Software Requirements Specification for OCRacle: Latin Alphabet Character Recognition

Phillip Tran

April 17, 2025

Contents

Revision History

Date	Version	Notes
January 27, 2025	1.0	Initial document creation
April 6, 2025	2.0	Addressed feedback from Dr. Spencer Smith and Hussein Saad

1 Reference Material

This section records information for easy reference.

1.1 Table of Units

This document does not use SI (Système International d'Unités) as the unit system because the problem domain is not a physical system.

1.2 Table of Symbols

The table that follows summarizes the symbols used in this document along with their units. The symbols are listed in alphabetical order.

symbol	**unit**	**description**
$I_{n \times m}$	N/A	Matrix representing an input image of n rows and m columns provided by the end user
L	N/A	The number of distinct labels that the model can predict
$P_{1 \times L}$	N/A	Probability vector representing the output of the model, with L columns representing the probability of each character
P_{pred}	N/A	The character predicted by the model.
$T_{w \times w}$	N/A	Matrix representing a $w \times w$ pixel image from the training data provided by the technical user
g_t	N/A	The gradient of the cost function with respect to the weights at time t
m_t	N/A	In the context of the Adam optimization algorithm, the first moment estimate at time t
t	N/A	In the context of the Adam optimization algorithm, the time step
v_t	N/A	In the context of the Adam optimization algorithm, the second moment estimate at time t
w	N/A	The number of rows/columns in the training dataset image
α	N/A	Learning rate for the model
ϵ	N/A	Small value to prevent division by zero
θ	N/A	Vector representing the weights of the model
$\hat{y}_i$	N/A	The predicted probability of character i
y_i	N/A	The actual probability of character i

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
OCRacle	Name of the Project
TM	Theoretical Model

2 Introduction

Researchers analyzing physical print documents such as newspapers, books, and letters often need a means of digitizing the text in these documents. This enables them to search and analyze the text data more efficiently. Especially in the case of historical documents, digitizing the text can help preserve the information contained in these documents.

Optical Character Recognition (OCR) is a technology that allows for the extraction of text information from scanned documents, images, and other optical formats where text may be present. This digitalization process enables researchers to use computer programs to find trends and patterns in the digitized text.

The OCRacle project aims to develop a program that can recognize and classify a single Latin alphabet character in an image.

The following section describes the purpose of the document, the scope of requirements, characteristics of the intended reader, and the organizational roadmap of the document.

2.1 Purpose of Document

This document serves as a software requirements specification for the OCRacle project. This includes the general system description, problem description, solution characteristics, and requirements. This document will be used a reference for building the solution.

2.2 Scope of Requirements

This program is intended for classifying a single handwritten Latin alphabet character in an image. Handwritten cursive characters are not an expected input of this program.

2.3 Characteristics of Intended Reader

The intended reader of this document should have knowledge equivalent to the following coursework:

- MATH 1B03: Linear Algebra
- COMPSCI 4ML3: Introduction to Machine Learning

2.4 Organization of Document

After the brief introduction, the document details the general system description, problem description, solutions characteristic specification, and requirements.

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

We can take two perspectives on the system context. The first is the technical user, who interfaces with the OCRacle program to produce a model that can predict the character in an image. The second is the end user, who uses the OCRacle model to predict the character in an input image.

Figure 1: System Context of End User

- Technical User Responsibilities:
  - Provide an image dataset to train the model following A1, A2, and A3.
- OCRacle Responsibilities:
  - Detect incompatible input.
  - Train the model on the image dataset provided by the user.
  - Output a trained model that can predict the character in an image.
- End User Responsibilities:

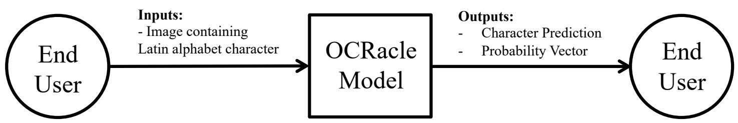


Figure 2: System Context of Technical User

- Provide an image following A3.
- Run the program on a compatible platform.
- Interface with the program.

- OCRacle Responsibilities:
  - Detect incompatible input.
  - Provide an interface to view the predicted character.
  - Provide an interface to view the probability distribution.

3.2 User Characteristics

Both the technical user and the end user of OCRacle should have an a basic understanding of the command line to setup the program. At a minimum, the users should have the knowledge contained within the MIT Missing Semester course.

The provided user manual should be approachable for users of this skill level and allow them to train and use the model effectively, even if they are not familiar with the technical details of the program.

3.3 System Constraints

The input image must contain a single Latin alphabet character. The system will be unable to identify multiple characters in a single image. It will also be unable to determine if there are no identifiable characters in the image.

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 the instance models.

4.1 Problem Description

OCRacle is intended to solve the problem of extracting text information from a scanned document, image, and other optical formats where text may be present, such that this textual data can be used for further analysis.

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:

- **OCR:** Optical Character Recognition, the process of extracting text
- **EMNIST:** Extended MNIST, a dataset of handwritten characters
- **Latin Alphabet:** The alphabet used in the English language
- **Character:** A single letter in the Latin alphabet
- **Image:** A 2D array of pixel values
- **Pixel:** The smallest unit of a digital image
- **Probability Vector:** A vector representing the likelihood of each character
- **Model:** A trained machine learning model
- **Prediction:** The output of the model
- **Preprocessing:** The process of preparing the image for input into the model
- **Label:** The correct character associated with an image
- **Training:** The process of teaching the model to predict characters
- **Training Data:** The dataset used to train the model
- **Downsampling:** The process of reducing the dimensions of an image by removing pixels while preserving the important features of the image

4.1.2 Physical System Description

The physical system of OCRacle, as shown in Figure 3, includes the following elements:

PS1: For training the model, the system requires a dataset of images of Latin alphabet characters, alongside a corresponding label for each image.

PS2: For using the model as an end user, the system requires an image of a single Latin alphabet character.

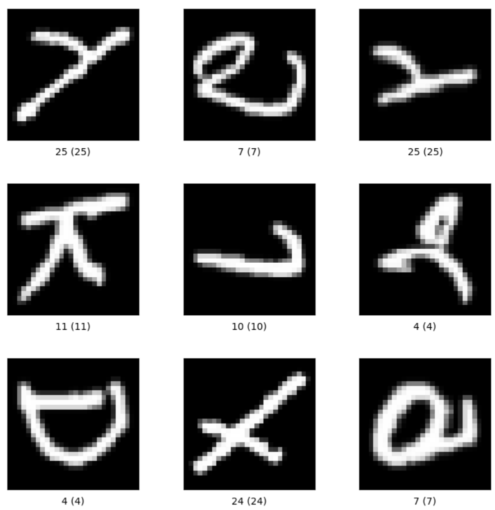


Figure 3: A sample of the EMNIST dataset, representing the Latin alphabet

4.1.3 Goal Statements

Given the training data provided by the technical user, the goal statements are:

GS1: Produce a trained model that can predict the character in an image.

Given an image containing a single Latin alphabet character from the end user, the goal statements are:

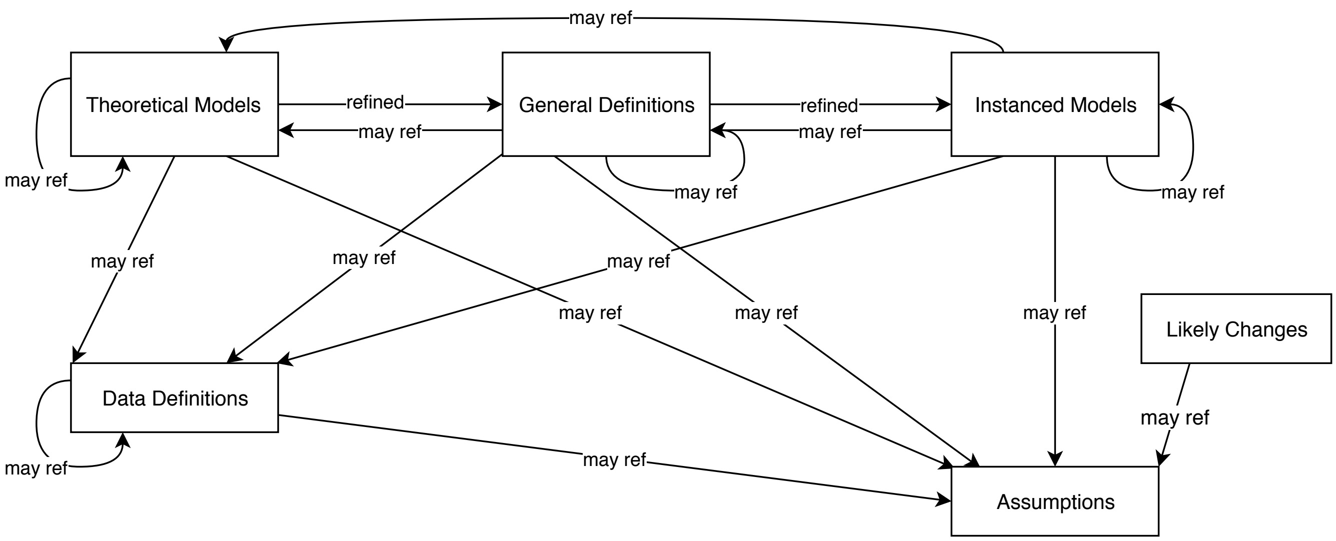
GS2: Predict the character.

GS3: Produce a probability distribution representing the likelihood of each character.

4.2 Solution Characteristics Specification

This section specifies the information in the solution domain of the system to be developed. This section is intended to express what is required in such a way that analysts and stakeholders get a clear picture, and the latter will accept it. The purpose of this section is to reduce the problem into one expressed in mathematical terms. Mathematical expertise is used to extract the essentials from the underlying physical description of the problem, and to collect and substantiate all physical data pertinent to the problem.

The relationships between the parts of the document are show in the following figure. In this diagram "may ref" has the same role as "uses" above. The figure adds "Likely Changes," which are able to reference (use) Assumptions.



The instance models that govern OCRacle are presented in Subsection 4.2.5. 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: The model will be trained on the EMNIST letters dataset where each image is 28x28 pixels as required by IM1. The dataset also treats uppercase and lowercase letters as the same character, so the model will be trained on both uppercase and lowercase letters.

A2: The input images and training images will contain only a single Latin alphabet character per image, as required by IM1 and IM3.

A3: All input images from the user undergo preprocessing such that they are in the same format as the training data, where TM1 and TM2 are applied in IM2.

4.2.2 Theoretical Models

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

RefName: **TM:BI**

Label: Bicubic Interpolation

Equation: $f(x,y) = \sum_{i=0}^{3} \sum_{j=0}^{3} a_{ij} x^i y^j$

Description: Bicubic interpolation is a method of interpolating data points on a 2D grid. The method uses a 4x4 grid of data points to estimate the value of a point within the grid. This is useful for downsampling an image from a higher resolution (e.g. 128x128) to a lower resolution (e.g. 28x28).

Notes: None.

Source: https://en.wikipedia.org/wiki/Bicubic_interpolation

Ref. By: A3

Preconditions for TM:BI: None

Derivation for TM:BI: Not Applicable

RefName: **TM:N**

Label: Normalization

Equation: $f(x) = \begin{cases} 0, & \text{if } \max(X) = \min(X) \\ \frac{x - \min(X)}{\max(X) - \min(X)}, & \text{otherwise} \end{cases}$

Description: Normalization is a method of scaling data to a fixed range. This is useful for ensuring that the input data to a machine learning model is within a certain range. For instance, the pixel values of an image are typically normalized to the range [0, 1], where 0 represents black and 1 represents white. We prevent a divide by zero case by checking if the minimum and maximum values are the same before performing the normalization.

Notes: None.

Source: https://en.wikipedia.org/wiki/Normalization_(image_processing)

Ref. By: A3, IM2

Preconditions for TM:N: None

Derivation for TM:N: Not Applicable

RefName: **TM:ADAM**

Label: ADAM Optimization

Equation:
$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$
$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2$
$\hat{m}_t = \frac{m_t}{1-\beta_1^t}$
$\hat{v}_t = \frac{v_t}{1-\beta_2^t}$
$\theta_t = \theta_{t-1} - \alpha \frac{\hat{m}_t}{\sqrt{\hat{v}_t}+\epsilon}$

Description: ADAM Optimization is used to train machine learning models. ADAM stands for Adaptive Moment Estimation, which means that the algorithm adapts the learning rate during training. The algorithm uses the first and second moments (m_t and v_t) of the gradients (g_t) to update the parameters θ_t. The hyperparameters β_1, β_2, α, and ϵ control the behavior of the algorithm. β_1 and β_2 control the exponential decay rates of the first and second moments, α is the learning rate, and ϵ is a small value to prevent division by zero.

Notes: None.

Source: Kingma and Ba (2014)

Ref. By: IM1, DD3, DD4, DD5

Preconditions for TM:ADAM: None

Derivation for TM:ADAM: Not Applicable

RefName: **TM:CELF**

Label: Cross-Entropy Loss Function

Equation: $L(y, \hat{y}) = -\sum_i y_i \log(\hat{y}_i)$

Description: The cross-entropy loss function is used to measure the difference between the predicted probability distribution $\hat{y}$ and the true probability distribution y. A greater loss is incurred when the predicted probability distribution is further from the true distribution. Heavier penalties occur when the prediction dictates there to be a near-equal likelihood that the true label could belong to multiple classes.

Notes: None.

Source: Mao (2023)

Ref. By: A3, IM1, DD6, DD7

Preconditions for TM:CELF: None

Derivation for TM:CELF: Not Applicable

RefName: **TM:ReLU**

Label: Rectified Linear Unit (ReLU)

Equation: $f(x) = \max(0, x)$

Description: The function returns 0 if x is negative. Otherwise, the function will return x. This function is used in neural networks as an activation function. An activation function determines the output of a node in a neural network.

Notes: None.

Source: Bai (2022)

Ref. By: GD1

Preconditions for TM:ReLU: None

Derivation for TM:ReLU: Not Applicable

RefName: **TM:SM**

Label: Softmax

Equation: $f(x)_i = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}}$

Description: The softmax function takes in a vector of n real numbers and returns a vector of the same length. The exponential function is applied to each element of the input vector, and the resulting vector is divided by the sum of the exponential values. This achieves a probability distribution over the input vector.

Notes: None.

Source: Gao and Pavel (2018)

Ref. By: GD1

Preconditions for TM:SM: None

Derivation for TM:SM: 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	**Neural Network**
SI Units	N/A
Equation	Input Layer: $x = I_{n\times m}$ Hidden Layer: $h = \text{ReLU}(W_1 x + b_1)$ Softmax Layer: $\hat{y} = \text{Softmax}(a)$
Description	A neural network consists of multiple layers of nodes. The input layer takes in the image data $I_{n\times m}$. The hidden layer applies a linear transformation $W_1 x + b_1$ to the input data, followed by a non-linear activation function (ReLU). The output layer applies the softmax function to the output of the hidden layer to produce a probability distribution $\hat{y}$ over the characters. The weights W_1 and biases b_1 are learned during training.
Source	Guo (2018)
Ref. By	IM1, IM3

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.

Number	DD1
Label	**Input Image**
Symbol	$I_{n \times m}$
SI Units	N/A
Equation	N/A
Description	I is a 2D array of pixel values representing an image. The image has n rows and m columns. Each pixel value is an integer between 0 and 255.
Sources	N/A
Ref. By	IM2, TM1, TM2

Number	DD2
Label	**Training Image**
Symbol	$I_{n \times m}$
SI Units	N/A
Equation	N/A
Description	T is a 2D array of pixel values representing an image. The image has n rows and m columns. Each pixel value is either 0 for black or 1 for white.
Sources	N/A
Ref. By	IM1

Number	DD3
Label	**Learning Rate**
Symbol	α
SI Units	N/A
Equation	N/A
Description	α is a hyperparameter that controls how much the model weights are updated during training. With a small learning rate, the model will learn slowly, but with a large learning rate, the model may never find the optimal solution.
Sources	Kingma and Ba (2014)
Ref. By	IM1, TM3

Number	DD4
Label	**Parameters**
Symbol	θ_t
SI Units	N/A
Equation	N/A
Description	θ_t is a vector of parameters that the model uses to make predictions. The parameters are updated during training to minimize the loss function. The t subscript indicates the parameters at time step t.
Sources	Kingma and Ba (2014)
Ref. By	IM1, TM3

Number	DD5
Label	**Gradient**
Symbol	g_t
SI Units	N/A
Equation	N/A
Description	g_t is a vector of the gradients of the loss function with respect to the parameters. The gradients are used to update the parameters during training. The t subscript indicates the gradients at time step t.
Sources	Kingma and Ba (2014)
Ref. By	IM1, TM3

Number	DD6
Label	**Probability Distribution**
Symbol	$\hat{y}_i$
SI Units	N/A
Equation	N/A
Description	In cross-entropy loss, the model outputs a probability distribution $\hat{y}$ over the possible classes. The probability distribution is used to calculate the loss function.
Sources	Mao (2023)
Ref. By	IM1, TM4

Number	DD7
Label	**True Label**
Symbol	y_i
SI Units	N/A
Equation	N/A
Description	y_i is the true label of the input image. The true label is used to calculate the loss function.
Sources	Mao (2023)
Ref. By	IM1, TM4

4.2.5 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 goal GS1 is achieved by IM1, and the goals GS2 and GS3 are achieved by IM2 and IM3.

Number	IM1
Label	**ADAM Optimization with Cross-Entropy Loss**
Input	Training dataset $T_{28\times28}$, learning rate α, parameters θ_{t-1}, gradients g_{t-1}, moments m_{t-1}, v_{t-1} hyperparameters β_1, β_2, ϵ, true labels y_i, predicted probability distribution $\hat{y}_i$
Output	Trained parameters θ_t[a]
Description	Given the training dataset, the parameters, the gradients, and the moments from the previous time step, the ADAM optimization algorithm updates the parameters to minimize the cross-loss entropy function. The hyperparameters β_1, β_2, α, and ϵ control the behavior of the algorithm. The cross-entropy loss function is integrated into ADAM optimization to measure the difference between the predicted probability distribution and the true labels. The gradients of the loss function are used to update the parameters.
Sources	Kingma and Ba (2014), Mao (2023)
Ref. By	A1, A2, R5

[a] Such that, at time t, the cross-entropy loss function is minimized.

Number	IM2
Label	**Pre-processing of Input Image**
Input	$I_{n\times m}$ from DD1
Output	$T_{28\times28}$
Description	T is the pre-processed image of I. The image is resized to 28x28 pixels using TM1, and the pixel values are normalized to the range [0, 1] using TM2.
Sources	N/A
Ref. By	A1, A3, R2

Number	IM3
Label	**Character Prediction Model**
Input	$T_{28\times 28}$ from IM2
Output	$P_{1\times L}$, P_{pred}
Description	P is the probability distribution output by the model, where P_i represents the likelihood of character i. P_{pred} is the character with the highest probability in P, where ties are broken by choosing the first character in the Latin alphabet.
Sources	N/A
Ref. By	A2, R4, R3

4.2.6 Input Data Constraints

Table 2 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.

The specification parameters in Table 2 are listed in Table 4.

Table 2: Input Variables

Var	**Physical Constraints**	**Software Constraints**	**Typical Value**	**Uncertainty**
$I_{n\times m}$	N/A	$0 \leq I_{ij} \leq 255**$	N/A	N/A
$T_{28\times 28}$*	N/A	$0 \leq I_{ij} \leq 1$	N/A	N/A

(*) The training images are restricted to a 28x28 pixel format because the EMNIST dataset is in this format.

(**) The pixel values are restricted to the range [0, 255] because they are typically represented as 8-bit integers.

Table 4: Specification Parameter Values

Var	**Value**
n_{min}	28
n_{max}	2048
m_{min}	28
m_{max}	2048

4.2.7 Properties of a Correct Solution

As reflected in Table 6, the correct solution from the model must exhibit the following properties:

- The resulting probability distribution must sum to 1.
- The predicted character must be one of the Latin alphabet characters from A to Z.

Table 6: Output Variables

Var	**Physical Constraints**
$P_{1\times L}$	$0 \leq P_i \leq 1$
P_{pred}	$P_{pred} \in \{A, B, C, ..., Z\}$

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: (Prediction) The program accepts an image from the user in JPEG or PNG format.

R2: (Prediction) The input image is processed such that it can be used for classification by the program as described in IM1.

R3: (Training) The OCRacle project produces a model that predicts the character in an input image as per IM3.

R4: (Training) The model produced by the program outputs a probability distribution to verify the prediction in a human readable-format as per IM3.

R5: (Prediction) The program outputs the most likely character prediction in a human-readable format as per IM3.

5.2 Nonfunctional Requirements

NFR1: **Accuracy** The accuracy of the the software shall exceed the previous OAR project. Since the OAR project provides an overall accuracy measurement, the overall accuracy of the OCRacle software shall be measured using the same method as the OAR project as described in section 6.2 of the VnV Plan (Tran, 2025).

NFR2: **Usability** A user with the skills specified in Section 4.2.2 of the VnV Plan should be capable of executing all the tasks described in that same section. (Tran, 2025)

NFR3: **Maintainability** The code should be highly modular, such that each processing step can be easily understood and re-implemented without disrupting the other previous and following processing steps.

NFR4: **Portability** The program will be compatible with Windows, MacOS, and Linux operation systems. Any modern computers capable of running the operating systems mentioned above should be able to run the program.

5.3 Rationale

A1 is necessary to provide a dataset for training the model. EMNIST provides a large dataset of pre-labeled handwritten Latin alphabet characters in both uppercase and lowercase, making it a suitable choice for this project.

A2 restricts which characters the program will focus on recognizing. This also makes comparing the accuracy of OCRacle to the OAR project more meaningful as NRF1 outlines.

Lastly, A3 is necessary to ensure that the input image is in a format that the model can use for classification.

6 Likely Changes

LC1: The program may be modified to work non-Latin alphabet characters instead. For instance, Chinese character recognition may become the focus of the program.

LC2: The program may be extended to recognize Latin alphabet punctuation characters.

7 Unlikely Changes

UC1: The program will not be expanded to recognize full words using Latin alphabet characters.

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 8 shows the dependencies of theoretical models, general definitions, data definitions, and instance models with each other. Table 9 shows the dependencies of instance models, requirements, and data constraints on each other. Table 10 shows the dependencies of theoretical models, general definitions, data definitions, instance models, and likely changes on the assumptions.

	TM1	TM2	TM3	TM4	TM5	TM6	GD1	IM1	IM2
TM1									X
TM2									X
TM3								X	
TM4								X	
TM5							X		
TM6							X		
GD1								X	
IM1			X	X			X		
IM2	X	X							
IM3						X	X	X	

Table 8: Traceability Matrix Showing the Connections Between Items of Different Sections

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

	IM1	IM2	IM3
R1		X	
R2			X
R3	X		
R4			X
R5	X		X

Table 9: Traceability Matrix Showing the Connections Between Requirements and Instance Models

	TM1	TM2	TM3	TM4	TM5	TM6	GD1	IM1	IM2
A1								X	
A2								X	
A3	X	X							X

Table 10: Traceability Matrix Showing the Connections Between Assumptions and Other Items

to should also be changed. Figure 10 shows the dependencies of theoretical models, general definitions, data definitions, instance models, likely changes, and assumptions on each other. Figure 9 shows the dependencies of instance models, requirements, and data constraints on each other.

9 Values of Auxiliary Constants

There are no auxiliary constants for this project.

References

Yuhan Bai. Relu-function and derived function review. *SHS Web of Conferences*, 144, 2022.

Bolin Gao and Lacra Pavel. On the properties of the softmax function with application in game theory and reinforcement learning. 2018.

Tianmei Guo. Simple convolutional neural network on image classification. 2018.

Diederik P. Kingma and Jimmy Lei Ba. Adam: A method for stochastic optimization. 2014.

Anqi Mao. Cross-entropy loss functions: Theoretical analysis and applications. 2023.

Phillip Tran. Verification and validation plan. https://github.com/ptrandev/OCRacle/blob/main/docs/VnVPlan/VnVPlan.pdf, 2025.