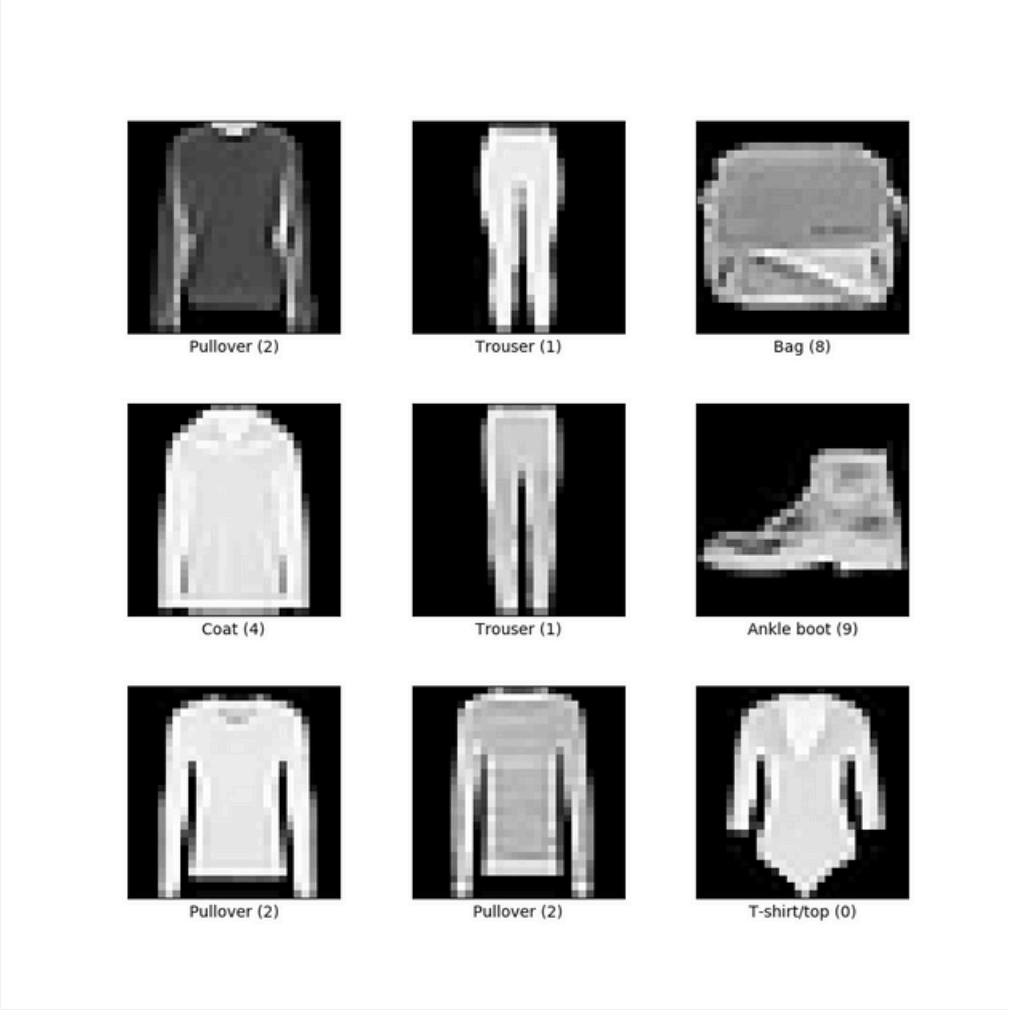
$$\begin{bmatrix} 0.05 \\ 010 \end{bmatrix}$$

Perform the following tasks:

- Perform a forward pass to compute the output of the network.
- Compute the error using Mean Squared Error (MSE) loss.
- Perform a backward pass to calculate the gradients of the weights and biases.
- Update the weights and biases using gradient descent with a learning rate of 0.5.
- Perform a forward pass with the updated weights and biases to compute the new output of the network.

FF Coding [500]

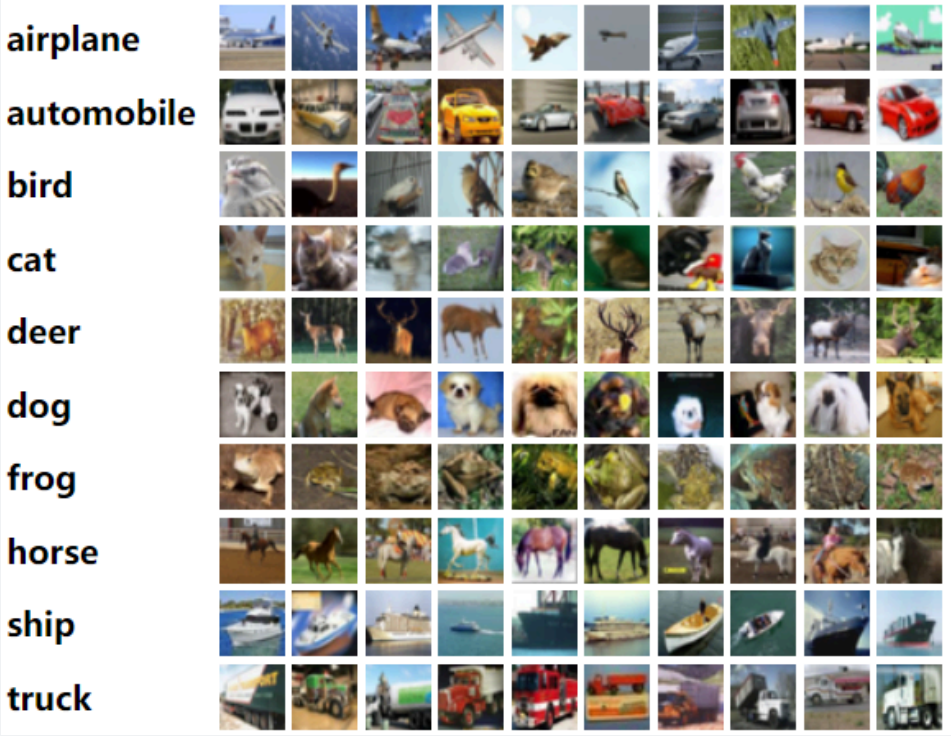
Build and train a neural network with dense layers using the Keras library in Python on the Fashion MNIST dataset.



Avoiding Over-fitting & Under fitting [500]

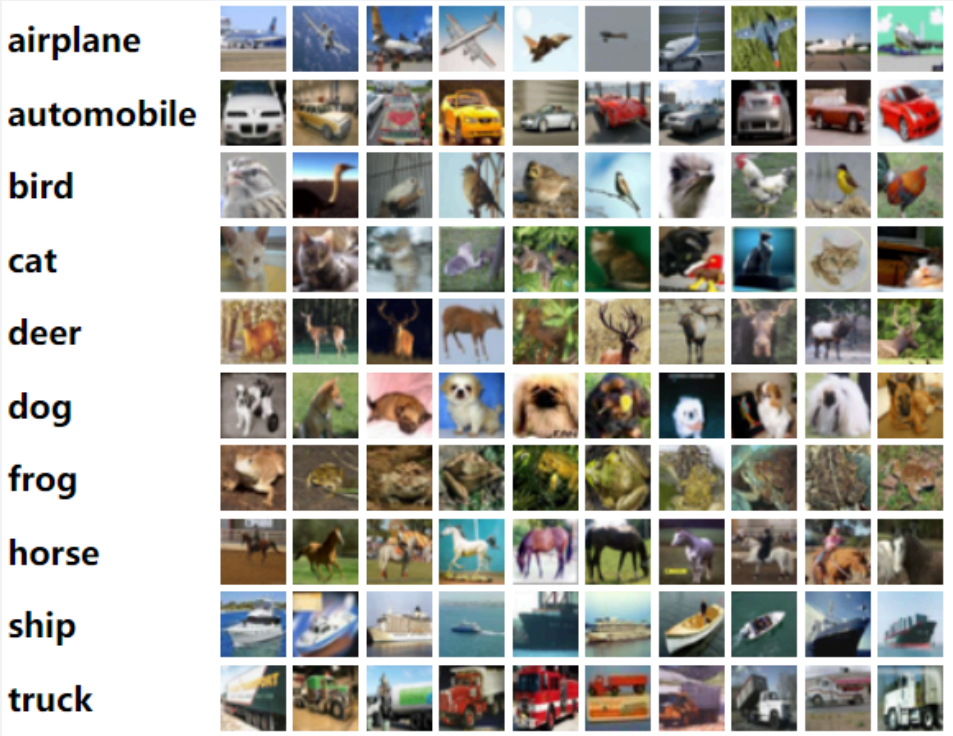
1. **Load the CIFAR-10 dataset and write a code to classify the classes.**

- Use only the dense layer.
- Use Overfit prevention methods. [Batch Normalization, Earlystop, Dropout, Data augmentation, ...]
- Use 10 % of the train-set as validation-set.
- Number of epochs: 50
- Before using data for training the model, **Concat generated data and original data.**
- Is your model suffering from overfitting or underfitting? Explain your answer.



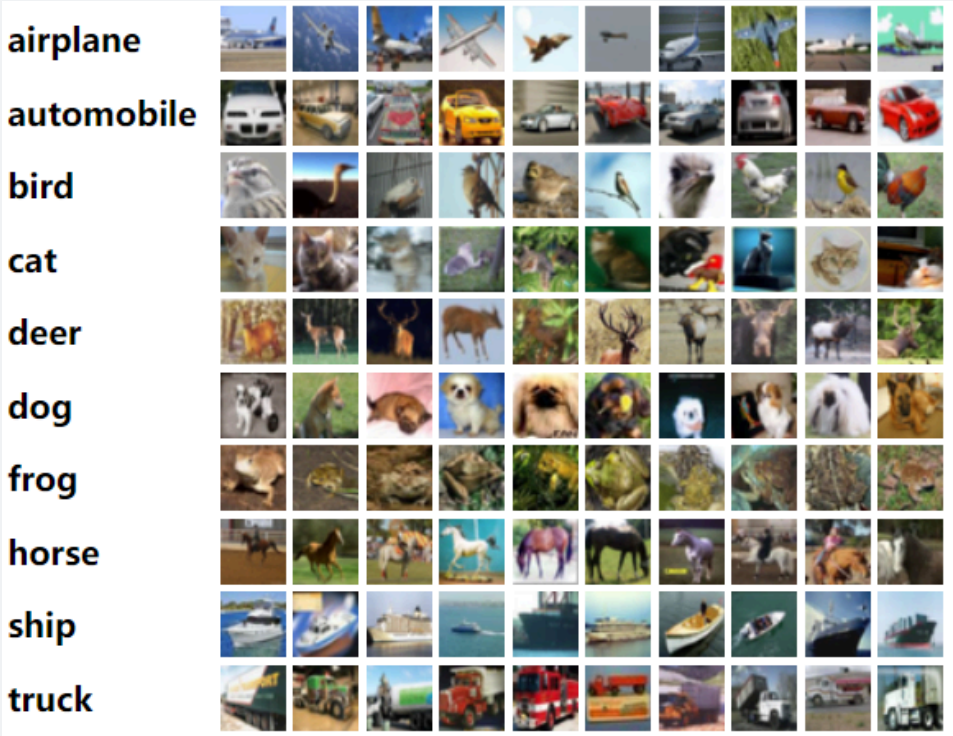
CNN [500]

build and train a convolutional neural network (CNN) using the Keras library in Python for image classification tasks. (CIFAR-10 dataset)



Data Augmentation [600]

implement data augmentation using the Keras library in Python to improve the performance of a convolutional neural network (CNN) on image classification tasks. (CIFAR-10 dataset)



Transfer Learning [500]

Use transfer learning to build and train a neural network for image classification using a pre-trained model (VGG16) (CIFAR-10 dataset)

RNN [600]

build and train a recurrent neural network (RNN) using the Keras library in Python for sequence prediction tasks. (use IMDB dataset)

AE [600]

build and train an autoencoder using the Keras library in Python for translating images of old faces to young faces. (UTKFace dataset or CACD dataset.)

GAN [800]

build and train a Generative Adversarial Network (GAN) using the Keras library in Python to generate synthetic handwritten digits similar to those in the MNIST dataset.

Pytorch [400]

Build and train a convolutional neural network (CNN) using PyTorch for image classification tasks. (CIFAR10 dataset)

HP Tuning [500]

Use the Keras Tuner library to perform hyperparameter tuning on a Classification neural network model. (to classify Fashion MNIST dataset classes)

Best Wishes