Software Requirements Specification for ANN: Artificial Neural Network

Tanya Djavaherpour

February 3, 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

In this document, we do not use any specific unit.

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 the heat transfer literature and with existing documentation for solar water heating systems. The symbols are listed in alphabetical order.

symbol	**unit**	**description**
a^L	–	Weighted input to neurons of layer L
W^L	–	Weight matrix of layer L in neural networks
b^L	–	Bias vector of layer L in neural networks
σ	–	Activation function
α	–	Learning rate in gradient descent
∇	–	Gradient operator
y_j	–	Target output for neuron j

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. 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. 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 32x32

A2: The input image contains only one object

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

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: https://en.wikipedia.org/wiki/Residual_sum_of_squares

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	https://en.wikipedia.org/wiki/Artificial_neural_network
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. The data we use to build our model is image. We use RGB images whcih means they have 3D matrices. Each matrix is a 2D matrix for three channels of Red(R), Green(G), and Blue(B).

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	https://en.wikipedia.org/wiki/Artificial_neural_network
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

4.2.8 Properties of a Correct Solution

A correct solution must exhibit [fill in the details —TPLT]. [These properties are in addition to the stated requirements. There is no need to repeat the requirements here. These additional properties may not exist for every problem. Examples include conservation laws (like conservation of energy or mass) and known constraints on outputs, which are usually summarized in tabular form. A sample table is shown in Table 2 —TPLT]

Table 2: Output Variables

Var	Physical Constraints
T_W	$T_{\text{init}} \leq T_W \leq T_C$ (by A??)

[This section is not for test cases or techniques for verification and validation. Those topics will be addressed in the Verification and Validation plan. —TPLT]

5 Requirements

[The requirements refine the goal statement. They will make heavy use of references to the instance models. —TPLT]

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: [Requirements for the inputs that are supplied by the user. This information has to be explicit. —TPLT]

R2: [It isn't always required, but often echoing the inputs as part of the output is a good idea. —TPLT]

R3: [Calculation related requirements. —TPLT]

R4: [Verification related requirements. —TPLT]

R5: [Output related requirements. —TPLT]

[Every IM should map to at least one requirement, but not every requirement has to map to a corresponding IM. —TPLT]

5.2 Nonfunctional Requirements

[List your nonfunctional requirements. You may consider using a fit criterion to make them verifiable. —TPLT] [The goal is for the nonfunctional requirements to be unambiguous, abstract and verifiable. This isn't easy to show succinctly, so a good strategy may be to give a "high level" view of the requirement, but allow for the details to be covered in the Verification and Validation document. —TPLT] [An absolute requirement on a quality of the system is rarely needed. For instance, an accuracy of 0.0101 % is likely fine, even if the requirement is for 0.01 % accuracy. Therefore, the emphasis will often be more on describing now well the quality is achieved, through experimentation, and possibly theory, rather than meeting some bar that was defined a priori. —TPLT] [You do not need an entry for correctness in your NFRs. The purpose of the SRS is to record the requirements that need to be satisfied for correctness. Any statement of correctness would just be redundant. Rather than discuss correctness, you can characterize how far away from the correct (true) solution you are allowed to be. This is discussed under accuracy. —TPLT]

NFR1: **Accuracy** [Characterize the accuracy by giving the context/use for the software. Maybe something like, "The accuracy of the computed solutions should meet the level needed for <engineering or scientific application>. The level of accuracy achieved by ProgName shall be described following the procedure given in Section X of the Verification and Validation Plan." A link to the VnV plan would be a nice extra. —TPLT]

NFR2: **Usability** [Characterize the usability by giving the context/use for the software. You should likely reference the user characteristics section. The level of usability achieved by the software shall be described following the procedure given in Section X of the Verification and Validation Plan. A link to the VnV plan would be a nice extra. —TPLT]

NFR3: **Maintainability** [The effort required to make any of the likely changes listed for ProgName should be less than FRACTION of the original development time. FRACTION is then a symbolic constant that can be defined at the end of the report. —TPLT]

NFR4: **Portability** [This NFR is easier to write than the others. The systems that ProgName should run on should be listed here. When possible the specific versions of the potential operating environments should be given. To make the NFR verifiable a statement could be made that the tests from a given section of the VnV plan can be successfully run on all of the possible operating environments. —TPLT]

- Other NFRs that might be discussed include verifiability, understandability and reusability.

5.3 Rationale

[Provide a rationale for the decisions made in the documentation. Rationale should be provided for scope decisions, modelling decisions, assumptions and typical values. —TPLT]

6 Likely Changes

LC1: [Give the likely changes, with a reference to the related assumption (aref), as appropriate. —TPLT]

7 Unlikely Changes

LC2: [Give the unlikely changes. The design can assume that the changes listed will not occur. —TPLT]

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 3 shows the dependencies of theoretical models, general definitions, data definitions, and instance models with each other. Table 4 shows the dependencies of instance models, requirements, and data constraints on each other. Table 5 shows the dependencies of theoretical models, general definitions, data definitions, instance models, and likely changes on the assumptions.

[You will have to modify these tables for your problem. —TPLT]

[The traceability matrix is not generally symmetric. If GD1 uses A1, that means that GD1's derivation or presentation requires invocation of A1. A1 does not use GD1. A1 is "used by" GD1. —TPLT]

[The traceability matrix is challenging to maintain manually. Please do your best. In the future tools (like Drasil) will make this much easier. —TPLT]

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 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. Figure **??** shows the dependencies of theoretical models, general definitions, data definitions, instance models, likely changes, and assumptions on each other. Figure **??** shows the dependencies of instance models, requirements, and data constraints on each other.

9 Development Plan

[This section is optional. It is used to explain the plan for developing the software. In particular, this section gives a list of the order in which the requirements will be implemented.

	TM??	TM??	TM??	GD??	GD??	DD??	DD??	DD??	DD??	IM??	IM??	IM?
TM??												
TM??			X									
TM??												
GD??												
GD??	X											
DD??				X								
DD??				X								
DD??												
DD??								X				
IM??					X	X	X				X	
IM??					X		X		X	X		
IM??		X										
IM??		X	X				X	X	X		X	

Table 3: Traceability Matrix Showing the Connections Between Items of Different Sections

	IM??	IM??	IM??	IM??	4.2.7	R??	R??
IM??		X				X	X
IM??	X			X		X	X
IM??						X	X
IM??		X				X	X
R??							
R??						X	
R??					X		
R2	X	X				X	X
R??	X						
R??		X					
R??			X				
R??				X			
R4			X	X			
R??		X					
R??		X					

Table 4: Traceability Matrix Showing the Connections Between Requirements and Instance Models

	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??	A??
TM??	X																		
TM??																			
TM??																			
GD??		X																	
GD??			X	X	X	X													
DD??							X	X	X										
DD??			X	X						X									
DD??																			
DD??																			
IM??											X	X		X	X	X			X
IM??												X	X			X	X	X	
IM??														X					X
IM??													X					X	
LC??				X															
LC??								X											
LC??									X										
LC??											X								
LC??												X							
LC??															X				

Table 5: Traceability Matrix Showing the Connections Between Assumptions and Other Items

In the context of a course this is where you can indicate which requirements will be implemented as part of the course, and which will be "faked" as future work. This section can be organized as a prioritized list of requirements, or it could should the requirements that will be implemented for "phase 1", "phase 2", etc. —TPLT]

10 Values of Auxiliary Constants

[Show the values of the symbolic parameters introduced in the report. —TPLT]

[The definition of the requirements will likely call for SYMBOLIC_CONSTANTS. Their values are defined in this section for easy maintenance. —TPLT]

[The value of FRACTION, for the Maintainability NFR would be given here. —TPLT]

References

W. Spencer Smith and Lei Lai. A new requirements template for scientific computing. In J. Ralyté, P. Àgerfalk, 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.

[The following is not part of the template, just some things to consider when filing in the template. —TPLT]
[Grammar, flow and LaTeXadvice:

- For Mac users `*.DS_Store` should be in `.gitignore`
- LaTeX and formatting rules
  - Variables are italic, everything else not, includes subscripts (link to document)
    - Conventions
    - Watch out for implied multiplication
  - Use BibTeX
  - Use cross-referencing
- Grammar and writing rules
  - Acronyms expanded on first usage (not just in table of acronyms)
  - "In order to" should be "to"

—TPLT]
[Advice on using the template:

- Difference between physical and software constraints
- Properties of a correct solution means *additional* properties, not a restating of the requirements (may be "not applicable" for your problem). If you have a table of output constraints, then these are properties of a correct solution.
- Assumptions have to be invoked somewhere
- "Referenced by" implies that there is an explicit reference
- Think of traceability matrix, list of assumption invocations and list of reference by fields as automatically generatable
- If you say the format of the output (plot, table etc), then your requirement could be more abstract

—TPLT]

Appendix — Reflection

The information in this section will be used to evaluate the team members on the graduate attribute of Lifelong Learning. Please answer the following questions:

1. Which of the courses you have taken, or are currently taking, will help your team to be successful with your capstone project.

2. What knowledge and skills will the team collectively need to acquire to successfully complete this capstone project? Examples of possible knowledge to acquire include domain specific knowledge from the domain of your application, or software engineering knowledge, mechatronics knowledge or computer science knowledge. Skills may be related to technology, or writing, or presentation, or team management, etc. You should look to identify at least one item for each team member.

3. For each of the knowledge areas and skills identified in the previous question, what are at least two approaches to acquiring the knowledge or mastering the skill? Of the identified approaches, which will each team member pursue, and why did they make this choice?