



UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática



**TFG del Grado en Ingeniería
Informática**

**Sky Classification through
Supervised Machine Learning
Technical Documentation**



Presented by Bardia Filizadeh
in the University of Burgos — June 12, 2024
Tutors: Diego Granados López - Nuño Basurto
Hornillos

Contents

Contents	i
List of Figures	iii
List of Tables	v
Appendix A Software Project Plan	1
A.1 Introduction	1
A.2 Time Planning	1
A.3 Project Planning	3
A.4 Viability Study	4
Appendix B Requirements Specification	7
B.1 Introduction	7
B.2 General Objectives	7
B.3 Catalogue of Requirements	7
B.4 Requirements Specification	9
Appendix C Design	13
C.1 Introduction	13
C.2 Data Design	13
C.3 Procedural Design	14
C.4 Architectural design	15
Appendix D Programmer Manual	17
D.1 Directory Structure	17
D.2 Programmer Manual	18
D.3 Project compilation, installation and execution	25

Appendix E User Documentation	29
E.1 Software Requirements	29
E.2 Installation	31
E.3 User Manual	32
Appendix F Curricular Sustainability Annex	33
F.1 Introduction	33
Bibliography	37

List of Figures

A.1	Simple Example of Issues in GitHub	2
A.2	Project Plan overview	3
B.1	Overview of how the functional requirements correlate	9
B.2	Use case diagram of the model	10
D.1	Directory Structure Standard of the Dataset	19
D.2	ImageDatastore method provided by MATLAB that loads the dataset	19
D.3	Function that sets the labels of the samples by splitting their file path until the numerical value of the class folder has been acquired	20
D.4	Parameter that will decide whether panchromatic images will be used (value == 1) or RGB images (value == 0)	20
D.5	Second set of training that would usually be performed using the RGB channel	21
D.6	Hidden Neurons Quantity and Training Functions used arrays	21
D.7	Setup Network Function of the MATLAB script	22
D.8	Array structure that contains all the results that are going to be saved	22
D.9	SaveResults function of the MATLAB script, it opens a csv file, or creates a new one if it doesn't exist and writes the newly acquired results into it.	23
D.10	Variable that decides whether RGB or panchromatic images will be used for prediction	24
D.11	Prediction dataset directory structure	24
D.12	ImageDatastore function call to load the prediciton dataset	25
D.13	Directory overview of MATLAB	25
D.14	Run buttons in MATLAB	26

D.15 Example of how the directory content should look like after running both scripts	27
E.1 Step 1 of MATLAB installation	30
E.2 MATLAB toolboxes used for this project	31
E.3 Matlab workspace folder location, TFG folder should be inserted there	31
E.4 Example current directory path of MATLAB	32
F.1 Example graphic of components that can make up sustainable code	35

List of Tables

B.1	Programmer use case 1.	11
B.2	Programmer use case 2.	12
C.1	Technical specifications of the sky scanner [5]	14
C.2	Technical specifications of the sky camera [5]	14

Appendix A

Software Project Plan

A.1 Introduction

Project planning is an essential step towards keeping a consistent and structured workflow. It is even more so important when working in a team towards a shared goal. A good project plan can allow an easy overview of tasks and deadlines. In this appendix, the planning of this project will be presented.

A.2 Time Planning

As with every particular problem that can be encountered, the measures to solve that problem can be different from one case to another. In the same sense, the time planning for this project specifically did not require much of a complex structure or intensive deadline management. Therefore, the tools used to track tasks were rather simple, and as mentioned in the "Techniques and Tools" chapter of the report, the use of GitHub was the most common when it came to project planning.

- **Issues - GitHub** : Issues on GitHub, are specified tasks that need to be realised either within or outside of the project. The usage of these issues, allows for a quick overview of how many tasks are left, and the procedure of finishing these tasks can be documented very well upon either upload of the implementation of the task, or simply once the task had been done.

- **Milestones - GitHub** : In the same sense as issues, milestones represent more overarching goals within the project that either span multiple issues or require an extended period of time to complete.



Figure A.1: Simple Example of Issues in GitHub

1.5 As can be seen in the figure A.1 above, the basic layout of an issue is a short description of the issue in the title, while a more extensive description can be written inside the issue's specifics. The shown example is a very simplified version of regular issues used for bigger projects, lacking a coherent numeric identifier for each issue to better reference it during discussions and updates.

A.3 Project Planning

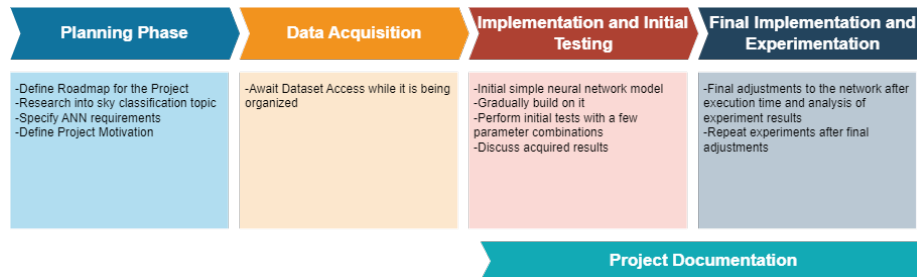


Figure A.2: Project Plan overview

Planning Phase

Timeline 26.02.2024 - 20.03.2024

With any project, the first thing to start with is planning the future roadmap for the project itself. The planning starts off initially by commencing research into the sky classification topic and similar works that had delved into the same or similar issues before. Once enough research had been done, it was time to specify what the specifics of the ANN were going to be.

Initial proposals were to identify the networks parameters and which ones would be used for experimentation. Additionally it is important to substantiate the works motivation for existing and emphasize itself as a study that can set itself apart from other works.

Data acquisition

Timeline 20.03.2024 - 08.04.2024

As the images for the dataset itself were already captured previously, acquiring the dataset was merely a matter of waiting for the database access to be granted, as well as the dataset being organised into a desirable structure. As this process spanned over the Easter holidays, it lead to minor delays towards the completion of this step.

Implementation and Initial Testing

Timeline 08.04.2024 - 03.05.2024

Once the dataset had been acquired, it was time to implement the network. Implementation first began by making a simple neural network

model and gradually building on to it. Once a functional simple model had been created, initial testing with a few parameter combinations began to get a first look at the results it may deliver. Once enough tests were done, the given results were discussed and further adjustments to the project were made to ensure consistency.

In order to have a safer way to have access to the results, more autonomy of the project script was considered and thus it was implemented. A few test runs were performed and given results were transcribed automatically.

Final Implementation and Experimentation

Timeline 03.05.2024-03.06.2024

After considerable execution time and analysis of experiment results, some final adjustments of the implementation were made to increase evaluation options and to better allow for result analysis. Once the final implementation changes were made, the experiments were performed again and stored inside the final version of the result spreadsheet.

Project Documentation

Timeline 10.04.2024-10.06.2024

This is the longest time period of the project, encompassing the writing of the report and annexes of this project. This is due to the documentation being a continuous process that is dependant on the progress of the project. It would have not been possible to write about project results without first having completely finished the experimentation of the project first. Therefore there were multiple times where the writing of the documentation had to be paused.

A.4 Viability Study

In this section we will be taking a look at the viability of the project, both in regards to economical cost and effort, as well as in terms of legality of the work.

Economical Viability

This section will cover the economic cost of realising the project with brief predictions of the implementation and application cost of it.

Model Implementation Cost

The cost of collecting the sky image dataset can be quite high. Through the usage of a sky scanner (MS-321LR sky scanner¹) and a sky camera (SONA 201-D²), images can be captured and labeled for the dataset. A cheaper alternative, would be the purchase of a commercial camera that can capture images at a high dynamic range(HDR). The average cost of such a digital camera lies at around 600€ [11]. Though it may be a cheaper alternative in terms of hardware cost, it does have its disadvantages. The camera would have to be set up and adjusted to the outdoor conditions in order to be able to capture the desired image format and to make capturing automatic.

As previously mentioned in the conclusion of the report, there still remains the option to further expand on the work and further research, experiment and fine tune each network variation.

In order to do so, it would be necessary to hire research personnel for multiple months to continue this research. When taking into account the minimum inter-professional salary of 2024, this would lead to a monthly expense of 1134€ per researcher [2], excluding additional costs such as social security and taxes.

Furthermore, there would be a need to purchase a MATLAB license for each of the researchers, forcing a price of either 525€ for an indefinite license or an annual cost of 262€ for academic institutes, per researcher [9].

Each researcher would also need a functional laptop or computer to work on. The price range for these devices can greatly vary, depending on the needed functionality. Devices with a high processing GPU can be of great use when performing artificial neural network training. Those processing units do however require even greater funding [7].

As the dataset used is not of an open source and belongs to the research group that captured said images at the University of Burgos, it would either be necessary to pay for access to the dataset, or one would have to acquire a dataset of sky images manually, either by outsourcing to specialists or by doing so themselves. Both options would demand an intensive time and resource investment, as payment would be required to both the people that setup the digital camera to capture the sky images, as well as having to manually classify each image after acquisition or through the use of a sky scanner, which would have to be purchased. Additionally, enough time

¹To get the precise pricing, a quote must be requested here: [3], though the prices lie at a minimum of 30000€

²Precise pricing requires a quote request: [1]

would have to pass for all 15 sky conditions to be present at the capturing location.

Legal Viability

In this final section, we will assess the legislation and licenses required to perform this project.

Firstly, the acquired dataset was provided by a research group from the University of Burgos. It is important to note that the dataset is only available publicly in .mat format here: [12]. If the datasets' usage is required, extensive communications with the university and research group would have to be made in order to get legal usage permission. As the photographs themselves are simply images of the sky, there would be no legal action required to capture sky images oneself for a dataset.

In terms of licensing, most of the software used is open source and free to use. The software that this applies to are GitHub and Overleaf. The only license that needs to be paid for is MATLAB. The acquisition of a MATLAB license is a simple process and can be done quickly, simply through payment. Though excel is a Microsoft licensed software, its usage through its online version is free, this version however does not suffice for our intents and purposes, as we require a local version of excel to work with. There is however other software that allow the usage of csv files for free [10] such as, LibreOffice Calc, OpenOffice Calc, gedit and more.

As a university student, my access to all of the aforementioned software has been free, granted by the licenses provided to students by the University of Burgos and my home university the University of Applied Sciences Hamburg.

Appendix B

Requirements Specification

B.1 Introduction

This section will contain the requirements for the project as well as its use cases.

B.2 General Objectives

The main objective of this work has been to find alternative training functions for Artificial Neural Networks that adapt for sky classification under the CIE standard. In addition to that, the application of image processing, tested by previous work [5] is applied to the current objective to further reassure its results.

B.3 Catalogue of Requirements

Functional Requirements

- **FR-1** The network must use the specified dataset for its training and testing.
- **FR-2** Training and testing must be done equally for each training function: SCG, OSS, RP and GDX.
- **FR-3** The dataset split for the training process must be replicable.

- **FR-4** Network training must utilise cross-validation to prevent over-fitted results.
- **FR-5** All samples from the dataset must be labeled using their specific classes to which they have been designated to inside their respective directory.
- **FR-6** All images must be resized before training to enable a smoother and efficient training process.
- **FR-7** All training must also be performed using the panchromatic channel of the images.
- **FR-8** All images inside the loaded dataset must be transformed into a 1 dimensional array of pixel values for the network to be used.
- **FR-9** The label layout of the images must be transformed into a matrix that specifies each sample's class using a binary value.
- **FR-10** All training results must be stored inside a csv file, the stored results must include: the date and time of training completion, the accuracy, the precision, the recall, the number of hidden layers used, the number of hidden neurons per hidden layer, the training function used and the training time.
- **FR-11** The resulting confusion matrix after testing must also be saved separately.
- **FR-12** The best resulting network, measured here by its accuracy, must be saved for later use.
- **FR-13** Once the best network has been defined and saved, it must be possible to load the network to predict classes using a specified prediction dataset
- **FR-14** The prediction must return a prediction values for each given sample.
- **FR-15** The prediction values must be stored in a csv file for later use.

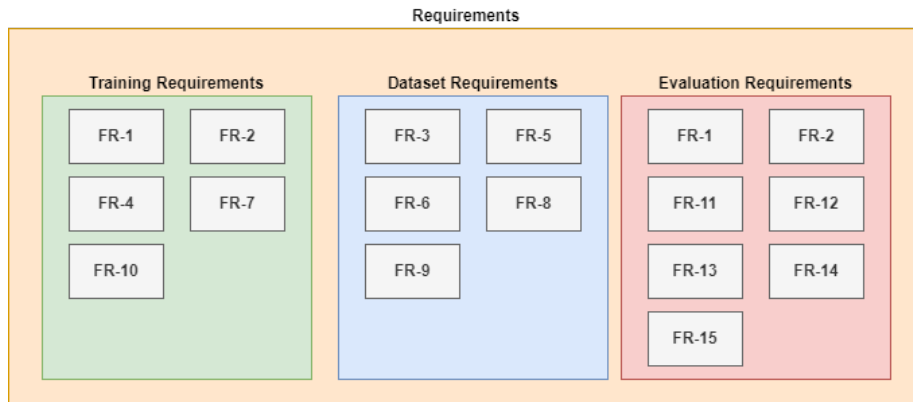


Figure B.1: Overview of how the functional requirements correlate

Non-Functional Requirements

- **NFR-1** The layout of the csv file for training results must be intuitive and readable.
- **NFR-2** Script layout must be clean and well commented.

B.4 Requirements Specification

Use Case Diagram

The use case diagram for the neural network can be seen in figure [B.2](#)

Use Cases

The use case tables can be found [B.1B.2](#)

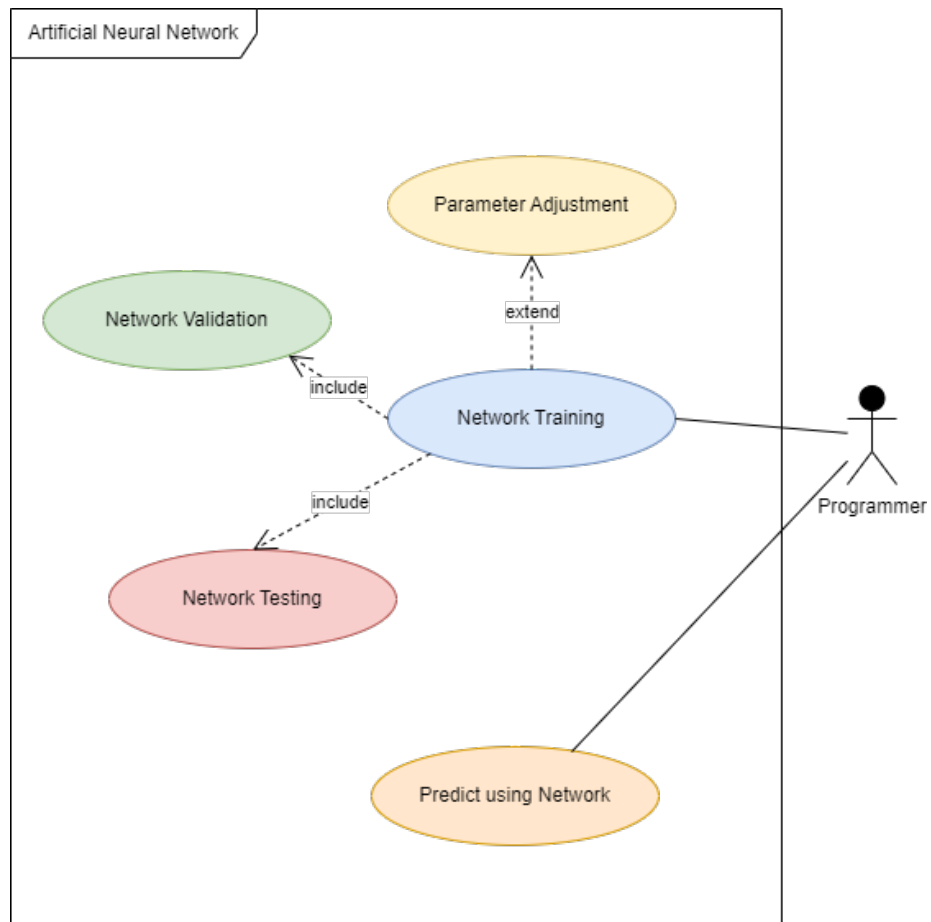


Figure B.2: Use case diagram of the model

Use Case-1	Train the network
Version	1.0
Author	Bardia Filizadeh
Associated-requirements	FR-1, FR-2, FR-3,FR-4,FR-5,FR-6,FR-7,FR-8,FR-9,FR-10,FR-11,FR-12
Description	The programmer trains the network with the given dataset.
Precondition	The dataset and MATLAB script
Actions	<ol style="list-style-type: none"> 1. The programmer specifies the filepath for the dataset inside of the script. 2. If needed: desired training parameters are specified inside the script by the programmer. 3. Programmer executes the script and networks are trained. 4. Once training is finished, the programmer can find the results inside the saved csv file.
Postcondition	The network is trained, results are saved in a csv file
Exceptions	If the dataset or the training parameters are faulty, then the training will result in failure and no network will be extracted.
Importance	High

Table B.1: Programmer use case 1.

Use Case-2	Prediction using the network
Version	1.0
Author	Bardia Filizadeh
Associated-requirements	FR-13, FR-14, FR-15
Description	The programmer predicts the classes of a dataset.
Precondition	The dataset in required format and MATLAB script for prediction, Trained network
Actions	<ol style="list-style-type: none"> 1. The programmer specifies the filepath for the prediction dataset inside of the script. 2. The dataset is loaded into the workspace. 3. The previously trained network is loaded into the workspace. 4. The network predicts the classes of the dataset and saves resulting data inside a csv file.
Postcondition	The network predicted the dataset classes, results are saved in a csv file
Exceptions	If the dataset or network is faulty, then the prediction will result in failure and no predictions will be saved.
Importance	High

Table B.2: Programmer use case 2.

Appendix C

Design

C.1 Introduction

This appendix will briefly go through multiple design choices, both design choices made and ones that were already prerequisite.

C.2 Data Design

As previously mentioned in the report, the dataset used was provided by a research group of the Higher Polytechnic School of Burgos, granting access to 1500 sky images of the Burgos sky. The images were captured using a commercial SONA 201-D sky camera. The camera was stationed at the top of the Higher Polytechnic School building and captured images every 15 seconds [5]. The images captured, have a high resolution of 1158x1172 pixels recorded with the RGB color model. A fish-eye lens was used to achieve the spherical look that can be seen in the report.

The sky types captured by the images were determined, using the Normalized Luminance method [5]. It is important to mention that sky classification under the CIE standard using photographs, requires High Dynamic Range (HDR) images, in order to capture the full sky luminance range [5].

Model	MS-321LR sky scanner
FOV	11°
Luminance	0 to 50 kcd/m ²
Radiance	0 to 300 W/m ²
A/D convertor	16 bits
Calibration error	2%

Table C.1: Technical specifications of the sky scanner [5]

Model	SONA 201-D
Sensor	CMOS-2.3 MP
Vision angle	<180°(fish-eye lens)
Operating temperature	-40°C to 55°C
Image format	RAW

Table C.2: Technical specifications of the sky camera [5]

C.3 Procedural Design

Initially the focus of the proposed work was to create a neural network that focuses on cloud classification, differentiating between different types of clouds. However during the planning phase for the design, it was decided that the network would be focused on sky classification due to time constraints related to the data acquisition of the initial proposal. When designing an ANN it is most important to start off with understanding the data that is provided or needs to be acquired, as well as a clear definition of the problem, that being: What problem is the ANN supposed to solve? Should it be a classification, regression or clustering problem? Thus the decision for a sky classification problem was made.

As is standard for all machine learning problems, the acquisition of data was prioritised first, then the shift towards implementation, experimentation and evaluation could begin

C.4 Architectural design

In terms of architectural design, the choices made for ANNs tend to be very similar. Firstly, after collecting the necessary data, it is important to investigate whether data cleaning¹ is required. As the dataset provided in this project had been pretty much perfect, there was no need for any data cleaning.

Once that step has been finished, the decision whether data transformation is needed is made. Data transformation can help neural networks understand the data more easily or make its data structure more standardized. In the case of this project, the images provided needed to be resized to a smaller size, in order to not overload the computation device, as well as reduce the number of values the neural network has to "memorize", which also speeds up the training time significantly and reduces the computational power requirement for the network.

Additionally for the sake of this experiment, the network was trained with both images of the RGB color channel and the same images using the Y color channel respectively.

The final decision that needs to be made for the data is to decide how to split it for the ANN's training. Data splitting usually splits the dataset into 3 splits, training set, validation set and test set. Common splits are 70/15/15% or 80/10/10% for training/validation/test respectively. It is very important that all splits contain multiple samples of each class that is going to be classified. The variety required by the training set is most important as its one of the most influential factors in how well the neural network will be able to generalize, that meaning how well it can classify on unknown data without reliance on already known data.

The next design decisions are made in regard to the network's architecture. As mentioned in the theoretical concepts chapter of the report, ANNs consist of 3 different layers, the input layer, the hidden layer(s) and the output layer. The design of the hidden layer is especially important as the layout required greatly varies from problem to problem. The most common decisions that need to be made are the number of hidden layers and the number of hidden neurons in each of them. For this project, after a few small experiments, the conclusion was made that this type of problem does not necessitate more than 1 hidden layer. Thus, training was performed using only 1 layer. The number of hidden neurons required can be very different based on the

¹Data cleaning refers to the handling of missing values inside the data, outliers that may be harmful for training, or noisy data that hinders improvement.

next parameter that had to be decided upon. So, for the sake of research, training was performed using a different number of hidden neurons, using every combination possible within a set interval of hidden neuron values.

The parameter that was being referred to is the training function. As one of the main focuses of this project is the comparison of viability of different training functions for sky classification, 4 training functions were chosen: One-step Secant(OSS), Gradient Descent with Momentum and Adaptive Learning Rate Backpropagation (GDX), Resilient Backpropagation (RP) and Scaled Conjugate Gradient (SCG). SCG was chosen as a comparative measure to the other training functions, as its viability had already been proven by previous work [5]. The other 3 training functions were chosen based on their usage of backpropagation for training, which can be a very useful measure for pattern recognition tasks such as this.

As mentioned in the theoretical concepts chapter of the report, the ANN has many more parameters that can be specified, however their significance was not a focus for this work, therefore mostly the default initialisation values for them were used for this project, as they delivered viable results.

Once the parameters have been set, training can begin. After the training is finished, further design choices can be made, such as which evaluation metrics to use to review the network's performance. The metric choices made for this work were, accuracy, precision, recall and training time. The f1-score was then calculated after choosing the best network configurations for each training function.

As the classes of the dataset were perfectly balanced, with a sample number of 100 images per class, the usefulness of precision, recall and f1-score may be seen as unnecessary. However, they are still a safety measure, to keep an overview as to whether a network truly does have the calculated accuracy.

The decision to save all evaluation metrics inside a csv file was also made to enable a quick and easy way to sort the acquired data, as well as have the possibility to create plots that show the differences between different parameter choices.

Appendix D

Programmer Manual

This appendix explains how the directories are organized and how to execute the network.

D.1 Directory Structure

The directories contain the datasets, scripts and result data required to both evaluate the network as well as use it for future purposes. The directory contains the following folders:

- **Training Dataset:** The most important and easy to understand folder. It contains all the sky images that will be used by the network for training and testing. The folder contains 15 subfolders representing the 15 sky types and each subfolder contains a folder for each individual images. It is crucial that the dataset folder only contains the 15 class folders, when using the database provided by the UBU research group, it is important to extract the extra folder "cielosmezcla" outside of the main dataset folder, as that one will be used for the prediction dataset.
- **ANN Training Script:** This script contains all the functions that are used for the network training and testing, as well as the evaluation metric calculations that are stored inside a csv.
- **ANN Prediction Script:** This script contains all the functions and procedures that are used for the network to predict on a new dataset. It also creates a csv file where the results of the prediction are saved.

- **Training Result Spreadsheets:** These csv files contain all the results from ANN training, including the date and time of training completion, accuracy, precision, recall, number of hidden layers, number of hidden neurons, training function used and training time. There are separate spreadsheets for training with each color channel.
- **Resized Images Folders:** These folders contain the newly resized images that were required for training and prediction and were transformed during preprocessing. There are separate folders for the RGB images and the panchromatic images.
- **Confusion Matrix Folders:** These folders contain the confusion matrices of all test results for all network variations. Their names indicate the date and time, training function used, as well as hidden layer size and number. There are separate folders for each color channel.
- **Best Network:** This .mat file contains the best network acquired from the ANN training. The network has to be used to predicted images of the same color channel with which it was trained on. It can be loaded using the MATLAB load method.
- **Prediction Dataset:** This folder contains all samples that are to be used for prediction after a network has been chosen from experimentation.
- **Prediction Results:** This csv spreadsheet contains the predictions performed by the ANN on the prediction dataset.

It is important to keep in mind that the directories should be kept available to the MATLAB workspace, as many of them will be edited during script execution. It is also essential that the result spreadsheets are not accessed during training, as it would deny MATLAB editorial access to the file, leading to execution failure.

D.2 Programmer Manual

Network Training

When trying to alter or adjust the network training parameters/dataset, it is most important to correctly load the desired dataset. The programmer has to keep in mind that the structure of the dataset directory is correct and

consistent. If it is not up to the set standard, the programmer must either change the directory structure, or adjust the loading and label splitting method inside of the scripts.

Beginning with the network model script, the dataset directory should have the following structure¹:

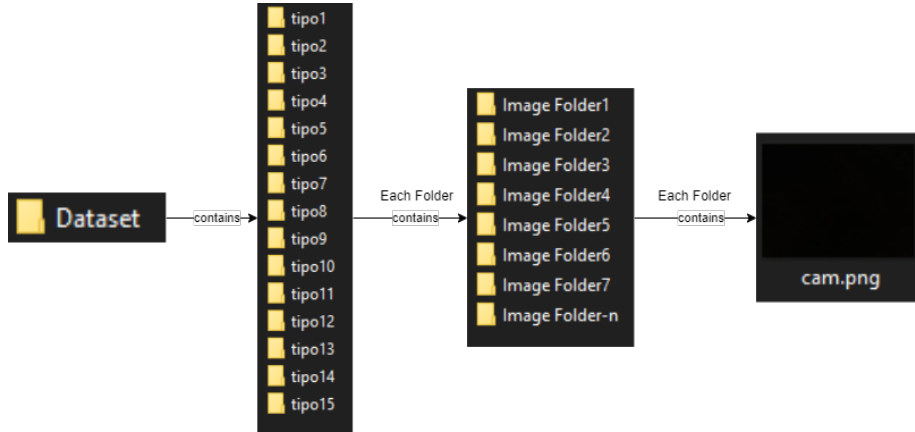


Figure D.1: Directory Structure Standard of the Dataset

If a change in folder name/structure is required, the programmer has to alter these 2 functions:

```

%%Load Dataset
dataset = imageDatastore("Dataset\**\cam.png", LabelSource="foldernames");
  
```

Figure D.2: ImageDatastore method provided by MATLAB that loads the dataset

Firstly, if a name change of the image files is performed, then the parameters for the imageDatastore method[8] in figure D.2 must be adjusted, as it currently loads all "cam.png" files inside the specified folder.

¹The names of the image folders have been altered for privacy reason, the ones in the figure are only example names.

```
% Sets the labels of the specified files by splitting their file path
function labels = setLabels(files)

    labels = split(extractAfter( files, "Dataset\\"), filesep);
    labels = split(labels(:,1));
    labels = split(labels, "tipo");
    labels = split(labels(:,2));
    labels = str2double(labels);

end
```

Figure D.3: Function that sets the labels of the samples by splitting their file path until the numerical value of the class folder has been acquired

Should either the directory structure or folder names be changed, then the `setLabels` function has to be modified by the programmer. As it currently stands, the function takes the file path of all images and splits it using the parent folder "Dataset" and further splitting them until the class folder (e.g. "tipo1") has been reached. Then the class folder name is split off at the "tipo" part and the sky type number is acquired. Changing the name of this folder or the structural layout of the directories would require different splitting.

Should the need arise to change the script so the network is only trained using images of 1 color channel, then the following needs to be changed:

```
12 %Variable to decide whether images should be processed into panchromatic
13 %channel after resizing or not
14 panchromatic = true;
```

Figure D.4: Parameter that will decide whether panchromatic images will be used (value == 1) or RGB images (value == 0)

The variable seen in figure D.4 needs to be changed appropriately to the desired color channel, 1 for panchromatic, 0 for RGB.

```

53    %%
54    %Update the Files property of the datastore with the resized filenames
55    resizedDataset = resizeDataset(files, 0, numFiles);
56    resizedDataset.Labels = labels;
57    inputSize = size(imread(resizedDataset.Files{1}));
58
59    %Transform image into 1D array of pixel values
60    trainImages = transformImagesTo1D(inputSize, resizedDataset);
61    %Change labels to binary values indicating the images class with a 1 in the
62    %corresponding row indicating the sky type
63    numClasses = 15;
64    trainLabels = transformLabels(resizedDataset.Labels, numClasses);
65
66    trainSkyNetwork(neurons, funcs, funcNames, trainImages, trainLabels, 0);

```

Figure D.5: Second set of training that would usually be performed using the RGB channel

Should the previously mentioned change be applied, then there is no need to perform training using both color channel variations. Therefore, the 2nd set of training that is performed after the first set has been completed can either be deleted or commented out [D.5](#).

If the need arises to either shorten or extend the list of different parameter combinations or the need to further fine tune other parameters of the network, then the following can be done:

```

%%
%Set array of neuron quantity to train
neurons = [1 3 5 10 15 20 30 40 50 60 70 80 90 100];
%Set the training functions that will be used for training
funcs = ["trainscg","trainrp" "trainoss" "traingdx"];
funcNames = ["SCG", "RP", "OSS", "GDX"];

```

Figure D.6: Hidden Neurons Quantity and Training Functions used arrays

Adjusting these arrays [D.6](#) specifies the parameter combinations that will be used during the training cycles. The network will be trained with every training function and for each training function, with every hidden neuron quantity. Shortening these arrays would reduce the number of training instances performed, while extending them will do the opposite. The total

number of training instances is $n*m$, with n being the number of values in the "neurons" array and m being the number of training functions in the "funcs" array.

In order to further fine tune other parameters of the network, the "setupNetwork" function can be further expanded, either through hard-coded value assignments or through additional parameters for the function [D.7](#).

```
% Sets up the network parameters according to the specified values, dataset
% split indices are applied
function net = setupNetwork(layers, maxVal, epochs, trainfcn, train_idx, val_idx, test_idx)

    net = patternnet(layers,trainfcn);
    net.divideFcn = 'divideind';
    net.trainParam.max_fail = maxVal;
    net.trainParam.epochs = epochs;
    net.divideParam.trainInd = train_idx;
    net.divideParam.valInd = val_idx;
    net.divideParam.testInd = test_idx;

end
```

Figure D.7: Setup Network Function of the MATLAB script

If there is a need to use further evaluation metrics or the desire to save more results from training and/or testing, then a few changes must be done.

```
%Define the data to be written, including the current date and time
currentDateTime = datetime('now');
results = {string(currentDateTime), accuracy,precision,recall, hiddenLayers, hiddenNeurons, funcName, time};
```

Figure D.8: Array structure that contains all the results that are going to be saved

Firstly, the result array [D.8](#) needs to be expanded to save the additionally wanted result values.

Once that is done, the "saveResults" method [D.9](#) needs to be adjusted so that the header of the result file contains the new variable name(s)², as well as the final writing access into the file ³, containing the added variable(s) value(s). It is important to note that when adding a variable for both header and variable value, the corresponding "fprintf" function call needs to include

²Via the "headers" variable

³Via the final "fprintf" of the method

an additional format specifier⁴ according to the format that the variable will have. Headers are always "s%" so that is always a simple addition..

```
%Training and test results are saved into a csv file, different file
%chosen depending on the color channel
function saveResults(results, panchromatic)

    %Specify the file name
    if panchromatic
        filename = 'resultsY.csv';
    else
        filename = 'results.csv';
    end

    %Check if the file exists
    if exist(filename, 'file') == 0
        %If the file doesn't exist, create a new one and write the headers
        fid = fopen(filename, 'w');
        headers = {"DateTime", "Accuracy", "Precision", "Recall", "Hidden Layers", "Hidden Neurons", "Training Function", "Training Time"};
        fprintf(fid, "%s, %s, %s, %s, %s, %s, %s\n", headers{:});
    else
        %If the file exists, open it to append data
        fid = fopen(filename, 'a');
    end

    %Write the data to the file
    fprintf(fid, '%s, %4f, %4f, %4f, %d, %d, %s, %4f\n', results{:});

    %Close the file
    fclose(fid);
    %Close all open file handles
    fclose('all');

    clear headers results fid filename filename
end
```

Figure D.9: SaveResults function of the MATLAB script, it opens a csv file, or creates a new one if it doesn't exist and writes the newly acquired results into it.

Prediction using the Network

When predicting new data using the trained network, there are a few things that need to be taken into account. Obviously, the network needs to be trained already and it needs to be trained using images of the same color channel for which the prediction dataset is going to be used. In order to specify which color channel shall be used for prediction, the following variable needs to be adjusted [D.10](#):

⁴The letter (s,d,f etc.) followed by "%" inside the fprintf method call string

```
%%
%Choose whether to use panchromatic channel or RGB channel
panchromatic = 1;
```

Figure D.10: Variable that decides whether RGB or panchromatic images will be used for prediction

A variable value of 1 signifies panchromatic images being used, a 0 means RGB images shall be used.

To Load the prediction dataset, a similar structure as the training dataset needs to be followed.

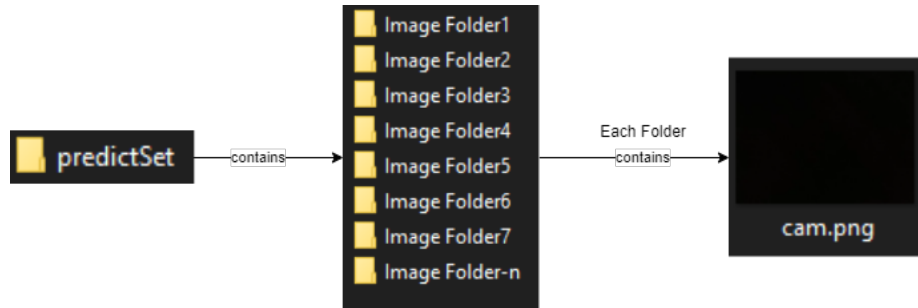


Figure D.11: Prediction dataset directory structure

As seen in figure D.11, the dataset follows a similar structure as the training dataset. The difference however, lies in the fact that the sky image class types should preferably be unknown, so that the network can show its functionality. The directory structure does not need to follow the exact structure as seen in the figure. The only important aspects are that the root folder of the dataset has the name "predictSet" and that all the images are in separate folders, to allow the same name usage "cam.png".

Should a change in name or structure be required, then the following function call[8] needs to be adjusted D.12:


```
%%Load Dataset  
dataset = imageDatastore("predictSet\**\cam.png");
```

Figure D.12: ImageDatastore function call to load the prediciton dataset

D.3 Project compilation, installation and execution

For a detailed explanation of how to install MATLAB and its necessary extensions, please refer to appendix E.

Once MATLAB has been succesfully installed, the process of executing the scripts is quite simple. Firstly, it is most convenient to open a MATLAB instance and open the desired script (either `tfgModel.m` to train the network or `predict.m` if a network has already been trained and predictions are desired). The TFG folder should preferably be saved inside of the MATLAB workspace folder. Once the script has been opened, check the directory overview on the left side of the MATLAB instance [D.13](#). The TFG folder should be visible there. In order for the script to work more conveniently, double-click the TFG folder to make it the current directory of the work space.

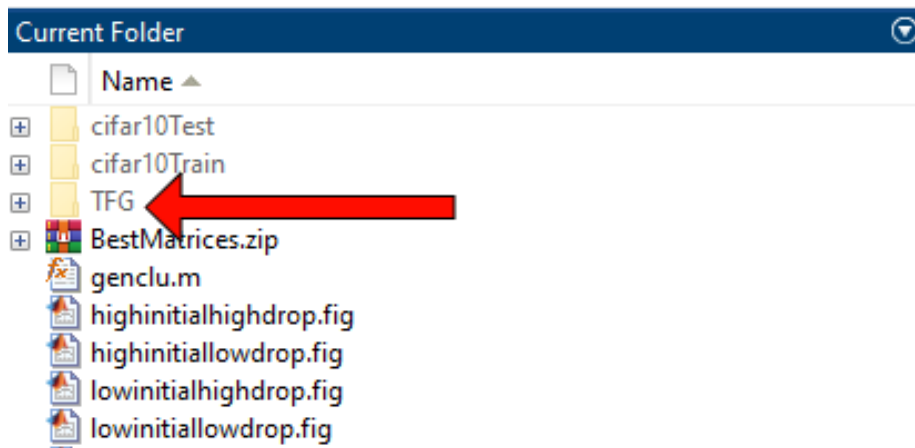


Figure D.13: Directory overview of MATLAB

Once that is done, the entire script can either be run step by step by using the "Run Section" or "Run and Advance" buttons, or by simply pressing

the "Run to End" or "Run" button to run the complete script. The buttons can be found at the top under the "EDITOR" tab.

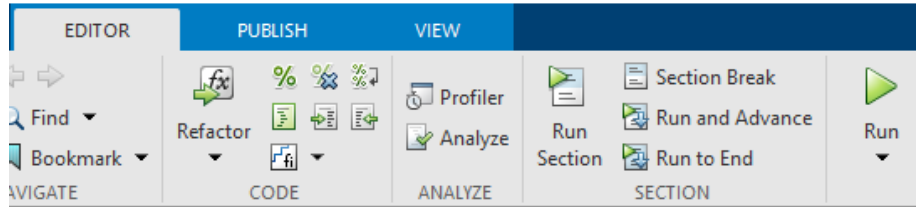


Figure D.14: Run buttons in MATLAB

Running the training script (tfgModel.m)

Running this script to completion can take hours once the "trainSkyNetwork" function has been called, as the script then starts looping over every parameter and training function combination possible to train networks. So it is best to either use the "Run" button or the "Run to End" button once the selected section contains the aforementioned method.

Once the script has been fully executed, you will find several new folders within the current work directory. These folders will contain the resized images of both color channel variations, as well as the confusion matrices of the test results. The results of each color channel training can be found inside their respective csv files "results.csv" and "resultsY.csv". The best networks found during training for each color channel will also be saved inside a MATLAB data file, "Skynetwork.mat" and "SkynetworkY.mat" respectively.

Running the prediction script

Once this script has been run to completion, the resized dataset of the prediction dataset can be found in their respective folders. Additionally, the result spreadsheet of the model's prediction can be found as well in "predictions.csv".

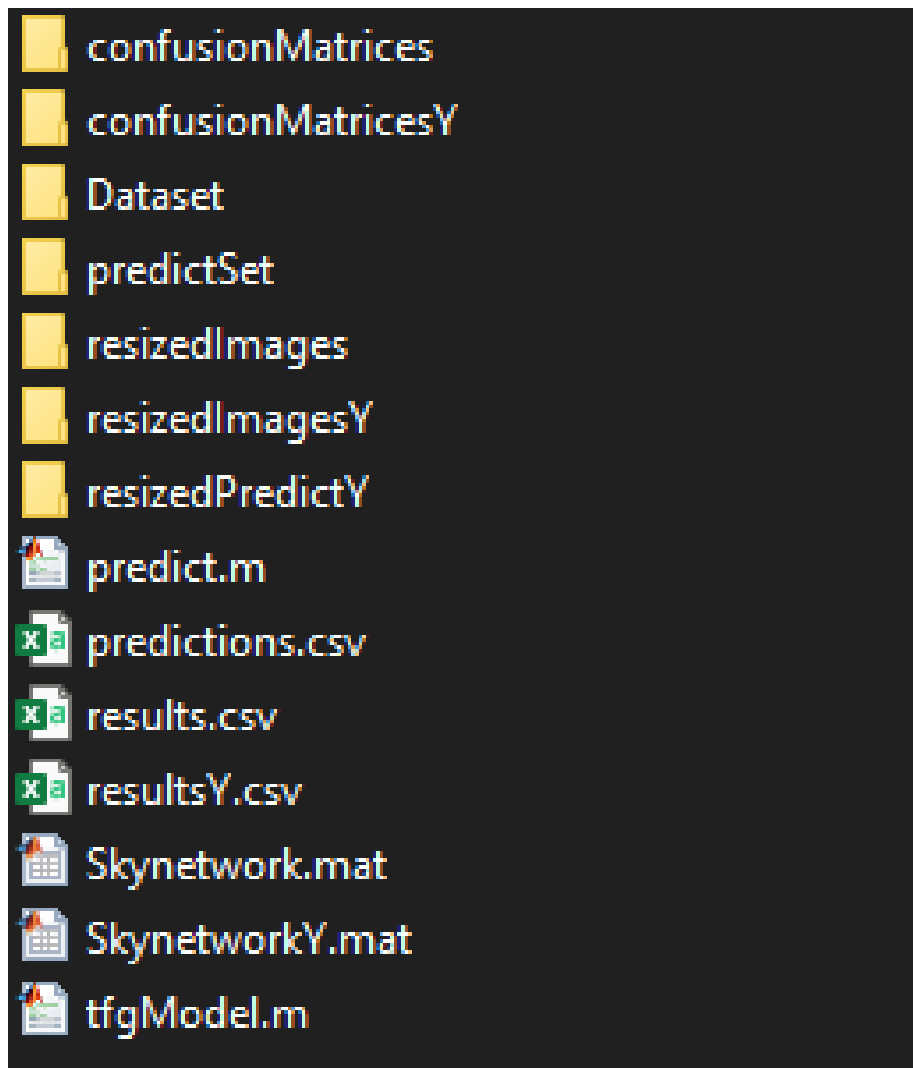


Figure D.15: Example of how the directory content should look like after running both scripts

Appendix E

User Documentation


This appendix explains the software requirements needed to execute the neural network training and prediction scripts for the network trained in this project.

E.1 Software Requirements

To be able to execute the scripts, the user will require the license for the software used in this project, which as mentioned in the "Techniques and Tools" chapter of the report, is MATLAB.

Acquiring a MATLAB license is quite simple, if you're a student/academic, your institution will most likely provide you with an academic license for MATLAB. If your institution does not provide you a license or if you aren't a student, then a license will have to be purchased, either by annual payment, or through an indefinite license [9]. Fortunately, licenses purchased for private purposes by individuals are substantially cheaper than the ones provided for institutions and companies. Private licenses are all indefinite and only require a one time payment.

Once a license has been acquired, the next step is installing the software. As this project was developed on MATLAB version R2024a, downloading that same version is essential. To do so, go to <https://www.mathworks.com/downloads/> and login using your newly acquired account that is linked to your license and choose the R2024a release of MATLAB to download for your operating system. Once downloaded, you simply have to run the executable and follow the installation steps. Below the carried out process is shown on the 64-bit Windows version.



Email

[No account? Create one!](#)

By signing in, you agree to our [privacy policy](#).

[Next](#)

Figure E.1: Step 1 of MATLAB installation

When opening the executable, you will be prompted by the following window [E.1](#). You will have to login using the account you used to download the executable. Once logged in you will find the terms of license agreement. After reading through the terms you can agree to them and continue. The next window should prompt you to select your license that is linked to your account. Select the license and continue. You will now have to select the destination folder of your MATLAB installation, either change the location to your liking or keep the default location. Once selected, the window will offer all the toolboxes that you want to add during the installation. The toolboxes needed for this project can be seen in figure [E.2](#)






Name	
	Deep Learning HDL Toolbox version 24.1
	Deep Learning Toolbox version 24.1
	Image Processing Toolbox version 24.1
	Parallel Computing Toolbox version 24.1
	Statistics and Machine Learning Toolbox version 24.1

Figure E.2: MATLAB toolboxes used for this project

If you accidentally miss one of these toolboxes before installation, there is no need to worry. The toolboxes can also be installed after MATLAB has been completely installed, using the "Add-Ons" button at the top of the MATLAB window.

After selecting all the needed toolboxes, you can simply continue the installation instructions and the installation should finish successfully. Now that MATLAB has been successfully installed, the project installation can begin.

E.2 Installation

After acquiring the TFG folder through either GitHub or USB-stick, the folder can be extracted and should be copied to the MATLAB workspace folder [E.3](#), which can usually be found under the device's "Documents" section.



Figure E.3: Matlab workspace folder location, TFG folder should be inserted there

Should the dataset be acquired separately after receiving the scripts from GitHub, then the dataset folder should be moved inside the TFG folder, on the same level as the scripts. Should the dataset folder include a "cielosmezcla" folder, then it should be moved outside of the folder on the same level as the scripts, this will be the test folder used for predictions using the final trained network. Just make sure to rename the folder to "predictSet" as that is required. To reassure that the folder structure of the main dataset is correct, please confirm with this figure [D.1](#).

E.3 User Manual

Once everything is in place, the scripts need simply be opened in MATLAB, by clicking the "Open" button at the top left under the "HOME" tab and selecting the "tfgModel.m" script inside of the TFG folder. Once the script is open, make sure that the current folder of the workspace is the TFG folder, it can be seen right under the previously selected "Open" button, and is a white horizontal line that spans across the window. Its last 2 folders should be the same as in this example [E.4](#)

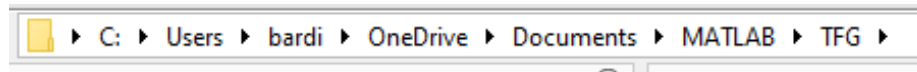


Figure E.4: Example current directory path of MATLAB

If the path does not end in "MATLAB -> TFG" then you should double click the TFG folder on the folder overview seen on the left side of the window. It can be seen in this figure [D.13](#) Now is the time to run the "tfgModel.m" script. Simply click the "Run" button seen in the "EDITOR" tab at top of the window [D.14](#) and the network's training will begin. After training is done (which may take an extended time period of a few hours), the network is ready to predict samples using the previously extracted "predictSet". Simply open the "predict.m" script and run it. There should now both be the training results and the prediction results inside of the TFG folder [D.15](#).

Appendix *F*

Curricular Sustainability Annex

F.1 Introduction

Sustainable development is an aspect that can determine both a projects longevity, as well as its importance on society. Its application can make or break a project and its importance has been recognized in this work. It can be best summed up as a projects ability to meet the needs of present requirements, without compromising its ability to be used, maintained or expanded in the future [6].

A few techniques that have been employed in this project that can be seen as sustainable are the following:

- **Open source development:** All the code used for the project can be found inside the public GitHub repository [4]. By publicly sharing the source code to everyone, we can help and encourage future projects by reducing their cost spent in buying either private sources or investing in private works. It also allows for free code that can be used as an example to study and learn from. Whether that be spotting mistakes, extending the code or adapting the code to different problems.
- **Code Comments:** By documenting simple and efficient code comments inside the project, future developers can more effectively understand the code's function and how to use it. It allows for easier learning and and maintenance, as well as paving the way for future extensions.
- **Split code structure:** Inside the scripts, the individual functions that are used are clearly separated and commented, their function calls

summarize what the function does, therefore increasing readability as well as making the process easier to understand. This will help future developers understand the code faster and how the functions are used.

- **Popular frameworks:** By using more commonly used frameworks, we make our application more manageable as the documentation and usage of these frameworks is more frequent, therefore bringing a higher number of users that have interacted and can interact with this project. Additionally, these frameworks allow for faster response times to questions asked on forums as there will be more active users. All of this makes the project more maintainable and allows for easier extension.
- **Framework Accessibility and ease of use:** It also helps sustainability, that MATLAB is a very user friendly in terms of its accessibility, especially when it comes to the area of this project, which is the usage of artificial neural networks. Especially for ANNs, MATLAB offers many examples and code explanations as well as video tutorials that all help make the code easier to understand. MATLABs accessibility also makes it a great option to perform experiments on, as going through the code step by step can severely help with understanding code sequences. By not having to go out of your way to go into debugging mode, the visibility and readability of the steps are a lot easier.

Unfortunately there is a part of this project that hurts the sustainability, and that is the requirement of a MATLAB license. As it is needed to run and edit the scripts properly, the purchase of the license can't be worked around without completely changing the working environment. Fortunately, as mention in appendix E in the software requirements, the acquisition of the MATLAB license isn't as difficult as it could be. With the addition of student discounts and lower prices for home users¹, buying a MATLAB license (which is a one time cost, for a permanent software) can be worth price.

The work itself, can also help contribute to society in a variety of ways that help overall sustainability and the environment. By being able to classify sky topology under the CIE standard, the usage of natural light and solar energy can be optimized for every location, helping reduce energy usage globally.

¹The license cost comes to around 69€ for students and 119€ for home usage

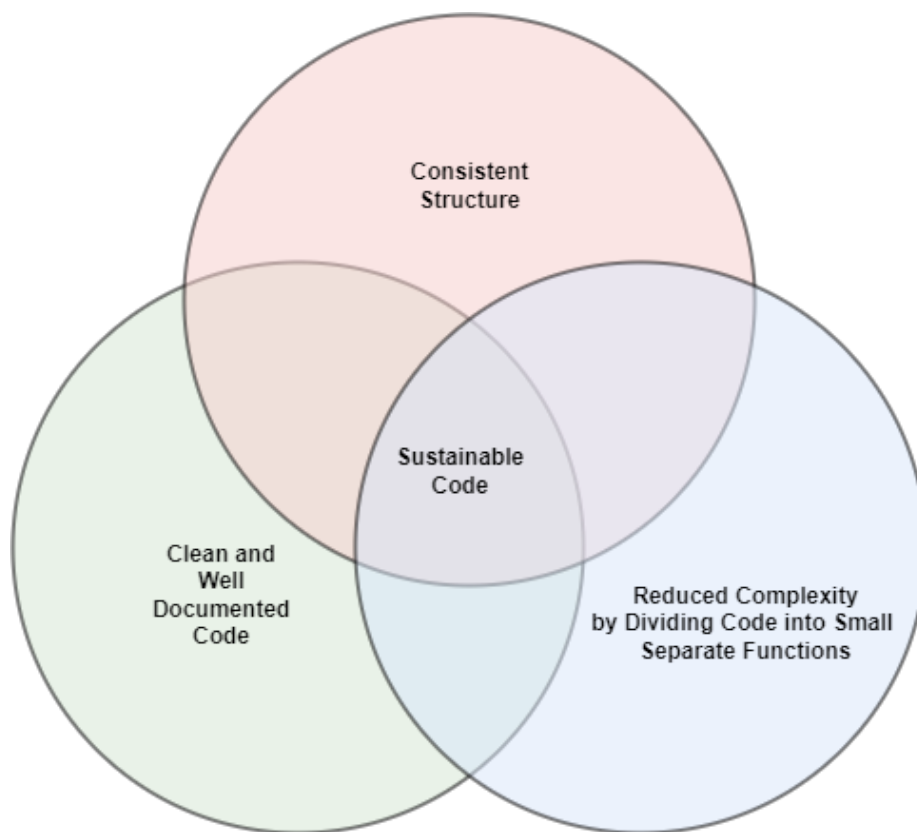


Figure F.1: Example graphic of components that can make up sustainable code

Bibliography

- [1] Sieltec Canarias. Sona, 2024. [Internet; downloaded 11th of June 2024].
- [2] Agencia Estatal Boletín Oficial del Estado. Documento boe-a-2024-2251, 2024. [Internet; downloaded 9th of June 2024].
- [3] EKO. Ms-321lr sky scanner, 2024. [Internet; downloaded 11th of June 2024].
- [4] Bardia Filizadeh. Tfg sky classification, 2024. [Internet; downloaded 10th of June 2024].
- [5] D. Granados-López, A. García-Rodríguez, S. García-Rodríguez, A. Suárez-García, M. Díez-Mediavilla, and C. Alonso-Tristán. Pixel-based image processing for cie standard sky classification through ann. *Complexity*, 2021, 2021.
- [6] IISD. Sustainable development, 2022. [Internet; downloaded 10th of June 2024].
- [7] Monica J. white Jacob Roach. Gpu prices and availability (q1 2024): How much are gpus today?, 2024. [Internet; downloaded 9th of June 2024].
- [8] MathWorks. imagedatastore, 2024. [Internet; downloaded 10th of June 2024].
- [9] Matlab. Pricing and licensing, 2024. [Internet; downloaded 9th of June 2024].
- [10] Relateable. Top 10 free csv readers in 2024!, 2024. [Internet; downloaded 9th of June 2024].

- [11] Jaron Schneider. The average digital camera price has doubled over the past three years, 2023. [Internet; downloaded 11th of June 2024].
- [12] Cristina Alonso Tristán, M^a Isabel Dieste Velasco, Montserrat Diez Mediavilla, David González Peña, María del Carmen Rodríguez Amigo, and Diego Granados López. Neural network database, 2021. [Internet; downloaded 11th of June 2024].