Title: Identifying aircraft from above

Name: Kai Roper-Blackman

Registration Number: 1602999

Supervisor(s): Adrian Clarke, Sebastian Halder

Second assessor: Luca Citi

Degree Course: BSc Computer Science (G400)

Contents

[Acknowledgements 3](#_Toc6315802)

[Abstract 4](#_Toc6315803)

[List of Symbols 5](#_Toc6315804)

[Project Aims and Objectives 6](#_Toc6315805)

[Primary Objectives 6](#_Toc6315806)

[Additional goals 6](#_Toc6315807)

[Methods 7](#_Toc6315808)

[Internal and External Libraries 7](#_Toc6315809)

[Data set 9](#_Toc6315810)

[Images 9](#_Toc6315811)

[Parsing 10](#_Toc6315812)

[Pre-Processing 10](#_Toc6315813)

[Histogram of oriented gradients 10](#_Toc6315814)

[Feature vector 11](#_Toc6315815)

[Machine learning 13](#_Toc6315816)

[Supervised learning 13](#_Toc6315817)

[Support vector machines 13](#_Toc6315818)

[Cross validation 15](#_Toc6315819)

[Technical Achievement 17](#_Toc6315820)

[Graphical user interface 17](#_Toc6315821)

[Standalone classification 18](#_Toc6315822)

[Large image search 19](#_Toc6315823)

[Saving of results 20](#_Toc6315824)

[Project Planning 21](#_Toc6315825)

[Momentum 21](#_Toc6315826)

[Adapting to change and dealing with risks 21](#_Toc6315827)

[Achievement 21](#_Toc6315828)

[Performance 21](#_Toc6315829)

[What have I learnt? 22](#_Toc6315830)

[Conclusion 23](#_Toc6315831)

[References 24](#_Toc6315832)

[Appendices 24](#_Toc6315833)

[Sustainability 24](#_Toc6315834)

[Legal 24](#_Toc6315835)

[Ethical 24](#_Toc6315836)

[Intellectual property 24](#_Toc6315837)

# Acknowledgements

A special thanks to my parents Annie Roper and Seon Blackman for supporting me throughout University.

Thank you to all the academic staff in CSEE, specifically Adrian Clarke, Sebastian Halder and Luca Citi for feedback and guidance throughout this project.

# Abstract

We all know what an aircraft looks like, but does a computer? A seemingly simple task that can be carried out by individuals at age two, poses a complex problem to modern technology. Machine learning is a relatively new field with little research but already boasts claim to many applications such as driverless cars and face recognition systems. The development of object recognition is the center of many companies’ business models and objectives, making aircraft identification such an interesting topic to research.

Existing images of ground and aircraft are pre-processed using Histogram of Gradients to create feature descriptors. Feature descriptors describe the orientation of a gradient within an image subsection. Support Vector Machines are passed feature descriptors with labels for training. Once training is completed, the support vector machine accepts a test set and returns predictions. Large image search takes a large image and looks within a smaller area for aircraft. Search area parameters are provided by the user.

The results obtained from cross validation show an accuracy of 100% when identifying standalone aircraft. However, when searching for aircraft in larger images, accuracy drastically decreases to around 55% as some aircraft are overlooked. After optimization, the system used to identify aircraft can be applied to other identification problems with possible military and commercial uses.

# List of Symbols

|  |  |
| --- | --- |
| **Symbol** | **Meaning** |
| SVM | Support Vector Machine |
| PNG | Portable Network Graphics |
| px | Pixels |
| HOG | Histogram of Oriented Gradients |
| PIL | Python Image library |
| GUI | Graphical User Interface |
| ML | Machine Learning |
| AI | Artificial Intelligence |
| UUID | Universally Unique Identifier |
| CNN | Convolutional Neural Network |
| MVP | Minimum viable product |

# Project Aims and Objectives

The original aims of this project have changed since the initial report. This was because of a change of approach to the given problem. The main objective is to allow a computer to differentiate between **aircraft** and **ground** images. This can be broken into several smaller objectives.

## Primary Objectives

1. To gather a set of aircraft and ground images to create a training and test set
2. To normalise the training and test set
3. Change the orientation of aircraft to ensure the face north
4. Resize the images so they can be pre-processed accurately
5. To pre-process the training set and label accordingly.
6. To Train an SVM with the Training set
7. To test the SVM with the test set
8. Further optimise the SVM by tuning decision boundaries
9. Cross-validate the training set
10. Display results of classification to user.

Assuming all primary objectives are completed successfully, additional objectives are to be attempted

## Additional goals

1. To attempt to recognise aircraft in a large image using user defined search criteria and show aircraft locations to user after search is completed.
2. To generate heat maps of large images to identify areas with a probability of containing aircraft
3. To attempt to recognise non-commercial aircraft such as private aircraft and helicopters.

Once all goals have been completed, a software program with realistic uses should be established. The training set can be varied to identify additional vehicles or identify completely different objects.

# Methods

## Internal and External Libraries

During the project, I have used a variety of libraries to aid the completion of the projects aims and objectives. Writing code to support all functionality provided by product would be very difficult to to achieve in the tight timescale given.

The language I have chosen for this project is python as it has a large range of image processing libraries available and offers a wide range of features such as object orientation. Internal libraries come as standard with the python package whereas external libraries are developed by third parties. All third-party libraries used during this project are used in accordance to licensing laws and agreements laid out in their terms and conditions.

#### OpenCV

The library used to read images from PNG format to NumPy arrays was OpenCV. OpenCV has tools for image manipulation and other image related functions. OpenCV was used to rotate images before training.

#### Scikit-Learn

Scikit-Learn is a well-documented Machine learning library that offers a wide range of tools for machine learning, data mining and data analysis. Scikit-learn has developed a Support Vector Machine that can be easily utilised for aircraft identification. The SVM forms the basis of my product and classifies objects as it sees fit.

#### Scikit-Image

Scikit-Image is a collection of image processing algorithms for the Python programming language. This library contains the HOG algorithm used in image pre-processing to allow the SVM to distinguish between objects in images.

#### FPDF

FPDF is a library that allows interaction between Python and pdf files. This allows the results of classification to be saved to and viewed by the user. This includes raw data that can be written to a table containing information about probability and image data that shows aircraft locations.

#### NumPy

NumPy is the fundamental package for scientific computing in Python [1]. It comes with a variety of features and functions such as mathematical, logical and shape manipulation. This makes it the perfect library to store images as arrays. Other libraries used during this project utlise NumPy often returning NumPy arrays after specific functions. During the training stages, images are rotated using NumPy’s rotate function.

#### Matplotlib

Matplotlib is a 2D plotting library that produces publication quality figures. During this project, I used its features to display results of classification to the user and allow them to interact with them using the libraries GUI. This allows them to zoom in and move Images and plots around as they wished. The user can also choose to save the output if they wish for later viewing.

#### Tkinter

Tkinter is the Python standard for GUI development. The library allows the development of complex windowed GUI’s. It allows the placement of buttons, labels, text boxes, images, radio buttons and drop-down menus. Tkinter also has GUI’s for functions such as file selection.

#### PIL

PIL is python’s image library that adds support for opening and saving images in multiple image formats. The library contains functions to easily manipulate images and draw shapes/ text over them. The shape functions have been utilised in this project.

## Data set

This chapter provides a description of the data required to train a ML model, how it was gathered and how suitable images were selected.

For a computer to recognise an object, first you must teach it what the object looks like. There are many different learning techniques and methods to do this. The technique I chose to implement was supervised learning. To carry out supervised learning, a training set must be created. The training set must show an accurate representation of the object or the model will fail to understand what it is identifying.

### Images

The data set currently consists of 250 images of aircraft and 250 images of ground. Images were obtained from google earth by taking screenshots of airports and proceeding to crop aircraft and ground. Cropped images are then saved as individual images. Cropped images of aircraft have little empty space around them so that the SVM can focus primarily on the aircraft. Examples of training images are shown in Figure 1.

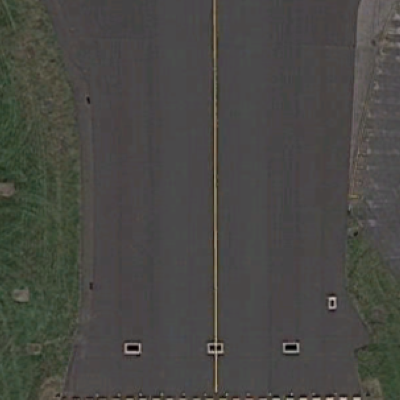


Figure 1: Image of ground (left) image of aircraft (right)

Originally, I ensured all images of aircraft were rotated so that the aircraft faced north. All images were also of the same dimensions (400x400 px) ensuring training was fair and balanced and are of the PNG format for simplicity.

However, after development of the image search classification method, it quickly became apparent that aircraft in real life can exist in many different sizes and rotations relative to image search criteria. At present, the user defines the search area size and movement settings meaning the SVM may not always have a perfect view of an aircraft. The aircraft may appear off centre meaning the SVM will struggle to classify it. To overcome this problem, I needed to generate a realistic training set. To do this I gathered images of some aircraft at random rotations and of varying sizes therefore filling different amounts of the image. This gave the SVM a more realistic understanding of how an aircraft appears in real life.

To again simplify the realistic data set, I ensured each photo contained no more than one aircraft. If an image contained more than one aircraft, it would interpret an aircraft as an object of two or made of two aircraft combined. Figure 2 shows an example of this occurrence.



Figure 2: Image not included in data showing two aircraft

Images of aircraft with complicated surroundings such as complex jet ways, terminals and vehicles in many cases were also disregarded. This is because it shifts the centre of attention of the SVM. The SVM will start to think an aircraft is an aircraft when it has an assortment of surrounding vehicles, buildings and jet ways in proximity. The idea behind the

### Parsing

An image in computer science is regarded as an array of values. Each field contains a value which is displayed onscreen as a channels colour intensity. To train a machine learning model, the training set is provided as an array of image data. The image data created by reading images from the file system using the OpenCV library. The OpenCV function to read images, returns a NumPy array of raw image data. The image data is then added to a large array containing other training data. The process is repeated for every image in the training set until the training data is contained in a single array. Concurrent to this process, a separate label array is filled with 1’s and 0’s. 1 = Aircraft, 0 = Ground. This array tells the SVM which image belongs to which classifier.

To add an element of further realism, the method that generates the training set for the large image search, manipulates images at random to remove any element of bias that may be incurred by the default training set. This is done before training. Images are rotated randomly by intervals of 90 degrees and then pre-processed (see next section). They are then added to the large training array.

## Pre-Processing

This chapter describes the methods used to process images before data is used for training and classification and defines how specific algorithms function and how the results3 yielded are of help to the overall system.

Pre-processing takes an image and extracts key information about it. There are several pre-processing methods that are available and commonly used in machine learning. Pre-processing is used to reduce the load on the machine learning model and only supply it with relevant information. When images aren’t pre-processed, the machine learning model is often slow and inaccurate. This is because of the excess data provided to it. Image pre-processing creates a feature vector that describes an image. The feature vector is then provided to the machine learning model for training and evaluation.

### Histogram of oriented gradients

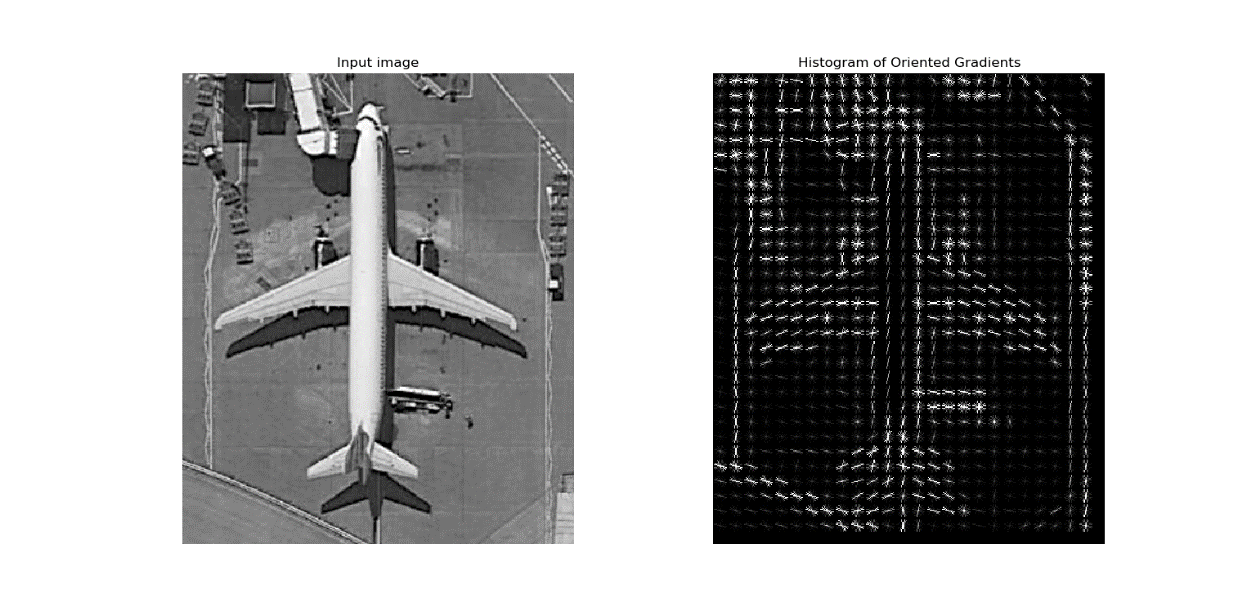
Histogram of oriented gradients is the current pre-processing method used in the parsing stage of the aircraft detection program. HOG counts the occurrences in gradient orientation in localized cells. The cells are normalised in blocks using contrast normalisation. A cell and blocks size can vary. Blocks can consist of any amount of cells. In this case, the blocks cells are normalised to ensure the results have low variance.

Contrast normalisation is a simple image enhancement technique that changes the range of image intensity values. Minimum and maximum values are supplied to a normalisation function which are applied to the image. This process is carried out to remove bias caused by illumination and shadowing in the given image.

After contrast normalisation is completed, the next step is to create a histogram for every cell in the image. Once the histogram is created, the orientation can be calculated. The direction of the gradient is added to a feature vector.

HOG was implemented in this program as it is widely used with SVM’s for object detection. Figure 3 shows a visualisation of the HOG algorithm on an aircraft from the training set.

Figure 3: Histogram of Oriented Gradients algorithm applied to a training image



The sizes of the cells are used in the HOG algorithm in the aircraft recognition program are 16x16px. The blocks consist of only one cell as this is the default value. If the SVM’s predictions were not accurate enough, the block sized can be increased if necessary, to take make full use of contrast normalisation.

### Feature vector

A feature vector is a vector containing information describing an objects important details and characteristics. It can describe features and characteristics of a small area of an image or can describe an entire image. A feature can take many forms such as average density, centroid (centre of image where all medians intercept), pixel colour count, gradient magnitude and orientation.

The feature vector produced by HOG in the aircraft recognition system contains number of blocks per row, number of blocks per column, number of cells per row, number of cells per column and number of orientations and, orientation of every cell processed. Take an image with dimensions of 400x400px:

|  |  |  |
| --- | --- | --- |
| **Image size: 400x 400px (RGB)** | **Calculation** | **Length** |
| **Image length flattened** | (400x400)x3 | 480000 |
| **Feature Vector** | (400x400)/16 | 10000 |
| **Percentage difference** | (480000-10000)/480000\*100 | -97% (470000) |

By applying the HOG algorithm before training, we can not only reduce the amount of information passed to the SVM by roughly 97% but, increase the accuracy of the data passed to it. This plays a large part in computational complexity as the training and classification of a single image is 48 times faster. It is argued the application of HOG can add complexity to the system however, HOG will not increase the complexity by the 48 times it was reduced. Figure 4 shows the difference between an aircraft and ground feature vector.

A screenshot of a computer

Description automatically generated

Figure 4: Ground feature vector (Red), Aircraft feature vector (Blue)

Figure 4 shows how the feature vectors obtained from HOG pre-processing vary dependant on the input image. The red points show that a ground image has quite a flat and consistent distribution of features as there aren’t many changes to gradient orientation. The blue points highlight the amount of changes to gradient orientation there are in a ground image. From this information, it can be predicted that high amounts of change to gradient orientation in an image point to the image being classified as an aircraft.

## Machine learning

This chapter explains the machine learning techniques used during this project to classify images of aircraft and ground.

Machine learning is an application of Artificial intelligence that allows computer to automatically learn and improve using statistical models, algorithms and access to data without explicit programming. Machine learning has multiple processes, the first being training. Training data is provided to the Machine learning model so it can observe the data and look for patterns. The more data fed to a machine learning model, the greater the understanding thus increase in accuracy. However, to get accurate and reliable results, the ML model must be passed reliable data otherwise its ability to predict correctly will be greatly flawed.

After the training stage is completed, the ML model can be passed test data. The model can evaluate the test data and return a prediction. In our case, whether the image contains an aircraft or is purely ground. After predictions are made, if correct, can be added to the training set to improve accuracy.

Optimisation is the final stage and can often make or break a ML models accuracy. Decision boundaries and functions created by the model can be altered by the input variables such as C, gamma and kernel. These essentially define the way the model can make decisions. While ML can analyse vast quantities of data and quickly yet accurately make predictions, it requires a large amount of time and resources to train it correctly.

### Supervised learning

Supervised learning is the process of providing a machine learning model with labelled training data. The ML model produces an inferred function to make predictions about the test data.

A set of N training examples in the form **{(x1, y1),…, (xN, yN)}**, **xi**representing the feature vector provided by HOG pre-processing, **yi** being the label to accompany the data that informs the ML model of its class, the ML model will produce a function in the form **g : X -> Y** where **X** is the input space and **Y** is the output space. The function **g** is referred to as the hypothesis space. Many ML algorithms are probabilistic meaning **g** takes the form of a conditional probability model where **g(x) = P(y|x)**. The model returns its prediction based on the highest probability.

To contrast, unsupervised learning allows a computer to draw its inferences from training data. This allows the ML model to group classes as it sees fit. This can yield unexpected results however after extensive training, become very accurate.

### Support vector machines

A Support vector machine is a supervised learning algorithm that analyses data. The common uses for SVM’s are classification and regression analysis. SVM’s are commonly used in real world problems such as protein classification, image classification, text classification and classification of handwritten characters.

A SVM is provided training data in the form **{(x1, y1),…, (xN, yN)}**, **xi**representing the feature vector provided by HOG pre-processing, **yi** being the data’s label. In our case, **xi** represents images of ground and **yi** is either a 1 or 0 dependant on the human classification of the image. The SVM plots a graph and plots feature vectors as individual points after kernel processing. After all feature vectors in the training set are plotted, a hyperplane is plotted between classes. This graph can exist in 3-Dimensions the hyperplane can exist as any shape. This can be seen as something similar to a line of best fit. An SVM can have as many hyperplanes as it wishes to group classes. The more hyperplanes present, the more complex classification becomes decision boundaries are challenged. Figure 5 shows an example of an SVM after training is completed.



Figure 5: An example of a support vector machine [2]

To make predictions, the SVM is provided with an unseen test set of feature vectors. This could be from a large image or smaller image. The SVM plots the feature vector and dependant on its position relative to the decision boundary, returns a probability and prediction.

#### Kernel

The kernel is a set of mathematical functions that control the way an SVM classifies data. There are several commonly used kernels such as linear, non-linear, polynomial and Radial Basis Function. The kernel often defines the shape of the hyperplane

This is the mathematical equation of the RBF kernel. RBF is commonly used as the default algorithm for ML models as it needs no prior knowledge about the dataset. Figure 6 shows the RBF kernel equation.

Laplace RBF kernel equation

Figure 6: RBF kernel equation

Feature vector **xi** and label **yi** is passed to the RBF kernel function. After computation is completed, the kernel returns coordinates to be plotted by the SVM. This equation is computed for every feature vector provided to the SVM during training and testing stage.

#### Hyper parameters and decision boundaries

The SVM can take parameters that help define decision boundaries. For many applications, the default values provided by the algorithm can be used. However, dependant on the type of data and shape, these may need to be tuned.

**C** is penalty parameter of the error term. This controls the distance of the margins from the hyperplane that ignores features that may have some form of error during training. Margins are shown either side of the yellow dotted line in Figure 5

**gamma** is the coefficient used in the RBF kernel along with many others. A high gamma value leads to high bias and low variance where as a low gamma variance leads to a small bias and high variance.

A hyperplanes margin is regarded as soft when it ignores features close to the hyperplane. This often leads to a better fit and overall increase in accuracy in many cases. A margin is hard when a it ignores no features and tries to divide classes perfectly down the middle.

#### Tuning

To optimally tune hyper parameters, a grid search algorithm is used. Grid search takes a set of seemingly random values and passes them to the SVM. In our case, C and gamma values are generated by decrementing from 10 and 1 respectively. The SVM is trained using its own data set and then tested. The grid search algorithm cross checks the prediction from the SVM with the human classifications obtained and data collection. Based on the highest scoring combination of C and gamma with the given test set, the algorithm chooses the optimal input parameters. We now know the values of C and gamma are optimised specifically for the purpose of classifying aircraft and ground images.

#### Under and Over fitting

Under and over fitting occurs when the hyperplane divides classes too closely or too loosely. Figure 7 shows different types of fits.



Figure 7: SVM hyperplane fit examples [3]

When a ML model is over fitted, it makes it difficult for future predictions to be accurate as they must fit a very specific criteria therefore often fail to reach the correct result. Overfitting occurs when a hyperplanes margin is hard.

Under fitting occurs when the ML model fails to gain an understanding of the underlying structure. This can be a result of selecting the wrong kernel with the shape of the data provided at training.

### Cross validation

k-Fold cross validation is a resampling technique used to evaluate a ML models accuracy and performance on a limited data set. It is often used to gain an insight into expected results with an unseen data set. The technique has a single parameter referred to as k. Parameter k defines the number of groups the total data set will be split into. If k = 10, the process will be called 10-fold cross validation. Cross validation is a popular method because it is simple to understand and usually results in less biased and optimal estimate of the model as opposed to a simple train/test split. It also shows how accurate the training data is. If the cross validations core of the training data is low, then the model may struggle to evaluate unseen test data.

The steps for cross validation are as follows:

1. Randomize the datasets order
2. Split dataset into k groups
3. For each group create

3a. Retain group for test

3b. Train model using remaining groups and evaluate accuracy with test set

3c. Retain the evaluation score and discard model

1. Summarize the skill of model using k evaluation scores

In most cases, the scores are averaged to create the most realistic score for the training set. To evaluate the performance of the aircraft recognition program, I cross validated the standalone and realistic data set.

#### Standalone dataset

|  |  |  |  |
| --- | --- | --- | --- |
| **10-fold CV of standalone dataset** | **Lowest Fold** | **Highest Fold** | **Average** |
| **400 x 400 px** | 0.94 | 1 | 0.99 |

Results of 10-fold cross validation of standalone dataset show that the accuracy of the ML model and data set is incredibly high at 99%. The variance between lowest and highest fold is 6% meaning the model is incredibly accurate. 8 folds achieved 100% during this test.

#### Realistic dataset

|  |  |  |  |
| --- | --- | --- | --- |
| **10-fold CV of realistic dataset** | **Lowest Fold** | **Highest Fold** | **Average** |
| **800 x 800 px** | 0.37 | 0.99 | 0.74 |
| **400 x 400 px** | 0.66 | 0.96 | 0.75 |
| **200 x 200 px** | 0.66 | 0.67 | 0.67 |
| **100 x 100 px** | 0.60 | 0.61 | 0.60 |

Results of 10-fold cross validation of realistic dataset show that when image size is increased, the average cross validation score increases however the variance between highest and lowest fold also increases. As variance was originally higher, images had to be deleted that didn’t confirm to dataset rules.

# Technical Achievement

This chapter explains the functionality of the finished product developed throughout the project.

## Graphical user interface

The graphical user interface is the starting point of the program. The user is able to select classification method and options for the image search classification process. Figure 8 shows the windowed GUI that the user is shown the following execution.

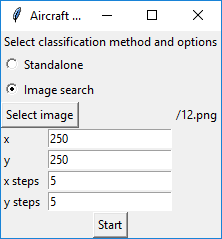


Figure 8: GUI shown to user after program execution

The user must first select classification method. Standalone classification requires no other input from the user. The user can now press Start to begin the training and classification process.

If the user wishes to search for aircraft in a larger image, the image to search must be selected. Clicking the select image button allows the user to navigate to the location of the image they wish to classify and select it. Figure 9 shows the window used to select images from files system



Figure 9: File selection dialog

After image to search is selected, the user must specify;

* x- The x dimension of the search area in px
* y- The y dimension of the search area in px
* x steps- The amount of moves along the x axis
* y steps- The amount of moves along the y axis

From this data, the program selects suitable x and y step values to ensure image boundaries are met.

After the user has selected search criteria, the Start button can be pressed to start the image search process.

## Standalone classification

Standalone classification is the recognition of aircraft images where images from a data set are selected at random for training and classification. There are 200 images in the standalone dataset. 100 images of ground and 100 images of aircraft. 5 images of aircraft and 5 images of ground are selected for testing. The remaining 180 images are used to train the ML model.

The images are put through usual process of parsing, HOG pre-processing before being passed to the SVM. After training is complete, test images are passed to the SVM. The SVM returns a set of predictions for the images tested. To show the user the results, the test images are displayed in a grid with accompanying predictions and probabilities. The probability shows how sure the ML model is that the image belongs to a class.

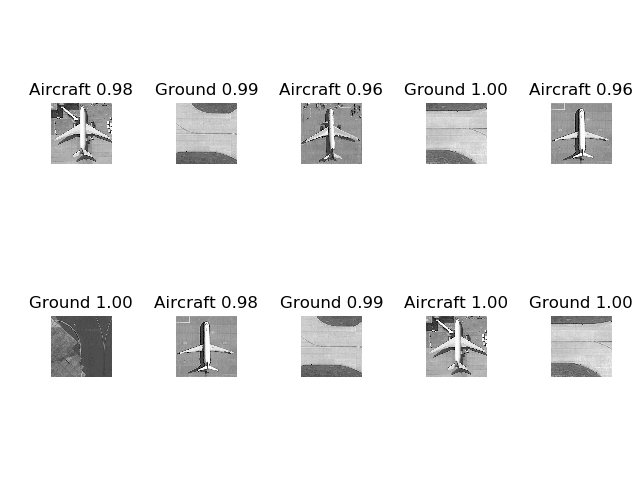


Figure 10: Results from standalone classification

The results in Figure 10 show that the program correctly identified all ground and aircraft. The accuracy of standalone classification in all test cases is 100%.

## Large image search

Large image search requires a more complex data set. The data set has more samples and the data in the image is more representative of aircraft in a real-world setting. Aircraft in a real-life setting appear at random locations with different backgrounds, are of different sizes and appear in a multitude of orientations. Training the SVM for large image search Is slightly more complicated than training for standalone classification. First all training images are resized and to ensure no element of bias is incurred. The training images already show aircraft at a variety of random orientations. On top of that, images are rotated in random increments of 90 degrees .

The basic principle behind large image search is that the user defines search criteria to the program. The program iterates the image searching in small portions for aircraft. The execution behind it is slightly more complicated.

The image is first divided into several smaller images. The image sizes are defined by the users input. If the user has given search settings that mean the images overlap, parts of the large image will be processed more than once. If the user has supplied search criteria that mean the search boxes do not overlap, some areas will be skipped. This meant the image will not be classified correctly. Originally the user defined the number of pixels in each step however this meant that boundary data was often ignored. This meant the bottom or right row would not be included in classification. To overcome his problem, the user defines how many steps they wish to carry out. The higher the step count, the more the overlap of search area.

The x and y steps are calculated as such

x\_step = image\_width – x\_search\_size / x\_steps

y\_step = image\_height – y\_search\_size / y\_steps

This ensures that no area of the image is missed.

The smaller images are parsed, HOG pre-processed and passed to the SVM for evaluation. The SVM returns a prediction array with associated probabilities. To display results to the user, there are two forms of output. Heat map and the original image with boxes overlaid to show the location of aircraft.

Heat maps highlight areas of the image that show a high probability of aircraft. They are generated by first creating a black image that is identical in size to the original search image. The probability of an area is plotted on the image by calculating the colour intensity values based on the probability. Once all areas are plotted, adjacent search areas are averaged to blend colour together.

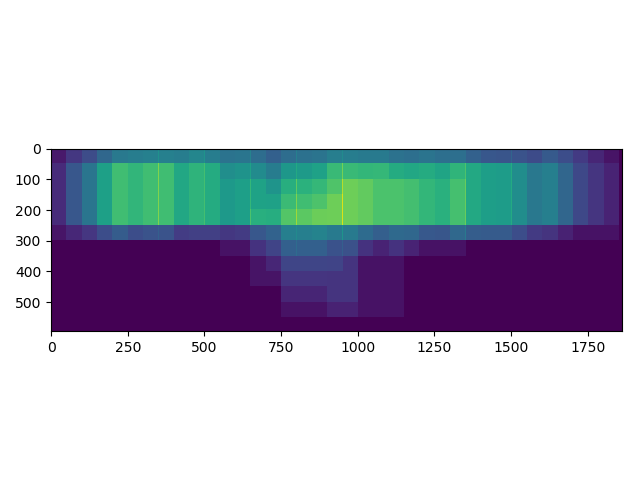


Figure 11: Heat map of aircraft probabilities

Figure 11 shows the results of an image search as a heatmap. The bright areas indicate the SVM has predicted there is a high probability of aircraft at that specific location.

To show the user the location of discovered aircraft, the original image is taken and copied. The predictions obtained from the SVM are used to determine the location of aircraft. Boxes are placed on the image to show the locations clearly to the user.



Figure 12: Image search results

Figure 12 shows where the SVM has found aircraft. The boxes overlap as the program was provided with overlapping search criteria. Accuracy of large image search around 60%. This is because it is difficult to provide search criteria that align the search area perfectly with aircraft. This means several aircraft are overlooked. It is difficult to reduce the number of boxes in a tight location because it is an area of interest.

## Saving of results

After classification of a large image is completed, the results are automatically saved to the users file system. The heat map and search results are assigned a UUID and saved to a folder. The newly generated images will have the same UUID for simplicity.

The results of classification appear in their own GUI window. The user is given options to move plots, zoom in and out, and further save the view. They can select choose a name for the results and define format before saving. The library used for this process is Matplotlib. The library used to automatically save the results is PIL.

INSERT IMAGE OF MATPLOTLIB

# Project Planning

During this project, I used CSEE Jira to manage issues arisen and tasks to be completed. The platform allows the project to be divided into tasks, sub-tasks, stories and epics. Main sections of the project are divided into stories and epics. For example, I created an epic called “GUI”. All tasks that relate to GUI programming are linked to the GUI epic. The same process was repeated for tasks that fell under technical documentation, final report, image segmentation, image capture and SVM. During every meeting with my supervisor, we agreed on work to be completed. Feedback of the meeting was left on the platform so that I could break it into smaller tasks and move to solve these issues.

## Momentum

To maintain momentum during the project, I stuck to a strict schedule to ensure the weekly 15 hours of work was met. This ensured the project received regular updates and changes to meet the final finished product. While I was working on tasks completed, I moved them to the in progress column of the table. I would log work on the task and talk about what was done in the allotted time. After a task was completed, I would test the task in progress works and interacts correctly with the system. If this was the case, I would move the task to the done column of CSEE Jira.

On occasion, bugs would plague the progress of small tasks such as formatting the training set data structure so that it would be accepted by the SVM. This took a large amount of time out the project as this specific task was blocking the progress of the rest of the project.

Approaching the Christmas break, the projects momentum slowed a little as deadlines for other modules approached however picked up nicely towards the end of the Christmas vacation. Since then, momentum has been in line with the requirements of the module.

## Adapting to change and dealing with risks

Adapting to change and risk during a project is always difficult. This is because it is hard to foresee problems that may arise. For example, a project may be based around a certain approach. If the approach fails, the project could collapse leading to a waste of resources. Project collapse happens very often and is often why new projects can struggle to gain momentum during early stages.

At several points during the project, I had to change the approach to a problem I was having. This is difficult to implement but often required in large projects. Time is often lost as the previous approach may be completely shelved and not used again hence a waste of man hours. During the early stages of the project, the original goal was to use a Convolutional neural network to classify aircraft and ground images. However, I quickly discovered that the data set required for accurate training of a CNN would be much larger than the data required for accurate training of an SVM. The switch was a difficult decision to make but the general consensus was the time required to create the data set for a CNN would hinder the progress of the project. The risk associated with the change was high as I would need to carry out new research on the project.

As this project used external libraries, there is always a danger that a new version of the library could become available. When a new version of a library becomes available, there is always a risk that the updated version may be incompatible with the current software through the deprecation and addition of methods and classes. This risk can be minimalised through monitoring of releases and specifying which version of the given library to be used in the requirements.bat file. This ensures the software can continue to function as anticipated.

## Performance

Overall, I feel I performed well during this project. I managed my time effectively, balanced my learning and deadlines from other modules and kept the project alive. The only part that has let me down so far is quality of research. The first few weeks of development were wasted as the approach I had research was deemed inappropriate to the given solution and timeframe. I started the project using the wrong ML model. This was a quick and easy fix however, could have been easily avoided by in-depth research of CNN’s. The extra research required by the change was additional work that I didn’t want to worry about and could have been undertaken at the research stage.

My personal achievement during this project was the development of a highly accurate standalone classification process and the development of large image search. Although the large image search shows lower accuracy scores of around 50-60%, to develop a program that detects aircraft in the given surroundings is a difficult task to undertake.

## What have I learnt?

During this project, I have learnt the importance of project management and gained an understanding of why large projects need to be managed effectively. The use of CSEE Jira really enabled me to track issues and look at my overall progress. It allowed me to identify issues within the way I work through the use of its analytic tools. I discovered I was taking too much time to solve small issues that really had no real impact on the overall project or that weren’t part of the project objectives. For example, development of a GUI that looks nice was a smaller priority than the overall objective of identifying aircraft however, I spent more time than I should have trying to perfect it when there were underlying issues with the classification. This issue links closely to time management as I now understand how important it is to prioritise tasks in order of importance to the MVP and overall project objectives.

I have discovered how important the quality of research for a new project can be. If the research is of a poor standard, the project may fail as it may begin using the wrong methods or working towards unachievable objectives. Poor research can lead to project collapse as the product may not even be required in the current market.

# Conclusion

Object detection is a highly complex problem that has taken many years to develop to the stage it has reached at present. This project highlighted the computational complexity of processing large amounts of data and how systems must be optimised for specific purposes to perform effectively.

The development of the aircraft identification system involved many steps. First the development of a training set. The training set defines what the ML model will be classifying. This step was time consuming as the repeated steps of cropping images from Google Earth was tedious. What to define as aircraft and ground was also a difficult task to undertake as results can only define how accurate the dataset is.

The

ocesses

What the results mean

The project shows the possibilities of computer vision and machine learning. Not only can it be used in private but to help the general public. Computer vision and machine learning can assist in areas such as healthcare, security, banking, agriculture, automotive and industrial.

# References

[1] <https://docs.scipy.org/doc/numpy-1.13.0/user/whatisnumpy.html>

[2] <https://medium.com/deep-math-machine-learning-ai/chapter-3-support-vector-machine-with-math-47d6193c82be>

[3] <https://medium.com/greyatom/what-is-underfitting-and-overfitting-in-machine-learning-and-how-to-deal-with-it-6803a989c76>

# Appendices

## Sustainability

Sustainability of software defines how software may be available in the future and the impact it will have and how it can be updated and adapted to meet newer needs. The software I have produced can be adapted to classify different data and theoretically an unlimited amount of classes. This means that differentiating the data set allows the software to classify completely different material. This allows the needs of the software to adapt to change. The program is runnable on the majority of operating systems.

## Legal

As computer vision and machine learning are such new topics, the legal issues surrounding the two subjects are unclear at best. This raises issues itself as the government is struggling to define laws at the speed that technology is being developed at.

There are specific laws that disallow the use of image capture equipment from being used in specific environments. However there are no clear rules about the processing of that data.

## Ethical

Many people are scared by the fact that technology can classify images. It’s a relatively new concept and provides very little insight to the general population.

WHAT ARE THE EHTICAL ISSUES SURROUNDING THE PROJECT

INVASION OF PRIVACY

## Intellectual property

Intellectual property is the system that aims to allow peoples creations to earn them recognition and financial benefit. It is supported by law through copyright, patents and trademarks. This allows a person’s creation to be protected from third parties. As I used third party libraries to develop my project, it raises the issue of who actually owns the entirety of the program. This is where licensing comes into play. Each library I have incorporated in the final project has an associated license.

The Berkley Software Distribution (BSD) Licence is a permissive license that imposes minimal restrictions on the use and distribution of covered software. Scikit-learn, Scikit-image, Tkinter, OpenCV and Matplotlib are covered by this license.

PIL follows a license similar to an MIT license. An MIT license is also a permissive license that enforces very limited restriction on reuse therefore has excellent licence compatibility. PIL gives permission to use, copy, modify, and distribute.

Both licences used allow the near unrestricted use and distribution of their software if the license agreement is copied and used in the release of the software. This means the software I have created is entirely owned by me.