Assignment 3

Deadline: Sunday 22 Mar 2020, 9pm, to be submitted on Brightspace.

Individual and Group Submission: Part 1 of this assignment must be submitted individually, each person submitting a single colab notebook. However, Part 2 of this assignment may be done in groups of 2, with a single colab notebook (different from the Part 1 notebook) to be submitted by just 1 of the group members. In this case, please make sure that the group notebook contains the names of both group members.

A3 Discussion document: As usual, we encourage online discussion about the assignment. The google doc for this is available here.

Introduction

In this assignment, you will gain a hands-on experience using the `PyTorch` toolkit. You need to design different variants of convolutional neural networks using PyTorch, principal component analysis (PCA) for feature selection and compare it with linear and non-linear autoencoders.

Questions

[4 pts total] Q1. Neural Networks in PyTorch

Since we are providing a lot of sample code for this question that can be worked on during lab, the question is provided in a shared colab notebook A3_Part1.ipynb.

[11 pts total (9 pts for Undergrads)] Q2. PCA, linear and non-linear autoencoders

PCA and autoencoders are frequently used for visualization and dimensionality reduction in machine learning. In this question, you will perform PCA on the Iris dataset for visualization, and on MNIST dataset for feature selection and compare its performance to linear and non-linear autoencoders.

a) **[2 pts] PCA for Data visualization:** Use sklearn's PCA for visualizing the features of the Iris dataset (refer this notebook for details about Iris). Display the Visualizations of different classes, with different colors, obtained using (i) first two principal components, (ii) last two principal components. Explain your observations from the two visualizations. Also, print the variance (information) contained by each principal component.

b) **[3 pts] PCA for feature selection:** Use PCA on MNIST dataset to select features by retaining the following levels of variance (information): $50\%, 75\%, 90\%, 95\%$. For each of this variance level, you need to (i) use the retained features to train a simple feed-forward neural network with one hidden layer to classify the images, (ii) reconstruct the 28×28 images from the retained features (obtained using PCA with different variance levels) and train the CNN network defined in **Q1** to classify these images. Plot the performances (in terms of accuracy) for parts (i) and (ii). Explain your observations from parts (i) and (ii) in terms of performance.

c) **[3 pts] Linear autoencoders for dimensionality reduction:** Design a linear autoencoder (layers with linear activation function) where the encoder and the decoder consists of a single hidden layer and the size of the bottleneck layer equal to the number of features retained in b(i). (i) Using this compressed representation obtained using autoencoders, train the same feed-forward network as in b(i) and display the performances. (ii) Repeat b(ii) by using the reconstructed images obtained using the decoder of the autoencoder. Plot the performances (in terms of accuracy) for parts (i) and (ii). Explain your observations by comparing the performance of autoencoders with respect to that of PCA.

d) **[1 pts] Non-linear autoencoders for dimensionality reduction:** Repeat Q 2(c) by using non-linear activation function for the hidden layers of the encoder and decoder of the autoencoder.

e) ***Only for grad. students*** **[2 pts] De-noising autoencoders:** Using the reconstructed images, obtained when variance = 90%, train a denoising autoencoder. Where the input to the denoising autoencoder will be the noisy images (i.e., the reconstructed images) and the output of the decoder need to be the noiseless image (original image). Use the denoised outputs of the denoising auto-encoder to train the CNN and report the observations in terms of performance when compared to the performance of CNNs trained using reconstructed images with variance equal to 50%

General Marking Notes and Tips

- You will be marked for accuracy, correctness and also clarity of presentation where relevant.
- Be selective in the experimental results you present. You don't need to present every single experiment you carried out. It is best to summarize some of your work and present only the interesting results, e.g. where the behaviour of the ML model behaves differently.
- In your answers, you want to demonstrate skill in using any required libraries to experiment with ML models, and good understanding / interpretation of the results. *Make it easy for the marker to see your skill and your understanding.*
- Justify your meta-parameter choices where appropriate.
- Except where you are explicitly asked to implement a particular algorithm/step yourself, if there is a procedure in sklearn that does the task, you are free to find it, read its documentation and use it. If you are not sure, it is best to ask on the shared notes or in class!
- Liberally add markdown cells to explain your code and experimental results. Make good use of the formatting capabilities of markdown to make your notebook highly readable.
- Add links to all online resources you use (markdown syntax is: [URL](anchor_text)). Avoid explicit long URLs in your markdown cells.