Abstract:

A Convolutional Neural Network (CNN) architecture is composed of a series of discrete layers that use a differentiable function to turn an input volume into an output volume. It also a deep learning model that is used to handle data with a grid pattern, such as images. The process of receiving an input and generating a class or a probability that the input is a specific class is known as image classification. The project is CNN architecture to classify the MNIST handwritten dataset. Target of this project to get more than or equal to 98% accuracy. Adam, SGD, RSMProp, 3 different optimizers are used in this project for testing purpose. For image processing task CNN is the best suited option.  MNIST dataset is a dataset of handwritten images. By adjusting different algorithm as describe previously, generate maximum efficiency with less loss.

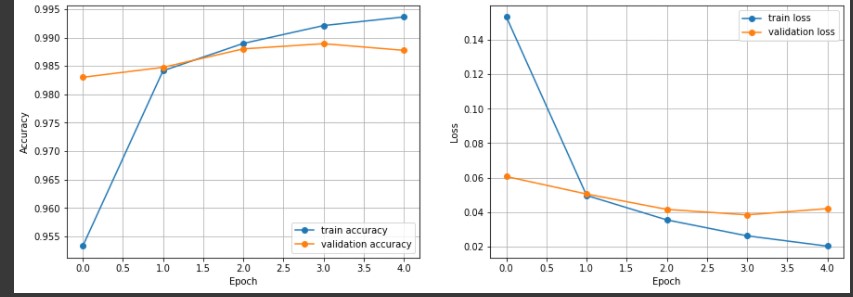
Introduction:

Convolutional neural networks (CNN) are neural networks with one or more convolutional layers that are primarily utilized for image processing, classification, segmentation, and other auto-correlated data. Data is used to train neural network algorithms to perform a number of machine learning tasks, such as categorization of distinct classes of objects, in various deep learning approaches. Convolutional neural networks (CNN) are deep learning algorithms that is used to analyze images very well. There are a variety of datasets that may be used to train convolutional neural networks. MNIST datasets used in this project.

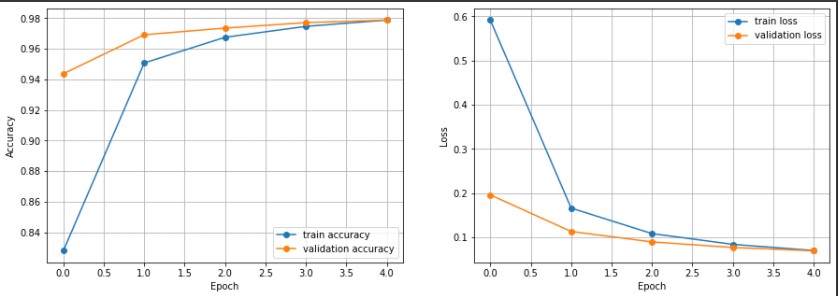
The Mixed National Institute of Standards and Technology (MNIST) has created a dataset of handwritten digits. This dataset is used to categorize handwritten digits and is one of the most investigated in machine learning. Because of its vastness, this dataset is useful for predictive analytics, allowing deep learning to work its magic quickly.

3 type of optimizer used in the project. Those are Adam, SGD and RSMProp. Adam is an optimization technique that is used to update network weights iteratively based on training data instead of the traditional stochastic gradient descent procedure. The iterative approach of stochastic gradient descent (SGD) is used to optimize an objective function with sufficient smoothness criteria. Because it substitutes the real gradient with an estimate, it may be considered a stochastic approximation of gradient descent optimization. This minimizes the computing cost, especially in high-dimensional optimization problems, allowing for quicker iterations in exchange for a reduced convergence rate. In neural network training, RSMProp is a gradient-based optimization strategy. Gradients in particularly complicated functions, such as neural networks, have a propensity to evaporate or explode as input passes through them. RSMProp was created as a stochastic mini-batch learning algorithm. RSMProp solves the problem by normalizing the gradient with a moving average of squared gradients.

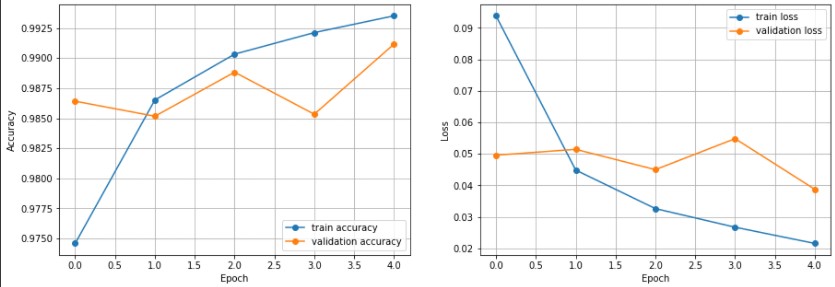
Result:

Adam Algorithm -> 

With an accuracy of 98.79% and a loss of 4%.

SGD Algorithm -> 

With an accuracy of 98.27% and a loss of 5.73 %.

RSMProp Algorithm ->

With an accuracy of 98.80% and a loss of 4%.

Here RSMProp is most efficient with only 4% loss and highest accuracy of 98.80%.

Discussion:

Optimizers are algorithms or methods that adjust the characteristics of a neural network, such as weights and learning rate, to decrease losses. Optimization algorithms are in charge of minimizing losses and delivering the most precise results possible. I utilized Stochastic Gradient Descent (SGD), Adam, and the RSMProp optimizer in my research. Adam, or Adaptive Moment Estimation, works with first and second order momentums. After computing the loss on each training sample, Stochastic Gradient Descent (SGD) seeks to change the model parameters.

After examining the data, it is clear that all of the optimizers worked admirably. First and foremost, the optimizer Adam was utilized, with an accuracy of 98.79 percent and a loss of 4%. The accuracy was 98.27 percent and the loss were 5.73 percent after applying SGD optimizer. Finally, after applying RSMProp, the accuracy was 98.80% and the loss was 4%. It may be argued that all three optimizers worked well in this case.