# NeuraViz: A Web Application For Visualizing Artificial Neural Network Structures

A Manuscript

Submitted to

the Department of Computer Science

and the Faculty of the

University of Wisconsin–La Crosse

La Crosse, Wisconsin

by

**Bennett Wendorf**

in Partial Fulfillment of the

Requirements for the Degree of

**Master of Software Engineering**

May, 2024

# NeuraViz: A Web Application For Visualizing Artificial Neural Network Structures

By Bennett Wendorf

We recommend acceptance of this manuscript in partial fulfillment of this candidate's requirements for the degree of Master of Software Engineering in Computer Science. The candidate has completed the oral examination requirement of the capstone project for the degree.

| | |
|---|---|
| Prof. Albert Einstein<br>Examination Committee Chairperson | Date |
| Prof. Isaac Newton<br>Examination Committee Member | Date |
| Prof. Marie Curie<br>Examination Committee Member | Date |

# Abstract

Wendorf, Bennett, "NeuraViz: A Web Application For Visualizing Artificial Neural Network Structures," Master of Software Engineering, May 2024, (Jason Sauppe, Ph.D.).

This manuscript describes the software engineering processes and principles adhered to during the development of Neuraviz, a web application for visualizing artificial neural network structures. Users upload pre-trained machine learning models from popular frameworks including Pytorch and Keras, and Neuraviz generates a visual representation of the model's architecture. The following manuscript focuses on the design, implementation, testing, and deployment of NeuraViz in an effort to comprehensively encapsulate the entire development process.

# Acknowledgements

I would like to extend my sincerest thanks to my project advisor, Dr. Jason Sauppe, for his guidance and support throughout the development of NeuraViz. His feedback was always crucial in pointing me in the right direction, especially when I was overwhelmed with possibilities.

Thank you also to the entire computer science department at the University of Wisconsin-La Crosse for tirelessly helping me through all my coursework and projects throughout my tenure at the university. My ability to complete this monumental task would not have been possible without them.

I would also like to thank the open source community for providing the tools and resources used in this project. Open source software is an integral part of the modern software space and none of our lives would be the same without it.

Finally, I would like to thank my parents, family, friends, and all the teachers, mentors, and colleagues who have helped, supported, and encouraged me throughout my life. I am more grateful thank I can possibly express in words.

# Table of Contents

# List of Tables

# List of Figures

# Glossary

# 1. Introduction

## 1.1. Overview

This project aims to develop a software system to visualize the architecture of artificial neural networks. Neural networks (NNs) are a class of machine learning models that are inspired by the structure and function of the human brain. They are composed of a large number of interconnected processing elements, called neurons, which work together to solve complex problems. The architecture of a neural network refers to the arrangement of neurons and the connections between them. In addition to the general structure of a network, the weights and biases that control its function can provide useful insight into the inner workings of the model. The final integral parts of the neural network architecture are the activation functions that govern how data propagates through the network. Visualizing the architecture of a neural network can help students and researchers understand the structure of the model, identify potential issues, and communicate the model to others. More information on neural networks in general can be found in section 1.4.

This project will develop a web application that allows users to upload pre-trained neural network models and generate visual representations of their architecture. The application will support models trained using popular machine learning frameworks such as PyTorch and Keras. The resulting graph structure will be visualized in a pannable and zoomable svg format that shows the ordering of neurons, biases on those neurons, the weights of the connections between them and activation functions on each layer.

The application will be designed to be user-friendly and accessible to a wide range of users, including students and researchers. It will be implemented using modern web technologies and will be deployed as a web service that can be accessed from any device with an internet connection. The project will follow best practices in software engineering, including requirements analysis, design, implementation, testing, and deployment. The resulting application will be a valuable tool for anyone working with neural networks, but will be primarily targeted toward post-secondary educational students learning about machine learning and neural networks for the first time.

## 1.2. Similar Projects

Neural networks are a notoriously difficult concept to understand, especially for those who are newer to the field of machine learning. The architecture of a neural network is a key component of understanding how the model functions, but it can be difficult to visualize and comprehend. There are a handful to tools available for visualizing neural network architectures, but they are often limited in their scope. For example, Google's TensorBoard is a popular tool, but only natively supports TensorFlow and Keras models. In researching existing tools such as TensorBoard, Netron, and ENNUI, it was found that none of them supported the same range of formats as NeuraViz. This project aims to develop a user-friendly tool that is accessible to a wide range of users, including students and researchers who are new to the field of machine learning, and to support a wider range of model formats than other offerings.

## 1.3. Goals

With an aim to solve the issues with existing tools, NeuraViz will be developed with a few key goals in mind. One main goal is to support a wide range of formats and to visualize each format in a standard way. This will allow users to upload models from a variety of frameworks and see a consistent and comparable visualization of their model.

NeuraViz is also designed specifically with user-friendliness and simplicity in mind. The minimal interface focuses on the visualizes of the neural networks themselves, with minimal distractions. The use of color and shape in the visualizations will help to make the network architecture more understandable at a glance, quickly identifying neurons and connections that are the most important. Where needed, users can also zoom in or out an click on elements to see more detailed information.

Portability is another key goal of the development. Modern web technologies ensure the application is accessible from a range of devices, though it is most optimized for a desktop or laptop experience. Since the application is deployed as a web application, users don't need to install software on their own machines, or even sign in to be able to use the tool. To enhance this ability further, graph representations can also be exported as raw SVG or in tikz format for use within LaTeX documents.

## 1.4. Neural Networks

Neural networks are a type of machine learning model that are inspired by the human brain. They are made up of layers of neurons, which are connected to each other. Each layer of neurons takes in a number of inputs, processes them, and then outputs a value. The value that is output is then passed to the next layer of neurons, and so on, until the final layer of neurons outputs the final result.

### 1.4.1. Neurons

In an artificial neural network, neurons are the primary pieces of the network that perform the computation necessary. They are organized into groups called layers, typically represented in a graph structure organized vertically so the neurons in a layer are in a sort of column. Typically, the first layer is called the input layer, and behaves differently than other layers. For this reason, input neurons are represented as grey squares in NeuraViz as opposed to the typical neurons' circles. Neurons run the computations needed for the network to function with complex algorithms that I'll skip over here for simplicity. In general, neurons in a layer are connected to all the neurons in the previous layer, and all the neurons in the next layer. These connections are represented as lines between the neurons in NeuraViz.

### 1.4.2. Edges

Edges are the connections between neurons in a neural network. They are the primary way that information is passed between neurons. In NeuraViz, edges are represented as lines between neurons. Edges also have weights, which are used in the computation to determine how important that connection is. For small enough networks, the weights can be seen by hovering over edges in NeuraViz.

### 1.4.3. Activation Functions

Activation functions are a key part of how neural networks work. They are used to determine the output of a neuron based on the inputs it receives. There are many different activation functions, but one of the most common ones is the sigmoid function. The sigmoid function takes in a number and returns a number between 0 and 1. This is useful because it allows the network to output a probability. In NeuraViz, activation functions are shown at a layer-level, and represented as icons toward the top of each layer. Hovering over these icons will show the activation function used in that layer.

# 2. Software Development Process

## 2.1. Overview

## 2.2. Life Cycle Model

## 2.3. Requirements

# 3. Design

## 3.1. Overview

## 3.2. UML Class Diagram

## 3.3. Database

## 3.4. User Interface

# 4. Implementation

## 4.1. Overview

## 4.2. Technologies Used

### 4.2.1. Client

### 4.2.2. Server

### 4.2.3. Data Layer

## 4.3. Development

## 4.4. Deployment

# 5. Testing

## 5.1. Overview

## 5.2. Verification

## 5.3. Validation

# 6. Security

## 6.1. Overview

## 6.2. Threat Model

## 6.3. Session Management

## 6.4. Web Application Security

# 7. Conclusion

## 7.1. Overview

## 7.2. Challenges

## 7.3. Future Work

# 8. Bibliography

# 9. Appendices