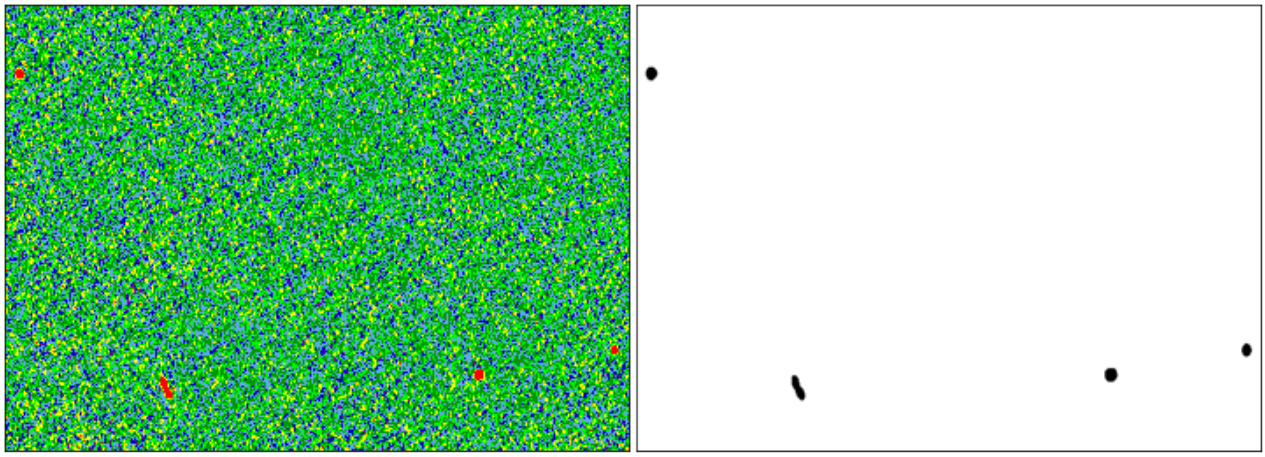
A comparative study of source finding techniques in H I emission cubes using SoFiA, MTObjects and supervised deep learning



Astronomy Masters Thesis: Data Science and Systems Complexity Track
Kapteyn Astronomical Institute, University of Groningen

Github Repository

Jordan Barkai (s3972097)

Supervisors:
Prof. M.A.W. Verheijen, Dr. E.T. Martínez, Dr. M.H.F. Wilkinson

January 14, 2022

Acknowledgements

This dissertation would not have been possible if it weren't for the great deal of support and assistance I received both professionally and from loved ones.

I would first like to thank all three of my supervisors, Marc Verheijen, Estefanía Talavera Martínez and Michael Wilkinson, who all come from different professional backgrounds and patiently worked together to guide me through this research. Each of them brought a different perspective and a wealth of knowledge that helped me tackle this research with well rounded support. In particular I would like to thank Marc and the Nederlandse Onderzoekschool Voor Astronomie' (NOVA) for making the opportunity possible for me and providing the funding for my studies.

I would like to acknowledge my colleagues from the H I research group, especially Doctor Julia Healy who constantly reassured me that I'd come out the other side. I would particularly like to thank Leon Doddema, from the Kapteyn Computer Group, for being so patient in helping me with all the technical problems I encountered throughout my research.

I would also like to thank my parents for always supporting me and inspiring me to push myself further. In addition, I would like to thank my partner, Alexander Weiss, for all the love and support and for helping me through times of doubt. Finally, I would like to thank Elena Colotti for always being there for me and for making Groningen a place I call home.

Abstract

Astronomical surveys map the skies without a specific target, resulting in images containing many astronomical objects. As the technology used to create these surveys improves with projects like the SKA, an unprecedented amount of data will become available. Hence the need for fast and accurate techniques to detect and locate sources in astronomical survey data. The challenge lies in the lack of clarity in the boundaries of sources, with many having intensities very close to the noise, especially in the case of radio data, resulting in a trade-off between accepting false positives and excluding true sources. This project therefore aims to find the best method for finding the most sources and least artifacts with the best mask quality in three-dimensional neutral hydrogen cubes. In this project the various existing methods are explored, including the traditional statistical approaches and machine learning techniques in attempt to create a pipeline to optimally mask and label the sources in 3D neutral hydrogen (H I) data cubes. This was achieved by first testing two traditional source finding methods, the well established H I source finding tool SoFiA and one of the latest best performing optical source finding tools, MTObjects. A new supervised deep learning approach was also tested, in which a 3D convolutional neural network architecture, known as V-Net, originally designed for medical imaging was used. These three source finding methods were also further improved by adding classical machine learning classifiers as a post-processing step to remove any false positive detections. The pipelines were tested on H I data cubes from the Westerbork Synthesis Radio Telescope (WSRT) with additional inserted mock galaxies. Due to work in other fields, such as medical imaging, it was expected that the best pipeline would involve the V-Net network combined with a Random Forest classifier. This, however, was not the case and instead it was a competitive second after SoFiA combined with a Random Forest classifier. This is suspected to be due to the many more mock sources in the training set than the real sources. There is therefore room to improve the quality of the V-Net network with better labelled data such that it can potentially out-perform SoFiA.

Keywords— Source finding, segmentation, data cubes, neutral hydrogen, machine learning, three-dimensional, max-trees, deep learning, convolutional neural networks

Contents

1 Introduction **6**

2 Relevant Previous Work **9**
2.1 Traditional Segmentation Methods . . . 9
2.2 Classical Machine Learning Segmentation Methods . . . 10
2.2.1 Support Vector Machines . . . 11
2.2.2 K-Nearest Neighbors . . . 11
2.2.3 Decision Trees . . . 12
2.2.4 Random Forests . . . 13
2.3 Deep Learning Segmentation Methods . . . 15
2.3.1 Volumetric Segmentation with Deep Learning . . . 17

3 Materials **19**
3.1 Observed Data . . . 19
3.2 Synthetic Data . . . 21
3.3 Inserting the Mock Galaxies . . . 25

4 Method **30**
4.1 SoFiA . . . 30
4.2 MTObjects . . . 31
4.2.1 MTO Architecture . . . 31
4.3 Convolutional Neural Networks . . . 34
4.3.1 Transfer Learning . . . 35
4.3.2 The Learning Architecture Design of V-Net . . . 35
4.4 Identifying real sources . . . 37
4.5 Post-Processing Machine Learning . . . 38

5 Experimental Setup **40**
5.1 Experiment 1: Comparing all three methods . . . 40
5.1.1 SoFiA Implementation . . . 41
5.1.2 MTObjects Implementation . . . 42
5.1.3 V-Net Implementation . . . 44
5.2 Experiment 2: Machine learning post-processing . . . 45
5.3 Experiment 3: Improving the V-Net network with detections of real sources . . . 46
5.4 Evaluation Metrics . . . 46

5.4.1 Merging Scores . . . 47
5.4.2 Physical Properties . . . 47
5.4.3 Degree of Asymmetry . . . 48

6 Results **49**
6.1 Experiment 1 . . . 49
6.1.1 Identifying real sources . . . 50
6.1.2 Evaluation . . . 58
6.2 Experiment 2 . . . 62
6.3 Experiment 3 . . . 65
6.4 Comparing Experiments . . . 68

7 Discussion **69**

8 Conclusion **72**

List of Figures

2.1 A schematic diagram of hyperplanes in both 2D and 3D feature spaces 11
2.2 A schematic diagram showing the decision boundary between two classes in a data-set determined by a KNN model 12
2.3 A schematic diagram showing the trade-off between the bias and variance of a model when choosing the model complexity 13
2.4 A schematic diagram of a Decision Tree 14
2.5 A schematic diagram of a Random Forest 14
2.6 A schematic diagram of a neuron in an Artificial Neural Network 16
2.7 An example of a convolution in a Convolutional Neural Network 16

3.1 The layout of the 40 Apertif compound beams and an example of a single channel Apertif H I cube 20
3.2 An example of real and mock sources and their corresponding global H I profiles 24
3.3 An example of a normalised Apertif cube with inserted galaxies 28
3.4 The summary of the mock galaxies and their properties 29

4.1 A schematic diagram of the source finding pipeline of MTObjects 32
4.2 The V-Net architecture diagram 36

6.1 The training and validation loss score of the V-Net network per epoch during the training process for Experiment 1. 50
6.2 Examples of masked H I moment zero contours of detections of real sources overlaid on their optical counterpart for each method. 51
6.3 A box plot of the degree of asymmetry of real sources and mock galaxies detected by every method. 52
6.4 Examples of masked H I moment zero contours of false positive detections overlaid on their optical counterparts for each method. 53
6.5 Venn diagrams of the number of each kind of detection by each method in Experiment 1. 54
6.6 Venn diagrams of the number of each kind of detection by each method in Experiment 1 continued. 55
6.7 The evaluation metrics for each method for the mock galaxies in each spectral window evaluated. 59
6.8 The 25th, 50th and 75th percentiles of the physical attributes of detected mock galaxies by each method per spectral window 60

6.9 Examples of masked H I moment zero contours of the detections of mock galaxies for each method. 61
6.10 The purity and completeness of each source finding method after using machine learning classifiers to post-process their detections. 63
6.11 Venn diagrams of the number of each kind of detection by each method in Experiment 2. 65
6.12 The training and validation loss scores of the re-trained and fine-tuned 3D CNN per epoch during the training process of Experiment 3.1 and 3.2. 66
6.13 The evaluation metric for each V-Net network for mock galaxies and real sources in each spectral window. 67

List of Tables

3.1 The properties of the seven spectral windows of the two pointings chosen from the Apertif medium-deep survey. 21
3.2 The number of galaxies inserted into the seven spectral windows of the two pointings chosen from the Apertif medium-deep survey. 27

5.1 The chosen parameters for SoFiA 2.3.1 42
5.2 The chosen parameters for MTObjects 43

6.1 The elapsed real time in seconds for each of the three source finding methods for a single spectral window of size 2.81664 Giga voxels or $652 \times 1800 \times 2400$ voxels. 50
6.2 The number of mock galaxies detected per method per spectral window of Pointings 1 and 2. 57
6.3 The time taken for each machine learning model in Figure 6.10 to both train and infer whether a value in the catalogue of all the source finding methods was a true source or not. 62
6.4 The importance of each feature determined by the Random Forest classifier. . . 64
6.5 The mean combined scores of all source finding pipelines. 68

Chapter 1

Introduction

Astronomical surveys map the skies without a specific target, resulting in images containing many astronomical objects. As the technology used to create these surveys improves with projects like the SKA (Weltman et al., 2020), an unprecedented amount of data will become available. Besides the time that it would require, manually detecting sources in data-sets of this volume would be unfeasible due to the low intensities of many sources, increasing their proximity to the level of noise (Haigh et al., 2021). Hence the need for fast and accurate techniques to detect and locate sources in astronomical survey data.

The problem at hand can be seen as an overlap between fields, where existing advances in computer vision could be used to solve the struggle of source finding in astronomy. In computer vision object recognition encompasses a collection of tasks, including object segmentation, which will be referred to interchangeably with source finding in this report. Object segmentation can be defined as drawing a mask around identified objects in an image and assigning them class labels (Kaur and Kaur, 2014). This is done by highlighting pixels (or voxels in the case of 3D data cubes) belonging to different objects such as astronomical sources and noise or background.

The challenge lies in the lack of clarity in the boundaries of sources, with many having intensities very close to the noise, especially in the case of radio data (Punzo et al., 2015). Additionally, as the sensitivity and depth in astronomical surveys will increase, so will the number of overlapping sources as fainter and more extended sources are observed (Burke et al., 2019). This concept is known as blending. Having this foresight, many astronomers have explored source finding and deblending solutions using simple statistical techniques (Jarvis and Tyson, 1981), (Bertin and Arnouts, 1996), (Andreon et al., 2000). However, these approaches are very sensitive to the input parameters and have been found to struggle with differentiating between faint sources and noise, resulting in a trade-off between accepting false positives and excluding true sources. While object segmentation in 2D images is considered a solved problem these days, the same task in 3D data cubes is still very new and unexplored in both astronomy and computer vision.

The 21 cm spectral emission line of atomic neutral hydrogen (H I) is one of the primary wave-

lengths observed in radio astronomy. It is detected via the emission of a photon due to the energy level transition at 21cm of the hydrogen atom. H I gas is observed as clouds in galaxies, which can be used to determine their structure, as well as free-floating gas outside of galaxies (Maddox et al., 2021). H I emission is intrinsically faint and sensitive to its environment and therefore large volume coverage and depth are required to map the H I Universe. Hence the need for large redshift range surveys using radio telescopes such as the precursors to the Square Kilometre Array (SKA) (Weltman et al., 2020), Meerkat (Booth et al., 2009) and ASKAP (De Boer et al., 2009), as well as the APERture Tile in Focus (Apertif) (van Cappellen et al., 2021). In order of size, Meerkat is a 64-dish array in South Africa, ASKAP consists of 36 dishes in western Australia, and Apertif is a 12-dish array upgrade of the Westerbork Synthesis Radio Telescope (WSRT) (Hogbom and Brouw, 1974) in the Netherlands. The radio spectral line emission of galaxies is captured and processed into floating point values whose intensities represent flux in a three dimensional mosaicked H I data cube. The cube consists of two positional dimensions (right ascension and declination) and one spectral dimension (frequency or velocity), with a high resolution to the order of hundreds of Gigavoxels (Moschini, 2016).

Sensitivity to instrumental and environmental effects cause interferometric radio data to be intrinsically noisy while most radio sources are faint and extended (Punzo et al., 2015). Due to the dependency of these effects on direction and their variation across the field of view, the noise that needs to be differentiated from actual sources is non-uniform and sometimes very similar to the signal of the sources (Gheller et al., 2018). Additionally, H I sources lack well-defined boundaries and have a low signal-to-noise ratio (Aniyan and Thorat, 2017). This has resulted in a trade-off between accepting false positives and excluding true sources with the existing source finding methods used.

This project therefore aims to solve the dilemma of this trade-off by creating the optimal source finding pipeline with better completeness, purity and mask quality than previous methods. To this aim, the contributions of this work are four-fold:

- This is the first work, to our knowledge, that provides the training and testing of a deep learning segmentation architecture on H I cubes which was originally created for medical imaging. This model could be adapted for data from different telescopes and could be used as a source finding method in future H I surveys.
- A classical machine learning solution is proposed for improving the purity of the catalogs created by source finders. This is trained on various methods to ensure its ability to be used on any H I source catalog.
- A new pipeline for optimally finding and masking H I sources in data cubes is introduced. This is achieved by comparing statistical and deep learning methods using the well-known tool SoFiA (Serra et al., 2015) as the baseline for the experiments. The final pipeline consists of a source finding method followed by a machine learning model for post-processing.
- A catalog of H I sources and their associated properties is made available for future use. These were created by the various experimented with pipelines and can be used for training classification models in the future.

An outline of relevant previous work on image segmentation and source finding is covered in Chapter 2. This is followed by an explanation of the materials or data used for this project in Chapter 3 and the theory behind the different source finding methods used in Chapter 4. An outline of the experiments undertaken, how they were implemented and how they were evaluated is covered in Chapter 5, followed by the results of each experiment in Chapter 6. Finally these results are discussed and compared in Chapter 7 with the conclusion and exploration of the potential for future work in Chapter 8.

Chapter 2

Relevant Previous Work

In this chapter the existing methods currently used for object segmentation in computer vision research are mentioned and their applications in astronomy are discussed. The existing detection techniques can be categorised into three types of approaches. The first are traditional methods, which use statistical techniques to identify sources, followed by shallow machine learning techniques and finally deep learning approaches, including convolutional neural networks (CNNs). Since most source finding techniques are originally designed for 2D images, pixels will often be mentioned, but it is important to note that this project worked with three-dimensional data made up of voxels.

2.1 Traditional Segmentation Methods

The two most common traditional source finding techniques in astronomy are thresholding and local peak search (Masias et al., 2012). The simplest of the two is thresholding, where a collection of connected pixels are masked as a source if they are above a specified threshold value. The threshold can be global to the image but due to background variations it is often chosen locally, depending on the neighbourhood, and in some cases an adaptive threshold is used. SExtractor (Bertin and Arnouts, 1996) is one of the most well-known source finding programmes in astronomy which uses a thresholding method. However, despite its high speeds and simplicity, thresholding methods have difficulty finding faint sources. The other most widely used traditional method is local peak search, which creates a list of candidate pixels of the central points of a source, chosen by the local maxima. The pixels around each candidate that decrease in intensity are considered a source. This method was already in use in the late 70s (Argueso et al., 2006) but is more suited for detecting point sources and struggles to find extended sources. A common weakness of both thresholding and local peak search methods are their difficulty in de-blending or differentiating between multiple sources (Haigh et al., 2021).

While thresholding and local peak search are the most common classical source finding techniques used, in recent years a variety of methods have been attempted, inspired by other computer vision applications. A popular classical segmentation technique in computer vision is edge based segmentation. First the algorithm detects the edges of a source by comparing their first derivative of intensity to a given threshold and then it connects them to create a

mask (Alamri et al., 2010). However, this is more appropriate for objects with well-defined edges, and therefore is not suitable for the detection of H I emission sources. Another popular classical segmentation method is the watershed transform (Bandara, 2018). This method treats the image as a topographic surface and interprets the gradient of the pixel intensity as the elevation. It then segments the image by acting as if water is poured into the minima and treats the boundaries as where the puddles would merge. This has been used in astronomy for source finding (Aptoula et al., 2006), where over-segmentation was avoided by considering only a few marked minima as water sources found with thresholding. While the resulting masks have continuous boundaries and are more stable than edge-based methods, the watershed approach suffers from the complexity of calculating the gradients (Kaur and Kaur, 2014).

Region based segmentation, while popular among computer vision segmentation tasks, has only recently been introduced in the context of astronomical source finding. The approach is essentially to segment regions together that share similar characteristics. This is usually achieved by growing regions from initial pixels or by iteratively dividing or splitting an image into regions and merging those that are similar and nearby. These methods have the advantage of being less sensitive to noise, particularly when prior knowledge can be used to define the similarity criteria. However, region based segmentation methods are also known to be memory and time intensive. A recent region based segmentation tool in astronomy is MTObjects (MTO) (Teeninga et al., 2016), which makes use of max-trees, see Chapter 4. This tool was originally designed for 2-dimensional optical data and is extended further for H I emission cubes by Arnoldus (2015).

Weighing up the pros and cons of each of these methods, the best of source finding solution for segmentation therefore depends on the astronomical data itself. Hence the creation of a general source-finding tool, the flexible Source Finding Application (SoFiA) (Serra et al., 2015). SoFiA uses a combination of source finding techniques and was designed to be independent of the type of data used. Currently, this is the most well-used pipeline for source finding in H I emission cubes and was therefore used as the benchmark for the experiments in this report.

2.2 Classical Machine Learning Segmentation Methods

In addition to the traditional statistical methods used for object segmentation, there are many machine learning approaches. Classical or shallow machine learning methods learn on the characteristics of the observational data, known as features, as opposed to learning directly from the raw data itself and determining its features as deep learning does. These features are chosen to represent the physical properties of the system and therefore their quality determines the efficiency of the learning algorithm. It is therefore important to have a good understanding of the data being used, and in this case radio astronomy, to implement these techniques.

Both classical and deep machine learning methods can be either supervised or unsupervised. Unsupervised methods find the boundaries of objects using their intensity or a gradient analysis which makes them better suited for identifying sources with well-defined boundaries. Supervised methods instead use prior knowledge through training samples. Since the H I sources in the data cubes are known to have ill-defined boundaries, this report will focus on supervised

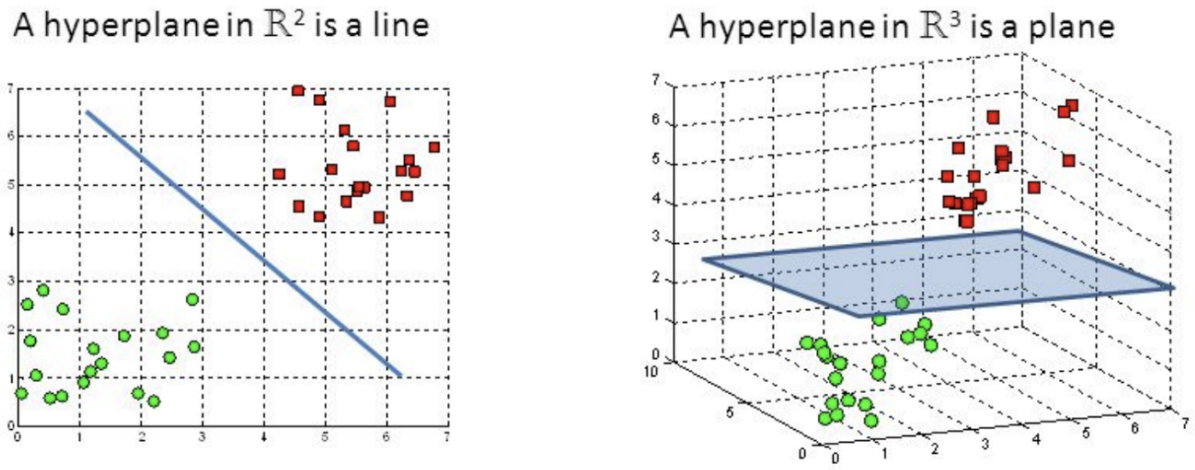


Figure 2.1: A schematic diagram of hyperplanes in both 2D and 3D feature spaces, taken from Gandhi (2018).

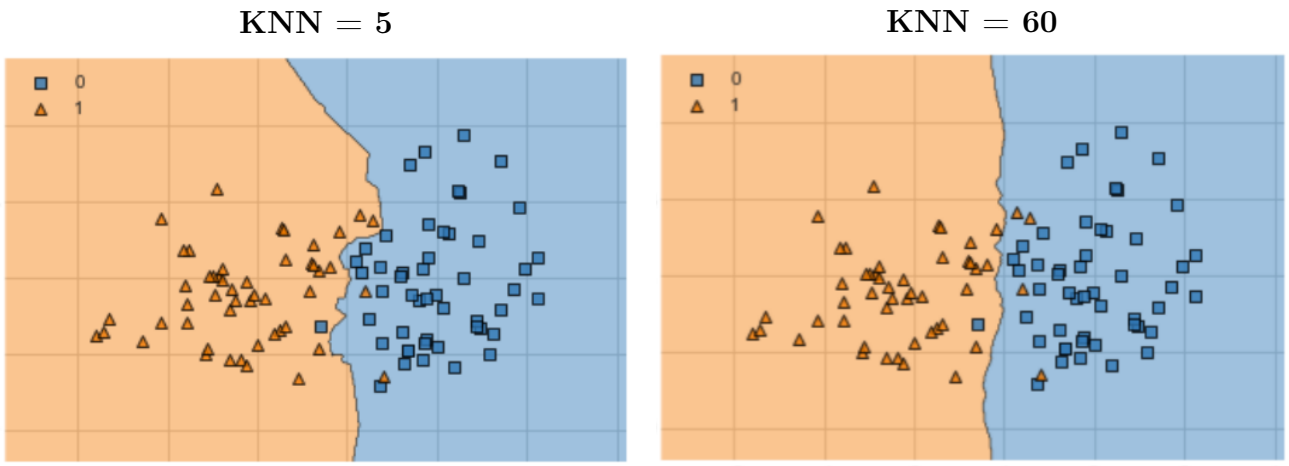
machine learning segmentation techniques. The most well-researched techniques use classification as a way of segmentation, including Support Vector Machines (SVM), K-nearest neighbors (KNN) and Decision Trees. The mathematical description of these algorithms can be found in Section 4.5 and a comparison of these methods for the classification of galaxy morphology can be found in Cheng et al. (2020).

2.2.1 Support Vector Machines

Support Vector Machines (SVMs) (Vapnik, 1995) is a kernel-based classification technique. Sources are separated into groups by a set of hyper-planes in an N-dimensional space, where N is the number of features. The optimal hyper-plane of separation is chosen to minimise the distance between the data points of different classes. This is done to ensure the confidence of the model in future classifications. An example of these hyperplanes separating two classes of data can be seen in both the two-dimensional and three-dinmensional feature spaces in Figure 2.1. This method has many advantages, namely its lack of sensitivity to outliers, the minimal number of free parameters needed and its computational efficiency. However, the results are not always transparent and it can struggle to perform on noisy data. In astronomy, SVMs are very popular due to their ability to handle data of high dimensions, see Zhang and Zhao (2014) for a summary of these applications.

2.2.2 K-Nearest Neighbors

K-Nearest Neighbors (KNN) (Cunningham and Delany, 2007) is one of the most extensively used pattern recognition methods, not excluding astronomical classification tasks such as Kügler et al. (2015). The way in which KNN classifies each test data point can be thought of in three simple steps. First the distances from the test point to each of the training points are calculated. From these distances, the K nearest neighbours are selected. Each of these neighbours casts a



(a) (b)

Figure 2.2: A schematic diagram showing the decision boundary between two classes in a data-set determined by a KNN model, depending on the chosen number of data points in a class, taken from Deepthi (2019). In each plot, the symbols of each data point represent the correct class in which it should be categorised. The orange triangles indicate the one class, for example spiral galaxies, while the blue squares represent the other class, for example elliptical galaxies. (a) Shows the decision boundary when K is set to five and (b) shows the decision boundary when K is set to sixty.

vote to classify the new test point into their labelled class, and the majority vote is taken as the final class label. The distance or similarity in the case of H I emission data cubes would not only be based on their location but also other features such as the voxel intensity and the velocity width, or even a weighted combination of them. The number of data points in each group, K, determines the shape of the decision boundary. An example of this can be seen in Figure 2.2. The choice of K, much like many machine learning model parameters, requires a trade-off between the bias and the variance, shown in Figure 2.3. A too low of a value for K results in a high variance, or over-fitting. Therefore, the larger the value of K, the more complex the model and the lower the variance but the higher the bias and the potential for under-fitting of the model. KNN has the advantage of a relatively good accuracy with low computational cost. However, in previous segmentation tasks KNN was found to be biased in the type of galaxies it detected from a sample consisting of an equal amount of each galaxy type (Cheng et al., 2020).

2.2.3 Decision Trees

A Decision Tree (Breiman et al., 1984) is a classifying method that builds a tree based on the features of a data-set. The root node of the tree is considered the most important feature while the leaf nodes are the class labels. Each node is split into child nodes in a way that decreases the impurity of the classifications, usually measured using the Gini index or the entropy. A schematic diagram of an example of a simple Decision Tree can be seen in Figure 2.4. The advantage of the decision tree is its simplicity and the ease of interpreting its results.

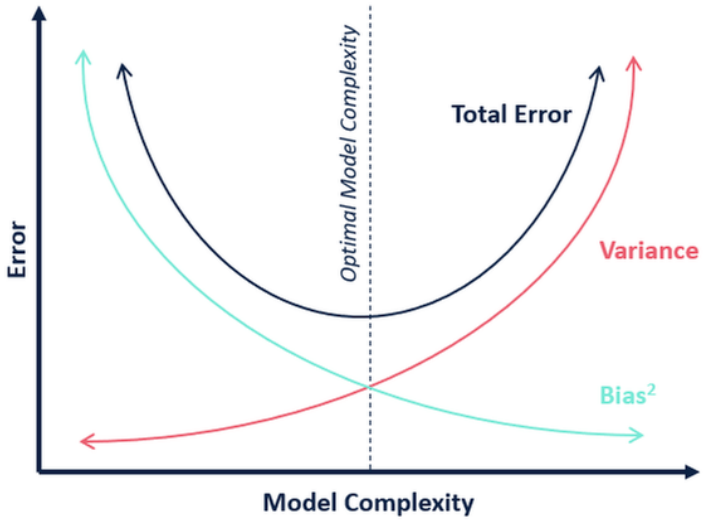


Figure 2.3: A schematic diagram showing the trade-off between the bias and variance of a model when choosing the model complexity, taken from Purva (2020). The plot shows the error as a function of model complexity with the blue line indicating the square of the bias and the red line indicating the variance. The dotted vertical line indicates the optimal model complexity, where both the bias and variance are minimised.

In addition, it is advantageous in its lack of pre-proccessing needed since it is less sensitive to missing values and outliers. The downfall of decision trees is their tendency to over-fit on the training data which has led to an ensemble approach known as Random Forests.

2.2.4 Random Forests

The Random Forest (Ho, 1995) algorithm, as its name suggests, creates a random forest of uncorrelated decision trees. To reduce variance, the decision trees are each trained on a subset of the training data and input features selected randomly with replacement, also known as bootstrapping. The bias in the learning error is reduced by merging the decision trees and taking the average result or the class label with the majority vote. A schematic diagram of a Random Forest can be seen in Figure 2.5. Random Forests are therefore advantageous in their accuracy, stability and few tuning parameters, but like SVMs, their internal process is not transparent. Random Forests have been used in astronomy for the classification of galaxy morphology, such as Beck et al. (2018) and Dubath et al. (2011). However, directly using pixels as the input of a Random Forest is not well-tested in astronomical work.

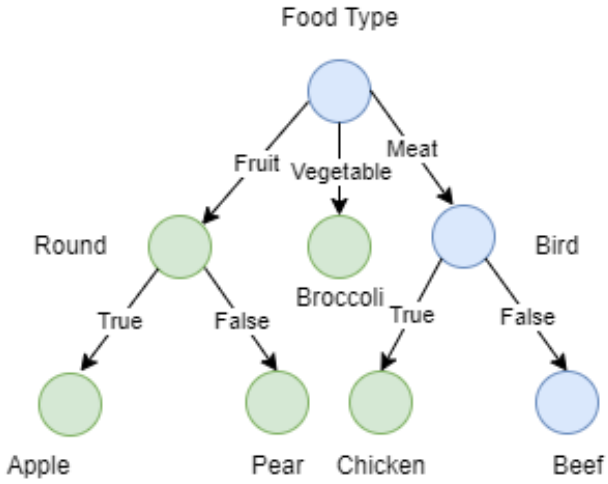


Figure 2.4: A schematic diagram of a simple Decision Tree to classify a food item. Note how the root node of the tree is the most important feature, **Food Type**, while the leaf nodes are the class labels, **Apple**, **Pair**, **Chicken** and **Beef**.

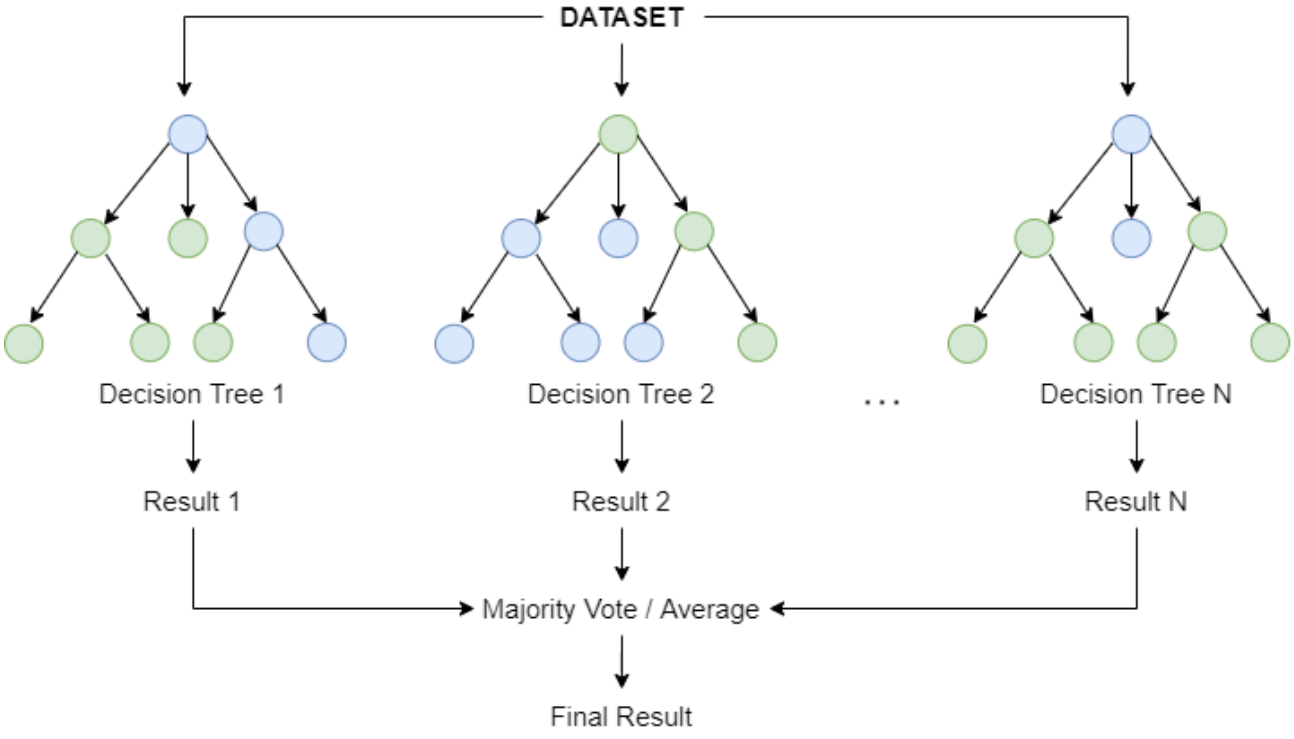


Figure 2.5: A schematic diagram of a Random Forest composed of N decision trees whose results are combined used a majority vote or average.

2.3 Deep Learning Segmentation Methods

Deep Learning is a type of machine learning which learns directly from observational data rather than its features. In recent years it has been the primary method used for object segmentation due to its high accuracy above traditional methods (Fernandez-Delgado et al., 2014). The training of features requires less prior knowledge of the data and therefore makes it most advantageous when the features are not enough to create an accurate model. The main downfall of deep learning methods is their computational expense and the need for large labelled training data-sets. However, there has already been a lot of work to overcome these two challenges (Samudre et al., 2020; Zhi et al., 2018; Bochkovskiy et al., 2020).

Artificial Neural Networks (ANN) are deep learning models that are based on the visual cortex of animal brains, which as the name suggests contains a network of neurons. Neurons receive and send signals to each other through connections known as synapses. In the same way, in ANNs an artificial neuron will receive input from the previous layer through synapses with associated weights. The sum of the signal from these synapses multiplied by their corresponding weights is used to create what is known as a filter or kernel before being passed to an activation function. The activation function then creates the output of the neuron, for example using a threshold, and passes it on as the input to the next layer of the network. A schematic diagram of this process is shown in Figure 2.6. ANNs can consist of many hidden layers, in between the input and output layers. Neurons in these hidden layers produce an output by applying an activation function to a weighted set of inputs, as explained. The most well-used method for image processing is the Convolutional Neural Network (CNN). A CNN is a type of ANN that contains one or more convolutional layers. The convolution is performed by moving a window across the receptive feeds of the inputted images and applying a filter or kernel. This results in a matrix the size of the window with a third dimension representing the number of filters when many filters are applied in a convolutional layer. In this way the convolutional layer creates an outputted feature map and uses it as the input for the next layer. An example of a performed convolution is shown in Figure 2.7.

In CNNs, the initial base layers generally alternate between convolution and max pooling layers. Max pooling applies a maximum filter to non-overlapping regions of the outputted feature maps of the previous convolutional layer. The benefit of using max pooling layers is that they reduce the size of the feature maps, also known as down sampling, and therefore reduce the computational complexity of the network. While the spatial dimensions are down sampled, the size of the receptive field increases as the network gets deeper, allowing for the filter to be applied to a larger context of the data. In addition, the dimension representing the number of filters increases in order to extract more complex features.

Down sampling therefore extracts the features which are important for the segmentation task, but in the process the information about their locations is lost. In order to revive this information, the process of up sampling is introduced, which makes use of de-convolutions to increase the spatial dimensions of the data back to its original input dimensions. While there are many techniques used for up sampling, the preferred method is known as transposed convolution or fractionally-strided convolution (Dumoulin and Visin, 2016). Transposed convolution essen-

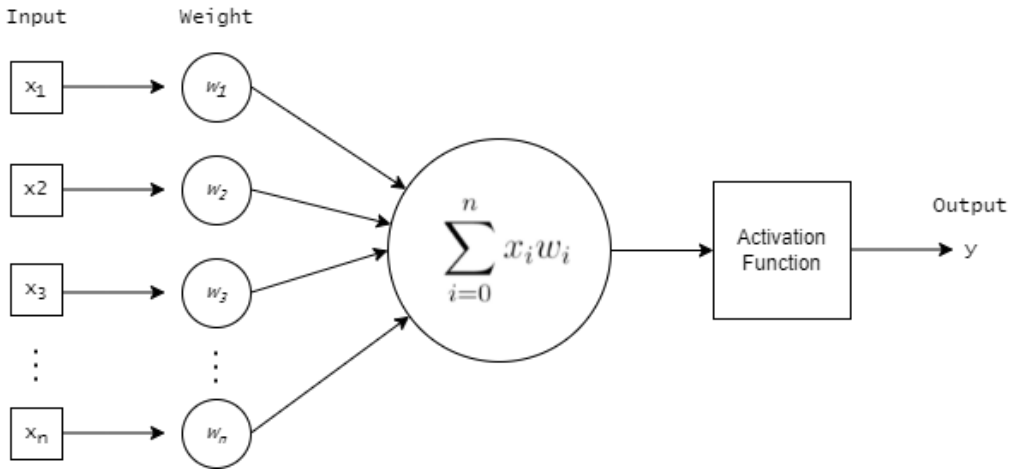


Figure 2.6: A schematic diagram of a neuron in an Artificial Neural Network. The input is denoted as x_i with where $i = 0, ..., n$ and n is the number of features. The weight of the synapses are denoted by w_i.

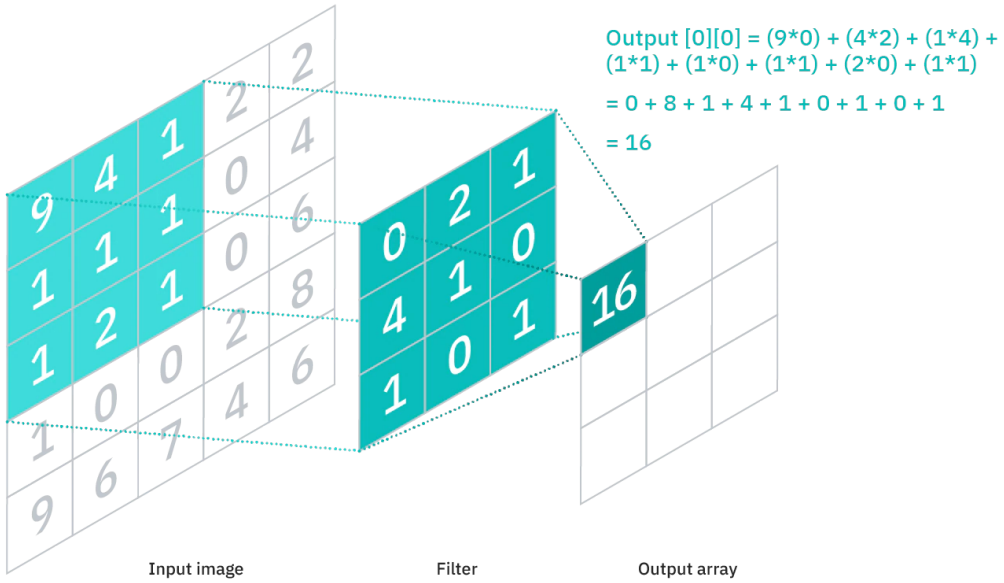


Figure 2.7: An example of a convolution in a Convolutional Neural Network taken from IBM Cloud Education (2020). Here the input image begins with 5x5 pixels and a 3x3 feature map is produced by applying a filter made up of a 3x3 matrix.

tially does the reverse of what is shown in Figure 2.7 by applying a filter to a fraction of the input that is smaller than the filter itself. In this way, an output larger than the input is produced. The process of down sampling followed by up sampling in order to perform image segmentation is used to form what is known as a Fully Convolutional Network (FCN) (Long et al., 2015).

In the first layers of a CNN the basic features are extracted, for example lines and dots, while the later layers extract more complicated features such as galaxies. In this way, the network learns more and more with each layer and its structure becomes hierarchical. CNNs usually consist of many so called epochs, each of which involve a full cycle of the CNN with the entire data-set. In each epoch, a cost or loss function is used measure the error in the predictions produced by the network by comparing them to the targets. With the weight initialized randomly, the highest loss score is produced after the first epoch. However, with each epoch the weights are updated to minimize the loss score and the CNN improves until it reaches a point that it starts to over-fit.

An advantage of CNNs is their ability to be re-trained on custom data-sets, which allows for more flexibility while shallow machine learning methods tend to be more domain-specific. It is not only widely used in the field of computer vision but has recently become a popular method in astronomical source finding in radio astronomy (Gheller et al., 2018; Lukic et al., 2019; Aniyan and Thorat, 2017), and was found to out-perform other machine learning methods when used for galaxy classification (Cheng et al., 2020; Alhassan et al., 2018).

2.3.1 Volumetric Segmentation with Deep Learning

Three dimensional data can be represented in a variety of different ways and can be categorized into Euclidean or non-Euclidean in structure, each of which require a different approach of deep learning (Ahmed et al., 2018). Euclidean data is structured as a grid, with the preservation of its properties, for example, having a common coordinate system, while non-Euclidean data does not. The 3D H I data cubes can therefore be categorised as Euclidean structured, volumetric, voxel-based data. Unfortunately, while this has the advantage of simplicity, voxel-based data inefficiently represents even non-occupied parts of the image, resulting in large memory and storage requirements.

While the use of CNNs for 2D images is well explored, the application of CNNs on three dimensional data cubes is still very new. The main challenge is that CNNs require a fixed-dimensional grid input. Many of the existing methods use two dimensional data to train the network due to the lack of labelled 3D data available. Many different slicing techniques exist, for example Çiçek et al. (2016) and Yang et al. (2021), but essentially these methods slice the cube into multiple 2D slices and then fuse them back together to create a masked cube. The identification of sources in each slice is done based off the top-ranking medical imaging CNN architecture called U-Net (Ronneberger et al., 2015).

As the name implies, the architecture emulates a U-shape with two paths. The first path, also known as the compression path or the encoder, down samples the data in different stages of convolutional and max pooling layers. This is followed by a decompression path or the decoder, which uses transposed convolutions to up sample the data to its original input dimensions. In this way it is built upon the architecture of an FCN, with the compression path

reducing spatial dimensions and extracting features while the decompression path increases the spatial dimensions in order to retrieve the location information. In order to link the extracted feature information to the location information, each stage links the two paths by concatenating the output of the transposed convolution layers from the decompression path with the feature maps from the compression path. Once this concatenation is performed, two consecutive regular convolutions are applied in order to assemble the output.

Applying 2D CNNs to slices of the data has the advantage of being able to make use of previously trained weights from works such as Gheller et al. (2018) and Lukic et al. (2019), which find sources in 2D radio data. However, while this saves computational costs, it is optimised for sources whose distinctive shapes are preserved in local projections and it is therefore very important how the data is sliced. A alternative approach is using an architecture that takes an entire volume as its input as opposed to slices by using volumetric convolutions. V-Net (Milletari et al., 2016) is an architecture that does just that, built following the architecture of U-Net.

While there is a variety of source-finding techniques that exist, the traditional methods require extensive prior knowledge and still have been found to fail at differentiating many faint H I sources from surrounding noise. Although both shallow machine learning methods and even deep learning have begun to be explored in astronomical source finding, there is very little work done that uses these techniques on three-dimensional H I emission data. In addition, the three-dimensional CNN architectures that do exist have not yet been tested for finding astronomical sources. In this project a three-dimensional CNN architecture, namely V-Net, was used to locate and mask H I sources in 3D data cubes and compared to the traditional tools, SoFiA and MTO. While the segmentation abilities of classic machine learning methods were not tested directly on the H I emission cubes, they were used as a post-processing step to filter out false positives from the resulting catalogs.

Chapter 3

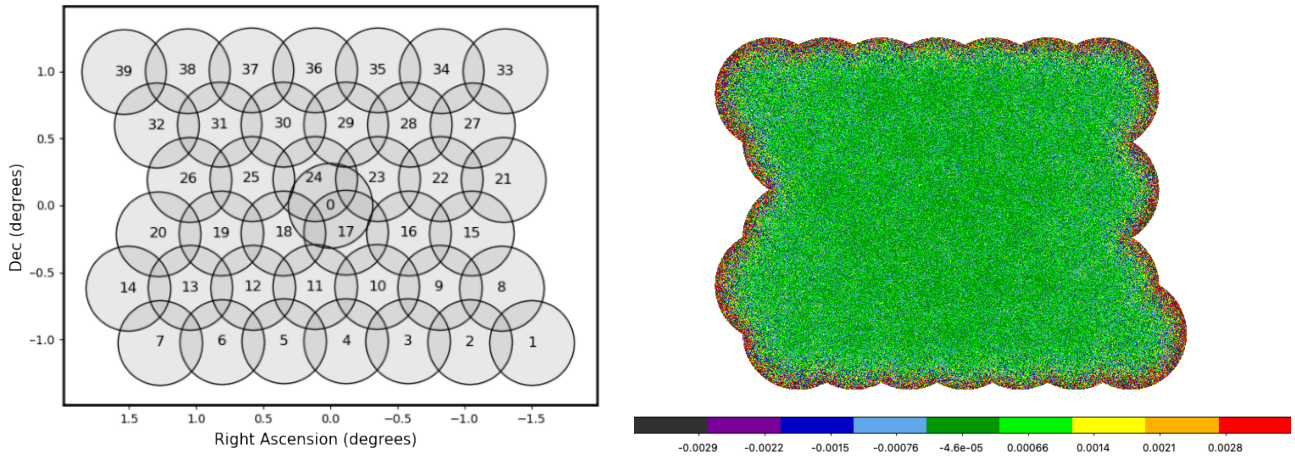
Materials

Supervised machine learning methods, especially CNNs, require a large sample of labelled training data, which is not easily available for radio images. Previous approaches, such as Galaxy Zoo (Lintott et al., 2008), have used crowd sourcing to create manually labelled data via visual inspection. However, even if manually labelled data was available, the ground truth is not known and some labels could be missed or incorrect, due to the difficulty of interpreting the 3D data cubes. Additionally, many of the sources of interest, such as ultra-diffuse galaxies, are rare and therefore very difficult to train to detect (Haigh et al., 2021). To tackle this problem simulated cubes of H I emission containing galaxies with labelled masks were used for this project. This chapter explains the observed data used and the production of the mock H I emission cubes used in the experiments.

3.1 Observed Data

To prevent the need to simulate noise, existing observed data cubes were taken from the H I medium-deep imaging survey of the APERture Tile in Focus (Apertif) (Apertif Science Team, 2016). Completed in 2019, Apertif is an upgrade of the Westerbork Synthesis Radio Telescope (WSRT) (Hogbom and Brouw, 1974), an aperture synthesis interferometer in the Netherlands. The upgrade of 12 out of the 14 dishes at the WSRT includes an array of 121 antennas in a single focal plane, known as a phased-array feed (PAF) (van Cappellen et al., 2021). The signal from these antennas is correlated across the dishes, resulting in 66 baselines and therefore 66 correlation measurements. From these measurements the visibilities in the U-V plane are created, containing both an amplitude and a phase. This provides 66 Fourier components, measured every 30 seconds, which can be transformed into the image plane of right ascension and declination using Fast Fourier Transforms (FFTs) (Brigham and Morrow, 1967). A consequence of this process is that the noise is correlated across the pixels of the image. The observations are then combined into a hexagonal mosaic of 39 beams, with a single beam placed at the location of the optical axis of the antenna for testing purposes. This makes the PAF 39 times faster than observing with dishes that have a single receiver and therefore enable surveys to faster observe larger areas of the sky.

Each of the beams cover $36'$ of the sky, resulting in the final mosaic covering about 2.4° all



(a) (b)

Figure 3.1: (a) Shows the layout of the Apertif compound beams mosaicked into a hexagonal shape, taken from van Cappellen et al. (2021). (b) Shows an example of a single channel of an Apertif H I emission cube, prior to noise normalisation. The colour scale represents the noise in the cube, which is non-uniform and increases towards the edges of the cube. The plot has been z-scaled for better visualisation using an `astropy` (Astropy Collaboration, 2018) module, `visualization`. A detailed explanation of the IRAF (Tody, 1986) z-scaling, which the astropy method follows, can be found in iraf.net (n.d.).

together. At the rest frequency of 1420 MHz, the angular resolution is 15″ in the East-West direction and 25″ in the North-South direction. The observations were done for 652 frequency channels, from 1279 MHz to 1425 MHz, which were then split into seven spectral windows of 23.84 Hz, for easier computation. These windows were split in such a way that they overlap to account for the frequency width of galaxies. The diffraction of the primary beam attenuates the signal and adds noise to the observations, particularly at the edge of the beams. In addition, not every beam is equally sensitive and therefore the observations contain many variations in the noise. In order to correct for these noise variations and allow for better source finding, the noise was normalised by dividing the H I emission cube by the spectral root mean square (RMS) at each pixel. The missing values in the corners of the H I emission cubes were replaced with normalised random noise following a Gaussian distribution with a full-width-half-maximum (FWHM) equal to the spatial separation of the pixels. This was done to prevent any errors from the source finders in case they could not take missing values in their input. Figure 3.1 shows both the layout of the beams that form the mosaic and an example of a single channel taken from one of the data cubes prior to its noise normalisation.

The observed areas of the sky in the imaging surveys performed with Apertif were chosen to include regions that have publicly available spectroscopic and optical observations. Of the two major imaging surveys, the data for this project was taken from the medium-deep survey (Apertif Science Team, 2016), which covers an area including the Perseus–Pisces Supercluster. This area consisted of nine pointings, each observed 10 times for 11.5 hours to reduce the noise by a factor of $\sqrt{10}$ and increase the sensitivity when combining them. Due to the partial com-

pletion of the observations at the start of this project, two out of the nine pointings were used, one for training purposes and one for testing, each containing seven spectral windows. Pointing 1 covers $1^h45^m29.82^s$ - $2^h5^m7.69^s$ in right ascension and $32°25'16.19''$ - $35°25'08.58''$ in declination. Pointing 2 ranges from $1^h45^m10.92^s$ - $2^h5^m26.57^s$ in right ascension and $34°51'04.49''$ - $37°50'56.31''$ in declination.

Each of the spectral windows from each pointing makes up a data cube with $652 \times 1800 \times 2400$ voxels in dimension, representing the frequency, declination and right ascension respectively. The spatial separation of the pixels is $6''$ and the spectral separation of the channels is 36.621 kHz. The properties of the seven spectral windows for the two pointings can be found in Table 3.1. The RMS of the noise prior to normalisation shows how the noise correlates with the frequency range, getting higher at more distant spectral windows. The only exception to this is the nearest spectral windows of both pointings, which is probably due to the interference of the H I emitted from our own Milky Way.

Table 3.1: The properties of the seven spectral windows of the two pointings chosen from the Apertif medium-deep survey.

Frequency Range (MHz)	Redshift Range	Distance Range (Mpc)	Recession Velocity Range (km/s)	RMS noise (mJy/Beam)
Pointing 1				
1401.25 - 1425.09	0.013669 - 0	59 - 0	4098 - 0	1.53
1381.04 - 1404.88	0.028506 - 0.011053	122 - 47	8546 - 3314	1.46
1360.82 - 1384.66	0.043785 - 0.025813	188 - 111	13126 - 7739	1.49
1340.61 - 1364.45	0.059524 - 0.041011	255 - 176	17845 - 12295	1.51
1320.39 - 1344.23	0.075745 - 0.056666	324 - 243	22708 - 16988	1.53
1300.18 - 1324.02	0.09247 - 0.072799	396 - 312	27722 - 21825	1.91
1279.96 - 1303.80	0.109724 - 0.089432	470 - 383	32894 -26811	2.43
Pointing 2				
1401.18 - 1425.02	0.013722 - 0	59 - 0	4114 - 0	1.74
1380.96 - 1404.80	0.028561 - 0.011106	122 - 48	8562 - 3329	1.69
1360.75 - 1384.59	0.043841 - 0.025868	188 - 111	13143 - 7755	1.71
1340.53 - 1364.38	0.059582 - 0.041067	255 - 176	17862 - 12312	1.72
1320.32 - 1344.16	0.075804 - 0.056724	325 - 243	22726 - 17005	1.73
1300.10 - 1323.95	0.092532 - 0.072858	396 - 312	27740 - 21842	1.77
1279.89 - 1303.73	0.109787 - 0.089493	470 - 383	32913 - 26829	2.17

3.2 Synthetic Data

The 3000 mock galaxies created by Gogate et al. (2020) were used as a parent catalog for the insertion into the H I emission cubes similar to that shown in Figure 3.1. These galaxies were created by assuming a ΛCDM cosmology with $H_0 = 70\,\mathrm{km\,s^{-1}\,Mpc^{-1}}$ and $\Omega_m = 0.3$ using GALMOD, which is a part of the Groningen Image Processing System (GIPSY) (van der Hulst

et al., 1992). In order to create the mock galaxies, GALMOD requires a rotation curve and a radial H I density distribution of each galaxy. The Universal Rotation Curve (URC) (Persic et al., n.d.) was used to calculate the rotational velocities of the mock sources and their radial H I surface density distributions. The parameters of the URC are prescribed by the B-band luminosity and the B-band radius, both of which are found through standard scaling relations.

The first step to calculating these parameters is the calculation of the H I mass, following the H I mass function (Martin et al., 2010), which can be described by the Schechter function (Schechter, 1976). The H I mass was randomly selected between $10^{8.5}\,M_\odot$ and $10^{10.5}\,M_\odot$, with the upper-bound cut-off excluding the more massive galaxies of the H I mass function. The well-known correlation between the H I mass and the H I diameter, D_{HI}, was used to calculated the H I diameter with parameters taken from other works (Broeils and Rhee, 1997; Verheijen, 2001; Swaters et al., 2002; Noordermeer et al., 2005; Martinsson et al., 2016). In consistency with Wang et al. (2016), the H I diameter was then calculated as follows:

$$\log(D_{HI}) = 0.54\log(M_{HI}) - 3.646, \tag{3.1}$$

where the disc size, D_{HI}, is measured in kpc and the H I mass, M_{HI}, is measured in solar masses.

Once the H I diameter is established, the B-band diameter at the 25 mag arcsec^{-2} isophotal level of the Ursa Major sample (Verheijen and Sancisi, 2001), D_{25}, was taken within $0.4D_{HI} < D_{25} < D_{HI}$. This can then be used to calculate the diameter enclosing 80% of the total blue light, $D_{80\%}$, as follows:

$$D_{80\%} = 0.642D_{25} + 10.83. \tag{3.2}$$

To give plausible luminosities to the mock galaxies, a box was taken around scattered measurements of the k-band absolute magnitudes as a function of the H I mass from Tully et al. (1996), Noordermeer et al. (2005), Swaters et al. (2002), Spekkens and Giovanelli (2006), Cappellari et al. (2011) and Ponomareva et al. (2016). Once the k-band absolute magnitudes were randomly selected from within this box, a similar box was created around the various scattered measurements (Martinsson et al., 2016; Tully et al., 1996; Noordermeer et al., 2005) of the B-K colour-magnitude diagram and used to select the B-band absolute magnitude, M_B.The B-band luminosity was then calculated as follows:

$$L_B = 10^{(\mathrm{Mag}_{\odot(B)}(B) - M_B)/2.5}, \tag{3.3}$$

where $\mathrm{Mag}_{\odot(B)} = 5.48$ is the B-band absolute magnitude of the Sun. The calculated B-band luminosity and B-band radius of each galaxy were inputted into the Universal Rotation Curve (URC) (Persic et al., n.d.), which was used to calculate the rotational velocities of the mock galaxies. The radial H I surface density distributions were calculated by slightly modifying the radial density distribution (Martinsson et al., 2016) to allow for changes in the inner part of their profiles. The cosine of the inclination and the position angles were chosen at random from uniform distributions between 0 to 1 and 0° to 360° respectively. The mock galaxy cubes were created with an angular resolutions of 15″ in the East-West direction and 25″ in the North-South direction.

Each of the mock galaxy cubes were created at 50 Mpc from the observer, with 400 channels and 400×400 pixels. The pixel size of these mock galaxy cubes were created to be 5″ with a channel width of 24.208 kHz, equivalent to a velocity width of 5.2 $\mathrm{km\,s^{-1}}$. An example of a mock galaxy cube and its intensity or global H I profile can be seen in Figure 3.2, where it is compared to a real H I galaxy, a source-less piece of noise from one of the H I emission cubes taken from the Apertif medium-deep survey and the result of adding the mock galaxy to the noise. The global H I profiles of both the mock and real galaxy show the double-horned profile, however, the perfect symmetry of this profile is only apparent in the mock galaxy. This demonstrates the limitations of the simulated galaxies due to their consistent symmetry, which is not necessarily the case for real sources.

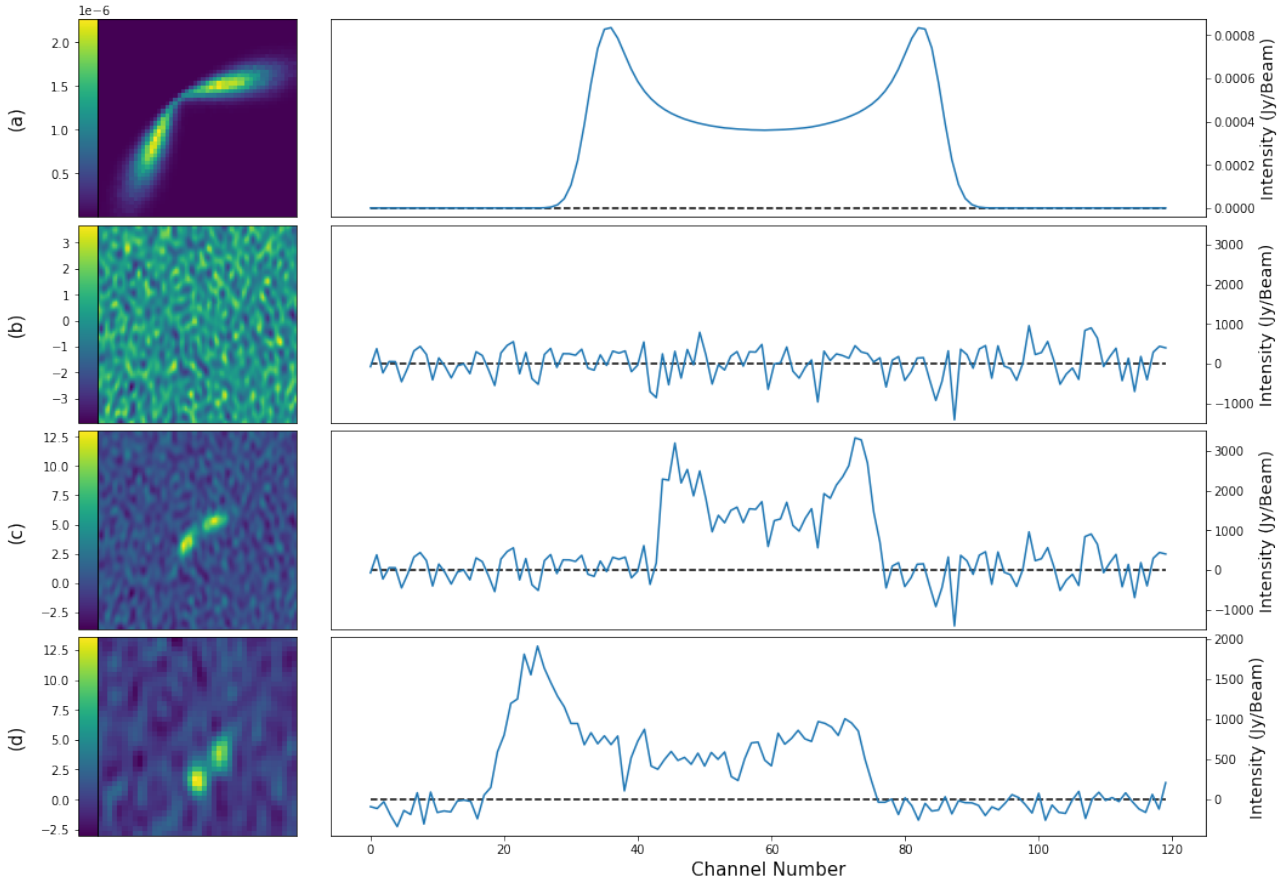


Figure 3.2: The left panel shows different zoomed in single channels of H I data with the colour bars representing the intensity of the simulated emission, while the right panel shows the corresponding global H I profile in Jy/beam with the black dashed line at zero Jy/beam. (a) Shows a mock H I galaxy without noise, valid at 50 Mpc and a velocity width of 5.2 km s^{-1}. (b) Shows a source-less part of the noise taken from the Apertif medium-deep survey at a redshift of 0.0233 and a distance of 99.77 Mpc. (c) Shows the mock galaxy inserted into the noise at this redshift of 0.0233, following the steps in Section 3.3. (d) Shows a real source among the noise in the same spectral window at a redshift of 0.0153 and a distance of 65 Mpc.

3.3 Inserting the Mock Galaxies

From the 3000 mock galaxies, 190 to 500 were randomly inserted into each of the spectral windows to prevent the training of any models from expecting a specific number of sources. The number of galaxies that were inserted into each cube can be found in Table 3.2, resulting in 4286 mock galaxies inserted in the training and test set in total. The insertion of the sources into the observed H I emission cubes was done for simplification purposes to avoid the complexity of simulating the noise.

As mentioned, the mock galaxies were scaled in size and flux to a distance of 50 Mpc with pixel sizes of $5'' \times 5''$, which corresponds to a physical width in kpc, $\mathfrak{R}_0$, of (Meyer et al., 2017):

$$\mathfrak{R}_0 = d \tan(p), \tag{3.4}$$
$$= 1.212\,\text{kpc}, \tag{3.5}$$

where d is the distance in kpc and p is the pixel size in degrees. The corresponding redshift at the distance of 50 Mpc, z, was found following:

$$z = \frac{\text{v}}{c}, \tag{3.6}$$
$$= \frac{H_0 d}{c}, \tag{3.7}$$
$$= 0.011675, \tag{3.8}$$

where $\text{H}_0 = 70\,\text{km}\,\text{s}^{-1}\,\text{Mpc}^{-1}$ is the Hubble constant, c is the speed of light and v is the corresponding recession velocity. Since H I emission has a rest frequency, ν_{rest}, of 1420.405752 MHz, at this distance of 50 Mpc, it is observed at a frequency of:

$$\nu_{\text{obs}} = \frac{\nu_{\text{rest}}}{1+z}, \tag{3.9}$$
$$= 1404.0143\,\text{MHz}. \tag{3.10}$$

At this observed frequency, since the channel widths of the mock galaxy cubes, $\Delta\nu$, are 24.208 kHz, the corresponding rest frame velocity width, dV, is calculated to be:

$$dV = c\frac{\Delta\nu}{\nu_{\text{obs}}}, \tag{3.11}$$
$$= 5.169\,\text{km}\,\text{s}^{-1}. \tag{3.12}$$

Since the angular resolution of the pre-existing mock galaxy cubes can only be reduced, not improved, the frequency in Equation 3.9 was taken as the maximum frequency at which the mock galaxies could be inserted into the Apertif H I emission cubes. Using this as an upper bound, each mock galaxy was inserted into the Apertif H I emission cubes by choosing a random channel from the spectral window for its center. The frequency of the chosen channel, ν_{obs}, was used to calculate the redshift, z, recession velocity, v_{obs}, and distance d_{obs} following Equations 3.6, 3.7 and 3.9. In order for the mock galaxies to be inserted into this chosen target redshift, z, the Apertif beam size was scaled by a factor of $(1+z)$. This resulted in an angular resolution

of $(1+z_{\rm obs})15''$ in the East-West direction and $(1+z_{\rm obs})25''$ in the North-South direction, which corresponded to a spatial resolution in Mpc as follows:

$$\mathfrak{R}_{EW} = d_{\rm obs}\arctan((1+z_{\rm obs})15''), \tag{3.13}$$

$$\mathfrak{R}_{NS} = d_{\rm obs}\arctan((1+z_{\rm obs})25''), \tag{3.14}$$

where $\mathfrak{R}_{EW}$ and $\mathfrak{R}_{NS}$ are the spatial resolutions in the East-West and North-South directions respectively. Since the voxels in the mock galaxy cubes were independent from each other, they were convolved from their original spatial resolution in Equation 3.4 to the spatial resolution of the chosen channel in Equation 3.15 using a Gaussian smoothing kernel with the following full-width-half-maximum (FWHM) in units of pixels:

$$\mathrm{FWHM}_{EW} = \frac{\mathfrak{R}_{EW}}{\mathfrak{R}_0}, \tag{3.15}$$

$$\mathrm{FWHM}_{NS} = \frac{\mathfrak{R}_{NS}}{\mathfrak{R}_0}, \tag{3.16}$$

where FWHM_{EW} and FWHM_{NS} are the full-width-half-maxima in the East-West and North-South directions respectively. The chosen distance at which each galaxy was inserted corresponded to a pixel width calculated as $r_d = d\tan(5''/3600)$. Therefore, once smoothed, the pixel sizes of the mock galaxies were adjusted from their original $5''$ to a new pixel size $5''(\mathfrak{R}_d/\mathfrak{R}_0)$. The mock galaxies were also re-sampled in their frequency axis. Following Equation 3.11, the rest frame velocity width of the Apertif cube at the chosen channel was calculated using the observed frequency of H I emission at that redshift and the channel width of 36.621 kHz. Therefore, for the insertion of the mock galaxies, the width of their channels were increased by a factor of dV_f/dV_i, where dV_i is the initial rest frame velocity width in Equation 3.11 and dV_f is the same calculation but at the chosen channel.

The flux of each mock galaxy was then re-scaled to account for the fact that the sources would be further away and therefore fainter. Since the flux scales with inverse proportionality to the distance squared, the mock galaxy cubes were multiplied by a factor of $(50\,\mathrm{Mpc}/d)^2$, where d is the distance at the new chosen channel. Finally, as a sanity check, the H I mass of each galaxy was calculated and compared to its original H I mass. To calculate the H I masses, first the cube was descaled to remove the noise normalisation from the calculation, by multiplying each channel by the rms map of that spectral window. The flux for each source, S^{ν}, was then calculated as (Meyer et al., 2017):

$$\mathrm{S} = \int_{\nu} S_{\nu} d\nu \tag{3.17}$$

$$= \frac{1}{\Omega_{bm}} \int_{\nu}\int_{\Omega} I_{\nu} d\Omega\, d\nu \tag{3.18}$$

$$= \frac{1}{\Omega_{bm}} \int_{\nu} \frac{c\Delta\nu}{\nu_{\rm obs}} I_{\nu}\, d\nu, \tag{3.19}$$

where S_{ν} is the flux density and $\Omega_{bm} = \pi ab/4ln(2)$ is the area of the telescope beam, with a and b representing the angular resolutions of the telescope. Since both a and b scale with $(1+z)$, the beam solid angle scales with $(1+z)^2$. Note that since the H I cubes were presented

in Jy/beam, in order to get the flux in $\mathrm{Jy\,km\,s^{-1}}$, the values of the pixels were summed over the region of the source and multiplied by the pixel area, $\int_\Omega I_\nu d\Omega$. The H I mass, M_{HI}, was then calculated as follows:

$$\frac{M_{HI}}{M_\odot} = \frac{2.36 \times 10^5}{(1+z)^2} \frac{\mathrm{d}^2}{\mathrm{Mpc}^2} \frac{\mathrm{S}}{\mathrm{Jy\,km\,s^{-1}}}. \tag{3.20}$$

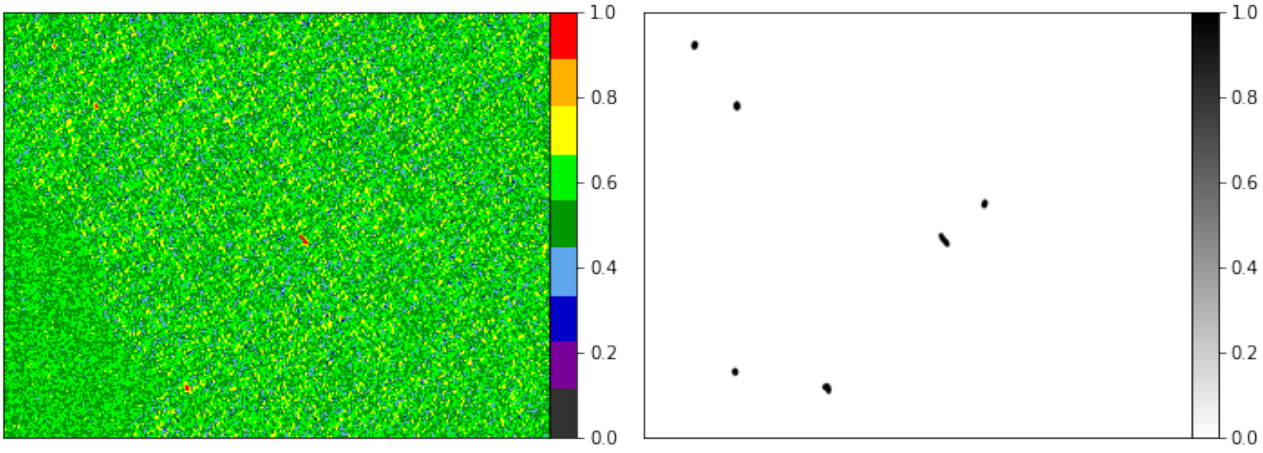
In attempt to correct for the low H I mass upper bound of $10^{10.5}\,\mathrm{M}_\odot$ for the mock galaxies and include more massive galaxies from the H I mass function, the H I masses of the galaxies was multiplied by a factor of ten. A consequence of this crude approximation is that the data-set excludes galaxies on the lower end of the H I mass function, below $10^{9.5}\,\mathrm{M}_\odot$. However, since these low H I mass galaxies are only detectable above the noise at higher frequencies, it only affects the source finders at the more nearby spectral windows. Therefore the mock sources less accurately represent the real sources in the more nearby spectral windows, but this would be the case for all source finding methods in this comparative study.

Finally, 190 to 500 smoothed, re-sampled and re-scaled mock galaxy cubes were randomly chosen and scattered in each of the noise-free cubes which corresponded to the spectral windows of the Apertif pointings. As mentioned, this was to represent a rough estimate of the number of galaxies per spectral window, without creating a bias for a set number of galaxies in any trained model. Additionally, binary masks were created for the corresponding spectral windows to be used as segmentation labels or ground truths in the training process. The number of galaxies that were inserted into each cube can be found in Table 3.2. Which shows that the number of inserted galaxies is independent of the spectral window.

Table 3.2: The number of galaxies inserted into the seven spectral windows of the two pointings chosen from the Apertif medium-deep survey.

Frequency Range (MHz)	Number of Inserted Galaxies	
	Pointing 1	Pointing 2
1401 - 1425	410	351
1381 - 1405	326	281
1360 - 1385	304	317
1340 - 1364	260	368
1320 - 1344	294	193
1300 - 1324	267	356
1280 - 1304	347	212
Total	**2208**	**2078**

The properties of the inserted galaxies, including the number of galaxies inserted into each spectral window, and the number of voxels occupied by the galaxies can be seen in Figure 3.4. Plot (b) demonstrates the randomness of the number of inserted galaxies, while Plot (d) shows the increased size of galaxies with frequency as the distance decreases. For the same reason, the spatial dimensions in Plots (h) and (i), as well as the number of channels in Plot (g) and the total flux in Plot (f) are seen to increase with frequency. Plot (a) demonstrates that a very



(a) (b)

Figure 3.3: (a) Shows a zoomed in, z-scaled single channel of an example of an Apertif cube. The cube has been noise normalised, with the corners filled with Gaussian noise and mock galaxies inserted. The colour bars represent the intensity of the simulated emission. (b) Shows the corresponding binary mask of the inserted mock galaxies.

small percent of the spectral windows actually contain sources as opposed to noise, which shows the imbalance between the two classes in this segmentation task, noise and sources.

It is important to note that should this work be extended to multiple data sources, for example from different telescopes, the need to standardise properties such as the dimensions of the cubes would be necessary for the deep learning source finding. Additionally, it is important to note the limitations of the data used, since only the inserted galaxies are labelled and not the pre-existing real galaxies in the cubes, resulting in partial labelling. In attempt to counter-act this issue and label the real galaxies, the resulting catalogs of the experiments were cross-matched with optical images from Panoramic Survey Telescope and Rapid Re-sponse System survey (Pan-STARRS) (Chambers et al., 2019), see Section 4.4. A zoomed in single channel of one of the H I cubes with the inserted mock galaxies can be seen in Figure 3.3.

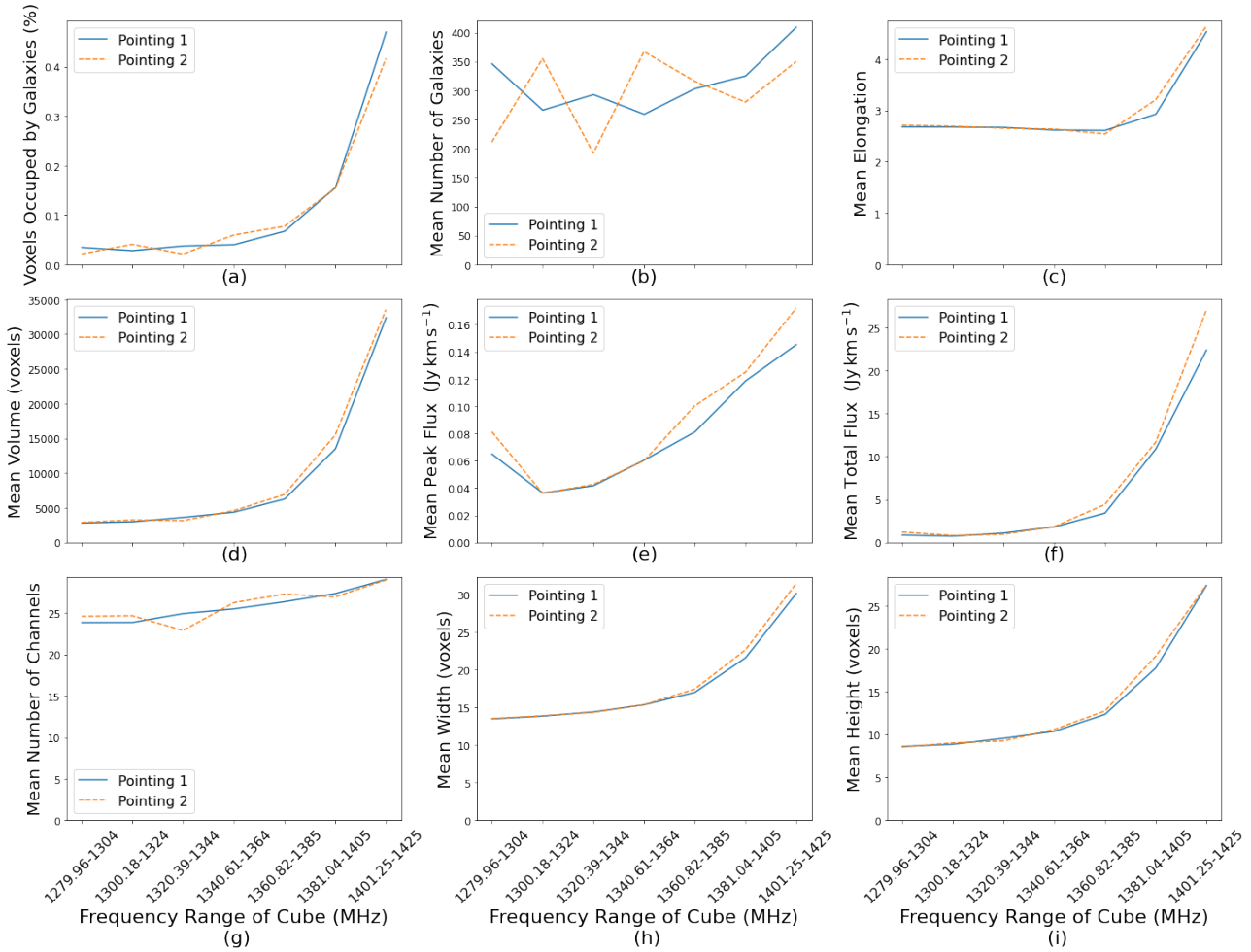


Figure 3.4: The summary plots of the properties of the mock galaxies inserted in the noise-free cubes as a function of their frequency ranges. (a) Shows the percentage of voxels occupied in the cube by the galaxies, (b) shows the number of galaxies inserted, (c) shows the mean elongation of the inserted galaxies, which is defined in Chapter 4, (d) shows the mean volume of the inserted galaxies in each cube, (e) shows the mean peak flux, (f) shows the mean total flux, (g) shows the mean number of channels occupied by the inserted galaxies in each spectral window, (h) shows the occupied width and (i) shows the occupied height. In all the plots the two pointings made up of 14 spectral windows are differentiated by a solid and dashed line.

Chapter 4

Method

In this chapter the theoretical background of the three investigated source finding methods are explained. This includes the well-known H I source finding tool, SoFiA (Serra et al., 2015), a recent approach found to be very successful in optical source finding, MTObjects (Teeninga et al., 2016), and finally a CNN architecture designed for segmentation in volumes, V-Net (Milletari et al., 2016). This is followed by a discussion of the process used to identify real sources from the resulting catalogs of each source finding method. The chapter end by explaining the theory behind the classic machine learning algorithms used for the post-processing of the created catalogs.

4.1 SoFiA

Reviewing the best of the traditional methods for segmentation, it is clear that it depends on the astronomical data itself and has therefore lead to the creation of a general source-finding tool, the flexible Source Finding Application (SoFiA) (Serra et al., 2015). SoFiA uses a combination of source finding techniques and has been designed to be independent of the type of data used. Currently, this is the most well-used pipeline for source finding in H I emission data and is therefore the benchmark for the experiments in this report. The pipeline begins by removing any variation in the noise to ensure the assumption of uniform noise. This is done by normalising the cube with a weights cube made up of the local noise. SoFiA has a number of detection algorithms to choose from, including a simple threshold technique, a characterised noise H I technique (CNHI) and a smooth and clip (S+C) technique.

The characterised noise H I (CNHI) method (Jurek, 2012) extracts individual one-dimensional spectra from the cube and uses the Kuiper test (Kuiper, 1960) to identify regions which do not consistently contain only noise. While it is able to detect a large amount of low flux sources, CNHI has been found to have the worst reliability of the detection algorithms available (Popping et al., 2012). The most common H I source finding method used in SoFiA is the smooth and clip (S+C) method (Serra et al., 2012*a*), which uses 3D kernels to iteratively smooth the cube and find sources on multiple scales, both spatially and spectrally. For each resolution, voxels are added to the mask on condition that they have a flux relative to the global RMS noise level above the chosen sigma-to-noise threshold. Since H I sources are known to have an exponential

radial surface brightness profile (Wright, 1974) and a double-horned spectral profile, see Figure 3.2, a Gaussian filter is used in the spatial domain and a boxcar filter in the spectral domain. This technique has been found to be the best source finding tool for extended sources (Popping et al., 2012) in H I data cubes and was therefore the chosen detection algorithm when using SoFiA in this project.

Once the voxels associated with sources are identified, an algorithm equivalent to the friends-of-friends algorithm Jurek (2012) is used to merge them into sources. This is done by looping over the segmented binary mask and assigning a label to unlabelled source-containing voxels as well as the neighbouring voxels within a chosen merging or linking length. In this process, sources that do not fit the given size criteria are rejected.

The reliability of the sources is then determined following Serra et al. (2012*b*). This works by assuming that a true source is positive and that the noise around it is symmetric. With this assumption it is implied that there is an equal number of source-containing voxels with a negative intensity as there are with a positive intensity. The reliability of a source can therefore be evaluated in the 3D parameter space by measuring the excess of positive or negative voxels and calculating the probability that a positive source is not just a noise peak. This metric is found as a function of the position of the source. Finally, detections which are found to have an integrated signal-to-noise (SNR) ratio below an assigned SNR limit are designated a reliability score of zero. The resulting reliability score can be thresholded to move unreliable sources.

4.2 MTObjects

4.2.1 MTO Architecture

MTObjects (Teeninga et al., 2016) is a region based source finding tool that makes use of max-trees. Max-tress are a special case of a component tree, which is classically defined as groups of voxels that are next to each other with the same intensity (Serra, 1988). The definition of a max-tree, according to Salembier et al. (1998), is a representation of an image as a tree-structure with the maxima as its leaves. The root of the max-tree consists of the entire image domain, while the leaves represent the voxels with the local maxima values. The nodes of the max-tree represent connected components of threshold sets of the image, where it is the grey scale of the image that is thresholded. Therefore there can never be more nodes than there are voxels in the image. Each connected component of the max-tree can be assigned attributes such as the volume and flux density, which allows for filtering based on a threshold for these attributes.

A max-tree is much more compact than a classical component tree, saving memory by not storing the elements of their children and consisting of only those component tree nodes that have at least one voxel at the appropriate threshold level. The advantage of max-trees is that it can filter the cubes without distorting any edge information because it operates on the connected components of an image rather than individual voxels (Salembier et al., 1998). The version of MTObjects implemented by Arnoldus (2015) was used for this project, as it was designed for 3D radio cubes. The pipeline consists of four steps that can be seen in the flowchart in Figure 4.1.

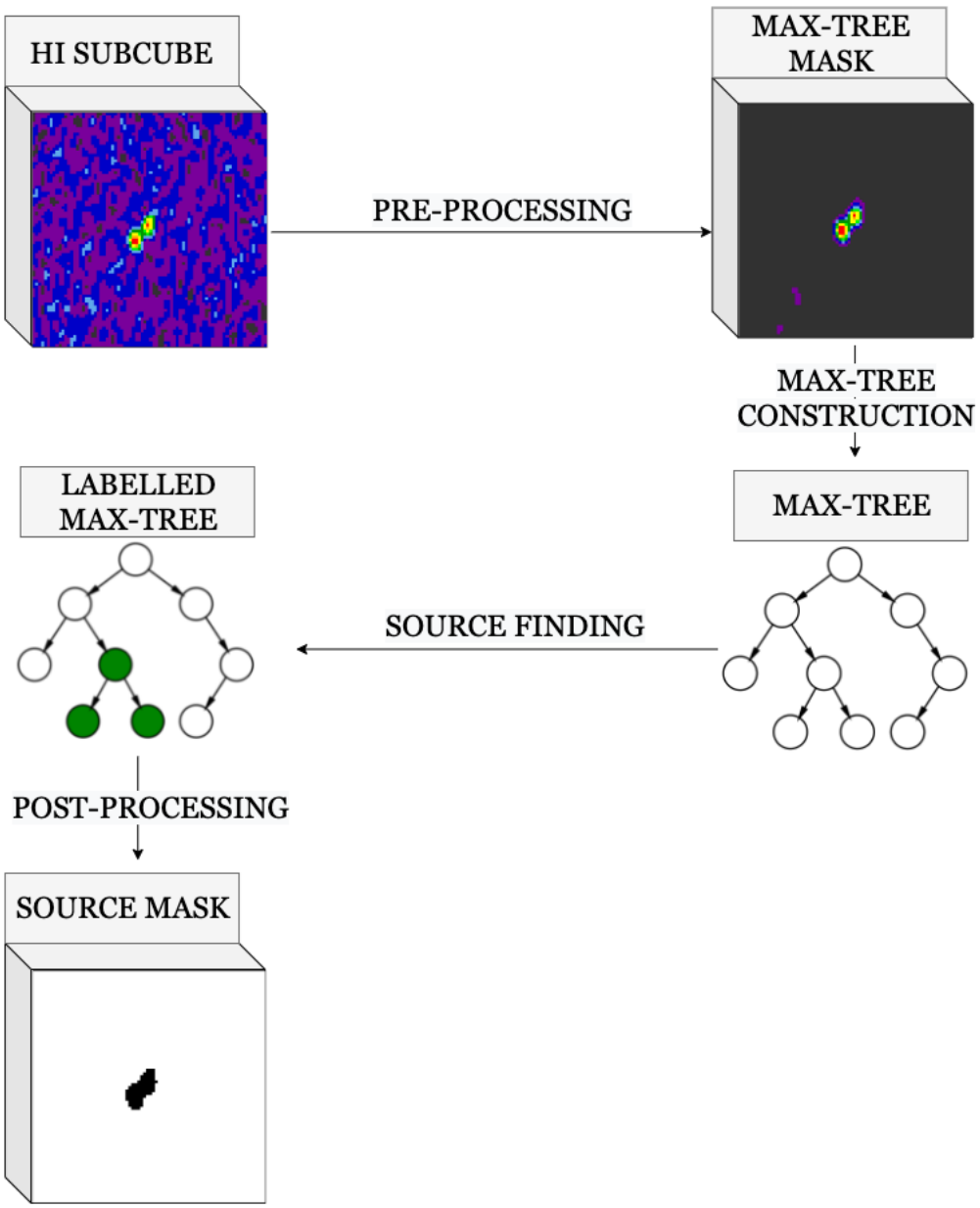


Figure 4.1: A schematic diagram of the source finding pipeline of MTObjects. An H I emission subcube, containing a source, is transformed into a max-tree mask with the use of both Gaussian and adaptive smoothing. A max-tree is then built from the max-tree mask and the source nodes are labelled using the attributes of the smoothed cube. Finally, post-processing is used to remove false detections.

The first step of the MTO pipeline is the pre-processing of the cube to create a max-tree mask. First the cube is smoothed with a Gaussian filter to stabilise the gradient approximation. This is followed by the application of spatial adaptive smoothing to ensure the preservation of the edges of sources. The adaptive smoothing technique used is known as the Perona–Malik diffusion filter (Perona and Malik, 1990). The filter models the smoothing as an anisotropic diffusion process, building a space-scale, $I(x, y, t)$ for an image, I. Note that the number of channels were not smoothed in the application of this filter on the H I emission cubes. The diffusion follows the second-order partial differential equation in time t as:

$$\frac{\partial I(x, y, t)}{\partial t} = c(x, y, t)\Delta I + \nabla c \cdot \nabla I, \tag{4.1}$$

where c is the conduction coefficient, defined as:

$$c(x, y, t) = \frac{1}{1 + \frac{|\nabla I(x,y,z,t)|^2}{K^2}}, \tag{4.2}$$

and K is a constant. Since time has no physical meaning in an image, it is taken to be a constant larger than zero which represents the degree of smoothing.

Both smoothing steps are used to increase the signal-to-noise ratio and are followed by a background estimation and to create the max-tree mask. The background is approximated to be a constant value calculated from the mean of the regions that contain no sources, also known as flat tiles. The tile size is chosen by first finding the largest flat tile in the image, starting with a set tile size and either increasing the tile size as much as it can if there is a tile this size or decreasing the tile size until a flat tile is found. The rest of the flat tiles are then found using two statistical tests. First the tile is tested for normality, rejecting those that do not follow a normal distribution according to the D'Agostino-Pearson K2-statistic (D'Agostino et al., 1990). This is followed by a t-test to test if the means of each half of the tile are equal, assuming they have equal variance.

Once all the flat tiles are found, the mean of the tiles is used to approximate the background, taking the overall mean value if no flat tiles were found. The background is then subtracted from the smoothed image and the resulting negative values are set to zero to ensure the lack of negative-total-flux nodes in the max-tree. Since this approach to background estimation was originally created for optical data, it does not account for the fact that the mean background of the noise in radio data should be zero, as it is sampled in the Fourier domain. This process of background estimation is therefore not necessarily optimal for working with H I emission data. An example of a cube before and after the pre-processing step, including the smoothing and background subtraction, can be seen in Figure 4.1.

Due to the limited classical definition of connectivity, Ouzounis and Wilkinson (2007) presented a new approach known as mask-based connectivity which instead relies on the pre-processed connectivity mask. However, Arnoldus (2015) adapts this definition of connectivity further still by storing the full original volume to calculate component attributes or properties at a later stage in the max-tree data structure. This allows for the enhancement of its segmentation by

using the pre-processed image while maintaining the original image for the statistical model.

Once the cube has been smoothed and background subtracted the next step of the pipeline is the construction of the max-tree. During the construction of the max-tree, the node attributes are also calculated to inform the removal of false detections at a later stage. However, with the sources being such a small percentage of the data cubes, there is of course a bias towards the noise that should be considered. The most commonly used attributes in astronomical data cubes include the volume, the total flux, and the peak flux (Arnoldus, 2015). However, there are also astrophysically motivated attributes for 3D sources to consider, including the spatial elongation. The elongation is defined by Westenberg et al. (2007) as $E = |\lambda_1|/|\lambda_2|$, where $|\lambda_1| \geq |\lambda_2|$ are the eigenvalues of the moment of inertia tensor of the set of voxels in a source.

Once the max-tree is constructed MTO finds and labels the nodes that belong to a source, this is the object detection step in Figure 4.1. The object detection filters the max-tree and chooses the nodes based on their attributes using statistical tests. Since the simple χ^2 would not suffice in modelling the correlated noise of the H I cubes, the statistical tests have been modified for radio data (Teeninga et al., 2013). The attribute used to detect sources was chosen to be the flux density as its is less sensitive to outliers (Arnoldus, 2015). The flux density is defined as follows:

$$F_{\text{density}}(P) = \frac{F_{\text{total}}(P)}{V(P)}, \tag{4.3}$$

$$= \frac{1}{V(P)} \sum_{x \epsilon P} f(x), \tag{4.4}$$

where P is a peak component of the max-tree mask, $V(P)$ is its volume, and f is the original volume. A node or part of the volume is included in a source mask if it has a greater value of this attribute than other children of its parent node or if it has no significant parent.

The implementation of the max-trees in the version of MTO used for this project involves a parallel algorithm to construct the max-tree (Moschini et al., 2018). This approach has both a top-down merging phase and a bottom-up flooding phase. The top-down phase merges the sorted single nodes into sub-trees to create a whole tree. This is followed by the bottom-up phase, which instead begins at the root node and traverses along higher intensity connected components to construct the full tree. This allows for a vast improvement in the performance of the method as it refines every level of the cube in its own thread, creating a multi-threaded approach.

4.3 Convolutional Neural Networks

As mentioned, the most well-used deep learning method for image processing is the Convolutional Neural Network (CNN). While CNNs can be designed from scratch, there are many already existing architectures that have been optimized for segmentation tasks. These architectures can be recycled for a variety of segmentation tasks and data-sets and can either be re-trained from scratch or the weights from previous tasks can be selectively updated.

4.3.1 Transfer Learning

The disadvantage of using CNNs for segmentation tasks on such large data-sets, such as the H I emission cubes, is the computational cost and time needed to train a model from scratch (Torrey and Shavlik, 2009). It is therefore common to use at least parts of a model that have already been pre-trained with another data-set. This is known as transfer learning because the knowledge of a previous task is transferred to a new task. Despite the lack of CNNs trained for the segmentation of H I emission galaxies, even the base layers of models designer for medical images could be used since they only learn low-level features, such as edge detectors, which are more generic and not specific to the data-set.

There are a few approaches to transfer learning, each of which use the pre-trained model to varying degrees (Pan and Yang, 2010). The first approach is known as weight initialisation. Weight initialisation uses some of the early layers of the pre-trained network and then integrates them into a new model, training both models together. The second approach is known as feature extraction. Feature extraction feeds the base layers of the pre-trained network into a new dense layer which has been trained from scratch on the data of the new task. The third approach is a much more involved technique known as fine-tuning. Fine-tuning involves freezing the weights from some of the lower layers of the network and selectively retraining some of the others, rather than just replacing the final layer of the network. The fourth approach, and the simplest of them all, directly uses the pre-trained model to segment the new data.

Despite the many benefits of transfer learning, it requires the availability of the pre-trained weights. Since 3D segmentation is a relatively newly explored task, the existing weights of 3D trained models are very hard to come by. In addition, the majority of the existing models have been trained on medical data-sets which are not licensed to be used for tasks outside the purpose of their creation. However, in this project transfer learning is used after training the weights on only mock galaxies to train further with identified real sources in the H I emission cubes.

4.3.2 The Learning Architecture Design of V-Net

The CNN architecture chosen for this project was the novel V-Net (Milletari et al., 2016) network. V-Net was originally designed for medical image segmentation and was chosen for this project due to its ability to take full volumes as its input as opposed to slicing 3D images. V-Net is a fully volumetric convolutional neural network built following the well-known architecture of U-Net (Christ et al., 2016). A schematic diagram of its architecture is shown in Figure 4.2. Following Figure 4.2 left to right, the model is composed of a compression and decompression path, much like U-Net.

Each path is made up of multiple stages containing one to three convolutional layers, which enable the learning of a residual function. This residual learning ensures convergence, unlike U-Net. The convolutional layers in the stages of the compression path have a volumetric kernel or filter made up of $5 \times 5 \times 5$ voxels. In order to halve the spatial sizes of the output feature maps and double the number of feature channels, a $2 \times 2 \times 2$ voxels filter is applied with a stride of two voxels. The non-linearity activation function used between each stage is the Parametric

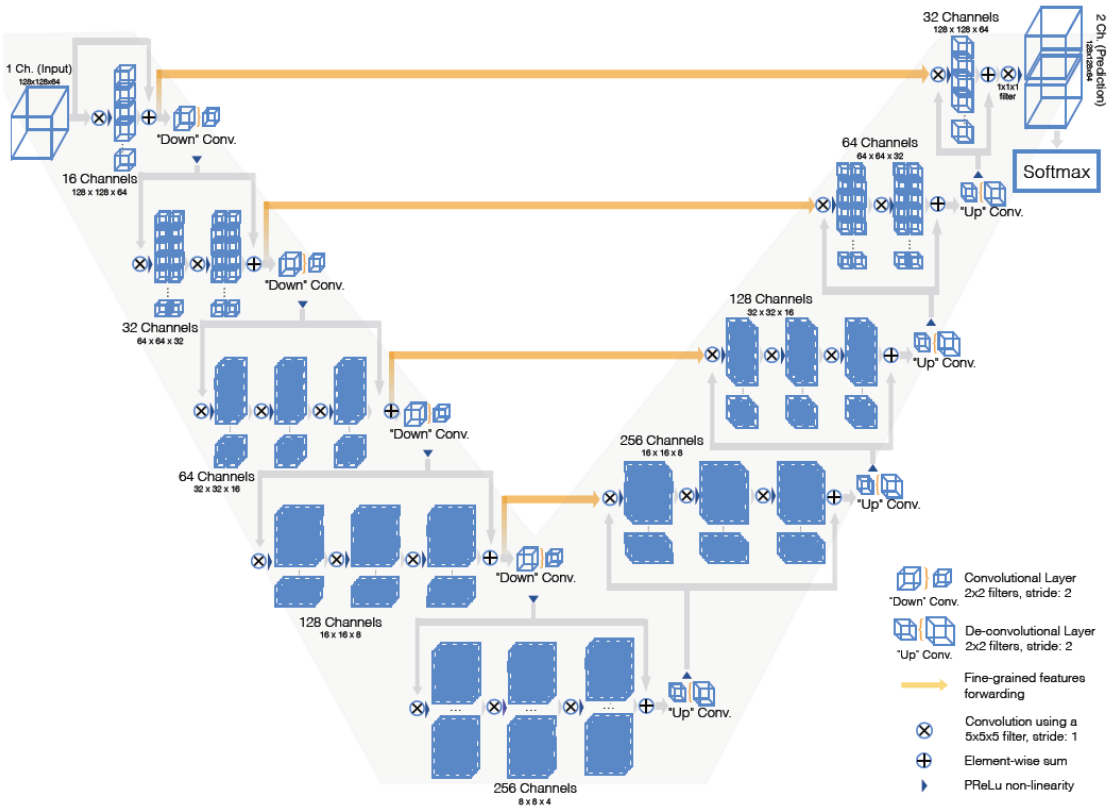


Figure 4.2: An architecture diagram of V-Net, taken from (Milletari et al., 2016), with the compression path shown on the left and the decompression path shown on the right.

Rectified Linear Unit (PReLU) (He et al., 2015). This is a modification of the well-known Rectified Linear Linear Unit (ReLU) (Agarap, 2018), which directly outputs the input if it is positive and otherwise outputs zero. PreLU simply generalises this by using a slope for negative values instead of just outputting zero.

The decompression path increases the spatial size of the input to each layer using de-convolution. The de-convolutions are then followed by one to three convolutional layers with filters the same size as those used in the compression path. With each convolutional layer, the number of filters is halved and again a residual function is learnt. As a result, two output feature maps are produced with the same dimensions as the volume inputted into the network. These two probabilistic maps of the foreground and background regions are produced using voxel-wise soft-max regression (Gao and Pavel, 2017), which essentially normalizes the map so that the voxel values follow a probability distribution that has a sum total of one. Much like U-Net, in order to revive the location information lost in the compression path, horizontal links are used to propagate extracted features to the decompression paths at each stage.

The sources aimed to be found occupy only a very small region of the H I cubes, see Figure 3.4. This increases the risk of the training process getting stuck in a local minimum of the loss function. This could result in predictions that are biased to the noise and the model may miss or partially detect sources. Since this has been an issue for medical imaging, sample

re-weighting based loss functions have been employed, which favour the sources during learning. However, another approach to counter-act this is to use a loss function and evaluation metric that is inherently balanced. The most common of these is the Dice Coefficient (Sørensen, 1948), also known as the F-score. The F-score maximizes the measure of the overlap of two masks, ranging from 0 to 1. For A and B it is defined as twice the intersection divided by the union of each mask as follows:

$$D = F_{score} = \frac{2(A \cap B)}{A + B}, \tag{4.5}$$

$$= 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}, \tag{4.6}$$

where the recall or the completeness measures the proportion of detected sources and the precision or purity measures the proportion of segments which match the ground truth or target masks as follows:

$$\text{precision} = \frac{tp}{tp + fp}, \tag{4.7}$$

$$\text{recall} = \frac{tp}{tp + fn}. \tag{4.8}$$

Here tp, tn, fp and fn are the true positives, true negatives, false positives and false negatives respectively. An adaption of the Dice Coefficient is used for the loss function of V-Net and is defined as follows (Milletari et al., 2016):

$$D = \frac{2\sum_i^N p_i g_i}{\sum_i^N p_i^2 + \sum_i^N g_i^2}, \tag{4.9}$$

where N is the number of voxels in the segmentation mask, p_i, with the ground truth mask g_i. The gradient with respect to the j-th voxel is therefore defined as:

$$\frac{\partial D}{\partial p_j} = 2\left[\frac{g_j(\sum_i^N p_i^2 + \sum_i^N g_i^2) - 2p_j(\sum_i^N p_i g_i)}{(\sum_i^N p_i^2 + \sum_i^N g_i^2)^2}\right]. \tag{4.10}$$

This allows a balance between the foreground and background voxels without needing to weight them and has been shown to outperform those with sample re-weighting methods.

4.4 Identifying real sources

During the evaluation of these methods, the location of the brightest voxel, see Section 5.4, for each of the detected sources was used to cross-reference with the ground truth mask and mark them as true or false positives accordingly. The F-score was calculated for each of the true positives and those with a score below 0.11 were re-labelled false positives to ensure that detected artifacts that happened to fall in the same position of a mock galaxy were not included in the true positive catalogue.

As previously mentioned, the ground truth masks created from the inserted mock galaxies were

considered only partially labelled as they do not include the existing real sources also within the cubes. Therefore the false positives of each method were inspected by eye, by overlaying the moment zero H I emission maps over their optical counterparts taken from the Panoramic Survey Telescope and Rapid Re-sponse System survey (Pan-STARRS) (Chambers et al., 2019). Note that the moment zero map is created by integrating the H I emission cube over all the channels.

To reduce the time spent inspecting these false positives, they were first filtered according to their spatial and spectral sizes to ensure only realistic sources were cross-referenced. Only masks with spatial sizes larger than the beam size but smaller than 0.3 Mpc and spectral widths between 7 $\mathrm{km\,s^{-1}}$ and 750 $\mathrm{km\,s^{-1}}$ were overlaid on their optical counter-parts. The detected sources that appeared to have optical counterparts or be realistic H I emission sources were then investigated and labelled as real sources.

4.5 Post-Processing Machine Learning

Some of the classical machine learning methods mentioned in Section 2.2 were used for post-processing purposes, including Support Vector Classification (SVC), K-Nearest Neighbors, Decision Trees and Random Forests. As previously mentioned, a Support Vector Machines (SVM) (Vapnik, 1995) is a kernel-based technique that finds the optimal hyper-plane in an N-dimensional space which maximizes the distance between the training data in each class. The hyper-plane is defined as follows:

$$\vec{w} \cdot \vec{x} - b = 0, \tag{4.11}$$

where $\vec{w}$ is a weighted vector, $\vec{x}$ is the input data, and b is the bias with the maximum distance to the nearest data for each class, such that:

$$|\vec{w} \cdot \vec{x} - b| = 1. \tag{4.12}$$

Note that there was no use of a regularization parameter to determine how much the model is penalised for misclassifications because it was found to have a negligible effect on the results. A kernel function is used to map the original finite-dimensional space to a higher-dimensional space, to make the classes more easily separable. For this experiment, a non-linear kernel called the Radial Basis Function (RBF) kernel function (Mark, 1996) was used with a kernel co-efficient:

$$\gamma = \frac{1}{N_{\mathrm{features}}\sigma^2}, \tag{4.13}$$

where N_{features} is the number of features and σ^2 is the variance of the data-set. This kernel function was chosen as it was found to best improve the purity of the source finding methods when applying SVC.

The weights associated with each class were automatically adjusted according to the frequency of that class by a factor of:

$$\frac{N_{\mathrm{samples}}}{N_{\mathrm{classes}}N_i}, \tag{4.14}$$

where N_{samples} is the number of samples, N_{classes} is the number of classes and N_i is the number of samples in class i.

K-Nearest Neighbors (KNN) (Cunningham and Delany, 2007) identifies groups of K many samples in the data-set based on feature similarity, and uses a majority vote to classify them. In this project the number of neighbors, K, was chosen to be one and the Minkowski distance (Li et al., 2011) of the order two was used to measure the similarity between the points in a sample size of n as follows:

$$d(q,p) = \sqrt{\sum_{i=1}^{n} |q_i - p_i|^2} \ , \tag{4.15}$$

where q_i and p_i are the two points in sample i.

A Decision Tree (Breiman et al., 1984) is a classifying method that builds a tree based on the features of the data. A Random Forest (Ho, 1995), on the other hand, creates an ensemble of uncorrelated decision trees and takes the mean or majority vote of their output. Similarly to the SVC, the class weights for both the Decision Tree and Random Forrest used in this project were automatically adjusted to the inverse of the frequency of each class. In both algorithms, the tree nodes are split downward by a decision making process which is based on the relevance of each feature. In this project, the Gini Index (Analyttica Datalab, 2018) was chosen to measure the quality of the splits. The Gini Index calculates the probability of a randomly selected variable being classified incorrectly, also known as the impurity, and is expressed as:

$$G = 1 - \sum_{i=1}^{n} (P_i)^2, \tag{4.16}$$

where P_i is the probability of a variable being classified into a distinct class. In both the Decision Tree and Random Forest used for post-processing the catalogues, the maximum depth of the trees was chosen to be twelve to optimize the purity of the source-finding methods.

Chapter 5

Experimental Setup

Three main experiments were investigated and compared in this project, the setup of which is covered in this chapter. The same machine was used for all the experiments, excluding the training of the 3D CNN. The experiments were run on a machine belonging to the astronomy department at the University of Groningen, **norma4**. The machine consists of two Intel Xeon E5-2698 v3 CPUs, each with 16 cores and 32 threads. In addition, it consists of 24X 32 GB DDR4 2133 MT/s of RAM. The training of the V-Net network was done using the **Peregrine** High-Performance Computing (HPC) cluster. This cluster also belongs to the University of Groningen and consists of 5740 CPU-cores and 220000 CUDA-cores, the details of which can be found in the Pergrine Wiki [1].

In the first experiment all three source-finding methods, SoFiA, MTObjects and V-Net, were run on the test data-set and compared. The implementation of each of these source-finding methods is discussed, as well as how the data was inputted into each of them. In the second experiment shallow machine learning models were trained and run on the resulting catalogs of Experiment 1 in order to reduce the impurity. The final experiment aimed to improve the quality of the V-Net network by adding real sources found in the previous experiments to the labels used in the training process. The chapter ends by explaining the evaluation metrics used to compare these experiments.

5.1 Experiment 1: Comparing all three methods

The first experiment involved running all three source-finding methods on the H I emission spectral windows containing the inserted mock galaxies. Masks indicating which pixels contained the inserted mock galaxies were used as a ground truth or label for both the training of the 3D CNN and for the evaluation of all the methods. The results were cross-matched with their optical counterpart from the Panoramic Survey Telescope and Rapid Re-sponse System survey (Pan-STARRS) (Chambers et al., 2019). This was done in order to identify real sources and remove them from the output catalogs of each method for the evaluation process. Although this approach does not include the detection of all the real sources, it removes the penalisation of

[1] https://wiki.hpc.rug.nl/peregrine/introduction/cluster_description#dokuwiki__top

each method for picking up potentially real sources. In this way the evaluation process for this experiment was based solely on the complete catalogue of the mock galaxies that were inserted. Each method was then evaluated and compared, not only looking at the completeness and purity of the produced catalogs, but also at the properties of the masks produced to evaluate their quality, see Section 5.4.

5.1.1 SoFiA Implementation

The latest version of SoFiA at the time of the project, version 2.3.1 (Flöer et al., 2021), was used for this experiment. This version was implemented in C, making it much faster than the previous versions implemented in Python. In addition, this version makes extensive use of a technique which distributes the processing of the code amongst several processors, known as multi-threading. Due to the design of SoFiA, it was able to take a full spectral window as the input without the need to break it up into smaller sub-cubes. In order to correct for spatial and spectral noise variations, the inputted spectral window was divided by its measured local noise and the noise was linearly interpolated between grid points. This was done by moving a window across the inputted H I emission cube with a grid size of 75 pixels spatially (**scaleNoise.windowXY**) and 51 channels spectrally (**scaleNoise.windowZ**). This measurement was made using all the voxels in the spectral window irrespective of their flux (**threshold.fluxRange**).

As previously mentioned, the smooth and clip (S+C) method was the chosen detection algorithm (**scfind.enable**), with the use of a boxcar smoothing kernel in the spectral direction and a Gaussian smoothing kernel in the spatial direction. These smoothing kernels were applied iteratively with the following full width at half maxima (FWHMs) of 0, 3.6, 6.6 pixels spatially (**scfind.kernelsXY**) and 0, 3, 5, 9, 15 channels spectrally (**scfind.kernelsZ**). These values were chosen experimentally to optimize the output by balancing the purity and completeness. In each of these smoothing iterations the flux threshold for the smooth and clip algorithm (**scfind.threshold**) was taken to be 3.5 times the measured noise level.

Once the voxels that contain sources were found, they were merged to create coherent detections (**linker.enable**). This was done with the maximum of the aforementioned linking length set to 133 channels in the spectral dimension (**linker.maxSizeZ**) and zero in the spatial dimensions (**linker.maxSizeXY**). The minimum linking length was taken to be three voxels spatially (**linker.minSizeZ**) and two channels spectrally (**linker.minSizeXY**).

A reliability score was assigned to the detected sources by estimating the density of the positive and negative detections in the 3D parameter space. The reliability score made use of a Gaussian kernel, scaled by a factor of 0.5 (**reliability.scaleKernel**). The reliability score was then thresholded at a value of 0.9 (**reliability.threshold**) and sources with a reliability below this value were removed from the catalogue. Lastly, sources with a signal-to-noise ratio below three (**reliability.minSNR**) were also removed from the catalogue. All the above mentioned parameters are listed in Table 5.1. A full description of SoFiA's control parameters and their defaults, which were taken for the unmentioned parameters in this section, can be found in the

Table 5.1: The chosen parameters for SoFiA 2.3.1

Parameter Name	Set Value
scaleNoise.windowXY	75
scaleNoise.windowZ	51
threshold.fluxRange	full
scfind.enable	true
scfind.kernelsXY	0, 3.6, 6.6
scfind.kernelsZ	0, 3, 5, 9, 15
scfind.threshold	3.5
linker.enable	true
linker.maxSizeZ	133
linker.maxSizeXY	0
linker.minSizeZ	3
linker.minSizeXY	2
reliability.scaleKernel	0.5
reliability.threshold	0.9
reliability.minSNR	3

SoFiA repository[2].

5.1.2 MTObjects Implementation

As previously mentioned, the Arnoldus (2015) implementation of MTObjects was used for this project, since it was designed for 3D radio data cubes. Due to the high memory usage of MTO, a sliding window was used to sweep through the cubes and create smaller inputs. Since the size of the windows were only limited by the memory of the machine used (**norma4**), the largest windows possible were taken. The windows therefore each consisted of $652 \times 200 \times 300$ channels and pixels respectively. In this way, the sliding window only needed to sweep across the pixels of the H I emission cube to ensure the coverage of the entire spectral window, as the full length of the channels were covered in a single window. Recall that the full spectral window contained $652 \times 1800 \times 2400$ channels and pixels, which may be able to be processed in full by a machine with higher memory. An overlap of these windows needed to be chosen to prevent cutting off galaxies as much as possible, however, to reduce computational cost it also needed to be minimized. Therefore, following the minimum average from Figure 3.4, the dimensions of the overlapping regions were taken to be 15×20 pixels, which are equivalent to 3.33% of the full spatial window regions.

Many of the parameters of MTO were chosen from the results of Arnoldus (2015) and can be seen in Table 5.2. The code was executed across 64 processes (**Threads**), making use of multi-threading much like SoFiA, and the number of bits per pixel was set to 32 (**BitsPerPixel**). Following the steps shown in Figure 4.1, for each window the background was estimated (**BackgroundEstimation**) by calculating the mean (**MeanBackground**) and standard devi-

[2] https://github.com/SoFiA-Admin/SoFiA-2/wiki/SoFiA-2-Control-Parameters

Table 5.2: The chosen parameters for MTObjects implemented by Arnoldus (2015). Note that the mean (**MeanBackground**) and standard deviation (**StdDev**) of the background were calculated for each window of the sliding window. In addition, **InverseGain** and **Lambda** are depreciated and therefore were not set above zero.

Parameter Name	Set Value
Threads	64
Connectivity	6
MeanBackground	Dependent on window
StdDev	Dependent on window
MoveupFactor	0.2
BackgroundEstimation	True
ShowObjects	True
InverseGain	0
Lambda	0
BitsPerPixel	32
SignificanceLevel	1×10^{-5}
ExportAttributes	False
StatisticalTest	"FluxDensityRadio"
SmoothingDegree	0
MoveWrtIntensity	False
DetectNestedObjects	True
MinSpatialWidth	3
MinSpectralWidth	2

ation (**StdDev**) of regions without sources, as explained in Section 4.2.1. A max-tree mask was then created by applying a Gaussian filter with a FWHM of 2 pixels, followed by the previously described Perona–Malik diffusion filter (Perona and Malik, 1990) with the time or degree of smoothing set to 10. Since this was manually implemented it was not a part of the parameter file, the smoothing parameter (**SmoothingDegree**) was not used. The max-tree was filtered using the flux density (**StatisticalTest**), with the significance level for the statistical test set to 1×10^{-7}. To prevent the ancestor nodes of sources from also being labelled as sources, labels are only able to move up by a factor of 0.2 (**MoveupFactor**). The source finding process allowed for the detection of nested objects (**DetectNestedObjects**).

Once the voxels containing sources were labelled, coherent detections were created by merging source-containing voxles within six voxels from each other (**Connectivity**). A minimum spatial width of three pixels (**MinSpatialWidth**) and a minimum spectral width of two channels (**MinSpectralWidth**) were used to remove unrealistic sources from the resulting catalogue. In order to combine the segmented sub-cubes of the sliding window, the masks were binarised and the maximum value in the overlapping regions was taken. This was to ensure that if any of the windows detected part of a source it was included. The mask was then labelled again by connecting any two neighboring voxels belonging to a source.

5.1.3 V-Net Implementation

The V-Net network has been implemented in PyTorch (Paszke et al., 2019) and is available through MedicalZooPytorch (Nikolaos, 2019). MedicalZooPytorch[3] is an open-source 3D medical segmentation library, implemented in PyTorch. PyTorch is an open source machine learning Python library which supports CUDA tensor types. It can therefore utilize GPUs for computational purposes, which is advantageous for computer vision tasks.

The model expects an input of dimensions $128 \times 128 \times 64$ voxels, with the channels as the last dimension as opposed to the first. Therefore, much like the implementation of MTO, a sliding window was used to create the input for the network. However, the window sizes were much smaller than those used by MTO and therefore required sweeping across the spectral windows both spatially and spectrally. The windows were given an overlap of size $15 \times 20 \times 20$ voxels, with the channels as the last dimension, which is 11.72%, 15.63% and 31.25% of the full spectral window dimensions.

In supervised machine learning models, the data used needs to be split into three subsets. First the data is split to create a test set which the model has never seen before, in order to fairly evaluate it. The remaining data is then further split into a training set and a validation set. This allows the model to continually learn by training on one subset of data and evaluating the progress of its learning with another.

In choosing how to split the data into these subsets, it is important to consider the potential for bias if each of these subsets are not equally representative of the data. This is particularly an issue for the H I emission cubes used, as each represents a different spectral window and this changes the appearance of the sources within them. To account for this, the test set was taken to be Pointing 1, which contained all seven spectral windows, see Table 3.1. The sliding window was then applied to all the spectral windows of Pointing 2 and 80% of these windows within each spectral window was taken for training and the remaining 20% was taken for validation. This ensured that the training and validation had an equal amount of sub-cubes from each spectral window. The segmentation task was treated as a two class problem, with the two classes being background voxels and source-containing voxels.

Optimizers are used in neural networks to modify their parameters in order to reduce their loss score. For the implementation of V-Net in this project, the Adaptive Moment Estimation (ADAM) (Kingma and Ba, 2017) was chosen as the optimizer as it combines the best properties of Resilient Backpropagation (RMSprop) (Igel and Hüsken, 2000) with those of another optimizer called An Adaptive Learning Rate Method (ADADELTA). It has been found to outperform the well-known Stochastic Gradient Descent (SGD) (Sebastian, 2016) when many parameters are used and when saddle points are encountered.

A batch size of four was taken, which determines the number of samples that are propagated at a time through the network of V-Net during the training process. The learning rate determines the amount the model changes when the weights are updated by the loss score. Through experimentation, a learning rate of 1×10^{-3} was chosen to maximize it without slowing down the

[3] https://github.com/black0017/MedicalZooPytorch

learning process too much. Early stopping was implemented to chose the number of epochs, essentially stopping the training process once the process has shown no learning improvement.

Once the network was trained on Pointing 2, Pointing 1 was used for testing the model. Much like the training, the spectral windows were reshaped to have the channels as the last dimension and a sliding window of size $128 \times 128 \times 64$ voxels, in pixels and then channels, was run over the full spectral window. The trained model was then used to infer a probability mask of the window, which was binarised by taken setting probabilities above zero to one and those below to zero. Once a mask was created for each sub-cube of the sliding window, they were stitched back together by taking the maximum value, either zero or one, in the overlapping regions. Each of the sources in the final binary masks were then labelled by connecting any two neighboring voxels belonging to a source.

5.2 Experiment 2: Machine learning post-processing

In attempt to improve the purity of all the methods a post-processing step was implemented after each to remove as many false detections as possible. The rejection of detections could have been based directly on how realistic the attributes are, for example the size of the galaxy. However, for larger volumes like the ones in this project, far too many would be flagged due to unrealistic assumptions. Therefore a shallow machine learning approach was used instead, including Support Vector Classification (SVC), K-Nearest Neighbors, Decision Trees and Random Forests, all of which are described in Section 4.5. In this project, all the models were taken from the scikit-learn (Pedregosa et al., 2011) package with the parameters chosen to optimize the reduction of the impurity for all methods.

All three source-finding methods were run on both Pointings 1 and 2, using the cross-matching and removal of real sources approach of Experiment 1. The catalogue of detected sources from the seven spectral windows of Pointing 2 were then used to train the four machine learning classification methods. The training catalogue of sources contained an over-representation of false positives and under-representation of true sources by MTO in the most distant spectral window. In order to prevent this from creating a bias in the machine learning models, the catalogue was under-sampled by randomly removing some of those false detections and the number of true sources detected by MTO in that spectral window were over-sampled. The over-sampling was done using the Synthetic Minority Oversampling Technique (SMOTE) (Chawla et al., 2002). SMOTE works by selecting a random variable from the minority class, randomly selecting one of K neighbors to that point, and then creating a synthetic example between the two points. For this experiment K was taken to be five. Both re-sampling methods were taken from the imbalanced-learn package (Lemaître et al., 2017).

The volume in voxels, the total and peak flux in $\mathrm{Jy\,km\,s^{-1}}$, the velocity width in $\mathrm{km\,s^{-1}}$, the spatial dimensions in kpc and the elongation of each remaining detection in the catalogue was then used to train each classification model. Note that the asymmetry was not used to train the model, nor was the properties of the ground truth masks, to prevent any bias in the models. Once trained, each classification model was run on the test cubes, and the detections classified as false detections were removed from the catalogue to re-calculate the purity and

completeness of each method.

5.3 Experiment 3: Improving the V-Net network with detections of real sources

In the third experiment, the optically cross-matched sources detected by each method in Experiment 1 were used to improve the ability of the 3D CNN to locate and mask not only mock galaxies but real sources. In order to create ground truth masks for the real sources found, the area of the mask was chosen based on which source finding methods detected them in Experiment 1. If a source was detected by both SoFiA and MTO, the union of both the detected masks was taken as the ground truth. This was the case even if V-NET also had a detection of the source, to prevent the lack of asymmetry in the V-NET detections interfering too much with the new ground truth. However, if a source was found by only one method, that mask was used.

There were two different approaches to improving the source finding and mask quality of the network for real sources. Both approaches used the same parameters for the model as in Experiment 1, and the same set of seven spectral windows for training and testing, with adjustments for each experiment. Experiment 3.1 involved the simplest attempt to improve the network by adding the newly found real source masks to the ground truth. The network was then re-trained from scratch with labels for both the mock galaxies and real sources. Experiment 3.2 attempted to improve the network using transfer learning, or more specifically fine-tuning, see Section 4.3.1. This was achieved by freezing all but the weights of the final layer of the network and using only the newly masked real sources to update them.

To extend each of these approaches to a full pipeline, the machine learning post-processing method in Experiment 2 was repeated on the output of each model. However, as the aim of this experiment was to improve the detection of real sources rather than mock galaxies, the model was re-trained with the training set over-sampled using SMOTE to ensure that there were as many real sources as mock galaxies. The classification model used for this post-processing step was chosen based on the best performing one in Experiment 2.

5.4 Evaluation Metrics

In order to evaluate and compare each of the source finding methods in each experiment, the resulting masks were mapped one-to-one to the ground truth masks for each source. The sources were matched using the brightest voxel of each source and taking the source with the voxel of the highest flux when a segment contained the brightest voxel of many ground truth masks. In order to give an overall comparison of different source-finding methods, two merging scores were calculated and used in conjunction with the F-score to find a final combined score. In addition, the physical properties of the masks created by the source finders were calculated and compared in order to find any significant differences between the methods. Finally, the degree of asymmetry was calculated to determine if there were any bias in any of the methods as a result of using perfectly symmetric mock galaxies.

5.4.1 Merging Scores

Two metrics were used to measure the accuracy of the segmented volumes of the sources detected, while penalising those that were found to divide up large sources. The under-merging score, UM, measures how much the segmentation method breaks up sources which should be a single segment and the over-merging score, OM, measures the opposite. Both are defined by Haigh et al. (2021), which adapted the work of Levine and Nazif (1981), and are extended to three-dimensions as follows:

$$\mathrm{UM} = \sum_{k=1}^{M} \frac{(V_k - (T_j \cap R_k))(T_j \cap R_k)}{V_k}, \tag{5.1}$$

$$\mathrm{OM} = \sum_{k=1}^{N} \frac{(v_j - (T_j \cap R_k))(T_j \cap R_k)}{v_j}, \tag{5.2}$$

$$\tag{5.3}$$

where $R_1, ..., R_N$ are the ground truth segments with volumes $V_1, ..., V_N$ and $T_1, ..., T_M$ is the detected segments divided into M segments to maximise each $T_j \cap R_k$. The combination of the two metrics results in the volume score which indicates the quality of the mask. The volume score is defined as follows Haigh et al. (2021):

$$\mathrm{V_{score}} = 1 - \sqrt{\mathrm{OM}^2 + \mathrm{UM}^2}. \tag{5.4}$$

A final combined score was used and the mean of this value over each frequency range was calculated to rank and compare all the source finding methods. This score took the precision and recall into account by using the F_{score} from Equation 4.5, as well as the merging scores as follows:

$$\mathrm{Comb} = \sqrt[3]{(1 - \mathrm{OM})(1 - \mathrm{UM})F_{\mathrm{score}}}. \tag{5.5}$$

5.4.2 Physical Properties

The physical properties of the masks created created by each source-finding method were calculated to compare and search for any significant differences between the methods. The spatial sizes of the masks were calculated following Meyer et al. (2017):

$$\Delta x = d \tan d_x N_x, \tag{5.6}$$

$$\Delta y = d \tan d_y N_y, \tag{5.7}$$

$$\tag{5.8}$$

where d is the distance in Mpc, d_x and d_y are the spatial voxel separations in radians, and N_x and N_y are the voxel widths accordingly. The velocity width was calculated as:

$$\Delta v = \frac{N_z d_\nu c}{\nu_{\mathrm{rest}}}, \tag{5.9}$$

where d_ν is the spectral voxel separation in Hz, N_z is the spectral voxel width, c is the speed of light and ν_{rest} is the rest frequency of neutral hydrogen. In addition to the physical size of the object, the spatial elongation of each mask was calculated following the description in Section 4.2.1. The peak and total fluxes were calculated using Equation 3.17 and the H I mass was then calculated using Equation 3.20.

5.4.3 Degree of Asymmetry

To determine the effect that training the V-Net network with only perfectly symmetric mock galaxies had on its ability to mask asymmetric real sources, the degree of asymmetry was calculated and compared for every mask of each source-finding method. The calculation of the asymmetry first required the creation of a signal-to-noise map for the segmented masks as follows:

$$\mathrm{SNR} = \frac{\sum I}{\sqrt{N}}, \tag{5.10}$$

where SNR is the signal-to-noise ratio at a certain pixel, $\sum I$ is the summed intensity over all channels in that pixel, and N is the number of channels in that pixel.

For each source, all three source-finding methods produced different masks contained in different size bounding boxes. Therefore, to ensure that the asymmetry metric was calculated within the same size bounding box for a single source for every method, the largest of these boxes was used for the calculation. The signal-to-noise map was weighted accordingly to ensure that the fainter regions of a source contributed less to its degree of asymmetry. The assigned weight at each pixel, w, was defined as:

$$w = \begin{cases} 1/3, & \text{if } 0 < \mathrm{SNR} \leq \sigma, \\ 2/3, & \text{if } \sigma < \mathrm{SNR} \leq 3\sigma, \\ 1, & \text{if SNR} > 3\sigma, \end{cases} \tag{5.11}$$

where σ is defined as the standard deviation of the SNR value of that pixel. The final asymmetry metric, M_A, was then computed as follows:

$$M_A = \frac{\sum |w - w_{180}|}{\sum w + \sum w_{180}}, \tag{5.12}$$

where w_{180} is the weighted value at a given pixel after the image has been rotated by 180 degrees.

In order to calculate w_{180}, an axis of rotation needed to be chosen. First the optical counterpart was cross-matched and re-projected, following Experiment 1. However, since H I gas is much more disturbed and usually extends much farther out than its optical counterpart, the SNR map was used to mask the re-projected optical image. Pixels of the signal-to-noise map that were above two standard deviations were used to mask the optical image and within this mask the brightest optical pixel, the center of mass, was chosen as the axis of rotation. The resulting asymmetry metric ranges between zero and one, with larger values indicating a more asymmetric mask.

Chapter 6

Results

This chapter covers the results of the three experiments conducted in this project. The first experiment, covered in Section 6.1, tests and evaluates all three source finding methods, SoFiA, MTO and V-Net, and their ability to find mock galaxies. The second experiment, described in Section 6.2, used classical machine learning algorithms to improve the purity of all three source finding methods. The final experiment, described in Section 6.3, aimed to improve the ability of the V-Net network to find and correctly mask not only mock galaxies but also real sources.

6.1 Experiment 1

The first experiment began by training the V-Net 3D CNN on all seven spectral windows from Pointing 2. The network was trained using the masks of the mock galaxies as the ground truth. The learning progress of the network can be seen in Figure 6.1, with both the training and validation loss scores as a function of epoch. Up to the ninth epoch the validation loss score decreases with the training, with only a small difference between it and the training loss score. The model used to detect sources in the seven spectral windows of Pointing 1 was therefore taken after nine epochs of training. This prevented over-fitting and ensured the generalization capability of the model, while taking the lowest possible loss for the best quality model. In this way the trade-off between the bias and variance was taken into consideration, see Figure 2.3. It took about 13.5 days to train these nine epochs, with the model taking around 1.5 days per epoch to train and validate. Note that these times are specific to the machines used, namely the Peregrine cluster mentioned in Chapter 5.

Once the network was trained, the seven spectral windows of Pointing 1 were used as the input for all three methods to produce the detections. The time taken to run the segmentation on each spectral window of Pointing 1 was measured for all three source finding tools, this can be seen in Table 6.1. The same machine was used for all the source finding methods, namely **norma4**, mentioned in Chapter 5.

SoFiA was found to be the fastest of the three, followed by MTO, which took just under three times the amount of time to run. V-Net was found to be the slowest, taking over 6.9 times longer than SoFiA and over 2.5 times longer than MTO. However, this was expected, as

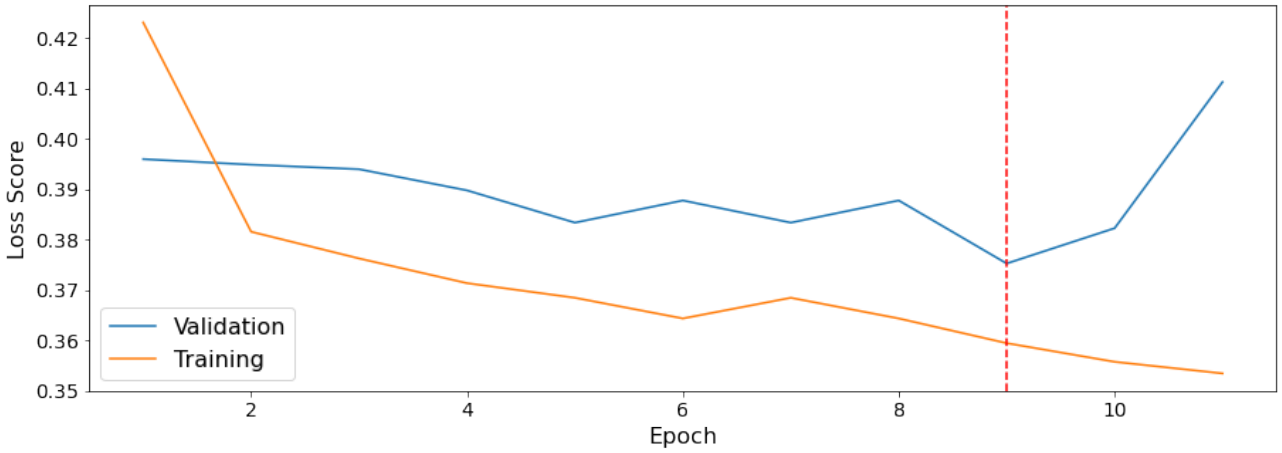


Figure 6.1: The training and validation loss scores of the V-Net network per epoch during the training process for Experiment 1. The blue line shows the validation loss score while the orange line shows the training loss score, with the red dashed line indicating the point where the model was taken for inference.

Table 6.1: The elapsed real time in seconds for each of the three source finding methods for a single spectral window of size 2.81664 Giga voxels or $652 \times 1800 \times 2400$ voxels.

Method	Elapsed Time
SoFiA	3439.49 s
MTO	9524.68 s
V-Net	23644.80 s

SoFiA is run on the entire spectral window while MTO is run on smaller pieces of a sliding window and V-Net uses even smaller window sizes than MTO. The large time difference is most likely primarily due to the need to run the source finders on each window and stitch them together, which was done sequentially. It is important to note that this is an evaluation of the time taken to find detections in the spectral windows from Pointing 1 and does not include the time taken to train the V-Net network.

6.1.1 Identifying real sources

To remove any potentially real sources from the evaluation process, the detections that were not found to match any mock galaxies were then filtered by their physical dimensions and cross-matched with optical data, following Section 4.4. An example of the H I moment zero contours, masked by the detections of each method and overlaid on an optical image of the same area of the sky can be seen for each spectral window in Figure 6.2. In these examples of real sources there appears to be sources of asymmetry in the detections of SoFiA and MTO. For example, in the (b) there appears to be signs of the gas being stripped from the galaxy, also known as ram pressure stripping, and in the (d) there appears to be two galaxies interacting. However,

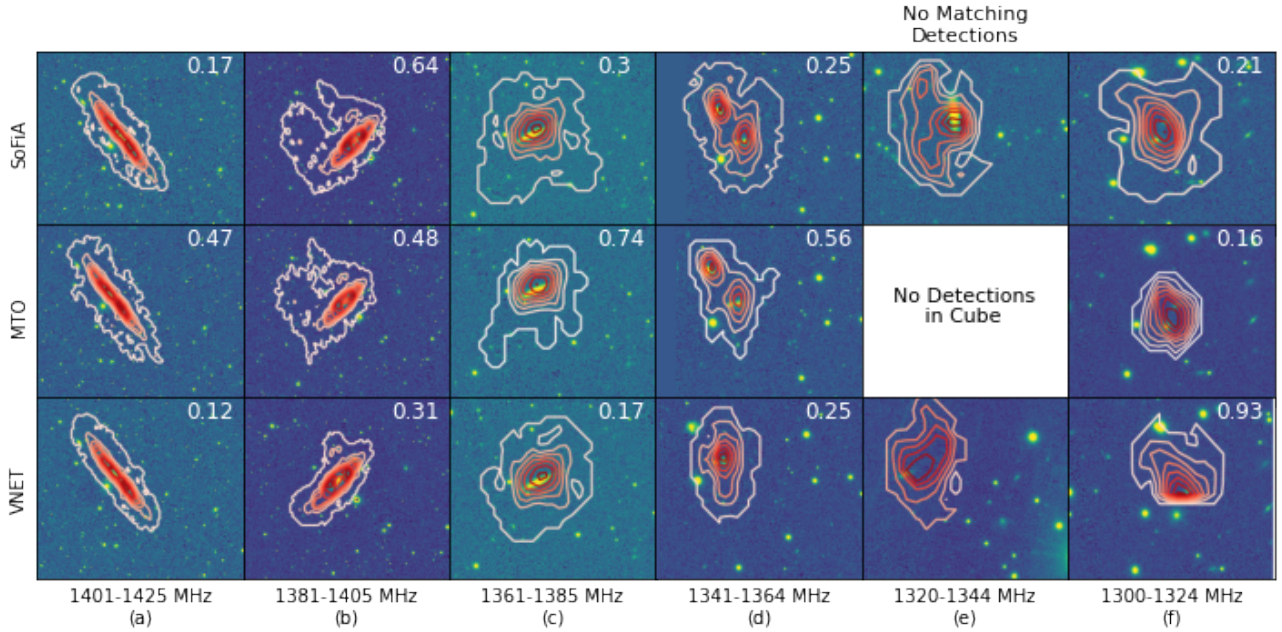


Figure 6.2: Examples of masked H I moment zero contours of detections of real sources overlaid on their optical counterparts for each source finding method. From left to right shows an example from each spectral window of Pointing 1, with the mask from SoFiA, MTO and V-Net, from top to bottom respectively. Note that the last spectral window was excluded due to the lack of real sources detected by each method. All the examples are matched across the methods by their brightest voxel, except for the 1361-1385 MHz spectral window in (e), since that spectral window contained no overlapping detections between the methods or any detections of real sources by MTO. In the top right corner of each image is the degree of asymmetry for that masked source.

these asymmetries are not covered by the mask of V-Net, as illustrated by the detection in the (b) and (d) of Figure 6.2. This suggests that the V-Net struggles to mask asymmetry, which is expected as the network was only trained on symmetric mock galaxies.

To investigate to what extent the trained V-Net network further struggled to mask asymmetry, the degree of asymmetry of each masked source was measured, following Section 5.4. The resulting distribution of the degree of asymmetry for each real source detected by each source finding method can be seen in Figure 6.3. This demonstrates that SoFiA finds the most asymmetry in these real sources but V-Net does not seem to mask these real sources as purely symmetrically as expected. Taking a look at the overlaid asymmetry metric in Figure 6.2 in Plot (f), V-Net appears to truncate the source, resulting in a highly asymmetric mask. This appears to be the case for a number of sources, which means that although V-Net struggles to mask the asymmetric sources correctly, this is not reflected in the overall distribution of the degree of asymmetry because it is also truncating other sources. Also note the contribution of about 20% from the noise to the degree of asymmetry in the ground truth and to the detections of all three methods.

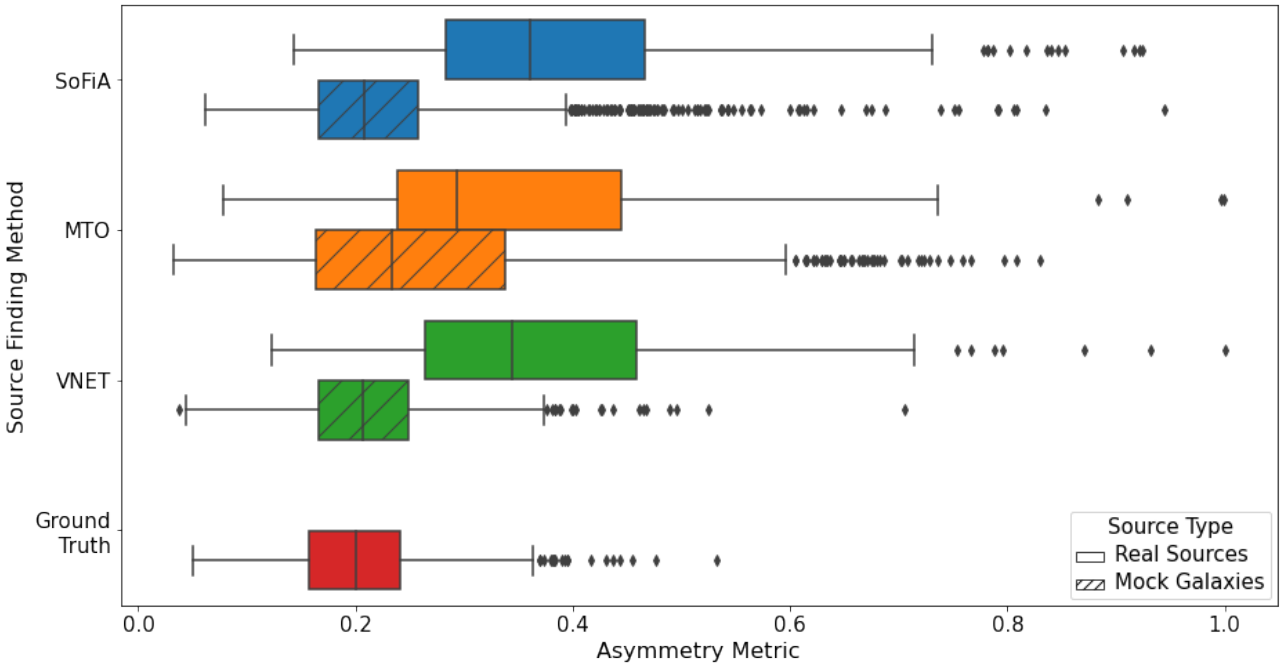


Figure 6.3: A box plot of the degree of asymmetry of real sources and mock galaxies detected by every method. The y-axis shows the different methods and the x-axis shows the different quantiles of the degree of asymmetry for the real sources and the mock galaxies that were detected by all of the methods. The real sources are denoted by the clear boxes while the mock galaxies are denoted by the hatched boxes. The detections of SoFiA are represented by the blue box plots, the detections of MTO are represented by the orange box plots, the detections of V-Net are represented by the green box plots and the ground truth of the mock sources are represented by the red box plot. Note that the ground truth is not purely symmetric because it includes the noise within that mask.

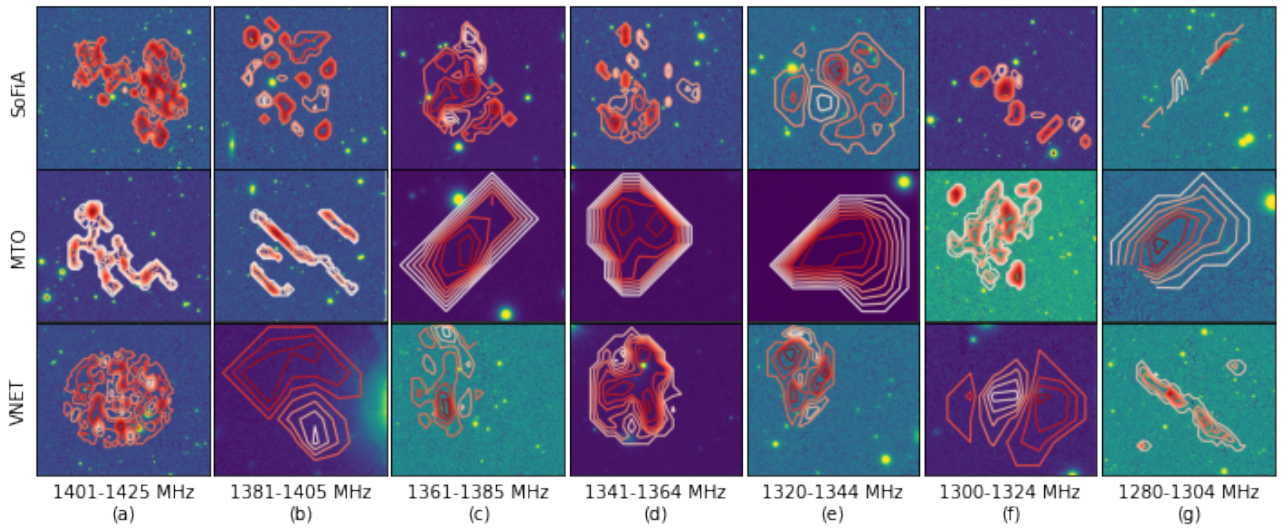


Figure 6.4: Examples of masked H I moment zero contours of false positive detections overlaid on their optical counterparts for each method. From left to right shows an example from each spectral window of Pointing 1, with the mask from SoFiA, MTO and V-Net from top to bottom respectively.

The number of detected mock galaxies, real sources and false positives by each method can be found in Table 6.2. In order to visually summarise these results, Figures 6.5 and 6.6 show Venn diagrams with the total number of detections from all seven spectral windows of Pointing 1.

With 138 real sources only picked up by SoFiA, it is clear that it is the most successful at finding the real sources. However, V-Net is not far behind with 54 exclusive detections of real sources and it succeeds best at locating the mock galaxies, with 124 only picked up by V-Net. MTO appears to find the least detections of the three methods, with 14 real sources and two mock galaxies that the other methods failed to detect. In addition, MTO picks up the most false positives, with as many as 15508 false positives that neither SoFiA or V-Net picked-up. For all the methods, the number of both real sources and mock galaxies tends to decrease with the decrease in frequency, as the sources become smaller and fainter. For the same reason, for most cases all the methods tend to pick up more false positives for the more distant spectral windows. The exception to this is the detections of SoFiA and MTO in the most nearby spectral window at 1401-1425 MHz, where many false positives are found in the 1420 MHz channel, which suggests that the H I emission of the Milky Way is interfering with their source finding.

Some examples of false positive detections overlaid in the same way as Figure 6.2 can be found in Figure 6.4. However, these examples are not cross-matched as there was not enough matching detections of false positives across the source finding methods. This indicates that the different methods have different sensitivities to structures in the noise. From these examples, it appears that SoFiA tends to pick up local variations of high noise and misinterpret it as sources, seen in the fragmented masks of Figure 6.4. Additionally, SoFiA appears to detect noise at the edges of the spectral window, seen in the Plot (g) of Figure 6.4. MTO, on the other hand,

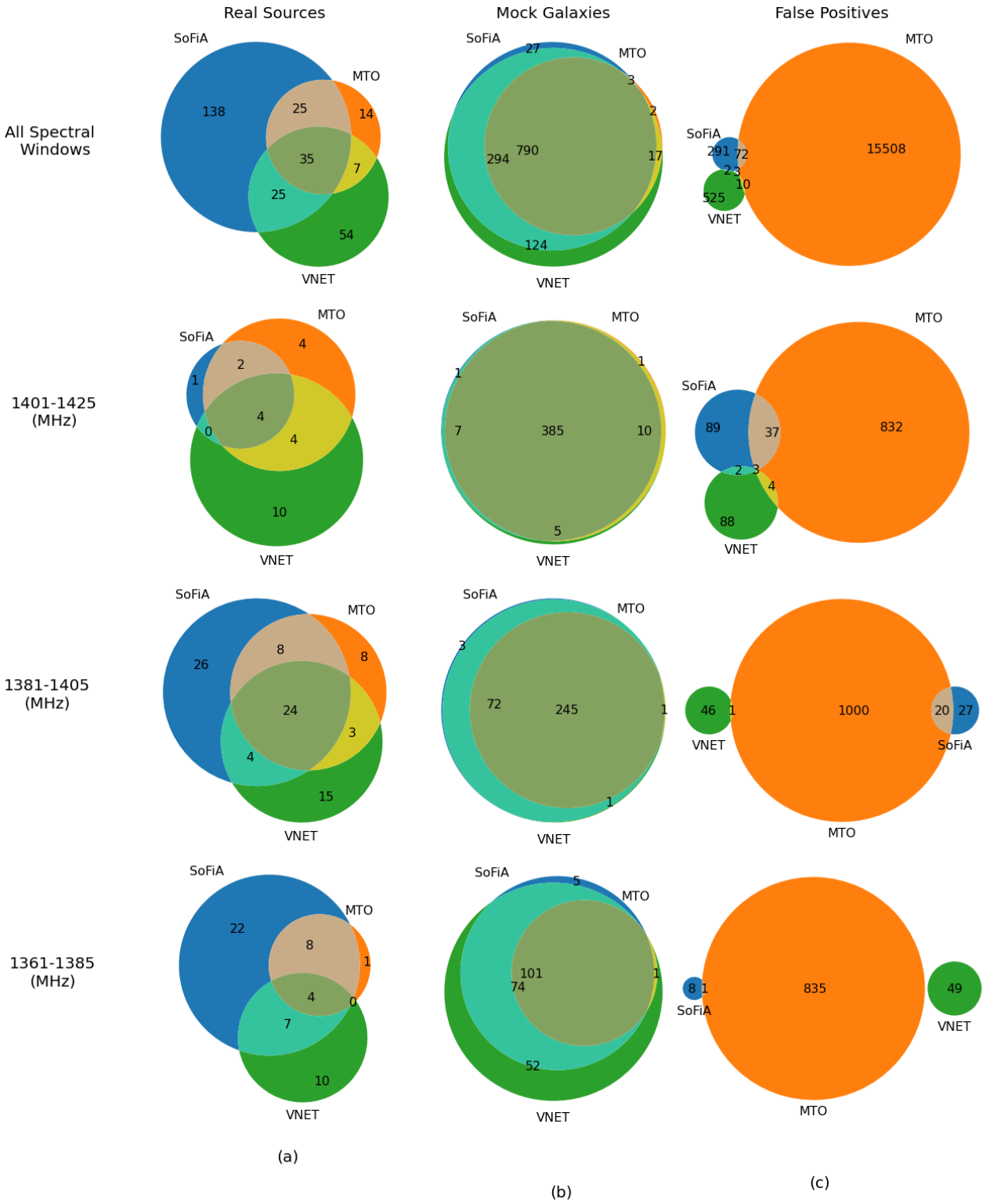


Figure 6.5: Venn diagrams of the number of each kind of detection by each method in Experiment 1. The first row shows the total detections for all seven spectral windows of Pointing 1 and the rest of the rows show the number of detections for three of the spectral window. (a) Shows the number of detected real sources (b) shows the number of detected mock galaxies and (c) shows the number of detected false positives. SoFiA is represented by the blue circle, MTO is represented by the orange circle and V-Net is represented by the green circle.

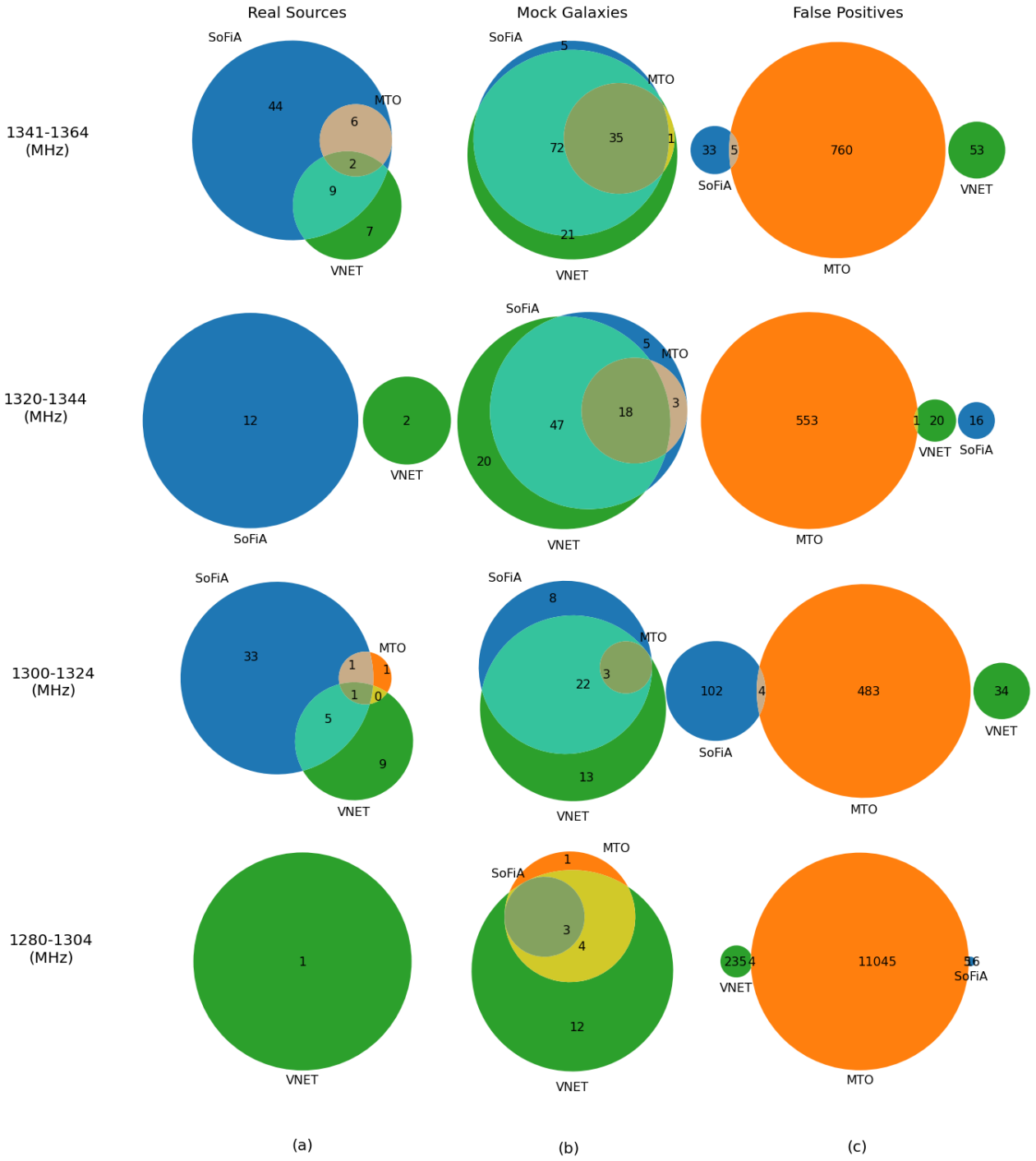


Figure 6.6: Venn diagrams of the number of each kind of detection by each method in Experiment 1 continued. Each row shows the number of detections for four of the seven spectral windows of Pointing 1. (a) Shows the number of detected real sources (b) shows the number of detected mock galaxies and (c) shows the number of detected false positives. SoFiA is represented by the blue circle, MTO is represented by the orange circle and V-Net is represented by the green circle.

tends to pick up a lot more artifacts, especially the more extended artifacts and sharp edged geometric artifacts, implying that they are very small in voxel size. V-Net appears to favour artifacts slightly similar to those picked up by MTO, but also some large round artifacts, as in Plot (a). Each method therefore has its own failings, which is expected due to the difference in their approaches.

Table 6.2: The number of mock galaxies, real sources and false positives detected per method in each spectral window for Pointings 1 and 2. For the mock galaxies, the percentage of the total number of mock galaxies in the spectral window is shown in brackets.

Pointing 1										
Frequency Range (MHz)	Mock Galaxies				Real Sources			False Positives		
	SoFiA	MTO	V-Net	**Total**	SoFiA	MTO	V-Net	SoFiA	MTO	V-Net
1280-1304	3 (1%)	8 (2%)	19 (5%)	**347**	0	0	1	21	11054	239
1300-1324	33 (12%)	3 (1%)	38 (14%)	**267**	40	3	15	106	487	34
1320-1344	73 (25%)	21 (7%)	85 (29%)	**294**	12	0	2	16	554	21
1341-1364	112 (43%)	36 (14%)	129 (50%)	**260**	61	8	19	38	765	52
1361-1385	180 (59%)	102 (34%)	228 (75%)	**304**	41	13	21	9	836	48
1381-1405	320 (98%)	246 (75%)	319 (98%)	**326**	64	43	46	44	1021	47
1401-1425	393 (96%)	396 (97%)	407 (99%)	**410**	7	14	18	131	875	97
Total	**1114**	**812**	**1225**	**2208**	**225**	**81**	**122**	**365**	**15592**	**538**
Pointing 2										
Frequency Range (MHz)	Mock Galaxies				Real Sources			False Positives		
	SoFiA	MTO	V-Net	**Total**	SoFiA	MTO	V-Net	SoFiA	MTO	V-Net
1280-1304	2 (1%)	16 (8%)	33 (16%)	**212**	0	0	19	26	12324	538
1300-1324	32 (9%)	8 (2%)	95 (27%)	**356**	15	9	12	99	3287	40
1320-1344	20 (10%)	8 (4%)	62 (32%)	**193**	8	1	6	35	1766	26
1341-1364	99 (27%)	50 (14%)	200 (54%)	**368**	31	11	13	36	2057	60
1361-1385	193 (61%)	61 (19%)	257 (81%)	**317**	19	4	9	86	31	39
1381-1405	264 (94%)	219 (78%)	266 (95%)	**281**	85	56	47	70	2211	60
1401-1425	339 (97%)	344 (98%)	350 (100%)	**351**	6	15	22	78	1011	55
Total	**949**	**706**	**1263**	**2078**	**146**	**96**	**128**	**430**	**22687**	**818**

6.1.2 Evaluation

The real sources were removed from the detection catalogue of each source finding method in order to evaluate each method purely using the mock galaxies as the ground truth. The completeness, purity, volume score and combined score for each frequency range can be found in Figure 6.7, which were calculated following Section 5.4. From these plots it is clear that all the methods improve both their completeness and purity with frequency, as the spectral windows are closer to the observer and therefore contain seemingly brighter and larger sources. In addition, all the methods have very high volume scores above 0.8, demonstrating the high mask quality of them all.

When comparing the methods, there appears to be an alternation of the best performing method between V-Net and SoFiA, depending on the spectral window. MTO, on the other hand, while a close competitor with the other methods in its completeness and mask quality, indicated by the volume score, seems to struggle most with its purity. This is most likely due to the way in which the noise was modelled, allowing MTO to identify the noise as sources.

In addition, the 25th, 50th and 75th percentiles of the physical properties for the masked sources were investigated in order to compare the quality of the masks. In Figure 6.8, the spatial dimensions and the total flux of the masked mock galaxies detected by all the methods appear to be almost in agreement, only slightly veering away from each other and the ground truth at the lower frequencies. The masks of all the methods appear to also follow the ground truth when looking at the peak flux of the masked source, decreasing in similarity with distance and finally far over-estimating the peak flux at the most distant spectral window, as well as largely increasing in spread. The case is similar for the log of the H I mass, which becomes less and less correct for all three methods as the frequency decreases. However, the velocity width, elongation and asymmetry seem to be quite different for the detections of each method, all seeming to struggle to get similar values to the ground truth, with V-Net being the most accurate.

For most spectral windows, all the methods tend to overestimate the asymmetry and velocity width. However, MTO shows a much larger spread in these values. Note that the ground truth masks are not perfectly symmetric either due to the contributions from the noise. In addition, all the methods tend to overestimate the H I mass. Note that for all the over-estimations the difference between the detections and the ground truth values decrease at higher frequencies, again showing better mask quality for the more nearby spectral windows. Finally the elongation appears to be well masked by MTO and V-Net, both almost managing to follow the slight upwards trend with frequency, but SoFiA appears to underestimate this value. An example of a mock galaxy detected by each method for each spectral window can be seen in Figure 6.9.

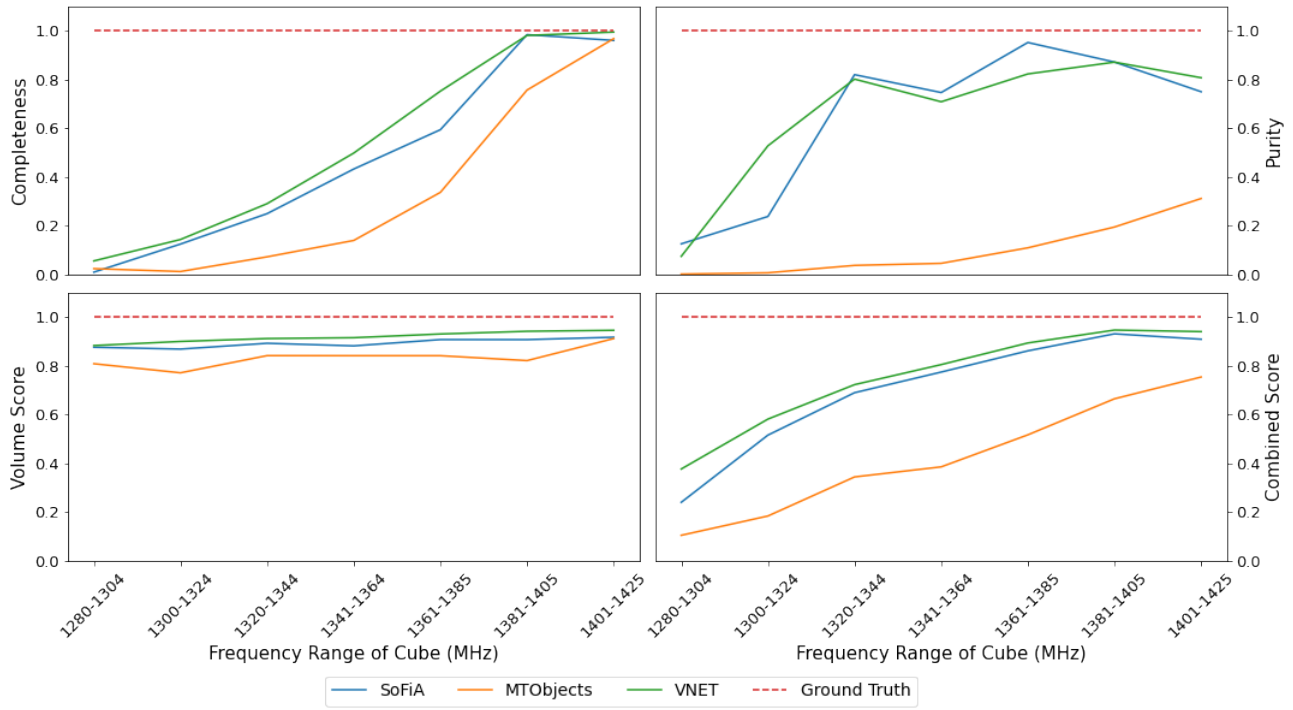


Figure 6.7: The various evaluation metrics of the three experiments evaluated with the mock galaxies as a function of the frequency range corresponding to each spectral window. For each plot, the results of SoFiA are shown in blue, the results of MTO in orange, the results of V-Net in green and the ground truth is shown as a dashed red line. The panel on the top left shows the completeness and the top right shows the purity. The mask quality is shown with the volume score in the bottom left plot, and the combination of all merging scores, purity and completeness are shown with the combined score in the bottom right plot. For all the plots the x-axis shows the frequency range associated with each spectral window of Pointing 1.

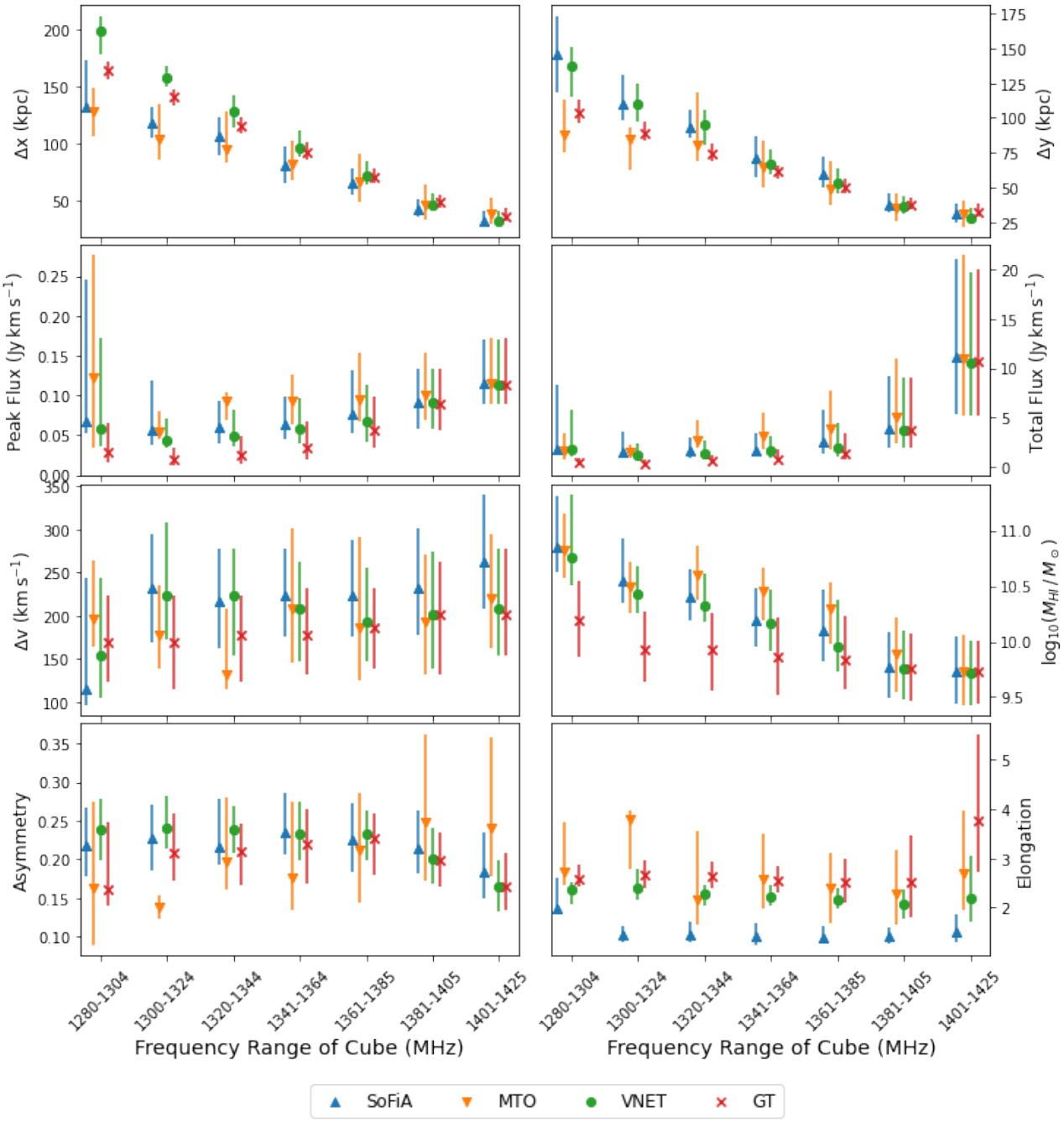


Figure 6.8: The 25th, 50th and 75th percentiles of the physical attributes of detected mock galaxies by each method per spectral window. For all the plots the x-axis shows the frequency range associated with each spectral window, and the markers indicate the median or 50th percentile. The blue triangles showing the detections of SoFiA, the orange triangles showing the detections of MTO, the green circles showing the detections of V-Net and the red crosses showing the ground truth masks. The lines below and above each marker indicate the 25th and 75th percentiles respectively. The top panel shows the two spatial dimensions Δx and Δy in kpc, while the second panel shows the peak flux and the total flux, the third panel shows the velocity width Δv in km/s and the log of the H I mass, and the last panel shows the asymmetry and spatial elongation.

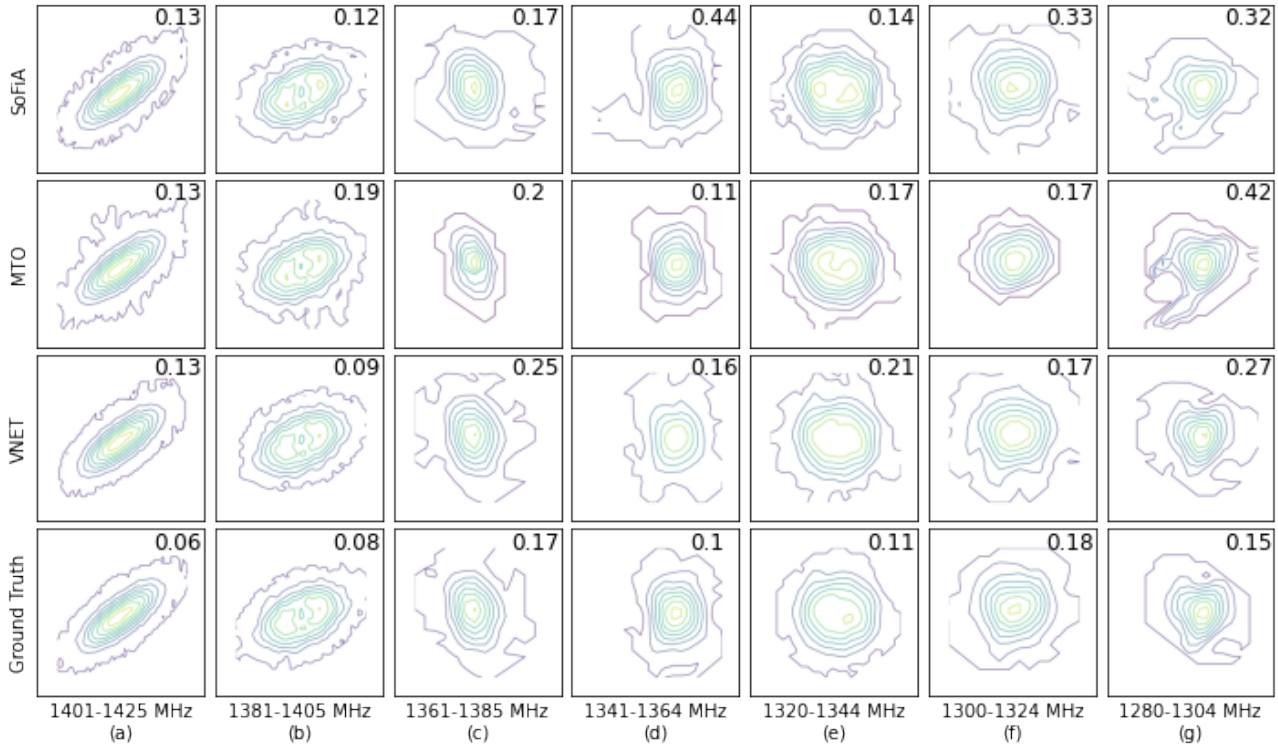


Figure 6.9: Examples of masked H I moment zero contours of the detections of mock galaxies for each method. From left to right shows an example from each spectral window of Pointing 1, with the mask from SoFiA, MTO, V-Net and the ground truth from top to bottom respectively. Note that the ground truth is not purely symmetric because it includes the noise within that mask. All the examples are matched across the methods by their brightest voxel. In the top right corner of each image is the degree of asymmetry for that masked source.

6.2 Experiment 2

In the second experiment, classical machine learning algorithms were used to find and remove false positives from the detection catalogue of each method, therefore improving their purity. In order to do this, all three source finding methods were run on the training set as well as the test set, using the cross-matching and removal of real sources approach of Experiment 1. The number of different types of detections for the spectral windows from Pointings 1 and 2 by each method can be seen in Table 6.2.

This set of training data was then re-sampled to obtain a better balance between the different methods, and used to train the four classical machine learning classification models. These models included a Decision Tree, a Random Forest, K-Nearest Neighbours and Support Vector Classification. Once the four machine learning models were run on the test set, the detections classified as false positives were removed from the catalogues and each of the three source finding methods were re-evaluated by assessing the purity and completeness. The resulting purity and completeness of each of the three source finding methods, after using these classifiers to post-process their detections, can be seen in Figure 6.10.

For all the four machine learning models there appears to be improvement in the purity of all three of the source finding methods, without doing much damage to the completeness. In addition, the re-sampling shows a huge improvement for all four machine learning models on the more distant spectral windows. This is particularly the case for MTO. While all four of the machine learning models were very close in competition, the main difference appears to be in the purity of SoFiA's results at the furthest and nearest spectral windows, see the first row of the second column. This same plot shows a clear struggle in the purity of the Decision Tree and SVC at the most distant spectral window. In addition, KNN appears to barely improve the purity at the nearest spectral window when comparing it to the dashed line representing the purity in Experiment 1. This leaves the Random Forest as the best performing model, only slightly above the rest for all the source finding methods.

In the most distant spectral window, all the machine learning models classified all of the detections of MTO as false positives. Therefore all the MTO detections from the most distant spectral window were removed from the evaluated catalogue, resulting in both a purity and

Table 6.3: The time taken for each machine learning model in Figure 6.10 to both train and infer whether a value in the catalogue of all the source finding methods was a true source or not.

Model	Before Re-sampling		After Re-sampling	
	Training Time	Inference Time	Training Time	Inference Time
DT	0.129 s	0.004 s	0.096 s	0.002 s
RF	2.763 s	0.036 s	0.096 s	0.002 s
SVC	30.999 s	1.253 s	0.096 s	0.002 s
KNN	0.028 s	0.048 s	0.011 s	0.054 s

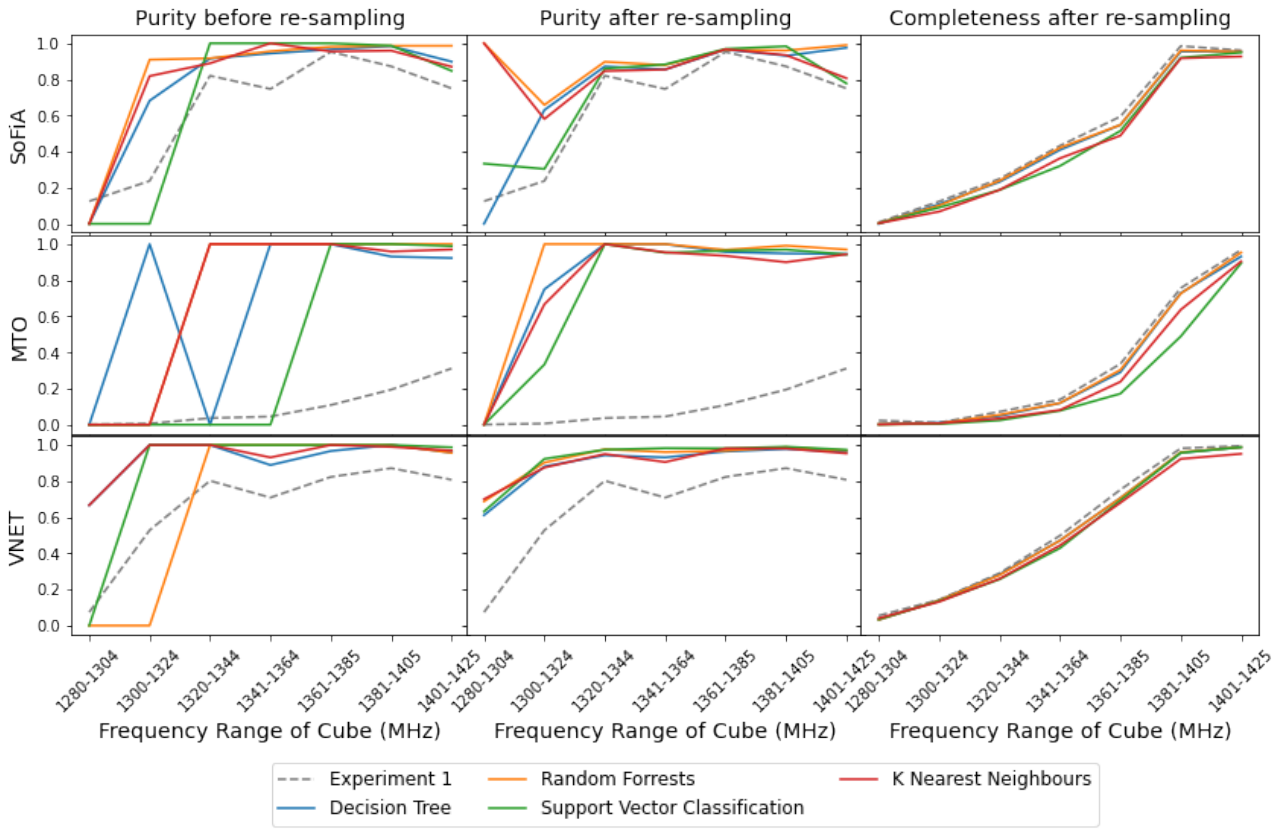


Figure 6.10: The purity and completeness of each source finding method after using machine learning classifiers to post-process their detections. The first two columns show the purity before and after re-sampling, while the completeness is shown in the last column. From top to bottom, the results of SoFiA, MTO and V-Net can be found. In all the plots, a grey dashed line shows the results prior to any post-processing, in other words the results from Experiment 1. The Decision Tree is represented by a blue line, the Random Forest by an orange line, the Support Vector Classification by a green line and the K-Nearest Neighbor model by a red line. For all the plots the x-axis shows the frequency range associated with each spectral window from Pointing 1.

Table 6.4: The importance of each feature determined by the Random Forest classifier.

Feature	Random Forest Importance
Volume (voxels)	0.279
Δv ($\mathrm{km\,s^{-1}}$)	0.268
Total Flux ($\mathrm{Jy\,km\,s^{-1}}$)	0.144
Peak Flux ($\mathrm{Jy\,km\,s^{-1}}$)	0.087
Elongation	0.082
Δy (kpc)	0.077
Δx (kpc)	0.063

completeness of zero. Evaluating the remaining spectral windows using this post-processing approach for the top performing models now puts MTO in the lead in terms of purity. Closely following is V-Net and then SoFiA, both of which still benefited from this post-processing. Note that the time taken to train and use these models for post-processing can be found in Table 6.3 and takes no more than fractions of a second for all of them.

The features and their importance determined by the Random Forest, modelled on the re-sampled catalogue, can be found in Table 6.4. The volume in terms of voxels and the velocity width in $\mathrm{km\,s^{-1}}$ both appear to be the most important features, reflecting the importance of the physical size of the sources. This also explains the low ranking of the spatial dimensions, as they give repetitive information that the top ranking features already give. This is followed by the total and peak fluxes of the sources in $\mathrm{Jy\,km\,s^{-1}}$. This gives useful insight into how to better inform rule based source finding methods such as MTO.

Once the best of the models was applied, namely the Random Forest on the re-sampled data, and the flagged false positives were removed and the resulting number of sources detected by each method were summarized in a Venn diagram (Suppes, 1957) in Figure 6.11. Comparing this to the first plot in Figure 6.5, it shows the drastic improvement by decreasing the number of false positive detections in all methods, as expected. However, this comes at the price of a decrease in the number of real sources detected. With this post-processing, V-Net overtakes SoFiA with less false detections, making V-Net the winning method in terms of completeness, except for its low detection rate of real sources.

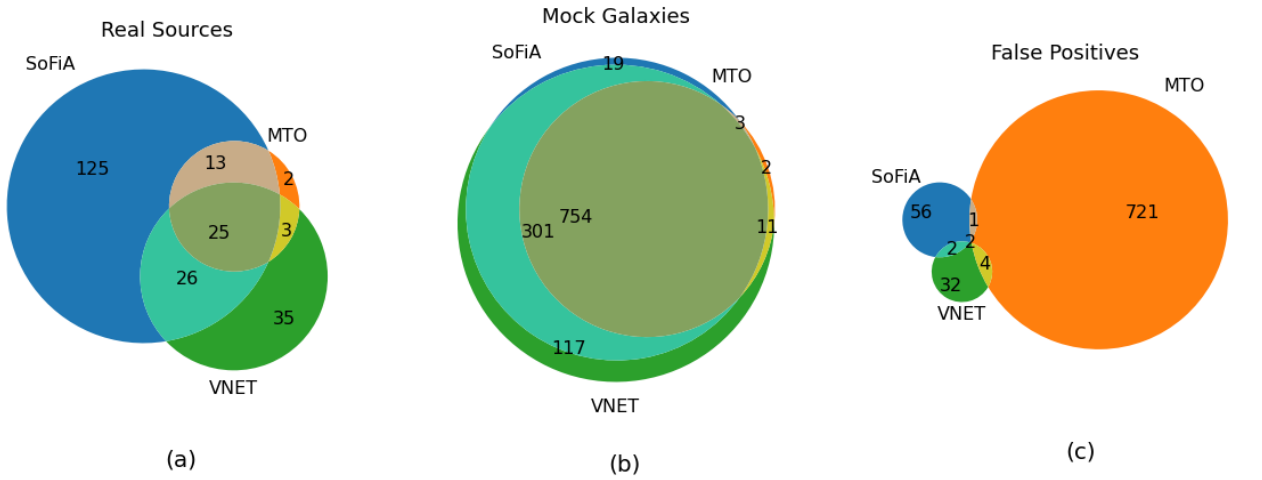


Figure 6.11: Venn diagrams of the number of each kind of detection by each method in Experiment 2. (a) Shows the number of real sources detected, (b) shows the number of mock galaxies detected and (c) shows the number of false positives detected. SoFiA is represented by the blue circle, V-Net is represented by the green circle and MTO is represented by the orange circle.

6.3 Experiment 3

The final experiment consisted of two sub-experiments where the real sources detected in Experiment 1 were used to improve the V-Net network. In Experiment 3.1 the 3D CNN was retrained from scratch using the mock galaxy and real source masks as the ground truth. Experiment 3.2 only used the real masks as the ground truth to re-train only the last layer of the network. The learning progress of both sub-experiments can be seen in Figure 6.12, with both the training and validation loss scores as a function of epoch. Both models were trained up to seven epochs, with Experiment 3.1 taking 21 days and Experiment 3.2 taking 10.5 days to train. The loss score already indicates that the model trained from scratch will outperform the transfer learning model.

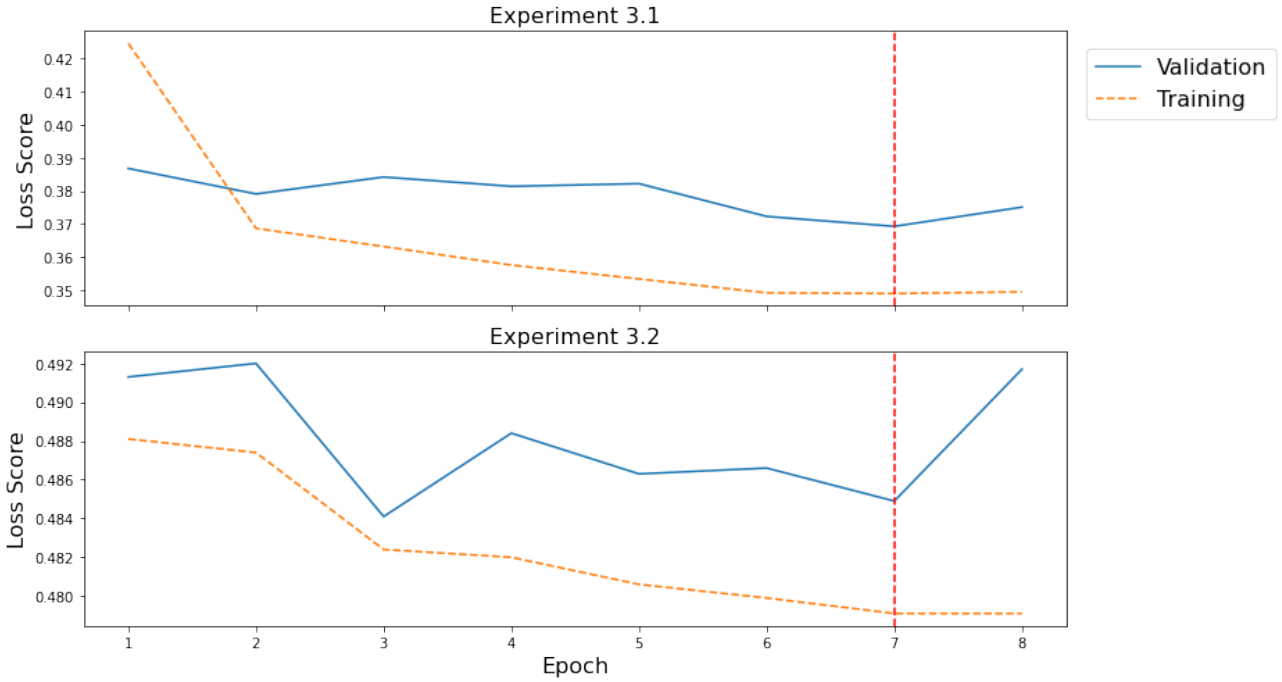


Figure 6.12: The training and validation loss scores of the re-trained and fine-tuned 3D CNN per epoch during the training process. The blue line shows the validation loss score while the orange line shows the training loss score, with the red dashed line indicating the point where the model was taken for inference. The top panel shows the loss scores of Experiment 3.1 and the bottom panel shows the loss scores of Experiment 3.2.

Both models were used to locate and mask sources in the spectral windows of Pointing 1 containing both real sources and inserted mock galaxies. The real sources and mock galaxies detected were evaluated separately by comparing the results to the catalogued ground truth of the real sources and mock galaxies separately. To improve the purity of the results, they were post-processed using the Random Forest classifier from Experiment 2. The catalogues were again over-sampled to ensure that the number of real sources was equal to the number of mock galaxies and prevent any bias in the post-processing model. The resulting evaluation metrics for the models after the post-processing can be found in Figure 6.13, where they are compared with the network trained in Experiment 1.

As shown in Figure 6.13, there is a huge improvement in the completeness, seen in the upper left plot, when compared to that of Experiment 1, shown in red. Particularly for the real sources, indicated by the dashed lines, detected by both the re-trained and fine-tuned V-Net networks, shown in blue and turquoise respectively. However, this comes at the expense of the purity, seen in the upper right plot, which drops significantly for both real sources and mock galaxies with the newly trained networks, even after applying the Random Forest classifier. While the mask quality appears to barely change for either, the combined score shows that Experiment 3.1 is the best V-Net network for finding real sources. Due to the lack of purity and a slight drop in mock mask quality, Experiment 3.2 ranked lowest of the three V-Net networks. It is important to note that the completeness of the real sources for all three experiments remained

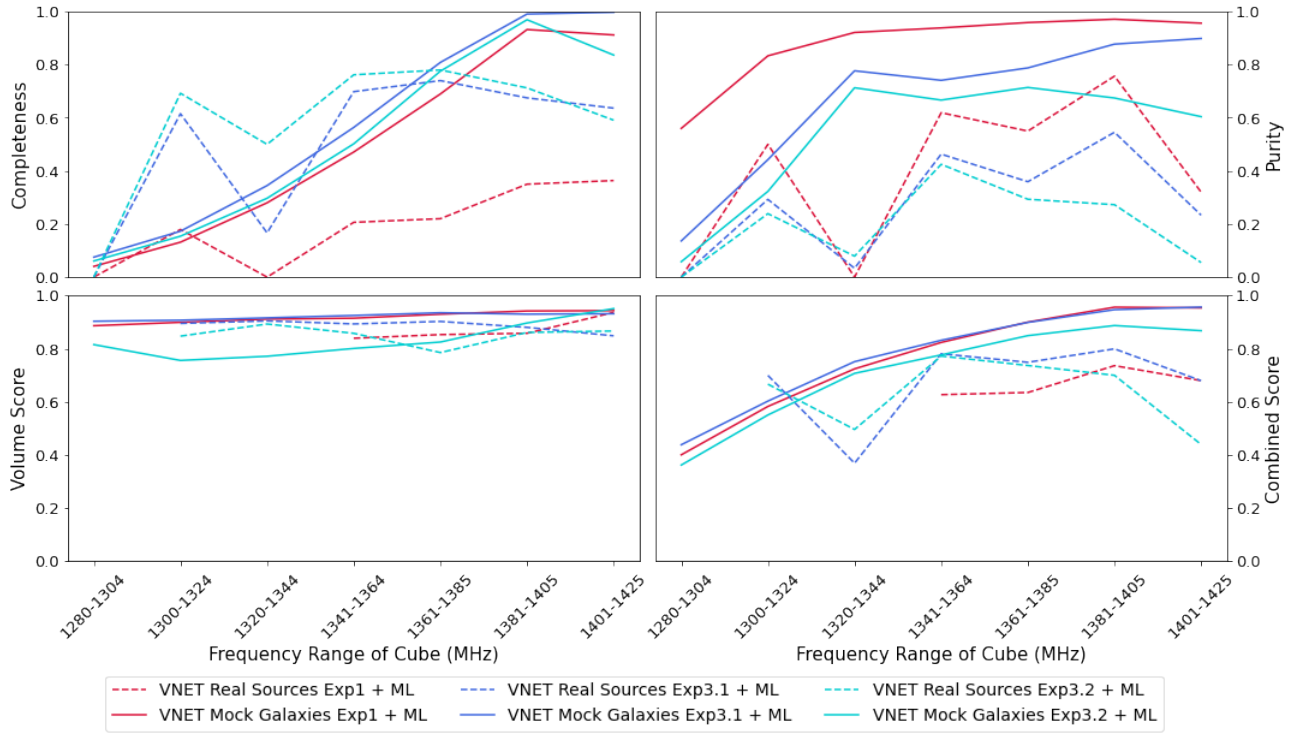


Figure 6.13: The various evaluation metrics of the three V-Net networks, combined with a Random Forest classifier, evaluated with the mock galaxies and real sources as a function of the frequency range of each spectral window from Pointing 1. For each plot, results from the V-Net network in Experiment 1 are shown in red, results from the re-trained V-Net network in Experiment 3.1 are shown in blue and results from the fine-tuned V-Net network in Experiment 3.2 are shown in turquoise. For all the networks from each experiment, the dotted line indicates the metric for the real sources while the solid line indicates the metric for the mock galaxies. The panel on the top left shows the completeness and the top right shows the purity of the samples. The mask quality is shown with the volume score in the bottom left plot, and the combination of all three above metrics are shown with the combined score in the bottom right plot. For all the plots the x-axis shows the frequency range associated with spectral window of Pointing 1.

unchanged after applying the Random Forest classifier, ensuring that the post-processing was not filtering out any real sources.

6.4 Comparing Experiments

For a final comparison of all the pipelines proposed in each experiment, the mean combined score across the spectral windows was calculated for each. It is also important to note that for the two closest spectral windows, the mock galaxies do not include the fainter low H I mass sources and therefore the mock score is not representative of real galaxy source finding at these spectral windows. However, since this is only the case for the few closest spectral windows, they contribute less to the mean score than the rest of all the other spectral windows.

Separating the metric by real sources and mock galaxies, the top ranking method for finding real sources was found to be SoFiA combined with the Random Forest classifier, while the top ranking method for finding mock galaxies was the V-Net network in Experiment 3.1 combined with the Random Forest classifier. Therefore, evaluating the pipelines with all the real sources and mock galaxies together, resulted in the V-Net network in Experiment 3.1 combined with the Random Forest classifier as the top ranking method. However, since the ground truth catalogue contains many more mock galaxies than real sources, a better representative metric is to consider the mean of the real and mock combined scores. With this approach, SoFiA combined with the Random Forest classifier was found to be the best ranking of them all.

Table 6.5: The mean combined scores of all source finding pipelines. The Total score represents the ability of the method to find all the real sources and mock galaxies, while the Mean score represents the mean of the Mock and Real scores. For all V-Net networks, the experiment is indicated to show which model it is and the methods with + RF indicate that the post-processing using a Random Forest classifier was used.

	Real	Mock	Total	Mean
V-Net Experiment 3.1 + RF	0.577	**0.774**	**0.799**	0.676
V-Net Experiment 1 + RF	0.472	0.763	0.773	0.610
V-Net Experiment 1	0.449	0.752	0.764	0.600
SoFiA	0.651	0.703	0.747	0.677
SoFiA + RF	**0.689**	0.684	0.730	**0.686**
V-Net Experiment 3.2 + RF	0.507	0.691	0.725	0.599
V-Net Experiment 3.1	0.398	0.678	0.703	0.538
MTO + RF	0.317	0.553	0.563	0.435
MTO	0.201	0.421	0.420	0.311
V-Net Experiment 3.2	0.204	0.348	0.378	0.276

Chapter 7

Discussion

In the search for the optimal pipeline to locate and mask H I sources in a 3D H I emission cube, three source finding methods were compared, namely SoFiA, MTO and a 3D CNN using the V-Net architecture. In addition, a post-processing classical machine learning step was used in attempt to filter out the false positives of each of these source finding methods. Two pointings of H I emission data taken from the Apertif medium-deep survey were used for this project. Seven spectral windows from Pointing 2, each corresponding to a different frequency range, were used for training purposes and another seven from Pointing 1 with equivalent frequency ranges were used for testing. Since the data used came from a single telescope, the ability to generalise these results would need to be tested on data from a variety of telescopes, each with their own resolution, sensitivity and noise characteristics.

Due to the lack of labelled data, mock galaxies were first inserted into existing H I emission cubes and then the results of each source finding method were cross-matched with optical catalogs (Chambers et al., 2019) to label real sources and add their masks to the ground truth. While this enabled an evaluation of the methods with both real sources and mock galaxies, manually finding the real sources during the cross-matching process was very time consuming and still resulted in an over-representation of the mock galaxies relative to the real sources. This indicates that this approach to labelling the ground truth was not the most effective. The over-representation of mock galaxies not only biased the training of the V-Net network but also could have biased the evaluation of all the pipelines for more symmetric, bright sources. In addition, the mock galaxies introduced a bias towards larger mass galaxies due to the scaling by a factor of ten. However, as previously mentioned, this only affects the source finding in the more nearby spectral windows. A final limitation to the data used was the partial labelling of the data-set, since there is no guarantee that all the real sources existing prior to the mock galaxy insertion were labelled.

For the training of the post-processing machine learning models, the use of re-sampling to balance the number of more distant detections by MTO showed improvement for all the machine learning models at lower frequency ranges. The re-sampling showed an increase in purity for all the source finding methods combined with the Random Forest classifier by at least 60%. Among the traditional machine learning algorithms tested for this post-processing step, the

Random Forest classifier was found to be the model that best improved the purity of all the source finding methods, without compromising their completeness. This was expected, as it is known to be the best performing algorithm for classification (Fernandez-Delgado et al., 2014) due to its use of cross-validation and its robustness to outliers.

The top features ranked according to the Random Forest classifier were found to be the velocity width and the total volume in voxels, demonstrating the importance of the physical size of the detections for their reality. This could help inform the filtering process in rule-based methods like MTO. An additional commonality across all source finding methods was the improvement in both their completeness and purity with increasing frequency. This was also expected as the higher frequency spectral windows are more nearby and therefore the sources appear brighter and larger. As a consequence, at the most distant spectral window with a frequency range of 1280-1304 MHz, all the source finding methods reached an unusable level of completeness.

Investigating the physical properties of the masked mock galaxies detected, the V-Net network was found to yield the closest values to the ground truth and outperform the other source finding methods at masking the mock galaxies. This suggests the potential for the V-Net network to outperform SoFiA overall, with an even better mask quality, should it be trained on more real sources. For all the methods the difference between the physical properties of the masked detected sources and the ground truth values for all the physical properties tend to decrease at higher frequencies, which shows that the mask quality is in fact better for more nearby spectral windows.

A more detailed investigation of the false positives detected showed the different failings of each of the source finding methods. SoFiA tended to pick up more regions with higher local noise, MTO picked up more extended and very small artifacts and the V-Net network favoured extended and large round artifacts. The very few overlaps in false positive detections between the methods was expected since the different approaches make each source finding method vulnerable to different types of artifacts and noise peaks.

Although MTO struggled with a low purity on its own, this was expected since Arnoldus (2015) used the reliability filter from SoFiA to tackle this and remove artifacts and noise detections. Since the use of the reliability filter was found to decrease the completeness, in this project the purity was instead improved by adding a Random Forest classifier. This approach even further improved the purity of MTO such that it outperformed the other source finding methods with little to no effect on the completeness. The poor purity of MTO on its own is most likely due to the way in which the background is estimated, which is not suitable for radio data with a mean near zero. This noise modelling could be the cause of the struggle of MTO to differentiate sources from the noise. However, since this is resolved with the Random Forest classifier, perhaps implementing the RF classifier within the max-tree would make the purity of MTO competitive on its own.

MTO was found to have a high mask quality with a volume score consistently above 80% and despite being the lowest scoring in this regard, it was never more than 10% lower than the other source finding methods. While the completeness of MTO was not necessarily expected to overtake SoFiA, it was not as competitive in this regard as found in Arnoldus (2015), reaching

up to 30% difference at some spectral windows when compared to SoFiA. This could be caused by the lack of spectral smoothing since only the spatial dimensions were smoothed anisotropically during the pre-processing. As Arnoldus (2015) suggests, this could perhaps be improved with some experimentation of the spectral smoothing, including the use of isotropic smoothing.

SoFiA, combined with the Random Forest classifier, was found to be the best ranking when evaluated out of the real source detections. The overall mean combined score of both the real sources and mock galaxies also ranked the combination of SoFiA and the RF classifier as the best overall. Since SoFiA is the most widely used and optimized source finding tool for H I emission data, it makes sense that of all the source finding methods it is most suited to the data-set used. This is perhaps due to its ability to perform local noise scaling as well as its reliability filter which reduces the false detection of noise and artifacts. Of all the methods, it also takes the least amount of time to run and with the addition of a Random Forest classifier it even further improves in purity.

The V-Net network was found to be the slowest source finder due to the need to run the inference on each of the sliding windows sequentially and stitch them together. It is also important to point out the large amount of computational resources and time used in the training process of the network in each experiment. Therefore, it is suggested that in future work the sliding window is applied in parallel or the architecture is modified to take larger inputs, since both would reduce the time spent on the training and inference.

Ranking the mean combined scores of each source finding method, the 3D CNN with the V-Net architecture was found to be the best at finding the mock galaxies when combined with a Random Forest classifier. It is also important to note its advantage of being less sensitive to input parameters than the other source finding methods. However, it did not perform best at finding real sources, which could be due to its struggle to correctly mask asymmetric sources, such as those containing ram pressure stripping or interacting binaries. This is likely due to the fact that it was trained primarily on symmetric mock galaxies and could be over-fitting to them (O'Mahony et al., 2019).

When looking at the values of the mean combined scores for mock galaxies and real sources, the V-Net network ranked as a very close third best, scoring only 1% lower than SoFiA with the RF classifier. In addition, re-training of the network with some real sources and not only mock galaxies, showed an increase in its ability to find real sources by about 20%. This shows the potential for a deep learning approach like the V-Net network to eventually outperform SoFiA if it were to be trained on more real sources. Alternatively, to remove the difficulty of not enough labelled real sources, there is potential to investigate an unsupervised deep learning solution.

Chapter 8

Conclusion

Due to the improvement in technology, the amount of data coming from astronomical surveys continues to increase, and therefore so does the need for fast and accurate techniques to detect and characterise sources. The challenge lies in the lack of clarity in the boundaries of sources, with many having intensities very close to the noise, especially in the case of radio data. This project therefore aimed to find the best method for finding the most sources and least artifacts with the best mask quality in three-dimensional neutral hydrogen cubes.

This was done by testing two existing statistical methods, SoFiA and MTObjects, and a well-known medical imaging 3D CNN architecture called V-Net. In addition, classical machine learning models were used as a post-processing step for the resulting catalogs of each method. Two pointings of H I emission data taken from the Apertif medium-deep survey were used for this project. Seven spectral windows from Pointing 2, each corresponding to a different frequency range, were used for training purposes and another seven from Pointing 1 with equivalent frequency ranges were used for testing. Due to the lack of labelled data, 4286 mock galaxies were inserted into the fourteen spectral windows and used as the ground truth. In addition, the results of each method were cross-matched by eye with optical catalogs to add 571 real sources to the ground truth.

The recommended source finder out of all the methods tested in this project was SoFiA combined with a Random Forest (RF) classifier. This is due to this pipelines ability to best find and mask not only mock galaxies but also real sources, as well as the small amount of time and computational resources needed relative to the other methods. The addition of the Random Forest classifier into the built-in software of SoFiA could improve ease of use of this post-processing step. In addition, SoFiA is publicly accessible and the Random Forest classifier trained in this project will also be made available. SoFiA, however, breaks down at the lowest frequency range of 1280-1304 MHz, the limit at which all the source finding methods eventually reached an unusable completeness. This limit could also be investigated further by repeating the experiments with the adjustment of the signal-to-noise ratio of the mock galaxies before inserting them to see how this affects the source finders.

MTObjects was found to be competitive with the other source finding methods in most re-

gards and even ranked the highest in terms of purity when a Random Forest classifier was applied to its catalogue. However, its low completeness leaves room for future improvements. Further experimentation of different spectral smoothing techniques, anisotropic or isotropic, could aid an increase in its completeness. On its own, MTO suffers from a low purity due to the way in which the background is estimated. Either this could be modified to better suit H I emission data or a Random Forest classifier could be built into the max-tree to ensure the removal of artifacts and noise detections.

It is important to note that V-Net, trained with both real sources and mock galaxies, combined with a Random Forest classifier was found to be a competitive second best source finder, scoring only 1% lower than SoFiA when combined with the RF classifier. While the V-Net network followed by the RF classifier was found to be the best at finding mock galaxies, for which it was trained, it fell short with real sources due to its inability to correctly mask asymmetry. This suggests the potential of the network to improve with the use of more labelled real sources in future training. An additional improvement of the network that could be explored in future work would be the use of the combined score as the loss function during training. Future work could also explore the minimum amount of data needed to get the same performance as well as the use of parallel programming, both of which could improve the time and resources needed for training the network. Finally, there is also potential to explore different 3D CNN architectures for similar H I source finding tasks as well as experimenting with unsupervised deep learning approaches in the future.

As the everyday applications of computer vision continue to increase, so does the amount of astronomical data produced by ever-improving telescopes. With this progress, grows the need for astronomers to keep up-to-date with the advances of computer vision. In this project a computer vision solution was applied to an astronomical task and found to be competitive with the traditional tools. In addition, with the implementation of suggested improvements it is likely to even overtake the traditional astronomy methods. This demonstrates that the interchangeability of solutions between the two fields will become fundamental to the growth of data analysis in future astronomy.

References

Agarap, A. (2018), 'Deep learning using rectified linear units (relu)', *arXiv preprint arXiv:1803.08375* .

Ahmed, E. et al. (2018), 'A survey on deep learning advances on different 3d data representations', *arXiv e-prints* .

Alamri, S., Kalyankar, N. and Khamitkar, S. (2010), 'Image segmentation by using edge detection', *International Journal on Computer Science and Engineering* **2**.

Alhassan, W., Taylor, A. and Vaccari, M. (2018), 'The first classifier: compact and extended radio galaxy classification using deep convolutional neural networks', *Monthly Notices of the Royal Astronomical Society* **480**(2), 2085–2093.

Analyttica Datalab (2018), 'Gini coefficient or gini index in our data science analytics platform'.

Andreon, S. et al. (2000), 'Wide field imaging - i. applications of neural networks to object detection and star/galaxy classification', *Monthly Notices of the Royal Astronomical Society* **319**(3), 700–716.

Aniyan, A. and Thorat, K. (2017), 'Classifying radio galaxies with the convolutional neural network', *The Astrophysical Journal Supplement Series* **230**(2), 20.

Apertif Science Team (2016), 'Apertif survey plan ii'. (Accessed on 06/30/2021).

Aptoula, E., Lefèvre, S. and Collet, C. (2006), Mathematical morphology applied to the segmentation and classification of galaxies in multispectral images, *in* '2006 14th European Signal Processing Conference', pp. 1–5.

Argueso, F., Sanz, J., Barreiro, R., Herranz, D. and Gonzalez-Nuevo, J. (2006), 'Statistical analysis of undetected point sources in cosmic microwave background maps', *Monthly Notices of the Royal Astronomical Society* **373**(1), 311–320.

Arnoldus, C. (2015), A max-tree-based astronomical source finder, PhD thesis, University of Groningen.

Astropy Collaboration (2018), 'The astropy project: Building an open-science project and status of the v2.0 core package', *Astronomical Journal* **156**(3), 123.

Bandara, R. (2018), 'Image segmentation using unsupervised watershed algorithm with an over-segmentation reduction technique', *arXiv e-prints* p. arXiv:1810.03908.

Beck, M. et al. (2018), 'Integrating human and machine intelligence in galaxy morphology classification tasks', *Monthly Notices of the Royal Astronomical Society* **476**(4), 5516–5534.

Bertin, E. and Arnouts, S. (1996), 'Sextractor: Software for source extraction.', *Astronomy & Astrophysics Supplement Series* **117**, 393–404.

Bochkovskiy, A., Wang, C. and Liao, H. (2020), 'Yolov4: Optimal speed and accuracy of object detection', *arXiv e-prints* .

Booth, R., de Blok, W., Jonas, J. and Fanaroff, B. (2009), 'Meerkat key project science, specifications, and proposals', *arXiv e-prints* .

Breiman, L., Friedman, J., Stone, C. and Olshen, R. (1984), *Classification and Regression Trees*, Chapman and Hall/CRC.

Brigham, E. and Morrow, R. (1967), 'The fast fourier transform', *IEEE Spectrum* **4**(12), 63–70.

Broeils, A. and Rhee, M. (1997), 'Short 21-cm wsrt observations of spiral and irregular galaxies. hi properties.', *Astronomy & Astrophysics* **324**, 877–887.

Burke, C., Aleo, P., Chen, Y., Liu, X., Peterson, J., Sembroski, G. and Lin, J. (2019), 'Deblending and classifying astronomical sources with mask r-cnn deep learning', *Monthly Notices of the Royal Astronomical Society* **490**(3), 3952–3965.

Cappellari, M. et al. (2011), 'The atlas 3d project - i. a volume-limited sample of 260 nearby early-type galaxies: science goals and selection criteria', *Monthly Notices of the Royal Astronomical Society* **413**(2), 813–836.

Çiçek, Ö., Abdulkadir, A., Lienkamp, S., Brox, T. and Ronneberger, O. (2016), '3d u-net: Learning dense volumetric segmentation from sparse annotation', *arXiv e-prints* p. arXiv:1606.06650.

Chambers, K. et al. (2019), 'The pan-starrs1 surveys', *arXiv e-prints* .

Chawla, N., Bowyer, K., Hall, L. and Kegelmeyer, W. (2002), 'Smote: Synthetic minority over-sampling technique', *Journal of Artificial Intelligence Research* **16**, 321–357.

Cheng, T. et al. (2020), 'Optimizing automatic morphological classification of galaxies with machine learning and deep learning using dark energy survey imaging', *Monthly Notices of the Royal Astronomical Society* **493**(3), 4209–4228.

Christ, P. et al. (2016), Automatic liver and lesion segmentation in ct using cascaded fully convolutional neural networks and 3d conditional random fields, *in* S. Ourselin, L. Joskowicz, M. Sabuncu, G. Unal and W. Wells, eds, 'Medical Image Computing and Computer-Assisted Intervention – MICCAI 2016: 19th International Conference, Athens, Greece, October 17-21, 2016, Proceedings, Part II', Springer International Publishing, Cham, pp. 415–423.

Cunningham, P. and Delany, S. (2007), 'K-nearest neighbour classifiers'.

D'Agostino, R., Belanger, A. and D'Agostino, J. (1990), 'A suggestion for using powerful and informative tests of normality', *The American Statistician* **44**(4), 316–321.

De Boer, D. et al. (2009), 'Australian ska pathfinder: A high-dynamic range wide-field of view survey telescope', *Proceedings of the IEEE* **97**(8), 1507–1521.

Deepthi, A. (2019), 'Knn visualization in just 13 lines of code'. (Accessed on 12/07/2021).

Dubath, P. et al. (2011), 'Random forest automated supervised classification of hipparcos periodic variable stars', *Monthly Notices of the Royal Astronomical Society* **414**(3), 2602–2617.

Dumoulin, V. and Visin, F. (2016), 'A guide to convolution arithmetic for deep learning', *arXiv e-prints* p. arXiv:1603.07285.

Fernandez-Delgado, M., Cernadas, E., Barro, S. and Amorim, D. (2014), 'Do we need hundreds of classifiers to solve real world classification problems?', *Journal of Machine Learning Research* **15**.

Flöer, L., Giese, N., Jurek, R., Meyer, M., Popping, A., Serra, P. andWestmeier, T. and Winkel, B. (2021), 'Sofia 2'.

Gandhi, R. (2018), 'Support vector machine — introduction to machine learning algorithms'. (Accessed on 12/07/2021).

Gao, B. and Pavel, L. (2017), 'On the properties of the softmax function with application in game theory and reinforcement learning', *arXiv e-prints* .

Gheller, C., Vazza, F. and Bonafede, A. (2018), 'Deep learning based detection of cosmological diffuse radio sources', *Monthly Notices of the Royal Astronomical Society* **480**(3), 3749–3761.

Gogate, A. et al. (2020), 'Budhies iv: Deep 21-cm neutral hydrogen, optical, and uv imaging data of abell 963 and abell 2192 at z $\simeq$ 0.2', *Monthly Notices of the Royal Astronomical Society* **496**(3), 3531–3552.

Haigh, C., Chamba, N., Venhola, A., Peletier, R., Doorenbos, L., Watkins, M. and Wilkinson, M. (2021), 'Optimising and comparing source-extraction tools using objective segmentation quality criteria', *Astronomy & Astrophysics* **645**, A107.

He, K., Zhang, X., Ren, S. and Sun, J. (2015), 'Delving deep into rectifiers: Surpassing human-level performance on imagenet classification', *arXiv e-prints* p. arXiv:1502.01852.

Ho, T. (1995), Random decision forests, *in* 'Proceedings of 3rd International Conference on Document Analysis and Recognition', Vol. 1, pp. 278–282.

Hogbom, J. and Brouw, W. (1974), 'The synthesis radio telescope at westerbork. principles of operation, performance and data reduction', *Astronomy & Astrophysics* **33**, 289.

IBM Cloud Education (2020), 'Convolutional neural networks'. (Accessed on 12/07/2021).

Igel, C. and Hüsken, M. (2000), 'Improving the rprop learning algorithm'.

iraf.net (n.d.), 'Advanced search - iraf.net'.

Jarvis, J. and Tyson, J. (1981), 'Focas: faint object classification and analysis system.', *Astronomical Journal* **86**, 476–495.

Jurek, R. (2012), 'The characterised noise hi source finder: Detecting hi galaxies using a novel implementation of matched filtering', *Publications of the Astronomical Society of Australia* **29**(3), 251–261.

Kaur, D. and Kaur, Y. (2014), 'Various image segmentation techniques: A review', *International Journal of Computer Science and Mobile Computing* **3**(5), 809–814.

Kingma, D. and Ba, J. (2017), 'Adam: A method for stochastic optimization', *arXiv e-prints* .

Kügler, S., Polsterer, K. and Hoecker, M. (2015), 'Determining spectroscopic redshifts by using k nearest neighbor regression. i. description of method and analysis', *Astronomy & Astrophysics* **576**, A132.

Kuiper, N. (1960), Tests concerning random points on a circle, *in* 'Nederl. Akad. Wetensch. Proc. Ser. A', Vol. 63, pp. 38–47.

Lemaître, G., Nogueira, F. and Aridas, C. K. (2017), 'Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning', *Journal of Machine Learning Research* **18**(17), 1–5.

Levine, M. and Nazif, A. (1981), *An Experimental Rule-based System for Testing Low Level Segmentation Strategies*, McGill Univ.

Li, Z., Ding, Q. and Zhang, W. (2011), A comparative study of different distances for similarity estimation, *in* R. Chen, ed., 'Intelligent Computing and Information Science', Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 483–488.

Lintott, C. et al. (2008), 'Galaxy zoo: morphologies derived from visual inspection of galaxies from the sloan digital sky survey', *Monthly Notices of the Royal Astronomical Society* **389**(3), 1179–1189.

Long, J., Shelhamer, E. and Darrell, T. (2015), 'Fully convolutional networks for semantic segmentation'.

Lukic, V., De Gasperin, F. and Brüggen, M. (2019), 'Convosource: Radio-astronomical source-finding with convolutional neural networks', *Galaxies* .

Maddox, N. et al. (2021), 'Mightee-hi: The h i emission project of the meerkat mightee survey', *Astronomy & Astrophysics* **646**, A35.

Mark, J. (1996), Introduction to radial basis function networks.

Martin, A., Papastergis, E., Giovanelli, R., Haynes, M., Springob, C. and Stierwalt, S. (2010), 'The arecibo legacy fast alfa survey. x. the h i mass function and Ω_h i from the 40% alfalfa survey', *Astrophysical Journal* **723**(2), 1359–1374.

Martinsson, T., Verheijen, M., Bershady, M. A., Westfall, K., Andersen, D. and Swaters, R. (2016), 'The diskmass survey. x. radio synthesis imaging of spiral galaxies', *Astronomy & Astrophysics* **585**, A99.

Masias, M., Freixenet, J., Lladó, X. and Peracaula, M. (2012), 'A review of source detection approaches in astronomical images', *Monthly Notices of the Royal Astronomical Society* **422**(2), 1674–1689.

Meyer, M. et al. (2017), 'Tracing hi beyond the local universe', *Publications of the Astronomical Society of Australia* **34**, 52.

Milletari, F., Navab, N. and Ahmadi, S. (2016), 'V-net: Fully convolutional neural networks for volumetric medical image segmentation', *Proceedings - 2016 4th International Conference on 3D Vision, 3DV 2016* pp. 565–571.

Moschini, U. (2016), Efficient morphological tools for astronomical image processing, PhD thesis, University of Groningen.

Moschini, U., Meijster, A. and Wilkinson, M. (2018), 'A hybrid shared-memory parallel max-tree algorithm for extreme dynamic-range images', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **40**(3), 513–526.

Nikolaos, A. (2019), Deep learning in medical image analysis: a comparative analysis of multi-modal brain-mri segmentation with 3d deep neural networks, Master's thesis, University of Patras.

Noordermeer, E., van der Hulst, J., Sancisi, R., Swaters, R. and van Albada, T. (2005), 'Vizier online data catalog: Hi observations of whisp disk galaxies', *VizieR Online Data Catalog* pp. J/A+A/442/137.

O'Mahony, N., Campbell, S., Carvalho, A., Harapanahalli, S., Velasco-Hernández, G., Krpalkova, L., Riordan, D. and Walsh, J. (2019), 'Deep learning vs. traditional computer vision', *CoRR* **abs/1910.13796**.

Ouzounis, G. and Wilkinson, M. (2007), 'Mask-based second-generation connectivity and attribute filters', *IEEE transactions on pattern analysis and machine intelligence* **29**(6), 990–1004.

Pan, S. and Yang, Q. (2010), 'A survey on transfer learning', *IEEE Transactions on Knowledge and Data Engineering* **22**(10), 1345–1359.

Paszke, A. et al. (2019), Pytorch: An imperative style, high-performance deep learning library, *in* H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox and R. Garnett, eds, 'Advances in Neural Information Processing Systems 32', Curran Associates, Inc., pp. 8024–8035.

Pedregosa, F. et al. (2011), 'Scikit-learn: Machine learning in Python', *Journal of Machine Learning Research* **12**, 2825–2830.

Perona, P. and Malik, J. (1990), 'Scale-space and edge detection using anisotropic diffusion', *IEEE Transactions on Pattern Analysis and Machine Intelligence* **12**(7), 629–639.

Persic, M., Salucci, P. and Ste, F. (n.d.).

Ponomareva, A., Verheijen, M. and Bosma, A. (2016), 'Detailed h i kinematics of tully-fisher calibrator galaxies', *Monthly Notices of the Royal Astronomical Society* **463**(4), 4052–4067.

Popping, A., Jurek, R., Westmeier, T., Serra, P., Flöer, L., Meyer, M. and Koribalski, B. (2012), 'Comparison of potential askap hi survey source finders', *Publications of the Astronomical Society of Australia* **29**(3), 318–339.

Punzo, D., van der Hulst, J., Roerdink, J., Oosterloo, T., Ramatsoku, M. and Verheijen, M. (2015), 'The role of 3-d interactive visualization in blind surveys of hi in galaxies', *Astronomy and Computing* .

Purva, H. (2020), 'Bias and variance in machine learning – a fantastic guide for beginners!'. (Accessed on 18/12/2021).

Ronneberger, O., Fischer, P. and Brox, T. (2015), 'U-net: Convolutional networks for biomedical image segmentation', *arXiv e-prints* .

Salembier, P., Oliveras, A. and Garrido, L. (1998), 'Antiextensive connected operators for image and sequence processing', *IEEE Transactions on Image Processing* **7**(4), 555–570.

Samudre, A., George, L., Bansal, M. and Wadadekar, Y. (2020), 'Data-efficient classification of radio galaxies', *Monthly Notices of the Royal Astronomical Society* .

Schechter, P. (1976), 'An analytic expression for the luminosity function for galaxies.', *Astrophysical Journal* **203**, 297–306.

Sebastian, R. (2016), 'An overview of gradient descent optimization algorithms', *arXiv preprint arXiv:1609.04747* .

Serra, J., ed. (1988), *Image Analysis and Mathematical Morphology, Vol. 2: Theoretical Advances*, Vol. 2, Academic Press, New York.

Serra, P., Jurek, R. and Flöer, L. (2012*a*), 'Using negative detections to estimate source-finder reliability', *Publications of the Astronomical Society of Australia* **29**(3), 296–300.

Serra, P., Jurek, R. and Flöer, L. (2012*b*), 'Using negative detections to estimate source-finder reliability', *Publications of the Astronomical Society of Australia* **29**(3), 296–300.

Serra, P. et al. (2015), 'Sofia: a flexible source finder for 3d spectral line data', *Monthly Notices of the Royal Astronomical Society* **448**(2), 1922–1929.

Spekkens, K. and Giovanelli, R. (2006), 'The structure of rapidly rotating late-type spiral galaxies. i. photometry, h i, and optical kinematics', *Astronomical Journal* **132**(4), 1426–1444.

Suppes, P. (1957), *Introduction to Logic*, Van Nostrand, New York.

Swaters, R., van Albada, T., van der Hulst, J. and Sancisi, R. (2002), 'The westerbork hi survey of spiral and irregular galaxies. i. hi imaging of late-type dwarf galaxies', *Astronomy & Astrophysics* **390**, 829–861.

Sørensen, T. (1948), 'A method of establishing groups of equal amplitude in plant sociology based on similarity of species and its application to analyses of the vegetation on danish commons', *Kongelige Danske Videnskabernes Selskab* .

Teeninga, P., Moschini, U., Trager, S. and Wilkinson, M. (2013), Bi-variate statistical attribute filtering: a tool for robust detection of faint objects, *in* '11th International Conference "Pattern Recognition and Image Analysis: New Information Technologies" (PRIA-11-2013)', pp. 746–749.

Teeninga, P., Moschini, U., Trager, S. and Wilkinson, M. (2016), 'Statistical attribute filtering to detect faint extended astronomical sources', *Mathematical Morphology - Theory and Applications* **1**(1), 100–115.

Tody, D. (1986), 'The iraf data reduction and analysis system', *Astrophysics Data System* .

Torrey, L. and Shavlik, J. (2009), *Transfer Learning*.

Tully, R., Verheijen, M., Pierce, M., Huang, J. and Wainscoat, R. (1996), 'The ursa major cluster of galaxies.i.cluster definition and photometric data', *Astronomical Journal* **112**, 2471.

van Cappellen, W. et al. (2021), 'Apertif, phased array feeds for the westerbork synthesis radio telescope', *Astronomy & Astrophysics* .

van der Hulst, J., Terlouw, J., Begeman, K., Zwitser, W. and Roelfsema, P. (1992), The groningen image processing system, gipsy, *in* 'Astronomical Data Analysis Software and Systems I', Vol. 25, pp. 131–+.

Vapnik, V. (1995), 'Support-vector networks'.

Verheijen, M. (2001), 'The ursa major cluster of galaxies. v. h i rotation curve shapes and the tully-fisher relations', *Astrophysical Journal* **563**(2), 694–715.

Verheijen, M. and Sancisi, R. (2001), 'The ursa major cluster of galaxies. iv. hi synthesis observations', *Astronomy & Astrophysics* **370**, 765–867.

Wang, J., Koribalski, B., Serra, P., van der Hulst, T., Roychowdhury, S., Kamphuis, P. and Chengalur, J. (2016), 'New lessons from the h i size-mass relation of galaxies', *Monthly Notices of the Royal Astronomical Society* **460**(2), 2143–2151.

Weltman, A. et al. (2020), 'Fundamental physics with the square kilometre array', *Publications of the Astronomical Society of Australia* **37**.

Westenberg, M., Roerdink, J. and Wilkinson, M. (2007), 'Volumetric attribute filtering and interactive visualization using the max-tree representation', *IEEE Transactions on Image Processing* **16**(12), 2943–2952.

Wright, M. (1974), 'Mapping neutral hydrogen in external galaxies', *Galactic and Extragalactic Radio Astronomy* .

Yang, A., Pan, F., Saragadam, V., Dao, D., Hui, Z., Chang, J. and Sankaranarayanan, A. (2021), Slicenets-a scalable approach for object detection in 3d ct scans, *in* 'Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)', pp. 335–344.

Zhang, Y. and Zhao, Y. (2014), Applications of support vector machines in astronomy, *in* N. Manset and P. Forshay, eds, 'Astronomical Data Analysis Software and Systems XXIII', Vol. 485 of *Astronomical Society of the Pacific Conference Series*, p. 239.

Zhi, S., Liu, Y., Li, X. and Guo, Y. (2018), 'Toward real-time 3d object recognition: A lightweight volumetric cnn framework using multitask learning', *Computers & Graphics* **71**, 199–207.

Appendix

SoFiA Parameter File

Attached is an example of the parameter file used for SoFiA when applied to the nearest spectral window.

```
# List of template parameter settings for SoFiA
#
# Note that SoFiA's default parameters are actually hard-coded in the
# file Parameter.c, and any changes to this file will have no effect.
# This file can instead be used as a template for setting up your own
# parameter files.


# Global settings

pipeline.verbose            =  false
pipeline.pedantic           =  true
pipeline.threads            =  10


# Input

input.data                  =  ./data/training/loudInput/loud_1245mosB.fits
input.region                =
input.gain                  =
input.noise                 =
input.weights               =
input.mask                  =
input.invert                =  false


# Flagging

flag.region                 =
flag.catalog                =
flag.radius                 =  5
flag.auto                   =  false
flag.threshold              =  5.0
flag.log                    =  false


# Continuum subtraction

contsub.enable              =  false
contsub.order               =  0
contsub.threshold           =  2.0
contsub.shift               =  4
contsub.padding             =  3


# Noise scaling
```

```
scaleNoise.enable          =  true
scaleNoise.mode            =  local
scaleNoise.statistic       =  mad
scaleNoise.fluxRange       =  full
scaleNoise.windowXY        =  75
scaleNoise.windowZ         =  51
scaleNoise.gridXY          =  0
scaleNoise.gridZ           =  0
scaleNoise.interpolate     =  true
scaleNoise.scfind          =  false


# Ripple filter

rippleFilter.enable        =  false
rippleFilter.statistic     =  median
rippleFilter.windowXY      =  31
rippleFilter.windowZ       =  15
rippleFilter.gridXY        =  0
rippleFilter.gridZ         =  0
rippleFilter.interpolate   =  false


# S+C finder

scfind.enable              =  true
scfind.kernelsXY           =  0, 3.6, 6.6
scfind.kernelsZ            =  0, 3, 5, 9, 15
scfind.threshold           =  5
scfind.replacement         =  2.0
scfind.statistic           =  mad
scfind.fluxRange           =  full


# Threshold finder

threshold.enable           =  false
threshold.threshold        =  5.0
threshold.mode             =  relative
threshold.statistic        =  mad
threshold.fluxRange        =  negative


# Linker

linker.enable              =  true
linker.radiusXY            =  3
linker.radiusZ             =  3
linker.minSizeXY           =  3
linker.minSizeZ            =  2
linker.maxSizeXY           =  0
linker.maxSizeZ            =  133
linker.positivity          =  false
linker.keepNegative        =  false
```

```
# Reliability

reliability.enable          =  true
reliability.threshold       =  0.3
reliability.scaleKernel     =  0.5
reliability.minSNR          =  3.0
reliability.catalog         =
reliability.plot            =  true
reliability.debug           =  false


# Mask dilation

dilation.enable             =  false
dilation.iterationsXY       =  10
dilation.iterationsZ        =  5
dilation.threshold          =  0.02


# Parameterisation

parameter.enable            =  true
parameter.wcs               =  true
parameter.physical          =  false
parameter.prefix            =  SoFiA
parameter.offset            =  false


# Output

output.directory            =  ./data/sofia_output
output.filename             =  sofia_loud_1245mosB.fits
output.writeCatASCII        =  true
output.writeCatXML          =  false
output.writeCatSQL          =  false
output.writeNoise           =  false
output.writeFiltered        =  false
output.writeMask            =  true
output.writeMask2d          =  false
output.writeRawMask         =  false
output.writeMoments         =  false
output.writeCubelets        =  false
output.marginCubelets       =  0
output.thresholdMom12       =  0.0
output.overwrite            =  true
```