Alhassane Samassekou

ITAI2376

Lab 01: Getting Started with PyTorch

This lab was a great introduction to using PyTorch for deep learning. It helped me understand how PyTorch handles tensors, which are essential for working with data in neural networks. I learned how to create, reshape, and manipulate tensors, as well as perform basic mathematical operations on them. It was interesting to see how PyTorch makes it easy to handle multidimensional arrays and perform operations efficiently using GPU acceleration.

One of the key concepts in this lab was indexing and slicing tensors, which is crucial for selecting and modifying specific parts of the data. I also explored different tensor operations, including element-wise arithmetic, dot products, and matrix multiplication. These operations are fundamental in machine learning, especially for training models and processing large datasets.

Another important topic was automatic differentiation, which allows PyTorch to calculate gradients automatically. This is a critical step in optimizing deep learning models because it enables backpropagation, which updates the model's parameters during training. I found it useful to see how PyTorch tracks operations and computes gradients efficiently.

The lab also introduced how PyTorch works with both CPUs and GPUs. It was interesting to see how computations can be accelerated using CUDA, making deep learning training much faster. Understanding how to move tensors between CPU and GPU is an essential skill for working with large-scale machine learning models.

Overall, this lab gave me a solid foundation in using PyTorch for deep learning. It was helpful to practice working with tensors, performing mathematical operations, and understanding automatic differentiation. I now feel more comfortable using PyTorch and look forward to applying these concepts in future labs.

Lab 02: How Neural Networks Learn

This lab was a great introduction to deep learning using the Fashion-MNIST dataset. It was interesting to see how a neural network learns to classify images and how PyTorch makes it easy to load and process data. I enjoyed visualizing the clothing items and understanding how the model started off making completely random predictions before training. This helped me realize that neural networks don’t have any built-in knowledge—they have to learn from the data. Adding a hidden layer and dropout layer was also an important step, as it showed how we can improve accuracy and prevent overfitting by adjusting the model’s architecture.

Throughout the lab, I learned how to load datasets in PyTorch, build a simple neural network using nn.Sequential(), and use cross-entropy loss for classification. Training the model helped me understand how weights are updated using stochastic gradient descent (SGD) and why multiple training epochs are necessary for improving accuracy. I also saw how overfitting happens when a model performs well on training data but struggles with validation data, and how dropout layers help prevent this issue by randomly removing some neurons during training.

Some parts of the lab were challenging, especially understanding how weights and biases are initialized and why the model's predictions were random at first. It also took some time to grasp how optimizers work and how they adjust the model's parameters during training. Interpreting the accuracy and loss graphs required some thought, particularly in understanding why validation accuracy was lower than training accuracy and whether the model was performing well enough.

Overall, this lab helped me understand the basics of neural networks and model training in a hands-on way. It was satisfying to see the model improve over time and to experiment with different settings like dropout and learning rate. The process of training and evaluating a model now makes more sense to me, and I’m excited to build more advanced neural networks in future labs.

Lab 03: First Example of Neural Networks

This lab was both fascinating and a bit challenging, but I learned a lot from it. The idea of applying a neural network to predict pet adoptions from the Austin Animal Center dataset was exciting. It gave me a real-world context to understand how machine learning can solve practical problems. What I found most interesting was how I had to process both text and numerical data before using it in a neural network. It was surprising to realize just how much cleaning and transforming data requires before it's ready to be used.

One of the most important things I learned was the process of cleaning and preparing text data. I had no idea how many steps were involved, from removing punctuation to dealing with stop words and stemming. It was interesting to see how these small details can have a big impact on the final outcome. I also learned about vectorizing text, which was a concept I hadn’t encountered before. Turning words into numbers allows the model to process and understand them, and I could see how that transformed the text data into something the neural network could use.

Another significant part of this lab was learning how to build a neural network using PyTorch. At first, I wasn’t sure how to structure the model or how different layers, activations, and learning rates affected the training process. It was tricky to figure out how to balance the model’s architecture, especially when I had to experiment with different numbers of layers and neurons to see what worked best. Watching the training process and tweaking the model was a great way to understand how deep learning models learn over time, but I had to be patient and persistent, especially when the model didn’t always improve after making changes.

Some parts of the lab were easier, like working with the numerical data and using tools like OneHotEncoder and MinMaxScaler. I could see how important it is to standardize and scale the data for the model to perform well. The hardest part for me was probably getting the neural network architecture right and choosing the best hyperparameters. I had to tweak things like the number of epochs, the batch size, and the learning rate. It was tricky because I didn’t always know how a change would affect the results until I tested it.

Even though there were moments where things didn’t work out as expected, it was rewarding to see my model improve as I adjusted parameters and architecture. It showed me how much trial and error is involved in machine learning, and how even small adjustments can lead to big differences in performance. By the end of the lab, I felt more confident in my understanding of how deep learning models are built and trained.

Overall, this lab was a really valuable learning experience. It gave me a clearer understanding of the steps involved in working with text and image data, and how neural networks process that data to make predictions