ECE 457B/657A : Winter 2025
University of Waterloo ECE

Assignment 3
Clustering and Classification with Deep Learning on "FashionMNIST with a Twist!"

Due: April 4, 2025 at 11:59pm EST

Overview

Broad objectives:

- Set up and train two types of Convolutional Neural Networks (one standard, one your own choice) using a modern python neural network library for classification on the FashionMNIST dataset.
- Produce a short, concise, and informative report on your design, experiments, results and insights. The focus is on design of your model and results, rather than data preprocessing or feature engineering for this assignment.

Specific objectives:

- Practice how to apply the methods discussed in class.
- Demonstrate understanding by making well reasoned design choices, and *explaining* any result you obtain in straightforward, short, text. Remember, there isn't always one particular right answer to how to do something, but you must justify what you did and demonstrate how it worked.
- Experiment with how to use different methods of feature extraction, parameter tuning, analysis and visualization to improve the performance of your model and your understanding of its results.

Collaboration/Groups:

Submission: Hand in one report per person, or group, to Crowdmark and LEARN. Your report should be submitted as two files:

- *To your group dropbox on LEARN:* Your code in the form of a jupyter notebook that has the code, and results already generated on the provided data in a readable way.
- *To Crowdmark:* Your report in the form of a pdf that is *no more than about 10 pages* ***Update:*** *Originally this said 20 pages. This limit is not that strict, but try to keep it concise.* You can use the jupyter notebook to generate this report document, but you need to **remove all the code and preliminary processing** from it. Only keep what is actually essential to explain what you did, why you did it and what your results show. It might be easier to copy the text, headings, plots and tables out into a separate document and save that.

Your group on LEARN has an associated **dropbox**.

Presentation of Results (Descriptive, Concise, Clear)

- **Report document:** All tasks and questions below should be carried out on just the provided dataset given as `csv's` on LEARN. ***Update:*** *Note that previously the assignment incorreclty implied you had to compare to "both" the original FashionMNIST dateset and our modified one, that was not the intention.* The report is a pdf document with your answers for the questions, description of your process, plots and tables for results. ***It does not contain code!***
- **Essential Plots and Tables:** Along the way, minimize the number of different plots shown to the *essentials* for the reader to understand what you did.
- **Code:** You must submit your code, in a self-contained python jupyter notebook to LEARN. You should collapse most of your code before printing to improve readability. You only need to show critical code which relates to the central task of the question or a point you are highlighting in your text.

Tools: You can use any libraries available in python, tensorflow, pytorch, keras, scikitlearn for this project. You need to mention explicitly which libraries you are using, any blogs or papers you used to figure out how to carry out your calculations.

Data Set - Fashion MNIST . . . with a Twist!

This is an image dataset based on publicly available dataset Fashion MNIST:

`https://github.com/zalandoresearch/fashion-mnist`

As shown in Fig. 1 The dataset is composed of small (28x28 pixels), grey-scale images of clothing items such as shoes, coats, pants, etc. It was created to serve as a direct drop-in replacement for the well known MNIST dataset (`http://yann.lecun.com/exdb/mnist/`) dataset for benchmarking machine learning algorithms. So the dataset shares the same image size and structure as the original MNIST.

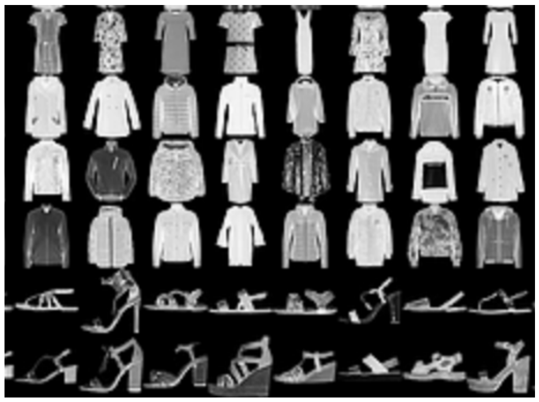

Figure 1: Small sample of greyscale elements of the FashionMNIST dataset.

The Twist...

The label in FashionMNIST was originally an assigned clothing type represented by an integer from 0-9. The training set contains 60,000 examples and the test set 10,000 examples.

For this assignment, we are providing you the same input features, but the **labels** you will be using will be a new ***mystery label***. We have calculated a new label based on the data and associated it with each entry. The `y_train` and `y_test` files provide this modified label which is a category numbered 0-4.

You primary goal is to train a deep learning model that accurately classifies these labels, and then to use an encoding from that model for a secondary task.

What does the mystery label number mean? *That's for you to figure out through your analysis!* While it doesn't matter if you get the interpretation of these labels right *exactly*, your analysis of it should make sense and follows from your results.

File Descriptions:

- `x_train.csv` - the training set for your model. The training file contains vectors of size 784 representing pixel values of a 28x28 image.
- `y_train.csv` - is the *mystery label*, a numeric target obtained using our formulation.
- `x_test.csv` - are the features for the test set to use for final evaluation.
- `y_test.csv` - the *mystery label* for the test data.

Classification with Convolutional Neural Networks

Q1: Default Network

Classify the data using a Convolutional Neural Network with the following architecture:

- input features x
- convolutional layer with 32, 3x3 filters, stride 1, padding 1x1
- max pooling layer 2,2
- convolutional layer with 32, 3x3 filters, stride 1, padding 1x1
- flatten the output
- output layer: one dense (fully connected) layer 512 nodes
- a softmax layer with five output values to transform the outputs into a multi-class probability distribution for classification
- activation functions: internal layers all use ReLU activation
- optimizer : Stochastic Gradient Descent
- loss function : use `categorical_crossentropy` as the loss function calculated using accuracy as the metric

Include a short printout of the summary of the model from your code. Present some runtime information from training of this model. Make sure the code for defining this model is available and clearly identified in your submission on LEARN.

Q2: Your Own Improvements

Improve the results of your model using any other architecture that you want to try. For example, you may want to put more fully connected layers after the last output of the CNN, or try more CNN layers, a different optimizer, regularization approaches such as dropout, or different parameters for the existing layers.

You could also try moderate improvements to architecture and perform data augmentation to increase the amount of data you have for training.

- You can use any CNN-based approach or other DNN variants to solve the classification problem. You can explore any architecture of the network you like. Be sure to cite any sources you used.
- Whatever you try, be sure to provide a concise but clear explanation of your model(s) (algorithms, network architecture, optimizers, regularization, design choices, numbers of parameters) that would be sufficient for someone to recreate it. Consider making a diagram to clarify the design.
- Include a short printout of the summary of the model from your code. Present some runtime information from training of this model.
- Make sure the code for defining this model is available and clearly identified in your submission on LEARN.

Note: All models for Q1 and Q2 need to be trained from scratch! You can't use a pre-trained image processing model for this assignment. If the computational wait time is too heavy, reduce the size of the network and explain, I don't mind changes to the default network if you're computation is too slow. Consider using online computational resources such as Google Colab, Google Cloud, Kaggle or Microsoft Azure. Most platforms free credits on initial signup.

Q3: Results Analysis

Briefly report and comment on the following:

- Runtime performance for training and testing.
- Comparison of the different hyper-parameters or designs you tried.
- Produce a plot showing the **training loss vs. training epoch** for the training data and the validation data on the same plot. Do this for each model.
- Produce another plot showing **classification accuracy vs. training epoch** for the training data and the validation data on the same plot. Do this for each model.
- You can use any additional plots or tables to explain the performance of your approach.

Q4: Using Your Own Encoding

One of the exciting things about deep learning is that the learned model is not just a black box producing predictions, or classification mappings. Your network is a huge, multi-layered machine turning the input data into a series of complex encodings. The `softmax` layer at the end of your network reads off one interpretation of these encodings to give you a classifier. Another thing you could do is take the last fully connected layer (or any layer you choose actually) and use that as a compact representation of the data, just as we treated the output of PCA and LDA.

In this part, you should define a simple intermediate layer model using one of the later internal layers ***from your trained network*** (ie. anything before softmax is possible, but some will work better than others). This is your **encoding**. Then you will treat the elements of this encoding as features in a dataset for clustering and visualization to help you understand what the *mystery label* for our dataset might mean.

Carry out the following activities:

- Visualize your encoding with the first two components from PCA, the colour mapping could be the label values.
- Perform some form of clustering algorithm (eg. DBSCAN, K-means) on the features that you have extracted from your own designed model with **5 clusters** (if the algorithm takes a parameter). Visualize and assess the results. You could, for example, use the resulting clusters as alternate colour mappings for the PCA plot above.
- Come up with some other task to try for using this encoding in a creative way. (eg. you could apply t-SNE on the features that you have extracted)
- Based on all these results of clustering can you guess what are the labels for the given dataset? It might also help to list out a random selection of data entries (the original images) for each cluster and their label value to help understand the patterns each cluster might represent.

All of these activities should be carried out by applying your *already trained network* to the test data and using the *output* of your designated encoding layer *as your features.*