Software Requirements Specification for ANN: Artificial Neural Network

Tanya Djavaherpour

February 14, 2024

Contents

Revision History

Date	Version	Notes
February 2, 2024	1.0	Initial Draft

1 Reference Material

This section records information for easy reference.

1.1 Table of Units

Throughout this document SI (Système International d'Unités) is employed as the unit system. In this document, we do not use any specific SI unit. While the SI unit system is primarily employed, for image dimensions, the unit of pixels (px) is used due to its relevance and standard usage in image processing tasks. Derived units are used as described below. For each unit, the symbol is given followed by a description of the unit and the SI name.

symbol	**unit**	**SI**
px	pixel (matrix entity)	Non-SI (custom)

1.2 Table of Symbols

The table that follows summarizes the symbols used in this document along with their units. The choice of symbols was made to be consistent with Artificail Neural Network literature and with existing documentation for Image Classification. The symbols are listed in alphabetical order.

symbol	**unit**	**description**
α	–	Learning rate in gradient descent
a^L	–	Weighted input to neurons of layer L
b^L	–	Bias vector of layer L in neural networks
${b_i}^L$	–	Bias of neuron i of layer L
∇	–	Gradient operator
σ	–	Activation function
W^L	–	Weight matrix of layer L in neural networks
${w_{ij}}^L$	–	Weight between neuron i and neuron j of layer L
y_j	–	Target output for neuron j
${z_i}^L$	–	Output of neuron i of layer L

1.3 Abbreviations and Acronyms

symbol	description
A	Assumption
DD	Data Definition
GD	General Definition
GS	Goal Statement
IM	Instance Model
LC	Likely Change
PS	Physical System Description
R	Requirement
SRS	Software Requirements Specification
ProgName	Artificial Neural Newtwork for Image Classification
TM	Theoretical Model

1.4 Mathematical Notation

Typographic Conventions: In this document, matrices are denoted by bold uppercase letters (e.g., W for weight matrices), vectors by bold lowercase letters (e.g., a for activation vectors), and scalars by regular font (e.g., α for the learning rate). Greek letters are used for specific functions or parameters (e.g., σ for the activation function).
Symbols for Mathematical Operations:

– ∇ (nabla): Represents the gradient operator, used in gradient descent.

– Σ (sigma): Denotes summation.

– ∂ : Symbol for partial derivatives, indicating the rate of change of a function with respect to one of its variables while keeping other variables constant. It is extensively used in backpropagation algorithms for calculating gradients.

2 Introduction

The evolution of machine learning and its profound impact on various fields has led to remarkable advancements, particularly in the area of image classification. This project delves into the exploration of enhancing an Artificial Neural Network (ANN) to tackle the challenge of classifying images from the CIFAR-10 dataset Krizhevsky (2009). This dataset, known for its diverse range of images, serves as a crucial benchmark in assessing the capabilities of image recognition algorithms. Our goal is to not only improve the accuracy of the ANN but also to make it robust enough to handle all ten categories of the dataset effectively.

2.1 Purpose of Document

The primary purpose of this Software Requirements Specification (SRS) document is to clearly define the software requirements for the development of an advanced neural network model for image classification using the CIFAR-10 dataset. This document aims to provide a comprehensive understanding of the system, addressing various aspects such as the project's goals, theoretical underpinnings, and fundamental definitions. It maintains an abstract perspective, focusing on identifying and specifying what needs to be solved rather than delving into the technicalities of how these solutions will be implemented. The document will describe the system context and constraints, delineate the specific problem definition, and outline the requirements and characteristics of the proposed solution. Additionally, it will address potential changes and modifications that could impact the development of the neural network model.

2.2 Scope of Requirements

The scope of this project is specifically tailored to developing an enhanced neural network model for image classification using the CIFAR-10 dataset Krizhevsky (2009). This involves:

- **Model Complexity:** The project will concentrate on enhancing an existing ANN architecture. It will not venture into developing entirely new models from scratch or exploring unrelated AI domains like natural language processing.
- **Operational Boundaries:** The tool will be designed for a general-purpose computing environment, without tailoring for specialized hardware like high-performance GPU clusters.
- **Functional Focus:** The primary function is to improve classification accuracy and efficiency. Features outside the realm of image classification, such as real-time data processing or integration with external APIs, are outside the project's scope.

For more details, you can also see the assumptions section (Section 4.2.1).

2.3 Characteristics of Intended Reader

The intended readers for this project, which involves developing an enhanced neural network model for image classification using the CIFAR-10 dataset **?**, are likely to be individuals with a solid background in computer science, specifically in the areas of machine learning and neural networks. However, as explained in Section: User Characteristics (Section 3.2), users of this project are a broad audience interested in image classification, without the necessity for a deep technical background in machine learning or neural networks.

2.4 Organization of Document

The organization of this document is modeled on the SRS template for scientific computing software as outlined by Smith and Lai (2005). This approach systematically presents the system's objectives, theoretical underpinnings, definitions, and foundational assumptions. To facilitate a top-down reading approach, the readers can begin by reading the system's goal statements (Section 4.1.3). Following this, the theoretical models are detailed, which build upon and clarify the goals. The document concludes with instance models, providing readers with a comprehensive understanding of the system's practical applications and functionalities.

3 General System Description

This section provides general information about the system. It identifies the interfaces between the system and its environment, describes the user characteristics and lists the system constraints.

3.1 System Context

The system's context, as depicted in Figure 1, comprises three main elements: circles, a rectangle, and arrows. The circles symbolize the user who interacts with the software. The rectangle signifies the truss tool software system itself. Arrows in the diagram are used to illustrate the flow of data, showing both the input provided by the user and the output data generated by the system that is relevant and useful to the user. This setup provides a clear visual framework of the system's interaction dynamics.

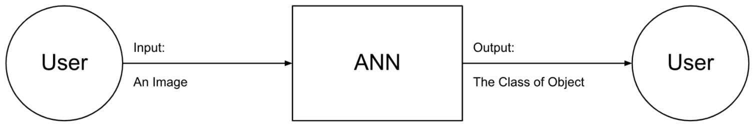

Figure 1: System Context

- User Responsibilities:
    - Input correct and relevant data into the system

  - Upload images that are compatible with the system's specifications and within the scope of the CIFAR-10 dataset **?** classifications

- ANN Responsibilities:
  - Detect data type mismatch, such as text file instead of an image
  - Process image data, ensuring timely and accurate output
  - Allows users to easily upload images and view classification results
  - Ensure the security and privacy of any data uploaded by users for classification

3.2 User Characteristics

The users of this image classification system are anticipated to have basic proficiency in using software for tasks such as uploading images. Their understanding of image classification should be fundamental, enough to set realistic expectations of the system's functionality. Extensive expertise in machine learning is not required, but a general familiarity with the types of images in the CIFAR-10 dataset **?** will be beneficial for choosing appropriate images for classification. Additionally, a keen interest or curiosity in the practical applications of machine learning, especially in areas like image classification, will likely enhance their experience and engagement with the system.

3.3 System Constraints

In this project, system constraints are critical as they shape the design space and specify how the system must integrate into the real world.

Technical Resource Limitations: The system's design is bound by the available computational resources, such as processing power and memory capacity, which influence the complexity and efficiency of the neural network model.

Dataset Constraints: The system is tailored specifically for the CIFAR-10 dataset **?**. This specialization may limit its effectiveness with other types of image data. Moreover, this system cannot classify any images that is not in the 10 classes.

4 Specific System Description

This section first presents the problem description, which gives a high-level view of the problem to be solved. This is followed by the solution characteristics specification, which presents the assumptions, theories, definitions and finally

4.1 Problem Description

The ANN in this project is intended to solve the complex challenge of image classification, a task that involves correctly identifying the content of digital images. The problem arises from the need for a highly accurate, efficient, and reliable classification model that can navigate through the intricacies of image data, discerning subtle differences and patterns.

4.1.1 Terminology and Definitions

This subsection provides a list of terms that are used in the subsequent sections and their meaning, with the purpose of reducing ambiguity and making it easier to correctly understand the requirements:

- **Artificial Neural Network (ANN):** A computational model inspired by the human brain's network of neurons, used for tasks like image recognition and classification.
- **Image Classification:** The process of identifying and categorizing elements in a digital image into predefined classes.
- **CIFAR-10 Dataset ?:** A collection of images used for evaluating machine learning models, consisting of 60,000 32x32 color images in 10 different classes.
- **Accuracy:** In the context of an ANN, the degree to which the model correctly classifies test images.
- **Efficiency:** Refers to the computational performance of the ANN, including processing speed and resource utilization.
- **Reliability:** The consistency of the ANN's performance across various datasets and under different conditions.

4.1.2 Physical System Description

This ANN system is primarily software-based, focusing on computational processes and algorithms for image classification, and consequently, does not have a physical description in the traditional sense. Therefore, its description revolves around software architecture, data processing methods, user interface design, and performance metrics, rather than physical characteristics.

4.1.3 Goal Statements

Given the input image, the goal statements are:

GS1: Predict the class of uploaded image

4.2 Solution Characteristics Specification

The instance models that govern ANN are presented in Subsection 4.2.6. The information to understand the meaning of the instance models and their derivation is also presented, so that the instance models can be verified.

4.2.1 Assumptions

This section simplifies the original problem and helps in developing the theoretical model by filling in the missing information for the physical system. The numbers given in the square brackets refer to the theoretical model [TM], general definition [GD], data definition [DD], instance model [IM], or likely change [LC], in which the respective assumption is used.

A1: All images are of a size of mxn based on the size of images of training data set

A2: The input image contains only one object

A3: Every uploaded image belongs to one of the 10 predefined classes in the CIFAR-10 dataset **?**

4.2.2 Theoretical Models

This section focuses on the general equations and laws that ANN is based on.

RefName: **TM:COE**

Label: Cost Function

Equation: $\text{Cost} = \sum_{j=0}^{n_{L-1}} \left(a_j^{(L)} - y_j \right)^2$

Description: In this project the cost function is sum of squared errors (SSE). Our goal is to minimize cost function.
$a_j^{(L)}$: Predicted value for the j-th output in the training data.
y_j: Actual value for the j-th output in the training data.

Notes: None.

Source: Wikipedia contributors (2024b)

Ref. By: GD1

Preconditions for TM:COE: None

Derivation for TM:COE: Not Applicable

4.2.3 General Definitions

This section collects the laws and equations that will be used in building the instance models.

Number	GD1
Label	**Gradient Descent**
SI Units	-
Equation	$(W, b) = (W, b) - \alpha \nabla \text{Cost}$
Description	In this project, Gradient Descent is a method used to minimize the cost function, essential for improving the neural network's accuracy. It adjusts the weights (W) and biases (b) of the network by subtracting a fraction (represented by the learning rate α) of the cost function's gradient (∇Cost) from the current values.
Source	Wikipedia contributors (2024a)
Ref. By	IM1

Detailed derivation of simplified rate of change of temperature

For updating the weights matrices and bias vectors using Gradient Descent, the chain rule is applied:

- **Last Layer Derivations:**
  - Weight: $\frac{\partial \text{Cost}}{\partial w_{jk}^{(L)}} = \frac{\partial \text{Cost}}{\partial a_j^{(L)}} \times \frac{\partial a_j^{(L)}}{\partial z_j^{(L)}} \times \frac{\partial z_j^{(L)}}{\partial w_{jk}^{(L)}} = 2(a_j^{(L)} - y_j) \times \sigma'(z_j^{(L)}) \times a_k^{(L-1)}$
  - Bias: $\frac{\partial \text{Cost}}{\partial b_j^{(L)}} = \frac{\partial \text{Cost}}{\partial a_j^{(L)}} \times \frac{\partial a_j^{(L)}}{\partial z_j^{(L)}} \times \frac{\partial z_j^{(L)}}{\partial b_j^{(L)}} = 2(a_j^{(L)} - y_j) \times \sigma'(z_j^{(L)}) \times 1$
  - Activation: $\frac{\partial \text{Cost}}{\partial a_k^{(L-1)}} = \sum_{j=0}^{n_{L-1}} \left(\frac{\partial \text{Cost}}{\partial a_j^{(L)}} \times \frac{\partial a_j^{(L)}}{\partial z_j^{(L)}} \times \frac{\partial z_j^{(L)}}{\partial a_k^{(L-1)}} \right)$
- **Other Layers Derivations:**
  - Weight: $\frac{\partial \text{Cost}}{\partial w_{km}^{(L-1)}} = \frac{\partial \text{Cost}}{\partial a_k^{(L-1)}} \times \sigma'(z_k^{(L-1)}) \times a_m^{(L-2)}$
  - Bias: $\frac{\partial \text{Cost}}{\partial b_k^{(L-1)}} = \frac{\partial \text{Cost}}{\partial a_k^{(L-1)}} \times \sigma'(z_k^{(L-1)}) \times 1$
  - Activation: $\frac{\partial \text{Cost}}{\partial a_m^{(L-2)}} = \sum_{k=0}^{n_{L-2}} \left(\frac{\partial \text{Cost}}{\partial a_k^{(L-1)}} \times \sigma'(z_k^{(L-1)} \times w_{km}^{(L-1)}) \right)$

j: This index represents a specific neuron in the current layer. It's used to refer to each individual output neuron in the last layer of the network.

k: This index is used to refer to neurons in the layer preceding (L-1) the current one. It's involved in calculations that connect the neurons of two consecutive layers.

m: The index m is used to refer to neurons or inputs from an earlier layer than k, indicating the propagation of influence from one layer to another in the neural network.

These derivations represent how the partial derivatives of the cost function with respect to the weights, biases, and activations are computed using the chain rule. For the last layer, these derivations are direct and involve the actual outputs of the network compared to the target values, which directly contribute to the cost. In other layers, the derivatives are computed as a sum over all neurons in the subsequent layer because each neuron's activation in layer L-1 contributes to the activations in layer L, thus indirectly affecting the cost. This approach ensures that the adjustments made to weights and biases through gradient descent are based on how much they contribute to the overall error in the network's predictions.

4.2.4 Data Definitions

This section collects and defines all the data needed to build the instance models. The dimension of each quantity is also given. In our model, we process RGB images, which are represented as three-dimensional (3D) matrices. However, each color channel (Red, Green, and Blue) in an RGB image is individually represented by a two-dimensional (2D) matrix. Therefore, an RGB image comprises three 2D matrices, one for each color channel, collectively forming a 3D matrix structure.

4.2.5 Data Types

This section collects and defines all the data types needed to document the models.

As outlined in the Data Definition section (Section 4.2.4), our project primarily uses images represented as 3D matrices. These matrices are comprised of numerical data, which includes pixel values of the images. This numerical aspect is fundamental for the processing and interpretation tasks the neural network undertakes, ensuring accurate image analysis and classification.

Type Name	Image
Type Def	3D matrix of numerical data
Description	Images are represented as 3D matrices, with each matrix element corresponding to pixel values across Red, Green, and Blue color channels.

4.2.6 Instance Models

This section transforms the problem defined in Section 4.1 into one which is expressed in mathematical terms. It uses concrete symbols defined in Section 4.2.4 to replace the abstract symbols in the models identified in Sections 4.2.2 and 4.2.3.

The goals GS1 are solved by IM1.

Instance Model 1

Number	IM1
Label	**Feedforward to find** $a^{(L+1)}$
Input	Raw pixel values of an image
Output	The predicted class of the input image using $a^{(L+1)} = \sigma\left(W^{(L+1)}a^{(L)} + b^{(L+1)}\right)$, $W^{(L+1)}$ and $b^{(L+1)}$ from GS1
Description	The feedforward process in neural networks involves input data propagation to output prediction or classification. This includes calculating weighted sums and biases in each layer, followed by an activation function. Network weights are initially set using random normal distributions, and biases start as zero vectors. These initial settings are crucial as they impact the network's learning and evolution during training. After each iteration the result of this part will be used to update the weights and biases. $a^{(L)}$: Output activations of the L-th layer in a neural network. $a^{(L+1)}$: Activations for the next layer, (L+1), resulting from the current layer's output. σ (sigma): The activation function applied to layer outputs, influencing neuron firing. $b^{(L+1)}$: Bias vector for the (L+1)-th layer, adjusting neuron activation thresholds. $W^{(L+1)}$: Weight matrix connecting layer L to layer (L+1), essential for learning patterns.
Sources	Wikipedia contributors (2024a)
Ref. By	GS1

4.2.7 Input Data Constraints

Table 1 shows the data constraints on the input output variables. The column for physical constraints gives the physical limitations on the range of values that can be taken by the variable. The column for software constraints restricts the range of inputs to reasonable values. The software constraints will be helpful in the design stage for picking suitable algorithms. The constraints are conservative, to give the user of the model the flexibility to experiment with unusual situations. The column of typical values is intended to provide a feel for a common scenario. The uncertainty column provides an estimate of the confidence with which the physical quantities can be measured. This information would be part of the input if one were performing an uncertainty quantification exercise.

Table 1: Input Variables

Var	Physical Constraints	Software Constraints	Typical Value	Uncertainty
Image Size	mxn dimensions	Match training data dimensions	mxn	None

5 Requirements

This section provides the functional requirements, the business tasks that the software is expected to complete, and the nonfunctional requirements, the qualities that the software is expected to exhibit.

5.1 Functional Requirements

R1: The system requires user-uploaded images in a specified format and resolution.

R2: Verify the given inputs are in correct form and satify the required physical constraints given in the Table 1. The system provides feedback by displaying the classification results for user confirmation.

R3: Classify the uploaded image from IM1.

R4: The system will provide its classification results based on the trained model, acknowledging that accuracy may not be 100%. To validate outputs, the system will incorporate a user feedback mechanism

R5: The system presents classification results in an understandable format, clearly indicating the identified image categories.

5.2 Nonfunctional Requirements

NFR1: **Accuracy** The system aims for higher classification accuracy than previous model, measurable through defined quantitative methods, such as cost function (TM:COE).

NFR2: **Usability** The system should be user-friendly and easy to navigate for users with varying levels of technical expertise.

NFR3: **Maintainability** The software should be easy to modify.

NFR4: **Portability** The system is compatible with multiple operating environment.

5.3 Rationale

In this Section, we delve into the reasoning behind key decisions in our documentation. The scope was specifically chosen to focus on the CIFAR-10 dataset **?**, targeting a manageable yet diverse range of image classifications to demonstrate the neural network's capabilities effectively. Our modeling choices, including the neural network architecture and algorithms like gradient descent, were selected for their proven effectiveness and relevance to the image classification task at hand. Assumptions were made based on realistic scenarios and limitations, ensuring that our model remains practical and applicable. The typical values chosen, such as image dimensions and parameter settings, were guided by standard practices in the field and the specifics of the CIFAR-10 dataset **?**, aiming to optimize the model's performance and accuracy. Each decision was made with careful consideration of its impact on the project's objectives and feasibility.

6 Likely Changes

LC1: A2 The system might be changed to evolve to identify multiple objects within a single image, enhancing its complexity and usability.

LC2: A3 The system can be trained on larger dataset, such as CIFAR-100 **?**, to be able to classify more objects.

7 Unlikely Changes

LC3: A1 The model architecture depends on the size of the training dataset. Consequently, the size of uploaded images should be always compatible with the size of training dataset.

8 Traceability Matrices and Graphs

The purpose of the traceability matrices is to provide easy references on what has to be additionally modified if a certain component is changed. Every time a component is changed, the items in the column of that component that are marked with an "X" may have to be modified as well. Table 2 shows the dependencies of theoretical models, general definitions, data definitions, and instance models with each other. Table 3 shows the dependencies of instance models, requirements, and data constraints on each other. Table 4 shows the dependencies of theoretical models, general definitions, data definitions, instance models, and likely changes on the assumptions.

The purpose of the traceability graphs is also to provide easy references on what has to be additionally modified if a certain component is changed. The arrows in the graphs represent

	TM:COE	GD1	IM1
TM:COE		X	
GD1	X		X
IM1		X	

Table 2: Traceability Matrix Showing the Connections Between Items of Different Sections

	IM1	4.2.7	R1	R2	R3	R4	R5
IM1					X	X	
R1							
R2		X					
R3		X				X	
R4		X			X		
R5							

Table 3: Traceability Matrix Showing the Connections Between Requirements and Instance Models

	A1	A2	A3
TM:COE	X		X
GD1	X		
IM1	X		
LC1		X	
LC2			X

Table 4: Traceability Matrix Showing the Connections Between Assumptions and Other Items

dependencies. The component at the tail of an arrow is depended on by the component at the head of that arrow. Therefore, if a component is changed, the components that it points to should also be changed.

9 Development Plan

In this section, we outline our approach to building the software. The plan is divided into phases, prioritizing core functionalities first.

- **Phase1:** Initial development focuses on basic functionalities like image uploading and primary classification training using the CIFAR-10 dataset ?. This phase ensures the system's ability to categorize images correctly.

- **Phase2:** The aim is to enhance the model's accuracy and efficiency. However, the extent of improvement will be explored during the development process, as we experiment with algorithm optimizations and neural network architectures.
- **For Coursework:** The primary goal within the course is to establish the foundational image classification functionality (Phase 1), while efforts to enhance accuracy (Phase 2) will be attempted but are subject to the scope and limitations of the coursework.

10 Values of Auxiliary Constants

The following are the values for the symbolic parameters used throughout this report:

- **Image Dimensions (mxn):** 32x32 (Standard dimensions of images in the CIFAR-10 dataset **?**.)
- **Maintainability Fraction (FRACTION):** 0.25 (Indicating that any change should require no more than 25% of the original development effort.)

These values are set here for ease of maintenance and to ensure consistency across the report.

References

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical Report, 2009.

W. Spencer Smith and Lei Lai. A new requirements template for scientific computing. In J. Ralyté, P. Ȧgerfalk, and N. Kraiem, editors, *Proceedings of the First International Workshop on Situational Requirements Engineering Processes – Methods, Techniques and Tools to Support Situation-Specific Requirements Engineering Processes, SREP'05*, pages 107–121, Paris, France, 2005. In conjunction with 13th IEEE International Requirements Engineering Conference.

Wikipedia contributors. Artificial neural network — Wikipedia, the free encyclopedia, 2024a. URL https://en.wikipedia.org/wiki/Artificial_neural_network. [Online; accessed 14-February-2024].

Wikipedia contributors. Residual sum of squares — Wikipedia, the free encyclopedia, 2024b. URL https://en.wikipedia.org/wiki/Residual_sum_of_squares. [Online; accessed 14-February-2024].