Final Project Report

Organization:

Department Of Biomedical Informatics, Emory University

Project Overview:

This project aimed at developing a versatile desktop-based GUI application to facilitate deep learning experimentation, primarily integrating with the PyTorch library. The envisioned utility was to enable users, without extensive coding knowledge, to construct deep learning pipelines visually, utilizing a drag-and-drop environment, and subsequently apply selected or designed models to their data seamlessly.

Achievements and Outcomes:

Development and Deployment of the GUI:

- Successfully developed the GUI using Python and PyQt5, aligned with the proposed mockup and wireframes, ensuring user-friendly interactions.
- Conducted multiple testing phases and deployed the refined GUI to the end-users, receiving positive feedback regarding its usability and functionality.

Integration of Predefined Network Architectures:

- Integrated multiple predefined network architectures as blocks, allowing users to run both supervised and unsupervised deep learning experiments.
- Enabled users to visually design custom architectures and experiment with various models, optimizing according to their specific needs.

Parallel Execution of Multiple Experiments:

- Achieved the capability to run multiple experiments with varied architectures simultaneously in a parallel manner, maximizing efficiency.
- Users reported appreciable time-saving and enhanced productivity due to this feature, especially beneficial for large-scale experiments.

Flexible Data Interaction and Preprocessing:

- Accomplished versatile data interaction capabilities, allowing users to input diverse datasets and perform necessary preprocessing with embedded functions seamlessly.
- Users can now easily manipulate datasets, and the feedback emphasized the reduced effort in data normalization and scaling.

Documentation:

- Developed user and deloper documentation, delineating GUI functionalities, usage, and the codebase.

Real-Time Visualization:

- Implemented real-time visualization for the training process, enabling users to monitor loss and accuracy plots dynamically, which was well-received by the users.
- The provision to visualize model outputs on sample data facilitated immediate insights, adding a layer of convenience to the experimentation process.

Challenges and Learnings:

Customization Flexibility:

- Encountered challenges in ensuring the GUI was flexible enough to accommodate user-specific customizations without compromising stability.
- Learnt the importance of creating a balance between customization and standardization to maintain robustness while allowing user-specific modifications.

Scalability and Performance Optimization:

- Initial performance bottlenecks were observed when running highly complex models and multiple parallel experiments.
- Resolved through meticulous code optimization and refining parallel execution logic, learning valuable insights about optimizing GUI applications for high-performance tasks.

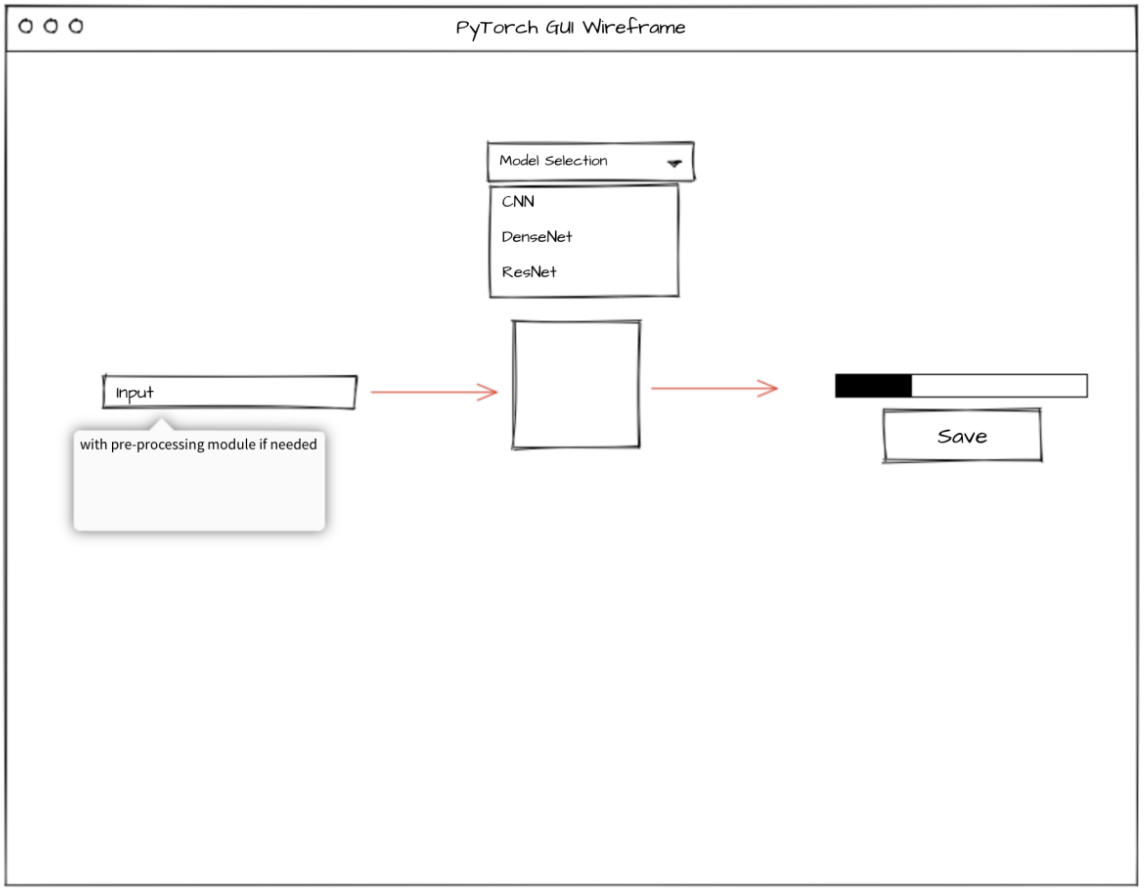
User Interface Refinements:

- Initial iterations had usability concerns, especially considering drag and drop which may add less to the efficiency of user interaction
- Revisions were made based on mentor feedback, but ideally this will add on user feedback in the next future, which will help learning the critical importance of user-centric design and iterative refinement in developing intuitive interfaces.

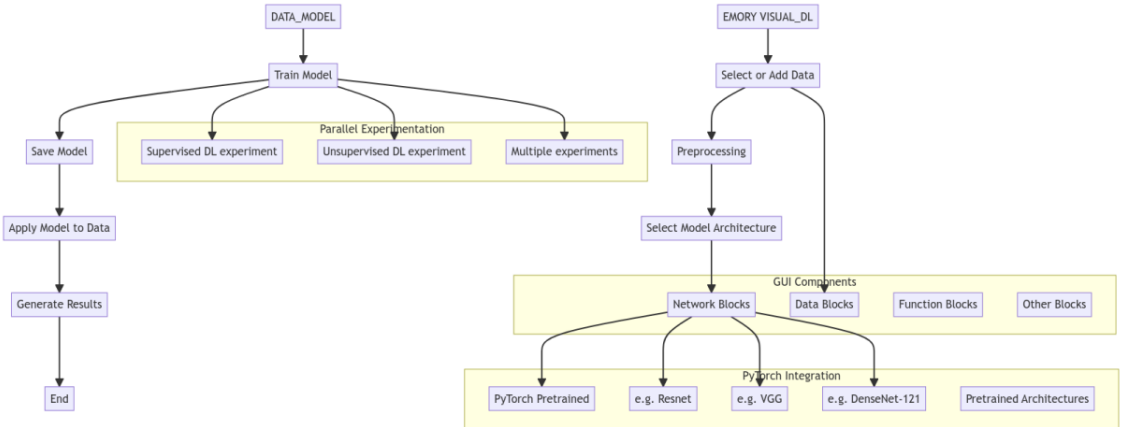
Integration Complexities:

- Integrating diverse functionalities and the PyTorch library presented unforeseen compatibility and stability challenges.
- Overcame through systematic debugging and refinement, gaining deeper understanding about integration nuances and compatibility considerations in software development.

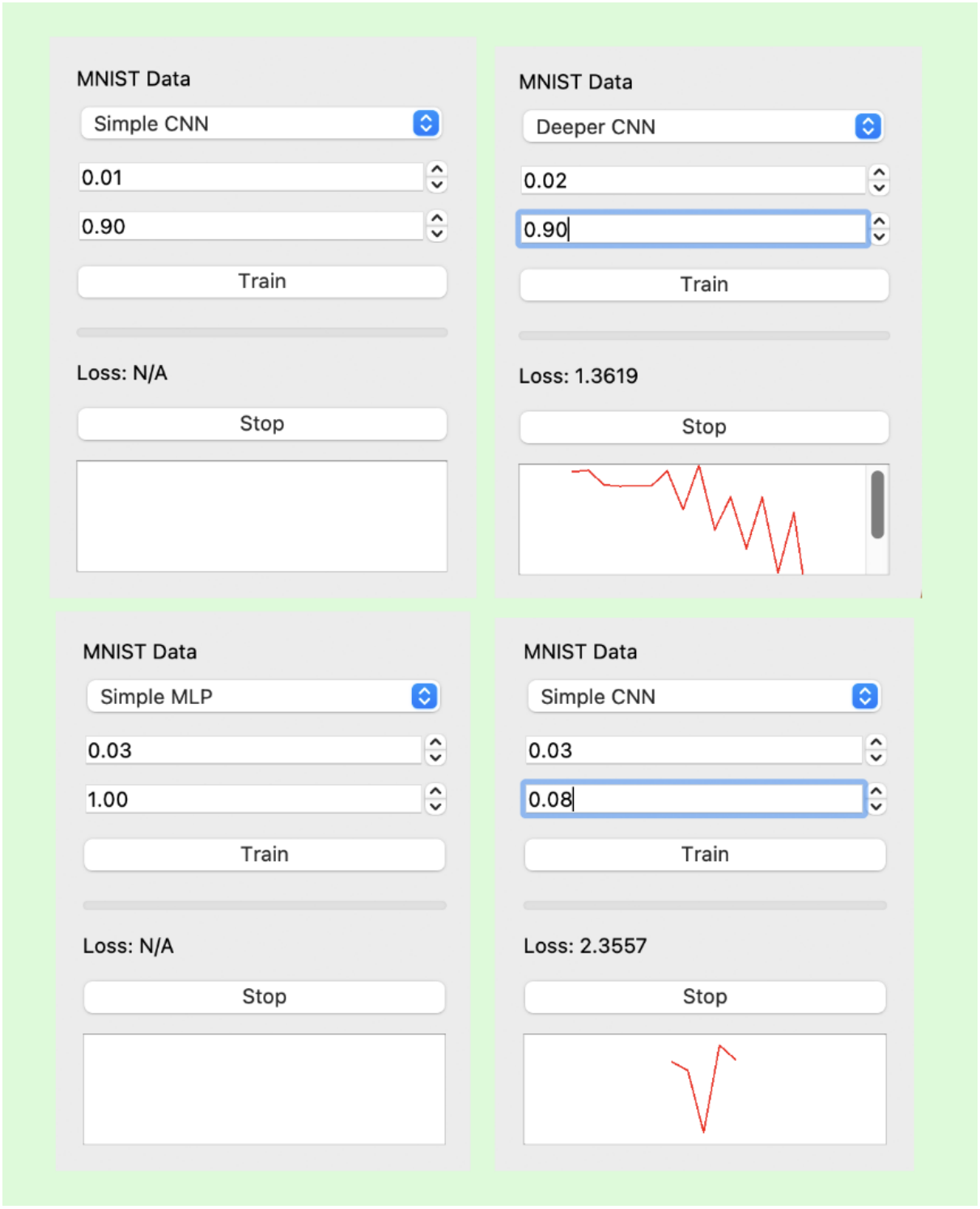
Screenshots and Diagrams:



Initial proposed wireframe architecture



Initial proposed architecture which was followed both for the parallel execution as well as the model creation for widgets and components of the PyQt5 GUI



Multi-GUI interface for parallel experimentation and model selection (on-going and post-GsOC work to convert this to drag and drop interface)

Future Work and Recommendations:

Extend Support for Additional Libraries and Formats:

- Plans are underway to extend support for additional deep learning libraries and different input formats like audio and video to make the application more versatile.

Enhance Scalability:

- Further optimization is needed to enhance the scalability of the application to efficiently handle even more complex models and large-scale data.

Augmenting Visualization Features:

- Additional visualization features and metrics representations can be developed to offer users more insights into their models and data.

Automated Model Deployment:

- Explore and integrate more options for model deployment, potentially automating the process further and offering varied deployment environments.

Conclusion:

The development journey of this GUI application has been intricate and enlightening, highlighting the importance of user-centric design, iterative refinement, and robust integration. The final product stands testament to the cohesive effort and adaptive learning throughout the project, aligning with the initial vision and expectations. The application has been tested in preliminary deployments, showcasing potential for extensive utilization in diverse deep learning experimentation scenarios, and holds promise for future enhancements and adaptabilities.

This project has contributed to the progression of user-friendly deep learning experimentation tools within the department and has set a solid foundation for future developments in the realm of intuitive, versatile, and robust deep learning experimentation interfaces.