Zaid Alayedy

Patricia Mcmanus

ITAI 1378

11 December 2024

Reflective Journal

A Description of the Workshop's Purposes and Methods.

Goals: The ultimate goal of this lab was to gain knowledge about MLPs and their use in image classification. It was about introducing a simple neural network model that could distinguish two visually distinct classes of images. That meant knowing exactly how MLPs work, what the benefits were and what their weaknesses were relative to more advanced systems such as CNNs.

Methods: The lab required GitHub to maintain version control, Google Colab to run code, and PyTorch to compile and train the MLP. This included preprocessing the data, forming the MLP model, training the model, and assessing its performance.

The process also involved using debugging tools and documentation to address coding challenges and enhance the model's accuracy. Additionally, collaborative tools such as GitHub were leveraged to maintain version control and manage code effectively.

Key Concepts Learned:

1- Image Classification: This concept involves understanding how MLPs analyze input features and categorize them into predefined classes. The lab highlighted the importance of feature extraction and how MLPs utilize weights and biases to make predictions.

2- MLP Architecture: The lab emphasized the fundamental components of MLPs, including the input, hidden, and output layers. Activation functions such as ReLU introduced non-linearity into the model, enabling it to handle more complex patterns, while softmax provided the probabilities for class predictions. Insights were gained on how to properly stack layers to balance performance and computational efficiency.

3- Data Preprocessing: Preparing data for training was a critical step. This involved resizing images to maintain uniformity, normalizing pixel values to improve training stability, and ensuring that datasets were properly formatted and labeled. These steps minimized potential errors and enhanced model performance during the training phase.

4- Error Debugging: The importance of identifying and addressing coding errors was a key takeaway. Leveraging resources like PyTorch documentation and online communities provided solutions to common implementation issues.

Challenges Encountered and How You Overcame Them.

One major challenge was ensuring that the MLP architecture was correctly implemented to process image data. Errors initially arose due to mismatched input dimensions, which required resizing and reshaping the image data to fit the network's expected input format. Debugging these issues involved reviewing PyTorch documentation, analyzing error messages, and testing different solutions iteratively. Another challenge was tuning the hyperparameters, such as learning rate, batch size, and the number of hidden units. These adjustments were made through trial-and-error, guided by validation performance metrics. Seeking guidance from tools like ChatGPT and engaging with online forums also played a crucial role in overcoming obstacles. Another significant challenge involved understanding the theoretical underpinnings of MLPs while simultaneously implementing them in code. This dual focus required revisiting foundational concepts in neural networks, such as weight initialization and gradient descent. By breaking the problem into smaller, manageable tasks, I was able to better understand the mechanics of MLPs and their practical applications.

Experiences Gained from Using MLPs in Image Classification

Understanding Basic Neural Networks: Building the MLP provided foundational knowledge of how neural networks operate, from the flow of data through layers to the impact of activation functions and loss calculations. These insights are essential for anyone pursuing advanced studies in machine learning.

Data Representation and Preparation: The importance of properly formatted and labeled datasets was emphasized. Errors during data preprocessing highlighted the critical role this step plays in ensuring smooth model training and testing. Learning to preprocess data effectively is a skill that will be invaluable in future projects.

Iterative Development and Debugging: Training an MLP involves multiple iterations of modifying the architecture, evaluating results, and refining the model to achieve better accuracy. This iterative process not only enhanced my technical skills but also reinforced the importance of patience and persistence in solving complex problems.

Broader Implications of MLPs: The lab also provided insights into how MLPs, despite being simpler than CNNs, are effective for certain tasks. This understanding lays a strong foundation for transitioning to more complex models in future labs.

This hands-on experience reinforced the importance of mastering foundational techniques before advancing to more sophisticated models like CNNs. The challenges and lessons from this lab have prepared me to approach future projects with greater confidence and a more structured methodology.

References:

PyTorch Documentation: https://pytorch.org/docs/stable/index.html

Neural Networks and Deep Learning by Michael Nielsen:

http://neuralnetworksanddeeplearning.com/