Tales Araujo Leonidas
Professor Patricia McManus
ITAI 2376: Deep Learning
February 08, 2024

AWS Academy Application of Deep Learning to Text and Image Data Labs 1, 2, and 3: Reflective Journal

Introduction

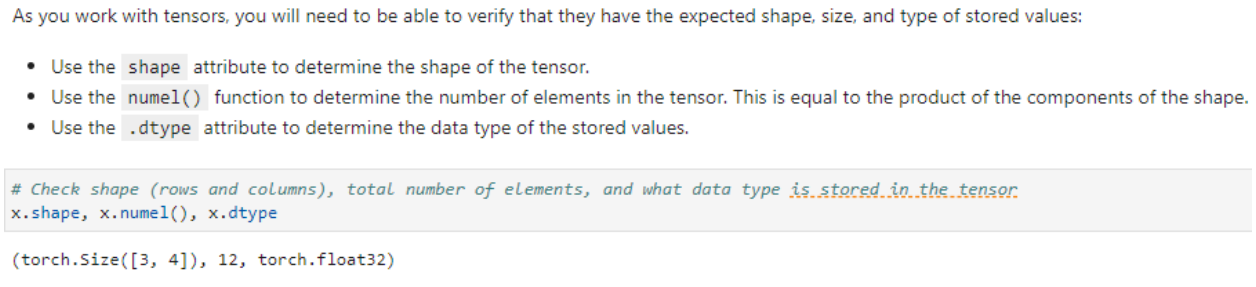
This reflective journal illustrates my hands-on experience with Labs 1, 2, and 3 of AWS Academy Application of Deep Learning to Text and Image Data. The labs delved into Pytorch, a deep learning framework, provided an analysis of a multilayer perceptron (MLP) training process and implementation of dropout layers to prevent overfitting of the neural network, and overviewed the building process of an end-to-end neural network solution to process text data. The goal of this journal is to compile the experience and achievements while reviewing the main learnings.

Activities Performed

All the labs were performed in AWS Academy Canvas using JupyterLab through SageMaker. After installing the initial requirements, I carefully read the markdown cells to understand the Labs' objectives, and run each code cell while reading the comments that explained the Python code. I practiced the knowledge acquired with the quizzes within the Notebook, which is a great memory check resource. Additionally, I searched and reviewed online for any term that I was unfamiliar with.

Results

Using torch, pandas, and numpy, I learned how to manipulate data to be ready for use in model training and performed some Tensor operations before practicing a crucial step in nearly all deep learning optimization algorithms: differentiation.



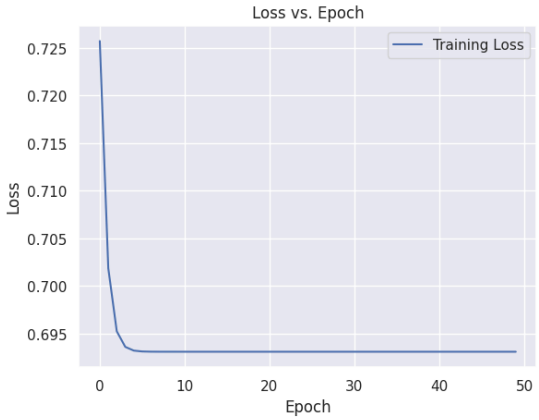
As you work with tensors, you will need to be able to verify that they have the expected shape, size, and type of stored values:

- Use the `shape` attribute to determine the shape of the tensor.
- Use the `numel()` function to determine the number of elements in the tensor. This is equal to the product of the components of the shape.
- Use the `.dtype` attribute to determine the data type of the stored values.

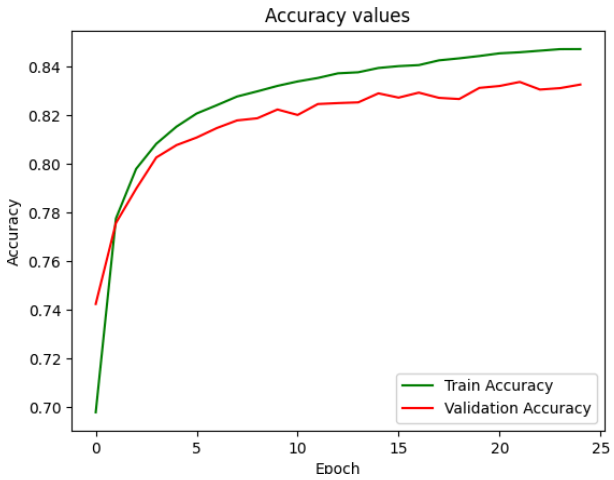
```
# Check shape (rows and columns), total number of elements, and what data type is stored in the tensor
x.shape, x.numel(), x.dtype
```

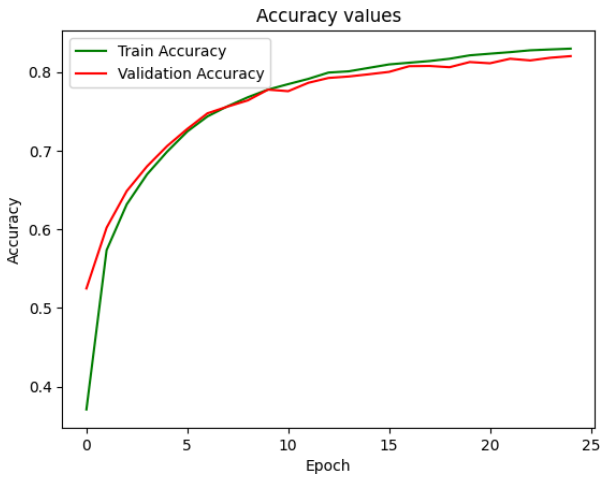
```
(torch.Size([3, 4]), 12, torch.float32)
```

Following the Lab, I was guided in using a neural network with image data to predict the category that an item belonged to. The image on the side pictures the loss function decreasing in response to the training “cycles”. In other words, the more the number of epochs, the less loss or error, which is the goal of training a neural network.

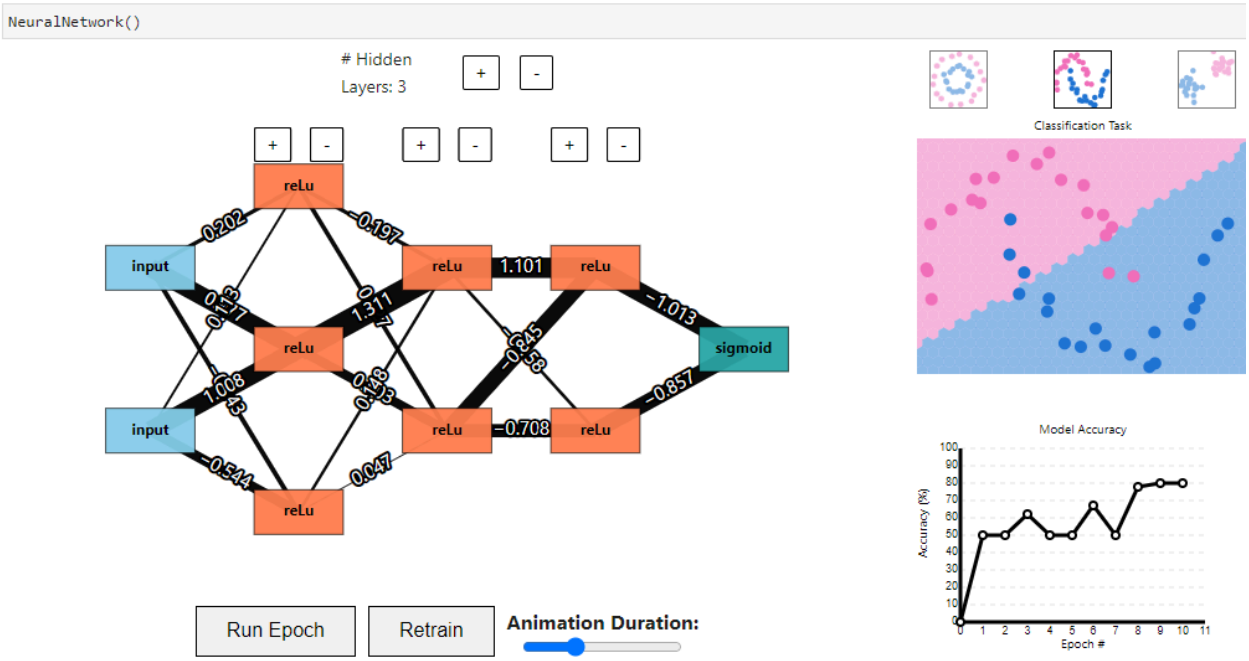


In the comparative analysis of the images from Lab 2 below, it is evident that the introduction of a dropout layer during the neural network's training phase has led to an improvement in the model's performance. By incorporating dropout, the network is compelled to develop a more robust set of features that are not overly dependent on any individual neuron. This technique effectively promotes the creation of a more generalized model that demonstrates enhanced performance, particularly on validation data.





The concluding lab brought together all previously learned data processing techniques to develop a full neural network solution. Displayed below is the output from the interactive 'NeuralNetwork()' function, showcasing the architecture and real-time training progress of the model. The classification task plot and accuracy graph provide a snapshot of the network's performance, improving incrementally with each epoch through backpropagation.



Key Learnings

- “PyTorch has two key features. First, GPU is well-supported to accelerate the computation [...]. Second, the tensor class supports automatic differentiation” (AWS Academy, 2023).
- “To prevent overfitting of neural networks, it's possible to randomly drop a certain percentage of the neurons (or nodes) in the input and hidden layers. This has proven to be an effective technique for regularization and preventing the coadaptation of neurons” (AWS Academy, 2023).
- The overall accuracy and weighted averages in the Lab 3 model indicate a reasonably strong model performance, but there may still be room for improvement, especially in correctly identifying more instances of class 0 (as indicated by the lower recall for class 0).

	precision	recall	f1-score	support
0	0.87	0.70	0.78	6267
1	0.80	0.92	0.85	8053
accuracy			0.82	14320
macro avg	0.83	0.81	0.82	14320
weighted avg	0.83	0.82	0.82	14320

Works cited

AWS Academy. (2023). Application of Deep Learning to Text and Image Data: Module 1. Retrieved from https://awsacademy.instructure.com/login/canvas