

Deep Learning And Neural network

Project 2



**Project**

**Neural Network and Deep Learning**

1. **Project Description:**

**CIFAR-10 Image Classification with CNN:**

This project aims to develop a Convolutional Neural Network (CNN) model for accurate classification of images in the CIFAR-10 dataset. The dataset consists of 60,000 color images of size 32x32 pixels, divided into 10 classes.

The project involves preprocessing the dataset by normalizing pixel values and augmenting the images to increase the training data's diversity.

A CNN architecture will be designed and implemented, leveraging convolutional, pooling, and fully connected layers to learn and extract meaningful features from the images. The model will be trained using the training set and its performance will be evaluated on the test set.

Hyperparameter tuning will be conducted to optimize the model's performance. Techniques such as regularization and dropout will be employed to prevent overfitting and improve generalization.

The accuracy, precision, recall, and F1-score of the CNN model will be evaluated to assess its classification performance on the test set.

The project will include visualization and analysis of the model's predictions, enabling insights into its decision-making process. This analysis will provide valuable information for potential model improvements and identifying misclassifications.

By utilizing the CIFAR-10 dataset and implementing a CNN model, this project aims to contribute to advancements in computer vision and improve the accuracy and efficiency of image classification tasks.

1. **Problem formulation:**

The problem at hand is image classification using the CIFAR-10 dataset. Given a set of 60,000 color images, each categorized into one of the 10 classes, the objective is to develop a Convolutional Neural Network (CNN) model that can accurately classify unseen images from the test set.

1. **Hyperparameter tuning**

A model hyperparameter is a characteristic of a model that is external to the model and whose value cannot be estimated from data. The value of the hyperparameter must be set before the learning process begins.

* Learning rate ῃ
* epochs

Run your neural network for tuning Learning rate and epochs using train test split validation. Fill in *Table 3.* Use the epoch and the learning rate with the best accuracy for further experimentation of the model.

**Validation Accuracy**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| epochs | 5 | 10 | 15 | 20 | 25 | 30 | 35 |
| ῃ = 0.1 | 0.1000 | 0.1000 | 0.1000 | 0.1000 | 0.1000 | 0.1000 | 0.1000 |
| ῃ = 0.01 | 0.1000 | 0.4848 | 0.1000 | 0.4199 | 0.5001 | 0.1000 | 0.4394 |
| ῃ = 0.001 | 0.6889 | 0.7044 | 0.7014 | 0.6748 | 0.6773 | 0.6839 | 0.6730 |

*Table 3.* Average Accuracy over folds

Plot train test graph for each of the learning rate and epoch value in Table 4. **[7 x 4 points]**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| epochs | 5 | 10 | 15 | 20 | 25 | 30 | 35 |
| ῃ = 0.1 |  |  |  |  |  |  |  |
| ῃ = 0.01 |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
| ῃ = 0.001 |  |  |  |  |  |  |  |

*Table 4. Train Test Learning Curves for each of the learning rate and epoch*

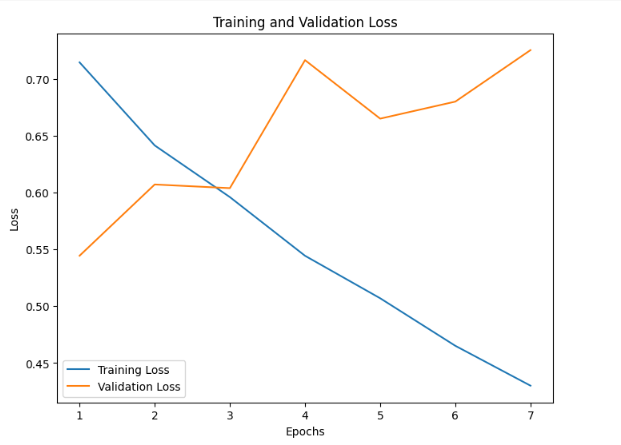
**USE THE LEARNING RATE AND THE NUMBER OF EPOCHS OBTAINED IN PART E FOR REPORTING RESULTS IN PART F, G,H**

1. **Cross validation**

Use the value k=3. Populate *Table 1* during experimentation.

|  |  |  |  |
| --- | --- | --- | --- |
| **Epoch** | **1st fold validation** | **Train Loss** | **Test Loss** |
| 1 |  | 1.5759 | 1.3128 |
| 2 |  | 1.2340 | 1.1623 |
| 3 |  | 1.0847 | 1.0798 |
| 4 |  | 0.9991 | 1.0183 |
| 5 |  | 0.9308 | 1.0071 |
| 6 |  | 0.8703 | 0.9720 |
| 7 |  | 0.8214 | 0.9773 |
| **Epoch** | **2nd fold validation** | **Train Loss** | **Test Loss** |
| 1 |  | 0.8661 | 0.8098 |
| 2 |  | 0.7984 | 0.7675 |
| 3 |  | 0.7432 | 0.8080 |
| 4 |  | 0.6980 | 0.7964 |
| 5 |  | 0.6497 | 0.8125 |
| 6 |  | 0.6129 | 0.8267 |
| 7 |  | 0.5736 | 0.9035 |
| **Epoch** | **3rd fold validation** | **Train Loss** | **Test Loss** |
| 1 |  | 0.7145 | 0.5444 |
| 2 |  | 0.6414 | 0.6071 |
| 3 |  | 0.5960 | 0.6038 |
| 4 |  | 0.5444 | 0.7164 |
| 5 |  | 0.5070 | 0.6650 |
| 6 |  | 0.4652 | 0.6799 |
| 7 |  | 0.4301 | 0.7252 |

1. **Generate the Train Test Error Learning Curve**

****

1. **Metrics to use for model evaluation**

For a balanced dataset and multiclass, you must use accuracy, F1 score, precision, recall, ROC AUC for the developed deep learning model evaluation. After each cross-validation fold, compute all the metrics above and fill in the values in *Table 2*.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Train Set** | **Accuracy** | **F1 score** | **precision** | **recall** |
| **Fold 1** | 83 | 0.828 | 0.835 | 0.826 |
| **Fold 2** | 86 | 0.854 | 0.863 | 0.856 |
| **Fold 3** | 86 | 0.858 | 0.865 | 0.86 |
| **Average** | 84 | 0.8466 | 0.8543 | 0.848 |
| **Test Set** | **Accuracy** | **F1 score** | **precision** | **recall** |
| **Fold 1** | 0.74 | 0.735 | 0.74 | 0.736 |
| **Fold 2** | 0.75 | 0.752 | 0.76 | 0.755 |
| **Fold 3** | 0.79 | 0.79 | 0.798 | 0.791 |
| **Average** | 0.76 | 0.76 | 0.76 | 0.76 |

*Table 2: Test and Train set Accuracy, F1 score, Precision, Recall,for each fold*