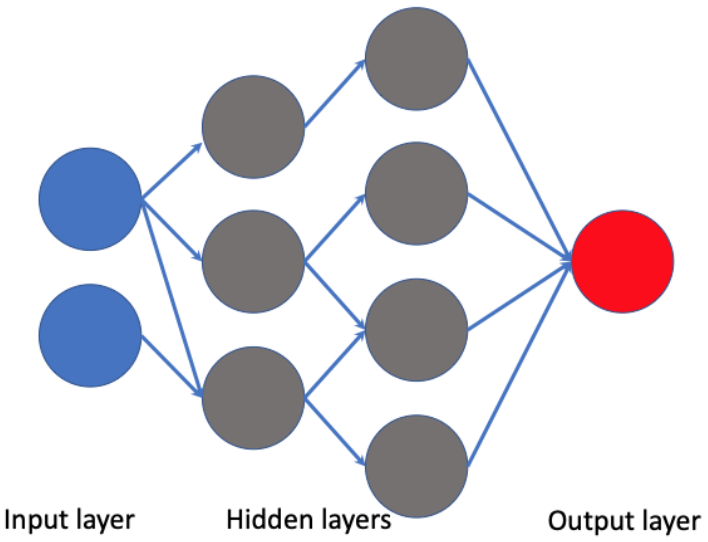
Feed-forward Neural Networks



The world runs on a wide variety of neural networks. In this assignment, you will play with the mother of them all, feed-forward neural networks (FNN). Theoretically, you can learn any linearly separable function with FNNs. But, have you wondered why we always have to invent newer and newer architectures?

However, in this assignment, you will build an FNN from scratch and apply your FNN to classify letters.

Basic Components

- [ ] Dense Layer: a fully connected layer, defined by the dimensions of its input and output.
- [ ] ReLU activation Layer
- [ ] Dropout Layer: check here for details.
- [ ] Softmax Layer: check here for details.

Write separate classes for each of the aforementioned building blocks. Vectorize your code whenever possible to speed up training and inference. Modularize your code well, set up the architecture in one place such that it is trivial to change the model architecture later on (for possible online and retraining).

You will have to implement the backpropagation algorithm to train the model. The weights will be updated using mini-batch gradient descent. No deep learning framework is allowed for your implementation. Since the architecture is not fixed, you have to modularize your code in such a way that it works for any architecture that uses the aforementioned modules. To make your implementation efficient, you have to write each operation as matrix multiplication, whenever possible.

For preparing and training your model, write your code in a file named as **train_<YourRollNo>.py**.

Dataset Description

You will use the EMNIST letters dataset. You can use the following code snippet to download them:

```
import torchvision.datasets as ds

train_validation_dataset = ds.EMNIST(root='./data', split='letters',
                                     train=True,
                                     transform=transforms.ToTensor(),
                                     download=True)

independent_test_dataset = datasets.dsets.EMNIST(root='./data',
                                    split='letters',
                                    train=False,
                                    transform=transforms.ToTensor())
```

The dataset contains 28x28 images of letters from the Latin alphabet. Split the train-validation dataset as 85%-15% to form your train set and validation set.

Preservation of Trained Model

You must save your final model in pickle (see: https://docs.python.org/3/library/pickle.html). You should write a separate python script (named: test_<YourRollNo>.py) that can load the pickle file of your trained model and use it to predict labels for query images (i.e., the images for which classification needs to be done). The testing dataset will also contain 28x28 images and it will be provided during evaluation. Do NOT share your pickle file with others.

Report Writing

- [ ] You have to report the training loss, validation loss, training accuracy, validation accuracy and validation macro-f1 for each full pass over the training set.
- [ ] Prepare graphs for four different learning rates and for three different models.
- [ ] Report the confusion matrix for each such model.
- [ ] Make sure you tune the learning rate (start from 5E-3 and decrease). Select the best model using validation set's macro-f1 and report the values of the above-mentioned scores. Try to have a validation macro-f1 of 0.75 or more.
- [ ] Finally, for the best (chosen) model, report the independent test performance.

Thrive for good results

You should train hard to get the best results, sky's the limit. Do not overfit, however. During online, a separate independent test set will be used to measure the performance of your model. The top three performing models in each section will be duly recognized.

Library usage

- [ ] Reading the images/possible augmentation: opencv, pillow
- [ ] Dataset loading: torchvision
- [ ] Visualization: matplotlib, seaborn
- [ ] Progress bar: tqdm
- [ ] Data manipulation: numpy, pandas
- [ ] Model saving and loading: Pickle
- [ ] Performance metrics and statistics: scipy, sklearn

Marking Rubric

TBD

Submission

1805xyz
|-- train_1805xyz.py
|-- test__1805xyz.py
|-- model_1805xyz.pickle [only for the best performing model]
|-- report_1805xyz.pdf

Zip the folder and rename it to **[Student_ID].zip**

Deadline:

- **22-Dec-2023 (Friday) 10:00 PM.**
- **As a winter vacation gift, the cut-off date in Moodle shall be 03-Jan-2024 (Wednesday) 10:00 PM.**

Honour Code

While you are encouraged to talk to your peers, ask help from teachers, and search relevant resources from the Internet, under no circumstances should you copy code from any source. If found out, you will be penalized with due negative marking. Copying a pickle file from others would result in a harsher punishment.

Version Control

- Version 0 [Dec 12, 2023]: Assignment Published
- Version 1 [Dec 15, 2023]: Moodle cut-off date updated