Training a Convolutional Neural Network (CNN) on the Fashion-MNIST dataset using PyTorch

Ashraf Mahmoud

Hamdy Zaied

ITI

13 / 5 / 2024

Table of Contents

Table of Figures

Abstract

Objectives: The major objective of this document is to implement convolution neural network for image classification problem for clothing dataset along with their performance comparison against different optimizers, pooling function, activation function or using augmentation and dropout or not.

Methods: The methods used here are, the *LeNet-5* CNN. Here image classification is performed on Fashion-mnist, clothing dataset using CNN with different optimizers, pooling function, activation function or using augmentation and dropout or not. The performance of the working of CNN in classifying images from fashion-mnist dataset is compared against:

- Different optimizers namely stochastic gradient Descent, RMS prop and Adam optimizer.
- Different activation functions namely Sigmoid, Tanh, and ReLU.
- Different pooling function namely maximum and average pooling.
- Dropout with probability=0.5 vs not using dropout.
- Data augmentation vs not applying data augmentation.

Keywords: CNN (ConvolutionNeural Networks); Optimization; SGD (Stochastic Gradient Descent); RmsProp (Root mean Square propagation); Adam (Adaptive moment estimation); Activation function; Tanh (hyperbolic tangent); ReLU (Rectified Linear Unit)

Introduction

Deep learning (DL) is a hierarchical structure network which through simulates the human brain's structure to extract the internal and external input data's features. Over the past few years, neural networks have been used in many tasks in computer vision, medical diagnosis, etc. Neural networks are designed to handle variety in the input data such that it can classify that variety in a generic way. Artificial neural networks do not have a concept of filters, or pooling, unlike CNN. The number of parameters that are supposed to be trained and altered in backpropagation to reduce the cost function is very large in number. This goes beyond the memory of our normal system as it slows down training the model. Secondly, training too much of neurons and more parameters also means overfitting, thus it can also affect the performance of our model. Another benefit of CNN is that they can capture or are able to learn relevant features from an image at different levels (since we use filters) similar to the human brain, a concept called feature learning. These details are explained by implementing image classification on a clothing dataset, Fashion-mnist. We used LeNet-5 architecture as the base model for our training, then we made modifications on the model in different ways and we calculated the accuracy each time to choose the best model parameters.

Methodology

Images from the Fashion-mnist dataset are taken in which 60,000 are considered as training samples and 10,000 as test samples, images of size 28*28*1. CNN is designed with two convolution layers of 6 and 16 filters respectively, different activation functions, pooling functions, and optimizers are considered besides dropout and augmentation to check the performance. Dataset considered, architecture along with trainable parameters are presented clearly in the next sections accordingly.

Dataset

The fashion MNIST dataset of clothing article images is the most easily available and a convenient way to consider and work with. Considering this as a base dataset for training/building the models for predictions (LeNet-5 CNN in this study), makes it much easier to implement and understand the diversified concepts of classification and prediction algorithms. Fashion MNIST dataset consists of 60000 training set examples and a test set of 10000 examples where each sample is a 28 x 28 grayscale image associated with a label of 10 classes. There are in total 785 columns the very first column being the class label to represent the article of clothing. In the study proposed here, the architectures are built for image classification using CNN, and the performance is assessed against different modifications for image classification.

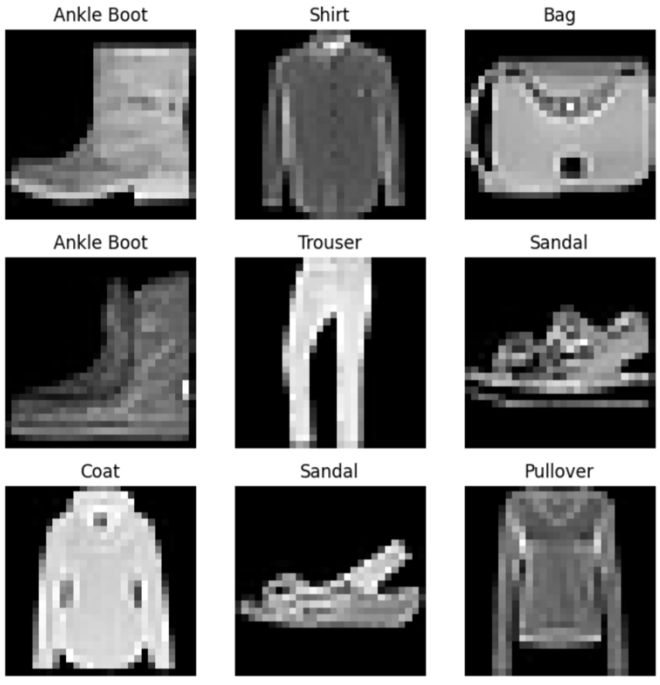

Figure 1. Fashion-MNIST Dataset

LeNet-5 CNN

LeNet-5 CNN is a sequential model consisting of a stack of layers where a lot of computation is done at each layer to figure out the most prominent features as we move deep into the network. We used the standard LeNet-5 architecture as the base model. The input images of size 28 x 28 x 1 are convolved by applying the 6 filters/ Kernel of size 5 x5 and padding of two to convert the size of the image to 32*32. The activation function used here is Tanh. The avg pooling operation is done by considering a kernel size of 2 x 2. This forms the first convolution layer. The second convolution layer consists of 16 filters of size 5 X 5, Tanh as the activation function, and avg pooling of 2 x 2. After having two consecutive convolutional layers, the next Layer is the first fully connected layer with 16*5*5 input features and 120 output features, the activation function used here is Tanh. The second fully connected layer with 120 input features and 84 output features, the activation function used here is Tanh. The output layer is simply a dense network of 84 input features and 10 output features which is the number of the classes. These models are compiled and trained for classification against different activation functions like Tanh, Sigmoid, and ReLU, different optimizers like SGD, Adam, RMS-prop, and Adam, different pooling functions and the use of dropout and augmentation or not to reduce the overall loss and make the models better at their predictions.

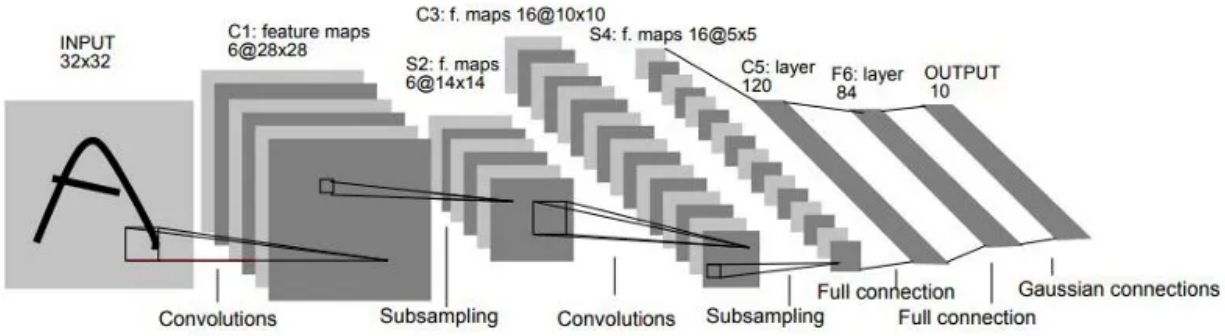

Figure 2. LeNet-5 Architecture

Performance Metrics

Focus on the accuracy of the training and validation data, training duration, and rate of model convergence.

Experimental Setup

Experiment 1

Apply different activation functions like Tanh, Sigmoid, and ReLU with other parameters held constant.

Experiment 2

Explore different pooling techniques while maintaining consistent activation and optimization settings.

Experiment 3

Evaluate various optimization algorithms like SGD, Adam, and RmsProp.

Experiment 4

Determine the effects of introducing dropout layers.

Experiment 5

Determine the effects of applying augmentation.

Results

The best model used in the notebook achieved an accuracy of 0.93868 on the training set, 0.91822 on the validation set, and 0.90944 on the test set. This model incorporated data augmentation and utilized the SGD optimizer. It featured ReLU activation functions, max pooling, and consisted of two layers with the following specifications: input channels = 1, output channels = 6, kernel size = 5, stride = 1, and padding = 2.

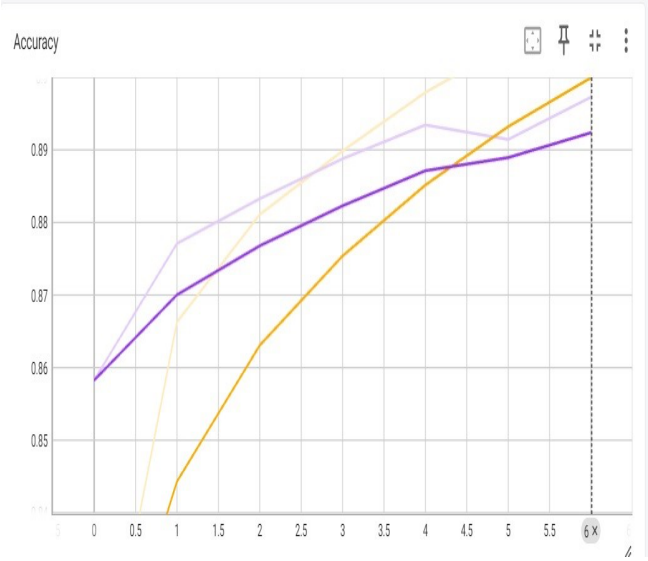

Figure 3. Training vs. Validation accuracy for the best model

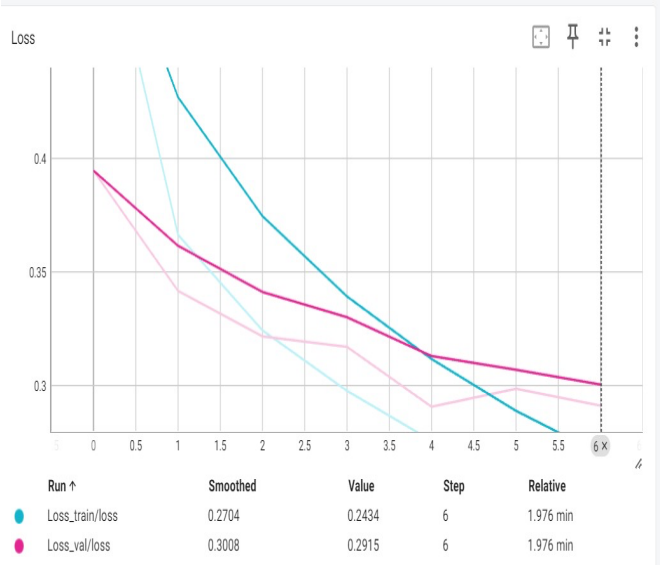

Figure 4. Training vs. Validation loss for the best model

The worst model used in the notebook achieved an accuracy of 0.78103 on the training set, 0.79422 on the validation set, and 0.77546 on the test set. This model incorporated data augmentation and utilized the SGD optimizer. It featured Tanh activation functions and max pooling.

Figure 5 Training vs. Validation Accuracy for the worst model

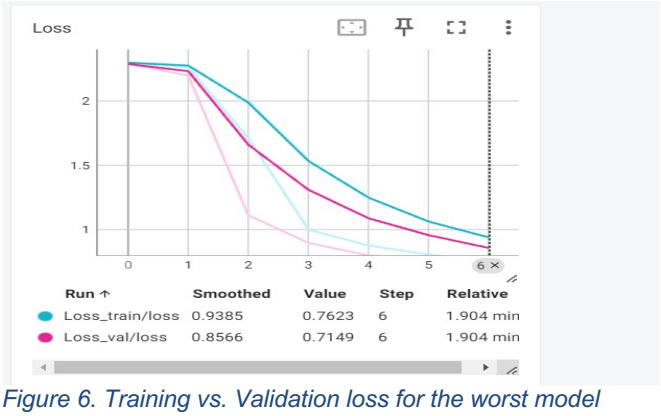

Figure 6. Training vs. Validation loss for the worst model

Analysis

we evaluated the performance of various models on our dataset, focusing on key metrics such as training accuracy, validation accuracy, and test accuracy. We applied data augmentation and used the SGD optimizer across all models, with variations in activation functions and network architectures.

Best Model Analysis

The best performing model achieved an impressive training accuracy of 0.93868, a validation accuracy of 0.91822, and a test accuracy of 0.90944. This model's architecture included the following specifications:

- Two layers with input channels = 1, output channels = 6, kernel size = 5, stride = 1, and padding = 2.
- Utilized ReLU activation functions, which are known for their efficiency in training deep neural networks by mitigating the vanishing gradient problem.
- Implemented max pooling to reduce the spatial dimensions of the output volume, thereby reducing the number of parameters and computational complexity.

The high accuracy across training, validation, and test sets indicates that this model generalizes well to unseen data, suggesting an effective learning of the underlying patterns in the dataset.

Worst Model Analysis

On the other hand, the worst performing model recorded a training accuracy of 0.78103, a validation accuracy of 0.79422, and a test accuracy of 0.77546. Key characteristics of this model included:

- Utilization of Tanh activation functions, which can suffer from the vanishing gradient problem, potentially leading to slower convergence and less effective learning.
- Implementation of max pooling, similar to the best model, to down-sample the spatial dimensions.

The comparatively lower accuracy across all sets for this model highlights its limited ability to learn and generalize from the training data. The choice of Tanh activation function might have contributed to its poorer performance due to its inherent limitations in handling gradient-based learning in deeper networks.

Conclusion

The research confirms that careful optimization of the LeNet-5 parameters—particularly the activation functions, pooling techniques, optimization algorithms, and dropout settings—can lead to marked improvements in model performance. Further investigations should aim to extend these results to more complex datasets and practical applications.

References

[1] "Stanford Vision and Learning Lab (SVL)," [Online]. Available: http://vision.stanford.edu/cs598_spring07/papers/Lecun98.pdf.

[2] "Indian Journal of Science and Technology," [Online]. Available: https://indjst.org/articles/implementation-of-cnn-and-ann-for-fashion-mnist-dataset-using-different-optimizers.

[3] "High performance of optimizers in deep learning for cloth patterns detection," [Online]. Available: https://www.researchgate.net/publication/373581508_High_performance_of_optimizers_in_deep_learning_for_cloth_patterns_detection.

[4] "arXiv," [Online]. Available: https://arxiv.org/abs/2304.11758.