CNN on Fashion MNIST

Dawson Dinh
ddinh7@uic.edu

1 Introduction

Describe the task, motivation, and highlights of your findings.

The dataset I chose to work on is the Fashion MNIST dataset. The goal of my project is to build a machine learning model for image classification using the Fashion MNIST dataset. Image classification is a very important part of today's society, as we use it for things like object recognition, medical diagnosis, social media, and etc. The motivation for using Fashion MNIST is to learn to strength and limitations of different models. Fashion MNIST provides a great benchmark for basic image classification, which can be used by researchers and developers. For this project I use a CNN model and additionally a SVM model to compare and contrast the two. The current state of the art approach to this problem is to use CNN which is popular for image classification problems. I chose to use CNN to better understand the different ways I can better the performance of my model. I also compare the results with SVM to see which provides better performance on this particular dataset.

2 Methods/Case Study

Describe the methods that you have explored or the study that you have conducted. Also describe the baselines that you have used to motivate your approach (if applicable).

For this project I used the Fashion MNIST dataset from Zalando research. I wanted to explore the difference in performance between two models and also find ways to better improve the performance of the CNN model. The main method I used is CNN, or Convolutional Neural Network architecture. I chose this method because it is known to be very effective in image classification tasks. Specifically I used a variant of LeNet architecture. For the CNN, the architecture I used is as follows:

- Convolutional layer with 32 filters, kernel size 3x3, and ReLU activation function
- Max pooling layer with pool size 2x2
- Convolutional layer with 64 filters, kernel size 3x3, and ReLU activation function
- Max pooling layer with pool size 2x2
- Flatten layer to convert the output of the convolutional and pooling layers into a 1D vector

- Fully connected layer with 128 neurons and ReLU activation function
- Dropout layer with a rate of 0.2 to prevent overfitting
- Output layer with 10 neurons (corresponding to the 10 classes in Fashion MNIST) and softmax activation function.

The hyperparameter search space included varying the numbers of filters, filter sizes, number of layers, learning rate, dropout rate, and the batch size. These values varied to provide want I can gather as the best performing model based on accuracy. I used the Adam optimizer for training the CNN model, which from what I can test, performs the best among the other optimizers. During training, I monitored the loss and accuracy on the training set and validation set to ensure that the model was not overfitting. Note: Since there was no validation set provided, I split the training set into training and validation sets. After training the model for CNN, I achieved an accuracy of 91.98%. This means that the model was effective. For SVM i managed to achieve an 86% accuracy. Between the two models, CNN seems to perform better.

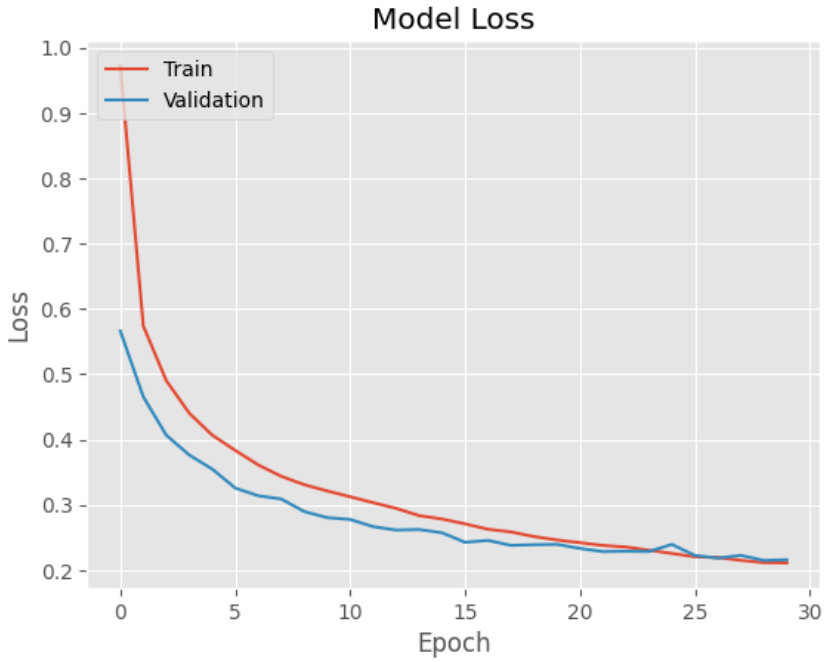

Figure 1: Training/Validation Loss

Here we have the training and validation loss of our CNN model. The training loss decreased as the number of epoch's increased, which should be expected since the model learns to fit the training data better. The validation loss can also be seen to decrease but the numbers seemed to show increase so this could be a sign for overfitting. So to prevent any overfitting, I changed the dropout value until performance was acceptable.

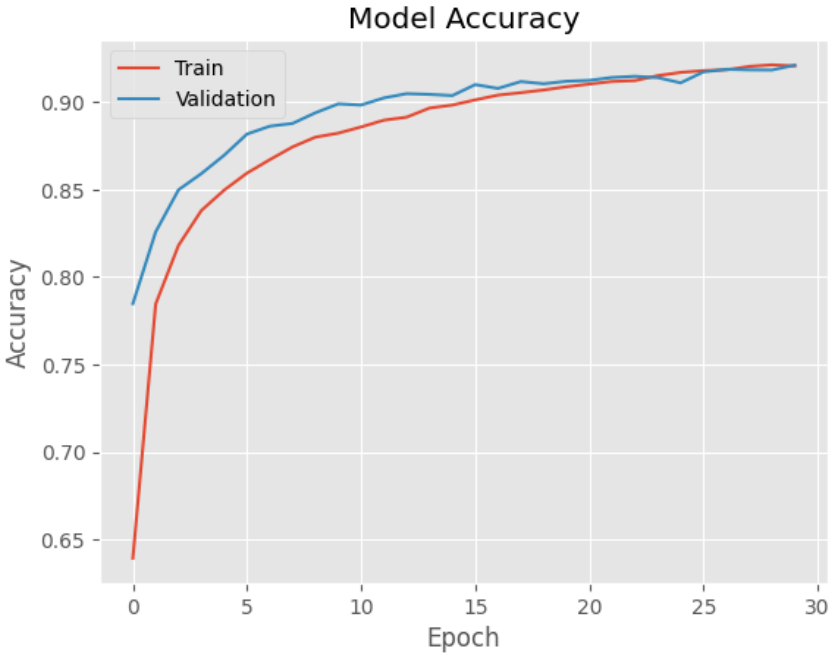

Figure 2: Training/Validation Accuracy

Here we have the accuracy for both training and validation. As we can see, the training accuracy greatly increases in accuracy for the first few epoch's but begins to flatten near the end. What I found interesting was how the validation accuracy began at a different number. We can see how both training and validation seem to almost converge at 91.98%. One thing did to try to improve performance is to decrease the amount of convolutional layers I had. Increasing layers did not manage to improve performance, in fact it caused overfitting. I managed to obtain a higher accuracy when using only one layer. I achieved an accuracy of 92.7%. The graph for training with one layer is provided at the very bottom. This means that the less convolutional layers, the better performance the model should achieve. The amount of layers needed should depend on the data being used and the architecture. With this information, I can conclude that with Fashion MNIST, more convolutional layers would lead to overfitting, when compared to a single layer.

When using the SVM model, I could not seem to achieve an accuracy higher then 86%. This makes sense since SVM's are not known to be very effective in capturing complex features in images. SVM's also seemed to be more time consuming compared to CNN's. SVM can work well with smaller datasets with fewer classes and smaller image sizes.

3 Results and Discussion

Discuss your findings and justify the observed outcome. You can discuss your other relevant observation.

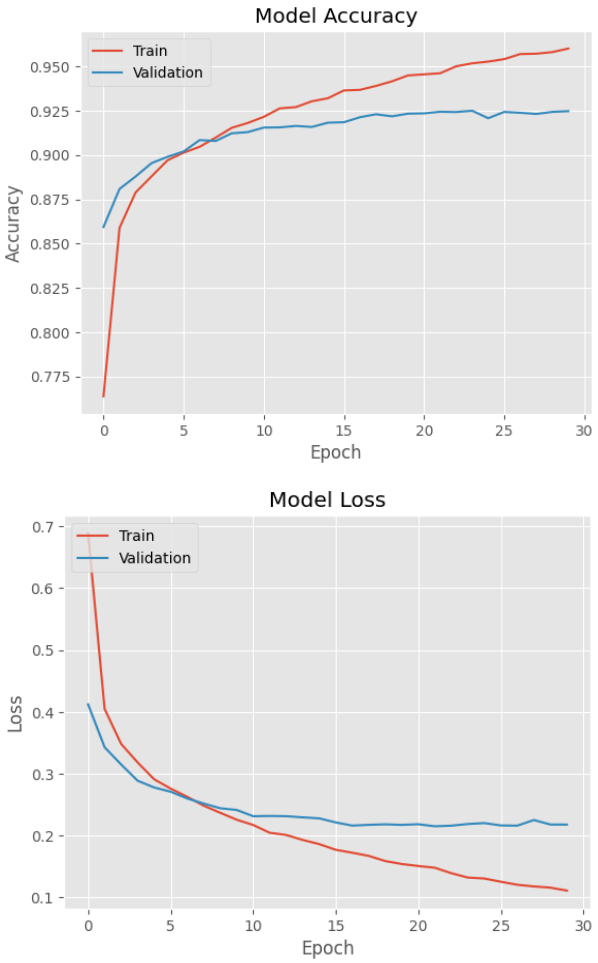

Figure 3: Graph for one conv layer

Based on the experiments conducted, the CNN model performed better than the SVM model in classifying the Fashion MNIST dataset. The CNN model while using three Convolutional layers, was able to achieve an accuracy of 91.98%. One way I was able to increase this number (no matter how small) was by decreasing the amount of layers I used. The CNN model was able to achieve a test accuracy of 92.7% with one layer. Increasing layers ended up causing overfitting and less accuracy. In addition, the SVM model achieved a test accuracy of 86%. This result is expected since CNN's are well-suited for image classification tasks and are able to learn spatial features effectively, while SVM's are known to work well in simpler classification tasks. I can also conclude that using less convolutional layers will improve the performance, as shown in figure 3.

Further experiments could be conducted to improve the performance of the CNN model. One approach could be by utilizing pre-trained models such as ResNet on the Fashion MNIST dataset. Additionally, the model's hyperparameters such as the learning rate and batch size

could be fine-tuned using a more thorough grid search approach. I did not end up getting better results while changing those parameters, but that could also be an error on my code.

4 Conclusion

Discuss what you done briefly and what you have learned from this project.

In this project, I explored the Fashion MNIST dataset and used CNN to perform image classification. I learned the different ways hyperparameters can affect overall performance of the model and used that to try and determine to best variant of the CNN model that could achieve the higher accuracy. I also did a small evaluation between CNN and SVM and determined that when it comes to image classification, CNN is better suited. Overall, this project allowed me to gain hands one experience with image classification using CNN and a completely new dataset. This project taught me the importance of tuning hyperparameters and using tools to visualize model performance. The findings highlight the effectiveness of CNNs in image classification tasks and the importance of proper model selection and hyperparameter tuning.

Reference

- Research, Zalando. "Fashion Mnist." Kaggle, 7 Dec. 2017, https://www.kaggle.com/datasets/zalando-research/fashionmnist.
- SVM and Kernel Methods lecture slides/lecture recording
- Team, Keras. "Keras Documentation: The Sequential Class." Keras, https://keras.io/api/models/
- Brownlee, Jason. "How Do Convolutional Layers Work in Deep Learning Neural Networks?" MachineLearningMastery.com, 16 Apr. 2020, https://machinelearningmastery.com/convolutional-layers-for-deep-learning-neural-networks/.
- My past homework assignments in this class