< Large-Scale Design & Analysis of Neural Networks >
System Requirements Specification
Version <1.0>
10/29/2024

Document Control

Distribution List

The following list of people will receive a copy of this document every time a new version of this document becomes available:

Teaching assistants:

Clay Pate

Customer(s): M. Ilhan Akbas

John M. Thompson

Project team members:

Daniel Machado

Simon Cole

Muhammed Sharif Adeptu

Leah Powell

Change Summary

The following table details changes made between versions of this document:

Version	Date	Modifier	Description
0.1	10/15/2024	Leah	Document Creation
1.0	10/29/2024	Leah	First Draft Finish

Table of Contents

1. Introduction

1.1. Purpose and Scope

This SRS covers the scope of the Large-Scale Design & Analysis of Neural Networks senior design project which aims to streamline the process of implementing deep learning using Bayesian Neural Networks. This is to address the lack of systemic optimization of a Neural Network's architecture from a statistical standpoint using the gradient of cost functions. This includes creating visualization tools for determining loss criteria and using histograms to analyze the weight distributions and convergence success for the neural networks created in this pipeline process.

The application of this system would be beneficial to the goals of any field in neural networks science. This is especially regarding the long-term accuracy of the training that results from these neural networks and the much-needed organization required to make these pipelined neural networks more efficient and consistent in accuracy/convergence.

1.2. Intended Audience and Reading Suggestions

This SRS document is intended to be used by any stakeholders of the project, members of the development team, or other collaborative parties such as teaching assistants. The rest of the SRS contains a product descriptive section (2.0) followed by a series of types of requirements. These include behavioral requirements, non-behavioral requirements, and any other non-specified requirements that might be necessary such as database requirements. After that are the project models which were specifically selected as appropriate for this project type and design. A typical reading sequence would be to start with the Introduction and Description, then read through the Analysis Models, then finally the requirements sections as directed.

1.3. Document Conventions

This document follows the typical template conventions.

1.4. Project References

Project Proposal Document

1.5. Definitions, Acronyms, and Abbreviations

1.5.1. Definitions

This section lists terms used in this document and their associated definitions.

Table 1: Term Definitions

Term	Definition
Pytorch	A machine-learning library based on the open source library on Lua.
Baye's Theorem	A theorem on probability, or statistical inference, that prior probability can be combined with new values and evidence to create a posterior probability which is updated as it becomes available.
Bayesian Neural Networks	Neural Networks whose weight distributions are modeled after probabilities using Baye's Theorem.
Architecture	Structural and logical layout and overall organization of a Neural Network.
Bins	Grouping continuous data into categories or intervals to use data as if it were discrete.
Weights	The connections between nodes/units in a neural network.
Histogram	Visualizing the distribution of continuous data into a graph and grouping into bins, similar to a bar-graph.
Convergence	Refers to the point where a model/network [learning] reaches a stable state and changes to its performance are at a minimum.
Tensors	Multidimensional arrays to store data used by neural networks

1.5.2. Acronyms

This section lists the acronyms used in this document and their associated definitions.

Table 2: Acronym Definitions

Term	Definition
SDD	System Design Document
NN	Neural Network
BNN	Bayesian Neural Network
BN	Batch Normalization
DL	Deep Learning
SRS	Software Requirements Specification
ML	Machine Learning
CONV	Convergence

1.5.3. Abbreviations

This section lists the abbreviations used in this document and their associated definitions.

Table 3: Abbreviations Definitions

Term	Definition
e.g.	For example
Etc.	Ex cetera

2. General Description

2.1. Product Perspective

This product serves to improve the very structural organization and implementation of designing Neural Networks, or Deep Learning Systems. Results from this product will affect development in areas such as how networks are generated, how networks are initiated, or how they shall be pipelined.

2.2. Product Features

Our product will have several tools for users to analyze and edit their neural networks more quickly. The features include a tool for quicky and efficiently Loding a large set of neural networks into a project, then analyzing the losses to be able to quickly see if their networks are converging, as well as binning the weights

2.3. User Classes and Characteristics

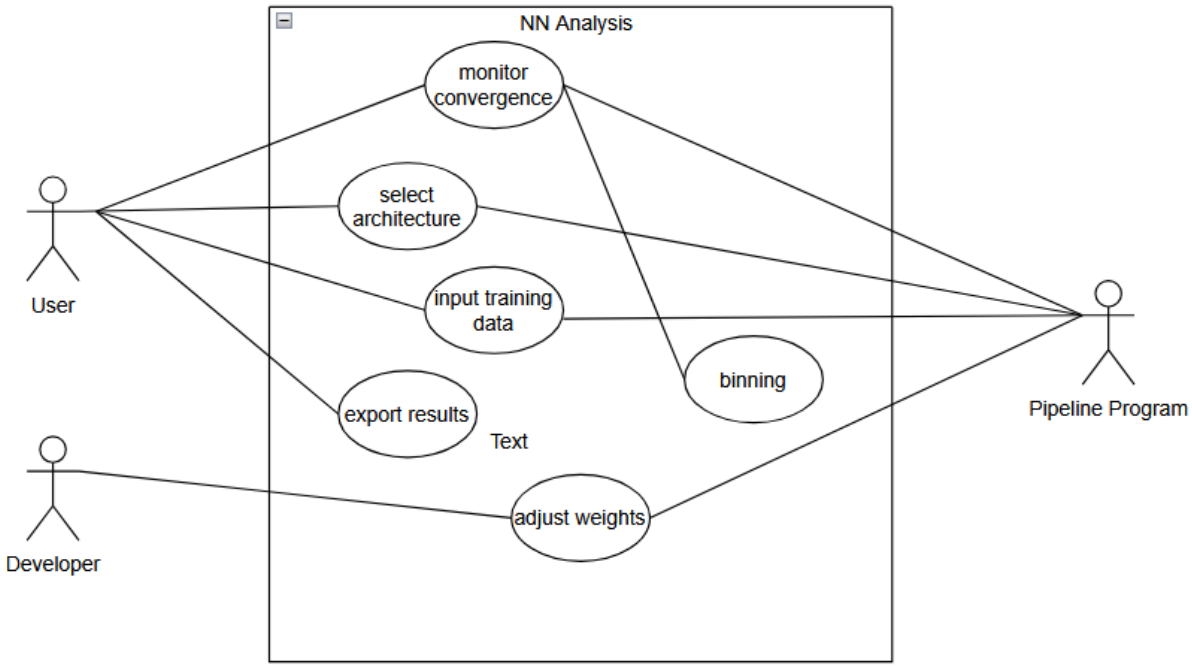

Figure 1. Use Case diagram of the system

2.3.1. Actors

This section presents the actors in the system.

- *User*
  - *An End User is the user which uses the system for enhancing their hiking experience.*
- *Developer*
  - *A developer is the user which initially writes and later maintains the system*

2.3.2. Use Cases

This section presents the Use Cases, developed for the system.

- *Monitor Convergence*
- *Select Architecture*
- *Input Training Data*
- *Export Results*
- *Adjust Weights*
- *Binning*

2.3.3. Scenarios

*This section presents scenarios for **each use case**, described in the previous section.*

2.4. General Constraints

The system will be only be accessible via code from a GitHub repository.

2.5. Operating Environment

The software for the project that controls network creation runs in Jupyter Notebook and requires decent hardware to test and run the models. No specific versions are currently required.

2.6. User Documentation

User documentation components are currently not required or implemented.

2.7. Assumptions and Dependencies

The development team has made the following assumptions:

- *Source code will be provided via GitHub*

3. External Interface Requirements

3.1. User Interfaces

[REQ-1] The terminal shall display to the user a prompt to input architecture.

[REQ-2] The user shall be able to input a text file.

[REQ-3] The user shall be able to specify the size of the test-set as a percentage of the training data, within a minimum of 10%.

[REQ-4] The user shall be able to customize the success criteria to evaluate each model.

3.2. Software Interfaces

[REQ-1] The system shall run with files provided by the user for architecture and training data.

3.3. Communications Interfaces

[REQ-1] The system shall be run using network connection.

4. Behavioral Requirements

4.1. Same Class of User

[REQ-1] The system shall have only one level of access privilege to use across developers and users of the system.

4.2. Stimulus

[REQ-1] The system shall automatically generate k Neural Networks based on architecture and training data provided by the user.

[REQ-2] The system shall evaluate each neural network based on user-defined success criteria.

[REQ-3] The user shall be able to specify different training hyperparameters for each neural network generated, such as learning rate or batch size.

4.3. Related Features

[REQ-1] The system shall have a tool which generates discretized bins for each weight of some NN.

[REQ-2] The system shall populate bins with the number of occurrences of that weight, in that range, across all NN's.

4.4. Functional

[REQ-1] The system shall run without crashing.

[REQ-2] The system shall alert the user if the maximum iterations is reached without success criteria being met.

[REQ-3] The user shall be able to specify a maximum number of training epochs.

[REQ-4] The user shall be able to specify a maximum number of attempts to generate a successful model.

5. Non-behavioral Requirements

5.1. Performance Requirements

[REQ-1] The system shall have a tool for systemic generation of loss graph.

[REQ-2] The system shall display the generated loss graph(s).

[REQ-3] The system shall use the loss graph tool to define the success criteria.

5.2. Safety Requirements

[REQ-1] The system shall not be used for exploitation in any way.

5.3. Qualitative Requirements

5.3.1. Availability

[REQ-1] The user shall be able to re-run the model if the network fails.

5.3.2. Security

[REQ-1] The system shall keep generated network data in a specified file.

5.3.3. Maintainability

[REQ-1] The system shall be clean of unnecessary lines or code

5.3.4. Portability

[REQ-1] The system shall be able to run after installation of package from git without extra installations.

5.4. Design and Implementation Constraints

[REQ-1] The software shall be developed in Jupyter Notebook using Pytorch

6. Other Requirements

6.1. Database Requirements

[REQ-1] The system shall use a file inputted by user for data.

6.2. Operations

[REQ-1] The user shall be able to save neural networks on a set location.

[REQ-2] The user shall be able to keep networks that meet success criteria and disregard those that do not meet it.

7. Analysis Models

7.1. Data Flow Model

<to be inserted eventually>

7.1.1. Data Sources

The data sources and their inputs to the system identified in the data flow model are as follows:

- *Network Layers*
- *Binning mechanism*
- *Histogram*
- *Pipeline*

7.1.2. Data Sinks

The data sinks and their system outputs identified in the data flow model are as follows:

- *Network Layers*
- *Tools*
- *Network Program*

7.1.3. Data Dictionary

The data types are described in the data dictionary below. This section includes the name of the data type; a description of the contained data; how the data is structured; and the range of values.

Name	***Description***	***Structure***	***Range***
Training data			
Number of networks			
Bins			
Epoch data/results			

7.1.4. Context Diagram (Level 0 Data Flow Diagram)

<to be inserted eventually>

7.1.5. Level 1 Data Flow Diagram

<to be inserted eventually>

7.1.6. Level 2 Data Flow Diagram

<to be inserted eventually>

7.2. Class Model

<to be inserted eventually>

7.3. State Model

<to be inserted eventually>.

8. To Be Determined List