FashionMNIST Classification With Convolutional Neural Networks

Chris DeVoe, Joey Gray, Danada Amalage Don

A large part of the Artificial intelligence and Machine learning fields are image classification. Not only within research but in the industry side as well, many computer scientists have been working with image classification to try and create models for object detection. With this idea of machine learning classification in mind our group aims to train an image classification model to help in object detection. Our model will be able to look at simple images of clothes using the fashionMNIST dataset and be able to output what the model believes to be within the image. We aim to show the accuracy of our model by using validation set to use after we have trained our model properly on our training data.

We are implementing a convolutional neural network to classify images containing different types of clothes. This is similar to the ImageNet classification problem shown in class, except we will be using a different dataset that is more focused on our objective. The motivation of this project is to test ourselves to apply the knowledge we learned in this class to create a model that can show high accuracy on the provided dataset.

Our question is to find the best model using methods that we have covered in class as well as our knowledge of PyTorch. This will be an experimental question, as there are many different layers and functions we can experiment with. Our solution will involve a CNN since that is the gold standard for image classification and uses PyTorch as our main library that we are building our model on. However, other things such as the layers around it, loss functions, and optimizers can create many different possible arrangements that can affect our score.

After downloading the dataset, we randomly split the dataset into training, validation, and testing splits with ratios 8:1:1 respectively. We put these splits into data loaders with a batch size of 200. The model is a dense convolutional neural network, inspired by DenseNet [5]. However, the dense CNN in that paper was much more complicated, as it is designed for more complex datasets such as CIFAR and ImageNet. We will train our model on the training dataset through a maximum of 20 epochs. Our convergence criteria would be when the training accuracy begins to decrease, but we hit the maximum epochs we set before that occurred. The parameters were set using cross entropy loss and PyTorch’s Adam optimizer. For a baseline, we used a most frequent class classifier as introduced in Assignment 2. With each test if our accuracy is not at a high enough standard we then go back and update our models layers until we can get our accuracy score higher. Once our model has been set properly, then we will test on actual test data, hopefully we shall achieve a better accuracy than initial test data and our model for image classification should be successful.

CNNs have already been widely used for image classification. One of the most recent papers involving the dataset we plan to use implements CNN to achieve an impressive 98% accuracy [1]. Another recent model using multiple CNNs on the dataset that we chose to use received an accuracy of 94% [3]. This similarly to the previous mentioned paper helps us set marks for what we want to aim for the accuracy metrics even though we are not expecting our results to be this good, however this study will give us a good direction in where to start.

To continue looking into previous works another paper from 2020 used CNNs as well to classify the FashionMNIST dataset helping to show even compared to the previous papers, that a 99% accuracy was possible with a CNN compared to the original paper this paper refers to that uses an SVM and got a 89% accuracy [4].This helps to show not to use SVM and that CNNs are the correct direction to go, giving us a great starting point to build our project on.

One block that is anticipated in experiencing will be in making sure our model will show good accuracy for the fashion image data set that we will be using. This will be a block in our project as we will need to update the weights, any biases and keep playing around with our settings until our model shows the accuracy that we are wanting to show. With this in mind we do need to take in account for recall and precision too, make sure they are all in the places we need it and to make sure the one we consider to be the most important, to be the highest. Also taking into account other research in the past, we can learn from any of their mistakes and use that information to better our own formula [2].

During the testing phase we encountered many obstacles such as taking too long to train data, and poor accuracy. To decrease training duration and increase accuracy we had to experiment a lot with the architecture of our model. Our main goal was to get substantial accuracy from our trained model while not taking too long to train, we experimented with the different number of layers in our model to both manage test duration and accuracy of the results.

We also had to experiment with training our model as well, like with the model architecture, our goal was to get the best accuracy and a less duration in training. We experimented with different hyper parameters, such as changing the number of epochs, adjusting learning rate, and trying different loss functions as well.

Initially when using food 101 dataset to train our model, we were getting extremely low performance when we ran through the test data. We were getting around 3 - 4% initially, but after some changes to the model itself and changing hyperparameters we were able to reach around 10%, which was still a pretty low accuracy. After few experimentation we decide to change our dataset to fashionMNIST which is less complex dataset, and after training process we test it and got around 90% accuracy

One of our major limitations that appeared while we were working on our model was the length it would take our model to run. On our testing nearing the end, we were able to get a model that converges in ~6 minutes on the GPU (~20s per epoch) and ~25 minutes (~1.25 min per epoch) on Google Collabs CPU. But before that we had models that could take upwards to 2 hours to run just a single epoch and us an average accuracy. This problem consisted due to the amount of layers and parameters that were being used within our models. In the end this was a challenge that we were able to work through with updating and tweaking our model so that we kept with good accuracy as well as a good runtime.

Another limitation we ran into was with the first dataset that we chose to use, our first dataset we used was the Food101 model provided by PyTorch. This dataset caused our model to have many issues including getting too complex that we would have some models run for an hour just to get an output, while no matter how complex or detailed we made our model our accuracy metric would never push above around 10%. This was heavily due to our model not being able to properly train the dataset, and no matter what layers we added or transformations we made we were unable to get our accuracy much higher than 10%. With this our group chose to move on from the dataset and move onto another dataset which was the FashionMNIST dataset. With this dataset we were able to update our model’s layers and parameters to start testing and training leading to our final model and results of 90%.

A large follow up that could be done with this project is looking back at where this project started, we did start with using the Food101 dataset provided by PyTorch and then after being unable to get our model to show accuracy metrics above 10% we moved onto trying to work with the FashionMNIST dataset. If we were able to have more time and were able to properly follow up on this project trying to take the knowledge that we learned from working with FashionMNIST and the failed testing of Food101 to try once again at training a model for Food101. This in itself could be an entirely new project, but because it was where this project began it would be a long term follow up goal for this or a future project.

Work Cited

[1]: M. Kayed, A. Anter and H. Mohamed, "Classification of Garments from Fashion MNIST Dataset Using CNN LeNet-5 Architecture," 2020 International Conference on Innovative Trends in Communication and Computer Engineering (ITCE), Aswan, Egypt, 2020, pp. 238-243, doi: 10.1109/ITCE48509.2020.9047776. keywords: {Deep learning;Computational modeling;Neural networks;Computer architecture;Search problems;Market research;Convolutional neural networks;Deep learning architectures;Fashion MNIST;Fashion Classification;Convolutional Neural Network (CNN);LeNet-5},

[2]: A. Şengür, Y. Akbulut and Ü. Budak, "Food Image Classification with Deep Features," 2019 International Artificial Intelligence and Data Processing Symposium (IDAP), Malatya, Turkey, 2019, pp. 1-6, doi: 10.1109/IDAP.2019.8875946. keywords: {Convolutional neural networks (CNN);pre-trained CNN models;deep feature extraction;SVM;food image classification},

[3]: Nocentini, O.; Kim, J.; Bashir, M.Z.; Cavallo, F. Image Classification Using Multiple Convolutional Neural Networks on the Fashion-MNIST Dataset. Sensors 2022, 22, 9544. <https://doi.org/10.3390/s22239544>

[4]: LEITHARDT, VALDERI. "Classifying garments from fashion-MNIST dataset through CNNs." Advances in Science, Technology and Engineering Systems Journal 6.1 (2021): 989-994.

[5]: Huang, Gao, et al. "Densely connected convolutional networks." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.