Neural Network Web Application

CA400

Functional Specification

Liam McAweeney

14415154

23/11/2018

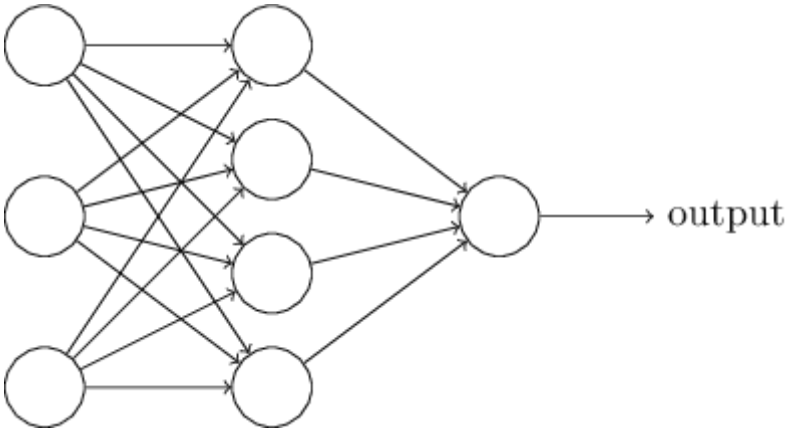
Table of Contents

Introduction

Overview

The system is a graphical user interface for designing and training neural networks. The user will drag and drop components, such as a hidden layer, onto the graphical representation of the neural network. The code for generating the neural network will then be dynamically created based on the users graphical design.

Creating a neural network can be a daunting task. When researching neural networks you will come across images like this:



After reading through tutorials about how a neural network does what it does, these images start to make sense. You can visualise an input going into a node, being altered by a weight and an activation function, then producing an output. When it then comes to actually writing the code to mirror your design it becomes a lot more daunting.

This system will allow a user to upload their training data. The user can then drag and drop components onto the GUI of a NN. Variables in the design of a network will number of nodes in hidden layer, number of hidden layers, bias and activation functions. Variables in the training of a network will be the cost function, learning rates, number of iterations. batch size and number of epochs. The user can then activate the training of there model. The user will be notified when training is complete and he accuracy will be displayed. The user can alter any variable and retrain the model. The model can be used on the application or the user can export the model.

Business Context

This application would be an ideal solution for both people wanting to learn how to implement neural network by themselves and for computer courses want to teach their students how to implement neural networks. Colleges could provide students with training sets and teach them how to design and train networks based on the data.

Glossary

Neural network - A Neural Network is an information processing paradigm that is inspired by the way biological nervous systems, such as the brain, process information.

Activation function - The activation function of a node defines the output of that node, or "neuron," given an input or set of inputs.

Loss function - A loss function is a function that maps an event or values of one or more variables onto a real number intuitively representing some "cost" associated with the event.

Epochs - One Epoch is when an entire dataset is passed forward and backward through the neural network only once.

Hash - A hash function is any function that can be used to map data of arbitrary size to data of a fixed size. The value returned is know as a hash.

Salt - a salt is random data that is used as an additional input to a hash , a password or passphrase.

GPU - Graphic processing unit.

GDPR - General Data Protection Regulation.

General Description

System function

Dynamic Neural Network design

This is the main function in of the system. The user will drag and drop different components into the neural network such as a hidden layer or multiple hidden layers. The user may also be prompted to assign a value to the component, such as a sigmoid as value for the activation function component and number of nodes in a layer.

Neural Network training

Another major function in the system is to train the network. The system will retrieve the neural network model which the user has created. The user will then have to specify different training variables such as the divide of the training/testing dataset and number of epochs. Training the network may take some time and computation power.

Data visualisation

The System will have Data Visualisation functionality to represent the accuracy of different versions of the user's neural network. The data will primarily be displayed in the form of line graphs. Data visualisation will be aid the user in deciding how to redesign and retrain their network's to improve the accuracy. An example of a graph the network's loss.

Storage

The User's data will be stored in a database. A trained neural network on the other hand can not be stored in a database. The network will need to be stored on a server or some other location where the system will have access to it. Datasets used to train a network must be stored on a server or in a database.

Notifications

The user will not be expected to be active while training is taking place as it may take some time. An automatic email will be sent to inform the user that the training is complete and give the an overview on the performance in testing.

User characteristics and objectives

The typical user of this system, be it a student from a college course or simply a hobbyist, would be expected to have a basic understanding of different components in a neural network.

As well as this, the ideal user would have at least a basic understanding of the various functions used such as an activation function, loss function and learning rate. However, to satisfy the design goal of accessibility and continual learning, information modals containing an explanation of such functions will exist on the application in order to ensure new users can quickly learn about each type of function used.

From the perspective of the user, the goal of this system is essentially to allow them to easily experiment with designs to improve the accuracy of the model.

Specifically, however, this goal can be broken down into **4 predominant objectives**

1. Dynamically design a neural network to a standard where it would be as least equal to the effort taken to generate the code manually, saving hours upon hours with configuring and implementing time which can instead be used to optimise the network's performance.
2. Train the Neural Network, the training phase may be implemented several times as the user will redesign the network to make it more accurate. Each iteration of training the model may take some time so an alert system will be set up to inform the user upon training completion.
3. Providing a visual representation of the networks performance. This will allow the user to make informed decisions in the redesign of their network to improve the networks accuracy. This application will automatically create graphs without any user input.
4. Provide the user a platform to input data and retrieve a classification from that data using the neural network which they designed and trained.

Operational Scenarios

There are three main operational scenarios that the user will undertake while using this application, which are represented as use cases in this section. Modules are listed as secondary actors, which corresponds to the concept of a model in the MVC design pattern that has been used to design this application (explained in section 4 of this document).

Use Case 1 - User signing up to Application

User Objective	User signs up to Application	
Goal in Context	User wishes to create a profile for application to gain access to applications features	
Scope & Level	Client & Server	
Preconditions	User has a registered email address which is not already associated with a profile	
Success End Condition	User receives confirmation email and gains access to the application's features	
Fail End Condition	User fails to create a profile	
Trigger	User chooses "Sign up"	
Description	**Step**	**Action**
	1	User enters email and password.
	2	User chooses username and enters personal details.
Extended	3	Controller checks with the Model to ensure the username or email does not already exist.
	4	Controller sends email to the user's specified address.
Extended	5	User Confirms account creation process via email link.
	6	Controller passes the new profile information to the model.
	7	Model adds SALT, hashes password and stores in database.
Extensions		**Branching Condition**
	3.1	Username or Email already Exists.
	5.1	User does not receives email.
Extensions		**Branch Action**
	3.1	View asks user to choose a different username or email.
	5.5	Confirmation link times out and user must contact support.


User signing up to application
Enter Email + Password
Check for existing Email or Username
Email Confirmation
Enter Username + Personal Details
Add New User
Salt + Hash Password + Sore in Database
User
Controller
Model

Use Case 2 - Design a neural network on Application

User Objective	User Designs a neural network on Application.	
Goal in Context	User wishes to design the networks structure and save the model for future use.	
Scope & Level	Design Module. (Client & Server)	
Preconditions	User must have an account registered with the application and is logged into their account.	
Success End Condition	User designs and saves a neural network model .	
Fail End Condition	User fails to design and/or save the model.	
Trigger	User chooses “Design Neural Network”.	
Description	**Step**	**Action**
	1	User selects Design Neural Network.
	2	User grabs a network component and drags it onto the visual neural network.
Extended	3	User will the specify variables for component from a dropdown menu.
	4	User selects “Save model”.
	5	User inputs a name for the model.
Extended	6	Controller passes the model to Storage
	7	View passes confirmation to user.
Extensions		**Branching Condition**
	3.1	User is not finished adding components.
	6.1	A model already exists with the inputted name.
Extensions		**Branch Action**
	3.1	User will repeat step 2 and step 3.
	6.1	View asks user to choose a different model name.


User designing a neural network
Select Design
Drags and Drops components
Enters Name and Saves
Add new Design Model
User
Controller
Model

Constraints

Time Constraint
This final year project must be completed by a fixed deadline May 2019. While I am confident I can complete this application to a satisfactory standard, this time constraint can have an impact on the scope of the project. I must respect this time constraint and produce a fully functioning application within the timeframe.

Hardware Constraint
Training classification algorithms will require significant processing power. These algorithms have significant time complexities too and analysing large datasets with a large number of attributes would have a large effect. In developing this final year project the processing power on my laptop may not suffice. However for most of the develop it will. I will have to upgrade the processing power when training larger datasets. I will use either an amazon or paperspace server which will give me access to a GPU.

Security Constraint
My application will need it's users to register an account, I will implement security protocols to ensure the user's information is secure.

GDPR Constraint
I must ensure that my application is compliant with GDPR recommendations. I will take a privacy by design approach and only take data that is absolutely necessary. Within the scope for my final year project my application will not be obtain and processing a user's data. Since this application possibly will have a greater scope I will provide the functionality to allow the user a copy of their data.

Functional Requirements

Designing Neural Network

- **Description** - The main functionality of this application is allowing the user to visually design a neural network. Therefore the user will not have to write any code to design the network.This approach has been chosen as it will save the user time in configuring the network which in turn will give them more time to optimise the network.

- **Criticality** - This is an essential feature of the application, given the fact that the inspiration for this system was the notion of creating a network without writing any code. If the code free designing functionality was omitted, the application would not truly reflect this inspiration.

- **Technical issues** - One of the main issues with the design functionality is that it will have to be diverse enough to give the user's different options for each component, ie more than one activation function. Since the system will have limited control over the design of the neural network, it must be capable of handling any design the user creates. It most recognise when the user may have made an error and inform them before the design is passed to training. The design model must be diverse enough to handle an design which the user creates.

- **Dependencies** - The design functionality is not dependent on any other functionality. It is however a dependency of the training functionality.

Data Visualisation

- **Description** - The data visualisation section will take the user to a dashboard which will convey, in the form of various charts and graphs, information on the performance of the machine learning module.

- **Criticality** - This feature is important as it will allow the user to see what could improve the networks accuracy. It will also provide the user with a convenient method of evaluating their own performance over the course of training one or more networks.

- **Technical issues** - A major difficulty with the visualisation of the data retrieved from training the network is that because the network is designed dynamically, the data visualization process must be completed dynamically too

- **Dependencies**- Data visualisation is dependent on the training functionality.

Training Neural Network

- **Description** - Another major functionality of this system is to train the network designed by the user. The training will involved running the dataset through the network a number of times. This will be done after the user has designed or redesigned a model. The training of network make takes some time so a notification will be sent to the user on completion.

- **Criticality** - Again this is an essential feature of the application. The network which the user has designed has no value until it has been trained. A model must be trained to allow the user to implement the model for predictions.

- **Technical issues** - Training the network may be computationally expensive. The system will need to be able to handle training multiple networks at the same time. The system will also need to be able to store multiple large datasets with fast access for training.

- **Dependencies** - The training functionality is dependent on the design functionality. It is also a dependency of the Data Visualisation and notification functionalities.

Notifications

- **Description** - This will alert the user when training has been completed. It will alert the user on the accuracy of the network and the other KPI's he/she may have set for alerts.

- **Criticality** - This is not a critical feature of the application, but it keeps the user informed so they can proceed with another iteration of optimisation.

- **Technical issues** - Notifications must be optional as not all user will want to receive alerts. Gaining access to the user's phone number must be GDPR complaint.

- **Dependencies**- Push notifications will be dependent on the training functionality and the data visualisation functionality.

System Architecture

This system is a web application so it's design is based on the MVC design pattern.
There are three main components which comprise this design pattern:

1. **Model**: This is a representation of data. Not to be confused with the data itself, the model can interact with, while ignoring the intricacies of, the underlying database. The model may also act as an abstraction layer between the data and the database, meaning it can be general to any of the various types of databases in question.

2. **View**: The view is the presentation layer for the model. Essentially, it is what the user will see in their browser while using the web application. Importantly, the view also boasts an interface which can accept user input.

3. **Controller**: Following the trend of rather aptly named components, the controller controls the flow of information between the model and the view (the "front-end" and the "back-end"). The controller uses programmed logic to interact with the database, and to collect information through the view.

This application has been designed with a React JS frontend and a Python Flask Backend. React JS will be the view, Python Flask will be the model and controller in the MVC structure.

The following section will describe the design of the system in terms of the traditional MVC pattern.

MVC in the application

Structure of the application:
The system will be broken into multiple different modules working together to form the larger system.
These modules are largely based on the functional requirements

1. The Design module
2. The Training module
3. The Data Visualisation module
4. The Prediction module

Each of these modules contain their own implementation of the MVC design pattern, meaning there are five separate models, views and controllers in the system.

The Design module

The design module represents the design phase in creating a neural network. Here the user will dynamic create the design of the neural network.

Model:
The design model will store the structure of the neural network which the user will create. This model may be broken down into smaller models as it will have a large amount of diversity. This is because the user will be able to create lots of different types of neural networks.

View:
The view of the design module will be the main page the user will use. The view will contain a two main components, a sidebar and and the main display. The user will drag components from the sidebar into the main display to configure the neural network. The view will also take in input data such as number of nodes per layer. The view will also display previously design networks for the purpose of redesign.

Controller:
The controller for the design module will pass the design specification from the view to the model. The controller will also retrieve previously designed networks from the mode and pass them to the view.

The training module

The training module will take a design from the design module and perform training on it based on the users specifications.

Model:
The training model will contain the dataset which will be used to train the user designed network. It will also contain the other information in regards to the training completion such as time taken and accuracy.

View:
The view will take user input such as the split of the dataset for training and number of epochs. This will be the dashboard of the application, it will display the training progress.

Controller:
The controller of this module will train the network. It will take the data from the model and will take the designed network. It will then perform the training on the network.

The data visualisation module

The data visualisation module will provide a visual feedback for the user about the network. It will display the accuracy and loss of the network.

Model:
The model for this module represents the data which was produced from training the network.

View:
The view for this module will allow the user to examine the performance of the network. The user will be able to compare multiple networks at the same time. This will allow the user to decide what network to proceed with and continue designing.

Controller:
The controller for this module will transport the visualised data from the model to the view, not before performing programmed logic to transform the data into the respective graphs and charts.

The Prediction Module:

The prediction module is where the user can use the neural network they designed to make a prediction.

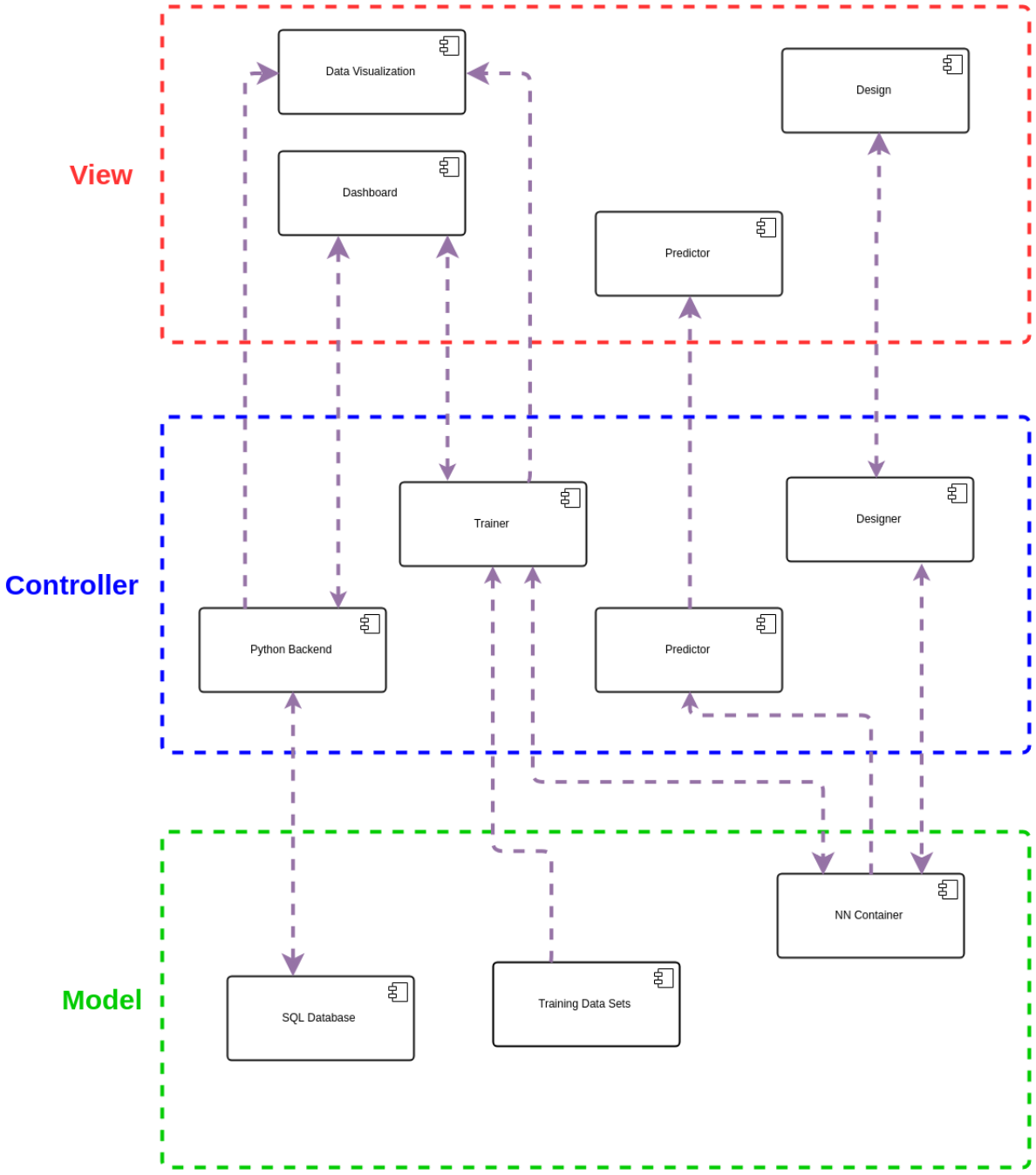
Model:
The model will represent the trained neural network. It will also contain information about data about the use of the classifier.

View:
The view is allows the user to input data to make a prediction. The view will also then display the out of the prediction.

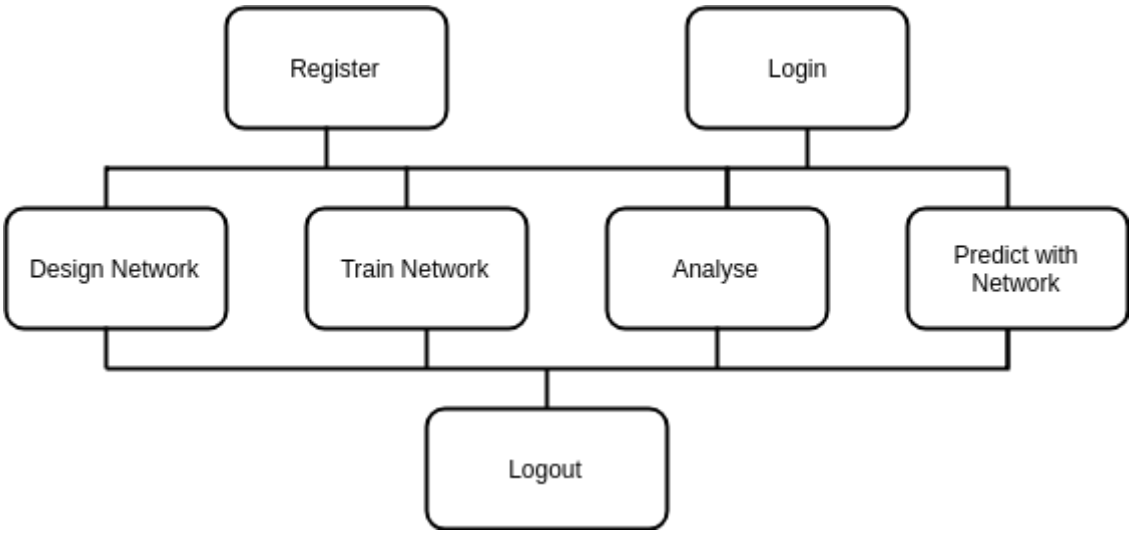
Controller:
The controller takes the user's input data from the view and takes the network from the model and makes a prediction. This prediction is then displayed in the view.

Component Diagram

Rather than drawing individual component diagrams for each of the above modules, a data flow diagram will delve into further detail in section 5.
The following is a component diagram for the wider system, meaning rather than focusing on the MVC design of each module, this diagram represents the resulting MVC design of the system as a whole.

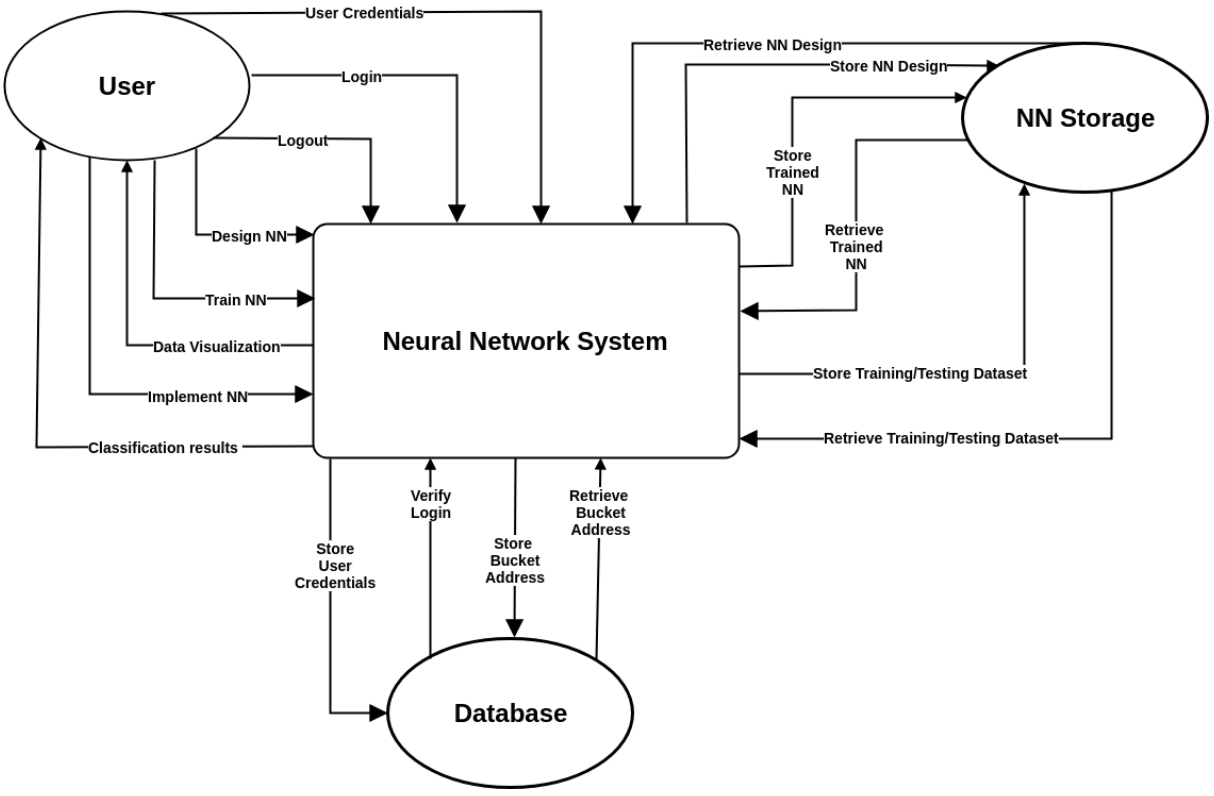


High Level Design



The above diagram illustrates the high-level design of the systems functionality. Initially the user will begin by registering with the application. One registered they can proceed to login. After successful authentication, the user can navigate through the application to design, train, analyse and use their neural network. The analyzation functionality will be provided through a serval graphs describing the network performance through training and testing. Lasty the user will be able to log out of the application.

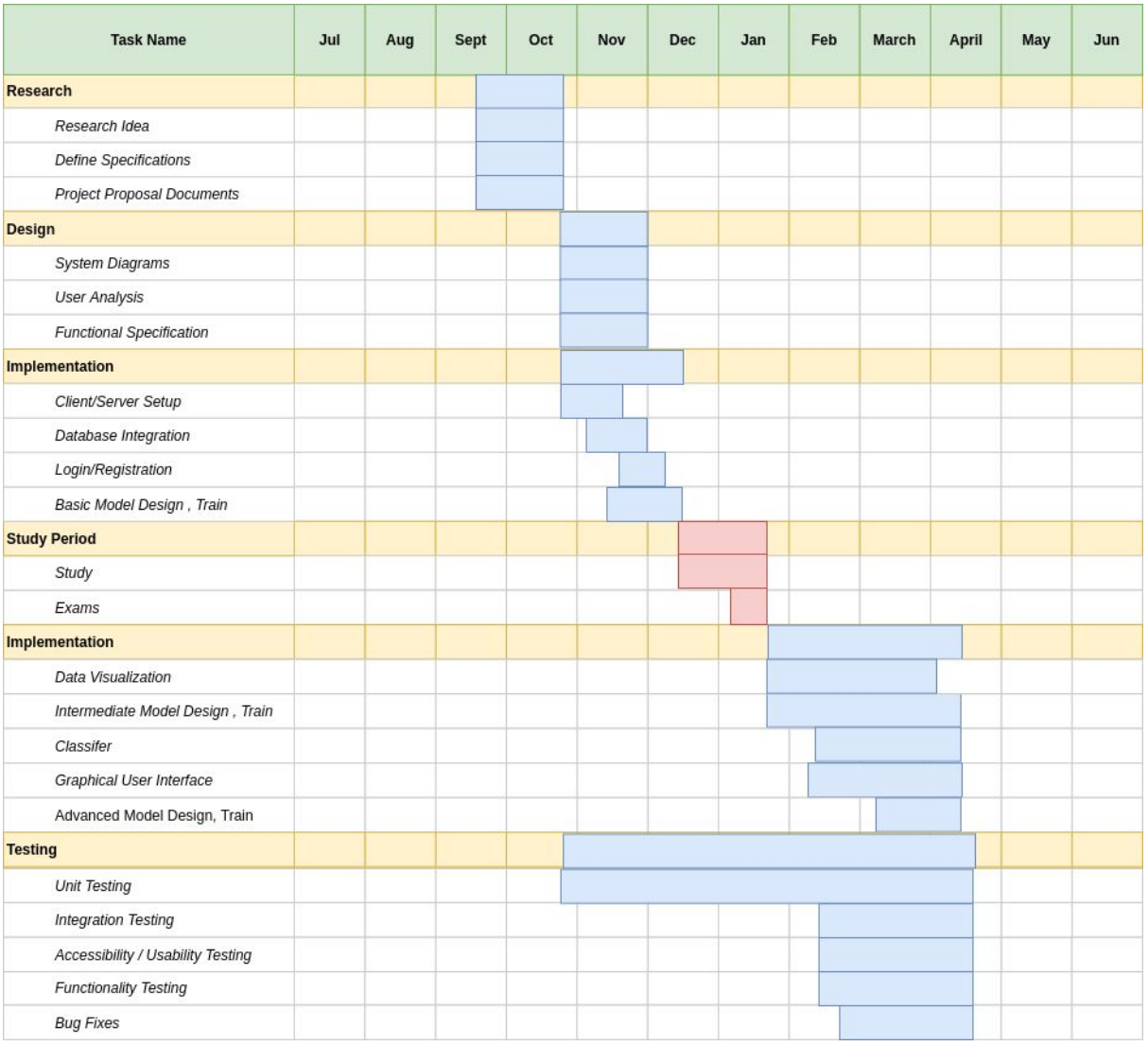
Context Diagram



The Context diagram above illustrates how the user, database and storage entities interact with the system. The user will enter their credentials when registering an account with the application. The user can then enter their credentials and the system will verify and authenticate their details.

Preliminary Schedule

Gantt chart



The current project schedule is shown above in the gantt chart. The research phase is complete and implementation of the application has commenced. By end of semester one I intend to have very basic design and train modules created. By January I plan to have a basic web application with minimal UI design completed. By the end of March I hope to have the majority of the application implementation complete and I can focus on testing and validating the code. Once all the development is finished I can focus solely on documentation and demo deliverables.

Appendices

Research

http://neuralnetworksanddeeplearning.com/
http://medium.com/

Diagram Creation

http://draw.io