WESTERN SYDNEY
UNIVERSITY

Unsupervised Machine Learning Clustering and Data Exploration of Radio-Astronomical Images

Mr Nicholas O. Ralph

A thesis submitted for the degree of
Master of Research
at
Western Sydney University

November 2018

Supervisors:
Professor Ray P. Norris
Associate Professor Gu Fang

Acknowledgements

Foremost I would like to acknowledge my supervisors Professor Ray Norris and Associate Professor Gu Fang for the time they have spent lending me their support and expertise. I thank the Radio Galaxy Zoo (RGZ) Project builders for their support and the volunteers for their contribution to RGZ dataset and labels used in this thesis. I also wish to acknowledge the researchers and academic staff of Western Sydney University (WSU), Dr Laurence Park and Dr Timothy Galvin for their advice and expertise in machine learning and, Dr Evan Crawford and Dr Nicholas Tothill for proofing and assistance in the examining process. Additionally, Dr Gregory Cohen who provided me invaluable experience on variational autoencoders while working as a research assistant, which provided the inspiration for pursuing autoencoders in this thesis. I also acknowledge the continued advice, collaboration and support of my colleagues, Andrew, Graeme, John, Kevin, Kieran, Mahdieh, Miranda, Miroslav, Nathan, Nigel, Pero, Peter, Rami, Ron and Wayne.

Outside of academia, I do not have enough space to acknowledge and thank all the friends and family who have shown their continuous support throughout my academic career. I wish to thank my parents and siblings for their endless encouragement from my first interests in engineering. Throughout my degree, my friends have been nothing but supportive, always keen to hear about my research. This journey would have been much less enjoyable without the well needed time off spent with Philip and Blake, in particular, Rhys and Sam for helping me recognise my stress throughout my studies.

This thesis uses data from the Karl G. Jansky Very Large Array:

> The National Radio Astronomy Observatory is a facility of the National Science Foundation operated under cooperative agreement by Associated Universities, Inc.

The work presented in this thesis is, to the best of my knowledge and belief, original except as acknowledged in the text.

I hereby declare that I have not submitted this material, either in full or in part, for a degree at this or any other institution.

. . . .

Mr Nicholas O. Ralph November 8, 2018

Foreword

This thesis is the culmination of my research conducted from January 4, 2018, to October 15, 2018, for the completion of my Master of Research and demonstrates my suitability for the Doctor of Philosophy.

My first author paper '*Radio Galaxy Zoo: Unsupervised Clustering of Convolutionally Encoded Radio-astronomical Images*' is featured in the Appendix as Ralph et al. (2018), was submitted to the Machine Intelligence in Astronomy and Astrophysics Special Issue of the Publications of the Astronomical Society of the Pacific on July 24, 2018 and is currently under the process of review (distribution of work amongst authors detailed in Appendix A). The preliminary methodology and results of this thesis were featured in Ralph et al. (2018), and final results are the results obtained after submission of the paper.

This thesis was written from an engineering point of view, based on my undergraduate studies. As a result, the methods explored in this thesis are not unique to the radio astronomy application, but typical of any image processing and machine learning task. I wrote this thesis with a mixed audience and the new 'machine learning astronomer' in mind. This assumes a basic understanding of computation and astronomy and was composed in a manner not to alienate either astronomers or machine learning readers.

Contents

List of Tables

List of Figures

Abstract

In this thesis, I demonstrate a novel and efficient unsupervised clustering and data exploration method with the combination of a Self-Organising Map (SOM) and a Convolutional Autoencoder, applied to radio-astronomical images from the Radio Galaxy Zoo (RGZ) dataset.

The rapidly increasing volume and complexity of radio-astronomical data have ushered in a new era of big-data astronomy which has increased the demand for Machine Learning (ML) solutions. In this era, the sheer amount of image data produced with modern instruments and has resulted in a significant data deluge. Furthermore, the morphologies of objects captured in these radio-astronomical images are highly complex and challenging to classify conclusively due to their intricate and indiscrete nature. Additionally, major radio-astronomical discoveries are unplanned and found in the unexpected, making unsupervised ML highly desirable by operating with few assumptions and without labelled training data.

In this thesis, I developed a novel unsupervised ML approach as a practical solution to these astronomy challenges. Using this system, I demonstrated the use of convolutional autoencoders and SOM's as a dimensionality reduction method to delineate the complexity and volume of astronomical data. My optimised system shows that the coupling of these methods is a powerful method of data exploration and unsupervised clustering of radio-astronomical images.

The results of this thesis show this approach is capable of accurately separating features by complexity on a SOM manifold and unified distance matrix with neighbourhood similarity and hierarchical clustering of the mapped astronomical features. This method provides an effective means to explore the high-level topological relationships of image features and morphology in large datasets automatically with minimal processing time and computational resources. I achieved these capabilities with a new and innovative method of SOM training using the autoencoder compressed latent feature vector representations of radio-astronomical data, rather than raw images. Using this system, I successfully investigated SOM affine transformation invariance and analysed the true nature of rotational effects on this manifold using autoencoder random rotation training augmentations.

Throughout this thesis, I present my method as a powerful new approach to data exploration technique and contribution to the field. The speed and effectiveness of this method indicates excellent scalability and holds implications for use on large future surveys, large-scale instruments such as the Square Kilometre Array and in other big-data and complexity analysis applications.

List of Abbreviations

Adam	Adaptive Moment
AGN	Active Galactic Nuclei
ASKAP	Australian Square Kilometre Array Pathfinder
BMU	Best Matching Unit
CNN	Convolutional Neural Network
CPU	Central Processing Unit
CTA	Cherenkov Telescope Array
DBM	Deep Belief Machine
DR1	Data Release 1
DT	Decision Tree
ELT	Extremely Large Telescope
EMU	Evolutionary Map of the Universe
FIRST	Faint Images of the Radio Sky at Twenty Centimetres
FITS	Flexible Image Transport System
GAN	Generative Adversarial Network
GNG	Growing Neural Gas
GPU	Graphics Processing Unit
HC	Hierarchical Clustering
AUI	Associated Universities, Inc
ID	Identification Number

IR	Infrared
ISM	Inter-Stellar Medium
KNN	k-Nearest Neighbour
LReLU	Leaky Rectified Linear Unit
LSST	Large Synoptic Survey Telescope
MNIST	Modified National Institute of Standards and Technology
MSE	Mean Square Error
MRO	Murchison Radio-astronomy Observatory
ML	Machine Learning
NaN	Not a Number
NN	Neural Network
NRAO	National Radio Astronomy Observatory
NVSS	NRAO VLA Sky Survey
PCA	Principle Component Analysis
PINK	Parallelized rotation and flipping Invariant Kohonen maps
PNG	Portable Network Graphic
PSF	Point Spread Function
RAM	Random Access Memory
RBM	Restricted Boltzmann Machine
ReLU	Rectified Linear Unit
RGZ	Radio Galaxy Zoo
RMS	Root Mean Squared
RTF	Random Tree Forest
SD	Standard Deviation
SDSS	Sloan Digital Sky Survey

SED	Spectral Energy Distribution
SETI	Search of Extraterrestrial Intelligence
SKA	Square Kilometre Array
SOM	Self-Organising Map
SVM	Support Vector Machine
TB	Terabyte
t-SNE	t-Distributed Stochastic Neighbour Embedding
umat	unified distance matrix
VLA	Very Large Array
VLBI	Very Long Base Line Interferometry
WISE	Wide-field Infrared Survey Explorer
WTF	Widefield Outlier Finder

Chapter 1

Background

Since the first observations, astronomers have been pursuing new technologies to peer ever deeper into the cosmos. This pursuit has seen the design focus of instrumentation aimed at pushing the boundaries of sensitivity and resolution. With these advancements, however, the volume and complexity of observational data have also significantly increased. This effect is compounded in radio astronomy with the advent of new large-scale radio telescope arrays. Merely applying progressively greater computation resources to this 'big-data' problem is not efficient and in many cases, not effective (Pruyt et al., 2014). Consequently, powerful Machine Learning solutions are now becoming a key focus to combat the inundation of highly complex observational data in this new modern era of big-data astronomy.

This thesis aimed to use Machine Learning (ML) methods trained on the Radio Galaxy Zoo dataset to bridge these research gaps. To achieve this aim, I developed a novel combination of latent image feature extraction using autoencoders to train a Self-Organising Map SOM for topological morphology analysis. Using this delineated SOM space, I investigated the K-means clustering algorithm for unsupervised clustering of radio-image morphologies. I assessed the performance of this hybrid system by its ability to reduce the volume and complexity of radio astronomical data for morphological exploration, complexity separation and processing time.

This Chapter gives a brief overview of astronomy, the impact of the field and the progression of radio astronomy instrumentation toward large-scale radio telescopes. The challenges and implications of these large-scale machines are discussed in Section 1.1.2, and the machine learning solutions to these challenges are explained in the subsequent Section 1.2. This background Chapter concludes with the research gaps found within literature, in addition to the research questions and aims of this thesis. Methods discussed in this Chapter that are specific to the applications of this thesis are explained in greater detail in my methodology, featured in Chapter 2.

1.1 Modern Astronomy

Astronomy is the science of the celestial objects that the compose universe. This astronomy study is based on observing electromagnetic radiation, cosmic rays, neutrinos and gravitational waves emitted by the astrophysical processes of these objects. It is through these observations that much of our knowledge of the universe has been founded (Riess et al., 1998).

The first astronomers began observing by the naked eye to monitor the visible light emitted from stellar objects. However, the sensitivity and resolution of what could be interpreted by eye prevented astronomers from viewing and distinguishing faint objects or resolving their full detail. As the field grew, these boundaries were gradually pushed with more sophisticated instruments such as optical telescopes. These telescopes pioneered the modern paradigm of collecting as much emission as possible from a large collection surface to maximise the resolution of detected emission.

Although the pursuit of higher sensitivity and resolution has been ceaseless, astronomers sought to observe more than just the visible light. The field gradually advanced to develop emission specific detection methods to capture the remainder of the electromagnetic spectrum outside visible light (Figure 1.1) with ever-improving instrumentation.

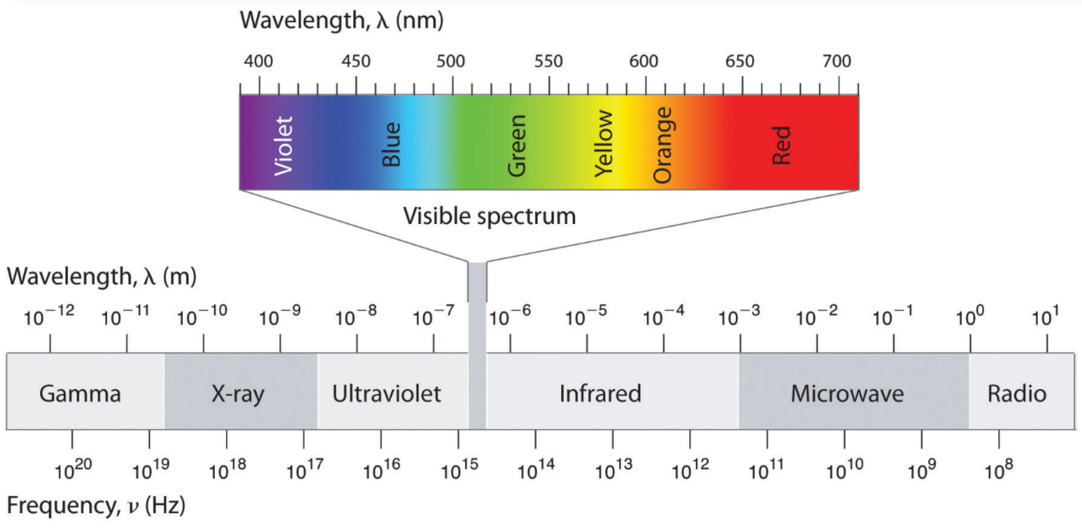

Figure 1.1: Standard model of the electromagnetic spectrum (Credit: Andrew Davidhazy[1]).

Researchers found each part of this spectrum offered different insights into astronomy through the various processes that produce their emissions. Shown in Figure 1.2, the crab nebula uncovers startlingly different features when imaged across

[1]https://saylordotorg.github.io/

the full spectrum. So-called 'multi-messenger' astronomy combines these observation methods with only relatively recently discovered exotic emission such as cosmic rays, neutrinos and gravitational waves to reveal further hidden information (Abbott et al., 2017).

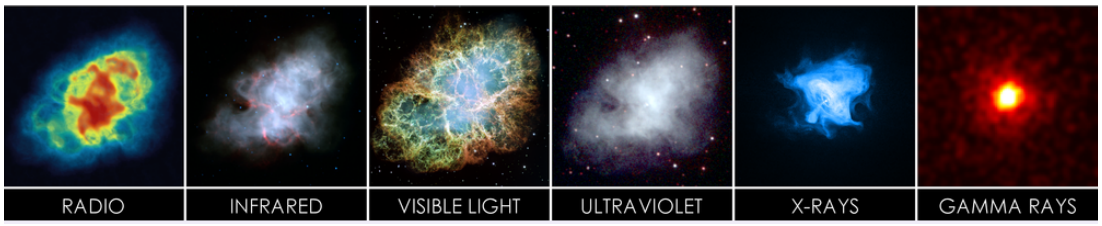

Figure 1.2: The crab nebula observed across the electromagentic spectrum (Credit: Cherenkov Telescope Array[2]).

As researchers further investigated the long wavelength end of the spectrum, they stumbled on the unique myriad of information previously trapped in long wavelength radio emission. These first steps into radio astronomy started in 1933 with engineer Karl G. Jansky's observation of astronomical radio emission from the Milky Way (Jansky, 1933). Since this discovery, radio astronomy has become a key part of the field with the exclusive insight, it provides into exotic objects such as quasars, Active Galactic Nuclei (AGN), the Inter-Stellar Medium (ISM) and the evolution of galaxies and star formation (Norris, 2017b). Much of this insight is gained by cross-matching radio sources with Infrared (IR) sources to differentiate between AGN and star-formation-dominated emission (Seymour et al., 2008; Banfield et al., 2015; Alger et al., 2018).

1.1.1 The Progression of Radio Telescopes to Large-Scale Instruments

Four years after Karl G. Jansky's extra-planetary radio detection, the first parabolic radio telescope was built by Grote Reber in 1937 (Figure 1.3). With this telescope, Reber pioneered the first radio sky survey as a collection of radio objects or 'radio sources' and physical properties such as position (Reber, 1940). Although the principle of these instruments has remained the same, they have expanded to larger single filled aperture telescopes such as the Parkes 64 m telescope in Figure 1.4, to many other variations such as radio dipoles to radio interferometers (Ryle, 1952) and phased array feeds (Johnston et al., 2007).

[2]https://www.cta-observatory.org/science/

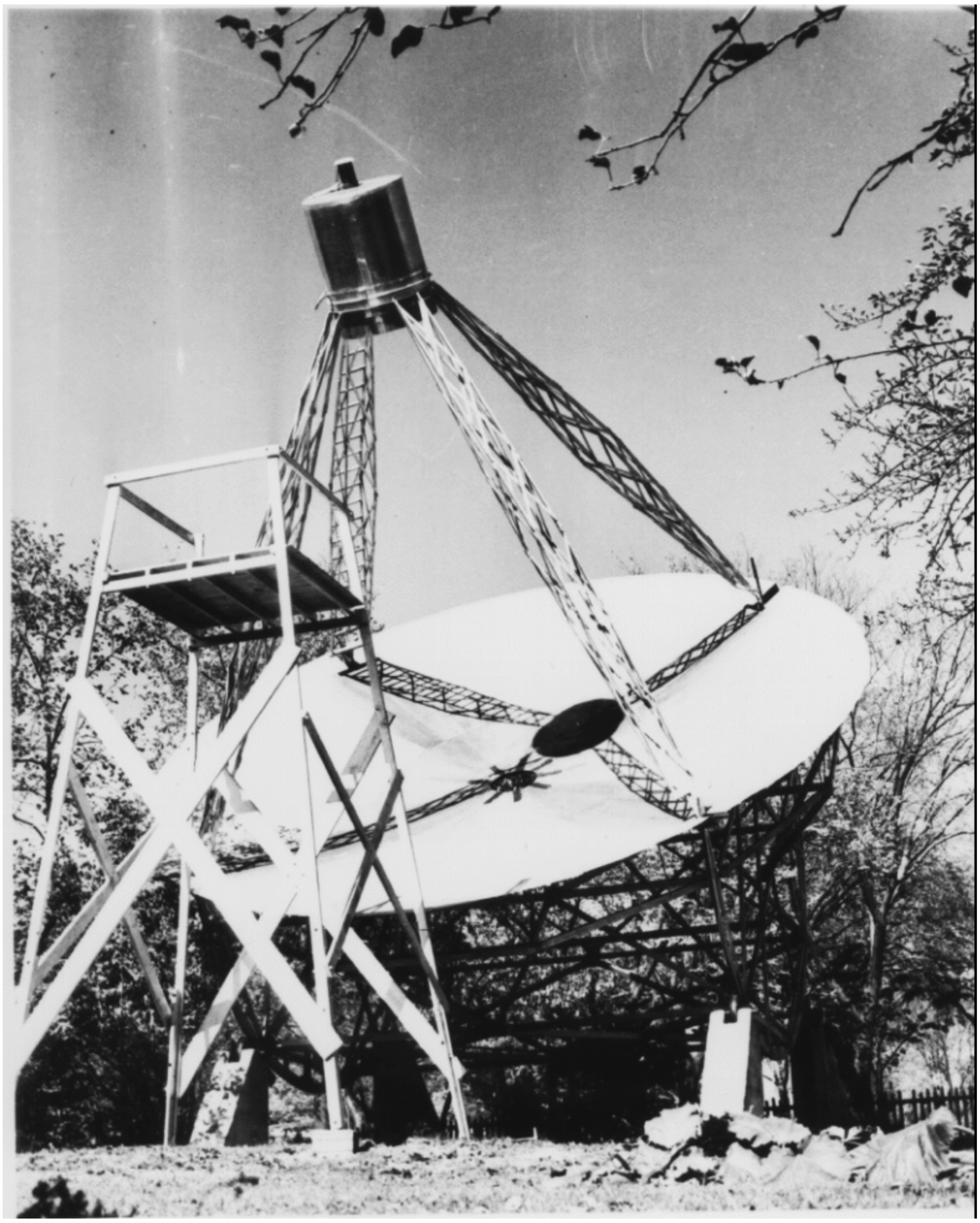

Figure 1.3: The first parabolic radio telescope was built by Grote Reber in 1937 with a parabolic dish made of sheet metal 9.56 m in diameter to focus radio waves to a point 6.1 m above the dish. The wooden tower at the left is used for access to the receiver (Credit: Image courtesy of NRAO/AUI[3]).

[3]https://www.nrao.edu/hist-reber.shtml

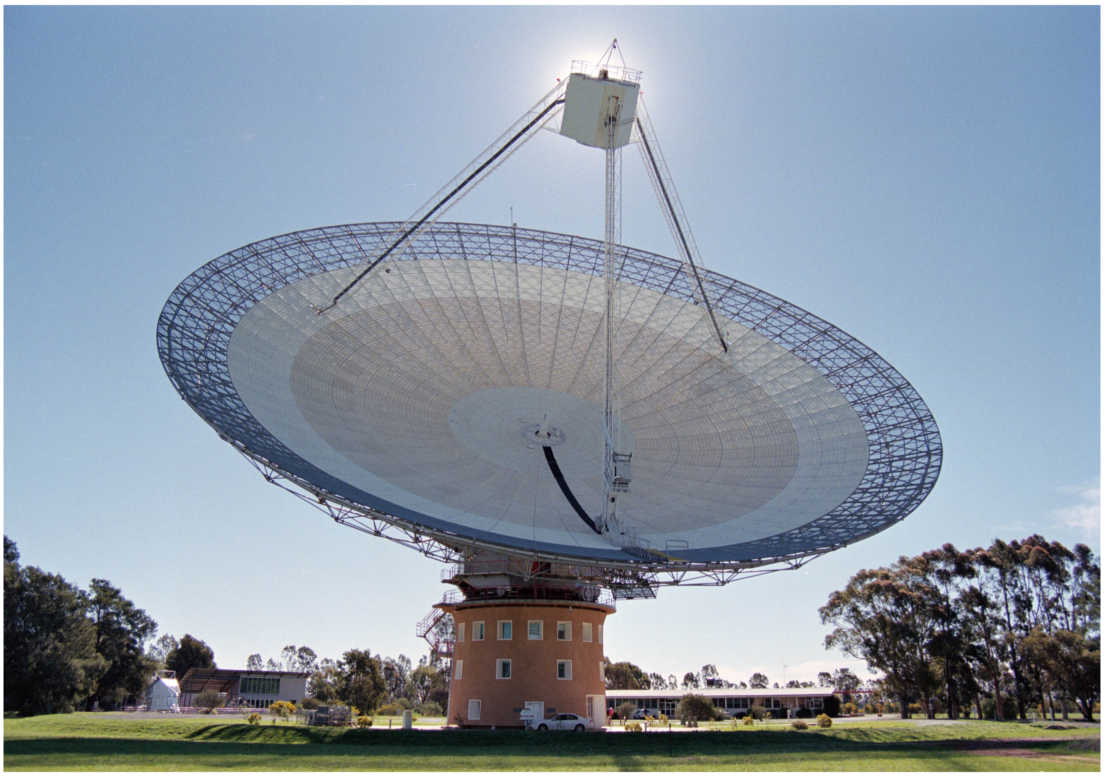

Figure 1.4: The Parkes 64 m steerable filled aperture radio telescope, located in Parkes, New South Wales, Australia (Credit: John Sarkissian[4]).

Parabolic dish systems used as radio telescopes are designed to measure the radio frequency power radiated by astronomical objects. In practical radio astronomy, these systems are known as 'filled aperture radio telescopes' (Findlay, 1971). Typical radio telescopes are analogous to reflective optical telescopes, where arriving electromagnetic waves are directed from a parabolic reflector to the receiver. In the case of radio astronomy, high-resolution imaging requires a large collection due to the implications of collecting long wavelength emission (Christiansen, 1953). As shown in Equation 1.1, the resolution as the smallest angular separation to distinguish separate objects is given by δ as the ratio of emission wavelength λ and the dish diameter D. As a result, this diameter is the essential limiting factor for resolution (Rohlfs and Wilson, 2013).

$$\delta = \frac{\lambda}{D} \tag{1.1}$$

To obtain a radio-brightness distribution of the whole sky or particular region, these antennas are directed to the area of interest, mechanically (or electrically in the case of steerable beams) or simply with the rotation of the earth (Findlay, 1971). Since steering these telescopes is necessary to focus on specific regions, mechanical terms limit the size of these dishes and in turn, restricts the collection area and

[4]https://www.skatelescope.org/news/parkes

thus the angular resolution. By observing radio emission from earth, we are limited to only detecting emission in a finite collecting area based on our perspective, known as flux, S, the density of which is given by the unit of Janksy, Jy where 1 Jy = 1x10^{-26} Wm^{-2}Hz^{-1}.

Despite differences in instrumentation, the same barriers of resolution and sensitivity that astronomers have always faced are no different in radio astronomy. These boundaries are very restrictive with a large collection area required to achieve reasonable angular separation and the typical signal to noise ratio of astronomical sources being between 1x10^{-15} and 1x10^{-20}.

The typical solution to improving these factors is to build telescopes with more sophisticated detection apparatus and a much larger emission collection area. As an inconceivably large dish is not practical or probable, these requirements have increased focus in radio interferometry techniques where observations from multiple radio telescopes can be combined to increase the resolution drastically. Based on Equation 1.1, the effective diameter D of a radio interferometer is given as the separation or 'baseline' between array telescopes.

In radio astronomy this focus has seen the advent of large radio telescope arrays such as the Karl G. Jansky Very Large Array (VLA) in Figure 1.5 and the Australian Square Kilometre Array Pathfinder (ASKAP) in Figure 1.6. Modern interferometry has also seen the implementation of so-called Very Long Base Line Interferometry (VLBI), where telescopes separated by great distances are used as one single large interferometer. The advantages borne from these developments have brought massive scale astronomical instrumentation to the foreground of modern science and engineering with the planning of future facilities such as the Square Kilometre Array (SKA) with arrays spanning multiple countries (Johnston et al., 2007). Similarly, the development of other large-scale instruments is now seen across many other astronomy disciplines with the Extremely Large Telescope (ELT), in optical astronomy and the Cherenkov Telescope Array (CTA) in gamma-ray astronomy.

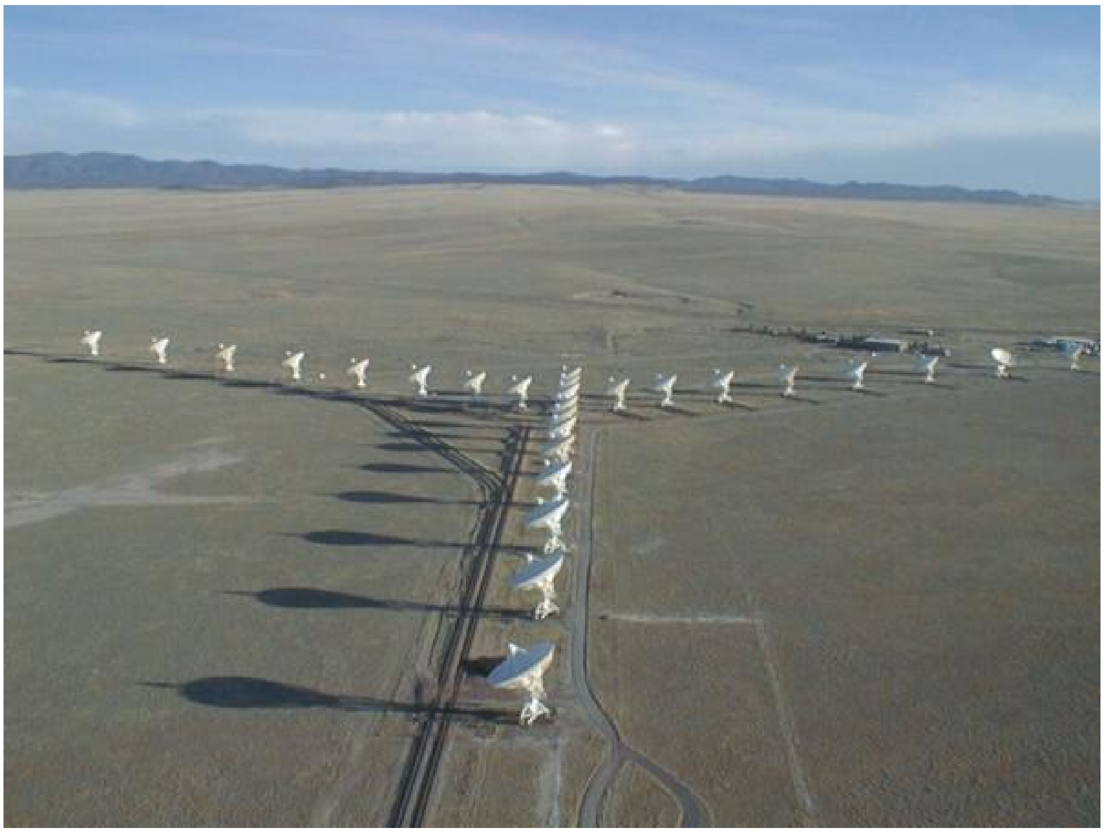

Figure 1.5: The Karl G. Jansky VLA operated by the NRAO and AUI, featuring 27 parabolic antenna's, each with a diameters of 25 m, arranged in a 'Y' shaped array, located in central New Mexico. Image courtesy of NRAO/AUINRAO[5].

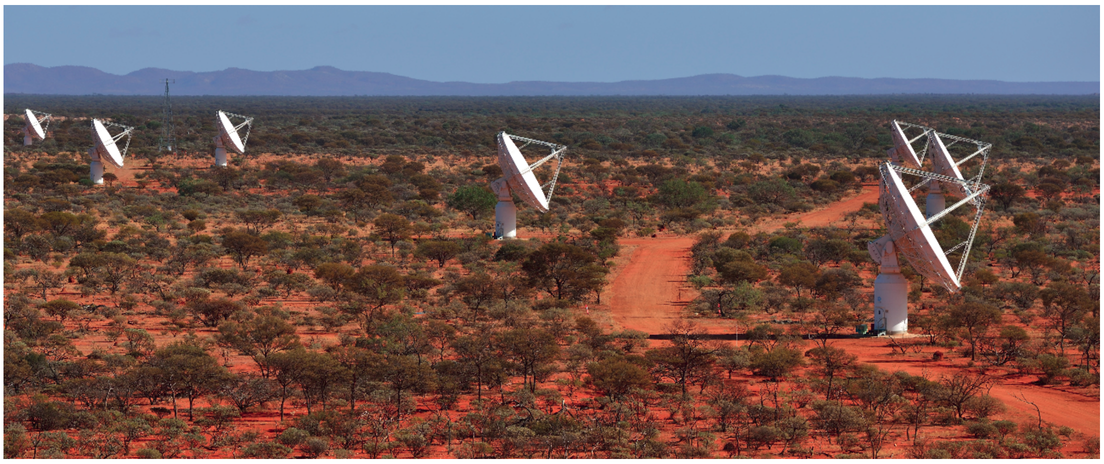

Figure 1.6: The ASKAP, featuring 36 parabolic antenna's (not all present in image), each with a diameter of 12 m, located in the MRO, Western Australia (Credit: Ray Norris, supervisor).

[5]http://images.nrao.edu/90

Although these new immense machines provide vastly deeper observations of the cosmos, they are not without their caveats. The next Section will discuss the data implications and challenges that these large-scale instruments bring to the field and how astronomy is evolving to accommodate them.

1.1.2 Challenges of Data Complexity and Volume in Astronomy

As new astronomical instruments and technologies have increased in scale and complexity, so has the data that these instruments collect (VanderPlas et al., 2012). Consequently, astronomy has entered the era of big data, where current surveys and observations now contain large volumes of highly complex data (Kremer et al., 2017). These large radio continuum surveys have played a key role in our understanding of the evolution of galaxies (Norris, 2017b) and without sufficient techniques to analyse them, discoveries may not be uncovered.

Large-scale radio observing campaigns have seen the production of expansive catalogues such as Faint Images of the Radio Sky at Twenty Centimetres (FIRST) (Becker et al., 1995) with over 950,000 sources and the NRAO VLA Sky Survey (NVSS) (Condon et al., 1998) containing over 1.8 million sources as shown in Figure 1.8. These pale in comparison to exceptionally large surveys such as the Evolutionary Map of the Universe (Norris et al., 2011, EMU) of ASKAP, which is expected to detect over 70 million radio sources. Illustrated in Figure 1.7, these surveys are not only contained an increased number of sources but also greatly increased limiting sensitivity. Although radio surveys have great complexity and volume, even the Evolutionary Map of the Universe (EMU) survey is dwarfed by the Sloan Digital Sky Survey (SDSS) currently with over 1.23 billion sources with an area of 14,555 deg^2 (York et al., 2000) and the planned Large Synoptic Survey Telescope (LSST) survey with a survey area of greater than 20,000 deg^2 (LSST Science Collaboration et al., 2009). Survey science with these massive catalogues further compounds the intricacy of collected data with many observations now spanning multiple wavelengths and surveys. Consequently, the demand for cross-matching radio sources with other observations has increased, with greater potential for radio cross-matching to provide deeper insight into sources and phenomenon found by observing at different wavelengths (Alger et al., 2018).

The sheer volume and complexity of these highly sensitive and expansive surveys is a problem for astronomers (Graff et al., 2014), particularly when traditional techniques for typical tasks such as morphological classification and outlier detection become insufficient for this new volume of multi-wavelength data. These demanding tasks can be segmented into a number of groups:

[5]https://www.sdss.org/dr14/scope/

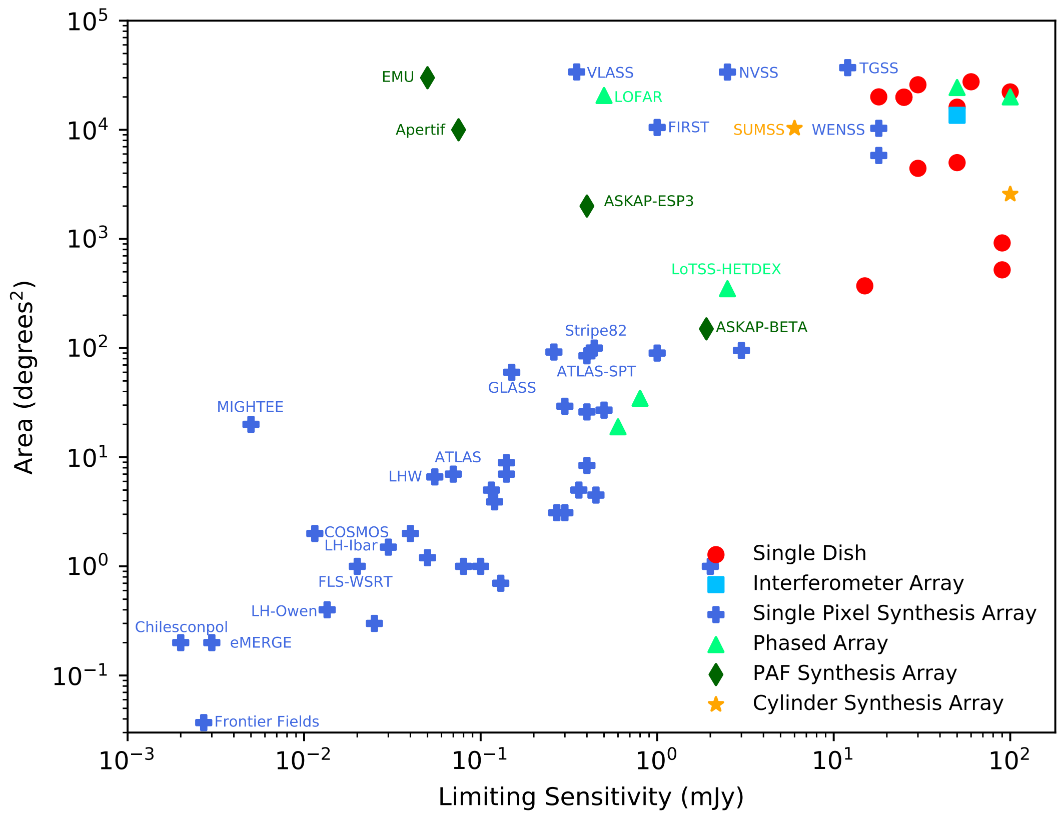

Figure 1.7: The survey sky area vs sensitivity of published radio surveys. Adapted from (Norris, 2017b).

1. Classification and regression;
2. Higher-Dimensional analysis with clustering and visualisation; and
3. Anomaly detection.

The difficulty presented by these tasks in the big-data era of astronomy is pushing researchers towards more sophisticated and automated solutions such as machine learning. The following Section outlines several machine learning methods and their use in astronomy literature to solve these challenges.

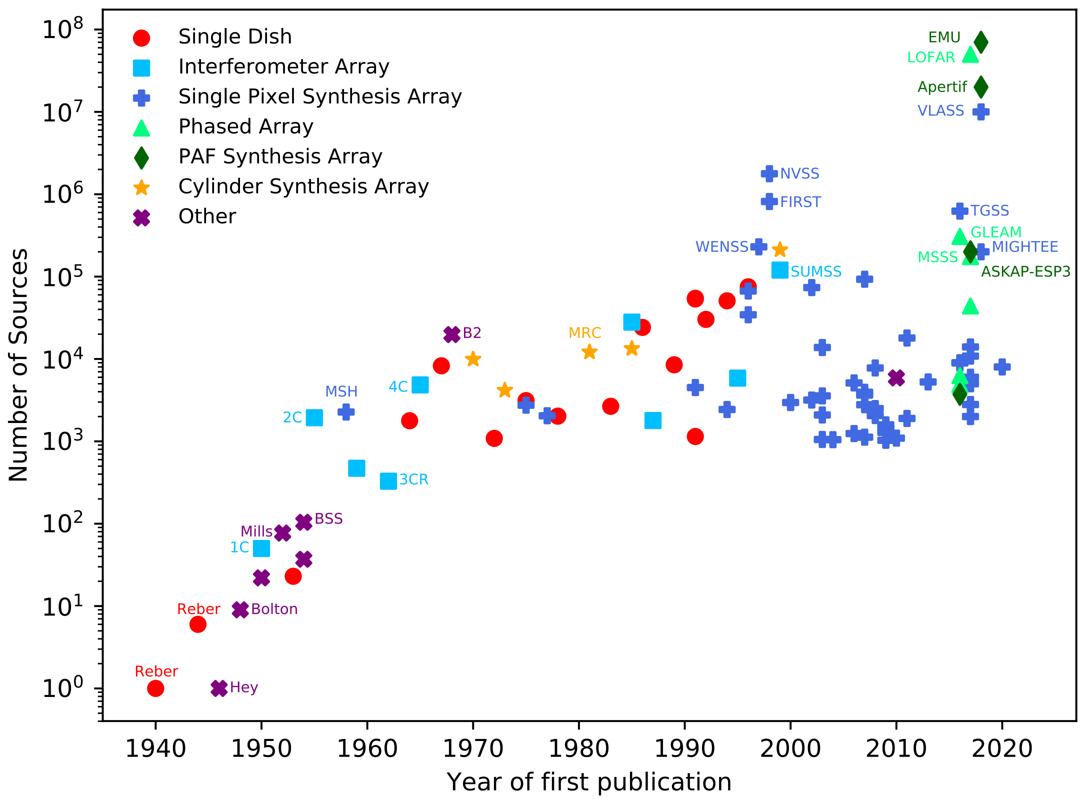

Figure 1.8: The number of known extragalactic radio sources discovered in surveys vs date of survey publication. Adapted from (Norris, 2017b).

1.2 Machine Learning Solutions to Modern Astronomy Challenges

The ML technique is a common approach in data mining and statistics, where a system is 'trained' to model the characteristics of a dataset (Mackay, 2003). Ideally, a properly trained ML system can predict the qualities or classification of a training set or new unencountered data (Kotsiantis et al., 2007). At their core, these ML systems work to optimise some representation of a dataset by updating its parameters based on the evaluation of predictions made on the training set (Domingos, 2012; Boyd et al., 2011).

The ability of ML methods to process highly complex data has attracted the zeal of many modern science applications from Genetics (Libbrecht and Noble, 2015), Medicine (Kononenko, 2001), Image Processing (LeCun et al., 2015), and the modern astronomer facing issues of big-data. Since its adoption in literature, this ML approach has been shown as a powerful tool in overcoming the challenges of modern radio astronomy and the wider field (Ball and Brunner, 2010). Despite these successes, there is no perfect single ML system or combination of ML systems as a hybrid or *ensemble system* (Opitz and Maclin, 1999) that balances speed and accuracy for all applications. Due to their inherent complexity and autonomous nature, these systems are often viewed as 'black-box' (Krause et al., 2016). As a result, ML methods require a careful application or robust ensemble combination to achieve sound results.

The complexity of ML systems can be found in its autonomous optimisation and the process of refining the trade-off between the sensitivity of the system to small changes (bias) and substantial changes (variance) in the input dataset (Geman et al., 1992). Excessive variance in training causes over-fitting, a problem of early methods (Dietterich, 1995), where the learned model too closely resembles the input training set to characterise new data adequately. Conversely, extreme bias causes under-fitting, where the characteristic features of the training set are overlooked.

Finding this balance is further complicated when a trained ML algorithm used to classify images with a specific orientation and scale encounters the same training image at an untrained angle or scale (Perantonis and Lisboa, 1992). Affine transformations such as rotation, scaling and translation, are a typical issue commonly resolved by augmenting a training set with random transformations (Polsterer et al., 2015) often at the cost of training time or affine invariant techniques (Memisevic and Hinton, 2007). Training for affine transformation and balancing the bias-variance trade-off is critical in producing a generalised system that can be trained to work across multiple applications.

The new ML subset of astronomy employs a wide range of ML methods. These are typically divided by their modelling approach and their unique training methods (Schmidhuber, 2015); *supervised learning* and *unsupervised learning*. Using the supervised learning paradigm, systems can be trained explicitly on labelled data to predict the labels of new unseen data (Kotsiantis et al., 2007). Conversely, unsupervised learning is conducted without labelled data, instead of detecting relationships within the training set (Langkvist et al., 2014). The choice between these training methods holds many implications on the performance and outcomes of the system. Although supervised learning allows the user to specify how the dataset should be classified by training set labels, this inherently produces a human bias and requires time-consuming manual labelling of training sets (Torralba and Efros, 2011). Consequentially, unsupervised methods are significantly more difficult as the human user has less control over how the system maps the training set. Nonetheless, this unsupervised paradigm has been shown to detect highly abstract relationships beyond human comprehension, without the bias and labour inherent in manual labelling (Radford et al., 2015).

Despite their differences and training methods, many machine learning methods can be used in a combinatorial sense, seen with the advent of hybrid and ensemble machine learning methods aiming to combine the advantages posed by multiple algorithms (Dietterich, 2000). As with these ensemble systems, training method combinations have been implemented with semi-supervised learning to combine the advantages of both training methods by functioning without a full set of training data labels (Zhu, 2006).

The following sections analyse the literature of machine learning and astronomy applied to the challenges detailed in Section 1.1.2. This ML literature is not exclusive to radio astronomy, as many appropriate ML solutions can be found from other fields and astronomy sub-fields. The various applications of these methods and their ability to solve the current challenges of data scale and complexity in astronomy are also discussed.

1.2.1 Astronomical Classification and Inference

Classification in ML is the mapping of data to a discrete class variable (Dougherty et al., 1995) to infer the class of new unknown data based on the learned relationships between training variables. ML related tasks such as regression map the continuous variables of a training set to some continuous output. These classification, inference and regression tasks are highly valuable for applications such as astronomy with the ability to automatically analyse and organise data. As a result, these ML techniques are commonly applied to Astronomy.

In Astronomy, ML classification is commonly applied to mapping an object to a class based on observed physical differences. Literature sees prevalent use of these ML systems in the galaxy morphology classification (Huertas-Company et al., 2008;

VanderPlas et al., 2012; Graff et al., 2014; Lukic et al., 2018). Additionally, ML classification has been used to discern between compact point sources or extended sources in star-galaxy separation (Kovcs and Szapudi, 2015). Many applications are focussed on the simple separation of sources into logical bins such as simple-complex classification employed in radio astronomy (Lukic et al., 2018), illustrated in Figure 1.9, as an extension of morphology classification. Although effective, many of these methods rely on supervised training, requiring large labelled training sets such as those provided by the RGZ datasets (Wong, 2018). The RGZ dataset is a citizen science project (Banfield et al., 2015) which produces a large image dataset with hand labelled annotations of the number of components and peaks for every resolved source in each image (Banfield et al., 2015). The majority of the radio image data in this dataset comes from the 1.4 GHz FIRST survey (Becker et al., 1995) version 14 March 2004 and the 3.4 μm mid-IR data from the WISE survey all-sky data release in March 2012 (Cutri and et al., 2012).

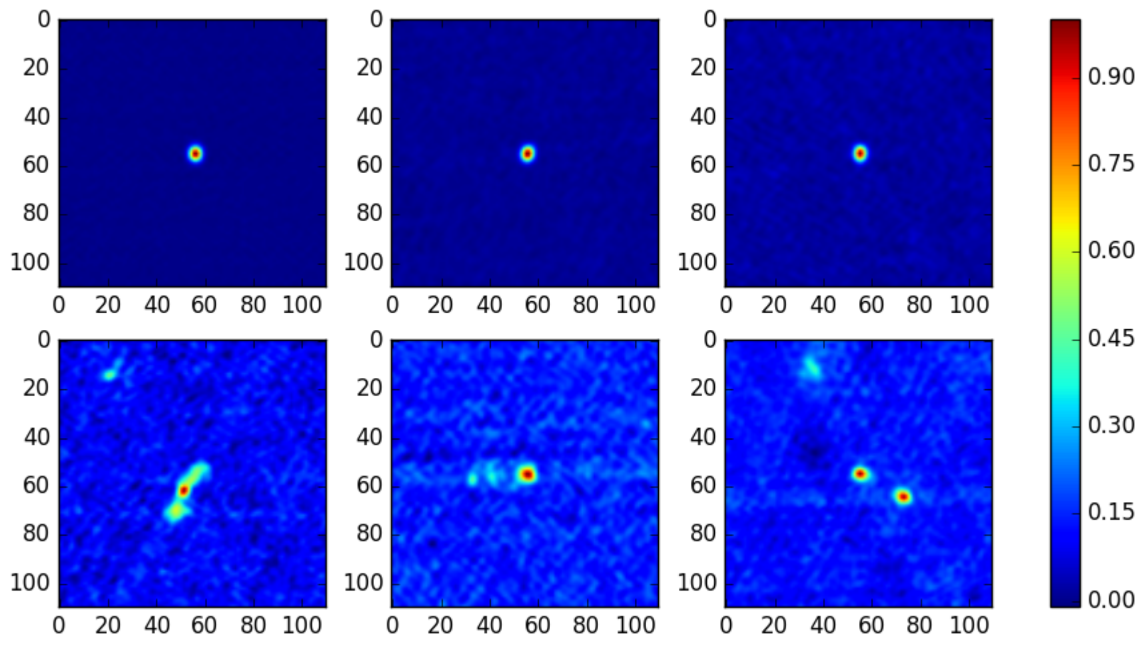

Figure 1.9: Examples of 'simple' compact (top row) and 'complex' multiple component extended source images (bottom row) from the RGZ dataset as the target of a ML classifier to distinguish (Lukic et al., 2018).

ML tasks such as these are difficult as astronomical data is not entirely discrete with features such as diffuse or extended structure and evolutionary stages that lack distinct characteristics or require cross-matching across multiple wavelengths. In many cases, radio sources contain several unresolved components and peaks, which requires sophisticated ML solutions to distinguish cross-matching classes in large data volumes as sources across wavelengths may represent vastly different phenomenon (Lukic et al., 2018). Originally conducted through visual inspection, this cross-matching process is commonly achieved with simple probabilistic and algorithmic approaches (Downes et al., 1986; Sutherland and Saunders, 1992; Fan et al., 2015), and has become a focus of ML astronomy in light of the new data deluge (Banfield et al., 2015; Alger et al., 2018).

Outside of the image domain, photometric 'redshift' estimation has been gaining attention in ML classification (Salvato et al., 2018), particularly in light of the increased size and sensitivity of new radio surveys. Inferring these redshifts allow astronomers to estimate the distance of extra-galactic sources. This is a relatively simple task with complete spectroscopic data, as an operation of detecting the shift in features of an observed Spectral Energy Distribution (SED) due to the expansion of the universe. With sparse and incomplete photometric data, this becomes a strong case for ML use, where redshift can be automatically estimated by fitting sparse data of a source SED to a known model (Bentez, 2000).

In the following sub-sections, the foundation methods of machine learning applied to these classification and inference tasks are demonstrated, and their effectiveness in astronomy discussed.

1.2.1.1 K-Nearest Neighbours Algorithm

The *k-Nearest Neighbour (KNN)* algorithm (Cover and Hart, 1967), is a simple machine learning technique that classifies data based on the 'voting' of K neighbour classes surrounding given data points or vector elements x_i and y_i, separated by a distance function d, such as Euclidean Distance D (Equation 1.2). This algorithm has seen successful use in astronomy in galaxy evolution classification using spectral data (Bundy et al., 2006), redshift estimation (Ball et al., 2007) and outlier classification (Marengo and Sanchez, 2009). The main advantage of the KNN algorithm is the relative simplicity of its operation and how it intuitively separates data. However, this algorithm is computationally intensive with every combination of dataset element within a given neighbourhood analysed for training and analysis.

$$D(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^{n} (y_i - x_i)^2} \tag{1.2}$$

1.2.1.2 Support Vector Machines

Literature has shown success with more complex approaches such as a Support Vector Machine (Cortes and Vapnik, 1995, SVM), for binary class separation. Support Vector Machines (SVMs) achieve this separation with a hyperplane decision surface shown in Figure 1.10, learned by raising the dimensionality of input data using the kernel method. Using this method, data instances in this space are defined by support vectors (maximum margins) that specify their position relative to this decision surface.

SVMs have demonstrated excellent performance in multi-wavelength object classification (Zhang and Zhao, 2004; Malek et al., 2013), stellar spectra classification (Liu et al., 2018) and star-galaxy separation (Fadely et al., 2012). Additionally, recent work with SVMs has shown successful implementation of fuzzy SVMs to take measurement error into account in object classification (Poliszczuk et al., 2018). Combining fuzzy operations in this way was shown as a robust learning method in classifying astronomical data despite the indiscrete nature of astronomical data. Much like the KNN approach, SVMs tend to produce an intuitive output; however, the underlying kernel trick operation is highly abstract.

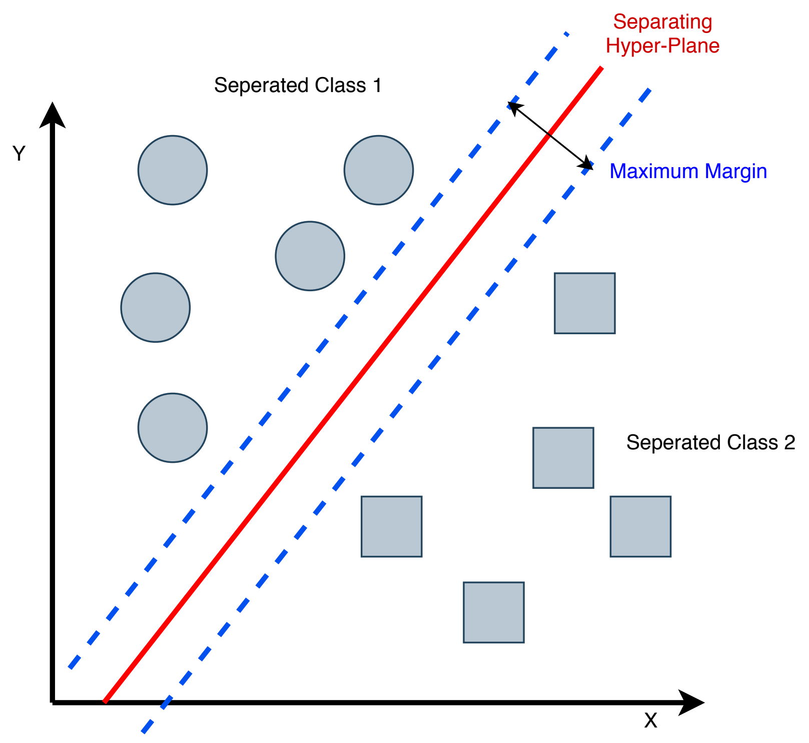

Figure 1.10: Demonstration of SVM separation of two classes using the calculated hyper-plane and surrounding support vectors as the maximum margins. Adapted from (Cortes and Vapnik, 1995).

1.2.1.3 Decision Tree Systems

Decision Tree (DT) systems (Morgan and Sonquist, 1963) operate on a similar paradigm as SVMs by classifying through separation, but with the ability to separate data into multiple topological classes. The structure of a DT system is shown Figure 1.11, as an acyclic graph containing a root node, which splits into child branches and nodes. These systems separate data from the root into child nodes by the attributes of a dataset iteratively, while minimising a given error criterion. The final nodes or 'leaves' represent the learned DT classes. Astronomers have used this intuitive approach with great success much like the KNN and SVM algorithms in redshift estimation (Suchkov et al., 2005), star-galaxy separation (Weir et al., 1995), (Ball et al., 2006), but also in efforts of Search of Extraterrestrial Intelligence (SETI) (Tarter, 2001). This method offers a unique advantage over other black-box machine learning methods by clearly showing the decision conditions that dictate the tree structure of the dataset. This interpretable nature is a quality well recognised by astronomers in literature (Morice-Atkinson et al., 2017). These systems bear a resemblance to similar tree systems employed with *genetic algorithms* (Mitchell, 1998), which also function to optimise toward some solution in an evolutionary manner, where successive 'generations' of solutions are iterated and the best chosen until convergence.

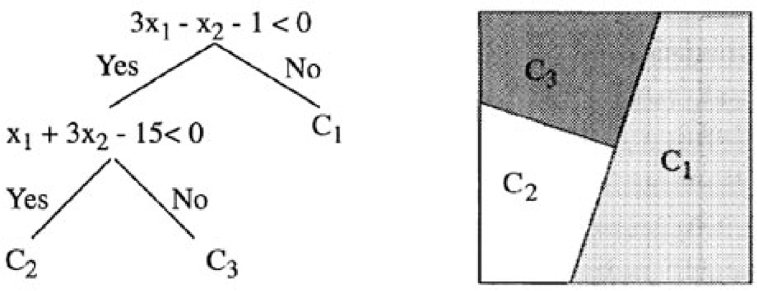

Figure 1.11: An DT system applied to astronomy (left) used to separate sources into classes C_1, C_2, and C_3 based on observed variables $x1$ and $x2$. Displayed on the right are the tree partitions as decision surfaces in the 2 dimensional attribute space (Salzberg et al., 1995).

Multiple DT systems are often combined into an ensemble as a 'bagging' operation to create a Random Tree Forest (RTF) (Ho, 1995; Breiman, 1996, 2001). In a RTF, each DT is trained on a subset of the data, the output of which is combined by mean and variance to create the final classification. As in the original DT approach, great success in classification has been achieved, but at the cost of increased computational time and reduced human-readability. Despite this trade off, RTFs have gained popularity in the astronomy community with use spanning from classification of periodic variable stars into importance, periodicity and amplitude classes (Dubath et al., 2011), photometric redshift estimation (Carliles et al., 2010; Carrasco Kind and Brunner, 2013), in addition to typical image classification approaches with supernova remnant (Bailey et al., 2007), multi-wavelength object (Gao et al., 2009) and active object classification (Zhao and Zhang, 2008; Richards et al., 2011)

1.2.1.4 Neural Networks

Inspired by biological neurology, Neural Network (NN) learning systems (McCulloch and Pitts, 1943) are widely considered the current state of the art in machine learning with resounding success in a wide range of applications (LeCun et al., 2015). NNs are networks of parametrised functions termed 'neurons', that operate as function approximators. In a standard *forward feed* NN, neurons accept data as a single element, whole dataset or a small dataset batch from the input layer, which is forwarded through successive hidden layers and finally an output layer as in Figure 1.12.

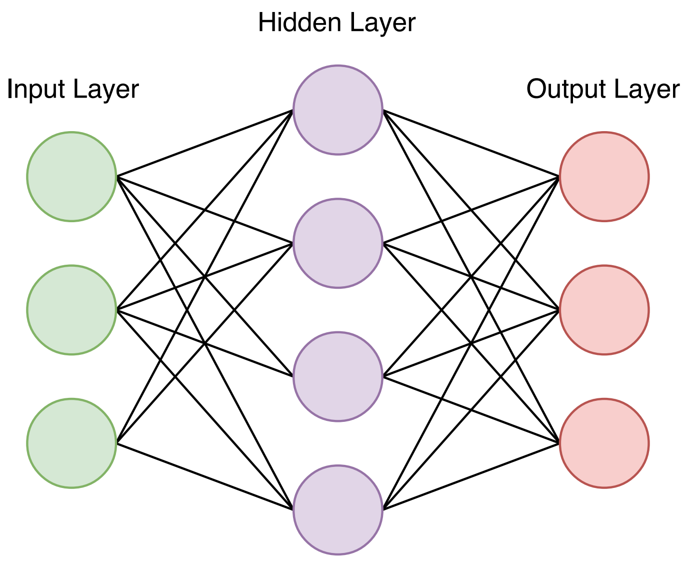

Figure 1.12: The network architecture of a simple fully connected NN with three layers; an input layer, a single hidden layer in the centre and an output layer.

Neuron outputs are determined by their specific activation function (Equation 1.3). A basic activation function as in Equation 1.3 will produce an output y from the input x by a weight product W, and summed with a bias b (Harvey, 1994).

$$y = f(w_i x + b_i) \tag{1.3}$$

This final layer output, if expressed as a single neuron, is the sum of all constitution neuron outputs as in Equation 1.4 and illustrated in Figure 1.12.

$$y_{kj} = \sum_{i=0}^{k} f(W_k x + b_k) \tag{1.4}$$

NN output layers can also possess many different structures to produce the desired trained output, ranging from a single value or continuous value *softmax* regression (normalised exponential, Equation 1.5) as an array of weighted neuron outputs. These *regression layers* are a common feature in NNs used to reduce neuron outputs or whole layers down to a single variable.

$$P(y = j \mid \mathbf{x}) = \frac{e^{\mathbf{x}^{\mathsf{T}}\mathbf{W}_j}}{\sum_{k=1}^{K} e^{\mathbf{x}^{\mathsf{T}}\mathbf{W}_k}} \tag{1.5}$$

The output of these final layers effectively constitutes the prediction of the network. Commonly, network prediction is ‘learned’ by iteratively turning neuron weight and bias parameters via mechanisms such as back-propagation (Harvey, 1994). In this process, weights and bias’ are updated using stochastic gradient descent (Robbins and Monro, 1951), scaled by a learning rate and a given loss function as the difference measure between the input and output prediction to minimise a general objective function. Typical examples of these include MSE (Equation 1.6 or Euclidean Distance (Equation 1.2) where x_i and y_i denote the i^{th} network output and y the i^{th} observation.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{k}(y_i - x_i)^2 \tag{1.6}$$

Recent literature has seen the development of the Adaptive Moment (Adam) optimiser (Kingma and Ba, 2014), as an algorithm to optimise objective functions with adaptive estimates of lower-order moments. This algorithm has seen considerable use and success as a simple, computationally efficient and effective method in large parameter networks (Ruder, 2016).

$$y_i = max(w_i^T x, 0) = \begin{cases} w_i^T x & \text{if } w_i^T x > 0 \\ 0 & \text{else} \end{cases} \tag{1.7}$$

Non-linear neuron activation functions such as *tanh*, *sigmoid* and *Rectified Linear Unit (ReLU)* (Equation 1.7) functions have greatly contributed to the success of NNs by allowing the networks to recognise the underlying non-linear characteristics of the input set (LeCun et al., 2015). Additionally, these functions provide a measure of normalisation, where a network can generalise even outliers which may possess data which resides in extreme ranges. This generalisation approach has also seen the implementation of techniques such as ‘drop-out’ (Srivastava et al., 2014) used to prevent over-fitting by randomly dropping neuron weights based on some given prior.

NNs have shown great success in multiple fields with a vast amount of tunable parameters and the ability to interpret complex non-linear and higher dimensional data (Krizhevsky et al., 2012). This success has been further improved with the advent of so-called ‘deep learning’ with deep NNs containing a large number of hidden layers to allow higher levels of abstraction (LeCun et al., 2015). Throughout the development of NNs, several augmentations to the structure, layers, connection types and neuron composition have been created to extend network capabilities further.

The addition of *Convolutional Layers* in NN as a CNNs (Szegedy et al., 2015) is heralded as a major turning point in NNswith a significant increase in a wide range of successful applications. Using a convolutional layer, a series of feature maps or kernels are spatially mapped to sections of the input data with shared weights. Addition of these layers provides NNs with the ability to create a series of receptive fields

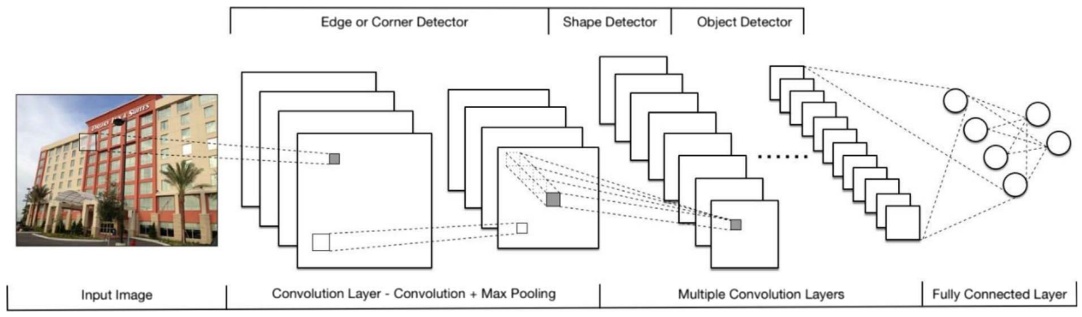

Figure 1.13: A common CNN architecture, processing an multi-channel input image using a series of convolution, max-pooling and fully connected layers to detect objects by a variety edge and shape features (Ma et al., 2018).

across input data. By learning with these fields and linking all previous layer neurons to a successive layer with *fully connected layers*, a CNN is capable of reasoning by interpreting high-level connectivity and spatial arrangements of input data (Girshick et al., 2014). *Pooling Layers* are often used in combination with these convolutions and fully connected layers to merge similar input features together by applying a pooling kernel, usually as a maximum pooling operation to cluster the specified kernel mask to a single output as the maximum value neuron of the mask. The effect of this is essentially equivalent to analyse the image from a different scale to focus on global features while still associating similar specially located features together.

The success of NNs with these architectures has seen significant use in common astronomy tasks, namely, galaxy colour (Ball et al., 2008), morphology classification (Lukic et al., 2018; Lukic and Brggen, 2017; Graff et al., 2014; Xie et al., 2012; Ball et al., 2008; Odewahn, 1995) and photometric redshift classification (Tagliaferri et al., 2003).

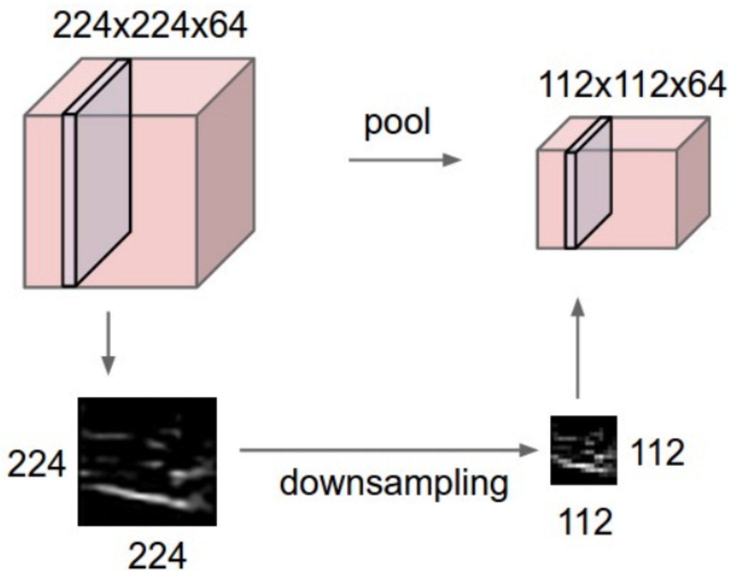

Figure 1.14: Demonstration of the down-sampling effect of an image using a max-pooling layer (Credit Andrej Karpathy[6]).

1.2.2 Higher-Dimensional Analysis with Clustering and Visualisation as Data Exploration Techniques

Current ML research applied to astronomy is focussed on the development of new classification and inference methods or optimisation of old approaches, despite the newly increased scale and complexity of astronomical data. While these methods are sound solutions to the current big-data challenges of the field, ML literature also contains alternate solutions that compliment classification and inference with a focus on reducing data complexity and volume (Al-Jarrah et al., 2015). Research on scalability, improved efficiency and lowered complexity alleviates the strain of big-data and maintains the applicability of techniques that cannot be trained with reasonable time or computational resources (Zhou et al., 2017). This research direction has been met with the application of dimensionality reduction methods (demonstrated in Figure 1.15, ML deep learning and information bottleneck methods to reduce data complexity and volume for more efficient information extraction.

[6]http://cs231n.github.io/convolutional-networks/

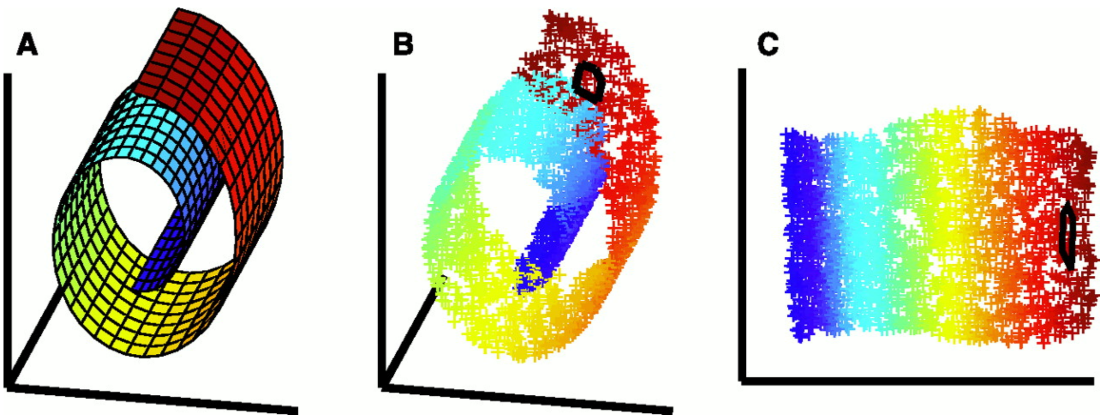

Figure 1.15: Classical dimensionality reduction of the 'Swiss roll' toy dataset, where complex three dimensional data with an additional colour channel can be reduced to two dimensions with minimal data loss and increased human readability as a visualisation technique (Roweis and Saul, 2000).

Astronomical data contains many extra dimensions, including not just intensity or morphology, but SED, redshift, polarisation, features across multiple wavelengths, and possible attributes that may be unnecessary to an experiment. Dimensionality reduction has been utilised across multiple fields to delineate data and reduce problem spaces (Roweis and Saul, 2000) by reducing the number of observational attributes while retaining as much relevant information as possible. By reducing this problem space, analysis may be streamlined, and vital latent relationships buried in these higher dimensions can be analysed (Tenenbaum et al., 2000). This reduction approach is commonly conducted by projecting data to a *learning manifold* in a two or three-dimensional feature space. In many cases, a reduction to a human-readable space may also provide a means to visualise the reduced data and reveal high-level relationships as a data exploration tool. Additional layers of abstraction are often provided with *clustering techniques* to associate points in these feature spaces to a class or group (Bezdek, 1981).

This Section will detail the fundamental dimensionality reduction and clustering techniques used in astronomy for analysis, unsupervised classification and visualisation.

1.2.2.1 Fundamental Dimensionality Reduction Methods

Algorithmic dimensionality reduction methods fundamentally aim to locate the most salient points within a dataset and project and project them to lower dimensional space. In image processing, these techniques are used to display abstract relationships and topology onto learning manifolds. Simple methods such as Principle Component Analysis (PCA) (Hotelling, 1933) are commonly employed to reduce these parameter spaces and reveal underlying structure. This mathematically robust method finds these latent relationships by locating linear combinations called *principle components* that successively have the maximum variance for a given data subject while still being uncorrelated with previous observations (Jolliffe, 2011). In this maximisation problem, PCA functions by solving for the eigenvectors of the input data's covariance matrix that correspond to the largest eigenvalues. These eigenvalues give the variances of their respective principal components and the ratio of the sum of the first principle components.

PCA has been applied to several astronomy tasks, namely reducing spectroscopic imaging of the ISM (Heyer and Schloerb, 1997), reducing stellar spectra (Ronen et al., 1999), clustering for compositional taxonomy of comets (Schleicher and Bair, 2010) and spectral line inversion (Rees et al., 2000), in addition to preprocessing for NN classification (Singh et al., 1998; Storrie-Lombardi et al., 1994).

Additionally, methods such as *t-Distributed Stochastic Neighbour Embedding (t-SNE)* (Maaten and Hinton, 2008) have also been shown to perform effective dimensionality reduction by converting high-dimensional Euclidean distances between data points x_j and x_i into conditional probabilities $p_{j|i}$ with a Gaussian 'perplexity' kernel (Equation 1.8).

$$p_{j|i} = \frac{\exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2 / 2\sigma_i^2)}{\sum_{k \neq i} \exp(-\|\mathbf{x}_i - \mathbf{x}_k\|^2 / 2\sigma_i^2)} \tag{1.8}$$

In this conditional probability space, similarity $q_{j|i}$ between data points projected into this lower space y_j and q_i, is given as Equation 1.9.

$$q_{ij} = \frac{(1 + \|\mathbf{y}_i - \mathbf{y}_j\|^2)^{-1}}{\sum_{k \neq l} (1 + \|\mathbf{y}_k - \mathbf{y}_l\|^2)^{-1}} \tag{1.9}$$

Conditional probability manifolds calculated with t-SNE has been shown to reveal well-separated groupings (Sedlmair et al., 2012). However, this method is used with caution with a tendency to producing unreliable projection features (Maaten and Hinton, 2008). Despite this caveat, t-SNE has been implemented in a number of astronomy applications from spectral classification and diagnostics of the Galah Survey (Traven et al., 2017), spectra dimensionality reduction for metalicity analysis (Matijeviffh et al., 2017) and projection of stellar abundance-space distribution (Anders et al., 2018), similar to typical Modified National Institute of Standards and Technology (MNIST) manifold analysis, shown in Figure 1.16.

Although these relatively simple methods are successful, competitive learning and dimensionality reduction NN variants such as *self-organising maps*, have gained considerable interest in recent years.

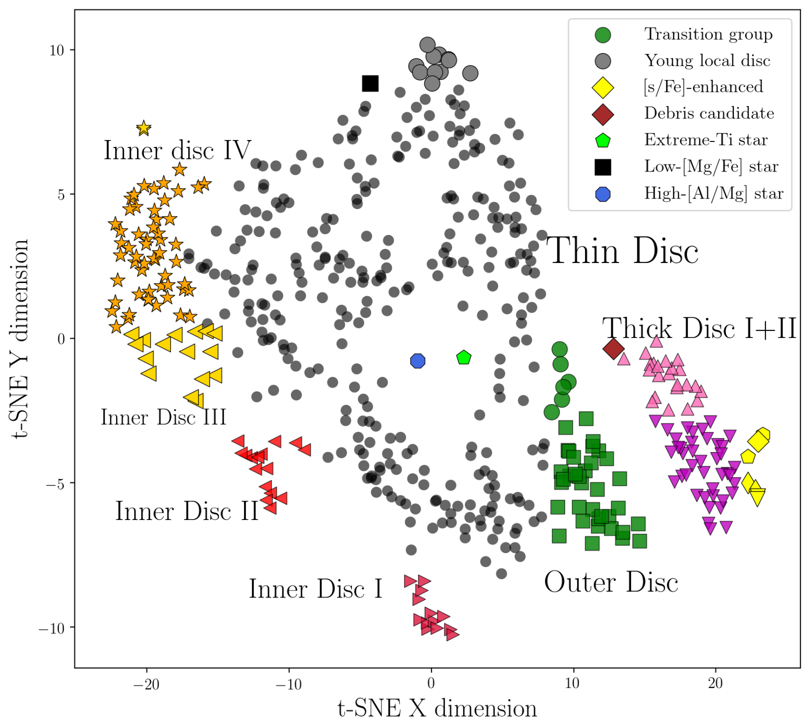

Figure 1.16: t-SNE projection of observed stellar chemical element abundance, highlighting several high level relationships in a human readable space (Anders et al., 2018).)

1.2.2.2 Self-Organising Maps

The *SOM* (Kohonen, 1997) is a NN variant that offers a more complex alternative to unsupervised clustering and data exploration. Shown in Figure 1.17, SOMs create learning manifolds as similarity maps of input data where distinct groups of neurons reflect latent relationships in the training data. As in Equation 1.10, grid neurons, i, are trained to iteratively update their weight vectors m_i by moving toward similar data points $x(t)$ on the manifold by optimising a neighbourhood distance h_{ci} by a learning rate α. A well-trained SOM after t epochs, will visualise the dynamic distribution of input data and display various high-level topological relationships and morphology distributions.

$$m_i(t+1) = m_i(t) + \alpha(t) \cdot h_{ci}(t)[x(t) - m_i(t)] \tag{1.10}$$

These maps have been recognised as powerful unsupervised data exploration tools in astronomy with quasar detection in the SDSS (Meusinger et al., 2012), cluster analysis of Gigahertz-peaked spectrum and high frequency peaking AGN (Torniainen et al., 2008), redshift estimation (Carrasco Kind and Brunner, 2014; Geach, 2012), star galaxy separation (Tagliaferri et al., 2003), galaxy classification (Molinari and

[7]http://matias-ck.com/mlz/somz.html

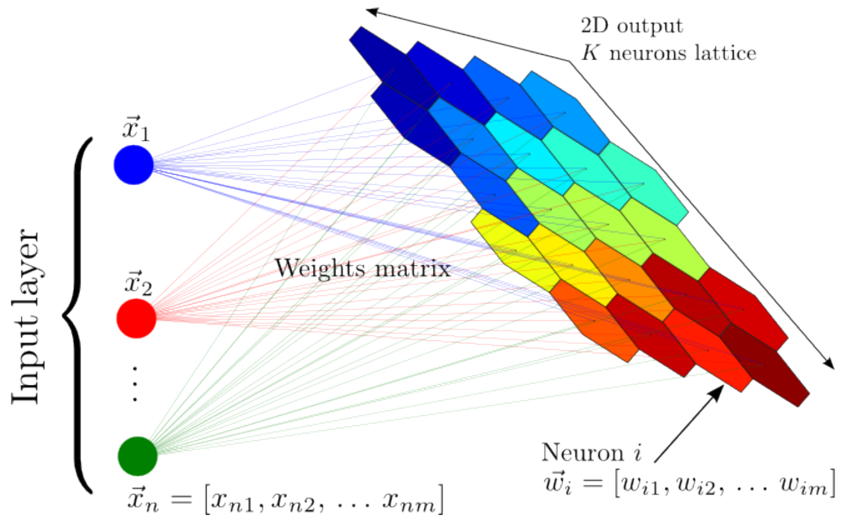
2D output
K neurons lattice
$\vec{x}_1$
$\vec{x}_2$
Input layer
Weights matrix
$\vec{x}_n = [x_{n1}, x_{n2}, \ldots x_{nm}]$
Neuron i
$\vec{w}_i = [w_{i1}, w_{i2}, \ldots w_{im}]$

Figure 1.17: A schematic representation of a self-organised map, trained to produces a non-linear mapping from a m-dimensional space of attributes to a two-dimensional neuron grid with neuron colour encoding object grouping by similarity (Credit: Matias Carrasco Kind [7]).

Smareglia, 1998; Naim et al., 1997) and general celestial object classification (Geach, 2012) . Notably, applications of SOMs have seen great success in radio morphology projection and manifold analysis (Polsterer et al., 2015) with maps of the softly clustered features relationships within a dataset, as shown in Figure 1.18. Variants such as topological SOMs with stacked multi-layer architecture (Zhao et al., 2015) have also shown successful implementation with specific subsets or regions of a trained SOM used to train successive SOMs for redshift estimation (Zitlau et al., 2016). Despite these successes, SOMs often require significant processing time to model large datasets (Kohonen, 1997).

Similar to SOM dynamic map size variants such as Growing Self-Organising Maps (GSOM) (Marsland et al., 2002), methods such as Growing Neural Gas (GNG) (Fritzke, 1995) have seen successful use in feature space modelling of early and late type galaxies to discriminate complexity and morphology (Hocking et al., 2018).

1.2.2.3 Feature Extraction in Constrained Networks

Recent advances in machine learning have seen the advent of ML systems for information extraction using *constrained* networks with deep learning (LeCun et al., 2015) concepts and *informational bottleneck theory* (Tishby et al., 2000). These unique ML systems effectively began with the Restricted Boltzmann Machine (RBM) (Smolensky, 1986) as a stochastically trained generative NN variant, and the successive Deep Belief Machine (DBM) (Hinton et al., 2006), trained for feature extraction by restricting connectivity between visible and hidden layers to form a bipartite graph. As more systems were developed with this constrained deep learning paradigm, more focus was

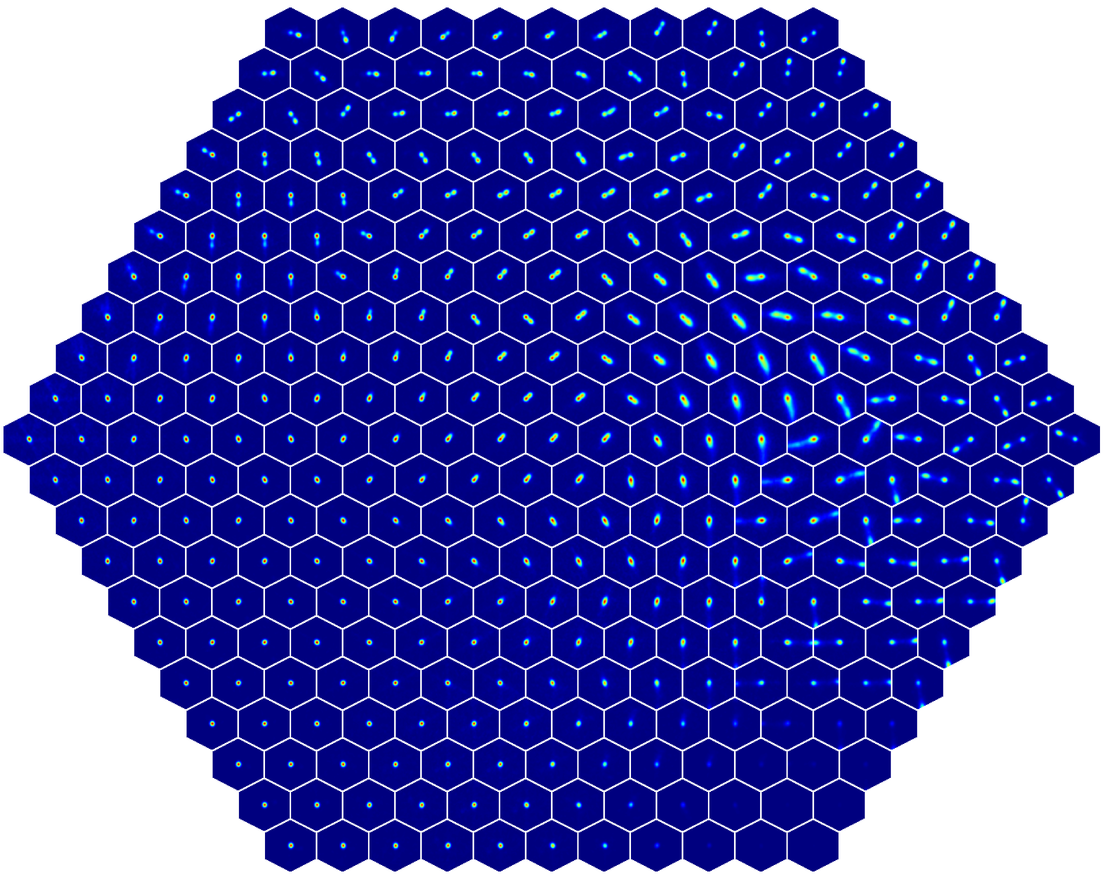
Figure 1.18: Self Organising Map projection of radio morphologies on a hexagonal grid configuration (Polsterer et al., 2015).

placed on networks such as the *autoencoder* NN variant (Ballard, 1987) designed with increased attention to information encoding (Srivastava and Salakhutdinov, 2012).

Autoencoder networks perform feature extraction through dimensionality reduction by compressing input data into a latent feature vector (Sanger, 1989). Often using a neural network structure, as shown in Figure 1.19, these networks restrict the dimensions of a central hidden layer to force the network to learn this feature vector as the most efficient representation of the input, required for restoration to its original dimensions (Schmidhuber, 2015). Ideally, an autoencoder is trained to flawlessly compress and restore this input data with no loss in fidelity. To achieve this compression, the layer configuration of an autoencoder reduces input data to a compact feature representation on the encoding side before returning it to its original form on the decoding side (Guo et al., 2016). As autoencoder loss is derived from the difference between input data and the decoded output, a label set is not required, and the network can be trained unsupervised. This loss is naturally an indicator of the performance of the network but can also be used as an outlier detection due to its sensitivity to differences between an input image and the training set (Salvato et al., 2018). More detail on the characteristic equations of the autoencoder are featured in Section 2.4.

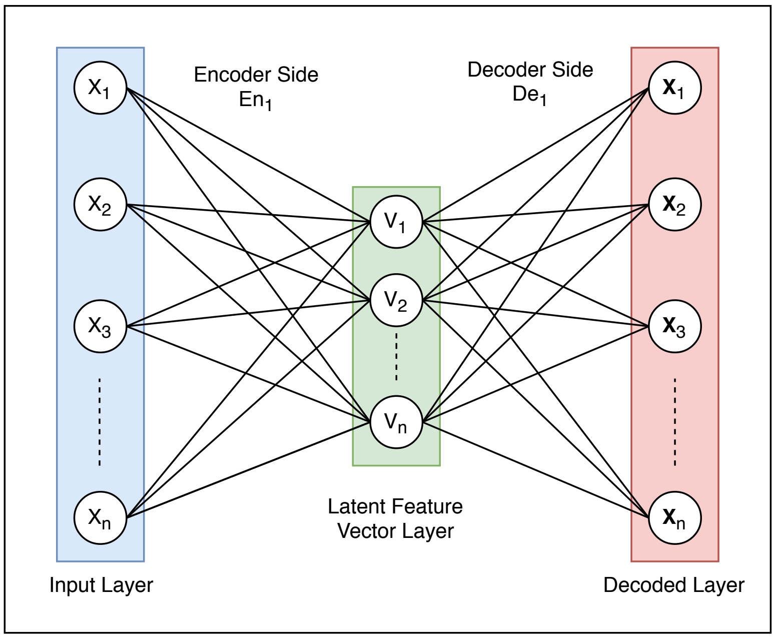

Figure 1.19: The network configuration of a simple fully-connected Neural Network autoencoder, featuring an the encoder input layer, the down-sampled latent feature vector layer and the reconstructed decoder output layer.

Autoencoders have seen further success with the addition of convolution, max-pooling for high-level abstraction and noise injection in *denoising autoencoders* for increased robustness in training (Xie et al., 2012). The inclusion of regression layers into these networks have additionally been shown to extend them to supervised learning methods. Numerous other variations have been developed since first implementations with, *stacked autoencoders* for multiple layers of abstraction, autoencoders robust to noise as *denoising autoencoders*, networks that encourage sparsity in *sparse autoencoders*, mean and variance latent layers,*variational autoencoders*.

Although the training goal of an autoencoder is regeneration after compression, interpretations of the latent feature vector are shown to reveal useful information (Hinton and Salakhutdinov, 2006; Hu et al., 2017) and are particularly useful in preprocessing data for more complex ML techniques (Larochelle et al., 2007). Using these latent vectors, other dimensionality reduction methods, namely PCA and t-SNE have been used to further reduce latent vector dimensions for visualisation of highly abstract relationships and topology in large datasets, as demonstrated in Figure 1.20. In astronomy applications, several autoencoder variations have been applied to the highly complex tasks of abstract representation extraction and cosmological modelling with weak lensing convergence maps (Singh and Bard, 2017), galaxy evolution exploration through latent vector manipulation (Schawinski et al., 2018), stellar atmospheric and spectra parametrisation (Pan and Li, 2017; Li et al., 2017), gravitational wave denoising (Shen et al., 2017) and star-galaxy separation (Hao-ran et al., 2017; Knollmller et al., 2018)

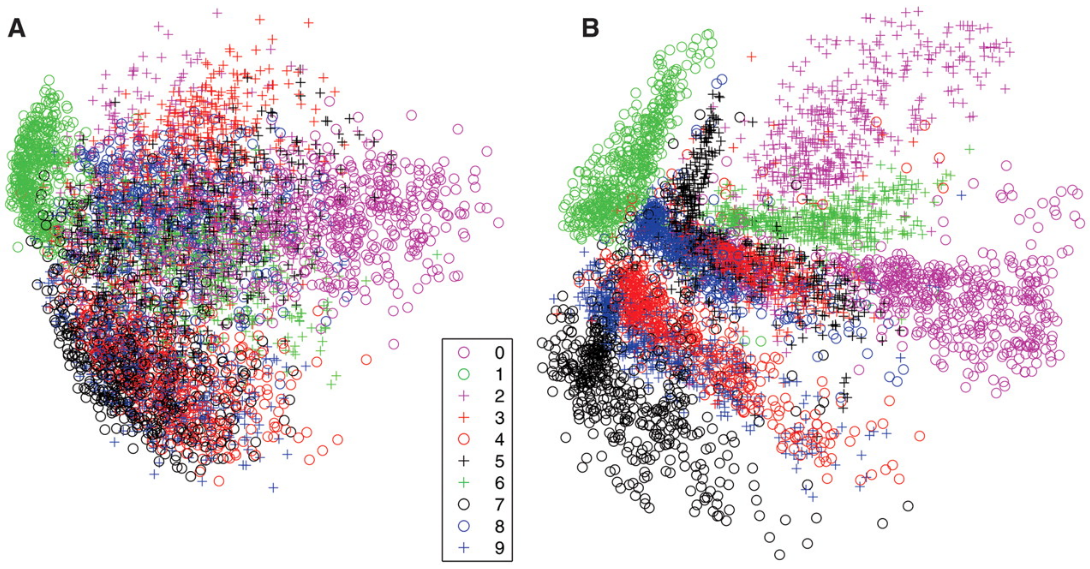

Figure 1.20: Demonstration and comparison of PCA and autoencoder dimensionality reduction on images of hand-written numbers from the MNIST dataset to reveal the underlying relationships between each class (Hinton and Salakhutdinov, 2006). The autoencoder used here reduced the dimensions of the 64×64 images down to only two dimensions for plotting and appears to produce more distance and complete clusters.

Similar architectures are exploited by other *adversarialy* trained NN variations such as the *Generative Adversarial Network* (Goodfellow et al., 2014, GAN). Generative Adversarial Network (GAN) applications have seen tremendous success in generative machine learning and image processing literature (Radford et al., 2015), although only a few cases of recent use in astronomy with separation of quasar point sources and host galaxy light (Stark et al., 2018), galaxy feature recovery and generation (Ravanbakhsh et al., 2016; Mustafa et al., 2017; Schawinski et al., 2017b,a), pulsar candidate identification (Guo et al., 2017), cosmic web simulation (Rodriguez et al., 2018) and extraction of exoplanet atmospheric features (Zingales and Waldmann, 2018).

Although the learning manifolds produced by these methods provide a new element of visual interpretation, these do not produce quantitative conclusions without techniques such as clustering.

1.2.2.4 Hierarchical Clustering

The HC approach aims to quantify features within learning manifolds by grouping data into hierarchical classes. One of the most commonly used HC algorithms, *K-Means* (MacQueen, 1967) seeks to group objects by assigning each input data instance to a discrete cluster based on a given distance metric (Lloyd, 1982). An example of this clustering is shown in Figure 1.21, where distinct groups are split based on these distances around a cluster centre point.

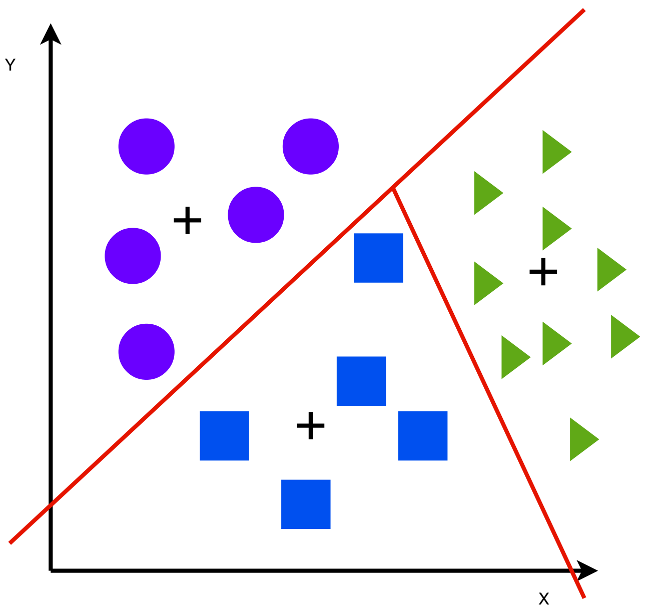

Figure 1.21: Demonstration of HC using the K-means algorithm, where the manifold is split into groups (colours, purple, green and blue) as a function of a given distance measure. Cluster centre points are shown as with a black plus and a basic decision surface shown as the red vectors.

HC methods such as K-means address the issues faced by many astronomers attempting to analyse incomplete or higher dimensional data and highlight the potential for unsupervised methods to operate outside the reaches of human understanding. Consequently, K-means has seen widespread use in astronomy with notable examples ranging from clustering and prediction of partially complete astronomical data (Wagstaff and Laidler, 2005), delineation and partitioning of stellar seismic source zones (Weatherill and Burton, 2009), detection of metal-poor galaxies (Morales-Luis et al., 2011; Snchez Almeida et al., 2016), stellar chemical abundance classification (Hogg et al., 2016) and general unsupervised classification of variable stellar objects, (Ng and Huang, 1999), gamma-ray bursts (Chattopadhyay et al., 2007) and optical spectra from the SDSS (Snchez Almeida et al., 2010). As K-means is capable of clustering learning manifolds, it has been successfully combined with other dimensionality reduction methods, such as the discussed SOM (MacDonald and Fyfe, 2000) and GNG methods (Hocking et al., 2018) to quantise projected groupings, as shown in Figure 1.22 with unsupervised galaxy morphology clustering.

1.2.3 Astronomical Anomaly Detection and Complexity Separation

Anomaly detection is a subset of ML classification focussed on identifying uncharacteristic behaviour compared to a learned model (Chandola et al., 2009). This is a subset of the more general operation of complexity separation, where varying de-

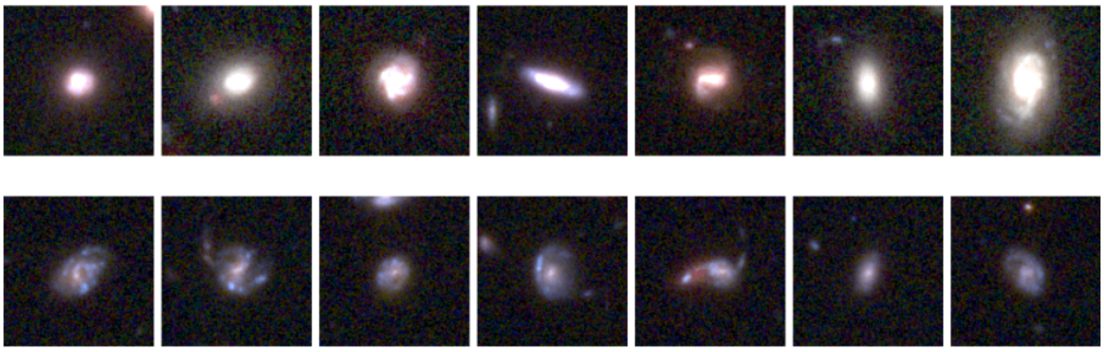

Figure 1.22: K-means clustering of Hubble Space Telescope Cosmic Assembly Near-IR Deep Extragalactic Legacy Survey (CANDELS) into distinct groupings (row-wise), ordered left to right by their similarity to the 'average' classification in the parameter space of the group (Hocking et al., 2018) (cropped from image rows 4 and 3 respectively from the bottom in Figure 12 for size constraints).

grees of 'anomalousness' is established to distinguish observations based on their relative complexity. Uncharacteristic behaviour and these varying degrees of complexity in observations, be it an object or process, may indicate interesting phenomena, inconstancies in astronomical understanding and possibly mark a discovery. Current efforts rely on researchers to make these discoveries while exploring a hypothesis. It has been shown, however, that most of the major discoveries in astronomy have been serendipitous (Ekers, 2009) as in Figure 1.23. These discoveries tend to be unplanned and found instead while pursuing an unrelated hypothesis (Norris, 2017a). Consequently, anomaly detection in astronomy has become a task of searching for discoveries. This paradox of anticipating serendipitous discovery seen has lead astronomers to ML methods to process the true depths of high volume observational and circumvent the human bias' and assumptions that prevent discovery.

Despite interchangeable terminology of 'anomaly', 'outlier' and novelties as interesting outliers, anomalies are generally organised by three categories (Chandola et al., 2009):

- *Point Anomalies* are single instances of uncharacteristic behaviours, such as unexpected spikes seen in time series data,
- *Contextual Anomalies* are a series of single or continuous events that under a different context would be normal,
- *Mixed Anomalies* mixed anomalies are a subtle combination of both point and contextual anomalies as a superset.

Several previously discussed supervised ML methods have been successfully applied to detecting point, contextual and mixed anomalies in astronomy using a training set containing known anomalies. These ML methods, along with basic statistical

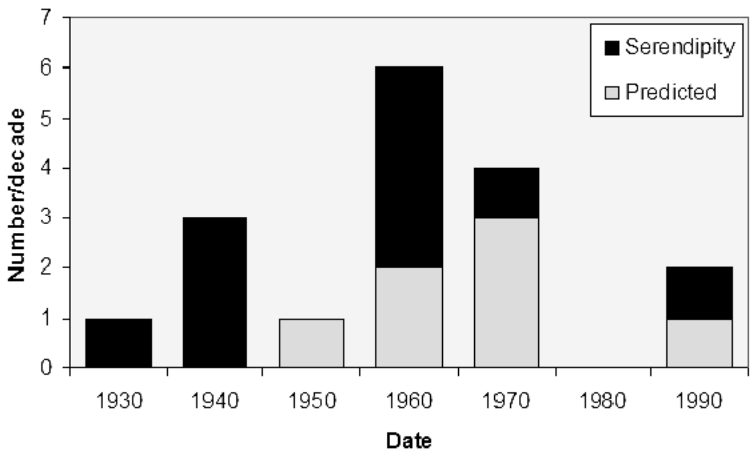

Figure 1.23: Comparison of the number of planned and serendipitous key discoveries made in modern astronomy (Ekers, 2009).

inference through regression (Kuo et al., 2018), have seen successful use in astronomical anomaly detection with many of the previously discussed methods demonstrating success, namely automatic anomaly detection and classification of time variable x-ray sources with a RTF (Lo et al., 2014), CNN transient source detection (Wright et al., 2010), and simple outlier detection through PCA dimensionality reduction (Dutta et al., 2007).

Although these methods are successful, they require pre-labelled training sets, effectively allowing these systems only to classify that which has already been discovered. This pursuit found unsupervised methods ideal in many cases where the ML system can be left to determine outlier classes while making few assumptions about input data and reducing the human bias inherent in training sets (Torralba and Efros, 2011). This has been demonstrated through the use of an unsupervised RTF system to quantify anomalous sources with a "weirdness score" (Baron and Poznanski, 2017), isolating counterparts of un-associated Fermi gamma-ray sources with K-means clustering (Mirabal et al., 2010) and notably, the previously discussed SOM data exploration tool (Polsterer et al., 2015) with a complexity separation method based on SOM umat neuron Euclidean distance and GNG-K-means hybrid (Hocking et al., 2018) morphology clustering methods.

Additionally, previously discussed multi-wavelength cross matching has also provided an excellent basis for outlier detection. This can be seen with observations of a source across multiple frequencies not necessarily displaying anomalous features, but when analysed collectively, unusual relationships and asymmetry can be found (Banfield et al., 2015).

1.3 Summary

Modern astronomy is pursing new instrumentation aimed at pushing the boundaries of sensitivity and resolution. The typical solution to improving these factors is to build telescopes with more sophisticated detection apparatus and a much larger emission collection area. In radio astronomy, these requirements have increased focus in radio interferometry techniques where observations from multiple radio telescopes can be combined to drastically increase the resolution. These advancements are not without caveats however, as these new astronomical instruments and technologies have increased in scale and complexity, so has the data that these instruments collect.

The sheer volume and complexity of current highly sensitive and expansive surveys is a problem for astronomers, particularly when traditional techniques for typical tasks such as classification and outlier detection become insufficient for this new volume of multi-wavelength data. This effect is compounded in radio astronomy with the advent of new large-scale radio telescope arrays. In this new modern era of 'big-data' astronomy, machine learning is becoming a key focus in an effort to combat the inundation of highly complex observational data.

The Machine Learning (ML) technique is now a common approach, where a system is 'trained' to model the characteristics of a dataset and predict the qualities or classification of a training set or new unencountered data. The ability of ML methods to process highly complex data has attracted the zeal of many fields and the modern astronomer confronted by the challenges of big-data. ML astronomy currently focuses on three main tasks of classification and regression, higher-dimensional analysis with clustering and visualisation, and anomaly detection.

As a salient task in astronomy, ML classification is commonly applied to mapping objects to a class based on observed physical differences such as morphology, with inference used to map continuous training variables to some continuous output. Using labelled data with supervised training and raw datasets with the unsupervised training paradigm, these tasks have seen successful implementation with several ML methods ranging from, the k-nearest neighbour algorithm, support vector machines, decision tree learning and most notably neural networks and the addition of convolutional receptive fields.

Although current ML solutions in astronomy have been hugely successful, these methods are still focussed on the development of new methods or optimisation of old approaches, despite the newly increased scale and complexity of astronomical data. Developing new techniques to process this data in its current volume or simple brute forcing of computation power is not directly targeting the problem of scale. Information extraction, dimensionality reduction and resulting visualisation have been utilised effectively to delineate data and reduce problem spaces while retaining relevant information. Additional layers of abstraction are often provided with clustering techniques such as the K-means algorithm as a hierarchical clustering method to associate points in these delineated feature spaces to a class.

Although these methods are capable of analysing existing data, they may not be suitable for discovery. Anomaly detection and complexity separation is a subset of classification that aims to bridge this gap by focussed on identifying uncharacteristic behaviour compared to a learned model. Uncharacteristic behaviour in an observation, whether it be an object or process, may indicate inconsistencies in astronomical understanding and possibly mark a discovery. Current efforts rely on researchers to make these discoveries while exploring a hypothesis, although most major discoveries in radio astronomy have been serendipitous and found instead while pursuing an unrelated hypothesis. This pursuit has seen the implementation of several techniques to locate unexpected, most notable of which utilise unsupervised training to locate discoveries not included in current training sets.

Astronomy requires the automatic classification, outlier detection and speed that machine learning can offer. The ideal ML solution to astronomy tasks is a combination of methods with robust classification, enhanced efficiency through information extraction, powerful higher dimensional analysis and anomaly detection with the unsupervised training paradigm. By operating strictly in this unsupervised manner, these methods require significantly less human intervention and mitigate the time usually required to label datasets manually. By leaving the analysis in the hands of the networks, the element of human bias is all but removed.

Unsupervised learning methods such as SOMs have demonstrated great success in unsupervised morphology analysis but at the cost of computing time. A sophisticated but efficient method such as the autoencoder NN method would be ideal as a SOM training preprocessor reduce the complexity and scale of training data. The addition of hierarchical clustering of the SOM as seen in literature will provide the final means to classify morphological groups and anomalies projected into the autoencoder trained SOM space. Sizeable ML astronomy datasets such as the RGZ provide an excellent testing ground for these methods.

1.4 Research Questions

I pose several questions in this thesis, based on the literature discussed within this Chapter:

i. Is unsupervised machine learning a practical solution to clustering, data exploration and complexity separation to large datasets typical of big-data astronomy?

ii. Can dimensionality reduction methods be used to delineate the complexity and volume of astronomical data for exploration and information extraction?

iii. Will an autoencoder produce efficient information extraction and dimensionality reduction of radio-astronomical images?

iv. Can an affine invariant SOM be trained on compressed information extracted from an image dataset instead of the raw images? Moreover, what is the effect of this training? And how can the relationships of encoded image representation be visualised?

v. Is visualisation in ML an effective method of data exploration and information extraction?

1.5 Thesis Aims

I aim to use a novel unsupervised clustering and classification system that combines HC with a SOM and convolutional autoencoder to investigate and answer the posed research questions.

i. Determine whether an autoencoder-SOM hybrid system can provide a deep understanding of the morphological relationships of radio-astronomical images and be used as a tool for complexity separation, visualisation and data exploration.

ii. Investigate latent feature vector extraction with convolutional autoencoders as a method to reduce the volume and complexity of radio-astronomical data.

iii. Determine the effects and efficiency of training a SOM on compressed latent feature vectors of images extracted using a convolutional autoencoder and develop an approach to visualise the SOM learned relationships of these latent feature vectors.

iv. Employ HC of SOM learned weights to cluster and segment radio-astronomical images using autoencoder trained SOMs.

v. Analyse the effect of autoencoder random rotation training augmentations on SOM affine transformation invariance.

Chapter 2

Methodology

This Chapter outlines the methods I used to investigate unsupervised clustering and classification as a ML solution to the challenges of big-data astronomy. In my approach, I implemented a K-means clustered SOM trained on autoencoder latent vectors. This method is based on my research questions of Section 1.4 and subsequent aims of Section 1.5.

My paper *Radio Galaxy Zoo: Unsupervised Clustering of Convolutionally Encoded Radio-astronomical Images*, (Ralph et al., 2018) featured in Appendix , formed the preliminary stage of this method. This paper is referred to as Ralph et al. (2018) throughout the remainder of this thesis.

This Chapter explains and justifies the methods I used in this thesis. Initially the hardware and software are outlined (Section 2.1), followed by the data used in this investigation (Section 2.2), the preprocessing method (Section 2.3), autoencoder implementation and tuning (Section 2.4), SOM implementation and tuning (Section 2.5) and the final system evaluation (Section 2.6). The results of these methods are demonstrated and discussed in Chapter 3.

2.1 Hardware and Software

The system outlined in this methodology was entirely implemented with the PYTHON programming language. I chose this language for fast development at the cost of computation efficiency that may be gained from C. I used several PYTHON libraries in this system:

- OPENCV (Bradski, 2000), PILLOW, and MATPLOTLIB (Hunter, 2007) for image processing and display.
- Google TENSORFLOW (Abadi et al., 2016) to construct the NN autoencoder. TensorFlow features the TensorBoard browser-based control panel to display plots

of the weights, bias and cost. This feature is only used for basic testing; all figures are produced using Matplotlib.

- SciPy (Jones et al., 2001), NumPy (Oliphant, 2006) and Scikit-Learn (Pedregosa et al., 2011) for high-level implementations of various algorithms.
- SomoClu (Wittek et al., 2013) to implement the SOM. I built an extensive visualisation package to work with SomoClu, given the original package included limited visualisation features.

I developed and tested the overall system using the 'big-data client' of Western Sydney University Kingswood big-data laboratory, featuring an Intel(R) Xeon(R) CPU E5-2650 v4 at 2.20GHz, with 32GB of Random Access Memory (RAM). All tests were conducted using Central Processing Unit (CPU) resources. Although Graphics Processing Units (GPUs) were available and implemented in the system, I aimed to demonstrate the ability of this system using minimal computational resources and ideally be implementable by most institutions and researchers.

2.2 Radio Galaxy Zoo Dataset

In this investigation, I use the first DR1 (Wong, 2018) of the RGZ project (Banfield et al., 2015) (outlined in Section 1.2.1) for training and validation of the ML systems I developed in this thesis. This was selected as one of the largest classified radio-astronomical image sets currently available.

The majority of radio image data in the RGZ dataset comes from the 1.4 GHz FIRST survey (Becker et al., 1995) version 14 March 2004. This is combined with mid-IR images at 3.4 μm from the Wide-field Infrared Survey Explorer (WISE) survey. FIRST covers over 9000 square degrees of the northern sky down to a 1 σ noise level of 150 μJy beam^{-1} at 5$''$ resolution. WISE is an all-sky survey at wavelengths 3.4, 4.6, 12, and 22 μm with 5 point source sensitivity in unconfused regions of no worse than 0.08, 0.11, 1.0, and 6.0 μJy (Wright et al., 2010). Only the 3.4 μm WISE data is used in RGZ.

My focus in this thesis is the radio-astronomical data of the DR1. The RGZ DR1 dataset contains 75,641 classified radio images and associated attributes. The images included in the dataset are Flexible Image Transport System (FITS) images. These are a standard astronomy application image type that represent images with a floating-point representation and are rich with meta-data. The main attribute I focus on in this thesis is the hand labelled component and peak counts for every resolved source. The catalogue also contains the number of participants who labelled each image and the consensus agreement between those participants (Banfield et al., 2015). I encoded these labels as components-peaks, e.g. an image with a single component and two peaks are labelled with a class '12'.

Table 2.1 and Figure 2.1 summarises the distribution of these classifications within the dataset. This Figure shows the dataset is largely unbalanced toward simple sources. The largest fraction of the dataset contains images labelled '11' (single component single peak source) at 61%, with the remaining data set comprised of '12' (single component double peak) at 12%, complex '22' sources at 14% and the remainder as rarer sources. For this investigation, I consider all '11' point sources as considered, while all other images as 'complex'. Additionally, I consider classes that make up less than 2% of the dataset as outliers.

RGZ Label	Population Division	Category
11	0.6110	Simple
12	0.1533	Complex
13	0.0153	Complex
14	0.0020	Anomalous
15	0.0003	Anomalous
16	0.0001	Anomalous
22	0.1438	Complex
23	0.0195	Complex
24	0.0028	Anomalous
33	0.0340	Complex
34	0.0053	Anomalous
35	0.0008	Anomalous
36	0.0003	Anomalous
44	0.0068	Anomalous
45	0.0014	Anomalous
46	0.0004	Anomalous
55	0.0020	Anomalous
56	0.0005	Anomalous
57	0.0002	Anomalous
67	0.0002	Anomalous

Table 2.1: RGZ DR1 classes by population

Illustrated in Figure 2.2, classified samples range from the simple 11 class 'point sources', extended multiple peak and component sources such as the 33 class, in addition to highly anomalous sources such as the 67 class that comprise only 0.0096% of the dataset. These samples demonstrate the variety of morphologies in the dataset and the often subtle differences to distinguish often confused sources such as '11' and '12' sources.

The consensus distribution across the dominant classes of the RGZ DR1 is described in Figure 2.3 by a 10,744 subset of the DR1 with sources above 0.6. As expected, increased complexity with higher peak and component counts results in a

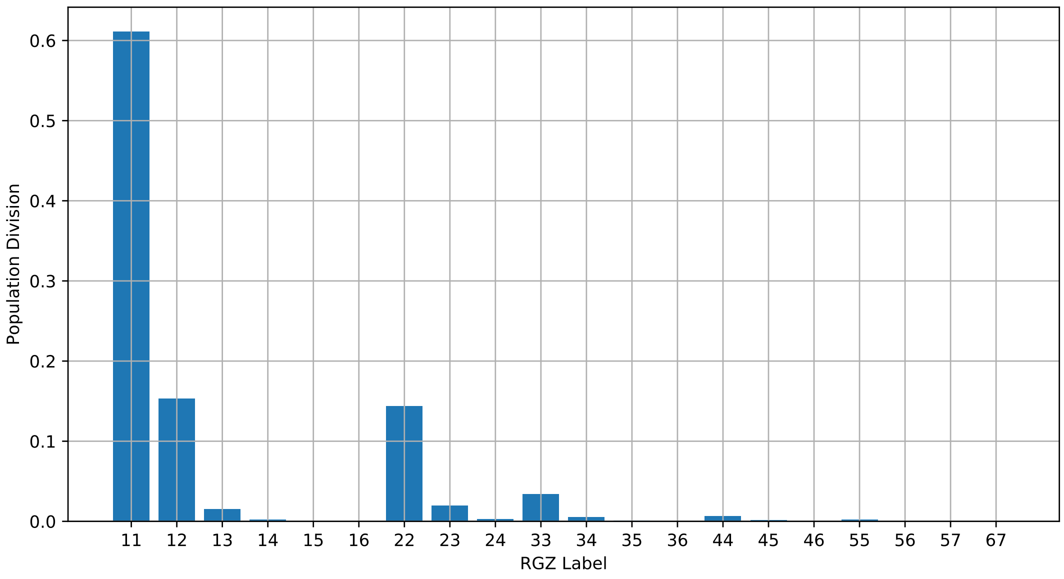

Figure 2.1: Radio Galaxy Zoo Data Release 1 class distribution.

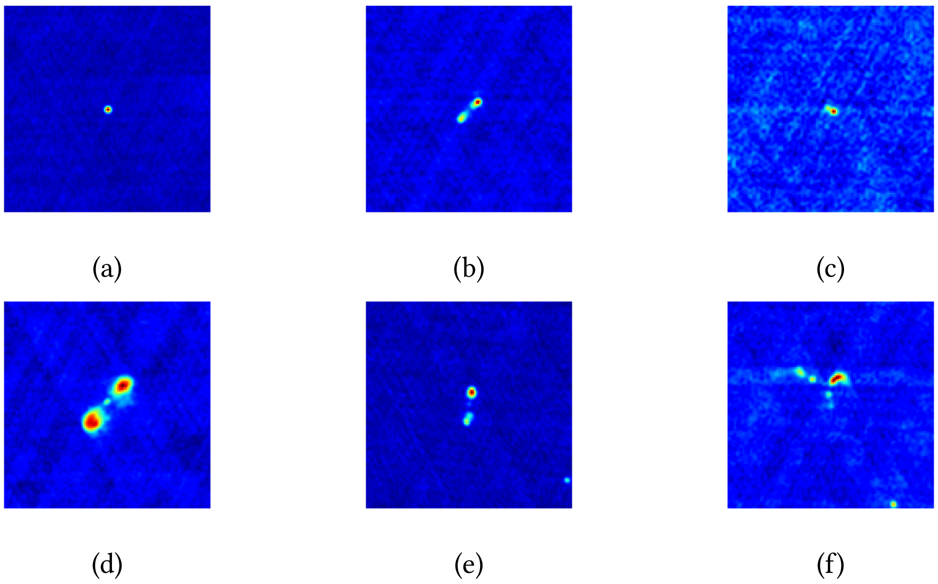

Figure 2.2: Six representative samples of the RGZ dataset. (a) simple point source as class 11, (b) class 12, (c) 11 source with considerable noise, (d) class 13 source as a classic triple radio AGN, (e) class 12 with an additional component in the bottom right and (f), a highly anomalous 55 source. I use these representatives throughout this thesis to demonstrate and compare the effects that each step of the system has on a variety of RGZ images

higher consensus than single peak and component sources. This trend indicates that the human classifiers have a greater tendency to disagree on the classification of complex sources. These possibly disputed labels highlight the potential inaccuracy inherent in the manually classified image labels of the RGZ DR1. Papers such as (Wu et al., 2018), use a subset of the RGZ data containing only sources with a consensus ≥ 0.6.

However, I perform my testing using the full dataset without this consensus filtering to retain as many ‘difficult’ to classify images as possible, which may represent anomalous sources. Consequently, validation in the final stages of this thesis will likely be affected by low consensus scores which potentially indicate images with false or disputed labels.

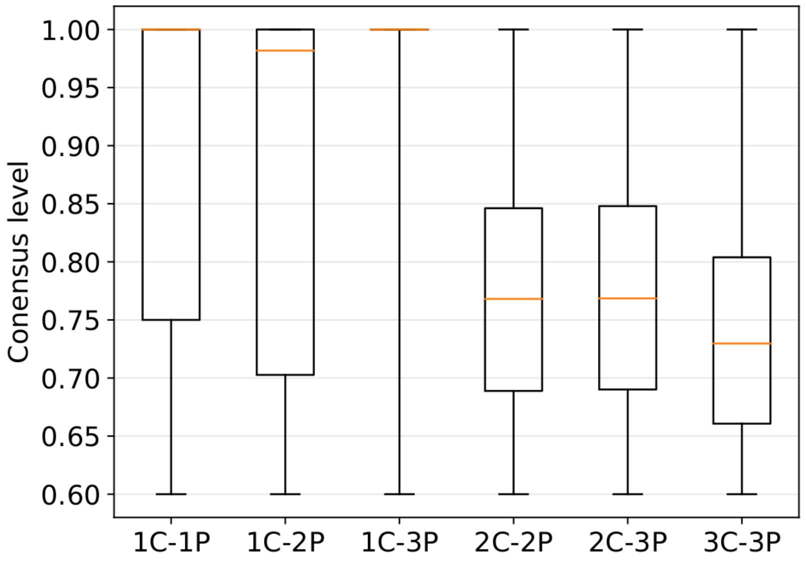

Figure 2.3: The distribution of the consensus level across six morphology classes in the data set that consists of 10,744 RGZ subjects selected from the DR1, where the authors encode 11 as 1C-1P, where C is the component count and P is the peak count. The whiskers above and below the box represent the maximum and minimum CL (fixed at 0.6 by the first criterion). The box itself spans the third and the first quartile consensus level. Note that since 80% of 1C-3P sources have a consensus level of 1.0, its box is reduced to a single horizontal line when its inter-quarter range becomes 0. The horizontal (orange) line inside each box is the median (Wu et al., 2018).

Despite the quality of images in the RGZ dataset, many images contain different background noise properties, instrumental noise and artefacts. I use the methods outlined in the following preprocessing Section to filter these features for improved ML training.

2.3 Preprocessing

The performance of ML training methods is contingent on the quality of training data. For a learning method to effectively map data to a class, it is best to remove features not used to distinguish classes. Particularly in the case of unsupervised methods, dominant features are typically recognised as the distinguishing characteristics in the dataset. If these features are not unique to the desired classes, the system may classify the dataset only according to these features.

Common solutions involve normalising images with the same pixel histogram (intensity distribution) or Root Mean Squared (RMS) noise, as changes in these image statistics may be recognised as a distinct feature and lead to classification based only on these measures. Radio images are also often contaminated by remnants of the telescope Point Spread Function (PSF). Termed 'side-lobes', this contamination commonly forms a significant component of the feature space of the RGZ training set.

To ensure proper ML training, I implemented two preprocessing methods to filter the RGZ images. These preprocessing methods were evaluated by their processing time and ability to filter noise while preserving critical astronomical features such as morphology and peak-component counts.

2.3.1 Mask filtering with Adaptive Otsu Thresholding

The first preprocessing method I developed in this thesis followed a typical image processing methodology. In this method, radio images are converted to an 8-bit Portable Network Graphic (PNG) image, filtered with an adaptive threshold mask and component count checking. This technique is used to remain agnostic to input images types, given many image processing and ML image sets are also 8-bit .PNG images. This generalisation of this method is at the cost of sacrificing the dynamic range of pixel intensity available in .FITS radio images. I implemented this system using the OpenCV and Numpy libraries. This preprocessing cleans images using the following procedure:

1. Input FITS RGZ images are converted to 8-bit grey-scale .PNG images. Image colour in the original file is a false colour representation. No colour channels are removed here, as .FITS images do not contain such channels, but a floating-point broadband intensity value.

2. Grey-scaled images are cropped to 120x120 for reduced data size to improve storage and processing times, in addition to ensuring consistent dimensions across the whole dataset. This cropping size was chosen carefully to avoid cropping out features.

3. Gaussian filtering is used to spread the intensity distribution of prominent features. The input image resolution is lowered using a 3×3 blurring kernel to preserve the local neighbourhood of bright regions in the next binary thresholding step.

4. Otsu adaptive thresholding of blurred image. Otsu Binarization (Maaten and Hinton, 2008) is used to segment background noise from the desired astronomical features. This method separates pixels $p(i)$, of an image into an object and a

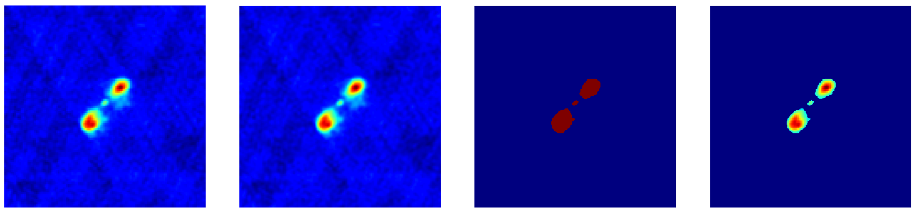

Figure 2.4: Demonstration of the adaptive filtering preprocessing method, where the input image (a), is blurred using the Gaussian filter (b). An adaptive threshold mask is created from this image using the Otsu algorithm (c). This image is then used as a segmentation mask to create the final image (d), which retains only the key astronomical features of the original image.

background class by a threshold t. The grey level histogram is normalised and regarded as a probability distribution:

$$\sigma^2{}_w(t) = \omega_0(t)\sigma_0{}^2 + \omega_1(t)\sigma_1{}^2(t) \tag{2.1}$$

Where $\sigma_0{}^2$ and $\sigma_1{}^2$ are the variance of the foreground and background classes with probability weight ω_0 and ω_1 computed from L histogram bins:

$$\omega_0(t) = \sum_{i=0}^{t-1} p(i) \tag{2.2}$$

$$\omega_1(t) = \sum_{i=t}^{L} p(i) \tag{2.3}$$

5. Images with an Otsu threshold mask containing more than five binary regions are discarded. Image with more than five components is too noisy or corrupted as the RGZ dataset does not include any objects with this many components.

6. Masking of the original image using Otsu binary Gaussian blur map to segment the most dominant features and surrounding regions.

7. Save images to disk as a .PNG

2.3.2 Sigma Clipping and Noise Injection Preprocessing

I tested an additional method in this system as a more traditional astronomy based method. This method is the preprocessing method of (Galvin et al., 2018) with results shown in Figure 2.5. This approach corrects blank pixels in images at the edge of the FIRST image mosaic, sigma clips noise and normalises pixel intensity.

1. Blank pixel regions found in images close to the edge of the FIRST mosaic are corrected. This correction replaces these masked values with a random sample of the mean and standard deviation of valid pixels around the outer edge region of the image (assuming a normal distribution). These samples are extracted from the outer 85% region of the image with few astronomical features to sample the background noise adequately (Figure 2.5, b1).

2. Noise is removed, and background flux is corrected with sigma clipping (Figure 2.5, c1). This operation subtracts the mean background pixel value and scales all pixel intensities below 1 σ to zero.

3. Intensity scaling is applied to normalise the global intensity of each image (Figure 2.5, d1).

4. All images are additionally cropped for this thesis to 120x120 from the centre to reduce the dataset size while preserving salient features.

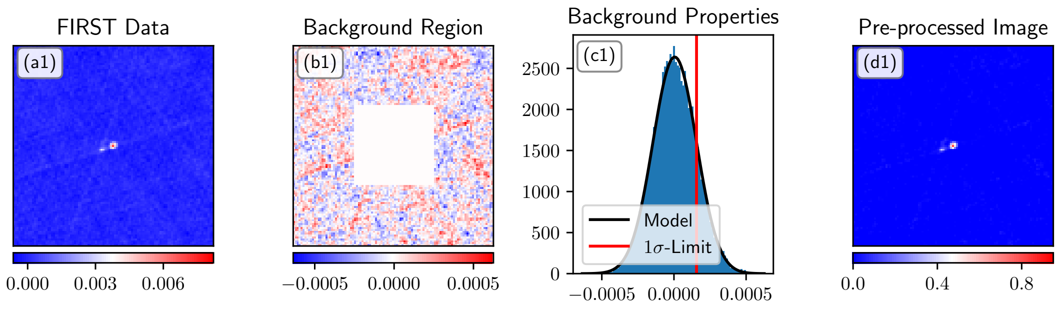

Figure 2.5: Demonstration of each step of the sigma clipping and noise injection pre-processing method from (Galvin et al., 2018), where the inner 15% of the input image (a1), is taken segmented (b1), to determine the background noise distribution (c1) which is used to calculate the 1 σ pixel cut-off for cleaning. This sigma clipped image is then cropped, and contrast corrected using global histogram normalisation, as shown in the final pre-processed image (d1).

2.4 Convolutional Autoencoders

The challenges of astronomy in the modern big-data era have resulted in significant research into Machine Learning solutions to classification, anomaly detection and higher dimensional analysis through data reduction (discussed within Section 1.2). In this thesis, I aim to investigate data reduction of radio-astronomical images using unsupervised feature extraction (outlined in Section 1.5). This methodology demonstrates my approach to analysing the data scale and complexity reduction of radio-astronomical images when reduced to a compressed latent feature vector using a convolutional autoencoder. This Section provides a detailed explanation of this autoencoder, how I chose a specific architecture and its use in delineating RGZ images. Using the final optimised autoencoder, I determine the efficacy and processing time reduction of a K-means clustered SOM trained on autoencoder extracted latent vectors of the RGZ dataset. With this method, I demonstrate this system as a solution to unsupervised classification and anomaly detection.

2.4.1 General Autoencoder Architecture

Introduced in Section 1.2.2.3, an autoencoder is a network (often a NN variant) that compresses input data to lower dimensions for later reconstruction. A typical autoencoder contains three main elements, the input encoder side, the hidden latent vector layer and the output decoder side, as shown in Figure 1.19.

The general framework for an autoencoder can be summarised as minimising an overall distortion or prediction error measure, $E(A, B)$ by finding an $A \in \mathcal{A}$ and $B \in \mathcal{B}$ when transforming an input vector $x \in \mathbb{F}^n$ into an output vector $A \circ B(x) \in \mathbb{F}^n$, (Baldi, 2012) as in Equation 2.4:

$$min\ E(A, B) = min_{A,B} \sum_{t=1}^{m} E(x_t) = min_{A,B} \sum_{t=1}^{m} \Delta(A \circ B(x_t), x_t) \tag{2.4}$$

Where $\mathcal{A}$ is a transformation class of encoding functions from original input to latent vector $\mathbb{G}^p$, to $\mathbb{F}^n$ and $\mathcal{B}$ is the transformation class of decoding functions from the latent vector $\mathbb{F}^n$ to the original dimensions $\mathbb{F}^n$. Given $\mathbf{X} = \{x_1, ..., x_m\}$ is a set of m training vectors in $\mathbb{F}$ as both network input and training target (in the case of this thesis implementing an auto-associative autoencoder, target $y_t = x_t$). $\mathbb{F}$ and $\mathbb{G}$ are sets of positive integers n and p, as the original and latent space dimensions. These integers are bound by $0 < p < n$ for a constrained autoencoder with fewer dimensions in the hidden latent feature layer than the input dimensions.

In this thesis, I define the Mean Square Error introduced in Section 1.2.1.4 as the distortion function Δ (often interchangeably referred to as cost, error or loss), given by Equation 2.5.

$$\Delta = \frac{1}{n} \sum_{t=1}^{m} (A \circ B(x_t) - x_t)^2 \tag{2.5}$$

Using these concepts, I implemented a compact convolutional autoencoder, outlined in the following Section 2.4.2.

2.4.2 General Network Architecture

The initial autoencoder architecture and training conditions I used in this thesis and my paper Ralph et al. (2018), were selected to ensure the network was both compact and produced a single channel latent vector with sufficient data compression. Additionally, I implemented convolutional and pooling layers to allow the network to analyse high-level features with a degree of scale invariance (based on affine invariant training concepts introduced in Section 1.2 and NNs of Section 1.2.1.4).

The general architecture of this first autoencoder as shown in Figure 2.6 contained four hidden layers, with three convolutional layers and a pooling layer in the encoder and mirrored in the decoder. In this configuration, I implemented a single deep convolutional layer to extract high-level features for the latent vector. I also included a convolutional layer with a single filter either side of the deep convolutional layer. This allows the deep convolutional layer to analyse these features in multi-channel fashion while accepting single channel data and producing a sufficiently compressed single channel vector for the max-pooling layer.

In my approach, the autoencoder latent vector resides between the encoder and decoder network as the output of the encoder max-pooling operation. As a mirrored architecture of the encoder, the decoder network input layer is a de-pooling operation. The output of this layer is fed to the successive convolutional layers and fully connected to the output layer as the network reconstruction. My system implements this pooling to allow the autoencoder to be relatively invariant to scaling effects. I kept this overall network configuration constant throughout the training and testing of this thesis.

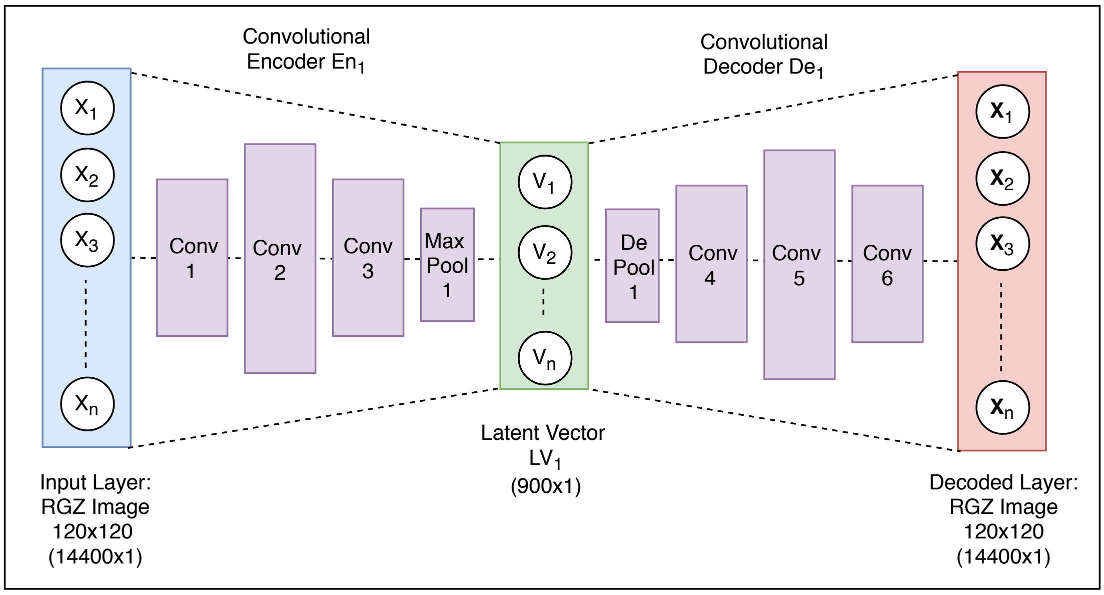
Convolutional Encoder En1
Convolutional Decoder De1
X1
X2
X3
Xn
Conv 1
Conv 2
Conv 3
Max Pool 1
V1
V2
Vn
De Pool 1
Conv 4
Conv 5
Conv 6
Latent Vector LV1 (900x1)
Input Layer: RGZ Image 120x120 (14400x1)
Decoded Layer: RGZ Image 120x120 (14400x1)

Figure 2.6: The initial architecture of the autoencoder used in this thesis, accepting 120×120 RGZ images into three convolutional layers and a max-pooling layer in the encoder, reduced to a central hidden latent feature vector layer with dimensions 900×1 and up-sampled through a mirrored architecture in the decoder to the original input images dimensions.

2.4.3 Autoencoder Training and Validation Dataset Division

In my method, the autoencoder training set is a randomly sampled 10,000 image subset of the full 100,000 imagesRGZ DR1 dataset division. The validation set is the remaining 90,000 images exclusive to the training set. This division ensures the autoencoder is evaluated only with images not previously used for training. Ideally, this validation set indicates the generalised modelling quality of the autoencoder. For this reason, all images encoded into the output SOM training and validation latent vector set are also from this validation set. As outlined in Section 2.2, processing images regardless of consensus also allow the autoencoder to be trained on images that are difficult to manually classify, which are likely to be highly complex or anomalous.

2.4.4 Network Optimisation and Hyper-Parameter Tuning

Outlined in Sections 1.2.1.4 and 1.2.2.3, the performance of a NN autoencoder in training and testing is dependent on the layer configurations and training hyper-parameters. I chose these intuitive options experimentally as a proof of concept. As a result, I developed a compact convolutional autoencoder architecture. This was featured in my paper Ralph et al. (2018). As my investigation progressed past the paper, more in-depth testing and result refinement were required to prove the actual efficacy of this method with RGZ data. To achieve this end, I used the datasets specified in

Section 2.4.3 to test the parameters of following Sections 2.4.2 - 2.4.5 to locate the best performing combination of hyper-parameters and architectures.

2.4.4.1 Convolutional Layers

Convolutional filters in a NN form receptive fields that take into account the spatial structure of input image features (Section 1.2.1.4. These filters operate as a discrete convolution operation, formalised in Equation 2.6 as the convolution function f and kernel function g with support on $\{-M, ..., M\}$ over image (x, y)

$$(f * g)(x, y) = \sum_{m=-M}^{M} \sum_{n=-N}^{N} f(x-n, y-m)g(n, m) \tag{2.6}$$

The dimensions, stride and amount of convolutional kernels bear consequences for the scale and amount of features extracted from input data. Features significantly smaller than the coverage of the filter size and its stride will likely produce a meagre response and be overlooked. Conversely, a kernel coverage significantly smaller than salient image features may not map large-scale features. Ideally, the number of kernels, their size and stride are selected to mitigate under-fitting and capture enough features to create a latent space representation fit for regeneration. By adjusting these kernel dimensions, the sensitivity of convolutional layers to particular features can be tuned. Additionally, the amount and depth of high-level features that are analysed by these layers can be tuned by adjusting the number of filters in each layer, as how many convolutions the layers effectively perform.

I experimentally chose the initial filter dimensions, and hyper-parameters for the paper Ralph et al. (2018). In these tests, I implemented a convolutional filter size of 3×3, given most features are roughly 3×3 pixels in the original image. Similarly, I applied a filter stride $[S_{batch}, S_x, S_y, S_{channel}]$, of $1 \times 2 \times 2 \times 1$. In this notation, all batches S_{batch} and channels $S_{channel}$ are taken into account (therefore value of 1 for all batches, where every batch is processed and only a single channel input is processed) and a stride of 2 pixels for directions x and y are taken to sufficiently sample the relatively small image space of the RGZ images and to reduce the original input dimensions. These stride concepts are illustrated in Figure 2.7.

Equal value for stride in each direction was chosen for simplicity in interpretation and to preserve feature structure. A stride smaller or greater than this may under-fit the training data and would likely considerably lower the latent vector dimensions, potentially increasing training time and lowering accuracy. My tuning method kept this stride constant due to these reasons and given the sheer number of additional testing combinations it may introduce. Additionally, I implemented zero or 'same' padding (also shown in Figure 2.7) to ensure consistent output dimensions with any changes in filter kernel dimensions. This padding replaces pixels not covered by a full kernel stride with a value of zero. I use this measure given it is unlikely to produce image artefacts as image preprocessing (Section 2.3) ideally reduces background noise to zero.

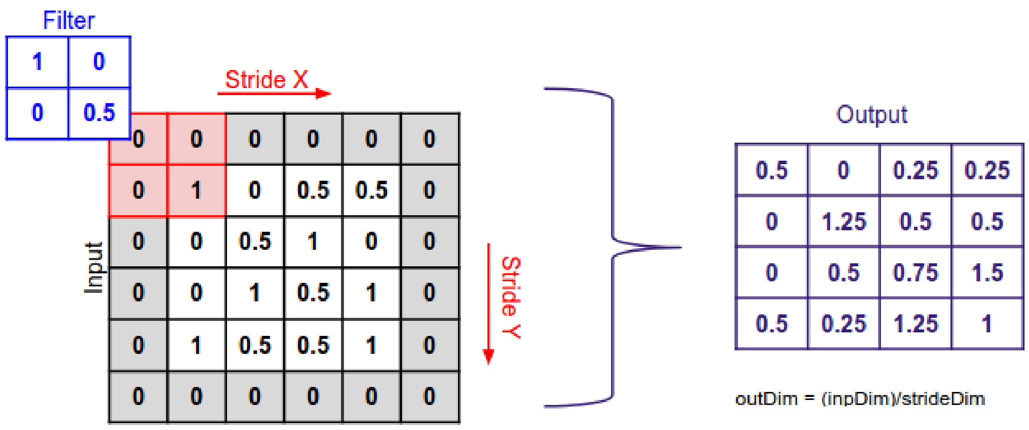

Figure 2.7: Demonstration of a 2 × 2 kernel filtering operation output as a product of a [1,2,2,1] stride and zero or 'same' padding (Credit: Allan Handan[1], cropped for conciseness).

In addition to kernel size, my approach takes into account the number of convolutions performed in each layer. Similar to the number of convolutional layers and kernel size, the filter counts dictate how many dimensions and representations are analysed. If too many filters are used, the network training may attempt to fit too many parameters each batch and cause instability. Too few filters may result in under-fitting and longer training time to map image features.

The preliminary network I developed used 64 filters for the deepest convolutional layer and one filter per hidden layer surrounding the deep layer. I chose these experimentally to sufficiently model the feature space while being able to accept a single channel image and produce a single channel latent vector. While testing this network, I trialled an iteratively increasing number of layers following the base two systems. A filter count that produced an error minimum was used as a median for successive tests to find the next lowest minimum.

In my approach, I tested a number of convolutional parameters to determine the best performing combination of filter size and convolution operations. I performed these tests using intuitive increments of convolutional kernel sizes of 2 × 2, 3 × 3, 5 × 5 and 7 × 7 from the smallest image features to the largest. As discussed in this Section, the preliminary stride of $[1, 2, 2, 1]$ is kept constant throughout testing. I trialled the number of convolutional filters per layer only for the deepest layer, as the layer conducting most of the feature analysis. I test the number of these convolutions

[1]https://labs.bawi.io/

incrementally based on a logarithmic scale.

2.4.4.2 Pooling Layers

As detailed in Section 1.2.1.4, the pooling operation is similar to a convolution, where a receptive field will perform an operation such as locating a maximum, and return a spatially ordered output reflecting these values, as shown in Figure 2.8. The autoencoder used in this thesis features a max-pooling layer in the encoder and de-pooling in the decoder for a degree of translation invariance. The de-pooling layer in the decoder is a linear interpolation to resize the latent vector to the image dimensions before the encoder max-pooling. The position of these layers is shown in Figure 2.6.

More care with kernel sizes and stride must be taken with pooling, as it is analogous to blurring. An excessively large kernel size may average too many features much smaller than the filter size. Given many RGZ images, features are comprised of 3 or more pixels (derived by visual inspection), the filter size and stride are initially set to the smallest reasonable size. Although scale invariance is ideal, resolution loss from pooling will become problematic if components and peaks are blurred together. Similar to the parameter selection used for convolutional layers, a max-pooling filter size of 2×2 with a stride of 1×1 was chosen for these reasons.

The final method of this thesis refines the filter size for best performance. This testing is conducted using filter sizes 1×1, 2×2, 3×3, 5×5, 7×7 to determine the dimensions most suitable for the convolutional layer outputs of input RGZ data. The stride is not changed as explained in the convolutional filter testing of Section 2.4.4.1. The number of these tests are kept low by only testing on a network configured with the optimal number of convolutional layers and filter sizes outlined in the tests of Section 2.4.4.1.

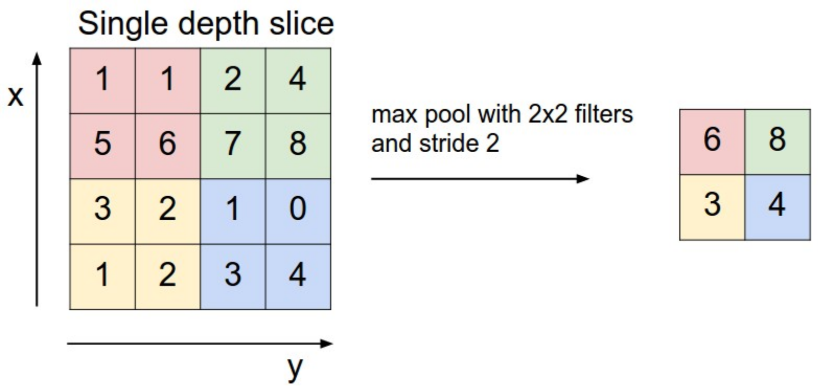

Figure 2.8: Demonstration of max-pooling layer output with a 2×2 filter, stride of 2 and no padding, where the response of the each filter element is the maximum element value of the corresponding quadrant on the single depth slice (Credit: CS231n Convolutional Neural Networks for Visual Recognition[2]).

2.4.4.3 Latent Feature Layer

Explained in Section 1.2.2.3 and 2.4.1, the latent feature vector is a hidden layer located between the encoder and decoder, at the centre of an autoencoder. In the compressive autoencoder architecture, I developed in this thesis, the dimensions of this layer are smaller than the input layer as a consequence of the preceding convolutional and max-pooling layers. The dimensionality reduction between the input p layer, and the latent vector n, are given by the reduction factor $\phi_{p,n}$:

$$\phi_{p,n} = \frac{|p|}{|n|} = k(s_{x,y})^2 \tag{2.7}$$

Where k is the number of transformation class functions as layers, from input to latent vector $\mathbb{G}^p$, to $\mathbb{F}^n$ (with notation defined in Section 2.4.1). While $s_{x,y}$ is the number of elements along the principal diagonal in all pooling and convolutional filters, given they are equal.

Given $k = 4$, with four layers in the encoder (three convolutional and a max-pooling), and $s_{x,y} = 2$, with all filters possessing a kernel size of 2×2, the reduction factor of this network is defined as:

$$\phi_{p,n} = \frac{n}{16} \tag{2.8}$$

This factor indicates a significant reduction in dimensions, with the latent vector containing 16 times fewer data, or only 6.25% of the elements from the network input. Consequently, the original dimensions of RGZ input images as 14400x1 (120x120), are compressed to 900×1 (30×30). For simple processing, I express this latent vector as this 1D array throughout the remainder of the system.

2.4.4.4 Activation Functions

Introduced in Section 1.2.1.4, the ReLU activation function (Equation 1.7) has demonstrated great success in NNs. Formalised in Equation 2.9 and demonstrated in Figure 2.9, this function is can augmented with a 'leaky' parameter to allow for a small, non-zero gradient when the neuron is saturated and not active.

$$y_i = max(w_i^T x, 0) = \begin{cases} w_i^T x & \text{if } w_i^T x > 0 \\ a w_i^T x & \text{else} \end{cases} \tag{2.9}$$

Termed LReLU, this activation function with a leakage parameter a, has been shown to further improve the convergence of the ReLU activation function (Maas et al., 2013). This improvement is found by the leaky parameter mitigating the problem of 'vanishing gradients', where neuron gradients may return to 0 if a neuron is

[2]http://cs231n.github.io/convolutional-networks/

not activated over successive batches (Pascanu et al., 2013). Given these successes, I developed the autoencoder of this investigation to include LReLU activation functions with a default leakage parameter a, of 0.2 in all layers. It is beyond the scope of this thesis to test other activation functions or leakage parameters given the number of tests already required for optimisation.

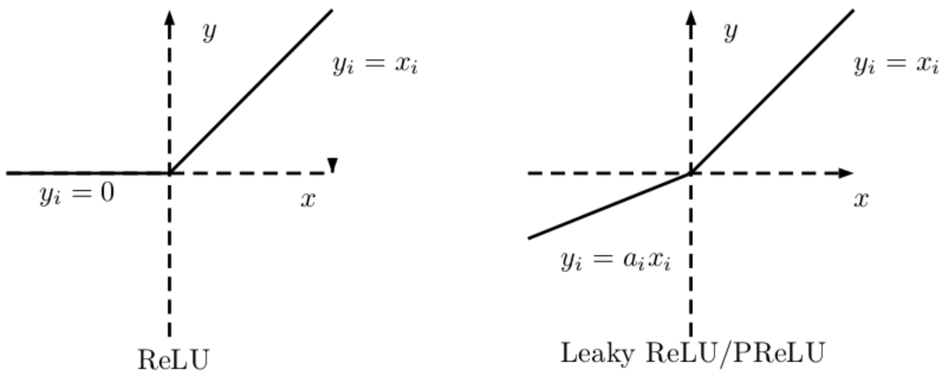

Figure 2.9: Demonstration and comparison of ReLU and LReLU, where the leaky parameter produces a small, non-zero gradient when the neuron is saturated and not active. Cropped from (Xu et al., 2015) 'Figure 1' for conciseness.

2.4.4.5 Cost Function

Autoencoder error is calculated by a distortion or difference metric between the network input and prediction output, as defined in Section 1.2.2.3 and 2.4.1. In this investigation I apply a simple and intuitive MSE metric (Equation 1.6, stated in Section 2.4.1) to quantify reconstruction fidelity. Given a low difference metric indicates a high regeneration accuracy, the difference metric indicates whether the latent vector contains sufficient features for accurate reconstruction. Consequently, this defines the accuracy of the latent vector as a lower dimensional feature space representation of the input images. Under the assumption the network can produce accurate reconstructions from this latent vector, this MSE measure can also be used to identify outlier images as reconstructions with an uncharacteristically high error.

During validation, I compliment the MSE metric with regenerated samples of the RGZ class representative images used throughout this thesis (first shown in Figure 2.2). This output features the input image, the autoencoder regeneration and a difference image to indicate region fidelity. These samples allow the operator to determine whether the autoencoder can retain the original component and peak features after compression. Retention of diffuse astronomical features are desirable; however, peak and component counts are the characteristics that define each image. This step is necessary in the case of an uncharacteristically low error that may have been reached without proper image reconstruction such as the autoencoder merely zeroing all features. To train the autoencoder, the MSE error metric is minimised to optimise the performance of the autoencoder.

2.4.4.6 Loss Optimisation In Training

Training in ML is a process of refining network weights and biases to reduce an objective function such as an error, loss or cost metric to a global minima (outlined with greater detail in Section 1.2.1.4). The autoencoder I developed in this thesis is trained to minimise a MSE cost function (outlined in Section 2.4.4.5) to convergence by optimising network parameters with the Adam Optimiser. I chose this optimiser due to its simplicity, computational efficiency and ability to handle the number of parameters within the convolutional autoencoder of this thesis. The convergence criteria I used is based on the average loss or error of the network in the last 10% of batches in each epoch. Alternatively, I could use a criterion that locates when neuron weights are no longer undergoing drastic changes. I chose the error rate convergence metric to better communicate convergence as a measure of an ideally global minima in network accuracy, which can be visually interpreted on in the reconstruction outputs.

The key parameter in this optimiser is the learning rate. Given there is no precise science in selecting this rate, a nominal rate of 0.01 was implemented. Additionally, the exponential decay rates for the 1st and 2nd moment estimates are the TensorFlow default values, 0.9 and 0.999 respectively. Due to the scope of this thesis, the learning rate and these decay rates are not optimised for improved accuracy or faster convergence. Instead, I tuned the number of samples trained simultaneously before optimisation as training batch size, given both of these tests achieve similar ends.

I optimise this batch size by testing iteratively increasing batch sizes on a logarithmic scale. This is similar to the convolutional parameter tests of Section 2.4.4.1. It should be noted that learning rate testing would also be required if the network could not be adequately trained with a learning rate of 0.01.

Autoencoder training in this thesis is halted when a set number of epochs have been training. Additionally, I developed an error convergence criteria to indicate when training has reached a minimum, before over-fitting. I implemented a criterion given as the average error in the last 10% of batches at the end of each epoch.

2.4.4.7 Rotational Affine Transformations

I investigated rotational invariance in the autoencoder and SOM by randomly rotating input images during autoencoder training in this thesis. As a typical ML problem, rotational variance prevents clustering methods from recognising rotation as a feature distinguished enough to separate it from its class. Using my method, I seek to determine whether rotational features will be encoded into the latent vector as a re-

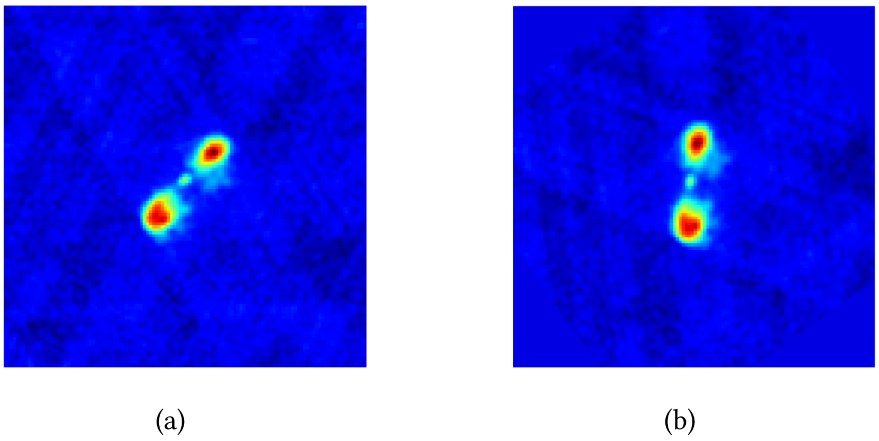

Figure 2.10: Demonstration of random rotation dataset augmentation, where the original image (a), is randomly rotated (b), with all NaN corner pixel values set to 0 as a zero padding operation.

sult of rotational dataset augmentations. Additionally, I take the size of the dataset into account, as this determines the number of random rotations each training image will receive to generalise over all potential rotations.

In my implementation, all Not a Number (NaN) pixels in the corners of input images are padded to zero, as the natural consequence of rotating a square image, as shown in 2.10 (b). Similar to the zero padding used in the convolutional layers (Section 2.4.4.1), this is the least harmful to image quality, given background noise is ideally zero. An alternate solution beyond project scope would be to inject characteristic noise into these values, as described in the sigma clipping preprocessing of Section 2.3.

I analysed the effects of rotation augmentations by comparing the autoencoder and SOM validation error and training time produced by regular training and with the augmented training images.

2.4.4.8 Noise Injection

Corrupted inputs ideally allow the autoencoder to extract more robust and useful features since higher level representations have more stability than the injected random noise (Vincent et al., 2010). In this investigation, I test autoencoder denoising with image impulse noise injection as demonstrated in Figure 2.11. I perform this test to determine whether impulse noise corruption in preprocessed RGZ images can reduce training time and improve robustness. Accuracy is also ideally improved with random noise-introducing enough variance to avoid over-fitting a training set. Training time will be marginally increased to generalise the noise and the operation of noise injection itself.

I conduct this analysis in the same manner as the rotation augmentations of Sec-

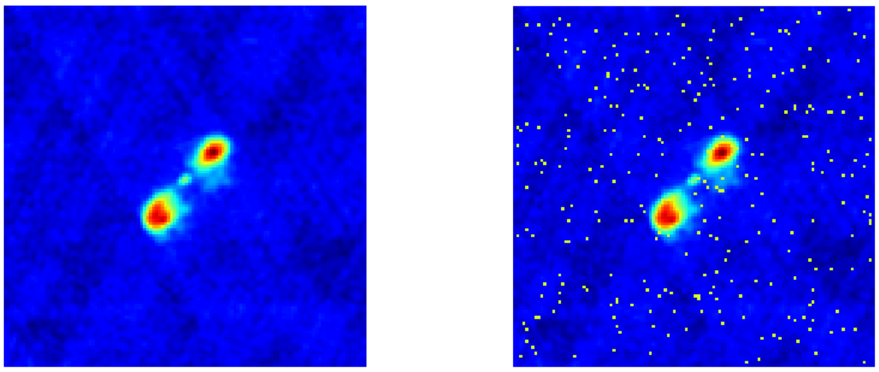

Figure 2.11: Demonstration of impulse noise injection augmentations, where the original image (a) is purposefully corrupted with random zero and full intensity pixels (b).

tion 2.4.4.7. I achieved this by comparing the autoencoder validation error and training time produced by regular training and with noise injected training images. Additionally, I combined this noise injection with the rotation augmentation to asses whether the techniques are complementary.

2.4.5 Overview of Autoencoder Testing and Tuning

In this investigation, I test for the optimal combination of training hyper-parameters and network architecture for training time and accuracy. In Ralph et al. (2018), these parameters were initially selected experimentally. As the investigation progressed past the paper, deeper testing and result refinement was required to prove the true efficacy of this method with RGZ data. As a result, my final method used the datasets specified in Section 2.4.3 to test the parameters of Sections 2.4.2 - 2.4.5 to locate this best performing combination of hyper-parameters and architectures. These tests are summarised into four steps:

- Convolution and max-pooling filter dimensions testing
- Convolution layer filter count testing
- Training batch size testing
- Training set augmentations with random rotation and noise injection testing

Several initial conditions are set for the autoencoder based on practices observed in literature:

- Convolutional filter size of 3×3 with a stride of [1,2,2,1]
- Max-pooling filter size of 2×2 with a stride of [1,2,2,1]

- Initial batch size of 128
- All tests conducted without training set augmentations

Using these initial conditions, I conducted the tests summarised above, by evaluating for autoencoder regeneration accuracy and training time. Results are displayed as the training and validation reconstruction error of the network per batch. This is accompanied by the time taken for these operations and sample reconstructions (Section 2.4.4.5). The tests I conducted to locate these optimal combinations are summarised as follows:

1. Determine the ideal convolutional and pooling kernel dimensions by testing square 2×2, 3×3, 5×5, 7×7 convolutional filter dimensions, with square, 1×1, 2×2, 3×3 and 4×4 max-pooling kernels.
2. Testing for an optimal number of convolutional filters for the convolutional layer with sizes increasing logarithmically. This testing is conducted on the network with the optimal combination of convolutional and max-pooling kernel sizes.
3. Determining an optimal training batch size using a network derived from the results of the previous two tests. This testing is also done on a logarithmic scale.
4. Analyse the effects on accuracy and training time of random rotations and noise injection dataset augmentations. As in previous tests, I conduct this analysis on the network configuration found as the results of the previous three tests.

In my system, I used the latent vectors calculated by encoding RGZ images produced by the refined autoencoders to train a SOM for clustering analysis with reduced processing time.

2.5 Hierarchically Clustered Self-Organising Map

In this Section, I outline how I use my approach to investigate the ability of an autoencoder latent vector trained SOM as an efficient unsupervised ML alternative to solve the problems of big-data scale and complexity in modern radio astronomical data.

Detailed in Section 1.2.2.2, SOMs are data analysis methods used in unsupervised clustering and data exploration. SOMs create similarity maps or learning manifolds of input data where distinct groups of neurons reflect latent clusters in the data. As discussed in Section 1.5, I aimed to investigate whether a SOM trained on the reduced space of autoencoder latent feature vectors can adequately visualise the dynamic distribution and high-level topological relationships of radio-astronomical images from the RGZ. Using this system I perform unsupervised anomaly detection, complexity separation and HC of RGZ images on the trained SOM umat to achieve my aims and answer the research questions of Section 1.5.

The SOM I developed in this thesis was implemented using the Somoclu package. I trained and validated this system using the latent vector representations of the RGZ dataset divisions outlined in the following sections. Using these input datasets, I trained the SOM with the procedure outlined in Section 2.5.1. The outputs of this training are discussed in Section 2.5.3 and 2.5.3. Following this, the HC clustering of SOM weights and their use in evaluating SOM modelling and overall system performance is outlined. Similar to the hyper-parameter and architecture optimisation of Section 2.4.4, I conduct several tests to locate an optimal combination of initial conditions to maximise SOM modelling performance.

2.5.1 Self-Organising Map Training

Introduced in Section 1.2.2.2, a SOM models datasets by iteratively updating a grid of neuron weight vectors m_t. This is achieved by moving toward similar data points $x(t)$ on the SOM manifold by refining neurons weights with a neighbourhood distance function h_{ci} of each neuron i, by a decaying learning rate α which is balanced to let all neurons stabilise in optimal time. A well-trained SOM after m epochs will visualise the distribution of the input RGZ training data as various high-level topological relationships and morphology distributions.

$$m_i(t+1) = m_i(t) + \alpha(t) \cdot h_{ci}(t)[x(t) - m_i(t)] \tag{2.10}$$

Using this concept, I implemented a SOM by training with the following procedure as a variation of the training summary summarised by (Geach, 2012):

1. Initialise SOM grid neurons with a PCA learning manifold (outlined in Section 1.2.2.1) of the latent feature vector set. This approach allows the SOM to model an already delineated PCA space.

2. Select a random latent feature vector from the training set.

3. Locate Best Matching Unit (BMU) neurons as the 'closest' neuron to the selected data point. A common and reliable distance metric used to calculate this is Euclidean distance (Equation 1.2).

4. Move all BMU neurons within the neighbourhood learning radius R, toward the data point by updating neuron weights $m_i(t)$ as a function of the distance metric

$h_{ci}(t)$ and learning rate $\alpha(t)$, as shown in Equation 2.10. This neighbourhood learning function or 'learning radius' can be represented with a several shapes by their radius, namely linear ('bubble') or Gaussian, defined by decay functions σ:

$$h_{ci} = h_{c0}\sigma \tag{2.11}$$

Where exponential decay is given by:

$$\sigma_{exp} = e^{-\frac{t}{\tau}} \tag{2.12}$$

And linear decay:

$$\sigma_{linear} = -\frac{t}{\tau} \tag{2.13}$$

Where τ is a decay constant, usually given as the number of epochs.

5. Update learning rate and radius based on respective input decay rates. Similar to learning rate decay, this neighbourhood decay function can be expressed with an exponentially or linearly decaying σ:

$$\alpha(t) = \alpha(0)\sigma \tag{2.14}$$

6. Iterate until a stop condition is met or learning and neighbourhood rates have decayed to a limit or zero. In this thesis, each iteration is considered an epoch as the entire dataset is taken into account with no mini-batch training.

Many options and parameters are available for SOM testing. The SOM grid can be projected to numerous spaces, namely rectangular and toroidal and neurons can be expressed as square or hexagonal. For simplicity in plotting and analysis, I used a square neuron shape at the cost of the additional two degrees of freedom offered by a hexagonal neuron. Furthermore, I implemented a toroidal map-projected to a 2D rectangular space as shown in Figure 2.12, due to edging effects observed when projecting to a rectangular SOM grid (Andreu et al., 1997). These edging effects could likely be avoided with a spherical space. However, this space is not as easily interpreted, as unlike the toroidal geometry, it cannot be as easily transformed to a square space without a degree of distortion.

Concerning learning rates, I used the nominal initial rate of 0.01 and converged to 0.001. However, some tests may manipulate the neighbourhood and learning rate

[3]http://pi.math.cornell.edu/ mec/Winter2009/Victor/part1.htm

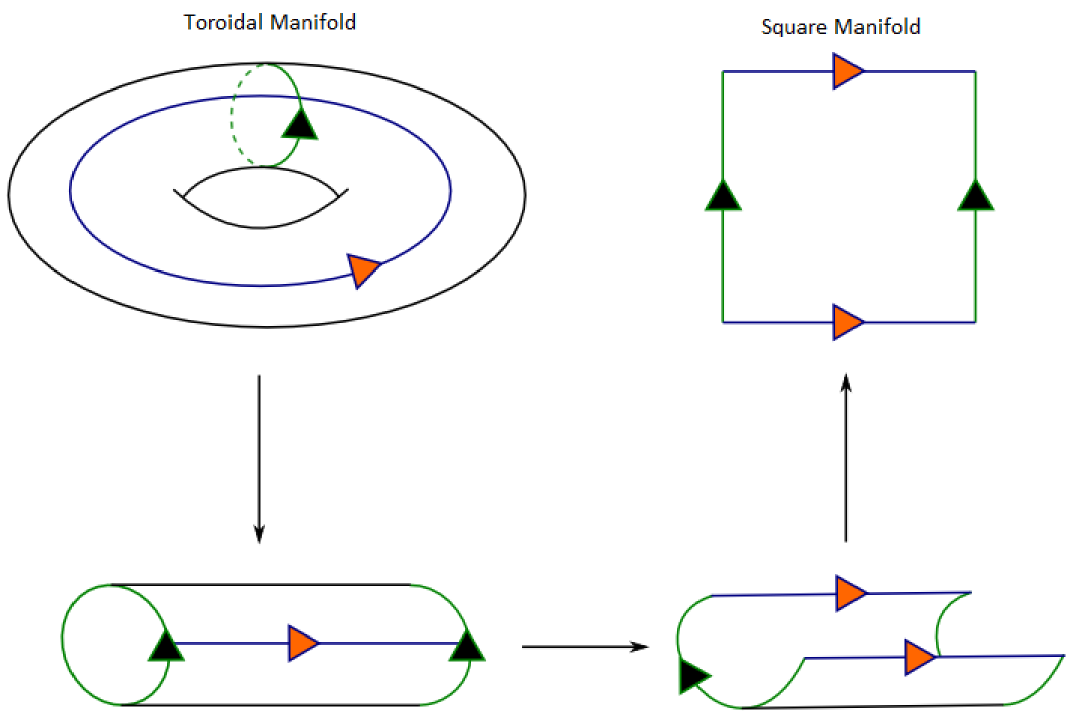

Figure 2.12: Demonstration of the transformation between a square and toroidal manifold. In this thesis, I train the SOM in a toroidal space with outputs displayed on a square space for readability. Adapted and modified for clarity from Cornell 'Flat Life' Winter 2009[3].

using the decay constant τ, to finish iterating on zero scale which may prevent some late-stage weight refinement.

All tests initialise the SOM space using a PCA projection of the training latent vector set rather than random projection. The PCA initialisation provides the SOM with a reasonable initial state and helps to maintain similar cluster locations and distributions across multiple tests for robust comparison (regarding the latent vector training set is the same).

The number of training epochs throughout these tests is kept constant at ten epochs. Ideally, tests will be stopped when neurons have converged, and the map is stable. This stability point is difficult to locate, and there are few conclusive metrics used to describe convergence. The absence of a definitive metric to describe a convergent SOM weight steady state has to lead to some authors such as (Geach, 2012) iterate for an intuitive amount, in this case, ten epochs, with the knowledge that each element of the dataset will be visited ten times during training. This thesis aimed to use ten training epochs for this reason and to reduce the number of potential optimisation tests. However, a subtle internal error in Somoclu prevents single epoch iterations, as its C back-end can only perform two epochs at a time. As I analyse the SOM outputs on a per epoch basis in addition to the final epoch, I settled on examining every second epoch, resulting in either a total of 8 or 12 epochs due to this error. I chose 12 total epochs for training in each test, given performance is vital to the aims of this thesis.

Although this thesis follows this typical training procedure step, SOM training is implemented using Somoclu which follows a common technique in In modern parallel computing where SOMs weights can be updated using a batch formulation toward final epoch f, where I use a full dataset batch size, (Wittek et al., 2013):

$$w_j(t_f) = \frac{\sum_{t'=t_0}^{t_f} h_{ci}(t')x(t')}{\sum_{t'=t_0}^{t_f} h_{ci}(t')} \tag{2.15}$$

2.5.2 Unified Distance Matrix and Activation Map Output

In this investigation, I developed a SOM system to produce several outputs (using the packages outlined in Section 2.1). The most useful output to this investigation is the umat. The umat is visualised as a heat map of the Euclidean distance between each neuron and its immediate neighbouring neurons.

Umat distance is given as $U(j)$, for neuron j with immediate neighbourhood $N(j)$

$$U(j) = \frac{1}{|N(j)|} \sum_{i \in N(j)} h_{ci}(m_i, mj) \tag{2.16}$$

In this thesis, I implemented a visualisation package to show the umat superimposed with the first closest matching RGZ image candidate on each neuron. This first matching candidate is the image with the 'closest' latent vector in the dataset to the neuron weight (where the distance is based on the Euclidean distance metric used in the SOM, in Section 2.5.1). I use this map to show the morphology distribution of RGZ images where the Euclidean distance between the learned weight of each neuron and their neighbouring neurons displayed as a heat map. As shown in Figure 2.13, larger SOM sizes produce a umat with a comparably more refined structure, where discrete clusters are formed with high distance regions acting as decision surfaces between learned clusters.

Additionally, the activation map can be displayed when examining sources case by case, to highlight a probability distribution of where the candidate source should reside on the map. In this thesis, the activation map is not used to interpret the full test set.

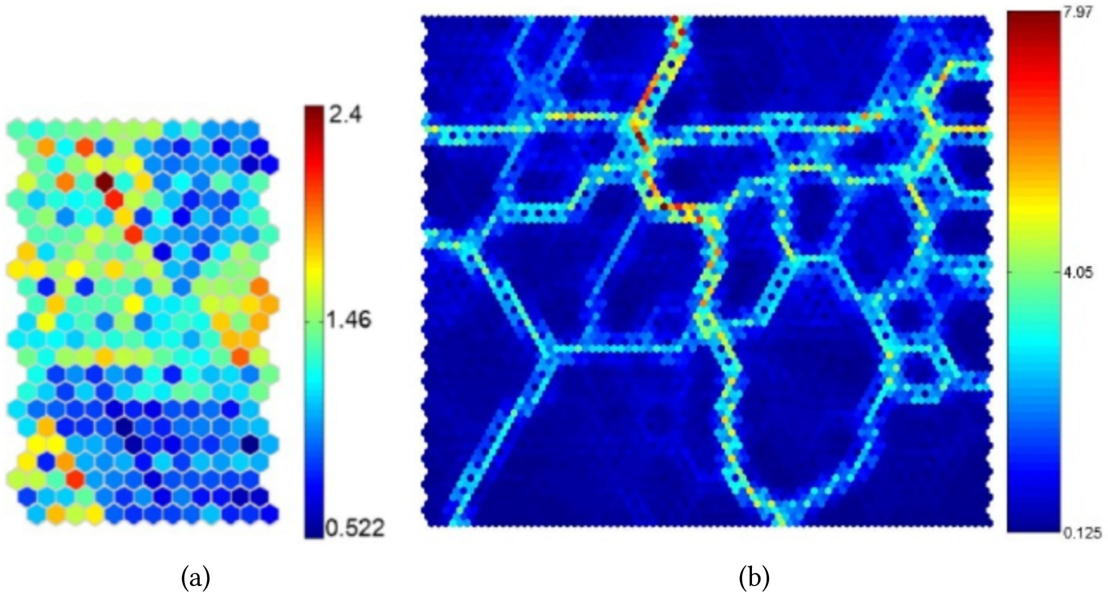

Figure 2.13: Comparison of the umat output between a small SOM (a) (Wang et al., 2014) and large SOM (b) (Credit: Miguel Barreto[4]). The larger SOM produces a more refined structure with discrete clusters segmented by high distance regions.

2.5.3 Self-Organising Map Weight Image Reconstruction

As outlined in Section 2.5.2, the umat SOM output displays neurons with the closest matching image from the dataset instead of the actual weights. While this gives the operator with an idea as to how the SOM clusters real dataset images, it provides an incomplete insight into what individual SOM weights have modelled. In the case of other SOMs trained on image morphology, such as (Polsterer et al., 2015), neurons are displayed with their weights as images which explicitly shows what each neuron has modelled. This common practice is trivial when trained on images, however, in the case of this investigation, I trained a SOM on a latent vector, which cannot be viewed as an image.

The later stages of this investigation, I enhanced the system with a novel feature to display the learned weights of each neuron as a close approximation by reconstructing all neuron weight vectors with the already trained convolutional autoencoder (outlined in Section 2.4.1). Given the decoder side of the network is trained to restore images from a latent vector, adequately trained SOM neuron weights would ideally be restored to the image it represents. Using this approach, all learned neuron weights are parsed to the autoencoder for decoding. The decoded weights as images are then superimposed over the original neurons. This allowed the viewer to understand better how the SOM has been trained. These weights are also superimposed over the

[4]https://www.slideshare.net/askroll/classification

umat to create a new map similar to the function of the umat first closest matching candidate superimposed map (Section 2.5.2)

2.5.4 Complexity Separation and Anomaly Detection

Investigation and development of anomaly detection and complexity separation framework using the latent vector trained SOM is key to my aims and research questions in this thesis (Sections 1.4 and 1.5 respectively). I achieved this end by investigating the correlation between the positions of complex and anomalous sources on high Euclidean distances in the trained SOM umat. In this thesis, I consider point sources with a RGZ label 11 as simple, while all other sources are complex. Additionally, I classify any source neither 11 or 12 labelled (single component radio source with a secondary peak) as anomalous.

As outlined in Section 2.5.2, the umat illustrates the Euclidean distance of neuron weights and their immediate neighbours. Given complex and anomalous sources are highly dissimilar to each other, and simple RGZ labelled 11 point sources, a properly trained map will place complex sources in a high distance region, as suggested with the umat distance distribution produced in (Polsterer et al., 2015). Using this concept, I implemented a complex and anomaly source flag that segments the matching candidates of each neuron based on the umat Euclidean distance of the neuron weight. I establish these flags with a decision surface at standard deviations 2-5 of the neuron Euclidean distance distribution across the umat.

The metrics I used to analyse the performance of this separation and detection feature are detailed further in Section 2.5.6.

2.5.5 Hierarchical Clustering of Self-Organising Maps

The final output of the SOM is the HC SOM. This map is divided into associated groups using the K-means algorithm (Detailed in Section 1.2.2.4). K-means segments neurons on the SOM into a ‘K’ number of clusters based on the learned weight of each neuron and a distance metric, such as Euclidean distance. This is an iterative process where the distance between each cluster is calculated as the average distance of its consistent objects. Input clusters are continually refined based on this distance until the changes in each cluster reach a stop condition. These clusters are discrete, where an object is assigned to only one cluster.

In my system, I use this clustering method to segment the learned SOM space and the matching RGZ images by their high-level topology and morphological relationships. Given this is an unsupervised method, these clusters will not be identical to the labels of the RGZ dataset. Instead, the clustering will be based on the relationships and features the system has learned from the within the training data. I analyse this clustering by the correlations formed between these groups and the relationships between the RGZ labels. I expect my system to classify based on deep morphological relationships, such as the extended nature of a source and the distance between central components and companion sources. Concerning evaluation, I am limited by the simplicity of the RGZ labels given they only describe very simple features peaks and components counts in each image. As a result, I can only assess the clustering by the correlations between the RGZ labels and how well the clustering resembles RGZ relationships.

The output of the K-means clustering is an arbitrary K cluster number. This thesis uses this number to produce a labelled umat and coloured visualisation of the clusters over the SOM manifold. Given K-means clustering is unsupervised, these clusters are arbitrarily identified and coloured. These colours and K-means identification numbers will change each time clusters are calculated. The K-means coloured clusters are blended with the umat first closest matching candidate superimposed map (Section 2.5.2) and the umat-decoded weight superimposed map (Section 2.5.3) to demonstrate how the clusters segment images by Euclidean distance and morphology.

Clustering output is also shown by two colour coded umats blended with the first candidate match and the reconstructed neuron weights. It is advised that these maps are viewed electronically, as displayed in print is difficult due to how many features need to be fitted to such a large grid. Finally, these clusters are detailed with several tables and figures detailing the RGZ label population of each cluster.

2.5.6 Evaluation of the Self-Organising Map and Overall System

I quantify the overall performance of the system by several metrics based on the SOM outputs and comparisons to similar systems in the literature. In all SOM configuration tests, I assess modelling quality by the performance of the SOM in anomaly detection and complexity separation. For HC testing, I analyse clustering quality by the correlation between the labels of neuron matching candidate images with discrete entropy and the purity of sources contained within each cluster.

2.5.6.1 Evaluation using Complexity Separation

I evaluate the anomaly detection using the following measures, as outlined by (Flach and Kull, 2015):

- Accuracy, used as a basic metric to describe the ratio of correct predicted values over the size of the dataset, as:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{2.17}$$

Where TP is the number of true positive cases, TN false negatives, FP false positives and FN as false negatives.

- Precision, as the proportion of true positives among the positive predictions. This metric describes the proportion of anomalous sources that are classified correctly as true positives with as few as possible false positives. I use this measure to provide insight into how many anomalies within the classification that have been correctly classified. This metric is given by:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{2.18}$$

- Recall, as the true positive rate, measures how well my detection method locates true positive anomalies within minimal false negatives. Also known as sensitivity, I use this measure to understand how many of the overall anomalies have been detected. This measure is highly important as it describes how many anomalies that may have been overlooked. Recall is given by:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{2.19}$$

- F1 score is my final definitive means of quantifying performance. This score is the harmonic average of the precision and recall as a robust combination of the two metrics, given by:

$$\text{F1} = \frac{2\text{TP}}{2\text{TP} + \text{FP} + \text{FN}} \tag{2.20}$$

A value of 1 in all these measures describes the best case, while 0 describes the worst.

2.5.6.2 Dataset Label Correlation Using Discrete Entropy

I determine the performance of the HC based on the purity and completeness of source classifications in each cluster and by discrete entropy. This purity and completeness refer to how well each cluster represents RGZ labels. A cluster containing a high level of 11 simple point sources with few other sources is considered pure. If this cluster contains a majority of the overall population of 11 sources in the dataset, it is then complete. The actual discrete definition of purity and completeness cannot be used appropriately in the case of my system as the SOM is not clustering by the RGZ labels that would be used to calculate these metrics. Given that unsupervised methods will determine their own relationships within the dataset, basic observations on the correlation between SOM detected morphological relationships is a fair approach. This is seen in literature with other radio-astronomical SOM morphology analysis papers such as (Polsterer et al., 2015) where clustering quality is inherently qualitative. Quantitative measures in this thesis are found with the anomaly detection of Section 2.5.6.1.

I use the discrete entropy measure as a tensor for the homogeneity of labels matching each neuron. A low entropy describes a neuron with little variation in matching source labels, while a high entropy describes a neuron that contains a wide variety of source labels. This is an appropriate measure given the SOM as an unsupervised method is not learning the labels of the dataset, but deep relationships within the data itself. Consequently, I use this metric for evaluating the correlation between classifications of the RGZ dataset and those found by the SOM and HC. This entropy measure is given as $\hat{E}$:

$$\hat{P}_i = \frac{n_i}{N} \tag{2.21}$$

$$\hat{E} = -\sum_i \hat{P}_i log_2 \hat{P}_i \tag{2.22}$$

Where n_i is the number of class occurrences i and N the total number of occurring classes. A low entropy indicates good consensus where most neuron image matches have the same label. Conversely, a high entropy indicates the matching images of a neuron has a wide range of different labels. I normalise this value into discrete entropy, with a range from 0 to 1.

2.5.7 Training and Validation Dataset Divisions

In this thesis, SOM training and validation use the encoded images of the RGZ DR1 outlined in Section 2.5.7. All tests are conducted using the RGZ DR1 with no consensus filtering. These tests demonstrate the SOM behaviour in the typical case, where the dataset is unbalanced, with a vast majority of point sources (class 11). Additionally, no consensus filtering is used, RGZ labels may be inaccurate. The training and validation sets each contain 30,000 labelled latent vectors. Similar to the autoencoder, the SOM is trained in an unsupervised manner. As a result, only the validation is affected by RGZ image consensus.

2.5.8 Hyper-Parameter Optimisation and Testing

The training hyper-parameters and network architecture of the SOM are refined through a series of tests to locate the optimal combination for best accuracy and low training time. These tests are conducted similarly to the autoencoder of Section 2.4.4, where each parameter is tested with a set of initial intuitive conditions for several different parameters. These tests are as follows:

1. Map size as grid dimensions: square 5,7,10,15,20,50. An initial grid of 10×10 square neurons with a toroidal projection is used as a basic initial condition, with 18 RGZ classes represented by a 100 neurons. It is not efficient or viable to test all combinations of these, simply testing the best possible generalised architecture for a medium-small map is most efficient, which will then be used as the initial conditions for larger map sizes.

2. Exponential and linear learning rate decay, with an initial learning rate of 0.01, decaying to 0.001. These decay rates bear more consequences on the performance of the SOM than learning rates (Chaudhary et al., 2015). As a result, I keep these initial and final learning rates constant at these nominal default SOMOCLU learning rates, but test for the optimal decay rate.

3. Gaussian and linear ('bubble') learning neighbourhoods, where the initial neighbourhood radius is half the overall map height and width (where given a square map, these quantities are congruent), so initially, most neurons have a chance to converge on the final solution. it is highly unlikely that there will be inactive neurons in corners not reached by a circular neighbourhood with a diameter equal to the width and height of the map given PCA initialisation will do an elementary grouping of the latent vectors, and a toroidal map is used.

4. Neighbourhood decay function: exponential or linear. This is closely related to the learning neighbourhood kernel and therefore should be tested with different combinations of linear and exponential with combinations of Gaussian and linear decay.

5. 'k' number of K-means clusters: 4, 8 and 16.

6. Analysis of affine invariance in the SOM trained on latent vectors. This test is done last on the optimal SOM by testing for the effects of training the SOM on latent vectors, where the autoencoder has been trained using rotation and noise injection augmentations.

Tuning training Hyper-parameters and architecture is not an exact science. Too many variables and too few constants or initial conditions can lead to testing a recursively increasing number of tests. Such a sheer number of trials is beyond the scope of this thesis and would likely provide only minor accuracy and time improvements that may not be generalisable to other data, images or otherwise. As a result, a number of initial conditions were selected based on educated intuition and benchmarked performance found in the literature. These formed the early methodology and results as featured in Ralph et al. (2018).

Training on larger and smaller maps to determine the ideal training hyper-parameters network configuration by analysing the effects of learning rate decay and neighbourhood types and decay in a large manifold is unnecessary until the final best map size is chosen, given I scale the parameters with the map size and I am training on a toroidal SOM space, meaning only a few cases will see neurons not adequately influenced by the training. If I was to train for significantly larger maps (such as an *emergent SOM*, where there are more neurons than data points), then this may become more necessary. As a result, I use only the most ideal map size in the first test for all subsequent tests, except where larger map sizes may be required to show more detailed morphologies or to deeper investigate the rotation invariance tests.

2.6 System Overview

In this Section, I outline the novel system I developed to extract features of RGZ images with a convolutional autoencoder to a compact feature vector for efficient clustering, visualisation and anomaly detection using a SOM. I developed this system using the Python Language on CPU only with an Intel(R) Xeon(R) CPU E5-2650 v4 at 2.20GHz. As outlined in Section 2.1 the Google TensorFlow Library was used to create the autoencoder, and Somoclu was used to implement the SOM.

The system is comprised of three distinct components; preprocessing, convolutional autoencoder and SOM, as shown in Figure 2.14.

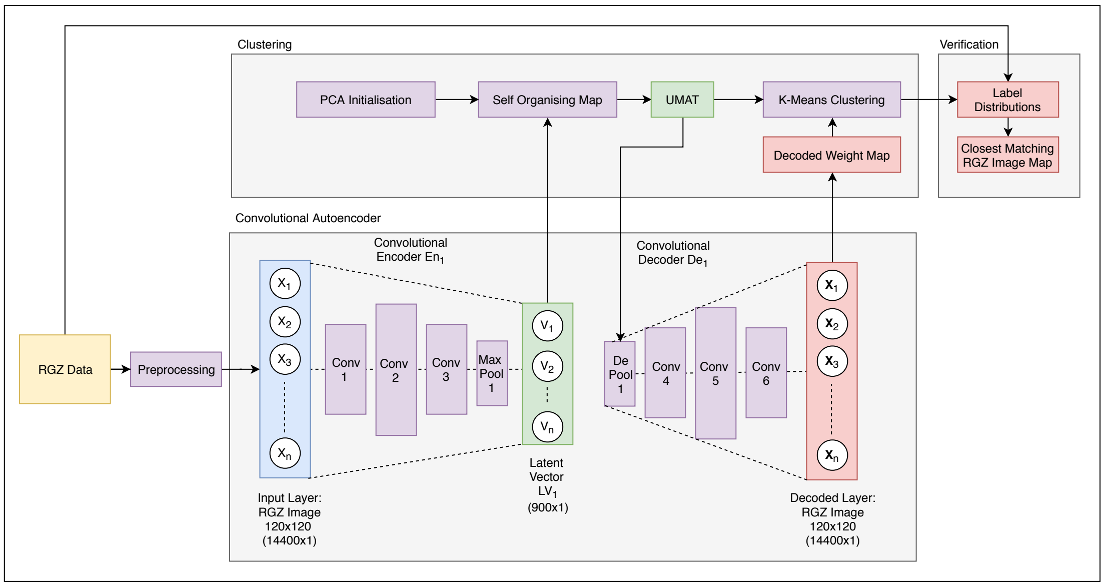

Figure 2.14: Overall system configuration with the preprocessing component, convolutional autoencoder, SOM and learned outputs with weight decoding, and HC with RGZ label validation.

In my system, all RGZ training and validation images are preprocessed for better accuracy in ML training by filtering noise while preserving critical astronomical features such as morphology and peak-component counts. In this system, I trail two preprocessing methods, as outlined in Section 2.3.

The second component of my system is the convolutional autoencoder. This network uses the preprocessed images for training and validation to analyse the data scale and complexity reduction of radio-astronomical images when reduced to a compressed latent feature vector. As shown in Figure 2.14, this network compresses the 120x120 input images with 14400 elements to the latent feature vector with 900 elements for clustering. Encoder and decoder architecture is identical with three convolutional layers. In Section 2.4.5 I outlined the training hyper-parameters and configurations of this autoencoder and how I optimised it for accuracy and training time on the RGZ DR1.

The SOM is the final component of the system. This is trained using the compact latent feature vector representations of the RGZ images produced by the optimised autoencoder. Using this component of the system, I determined whether training on autoencoder latent vectors can adequately visualise the dynamic distribution and high-level topological relationships of radio-astronomical images from the RGZ. The learned neuron weights on the SOM are parsed back into the decoder side of the network for reconstruction into their approximate image representation. Finally, I use the umat output of the latent vector trained SOM for anomaly detection, complexity separation and in HC of the RGZ dataset morphologies.

I evaluate the overall effectiveness of the system by several accuracy metrics describing the performance of the anomaly detection and the relationships between the umat and HC SOM outputs RGZ labels, as detailed in 2.5.6. I selected the final SOM network configuration and training hyper-parameters similar to the autoencoder optimisation of Section 2.4.4, as a result of several tests to maximise SOM modelling performance.

The following Results Chapter 3 details the optimisation and final results of my system. These implications and outcome of this thesis based on these results are outlined in the Discussion Chapter 4.

Chapter 3

Results

In this thesis, I developed a novel unsupervised ML system for the clustering and data exploration of radio-astronomical images in the RGZ dataset. In this section, I tested the combination of a convolutional autoencoder, SOM and HC as a practical, efficient and unsupervised solution to the problems of scale and complexity in the modern era of big-data astronomy. Outlined in Chapter 2, I conducted several tests to optimise this system and determine its success. This results Chapter summarises the outcome of these experiments. My paper Ralph et al. (2018), presented the preliminary results of this thesis, which were further expanded using the tests of my final methodology. The results of each part of the overall system are presented hierarchically from the preprocessing, autoencoder and SOM, and is evaluated using the performance measures of Sections 2.4.4.5 and 2.5.6.

3.1 Preprocessing Results

Two preprocessing methods were trialled in this investigation. I tested each method on the same image set and evaluated visually by their ability to filter out noise while preserving distinguishing astronomical features. The time taken to pre-process these images is also recorded; however, the quality of the preprocessing takes precedence in analysing performance as it has a more significant impact on ML training.

3.1.1 Adaptive Masking with Otsu Thresholding

The adaptive masking method using .PNG images, detailed in Section 2.3.1 performed well to filter out background noise and preserve most of the desired astronomical features at the cost of sacrificing the dynamic range of pixel intensity available in .FITS radio images, as shown in Figure 3.1.

It was found that this method produced several undesirable effects:

1. Tendency to filter out critical distinguishing astronomical features such as faint sources, extended emission and companion sources. This erroneous filtering is particularly noticeable in the filtering results of image (l) compared to the original image (i) in Figure 3.1

2. Rejection of too many images as 'noisy', with some cases seeing up to 25% of the dataset rejected as too noisy. This case is demonstrated with Figure 3.1 (c), wherein this case, the image is rejected, and instrument noise is amplified.

3. In rare cases, there is the production of erroneous components and features in cases where the adaptive threshold mask creates oddly shaped 'island' image regions out of noise, or by breaking single components into multiple components based on their peaks.

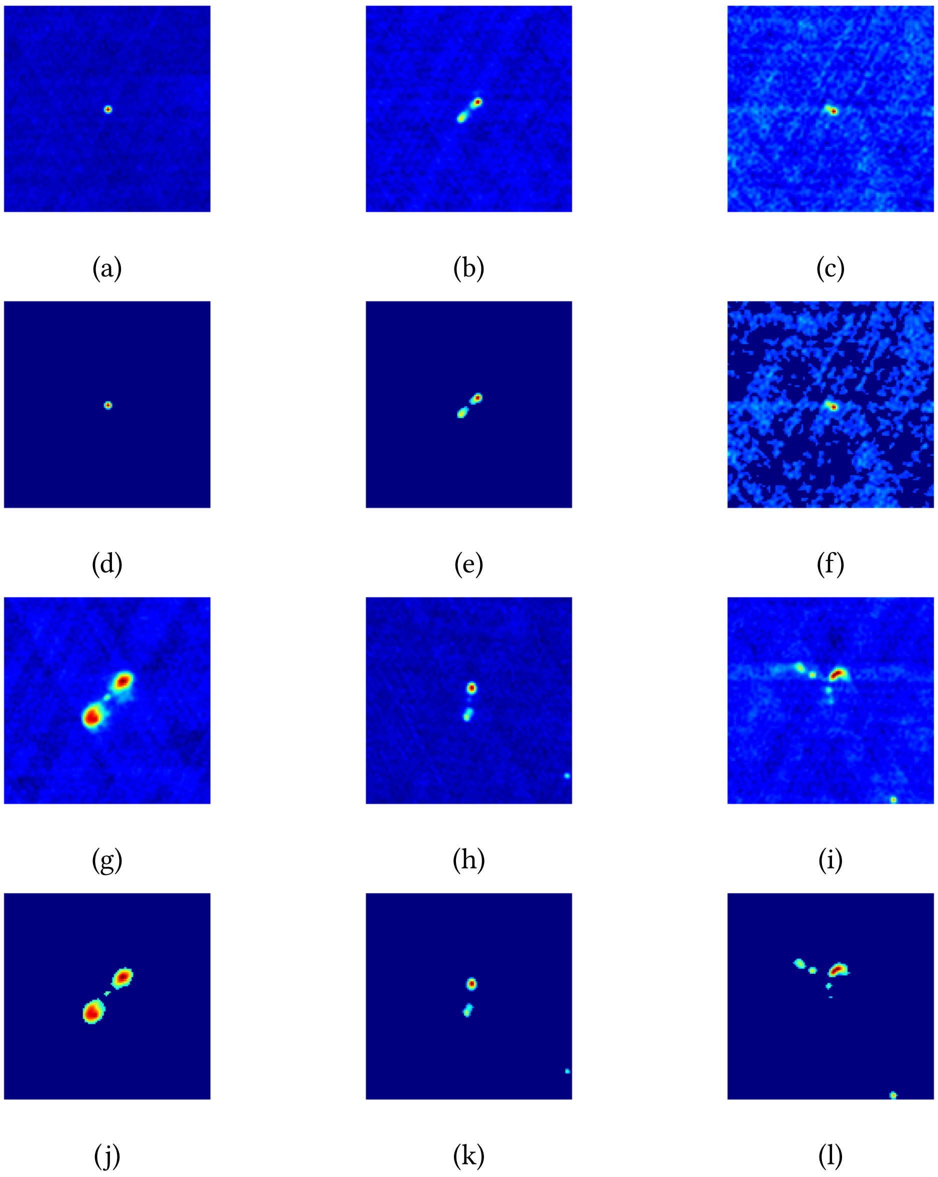

Figure 3.1: Otsu adaptive filtering results of the six representative samples of the RGZ dataset

3.1.2 Sigma Clipping and Noise Injection Correcting Preprocessing

Outlined in Section 2.3.2, the sigma clipping and noise injection correcting preprocessing method used an approach closer to typical astronomical image processing. This approach takes advantage of the dynamic range of radio-astronomical FITS images by segmenting by pixel intensity. Shown in Figure 3.2, the sigma clipping and noise injection correcting preprocessing method produced excellent noise filtering and equalisation while retaining critical distinguishing astronomical features.

Comparisons of the two preprocessing methods in Figure 3.3, confirm that this approach is less prone to producing artefacts or masking out faint sources and extended emission than the adaptive thresholding method.

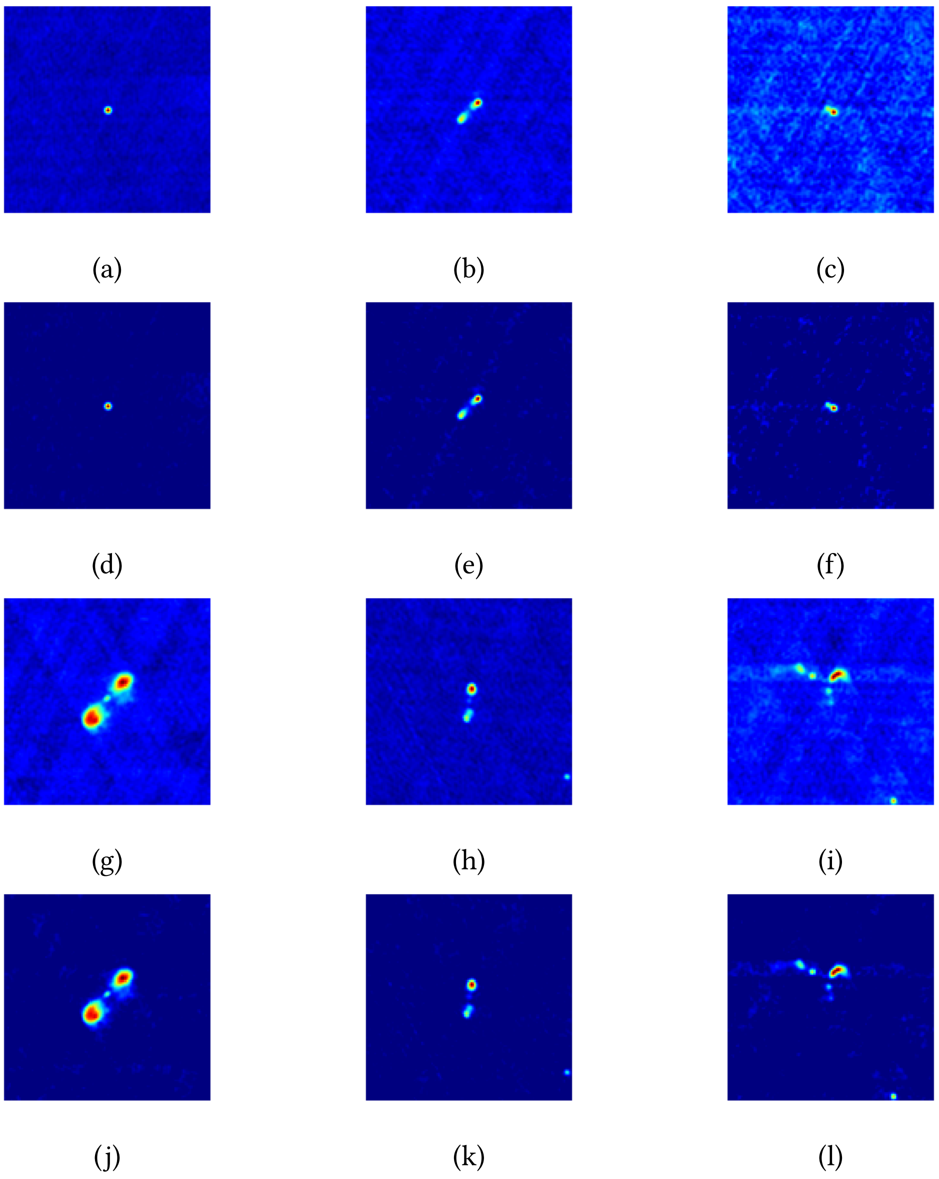

Figure 3.2: Sigma clipping and noise injection filtering of the six representative samples of the RGZ dataset

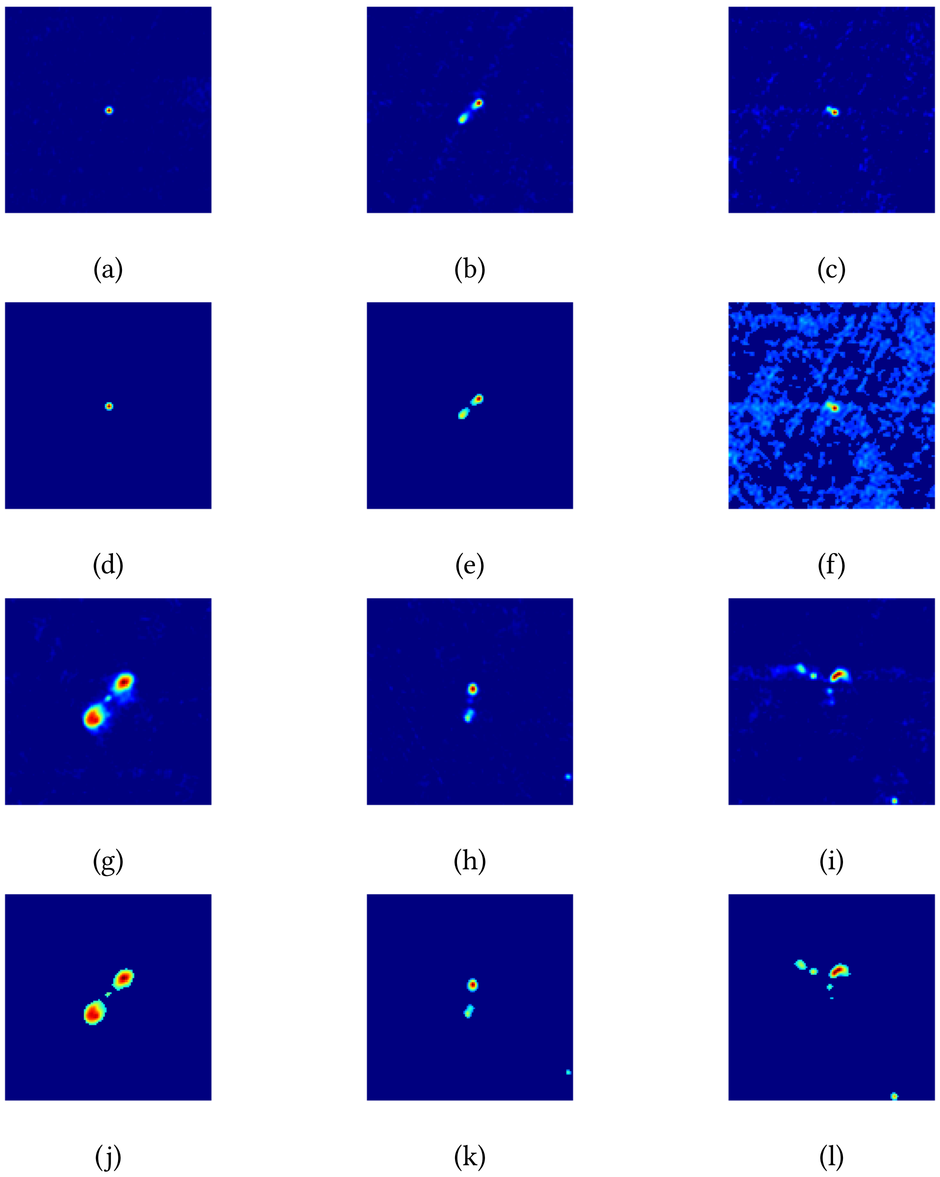

Figure 3.3: Preprocessing method comparison between the Otsu adaptive filtering and the sigma clipping on six representative samples of the RGZ dataset

Consequently, I implemented the robust cleaning produced by the sigma clipping method as the preprocessing method throughout my paper Ralph et al. (2018) and thesis.

3.2 Autoencoder Results

3.2.1 Preliminary Autoencoder Image Reconstructions

This Section demonstrates the results of the autoencoder I implemented using the method outlined in Section 2.4. These were featured as the results of Ralph et al. (2018). As discussed in Section 2.4.4.5, I evaluate these by the MSE and fidelity of the image reconstruction produced by the autoencoder. The format of these results is not altered from their original appearance in the paper to highlight the differences in final results with the refinement outlined in Section 2.4.4. This is unlike the preprocessing results of Section 3.1, which have been changed.

The autoencoder I implemented in Ralph et al. (2018) was trained on RGZ images and demonstrates successful compression and decompression across the dataset using the hardware outlined in Section 2.1. This is shown in Figure 3.4, where the reconstructed image strongly approximates the input image of the network. From this Figure, I determined that the autoencoder is capable of recognising and preserving enough key image features in the 900×1 compressed latent vector to successfully predict the original image. The difference images in the Figure show the autoencoder loses most fidelity around the edges of regions and reasonably reconstructs background noise. Images 1 and 3 of this Figure demonstrate the most error, with Image 1 merging a faint component into the central peak and Image 3 imposing a faint additional peak.

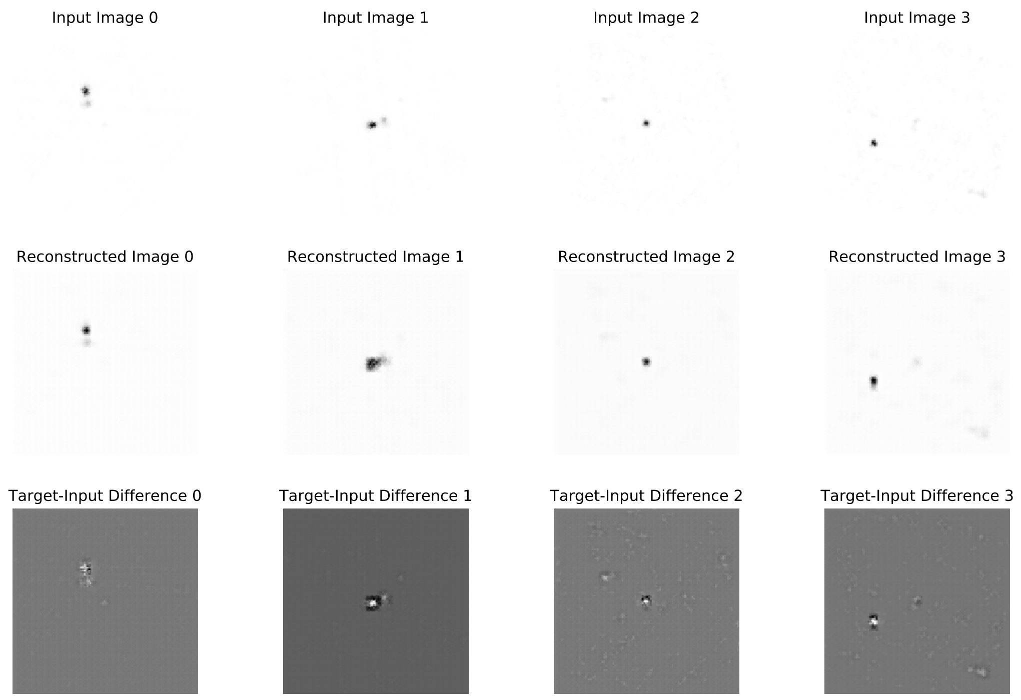

Figure 3.4: Convolutional Autoencoder prediction of Radio Galaxy Zoo input images after 20 training epochs. Top row: Original preprocessed input, Middle row: trained autoencoder prediction, Bottom row: Difference image between predicted and original image. .

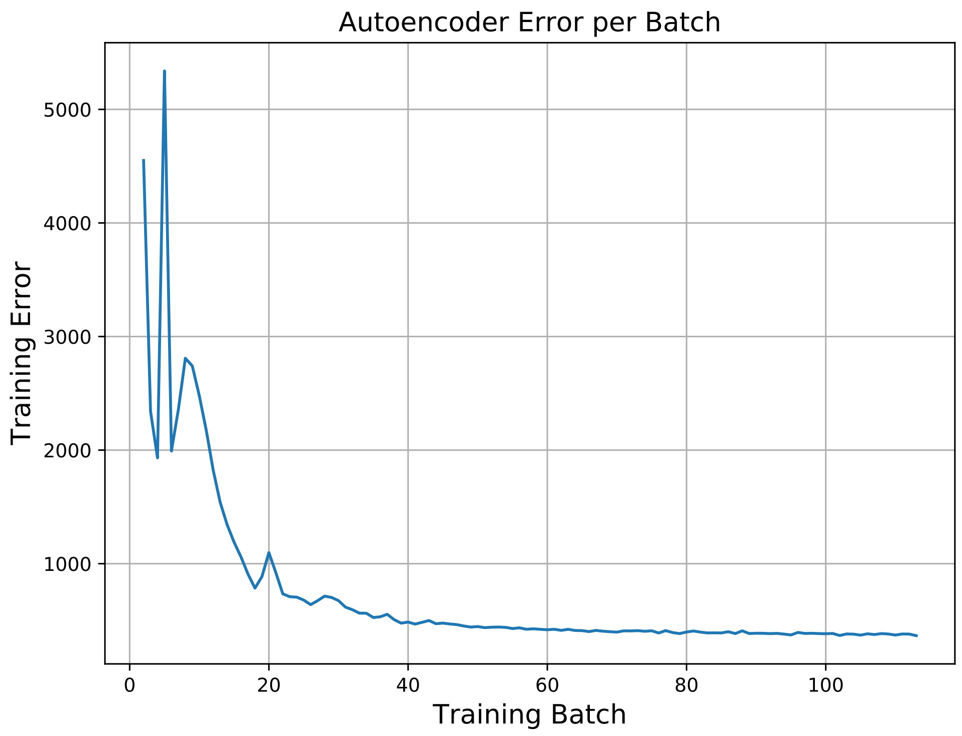

Figure 3.5: Autoencoder error per batch as mean squared difference between input target image and reconstructed image. Training error converges after 70 epochs, where the total training time for a full epoch is 2.2 minutes.

The average training time of this autoencoder is 1.2 seconds per batch of 128 images. The training time for a full epoch is 2.2 minutes. Figure 3.5 demonstrates training convergence after 70 epochs. Convergence here is defined as the earliest point in which error has less than a 10% change over 5 batches.

These results reflect the method only as a proof of concept and lack deep refinement. As discussed in Section 2.4.4, I augmented these results using a more refined method that included a better convergence metric and comparison of the autoencoder training and validation error to check model fit quality. Additionally, I show all image reconstruction comparisons with the same representative sample images used throughout this thesis for better comparison between methods (first shown in Section 2.2)

3.2.2 Final Results with Training Hyper-Parameter and Network Optimisation

As outlined in Section 2.4.4, my approach after the submission of my paper Ralph et al. (2018) was to refine the performance and hyper-parameters of the autoencoder. I achieved this end by optimising the autoencoder architecture and training conditions to determine the best configuration for the lowest network error and processing time. My aim in optimisation is to reduce the training time while achieving the lowest possible autoencoder error to ensure the latent feature vector has captured sufficient information for later analysis.

This Section evaluates the following hyper-parameters and network configurations (Section 2.4.5):

- Convolution and max-pooling filter dimensions testing
- Convolution layer filter count testing
- Training batch size testing
- Training set augmentations with random rotation and noise injection testing

With initial conditions:

- Convolutional filter size of 3×3 with a stride of [1,2,2,1]
- Max-pooling filter size of 2×2 with a stride of [1,2,2,1]
- Initial batch size of 128
- All tests conducted without training set augmentations

The following results are compared using several figures and tables. I demonstrate the optimal network and training configuration with a Figure showing the validation time per epoch and a second Figure illustrating a polynomial curve fit on a scatter plot of the average error at the end of each epoch. As outlined in Section 2.4.4.6, this convergence measure is the lowest of the average validation error in the last 10% of batches for each epoch. Using these figures, I determine the optimal variable by comparing the error and time taken to arrive at the convergent epoch. The configuration with the lowest error is deemed ideal unless indicated otherwise or if training time is of particular concern in the test. These results are also summarised in a series of tables and complimented where appropriate with comparisons of the autoencoder input images and corresponding reconstructions. Throughout these tests, I also show a series of reconstructions using the sample images used throughout the thesis. These images are shown with the original image and a difference image between the two. All of these figures contain only the reconstructions produced by the network in its trained state at the convergent epoch.

As discussed in Section 2.4.5, subsequent tests use the best configuration from previous tests to converge on the ideal overall solution. The final chosen parameter for each test is evaluated for training quality by comparing the validation and training error. In this comparison, a significant and consistently greater or smaller training error than the validation error indicates over-fitting and under-fitting respectively. Section 4.2 provides an in-depth discussion of these results, their implications and relation to the research questions and aims of this thesis (outlined in Sections 1.4 and 1.5 respectively).

3.2.2.1 Convolutional Filter Dimensions

The results of the convolutional filter size tests outlined in Section 2.4.4.1 are illustrated in Figures 3.6 - 3.11. The ideal convolution filter size in terms of accuracy for all layers is shown to be a 5×5, with the constant stride of [1,2,2,1] (discussed in Section 2.4.4.1) after 9 epochs.

As shown in Figure 3.7 and summarised in Table 3.1, this 5×5 receptive field size allows the autoencoder to reach convergence after 8 epochs in 1310.96 seconds with the lowest MSE validation error of all tests at 295.80. Although 2×2 filter it requires less training time and the 3×3 filter converges an epoch early, the error between these filers is significantly greater than the 5×5. Figures 3.9 - 3.11 also confirm the 5×5 filter as most suitable with comparably poor fidelity in networks trained with a 2×2 and a 7×7 filter. It is clear that across these figures that the resolution of the reconstruction is dependent on the filter size, with a small kernel producing the lowest resolution and the 7×7 with the greatest but requiring a great amount of training to refine. Figure 3.8 demonstrates the 5×5 filter is modelling correctly with no consistent variations between the two error metrics, indicating no obvious signs of under-fitting or over-fitting.

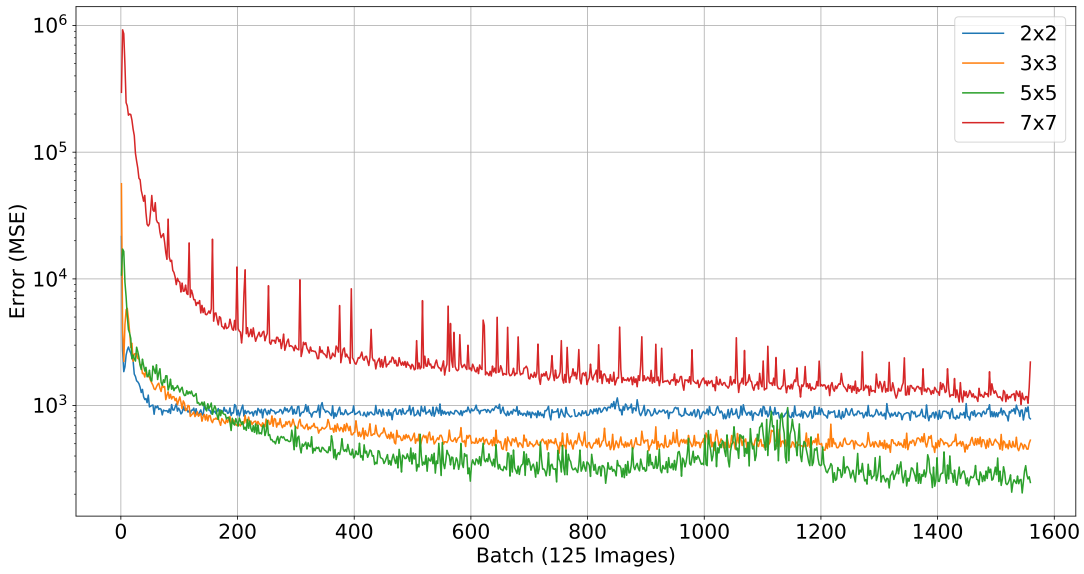

Figure 3.6: Comparisons of the validation MSE of the autoencoder with 2×2,3x3,5x5 and 7×7 sized convolutional filters, demonstrating the 5×5 as the best configuration with fast convergence and the lowest error.

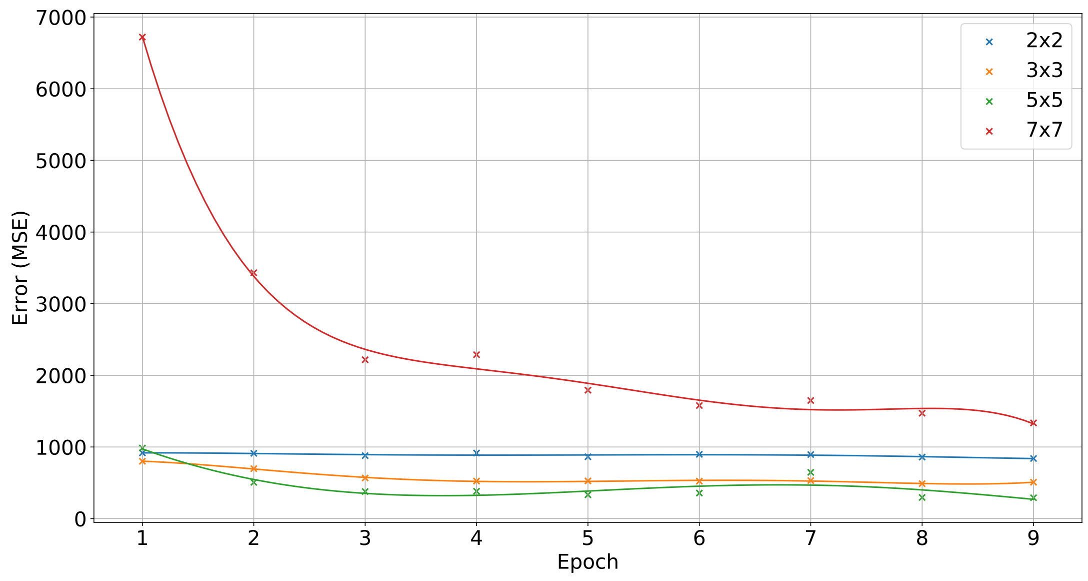

Figure 3.7: Scatter plot and polynomial fit of the average validation error in the final 10% of batches per epoch for all filter sizes. This Figure indicates the 5×5 filter is the most accurate.

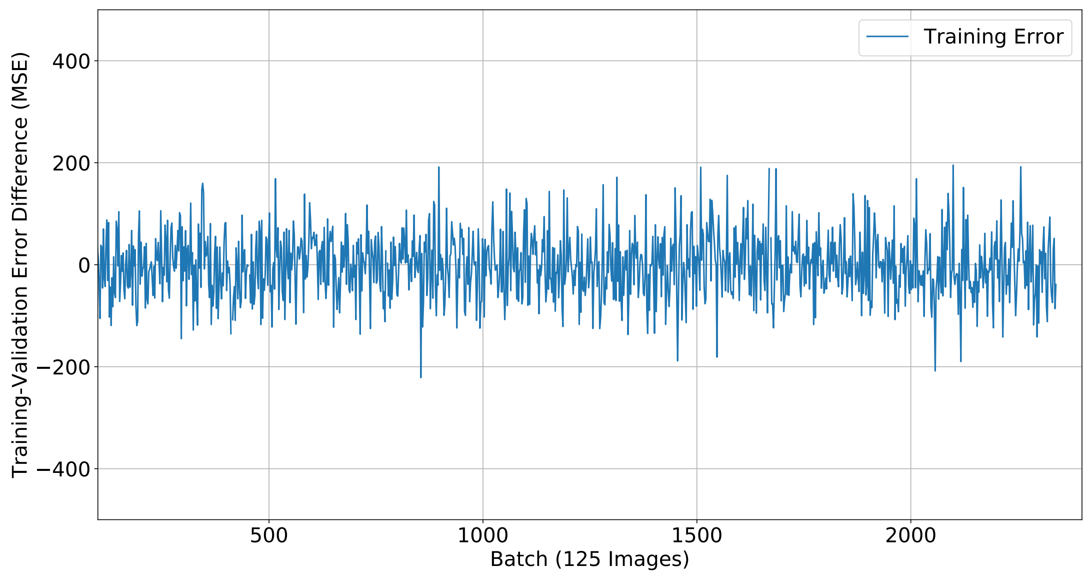

Figure 3.8: Per batch difference in training error and validation for 5×5 convolutional filter showing no obvious signs of over-fitting or under-fitting.

Table 3.1: Summary of convolutional filter size convergence statistics, indicating the 2×2 filter requires the least training time, while the 5×5 achieves the least error with an increase in training time

Convolutional Filter Size Test	Convergent Epoch	Convergent Epoch Error (MSE)	Training Time to Convergence (s)
2×2	9.0	858.57	**825.79**
3×3	**8.0**	488.55	954.67
5×5	9.0	**295.80**	1310.96
7×7	9.0	1471.42	1994.71

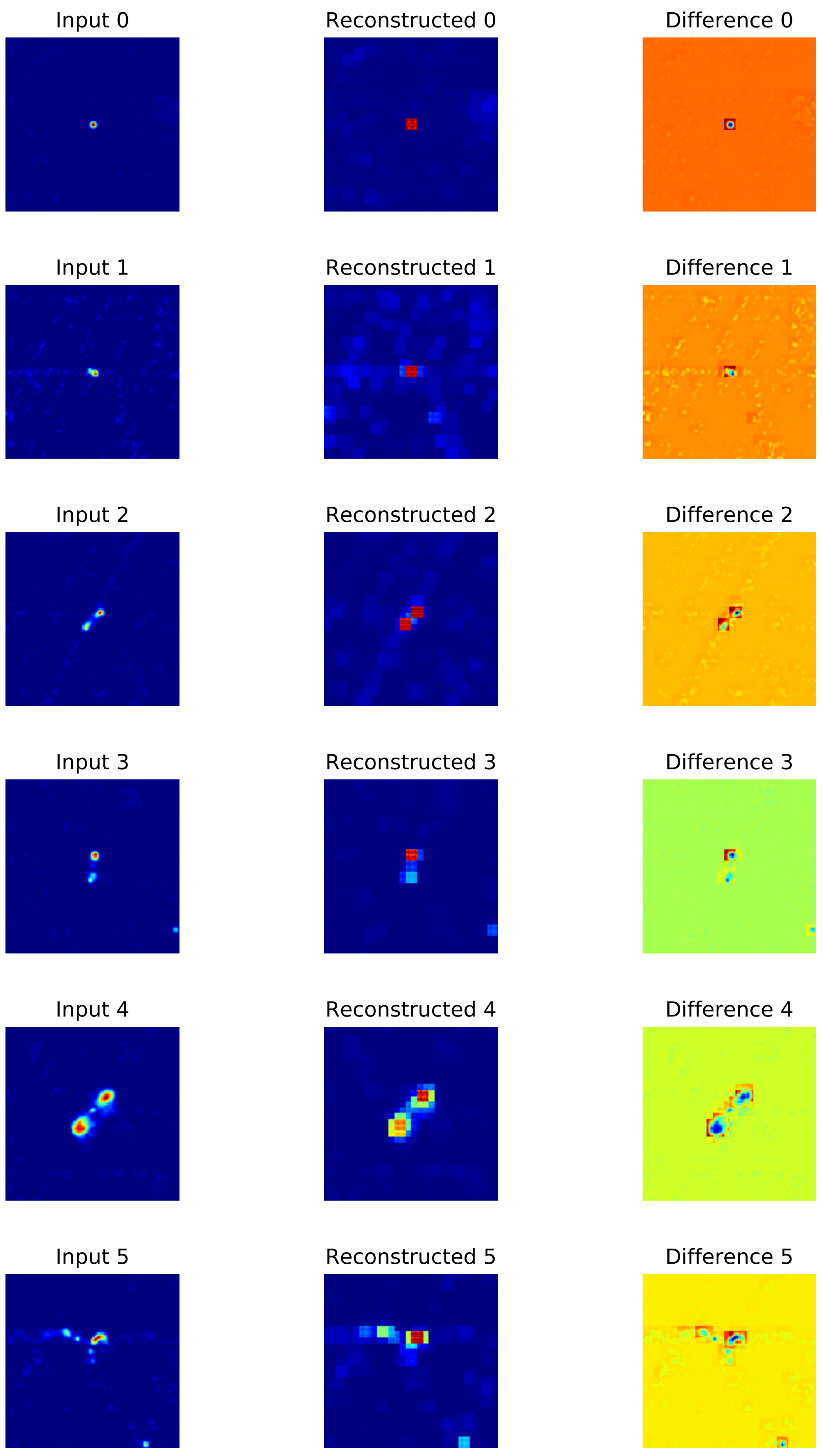

Figure 3.9: Sample input images and corresponding reconstruction and difference image outputs of an autoencoder with a 2×2 convolutional filter sizes. Evidently, a low receptive field will produce a correspondingly low resolution output.

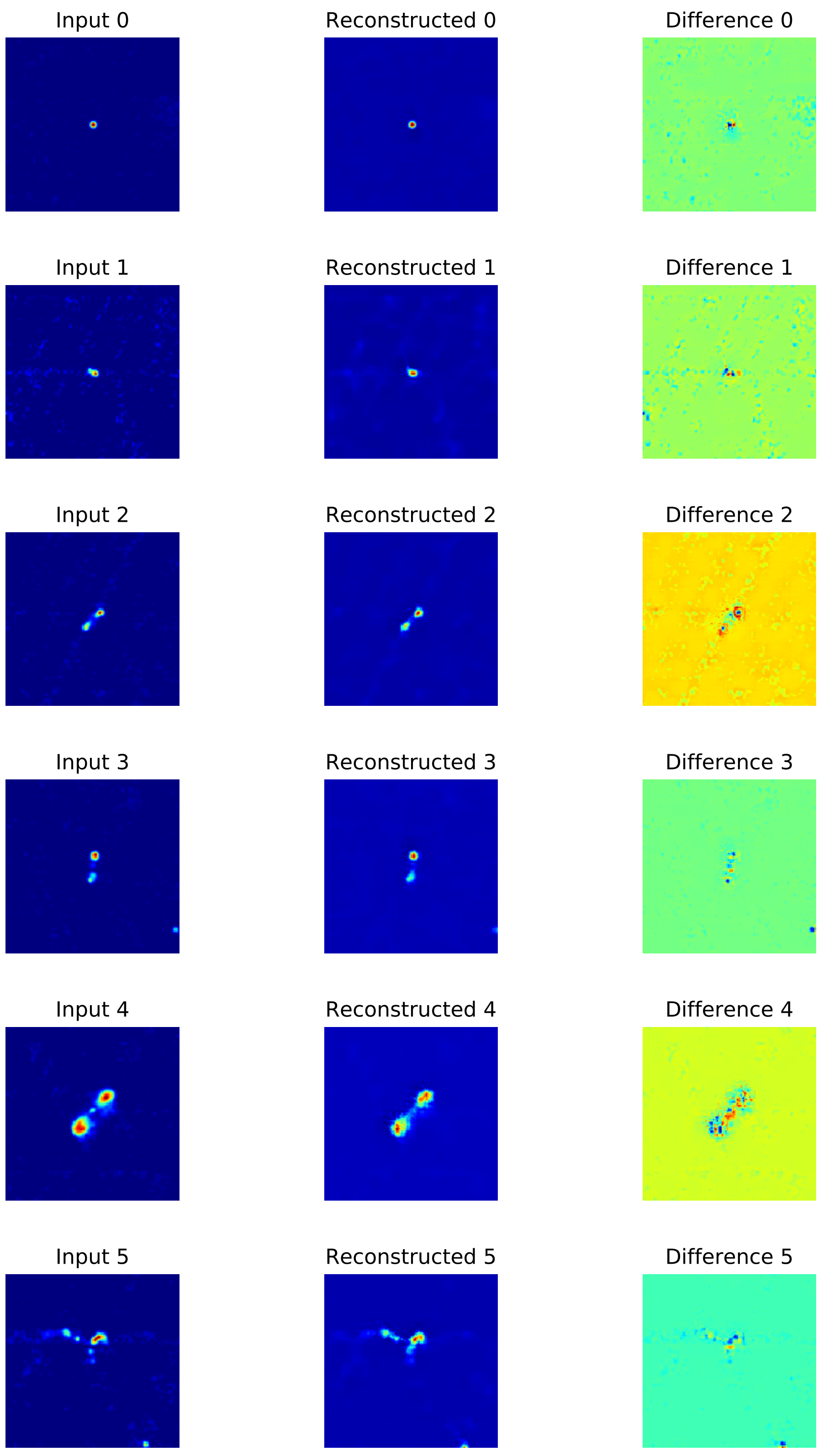

Figure 3.10: Sample input images and corresponding reconstruction and difference image outputs of an autoencoder with a 5×5 convolutional filter sizes. The lower error has a clear effect on the fidelity compared to the 2×2 filter, with clear preservation of components and peaks.

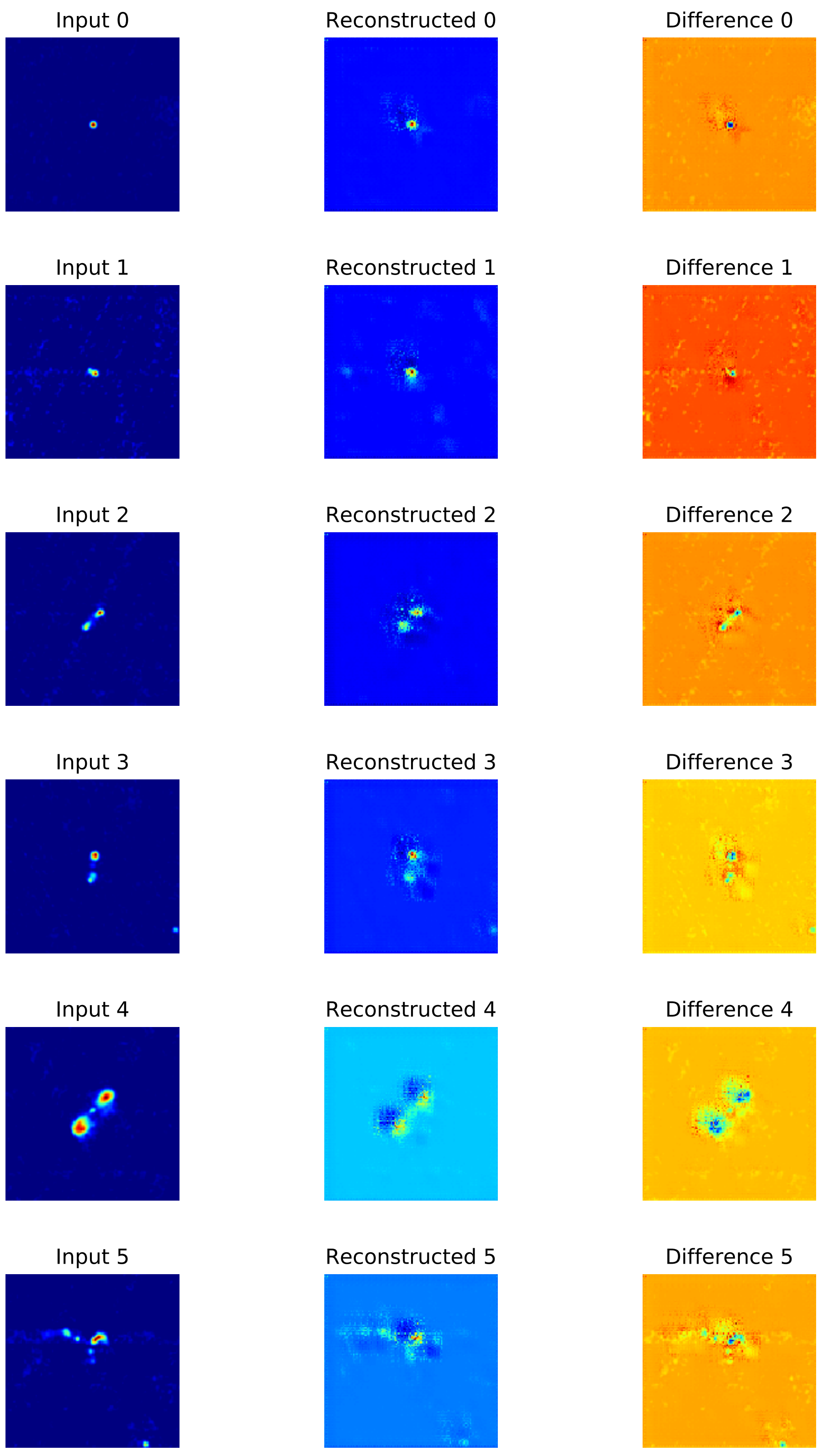

Figure 3.11: Sample input images and corresponding reconstruction and difference image outputs of an autoencoder with a 7×7 convolutional filter sizes. The reconstructions here poor fidelity and darkened regions from excessive zero padding, however the larger size of 7×7 filter is clearly attempting to fit high resolution detail to diffuse features.

3.2.2.2 Max-pooling Filter Dimensions

The results of the max-pooling filter size tests in the encoder side of the autoencoder are outlined in Section 2.4.4.2 are illustrated in Figures 3.12 - 3.18. The ideal max-pooling filter size here is more nuanced than simply selecting the most accurate in terms of low validation MSE. As outlined in Section 1.2.1, my purpose in including the max-pooling layers is to add a degree of translation invariance. As shown in Figures 3.15 - 3.18 and due to the principles pooling operates in, the 'blurring' effect is proportional to the size of the pooling filter. A filter of 1×1 is simply a perception layer and produces none of the blurring, but consequently produces no translation invariance. As a result, the 2×2 max-pooling filter was chosen as a trade-off between the most accurate with good fidelity in maintaining the number of peaks and components across all samples while still providing some essence of translation invariance. This filter is used with the constant stride of [1,2,2,1] (discussed in Section 2.4.4.2) and converges after 9 epochs.

As shown in Figure 3.12 and summarised in Table 3.2, this 2×2 filter allows the autoencoder to reach convergence after 9 epochs in 1473.98 seconds with the lowest MSE validation error greater than the 1×1 filter at 299.23. The training times, stability and convergence points across these networks do not differ enough to draw reasonable conclusions that would be observed in additional tests with different batch training samples. As in the previous convolutional filter size tests of Section 3.2.2.1, no obvious signs of under-fitting or over-fitting can be observed in the validation-training error of Figure 3.14.

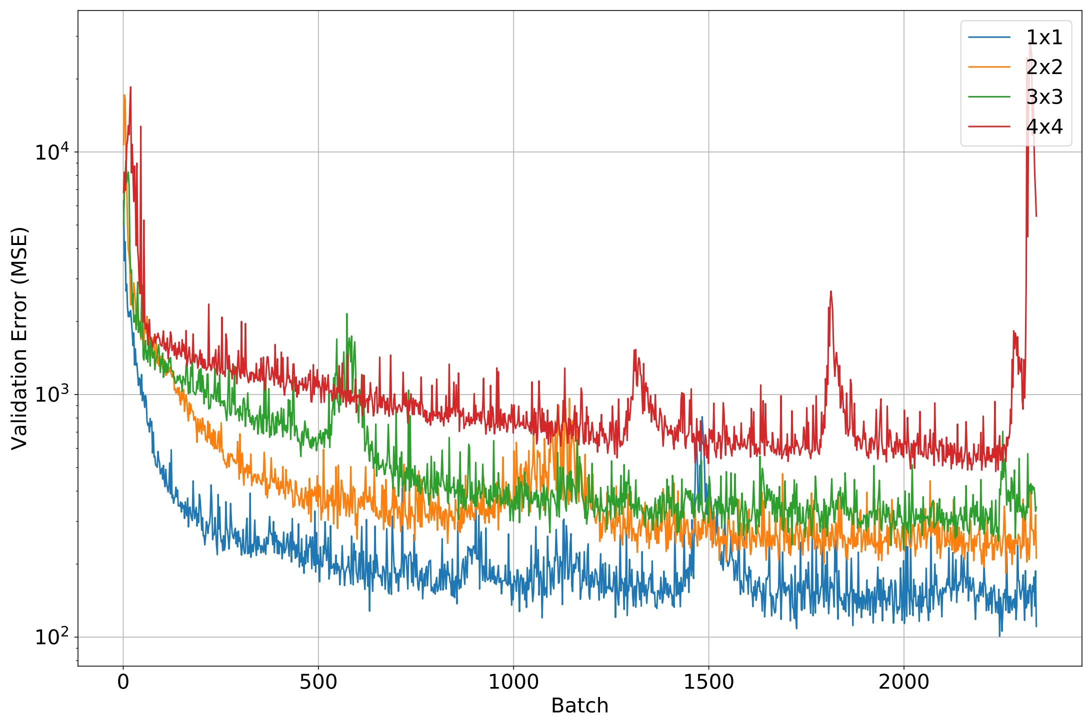

Figure 3.12: Comparisons of the validation MSE of the autoencoder with 2×2,3x3,5x5 and 7×7 sized convolutional filters, demonstrating the 5×5 as the best configuration with fast convergence and the lowest error.

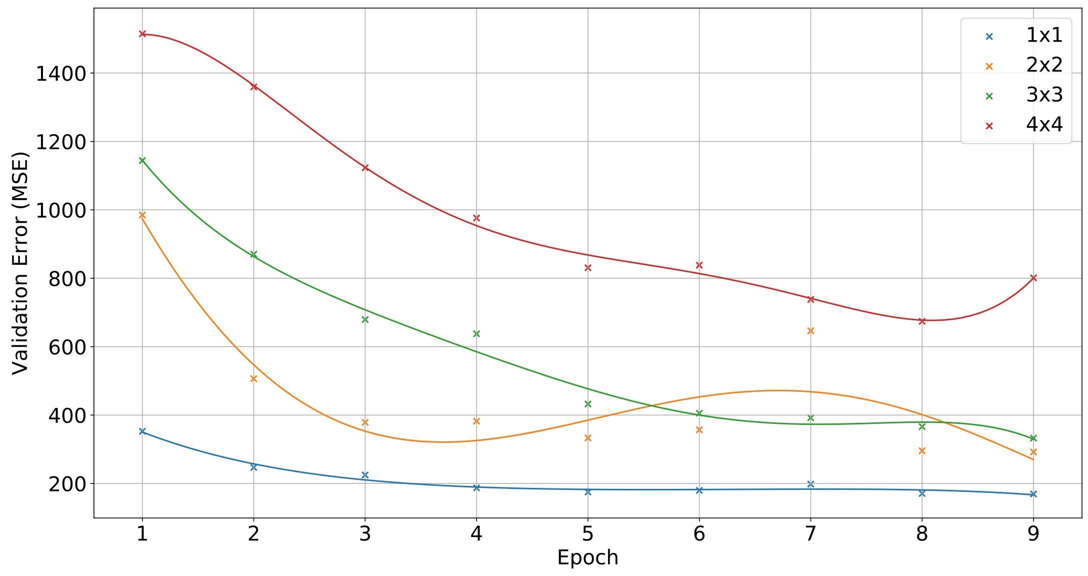

Figure 3.13: Scatter plot and polynomial fit of the average validation error in the final 10% of batches per epoch for all filter sizes. This Figure indicates the 2×2 filter is the most accurate filter that still carries the benefits of translation invariance found in filters greater than 1×1.

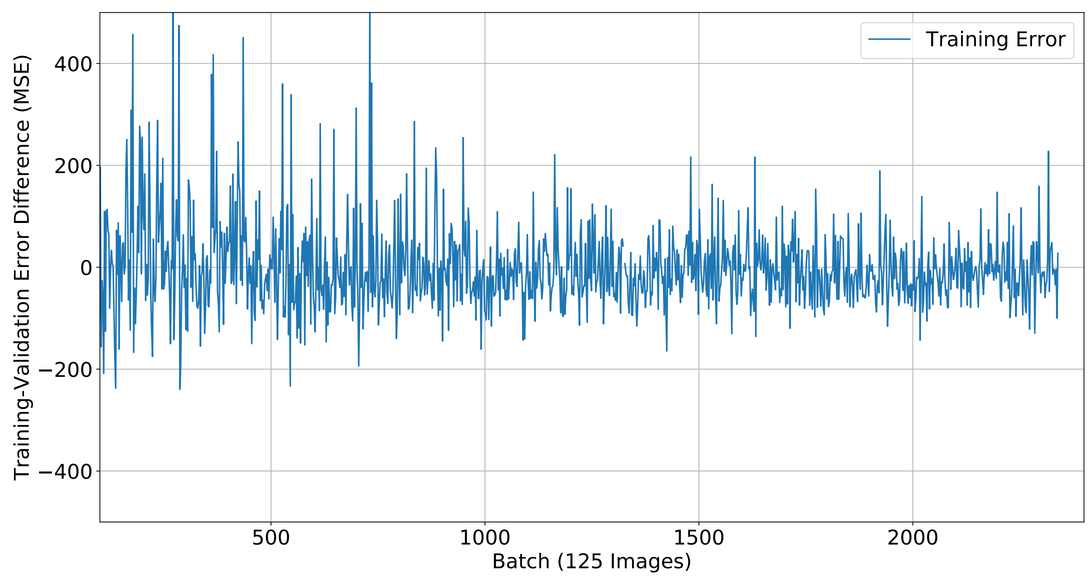

Figure 3.14: Per batch difference in training error and validation for 2×2 max-pooling filter, showing no obvious signs of over-fitting or under-fitting after the expected under-fitting in early batches that converges around the 1000 batch point.

Table 3.2: Summary of max-pooling filter size convergence statistics, indicating the 2×2 filter achieves the least error, discounting the 1×1 filter as inappropriate for true pooling. Training time across the tests have little variance

Max-Pooling Filter Size Test	Convergent Epoch	Convergent Epoch Error (MSE)	Training Time to Convergence (s)
1×1	9.0	**169.27**	1475.75
2×2	9.0	292.23	1473.98
3×3	9.0	332.72	1485.18
4×4	**8.0**	674.04	**1328.20**

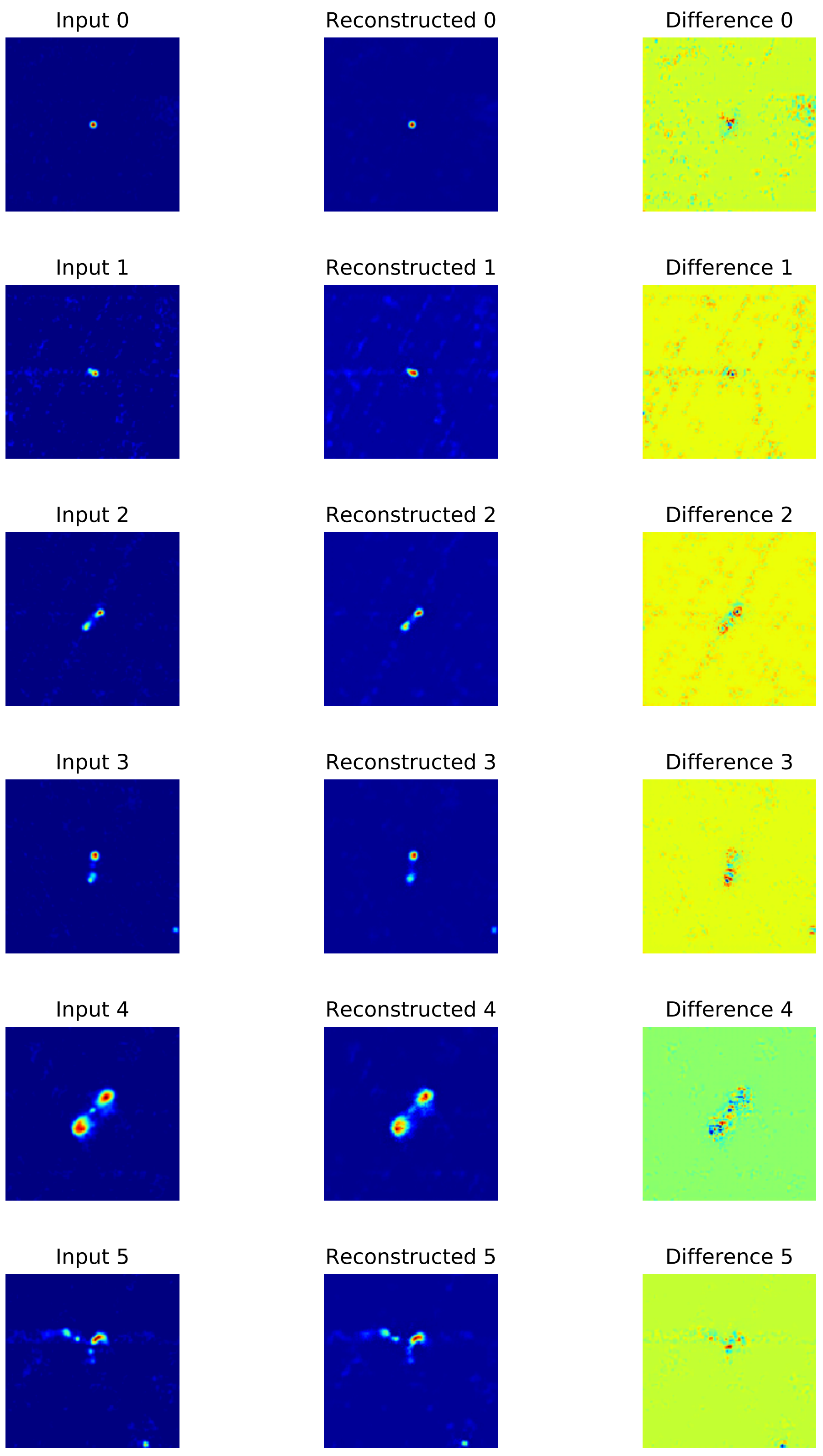

Figure 3.15: Sample input images and corresponding reconstruction and difference image outputs of an autoencoder with a 1×1 max-pooling filter size. The reconstructions have great fidelity and no blurring as the 1×1 filter acts as a perception layer.

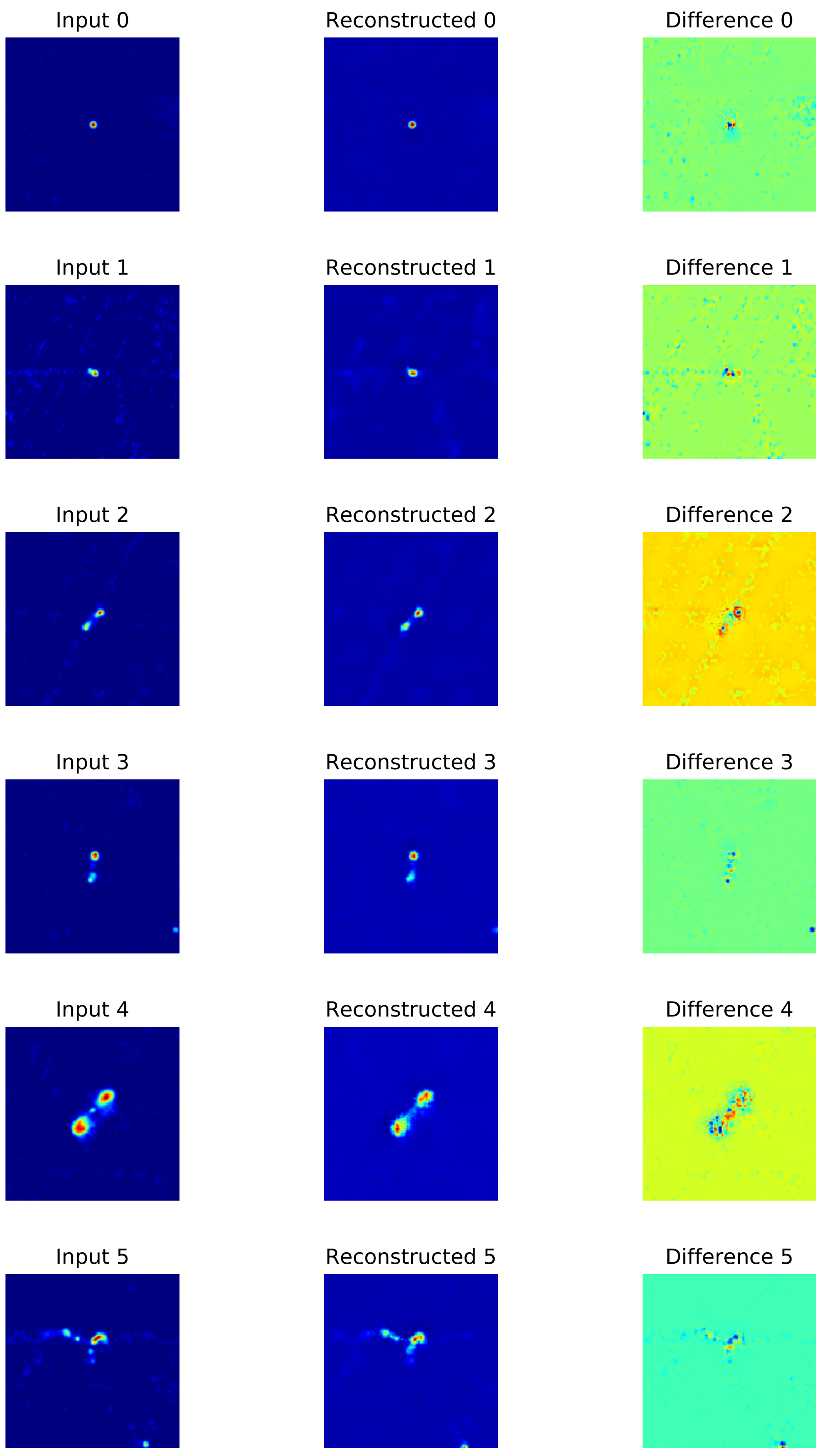

Figure 3.16: Sample input images and corresponding reconstruction and difference image outputs of an autoencoder with a 2×2 max-pooling filter size. These images show a slight drop in reconstruction compared to the 1×1 filter (Figure 3.15), but with the added element of minor translation invariance with the pooling.

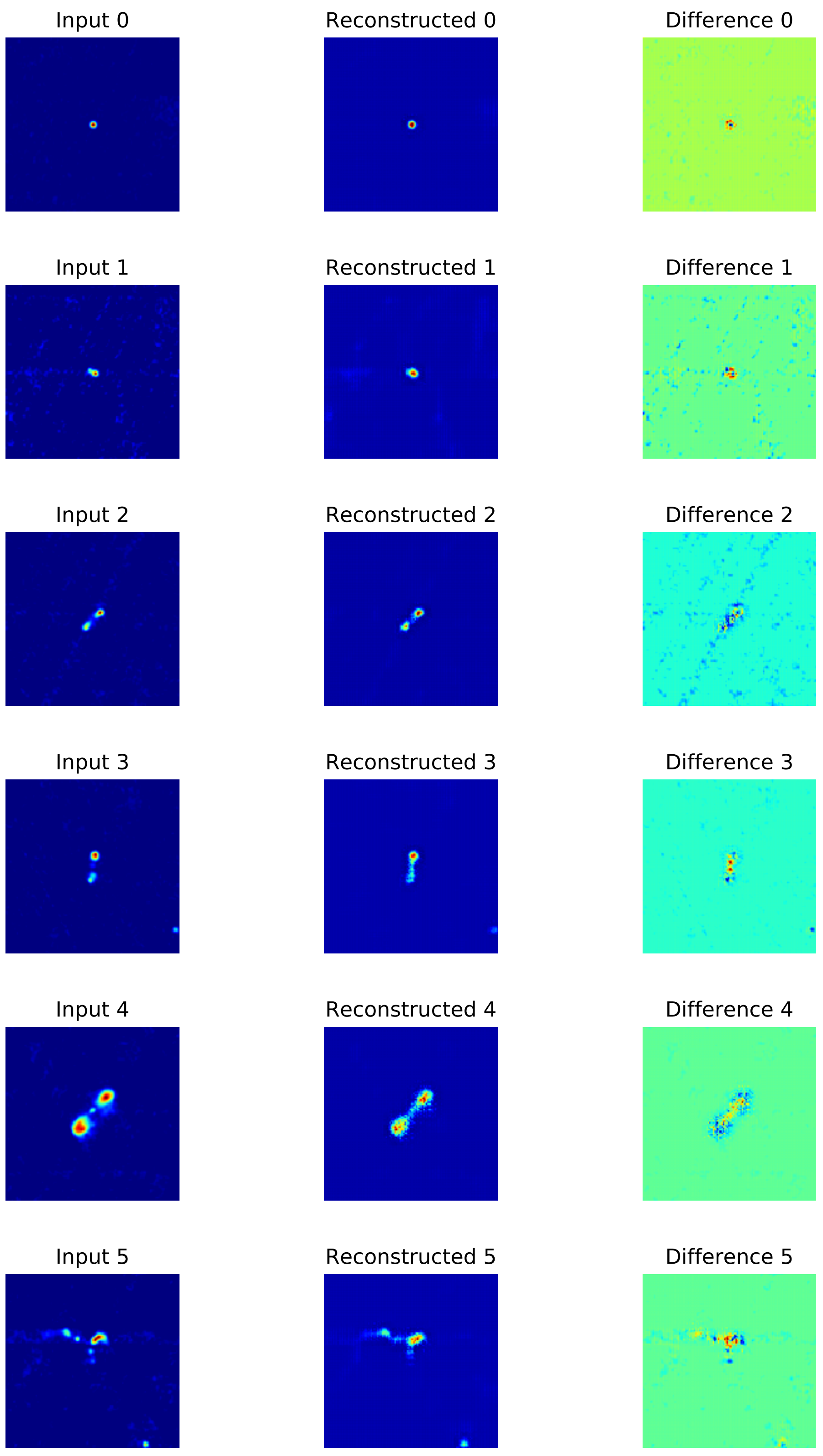

Figure 3.17: Sample input images and corresponding reconstruction and difference image outputs of an autoencoder with a 3×3 max-pooling filter size. These images show a continued blurring and degradation in reconstruction compared to the 2×2 filter (Figure 3.16).

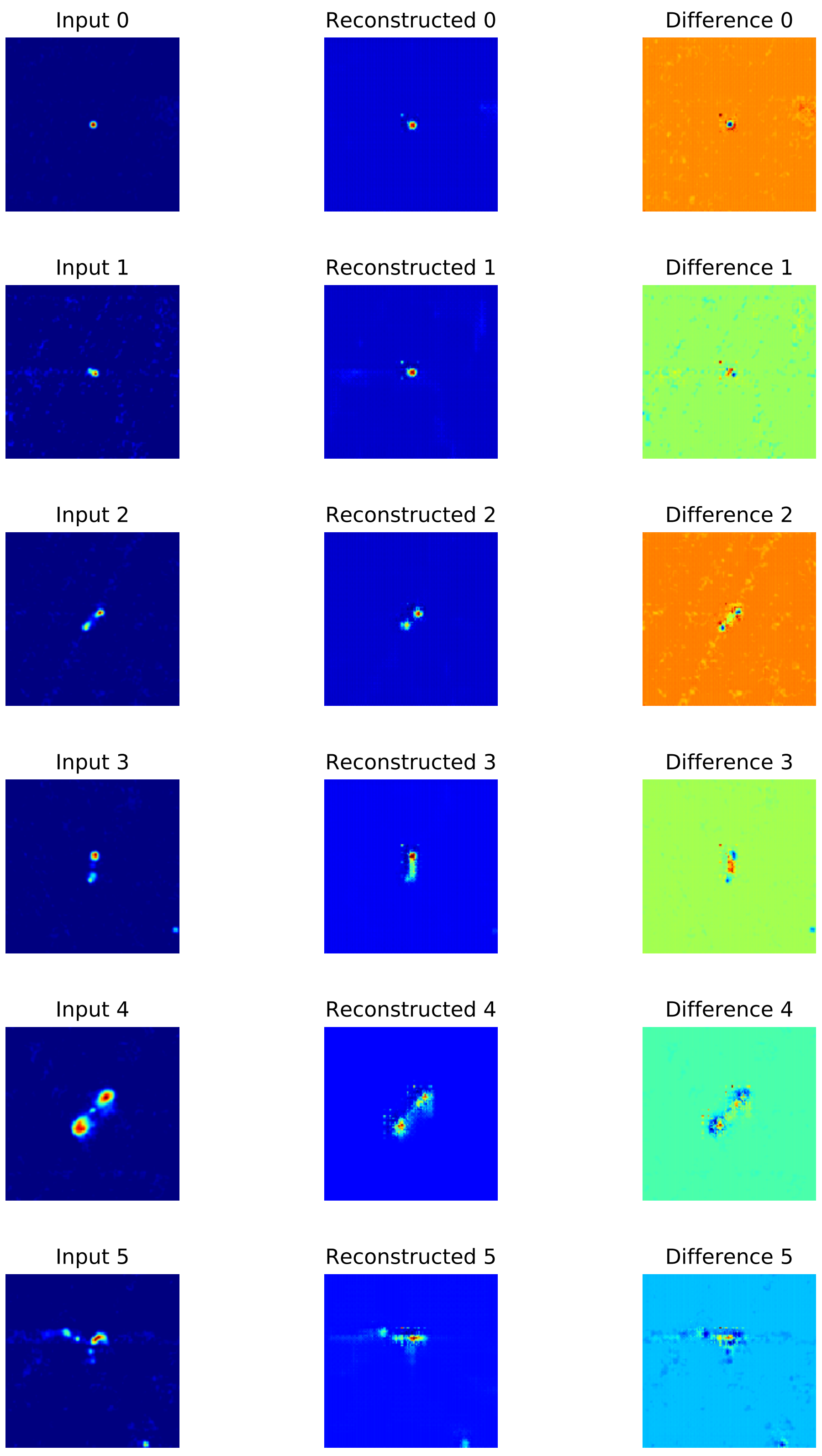

Figure 3.18: Sample input images and corresponding reconstruction and difference image outputs of an autoencoder with a 4×4 max-pooling filter size. These images show very poor fidelity and with components and peaks severely blurred compared to the 2×2 filter (Figure 3.16)

3.2.2.3 Convolutional Filter Counts

The results of the convolutional filter count tests as outlined in Section 2.4.4.1 are illustrated in Figures 3.19 - 3.28 and Tables 3.3 - 3.4. The ideal number of convolutional filters in the central hidden layer of both the encoder and decoder is shown to be 32 filters in terms of both accuracy and training time. As in previous tests, the network configurations and hyper-parameters I used in this test are the optimal parameters found in previous tests as 5×5 and 2×2 convolutional and max-pooling filters respectively (Sections 3.2.2.1 - 3.2.2.2). As shown in Figure 3.7 and summarised in Table 3.3, 32 convolutional filters allow the autoencoder to reach convergence after 8 epochs in 896.85 seconds with the lowest MSE validation error of all tests at 245.36. This is confirmed in Table 3.4 and Figures 3.21 - 3.22 where I refined the tests to examine values closer to the low error 64 and 32 configuration.

The training time observed in each test increases proportionally to the number of filters. This is similar to the trends seen while testing convolutional and max-pooling filter sizes (Sections 3.2.2.1 - 3.2.2.2). Shown in Figure 3.19 - 3.28, and summarised in Table 3.3, accuracy with few convolutional filters is acceptable and requires little training time. However, accuracy remains similar in tests with greater than 16 filters where changes in error are minimal but at the cost of increasing training time. In Table 3.4, the 24 layer configuration is an outlier, requiring more training time to converge as the consequence of requiring more overall time to reach a global minimum, not the training time per batch.

Figures 3.26 and 3.28 shows 32 and 128 layers respectively maintain the most components and peaks, although the 128 filter test illustrated in Figure 3.19 appears unstable with great fluctuations in error and appearing to converge the earliest as a result of these swings. Most tests show very similar convergence times and error. Tests with 64 layers are a close contender with very similar error, but with a substantial increase in training time. The fastest configuration with 32 filters is a clear choice with no accuracy increase offsetting the increased convergence time. Although the 128 filter configuration does converge after only 4 epochs, it requires comparatively more time to reach this point and with higher error than the 32 filter test.

As in previous tests, Figure 3.8 demonstrates the 32 filter configuration is modelling correctly with no consistent variations between the two error metrics, indicating no obvious signs of under-fitting or over-fitting. As the autoencoder is becoming more refined with the optimal convolutional and max-pooling filter sizes determined, this Figure is displaying less differences than previous tests in Figure 3.14 and Figure 3.8.

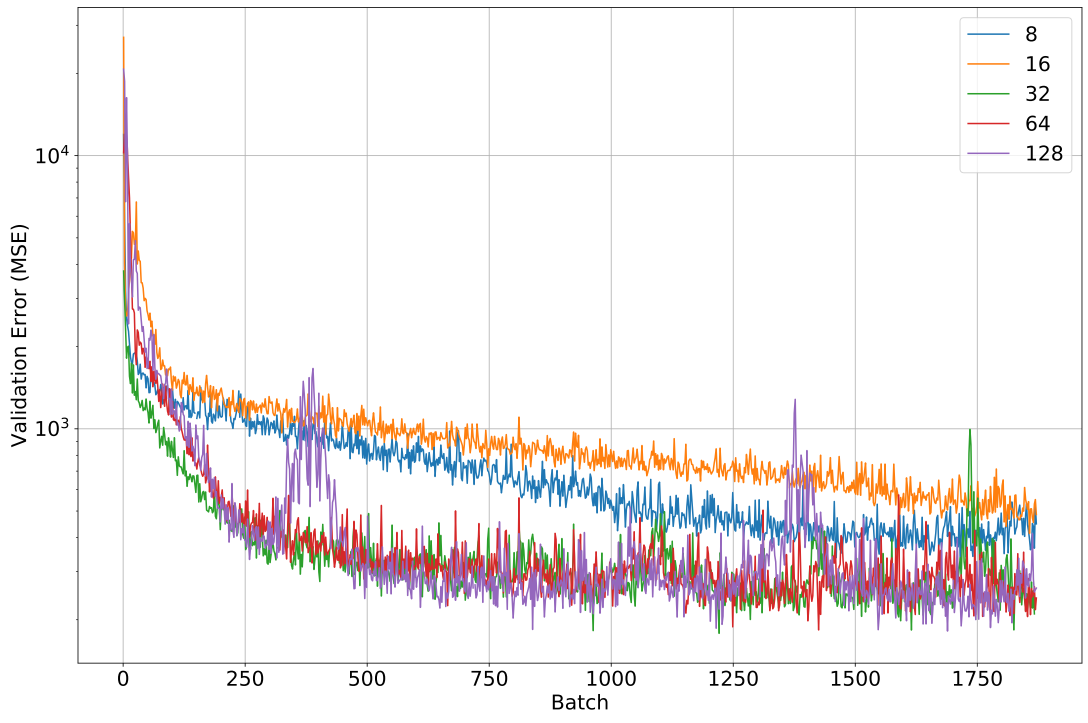

Figure 3.19: Comparisons of the validation MSE of the autoencoder with 8,16,32,64,128 convolutional filters in the hidden middle layer of the encoder and decoder. This Figure demonstrates 32 filters as the best configuration with the fastest convergence and the lowest error.

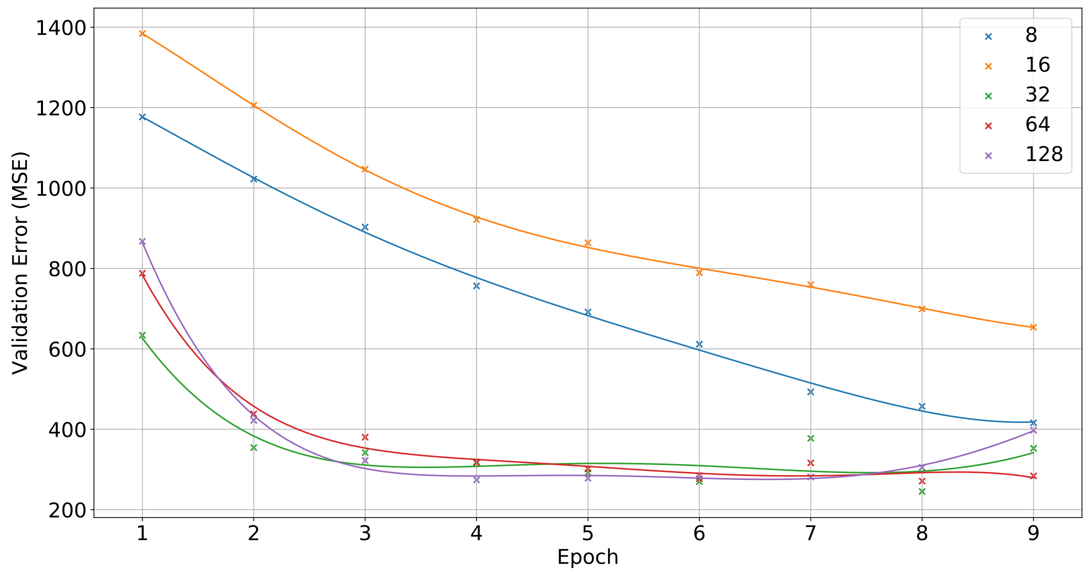

Figure 3.20: Scatter plot and polynomial fit of the average validation error in the final 10% of batches per epoch for all filter counts. This Figure indicates the 32 filter configuration is the most accurate.

Table 3.3: Summary of convolutional filter count statistics, indicating that the 32 filter configuration is the most accurate and fastest to converge in terms of time. The 128 filter configuration does converge after only 4 epochs but requires significant more time to reach this point, and with higher error than the 32 filter test.

Convolutional Filter Count Test	Convergent Epoch	Convergent Epoch Error (MSE)	Training Time to Convergence (s)
8	9.0	416.03	**543.05**
16	9.0	653.90	734.30
32	8.0	**245.36**	896.85
64	8.0	271.13	1321.78
128	**4.0**	273.99	1246.60

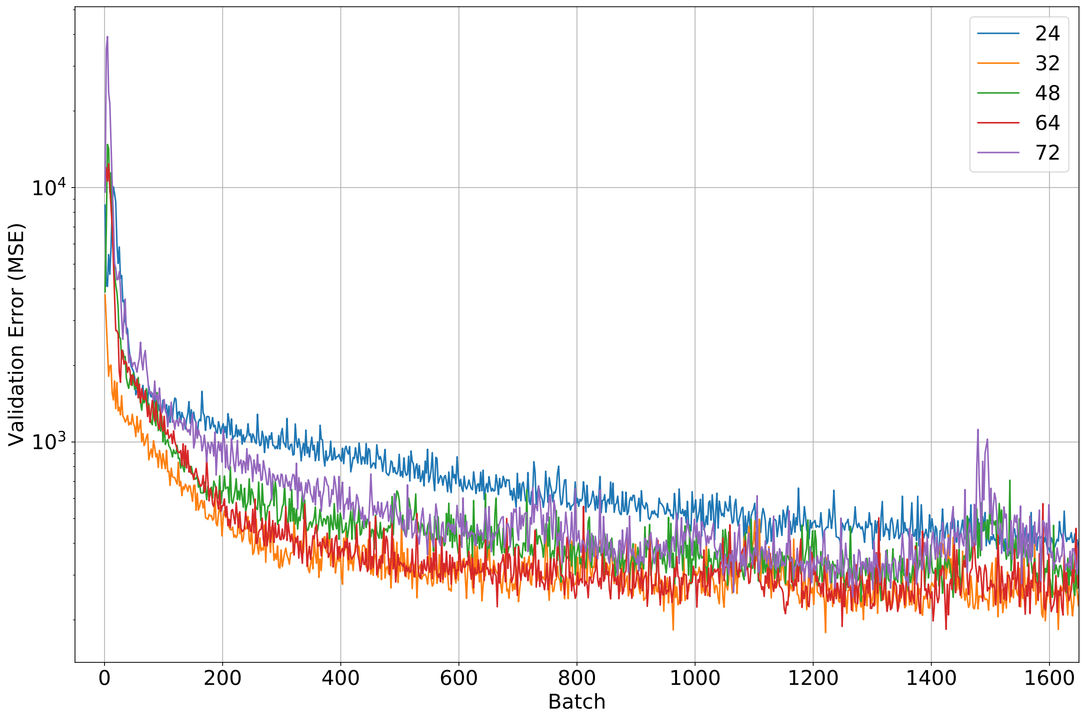

Figure 3.21: Comparisons of the validation MSE of the autoencoder using a more refined testing condition than earlier tests, with 24,32,48,64,72 convolutional filters in the hidden middle layer of the encoder and decoder. This confirms the 32 filter count as the best configuration with fast convergence and the lowest error.

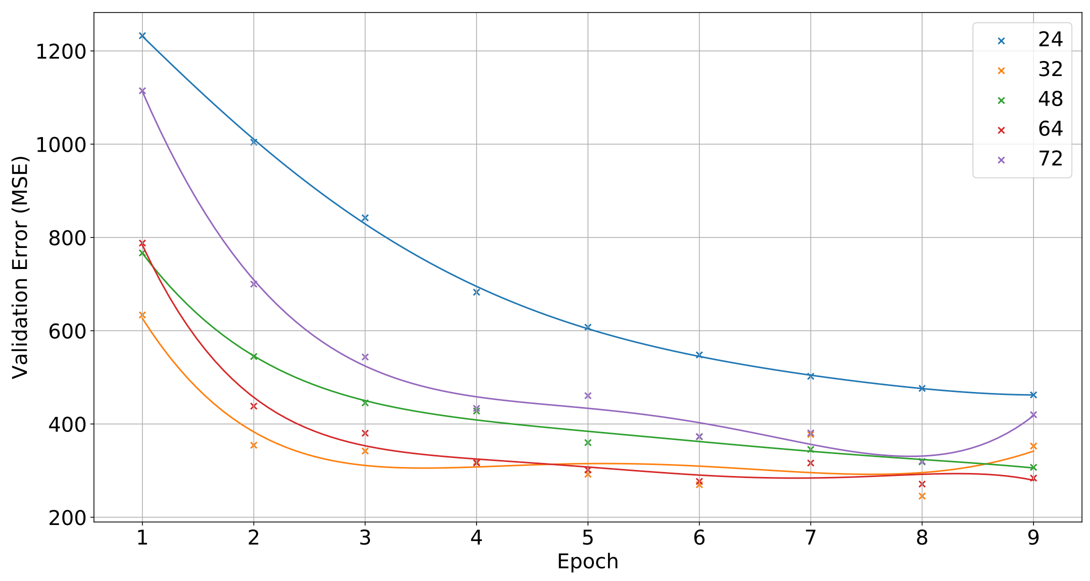

Figure 3.22: Scatter plot and polynomial fit of the average validation error in the final 10% of batches per epoch for the refined filter count tests. This Figure confirms the 32 filter configuration is the most accurate.

Table 3.4: Summary of the refined convolutional filter count statistics, confirming that the 32 filter configuration is the most accurate and fastest to converge.

Convolutional Filter Count Test	Convergent Epoch	Convergent Epoch Error (MSE)	Training Time to Convergence (s)
24	9.0	462.36	938.28
32	8.0	**245.36**	**896.85**
48	9.0	307.00	1249.08
64	8.0	271.13	1321.78
72	8.0	319.90	1535.96

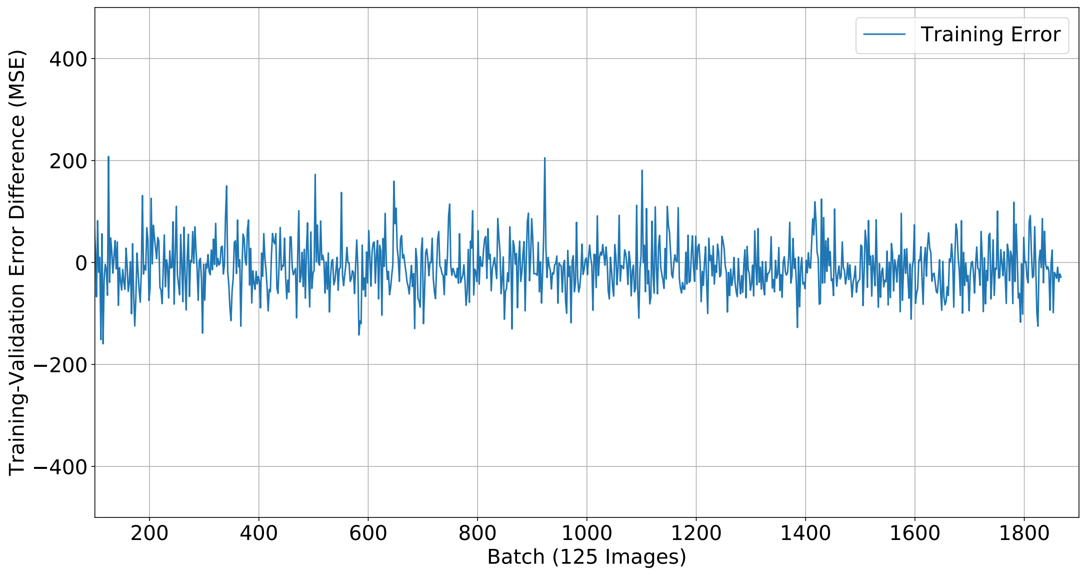

Figure 3.23: Per batch difference in training error and validation for the 32 convolutional filter configuration, showing no obvious signs of over-fitting or under-fitting. Clear improvement in modelling compared to Figure 3.14, where there are no major signs of under-fitting after 100 epochs.

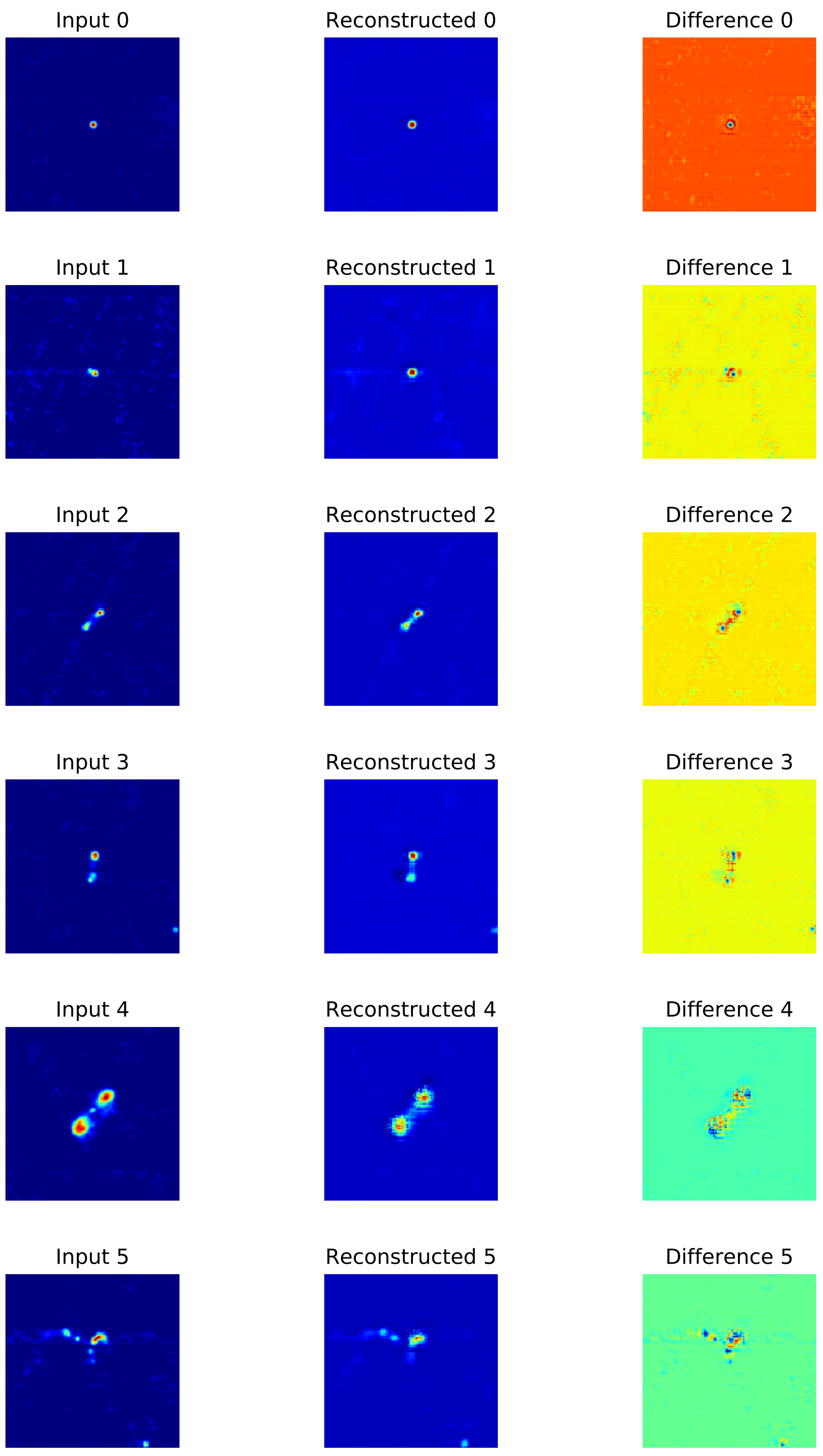

Figure 3.24: Sample input images and corresponding reconstruction and difference image outputs of an autoencoder with a 8 convolutional filter configuration. The reconstructions contain reasonable fidelity with moderate blurring and merging of separate components.

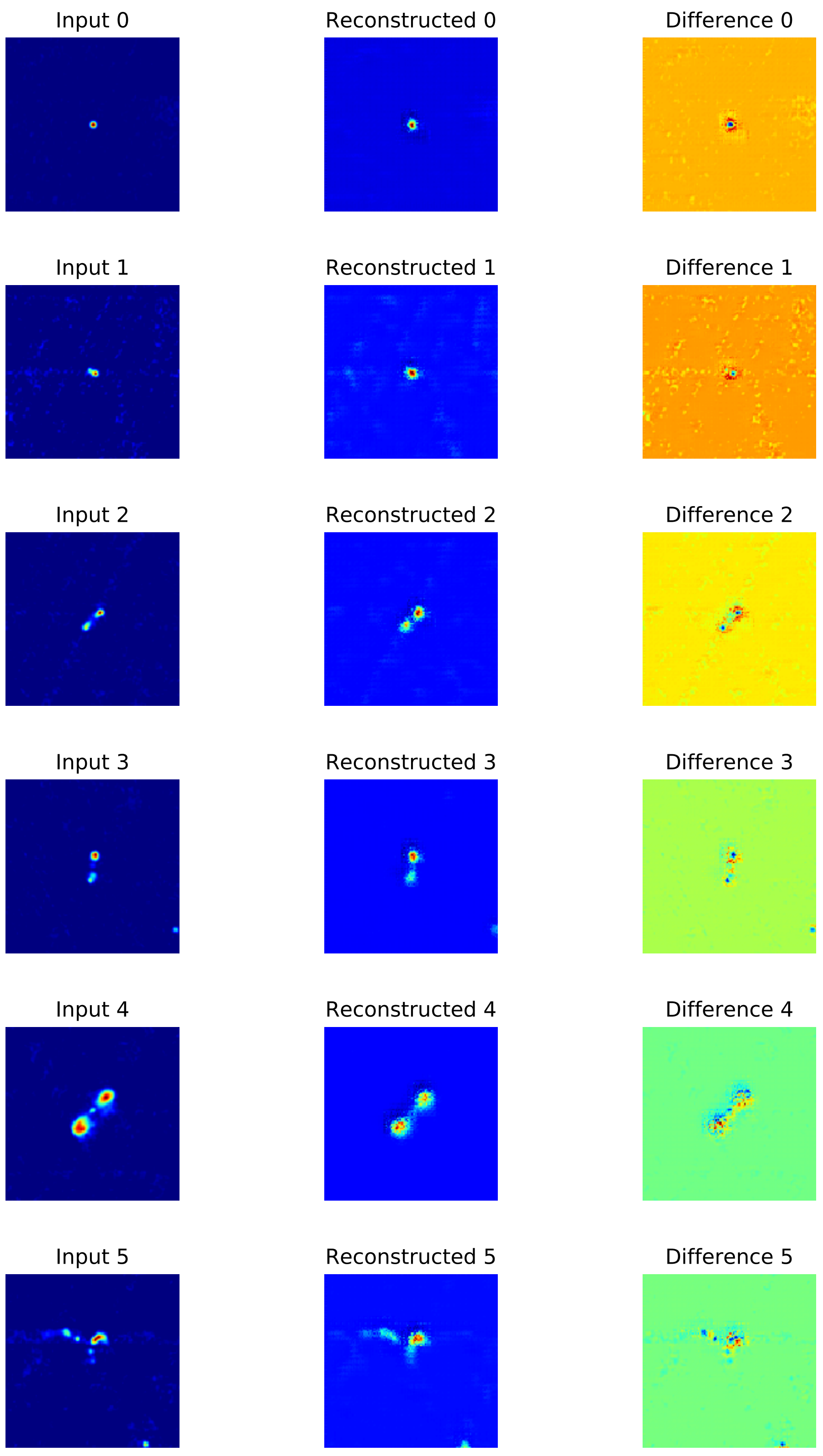

Figure 3.25: Sample input images and corresponding reconstruction and difference image outputs of an autoencoder with a 16 convolutional filter configuration. The reconstructions show improved quality with reduced component merging compared to the 8 filter configuration (Figure 3.24), but with poor reconstruction of the background noise.

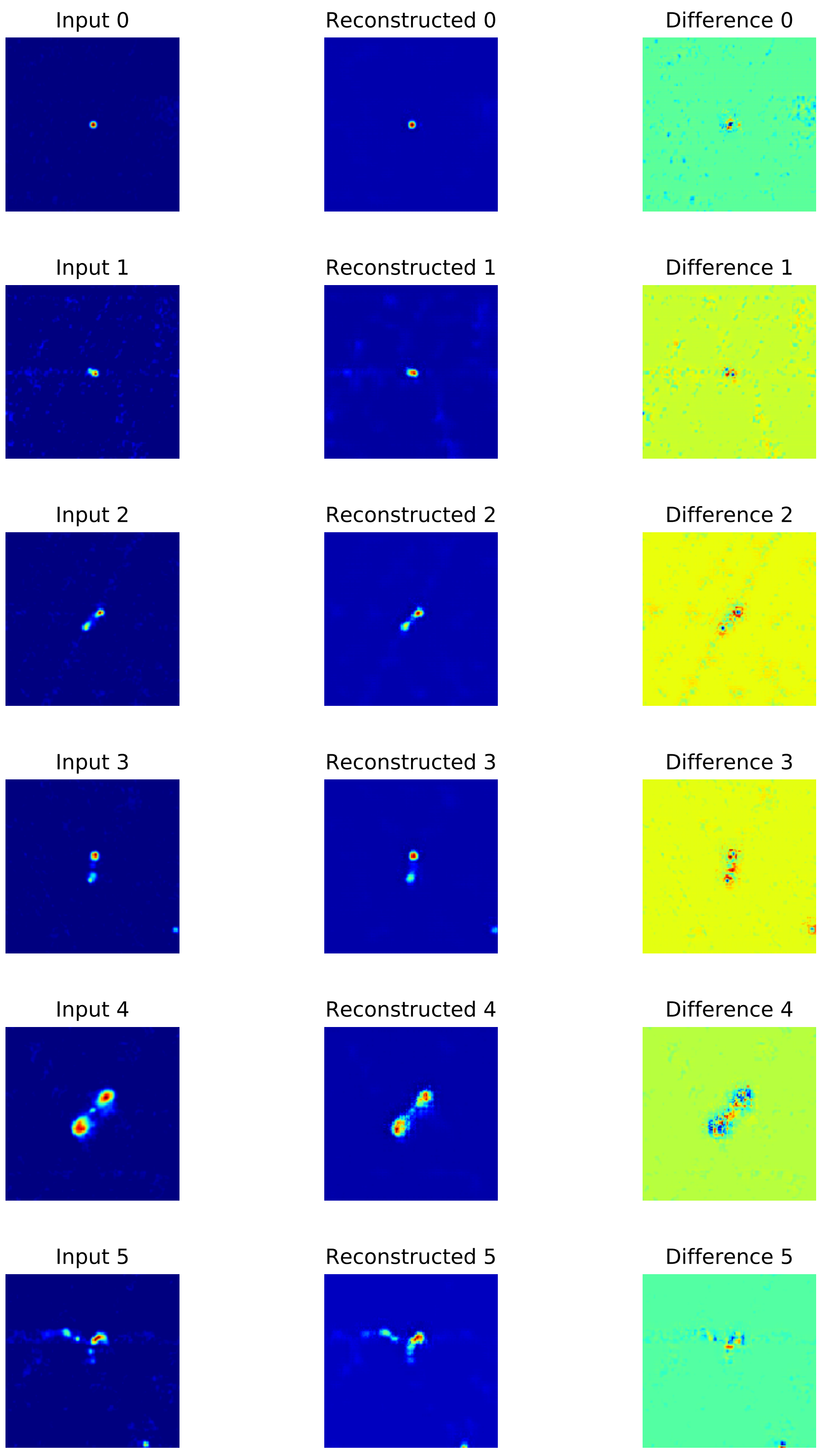

Figure 3.26: Sample input images and corresponding reconstruction and difference image outputs of an autoencoder with a 32 convolutional filter configuration. The reconstructions are highly accurate, with all components and peaks preserved and sound regeneration of the background noise. Most notable is the preservation of the central component of the AGN in Reconstructed Image 4, which has not been seen in previous tests.

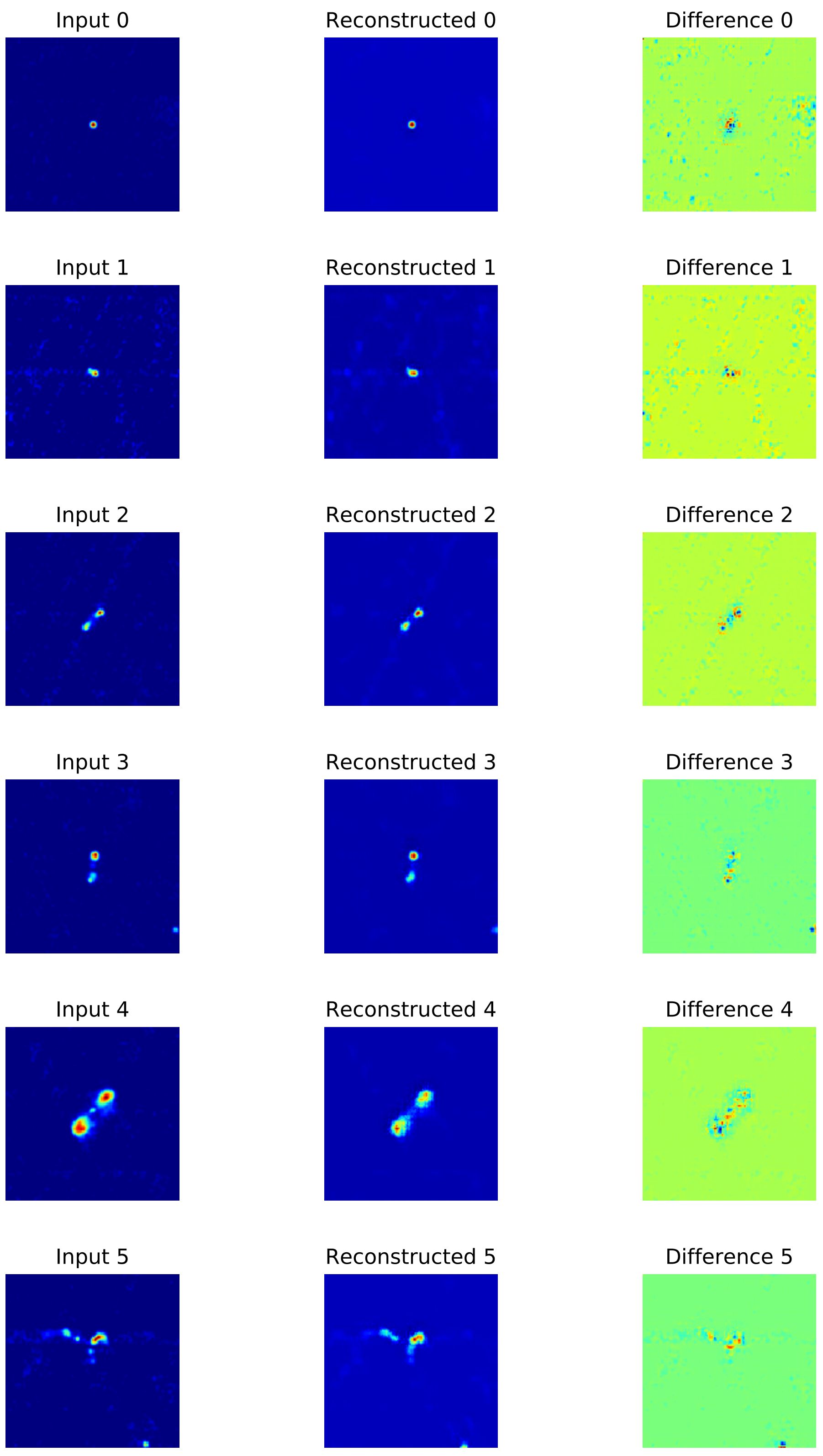

Figure 3.27: Sample input images and corresponding reconstruction and difference image outputs of an autoencoder with a 64 convolutional filter configuration. The reconstructions here are acceptable, but with lower quality than the previous 32 filter test (Figure 3.26) and loss of the central component in Reconstruction Image 4.

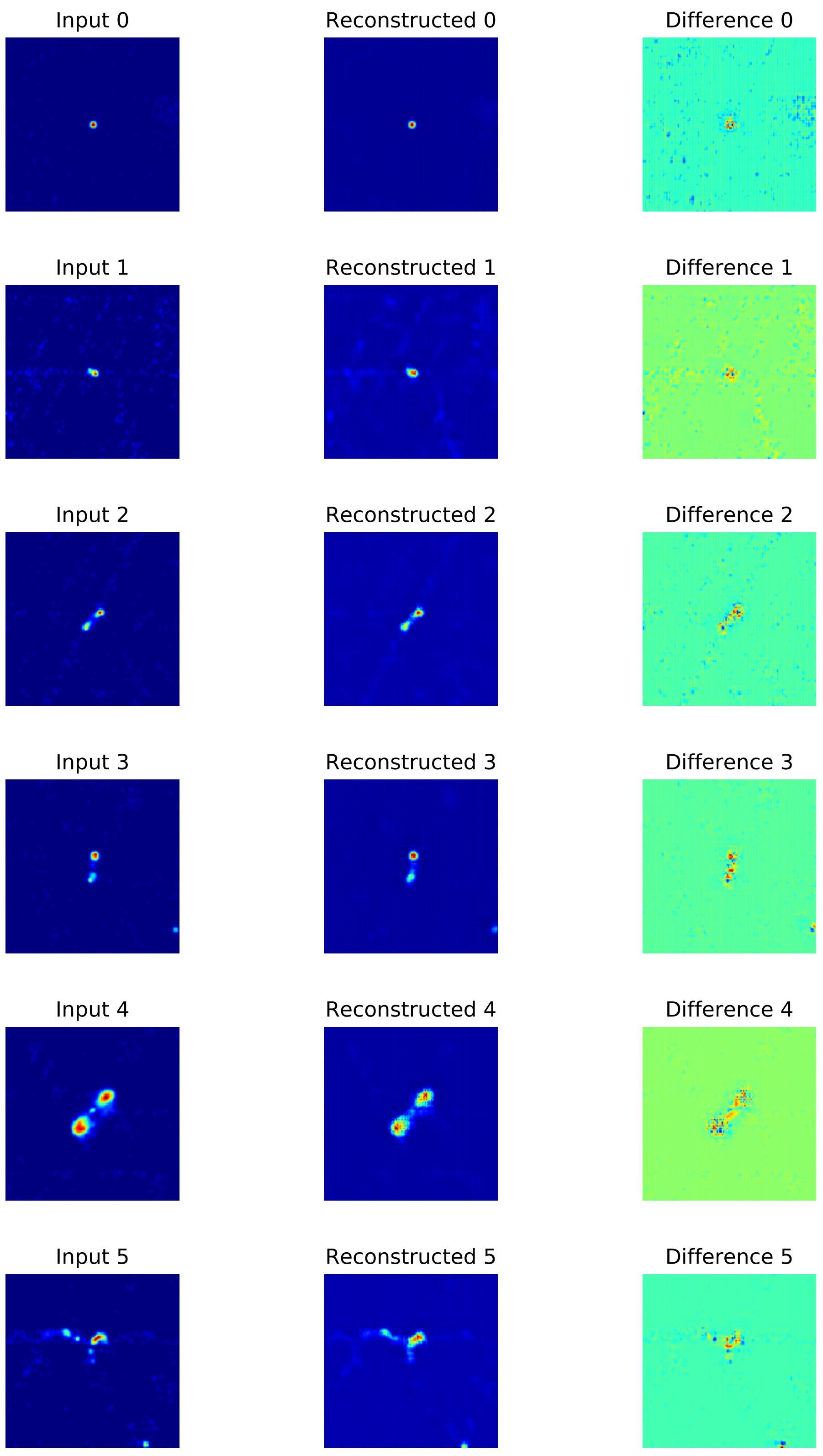

Figure 3.28: Sample input images and corresponding reconstruction and difference image outputs of an autoencoder with a 128 convolutional filter configuration. The reconstructions here are also highly accurate, with the least amount of blurring across all tests, but with lower quality than the previous 32 filter test (Figure 3.26) due to slightly fainter peaks.

3.2.2.4 Batch Size

The results of the training batch size tests outlined in Section 2.4.4.6 are illustrated in Figures 3.29-3.37 and Table 3.5. These tests show the ideal training batch size is 4 images per batch in terms of the trade off between accuracy, training time and stability for the autoencoder in my system. As in previous tests, the network configurations and hyper-parameters I used in this trial are the optimal parameters found in previous tests as 5×5 and 2×2 convolutional and max-pooling filters respectively (Sections 3.2.2.1 - 3.2.2.2) with 32 convolutional filters in the middle hidden layer of the encoder and decoder (Section 3.2.2.3). As shown in Figures 3.29-3.30 and summarised in Table 3.5, 4 RGZ images per batch allow the autoencoder to reach convergence after 3 epochs in 528.34 seconds with a MSE validation error at 7.48.

These figures indicate that batch sizes below 64 images are the most accurate and converge the fastest, but at the cost of error stability during training. This trend continues from the original batch size of previous tests at 128 images to larger batches of 256 images, where accuracy is more stable but gradually reduced with significant blurring in the reconstructions. These findings are confirmed with the gradually degrading reconstruction quality of autoencoder reconstructions in Figures 3.34-3.37 and Figure 3.26 as the batch size of 128 used in previous. In these figures, training batch sizes of 16 and below (Figure 3.35) appear to be the most accurate, which is reinforced in Figure 3.31. Although the batch sizes of 1 and 2 images are the most accurate and the fastest to converge, these are also the least stable, as shown in Figure 3.31. As a result, I chose a batch size of 4 as a trade-off between stability, accuracy and training time. This results in a marginally higher error and a slight increase in training time but allows the autoencoder to be more robust and generalisable to large variations in the images within training batches. Reconstructions using these batch sizes in Figure 3.34 compared to the batch size of 4 in Figure 3.26 justify this choice with comparatively little fidelity loss for a great stability increase. Although the overall validation accuracy here is higher, this is at the cost of not fully reconstructing some faint features such as the central peak at the centre of Reconstruction Image 4 of the 4 batch size test reconstructions.

As in previous tests, Figure 3.32 demonstrates training with a batch size of 4 allows the autoencoder to correctly model the training set with no consistent variations between the two error metrics, indicating no obvious signs of under-fitting or over-fitting. The autoencoder is continuing to be refined with the optimal convolutional and max-pooling filter sizes and counts determined. This is demonstrated where this Figure displays less differences than previous tests in Figures 3.14,3.8 and 3.23. Figure 3.33 illustrates the inherent instability of training with such a small batch size of 4 images. This shows where a difficult or complex training batch sample can lead to great fluctuations in error, an effect that can be observed in higher batch sizes, but to a lesser extent with the generalisation provided by a greater number of samples.

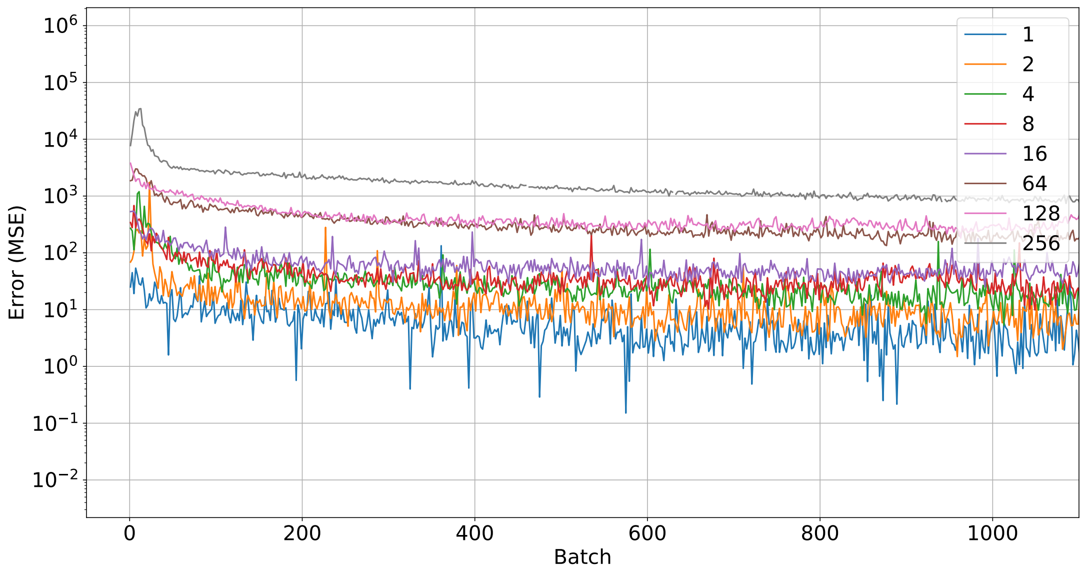

Figure 3.29: Comparisons of the validation MSE of the autoencoder with 2,4,8,16,64,128 batch sizes during training. This Figure demonstrates a batch size of 4 as the best configuration with the fast convergence and the lowest error. Full 10 epochs of data here is difficult to show meaningfully, only the first 1200 batches as in previous tests are shown. Error up to 50000 batches is shown in Figure 3.33.

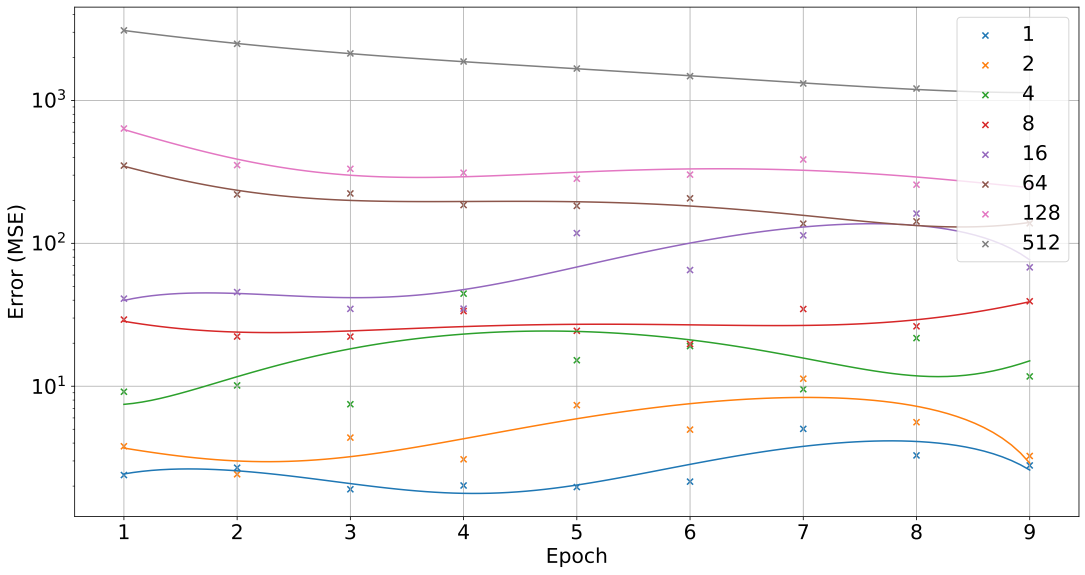

Figure 3.30: Scatter plot and polynomial fit of the average validation error in the final 10% of batches per epoch for all batch sizes. This Figure indicates a batch size of 4 is the most accurate, closely followed by a batch size of 8. A large increase in error is observed at epoch 9 for the 4 batch size test. This spike coincides with the peaks seen in Figure 3.33.

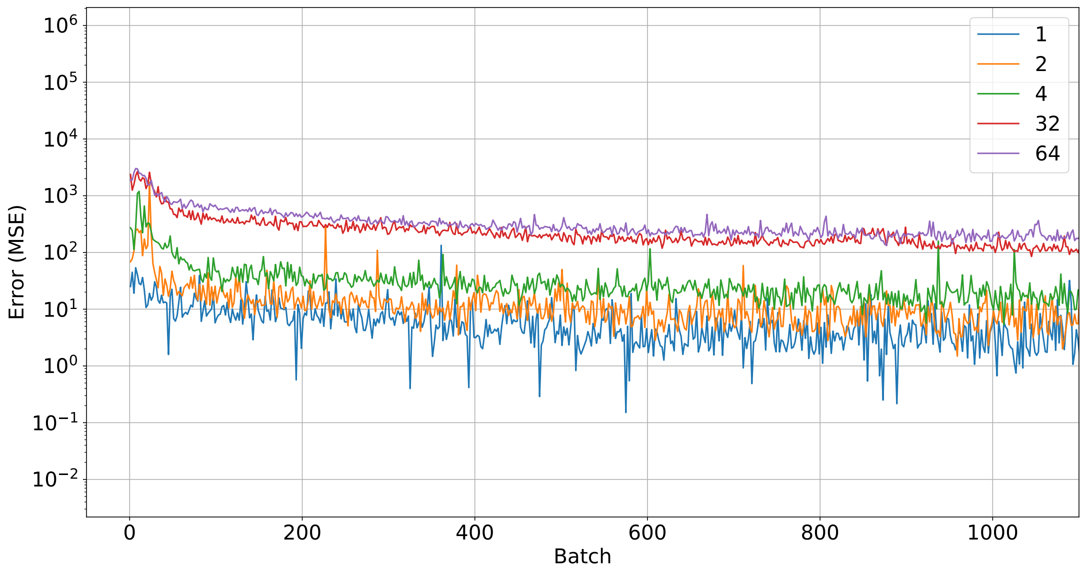

Figure 3.31: Comparisons of the validation MSE of the autoencoder with 1,2 and 4 batch sizes during training. This Figure demonstrates a batch size of 4 as the best configuration as a trade-off between low error and stability. Full 10 epochs of data here is difficult to show meaningfully, only the first 1200 batches as in previous tests are shown. Error up to 50000 batches is shown in Figure 3.33.

Table 3.5: Summary of batch size test statistics, indicating that the training batch size of 4 is the most accurate and fastest to converge in terms of time.

Batch Size Test	Convergent Epoch	Convergent Epoch Error (MSE)	Training Time to Convergence (s)
1	3.0	**1.90**	**465.69**
2	**2.0**	2.42	485.20
4	3.0	7.48	528.34
8	6.0	19.75	1205.27
16	3.0	34.78	589.98
64	7.0	137.28	640.24
128	9.0	251.53	983.02
512	9.0	1128.21	937.50

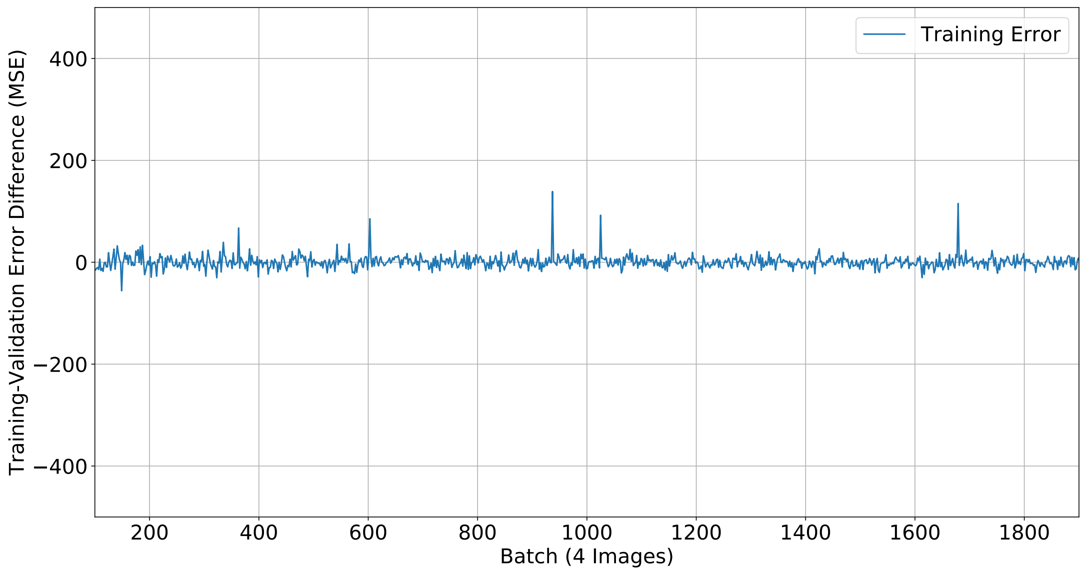

Figure 3.32: Per batch difference in training error and validation for a training batch size of 4, showing no obvious signs of over-fitting or under-fitting. Clear improvement in modelling compared to Figure 3.14, where there are no major signs of under-fitting after 100 epochs.

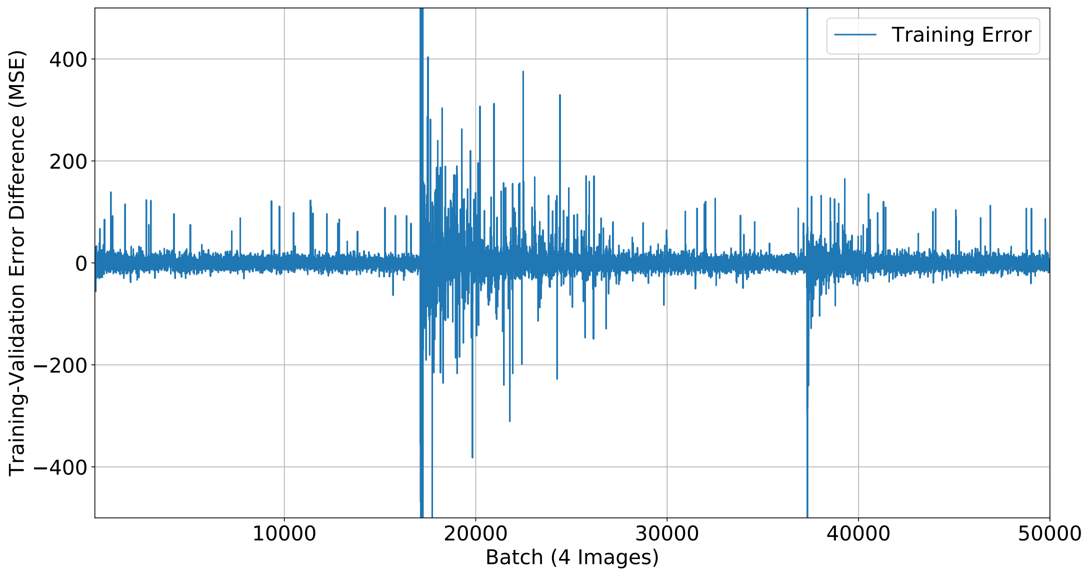

Figure 3.33: Per batch difference in training error and validation for a training batch size of 4 extended to 50000 batches. This Figure demonstrates the inherent instability of training with such a small batch size, where a complex or erroneous training batch sample can lead to great fluctuations in error if a significantly large gradient is back propagated, an effect observed in higher batch sizes, but to a lesser extent with the generalisation provided by a greater number of samples.

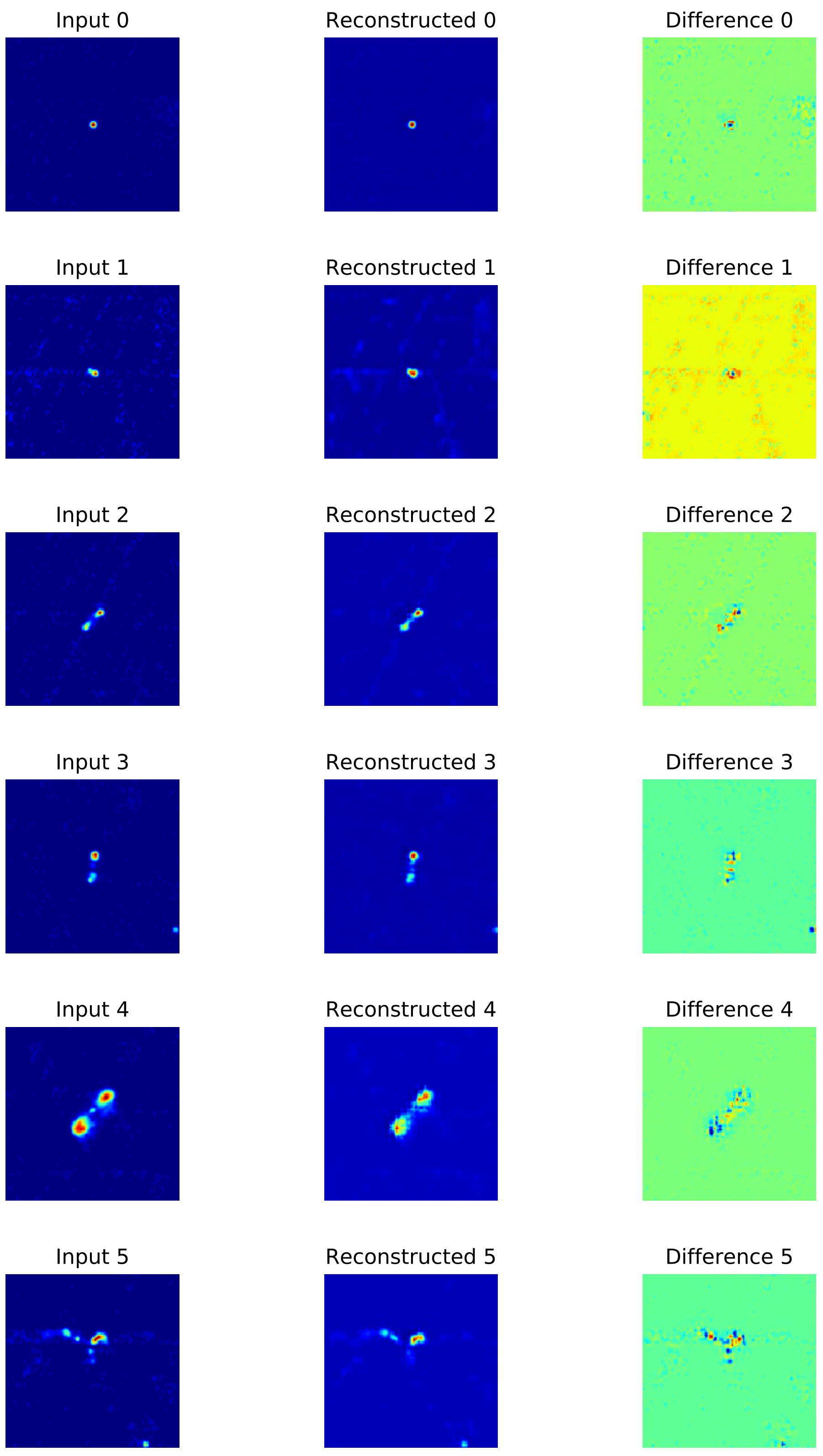

Figure 3.34: Sample input images and corresponding reconstruction and difference image outputs of an autoencoder with a training batch size of 2. The reconstructions show sound quality with preservation of all peaks and most components.

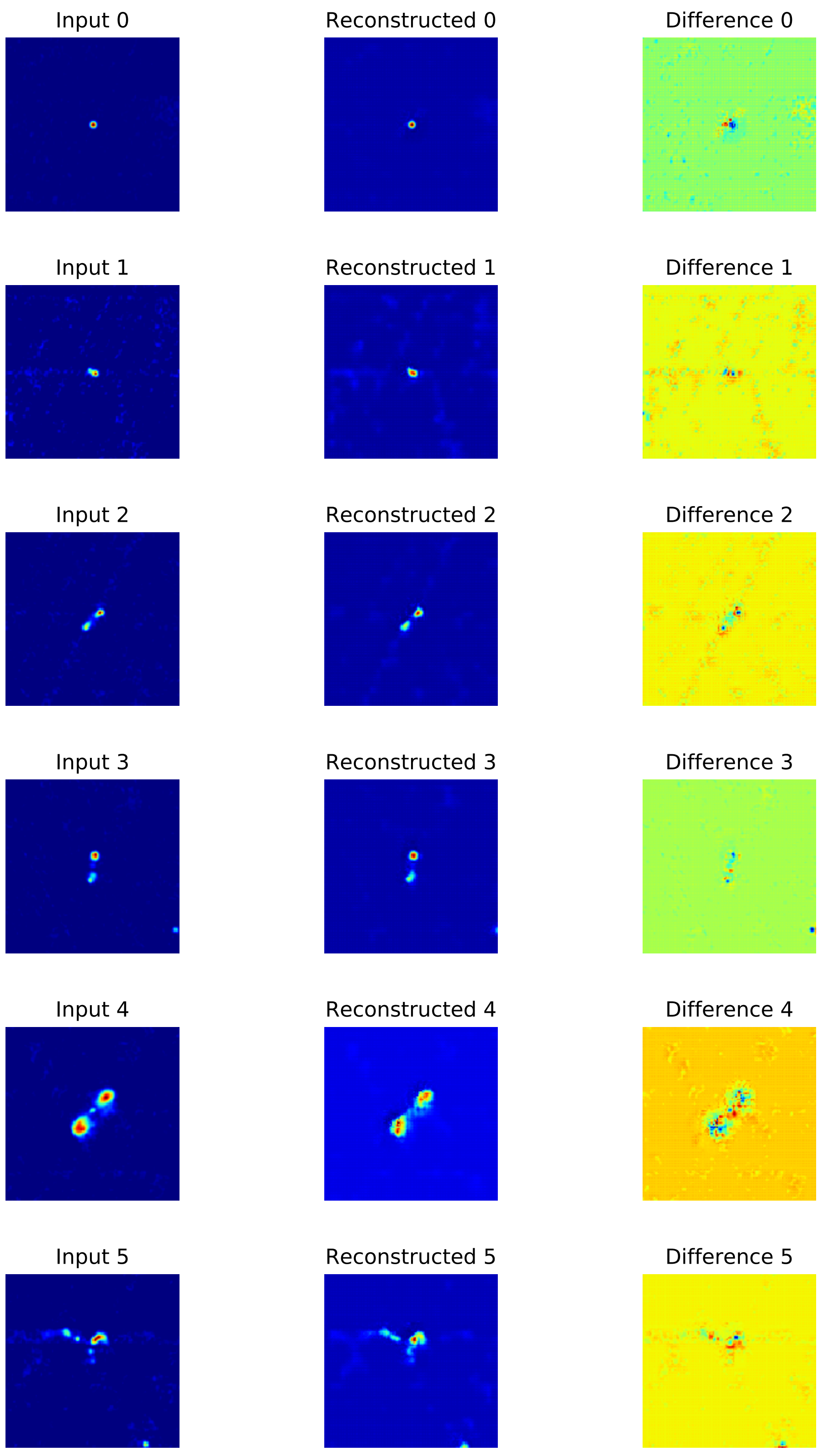

Figure 3.35: Sample input images and corresponding reconstruction and difference image outputs of an autoencoder with a training batch size of 4. The reconstructions are highly accurate with slight improvement in resolution compared to training with the batch size of 2 (Figure 3.34), where all components and peaks are preserved with sound regeneration of the background noise.

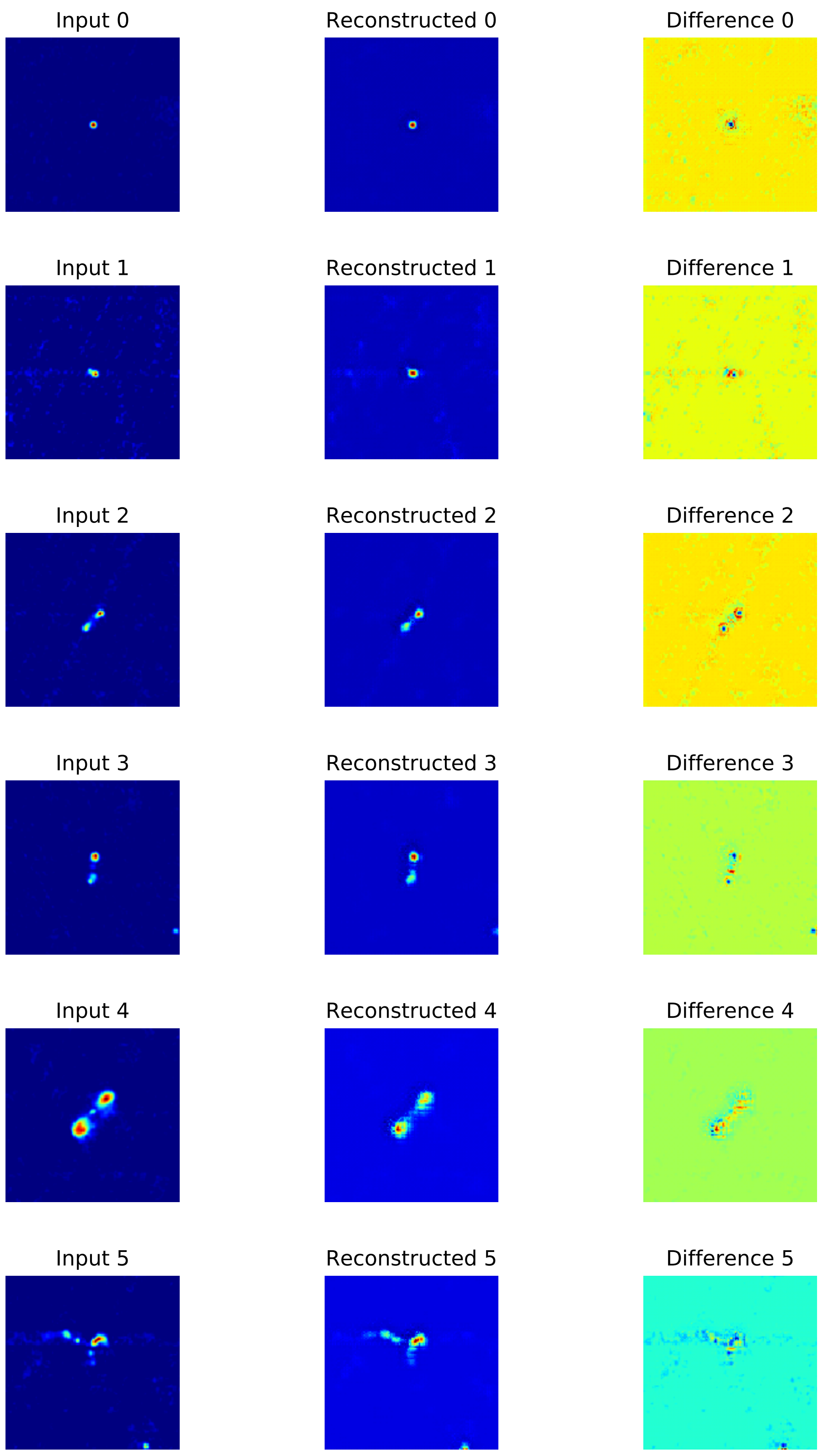

Figure 3.36: Sample input images and corresponding reconstruction and difference image outputs of an autoencoder with a a training batch size of 64. The reconstructions here are again highly accurate, but with lower quality than the previous 2 and 4 filter test (Figure 3.34 and 3.35) with lower resolution, minor blurring and loss of the central component in Reconstruction Image 4.

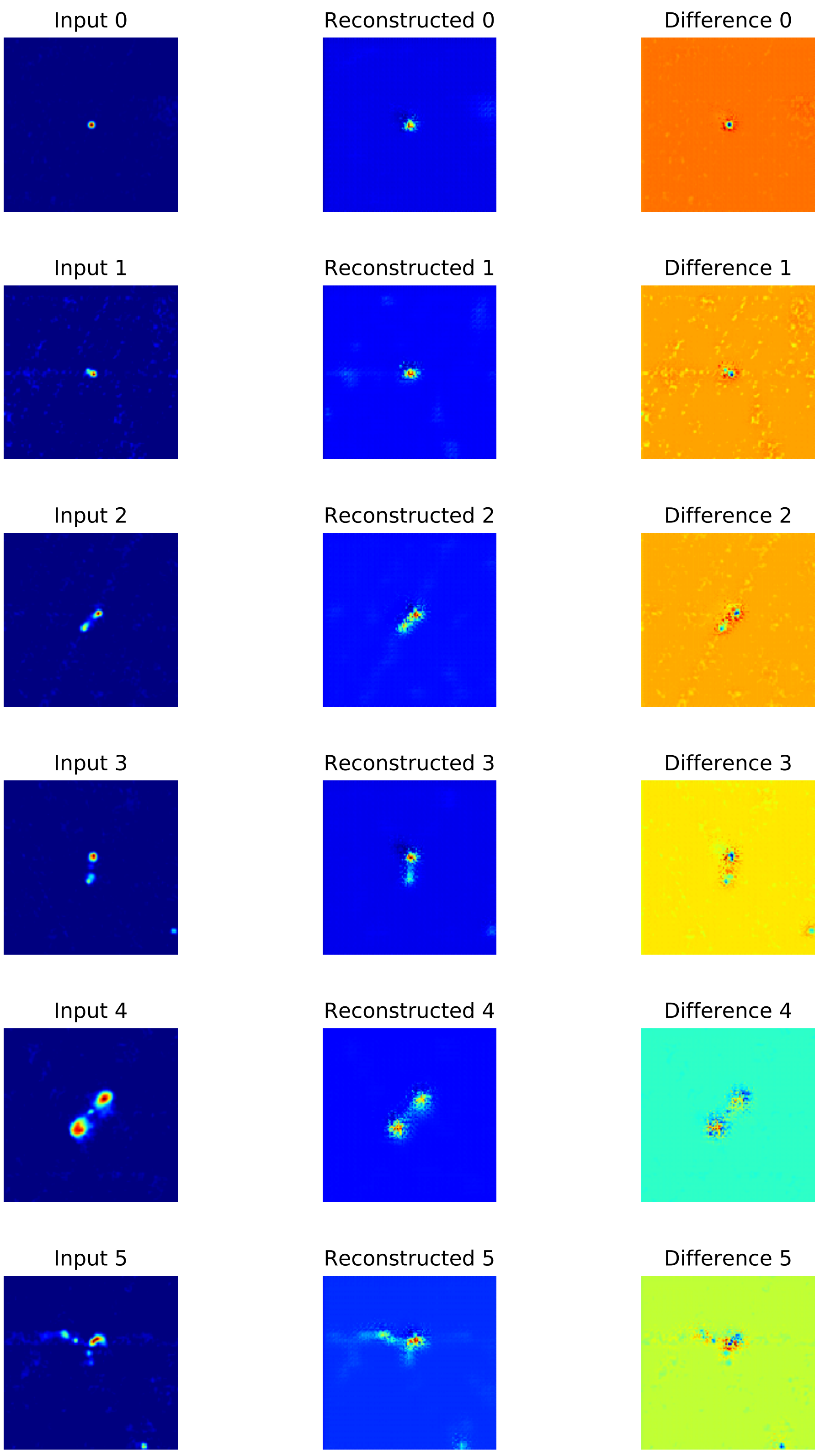

Figure 3.37: Sample input images and corresponding reconstruction and difference image outputs of an autoencoder with a training batch size of 256. The reconstructions here are low resolution and with moderate blurring compared to all previous batch size tests (Figure 3.34 - 3.36).

3.2.2.5 Affine and Noise Training Set Augmentations

The results of the affine and noise injection augmentation testing outlined in Sections 2.4.4.7 are illustrated in Figures 3.38-3.46 and Table 3.6. These tests indicate that random rotations and corruption with impulse noise injection in training images provide a slight increase in accuracy and stability with a minor trade-off in training time. As in previous tests, the network configurations and hyper-parameters I used in this trial are the optimal parameters found in previous tests as 5×5 and 2×2 convolutional and max-pooling filters respectively (Sections 3.2.2.1 - 3.2.2.2) with 32 convolutional filters in the middle hidden layer of the encoder and decoder (Section 3.2.2.3) and a training batch size of 4 RGZ images 3.2.2.4.

Summarised in Table 3.6, the tests indicate that the addition of corruption using noise injection to provides a minor improvement in accuracy and stability but requires more time to converge. Random rotations in the training set provide the greatest stability and the least error at 5.62 after 634.95 seconds. It is noted that the corruption test does reach a minimum quickly at epoch three but the lowest at the end of epoch 9, which increases this convergence time, but still with greater error than the random rotation tests. It is clear in these Figures that the rotation and corruption augmentation methods are not complementary, as combining noise injection with the rotations degrades the accuracy and stability of the rotation augmentations. Figures 3.40 and 3.40 demonstrates the slight accuracy improvement provided by the noise injection and the rotation augmentations.

Figure 3.45 indicates that autoencoder reconstructions of RGZ images with the rotation training augmentations are the most accurate with nearly all peaks and components restored. The central component of the AGN in Reconstructed Image 4 is very faint, with low-intensity emission produced in its wake. Although the error between previous and subsequent reconstruction images (Figures 3.46 and 3.44) is low, this image clearly contains the most fidelity and the greatest resolution. Reconstructions with noise injection are acceptable, but with clearly lower quality than the previous augmentation tests (Figures 3.46-3.45). Many of these reconstructed images show the merging of multiple radio peaks and components. Additionally, there is a degree of granularity added to Reconstructed Image 4, which is also missing the central component of the AGN. Finally, the combination of corruption and rotation in Figure 3.44 show reconstructions are also reasonable, most peaks and components reconstructed but with distortion of Reconstructed Image 4 and the creation of an additional peak in Reconstructed Image 5. Reconstruction of the background noise in these images appears to be highly accurate. Although this combination produces a higher error than both the corruption and rotation augmentations, many of the artefacts produced by the corruption augmentation are not present.

As in previous tests, Figures 3.42-3.43 demonstrates training with either rotations or noise injection does not have an adverse effect on fitting quality, with no consistent variations between the two error metrics, indicating no apparent signs of under-fitting or over-fitting. Compared to no augmentations (Figure 3.32), however, the differences between training and validation error are marginally more erratic with the dataset augmentations.

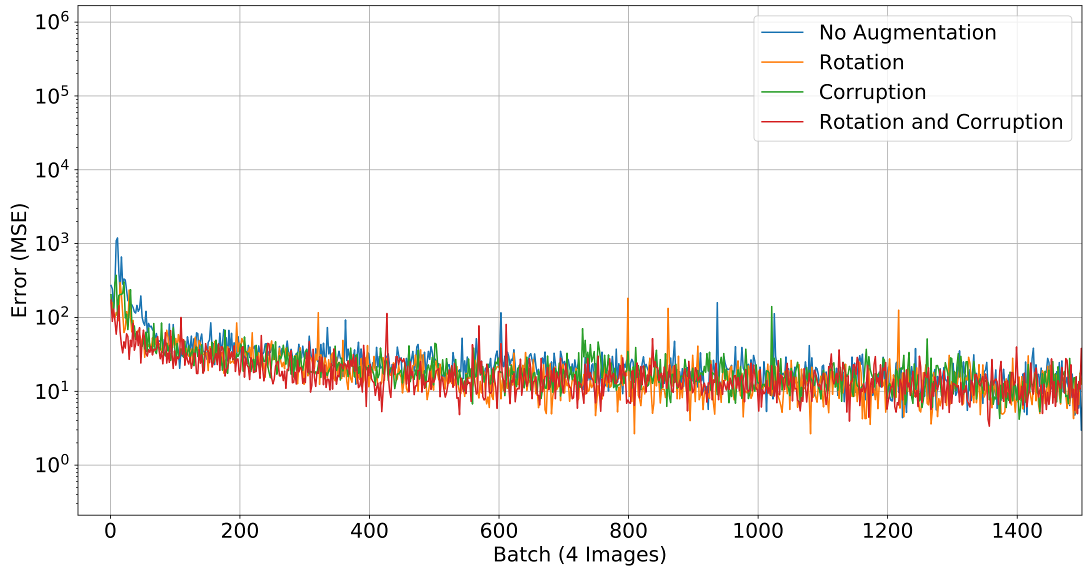

Figure 3.38: Comparisons of the validation MSE of the autoencoder across the training set augmentation tests. This Figure shows little difference in error and convergence without augmentations and between different dataset augmentations.

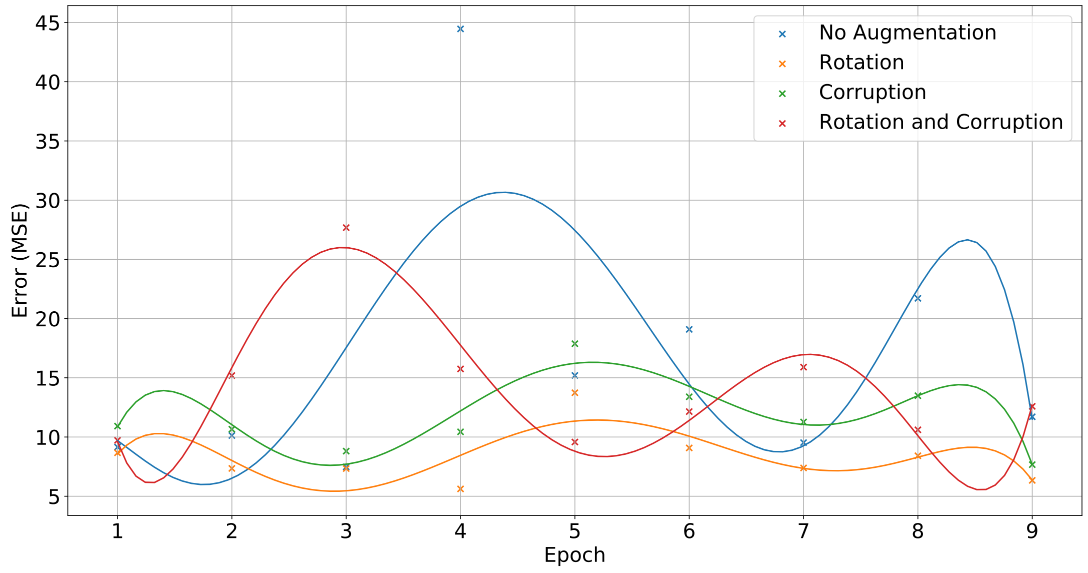

Figure 3.39: Scatter plot and polynomial fit of the average validation error in the final 10% of batches per epoch for all augmentation tests. This Figure confirms there is little error change across the dataset augmentations, but the rotation augmentations provide the most accuracy and improved stability.

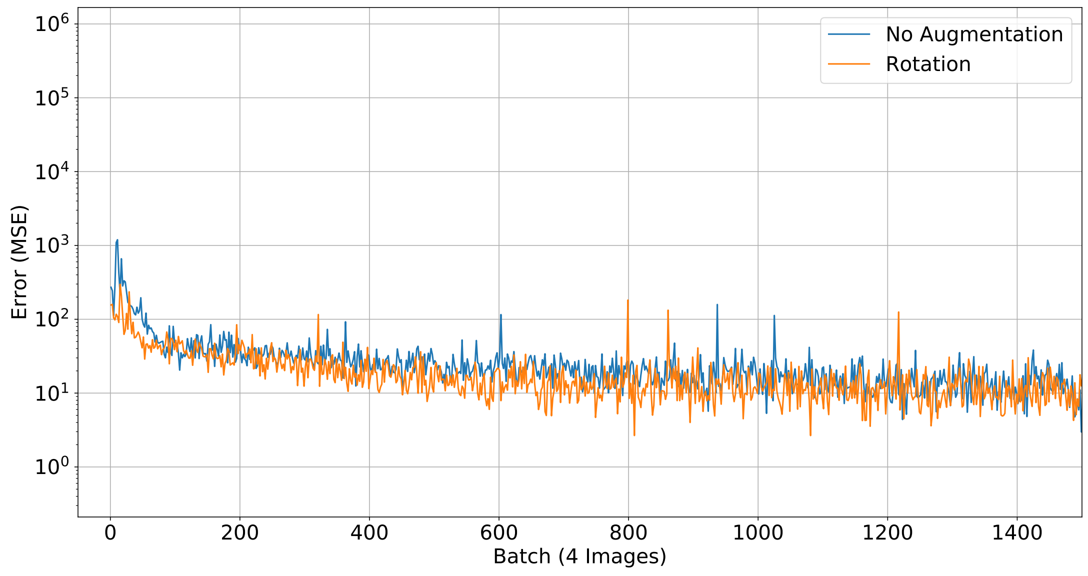

Figure 3.40: Comparisons of the validation MSE of the autoencoder between rotation augmentations and unaugmented training set tests. This Figure demonstrates shows the slight accuracy improvement provided by the rotation augmentations.

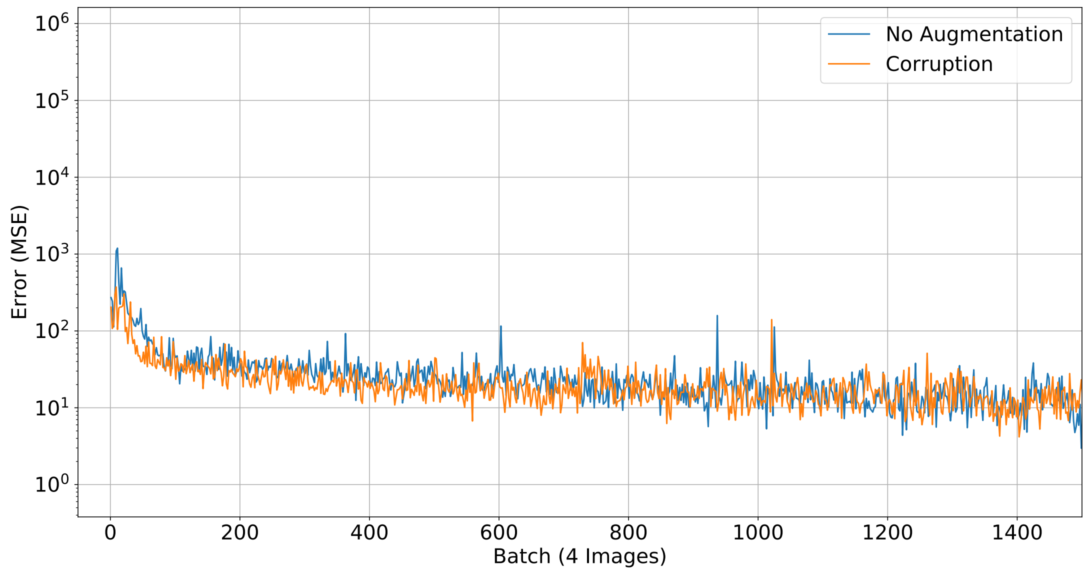

Figure 3.41: Comparisons of the validation MSE of the autoencoder between noise injection augmentations and unaugmented training set tests. This Figure demonstrates the slight accuracy improvement provided by the noise injection, similar to that of the rotation augmentations of Figure 3.40.

Table 3.6: Summary of the training set augmentation statistics, indicating that the inclusion of random rotations is the most accurate approach to training with a slight increase in training time.

Dataset Augmentations	Convergent Epoch	Convergent Epoch Error (MSE)	Training Time to Convergence (s)
No Augmentation	**3.0**	7.48	**528.34**
Rotation	4.0	**5.62**	634.95
Corruption	9.0	7.67	1773.74
Rotation and Corruption	5.0	9.58	1193.08

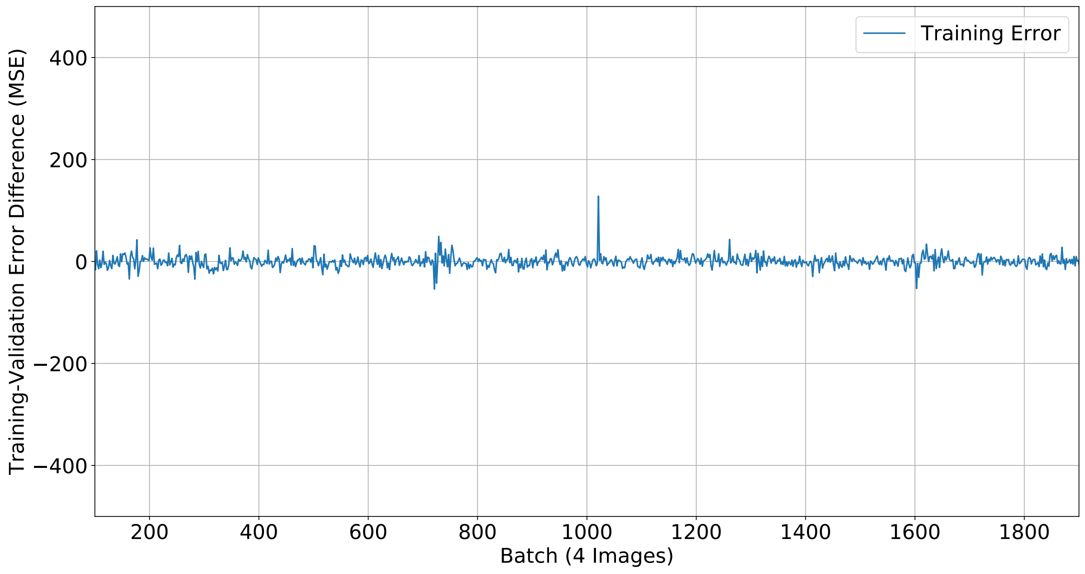

Figure 3.42: Per batch difference in training error and validation for the noise injection training set augmentation tests, showing no obvious signs of over-fitting or under-fitting. A slightly higher variation can be seen here compared to no augmentations 3.32.

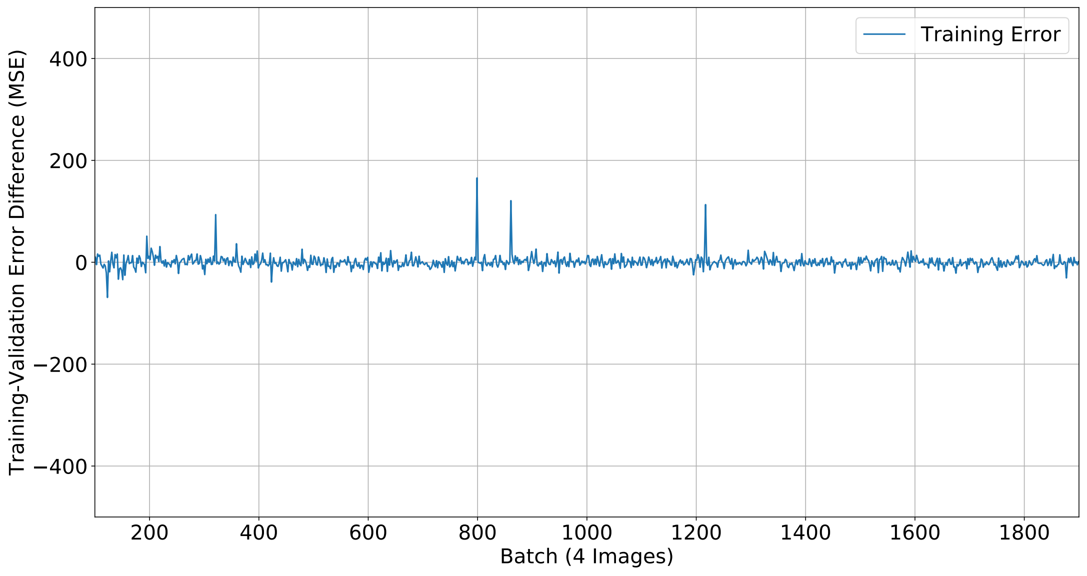

Figure 3.43: Per batch difference in training error and validation for the random rotation training set augmentation tests, showing no obvious signs of over-fitting or under-fitting. A slightly higher variation can be seen here compared to no augmentations 3.32, still not of consequence. No major change can be observed when compared to training with noise injection (Figure 3.42).

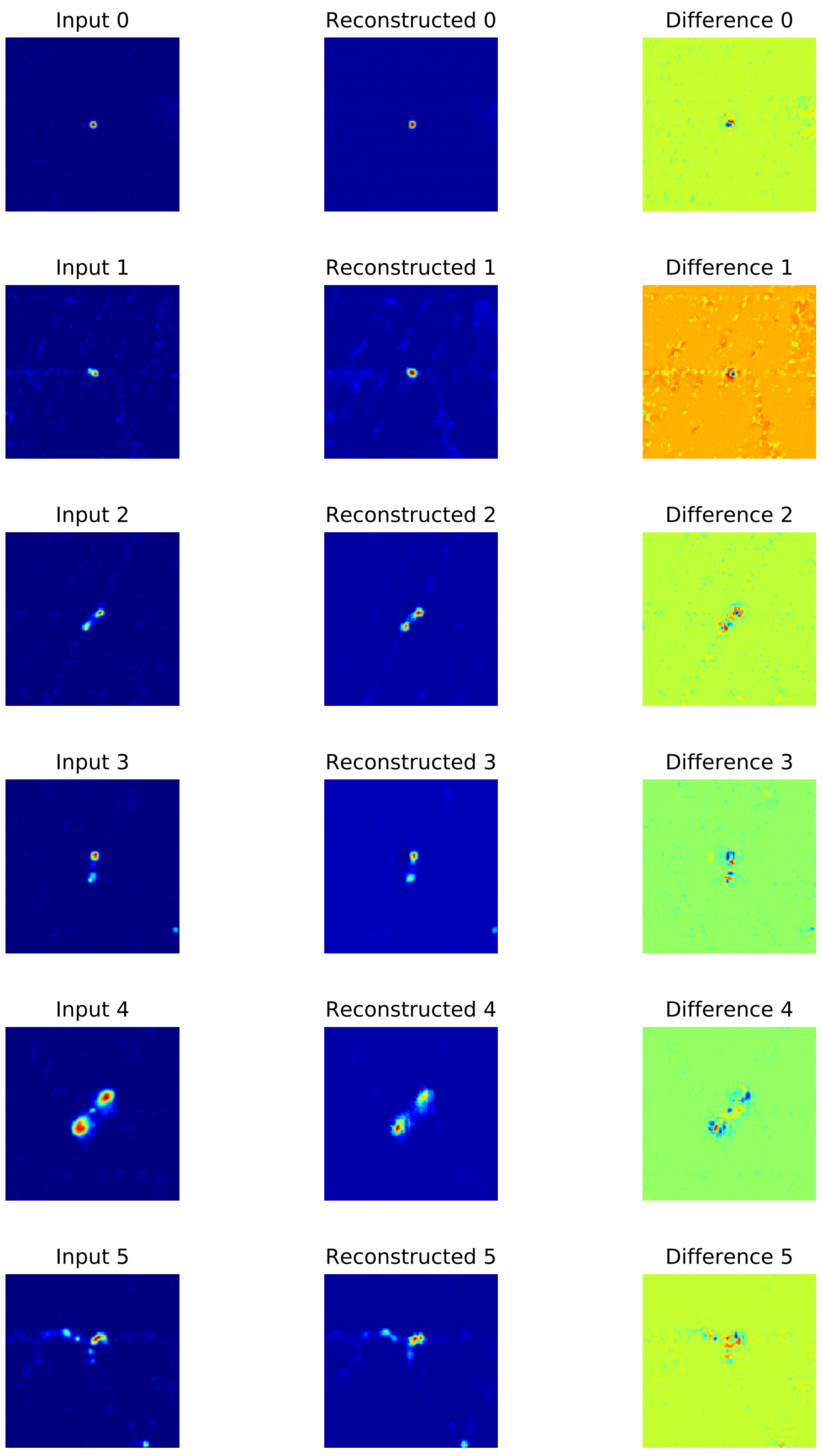

Figure 3.44: Sample input images and corresponding reconstruction and difference image outputs of an autoencoder with corruption training set augmentation. The reconstructions here are accurate with most peaks and components reconstructed but with distortion of Reconstructed Image 4 and the creation of an additional peak in Reconstructed Image 5. Reconstruction of the background noise in these images appears to be highly accurate.

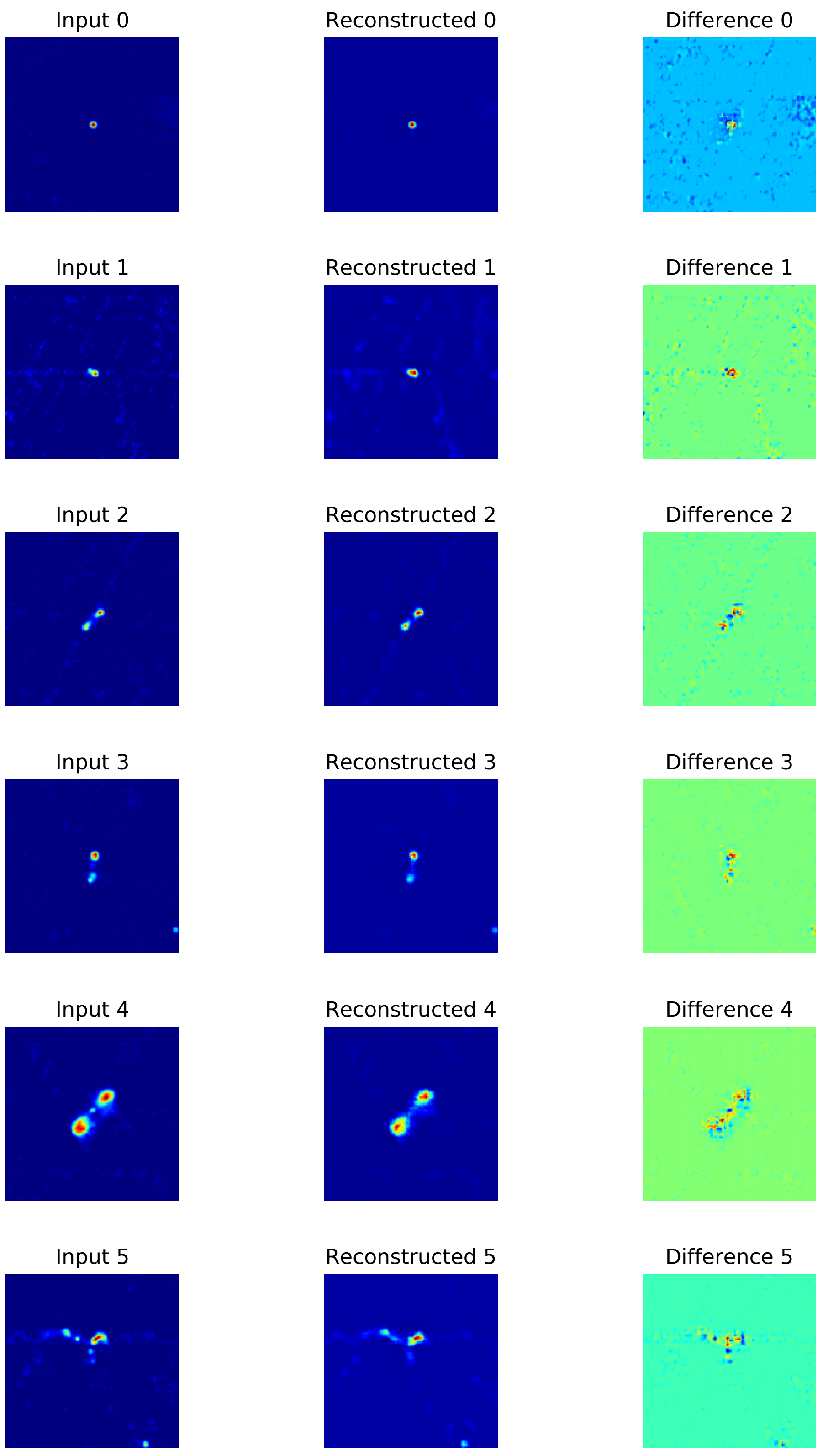

Figure 3.45: Sample input images and corresponding reconstruction and difference image outputs of an autoencoder with rotation training set augmentation. The reconstructions are highly accurate with nearly all peaks and components restored. The central component of the AGN in Reconstructed Image 4 is very faint, with only low intensity emission. Although the error between previous and subsequent reconstruction images (Figures 3.46 and 3.44) is low, this image clearly contains the most fidelity and the greatest resolution.

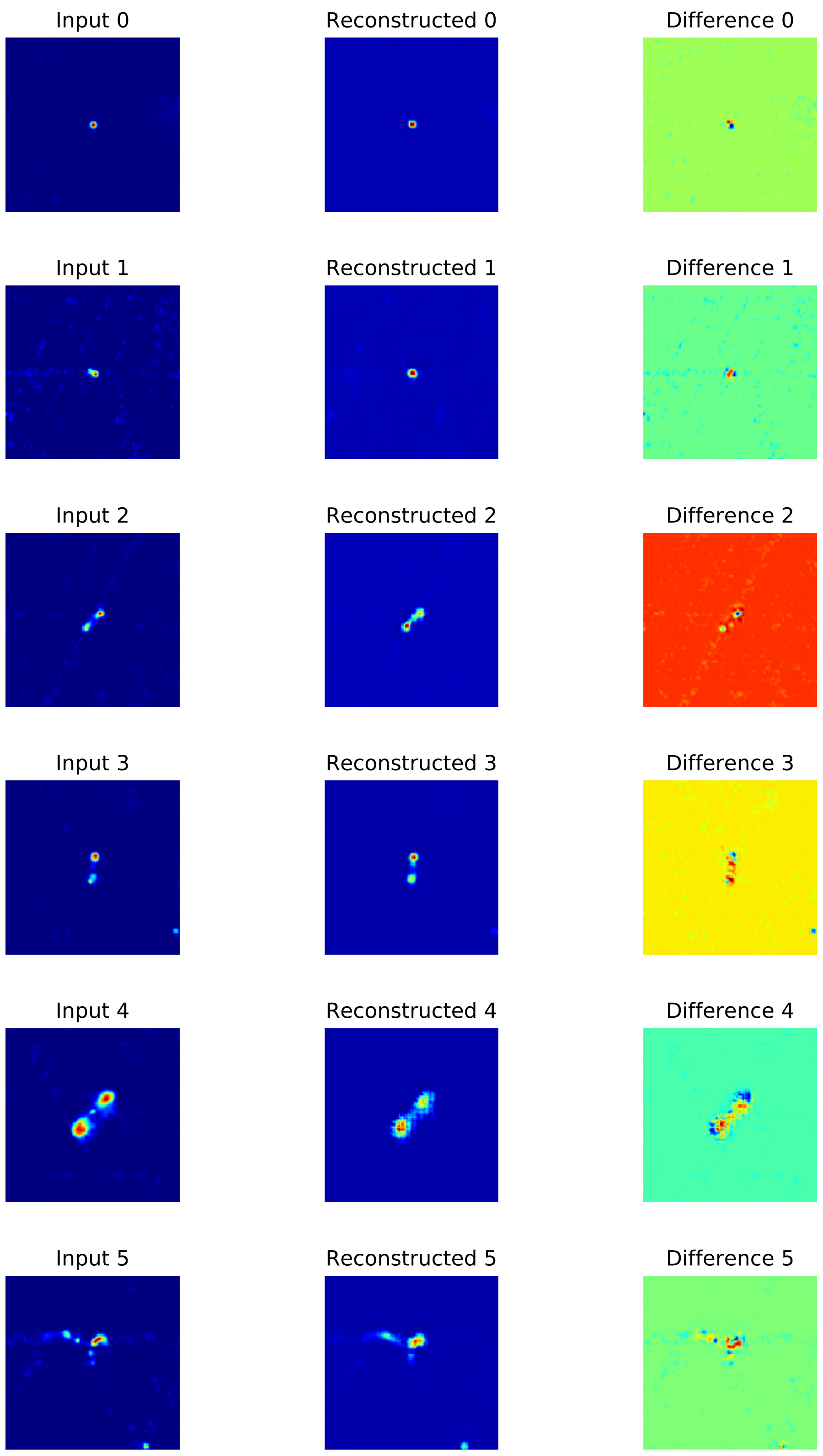

Figure 3.46: Sample input images and corresponding reconstruction and difference image outputs of an autoencoder with a combination of corruption and random rotation training set augmentation. The reconstructions here are acceptable, but with clearly lower quality than the previous augmentation tests (Figures 3.46-3.45), with the merging of multiple peaks and components across the reconstructed images and a degree of granularity added to Reconstructed Image 4 with no central component.

3.2.3 Overview and Final Autoencoder Training Hyper-Parameters and Configurations

The final autoencoder training hyper-parameters and architecture configuration is summarised in Table 3.7 and illustrated in Figure 3.47. These parameters are the result of the tests outlined in Section 2.4.5, shown in Sections 3.2.1 - 3.2.2.5.

Table 3.8 shows the gradual performance improvement and training time improvement in each optimisation step, with the final test error of the best-chosen configuration. In this Table, the final ideal set of training hyper-parameters and network configurations with no training augmentations, produces a MSE of 7.48, compared to the preliminary proof of concept network MSE $\sim$ 700. These results show the optimised autoencoder produces significantly better reconstructions of radio-astronomical image features with a MSE approximately 25 times lower than the preliminary network error. However, the final network does require additional training time, where the original configuration was trained in 2.2 minutes, and the optimised configuration was trained in 8.81 minutes. No apparent signs of under-fitting or over-fitting are observed in training after the first epoch throughout all trials. Changes in validation time (effectively encoding time) are negligible, with only the combined rotation and noise injection providing a slightly more noticeable time increase given the network is performing two augmentations processes on input images.

Testing autoencoder training augmentations confirmed that autoencoder denoising with image corruption and random rotations could improve autoencoder accuracy at the cost of training time. Tests showed random rotation augmentations provide the best performance out of the augmentations, with a MSE of 5.62 after 634.95 seconds. These tests also demonstrated that methods are not complimentary, where the combination of these two augmentations methods produce no reduction in error over the individual augmentation methods.

The final un-augmented autoencoder network can encode an image on average, in 0.008425 of a second, as shown in Table 3.8 with the results of the batch size tests. This encoding time indicates that on average, the time to encode 30,000 RGZ images for SOM training or validation is 252.75 seconds or 4.2125 minutes. While the random rotation augmentation results in a total encoding time of 440.25 seconds or 7.3375 minutes at an average of 0.014675 seconds per image.

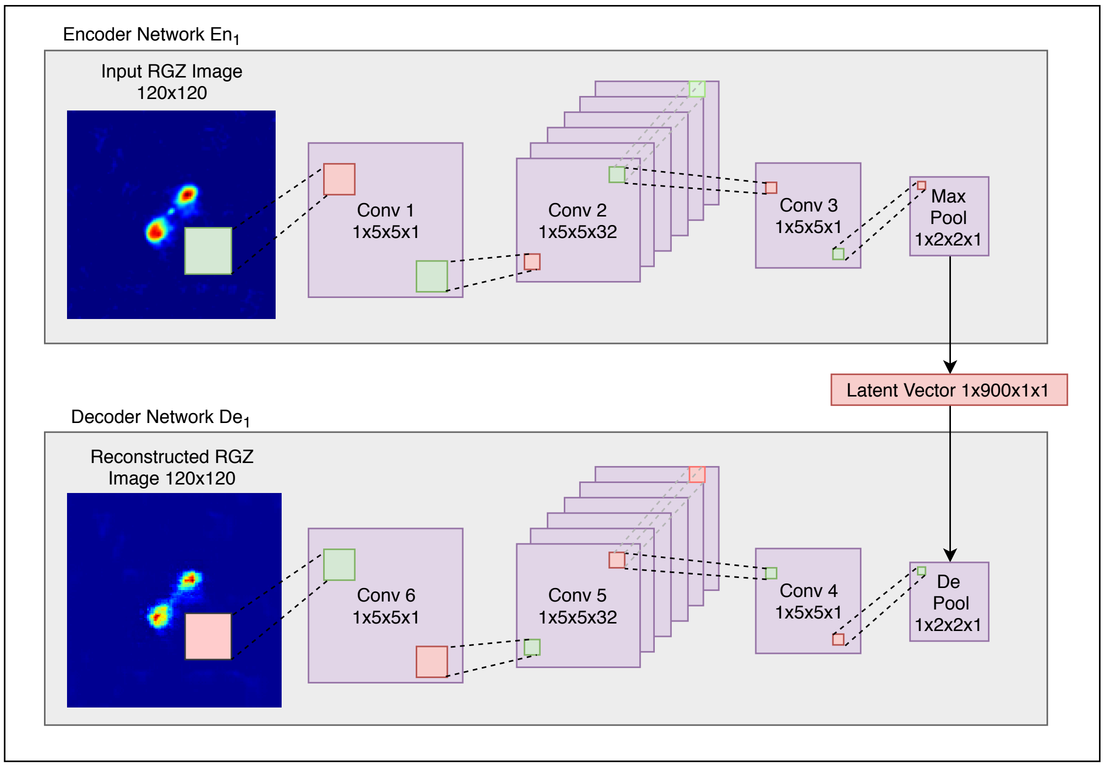

Figure 3.47: The final convolutional autoencoder architecture based on the results of the network configuration and hyper-parameter optimisation testing as summarised in Table 3.7. This schematic demonstrates the role of each layer in gradually compressing the input 120×120 image from the encoder network to the single channel per batch 900×1 latent vector, which is restored to the original dimensions by the decoder to interpret the latent vector and calculate network performance. The filter size and count notation is given as $S_{batch}, S_x, S_y, S_{channel}$, as described in Section 2.4.4.1.

Table 3.7: Outline of final optimised autoencoder architecture and layer configuration, where all convolutional layers use LReLU activation functions.

Network Section	Layer	Function	Input	Filter Size	Stride
Encoder	0	Input	$120 \times 120 \times 1$	-	-
	1	Convolution 1	$120 \times 120 \times 1$	$5 \times 5 \times 1$	$1 \times 2 \times 2 \times 1$
	2	Convolution 2	$60 \times 60 \times 1$	$5 \times 5 \times 32$	$1 \times 2 \times 2 \times 1$
	3	Convolution 3	$30 \times 30 \times 1$	$5 \times 5 \times 1$	$1 \times 2 \times 2 \times 1$
	4	Max-Pool 1	$30 \times 30 \times 1$	$3 \times 3 \times 1$	$1 \times 1 \times 1 \times 1$
Centre	5	Latent Vector	$30 \times 30 \times 1$	-	-
Decoder	6	De-Pool 1	$30 \times 30 \times 1$	$3 \times 3 \times 1$	$1 \times 1 \times 1 \times 1$
	7	Convolution 4	$30 \times 30 \times 1$	$5 \times 5 \times 1$	$1 \times 2 \times 2 \times 1$
	8	Convolution 5	$60 \times 60 \times 1$	$5 \times 5 \times 32$	$1 \times 2 \times 2 \times 1$
	9	Convolution 6	$120 \times 120 \times 1$	$5 \times 5 \times 1$	$1 \times 2 \times 2 \times 1$
	10	Output	$120 \times 120 \times 1$	-	-

Table 3.8: Outline of final optimised autoencoder architecture and layer configuration test results with the final test MSE for the optimised configuration, total training time and validation or encoding time per image.

Optimised Test	Final Test Error (MSE)	Training Time (s)	Average Validation and Encoding Time per Image (s)
Convolutional Filter Size (5×5)	488.55	954.67	0.0082
Max-Pooling Filter Size (2×2)	292.23	1473.98	0.0094
Convolutional Filter Count (32)	245.36	896.85	0.0077
Batch Size (4)	7.48	**528.34**	0.0084
Random Rotation Training	**5.62**	634.95	0.0147
Noise Injection Training	7.67	1773.74	0.0093
Noise and Rotation Training	9.58	1193.08	0.0155

3.3 Self-Organising Map Results

3.3.1 Preliminary Results

The SOM I implemented for this thesis and featured in Ralph et al. (2018) was developed using the method outlined in Section 1.2.2.2. The results of this Section form the preliminary results of my novel approach to training a SOM on latent vectors. Similar to the autoencoder results of Section 3.2.1, these tests were preliminary and obtained using a method presented only as a proof of concept in Ralph et al. (2018) with only basic optimisation. The final results using the optimised architecture are presented in the remainder of Section 3.2.

A 20×20 neuron toroidal umat is displayed in Figure 3.48. Using my system, this map was created in 40 seconds after 100 training epochs on the 10,000 encoded RGZ images for an average of 0.36 seconds per epoch. In this image, I superimpose the first closest matching RGZ image of each neuron to visualise the dataset topology, and latent relationships learned from the latent vectors (detailed in Section 2.5.2). This closest matching image represents the best matching image in the dataset to the learned weights with subsequent candidates decreasing in similarity (as Euclidean distance).

This umat clearly shows the morphology distribution of RGZ images where the Euclidean distance between the learned weight of each neuron and that of their immediate neighbouring neurons displayed as a heat map. Images placed in high distance regions have a latent feature vector with a high Euclidean distance to surrounding neighbours. This is confirmed with high distance regions highlighting outliers within the RGZ image set, particularly in the case of the upper left regions of the umat with sources showing complex morphology.

The morphological clusters in this map are continuous, with neurons essentially representing a probability distribution of latent feature vectors. These clustered regions are sub-clustered by orientation, with similarly oriented objects clustered together with gradual transitions between classes. I expect to see this gradual transition between classes of images given these objects do not have discrete features, but instead have a continuous unresolved morphology. The central low distance region of the umat contains closest matching images as single compact sources. Matching sources in this central compact region gradually progress in complexity to compact multi-point sources toward outer edges. Matching sources on the edges of the map in high distance regions show the highest complexity. These observations are confirmed by examining the RGZ labels and entropy of each matching source.

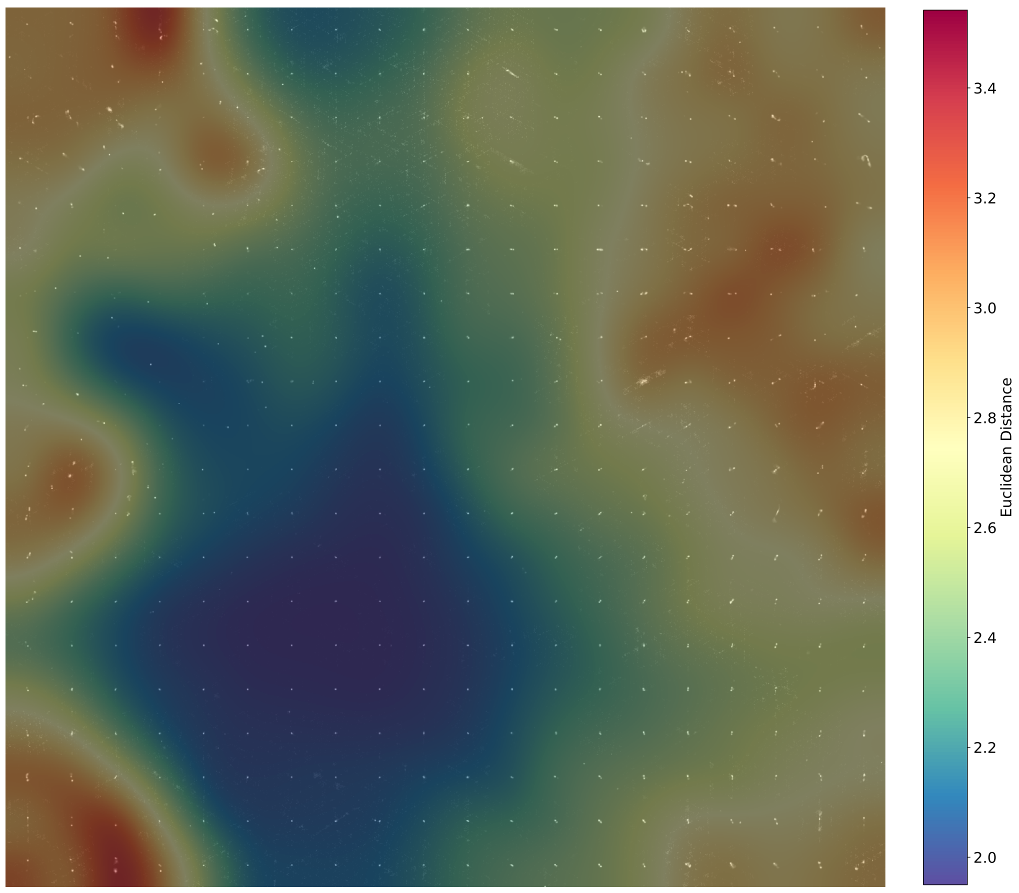

Figure 3.48: Toroidal umat as SOM output with transparent greyscale closest matching images over the heat map of the Euclidean distance between neuron weights. This map was created in an average of 0.4 seconds per epoch for 40 seconds after 100 epochs. Sources are softly clustered with a smooth transition between classes. The central low distance regions contain mostly compact single sources. Progressing to the outer high distance edges are sources with gradually increasing complexity. Outliers and complex sources reside in high distance regions while more common point sources remain in low distance areas.

3.3.2 Hierarchical Clustered Self Organised Map

Outlined in Section 2.5.5, I clustered the SOM weights to cluster neurons for complexity separation. In Ralph et al. (2018), I used three the K-means algorithm with three K clusters, 0, 1 and 2. In Figure 3.49, these are labelled in the centre of each neuron. These labels are colour coded cyan for cluster 0, red for cluster 1 and white for cluster 2. Table 3.10 details the proportion of neurons classified into each cluster with the minimum (min), maximum (max) and mean entropy (discussed in Section 2.5.6.1). Tables 3.11 and 3.12 list the proportion of first best matching units for each label across the clusters. These tables detail the overall class population of the cluster and the overall proportion of each class's dataset population in the cluster. Additionally, the min, max and mean entropy for each clustered label set is listed. From

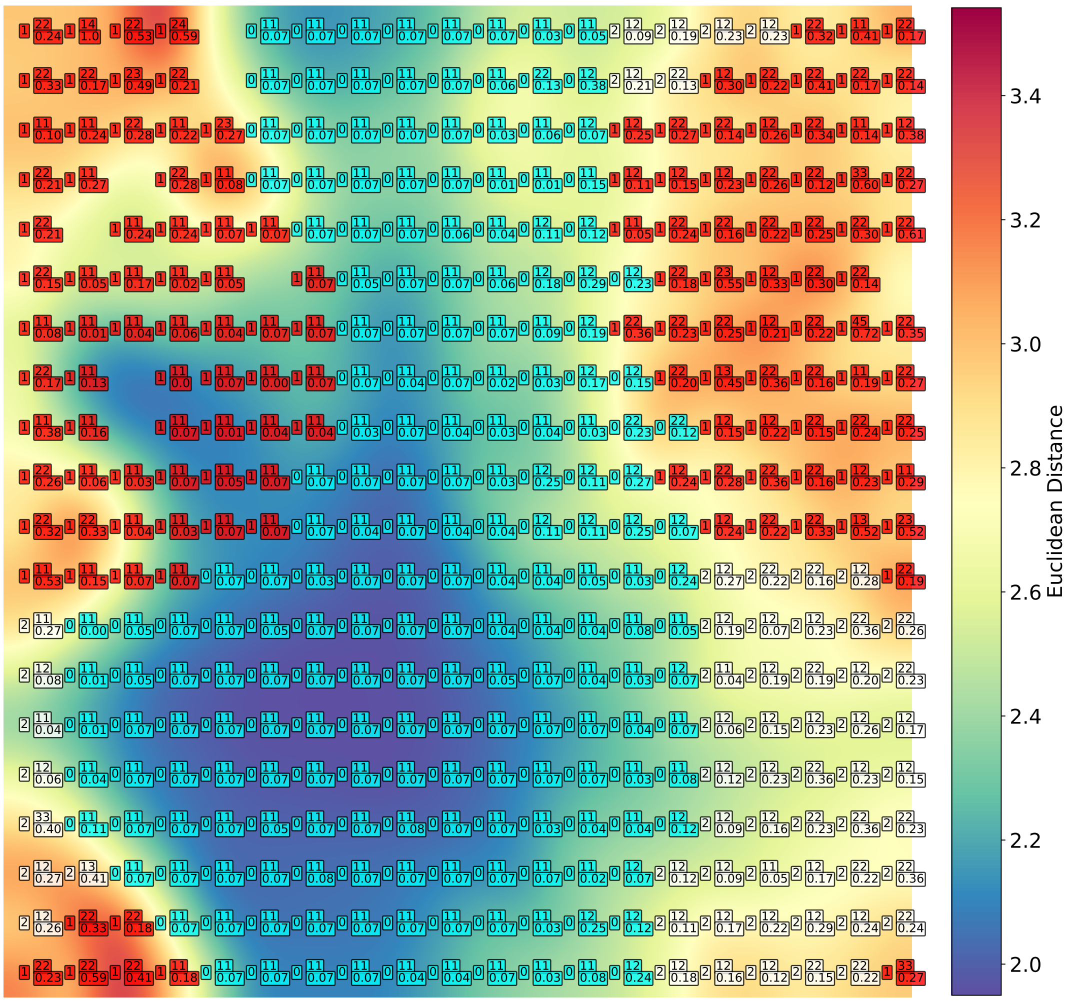

Figure 3.49: The same SOM as shown in Figure 3.48 with labels for class (0,1,2) colour coded, cyan, red and white respectively. Each neuron is labelled with three numbers, giving the properties of the best matching unit for that neurons. The first (0,1,2) shows the cluster number, the next is a two-digit number representing the labels, and the third is a floating-point number giving the entropy..

these tables and figures, it is clear these classes segment the SOM into three groups of simple, compact multi-feature and highly complex images.

Cluster 1 is located in the low distance centre of the umat, dominated by simple point sources with 79.13% of all neurons as single peak point sources. This cluster contains 78.37% of all class 11 in the dataset with very low entropy. As detailed in Section 2.2, my label encoding is components-peaks, where 11 is a source with a RGZ label of 1 peak and 1 component. The remaining 15.05% of neurons are highly compact 12 point sources with two peaks. Two 33 and 22 sources are likely caught in this cluster due to highly compact morphology. The high proportion of these 11 class sources and highly compact sources indicate that cluster 1 is effective in clustering simple objects.

Cluster 2, however, contains a mix of sparse and highly complex sources located mostly in the high distance regions of the umat. This cluster is highly interesting and contains 70.18% of all class 22 sources, making up 31.75% of the whole cluster (Table

Table 3.9: Proportion of labels expressed as the matching image candidate across the map with min, max and mean entropy.

Categories	Population	Match Population Over Map	Mean Entropy	Max Entropy	Min Entropy
11	63.78%	52.0%	0.44	0.07	1.52
12	13.86%	18.75%	0.97	0.21	2.27
22	14.3%	14.25%	1.30	0.10	2.57
33	3.28%	5.25%	1.36	0.29	2.29
23	1.94%	4.25%	1.39	0.12	2.25
13	0.78%	0.75%	1.67	1.12	2.40
44	0.76%	0.75%	1.02	0.68	1.60
45	0.2%	0.5%	2.60	2.19	3.00
34	0.42%	0.5%	1.92	1.92	1.92
14	0.08%	0.5%	4.32	3.04	5.61
16	0.02%	0.25%	1.66	1.66	1.66
46	0.08%	0.25%	3.32	3.32	3.32
55	0.14%	0.25%	1.28	1.28	1.28
24	0.16%	0.0%	-	-	-
35	0.08%	0.0%	-	-	-
36	0.02%	0.0%	-	-	-
15	0.02%	0.0%	-	-	-
56	0.04%	0.0%	-	-	-
57	0.02%	0.0%	-	-	-
67	0.02%	0.0%	-	-	-

3.13. The other dominant class in this cluster are sparse and complex class 11 sources in the upper left region with faint companions and noise. The remaining sources in this cluster include all of the low population outlier 14, 16, 44, 45, 46 and 55 classes in addition to 52.94% of the 23 sources and a majority of the 33 sources at 71.43%. These populations indicate that cluster 2 effectively segments outliers and sources of high interest.

It is clear that cluster 0 resides between these two clusters on the umat in medium distance regions with matching images containing classes that are not too complex for cluster 1 but not compact or simple enough for cluster 2. This is confirmed by Table 3.12, with highly compact multi-peak and multi-component sources in this cluster. This cluster is comprised 50.0% 12 class sources, 20.59% of 22 sources and a series of 11, 23, 33 and 13 sources. Many of the less dominant sources here are complex enough to remain out of cluster 1 but contain a compact enough morphology to keep them from the sparse and complex cluster 2. These observations indicate that cluster 0 segments medium complexity sources in a manner that allows cluster 1 and 2 to remain dominated by simple and complex sources respectively.

Table 3.10: Proportion of K-means clusters across the map with min, max and mean entropy.

K-means Cluster	Match Population Over Map	Mean Entropy	Max Entropy	Min Entropy
0	17.0%	0.21	2.27	1.15
1	51.5%	0.07	2.18	0.48
2	31.5%	0.10	5.61	1.19

Table 3.11: K-means Cluster 0: Proportion of image matches across the cluster, the overall dataset population of each class contained in the cluster with min, max and mean entropy.

Label	Overall Proportion of Cluster Population	Overall Proportion of Label in Cluster	Mean Entropy	Min Entropy	Max Entropy
12	50.0%	45.33%	1.14	0.36	2.27
22	20.59%	24.56%	1.32	0.72	1.94
11	11.76%	3.85%	0.60	0.21	1.05
23	11.76%	47.06%	1.33	0.24	2.25
33	2.94%	9.52%	1.68	1.30	2.06
13	1.47%	33.33%	1.12	1.12	1.12

Table 3.12: K-means Cluster 1: Proportion of image matches across the cluster, the overall dataset population of each class contained in the cluster with min, max and mean entropy.

Label	Overall Proportion of Cluster Population	Overall Proportion of Label in Cluster	Mean Entropy	Min Entropy	Max Entropy
11	79.13%	78.37%	0.41	0.07	0.75
12	15.05%	41.33%	0.80	0.21	2.18
33	1.94%	19.05%	0.64	0.29	1.28
22	1.46%	5.26%	0.81	0.22	1.27

Table 3.13: K-means Cluster 2: Proportion of image matches across the cluster with min, max and mean entropy.

Label	Overall Proportion of Cluster Population	Overall Proportion of Label in Cluster	Mean Entropy	Min Entropy	Max Entropy
22	31.75%	70.18%	1.33	0.10	2.57
11	29.37%	17.79%	0.55	0.10	1.52
33	11.9%	71.43%	1.52	0.78	2.29
12	7.94%	13.33%	0.94	0.31	1.31
23	7.14%	52.94%	1.44	0.12	2.01
44	2.38%	100.0%	1.02	0.68	1.60
45	1.59%	100.0%	2.60	2.19	3.00
34	1.59%	100.0%	1.92	1.92	1.92
14	1.59%	100.0%	4.32	3.04	5.61
13	1.59%	66.67%	1.94	1.48	2.40
16	0.79%	100.0%	1.66	1.66	1.66
46	0.79%	100.0%	3.32	3.32	3.32
55	0.79%	100.0%	1.28	1.28	1.28

3.3.3 Final Results with Hyper-Parameter and Network Architecture Optimisation

I conducted several tests to locate an optimal combination of initial conditions to maximise SOM modelling performance of the SOM used in Ralph et al. (2018). These tests are outlined in Section 2.5.8 and are similar to the hyper-parameter and architecture optimisation of Section 2.4.4, which include:

1. SOM map size: 5×5, 10×10, 20×20 and 40×40 neurons square.
2. Exponential and linear learning rate decay functions.
3. Gaussian and linear learning neighbourhoods functions with exponential and linear neighbourhood decay functions.
4. Analysis of affine invariance in the SOM trained on latent vectors.
5. 'K' number of K-means clusters: 4,8 and 16 clusters.

Throughout these tests I use a set of initial conditions (detailed in Sections 2.5.1 and 2.5.8), some of which are gradually tuned based on the results of the relevant trial:

1. Toroidal map
2. Training for 12 epochs
3. Neuron weight initialisation using PCA
4. Initial neuron learning rate of 0.01 and a final weight of 0.001
5. Linear learning rate decay function
6. Linear neighbourhood function
7. Linear neighbourhood radius decay function
8. Analysis of affine invariance in the SOM trained on autoencoder latent vectors learned using random rotation augmentations.

In the following Section, I outline the results of these tests to demonstrate the best performing conditions and the overall results of this system. Detailed in Section 2.5.6.1, these results are evaluated by accuracy metrics concerning complexity separation, with the best scoring test in boldface. All tests have a set of initial conditions which are gradually refined, where subsequent tests use the optimal combinations of the previous tests. The final optimised network is used to evaluate the affine invariance of the autoencoder and the HC.

As outlined in Section 2.5.2, I display the SOM umat with neurons showing the closest matching RGZ image latent vector to each learned neuron weight. Using this map, I also display the distribution of Euclidean distances across all neurons for the complexity separation. In these figures, I also show the proportion of each neuron distance bin with colour coding of the four dominant RGZ source labels, 11, 12, 22, and 33, in addition to simple (11's), complex (not 11) and anomalous (peak and or component count greater than 3) labelled sources, as outlined in Section 2.2. As a development over Ralph et al. (2018), I also attempted to decode the neuron weights using the decoder side of the autoencoder network to visualise the relationships each neuron has learned. Additionally, I display these decoded neuron images over the umat to visualise the neighbourhood Euclidean distance of each neuron. In tests with HC, I colour code neurons with the closest matching image and decoded weights based on their K cluster Identification Number (ID).

It should be noted that some SOM large map sizes in the following sections are not easily displayed or interpreted in paper format due to limitations on page and margin size and should be viewed electronically for full resolution. Moreover, decoded neuron weights tended to become quite faint with highly complex morphologies. As a result, all decoded neurons weights have their image intensities 'min-maxed', where I scale the peak pixel intensity to the highest intensity (255 in this case, where I convert them to 8-bit intensities). This results in some decoded neuron images with exceptionally high background noise. Using this process, visualisation is improved without a negative impact on system performance given this is a straightforward operation and matching is applied only to the latent vectors and neuron weight, not the decoded images.

3.3.3.1 Map Size

The results of this Section indicate that the 10 × 10 map is the ideal SOM map size for the visualisation, clustering and anomaly detection of radio-astronomical images using my approach and the RGZ dataset.

I trialled four toroidal square SOM sizes; 5×5, 10×10, 20×20 and 40×40. These maps were trained for 12 epochs using 30,000 RGZ images for training and 30,000 RGZ images for validation, as outlined in Section 2.5.7. The trained maps illustrated in Figures 3.50 - 3.69 show that across all map sizes, the SOM successfully visualises a wide array of latent relationships and features from within the RGZ data. These maps detail the various morphologies found within the radio-astronomical images of the dataset and arrange them on the manifold by their similarity. In these continuous clusters, radio sources can be seen to gradually develop an impressive array of features and structures that evolve over the surface from simple point sources to highly complex multi-peak, multi-components sources.

Starting from the smallest SOM, the 5 × 5 map in Figures 3.50 - 3.52, contain only very basic morphologies. On the decoded neuron image and decoded neuron umat, point sources are positioned in low Euclidean distance areas from neurons in the lower left. The decoded neurons weights and matching images that gradually progress to higher Euclidean distance regions toward the upper and lower extremes of the right side of the map with more complex morphology. As the SOM map sizes increase, the transition in neuron morphology and distance source types becomes more gradual. As in the preliminary results, clustering of the radio source morphology in this map is continuous. The radio source morphology in these SOMs appears to be sub-clustered by orientation, with similarly oriented objects clustered together with gradual transitions between classes.

The decoded neuron images demonstrate an interesting phenomenon not observed in the preliminary results. In these decoded neuron weight maps, the learned latent vectors of the RGZ dataset display a gradually expanding morphology from a circularly distributed set of neuron features, where the centre is a neuron that appears as a rotated average of a central source radio image and extended emission.

SOM umat images with matching RGZ images on each neuron show similar but more ordered matches than the preliminary tests, where the morphology distribution of the radio-astronomical images where the Euclidean distance between the learned weight of each neuron and their immediate neighbouring neurons displayed as a heat map. Images placed in high distance regions have a latent feature vector with a high Euclidean distance to surrounding neighbours, which indicates relative anomalousness and complexity. Simple point sources are gathered in low distance regions, which gradually develop additional peaks and components as neurons progress towards higher distance regions. As in the preliminary tests, these high distance regions highlight the highly complex structure and outliers within the RGZ image set.

Seen in many figures such as Figure 3.62, the matching RGZ images on the umat show the high Euclidean distance areas that are matched to complex sources such as large radio triple sources, AGN and novel bent-tail galaxies (neuron 1,13). More specifically, these objects match to neurons which show high intensity and globular type structure. This is most pronounced in larger maps, where high distance regions show many of these interesting sources. In all tests, I show the distribution of all RGZ validation images matching neuron Euclidean distance. I label these by the RGZ labels and by a simple-complex classification. These figures show a clear correlation between the high distance areas and complexity. As sources increase in complexity, so do does the mean neuron Euclidean distance. I used this correlation to create the complexity separation accuracy, which flags images by standard deviations 1-5, which I term SD. These accuracy metrics accompany all SOM tests.

As the map size increases the number of neurons, more neuron Euclidean distances are displayed. With these larger map sizes more gradual transitions between radio-image morphologies and neuron Euclidean distances show point sources and less complex morphologies accumulating in lower Euclidean distance areas. In these cases, point sources reside in a broad peak with low Euclidean distance and anomalous sources spreading further into the higher distance range, as shown in Figures and for the 40×40 test. Across all of these distributions, there is a distinct drop in the number of complex sources that varies across each map size. This effect can be seen even without labelling, with a slight second peak forms after the initial main peak rapidly decreases.

Concerning accuracy in complexity separation, the most accurate map using the initial conditions is the 40×40 map with 81.4% and the highest F1 score of 0.779. High F1 score in the 40×40 map size appears to be at the cost of recall at 0.873, which is lower than previous map sizes but a higher precision. These metrics indicate the 40×40 map size complexity separation produces more true positives but slightly more false negatives. The best performing SD complexity separation point on the Euclidean distance distribution map appears to be inversely proportional to the map size, where the most accurate SD, draws closer to the point of zero variance.

The training time for all SOM increase proportionally to the map size. In these tests, the fastest performing SOM is the 5×5 with a total time of 3.475 seconds and the slowest as the 40×40 with a total time of 22.424 seconds. These times are very fast, with less than a minute even for the largest SOM size.

Based on these test I chose the 10×10 SOM for all subsequent tests, as it shows a wide range of RGZ morphologies for a relatively small map size while still producing reasonable accuracy and F1 scores. The 5×5 map is not featured in any of the following tests due to the few relationships it displays.

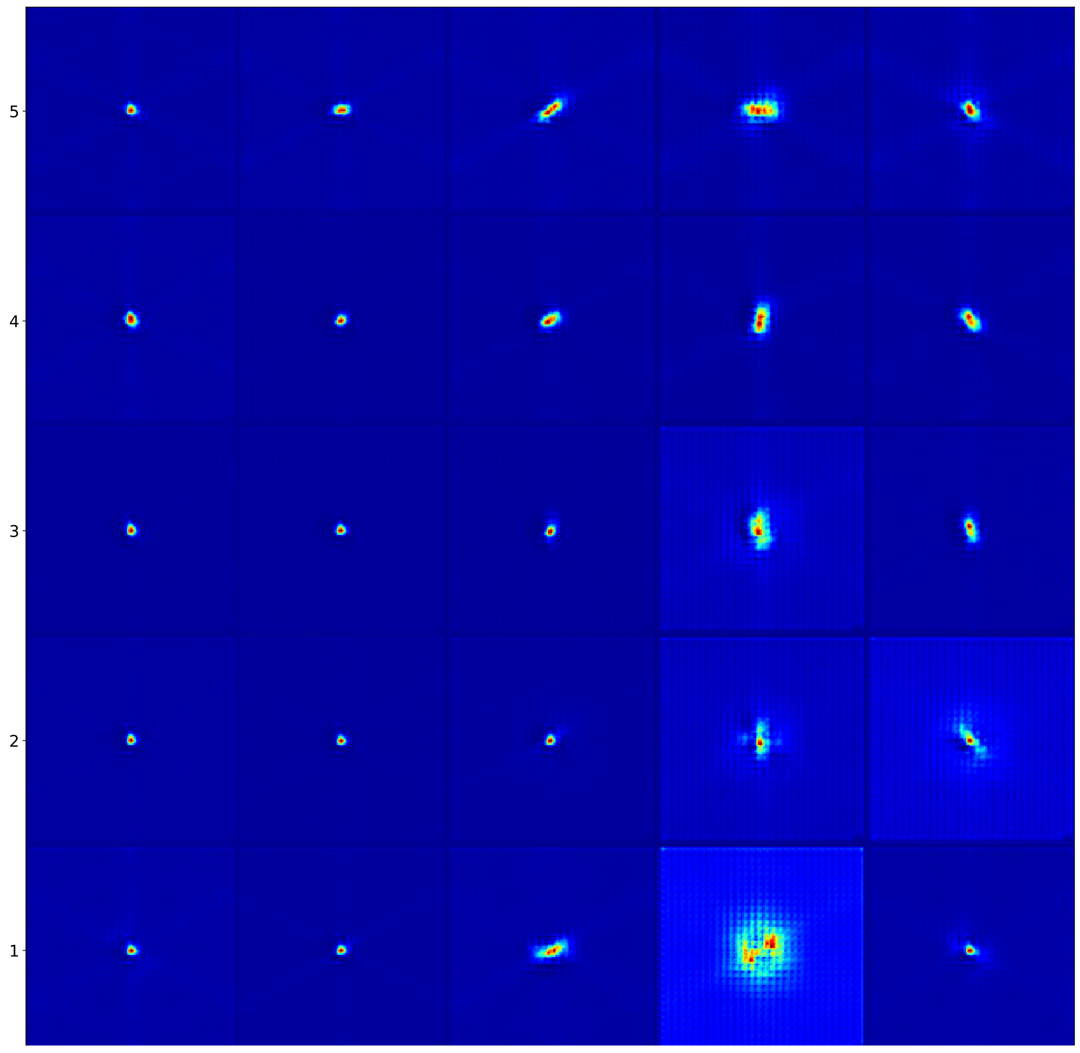

Figure 3.50: 5×5 toroidal SOM trained using linear learning rate decay, radius function and radius decay with each neuron displaying false colour decoded neuron weight images. Complex neurons weights here are unresolved and appear largely as large high intensity regions, usually with single peaks.

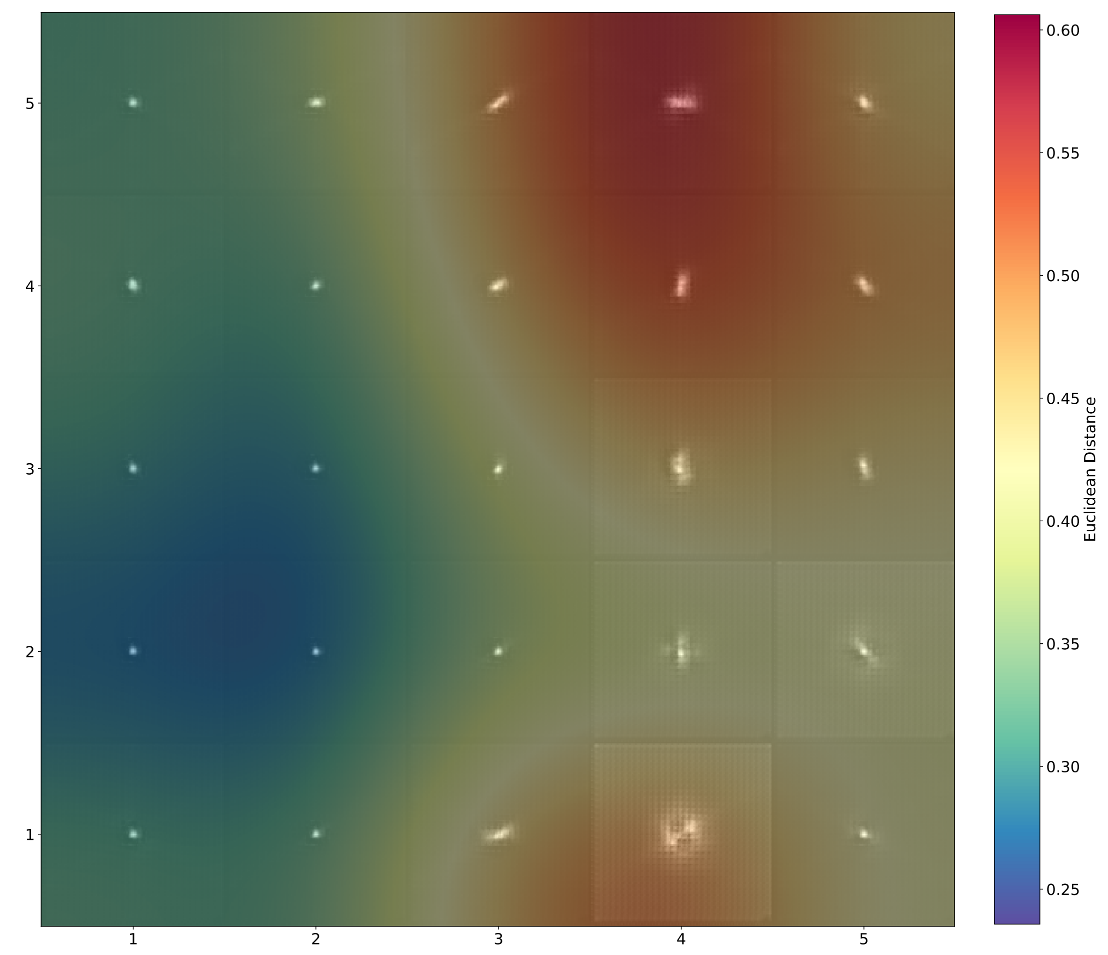

Figure 3.51: 5 × 5 toroidal SOM umat trained using linear learning rate decay, radius function and radius decay with each neuron displaying false colour decoded neuron weight images. The highest intensity sources here reside in high Euclidean distance regions.

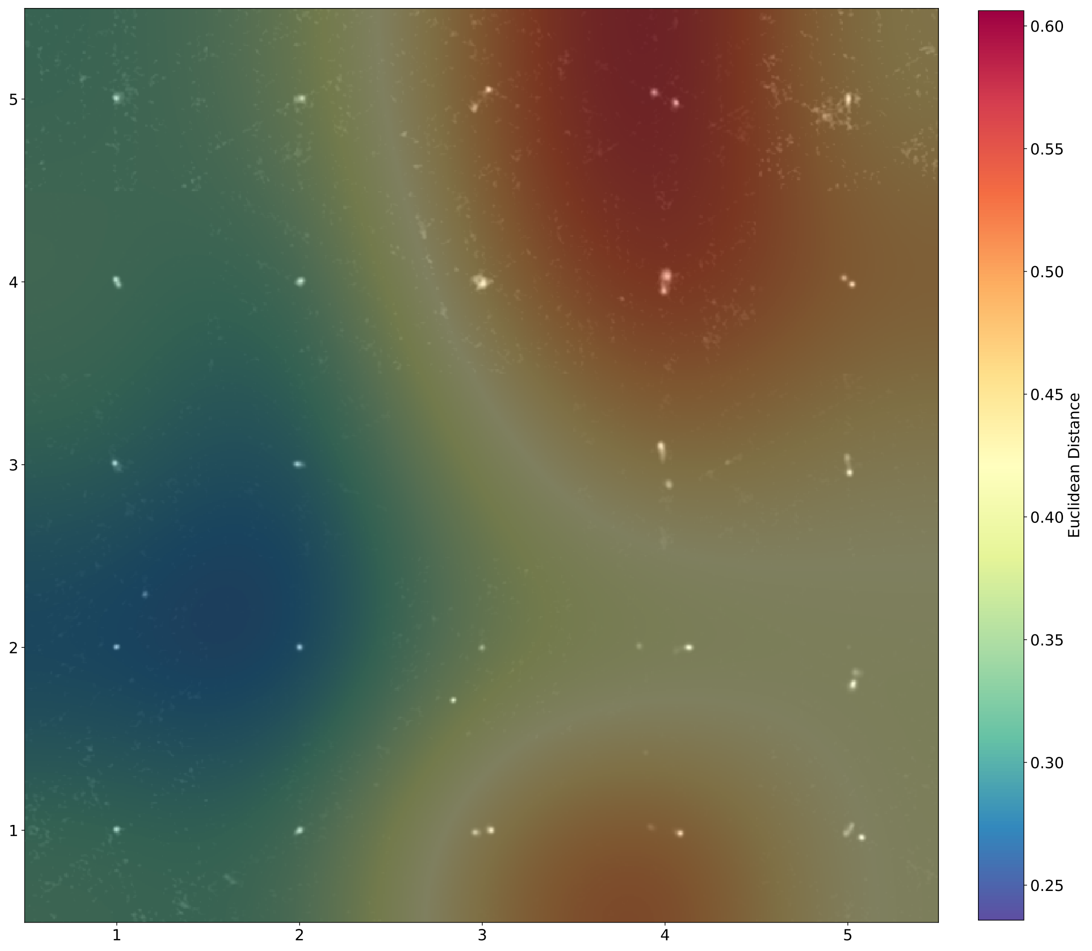

Figure 3.52: 5 × 5 toroidal SOM umat trained using linear learning rate decay, linear radius function and linear radius decay with each neuron displaying the RGZ image with the closest matching (Euclidean distance) latent vector transformation to the learned neuron weight. This small sized map fails to show the gradually developing morphology usually seen in larger maps such as the 10 × 10 in Figure 3.55.

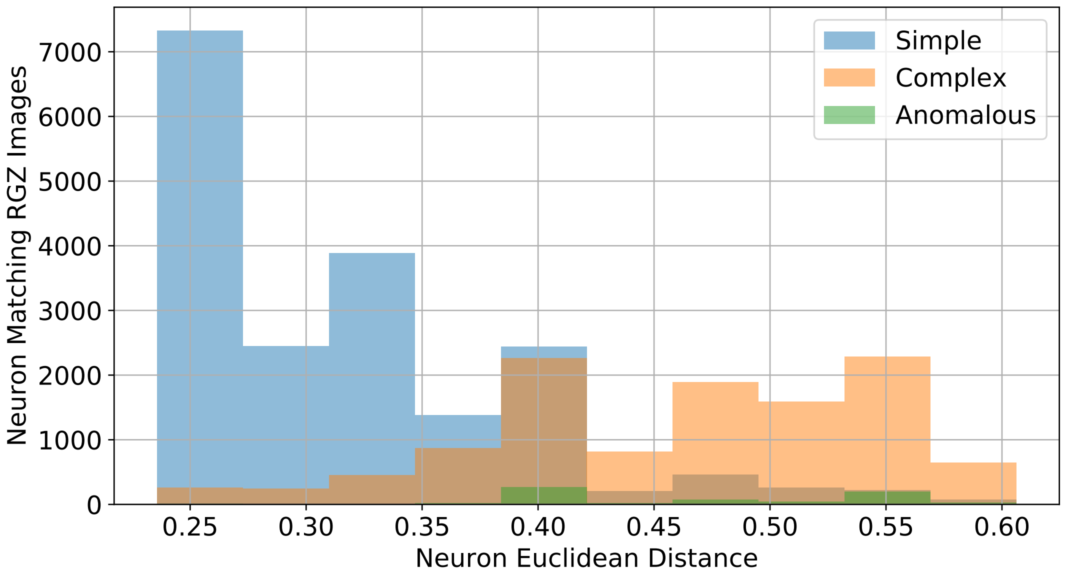

Figure 3.53: Distribution of neuron Euclidean distance from a 5×5 toroidal SOM umat trained using linear learning rate decay, radius function and radius decay. Colour blending indicates RGZ label derived complexity and anomalousness of all RGZ validation images matching each neuron. In these histograms, In these figures, I also display the proportion of each neuron distance bin with colour coding of RGZ labelled simple (11's), complex (not 11) and anomalous (peak and or component count greater than 3) sources, as outlined in Section 2.2.

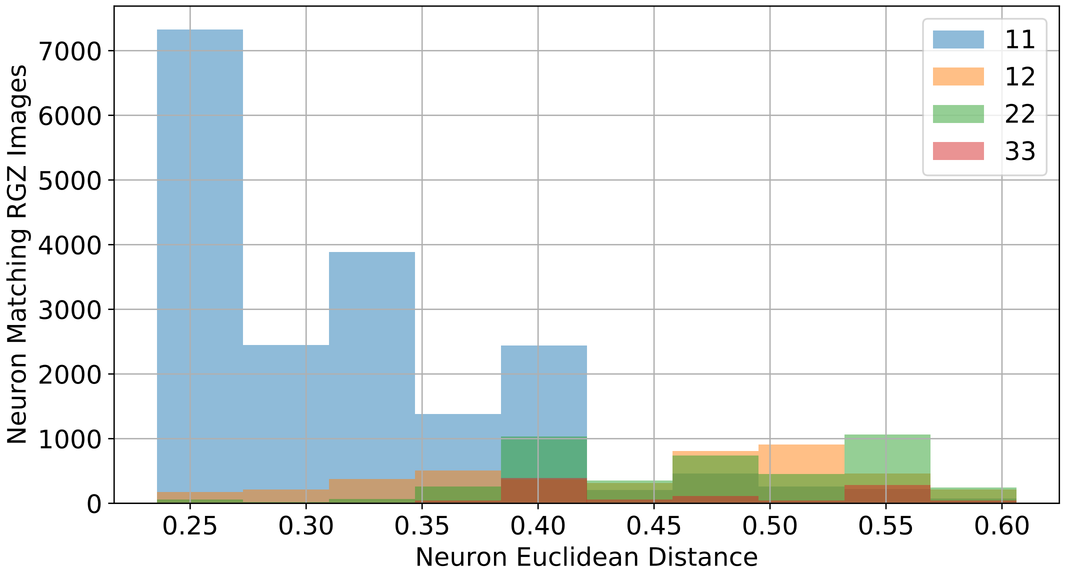

Figure 3.54: Distribution of neuron Euclidean distance from a 5×5 toroidal SOM umat trained using linear learning rate decay, radius function and radius decay. Colour blending indicates the RGZ label of the four most dominant labelled RGZ validation images matching each neuron. In these histograms, In these figures, I display the proportion of each neuron distance bin with colour coding of the four dominant source labels, 11, 12, 22, and 33, as outlined in Section 2.2.

Table 3.14: Accuracy and performance metrics of anomaly detection by separating neuron matches sources based on the Euclidean distance of neurons a 5 × 5 toroidal SOM umat at each SD 1-5, trained with a linear learning rate decay, radius function and radius decay

5 × 5 Map Size Neuron Euclidean Distance SD	Euclidean Distance at SD	Accuracy	Precision	Recall	F1 Score
SD1	0.106	0.377	0.377	1.000	0.547
SD2	0.212	0.377	0.377	1.000	0.547
SD3	0.318	0.746	0.603	0.953	**0.739**
SD4	0.424	**0.823**	0.855	0.639	0.731
SD5	0.531	0.711	0.908	0.259	0.403

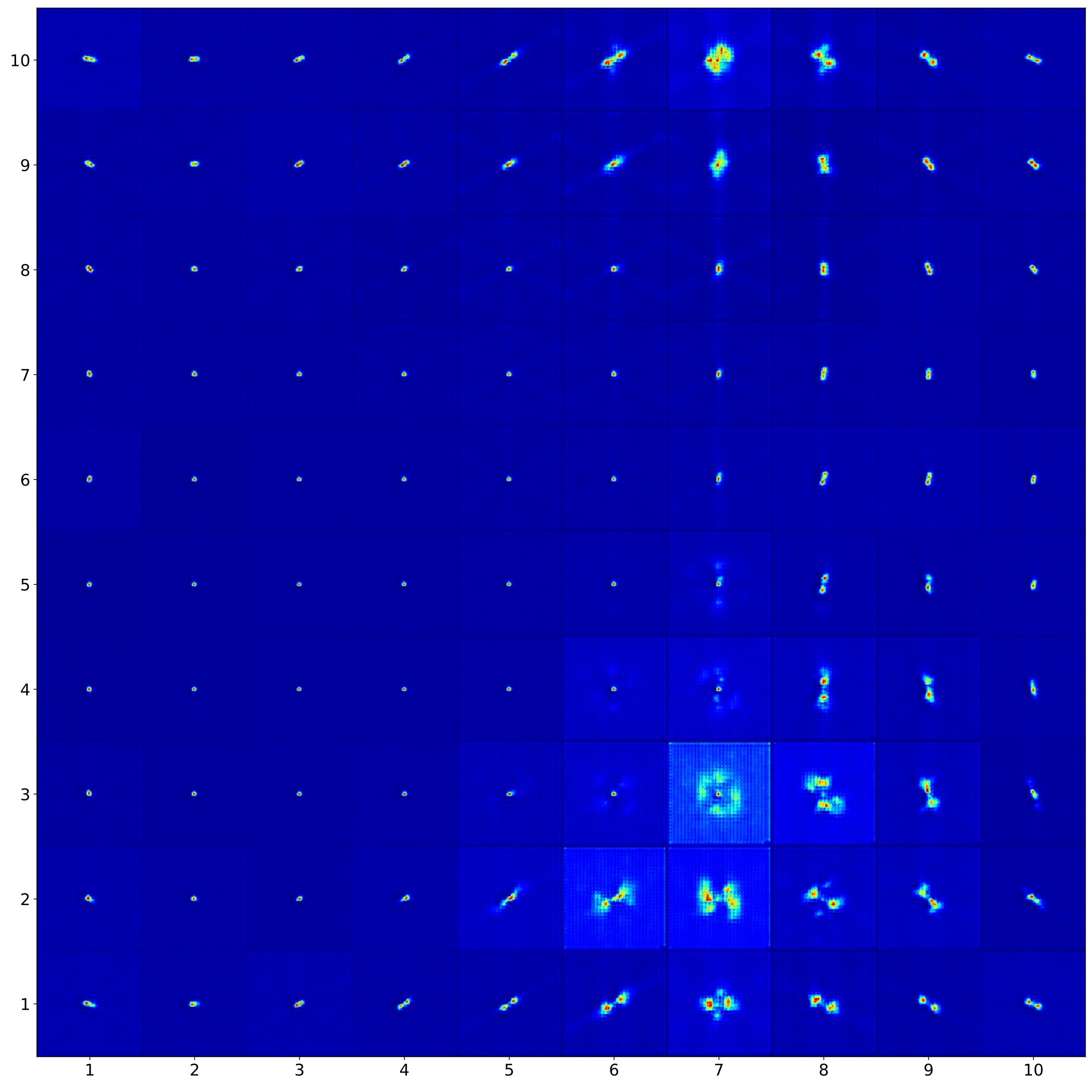

Figure 3.55: 10 × 10 toroidal SOM trained using linear learning rate decay, radius function and radius decay with each neuron displaying false colour decoded neuron weight images

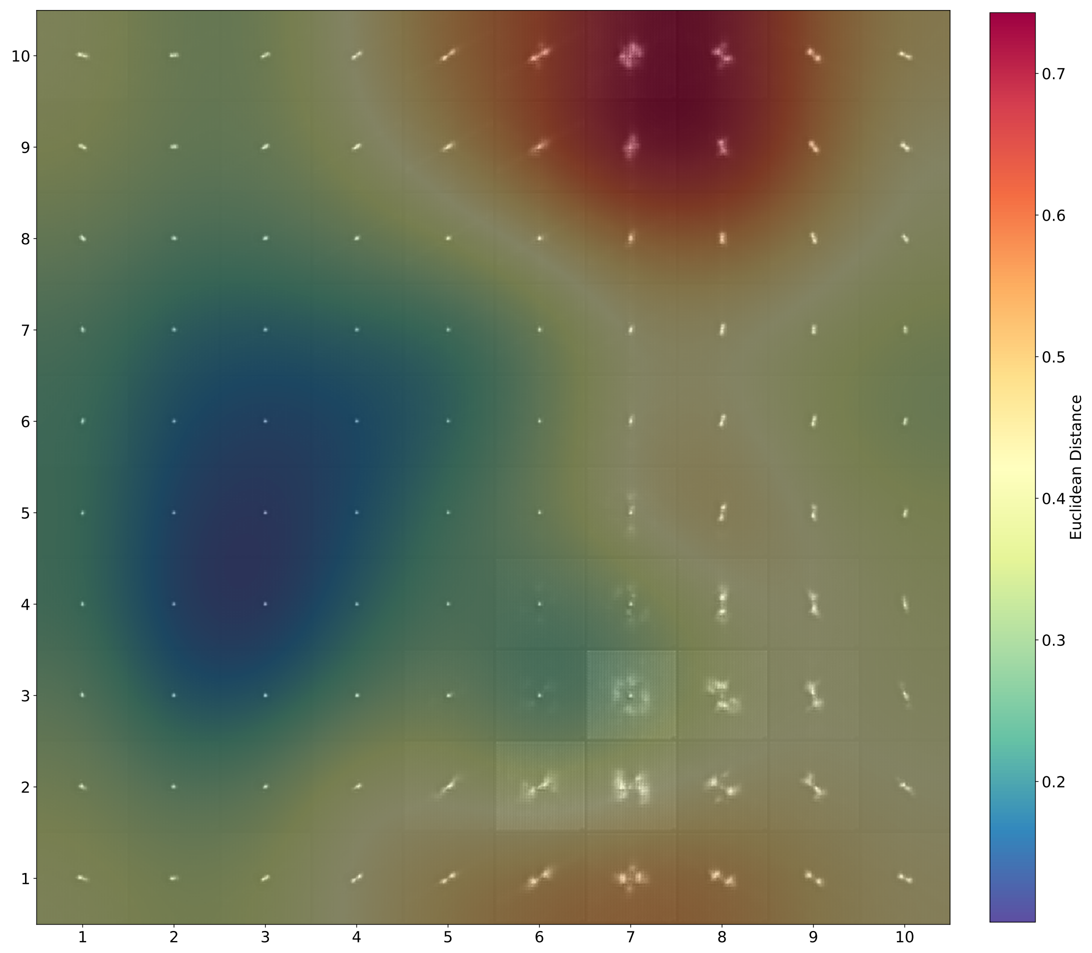

Figure 3.56: 10×10 toroidal SOM umat trained using linear learning rate decay, radius function and radius decay with each neuron displaying false colour decoded neuron weight images.

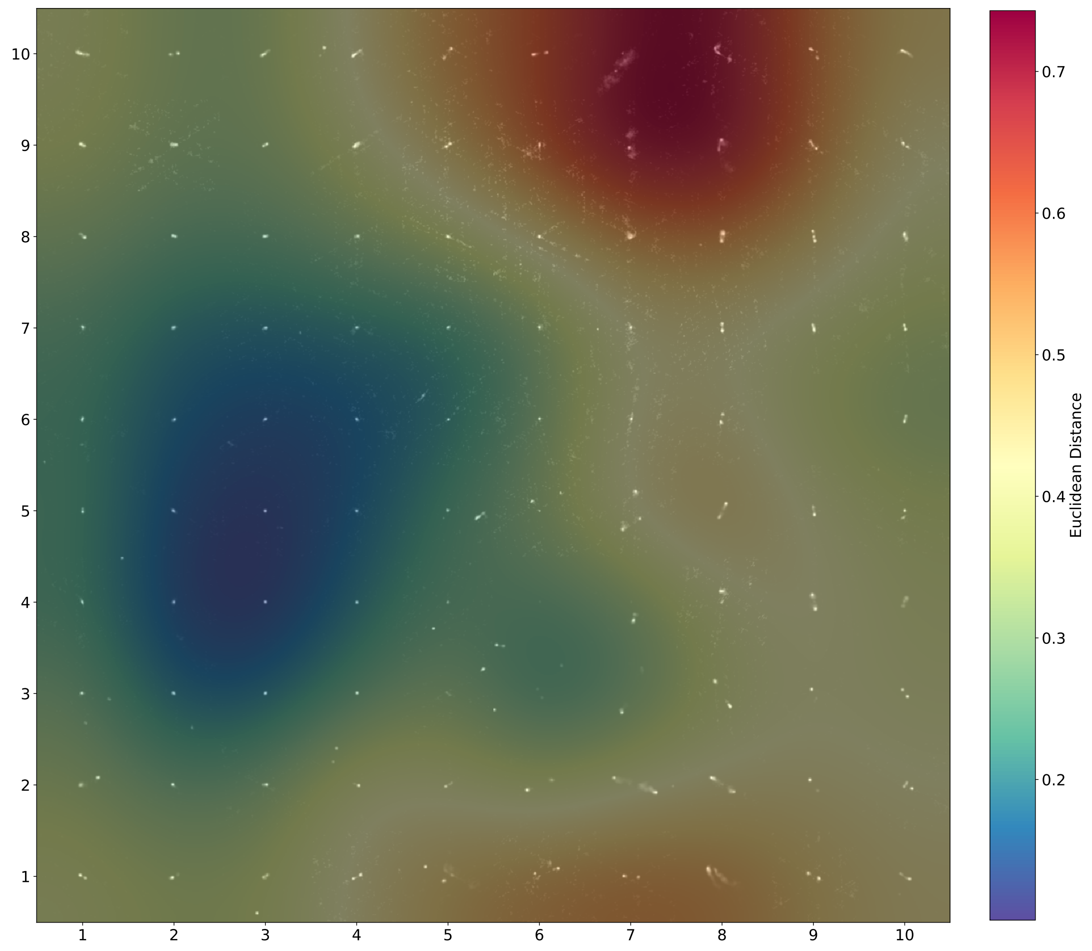

Figure 3.57: 10 × 10 toroidal SOM umat trained using linear learning rate decay, linear radius function and linear radius decay with each neuron displaying the RGZ image with the closest matching (Euclidean distance) latent vector transformation to the learned neuron weight.

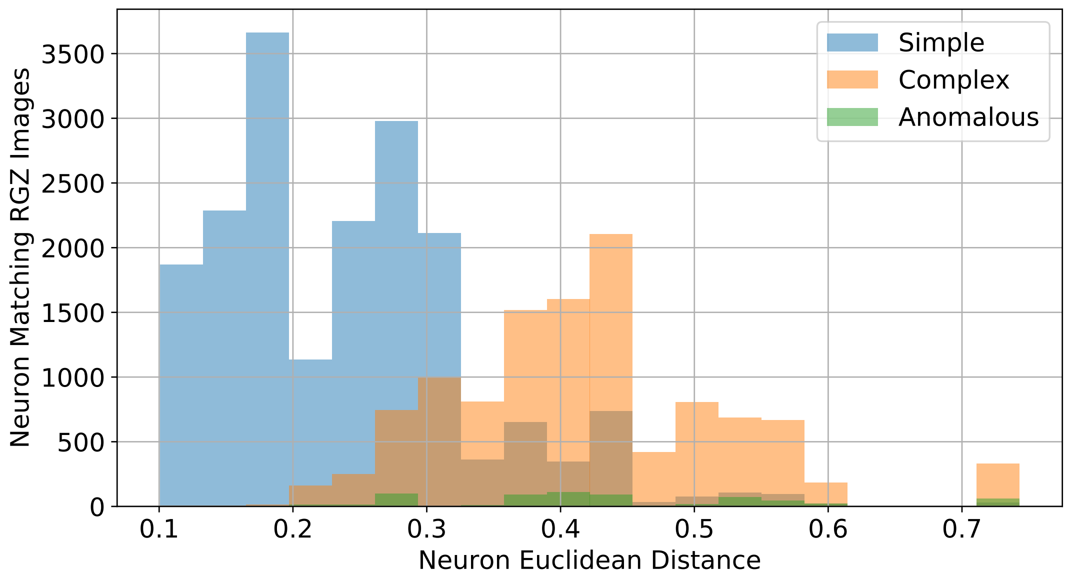

Figure 3.58: Distribution of neuron Euclidean distance from a 10 × 10 toroidal SOM umat trained using linear learning rate decay, radius function and radius decay. Colour blending indicates RGZ label derived complexity and anomalousness of all RGZ validation images matching each neuron.

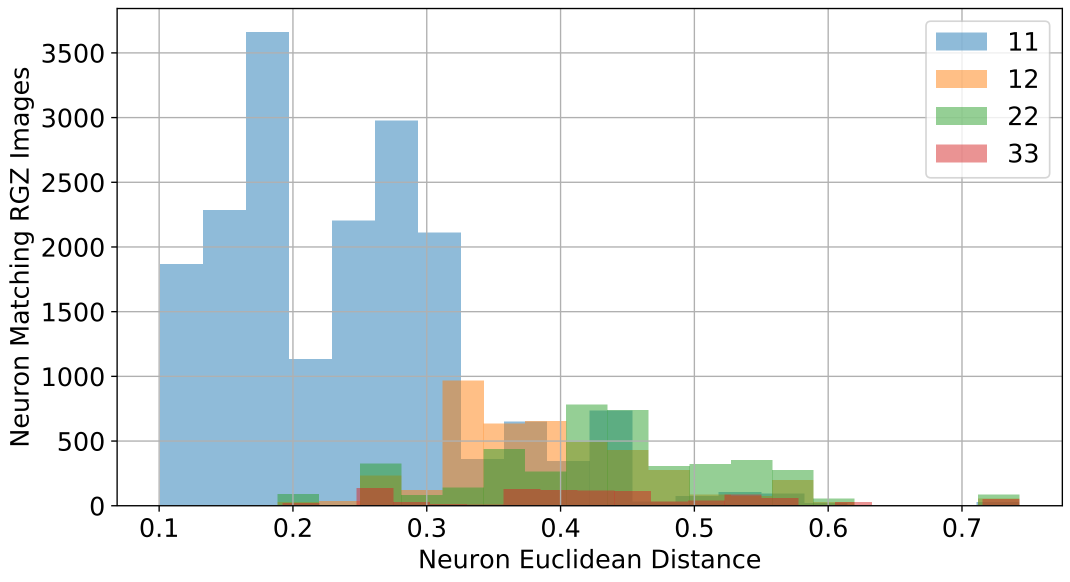

Figure 3.59: Distribution of neuron Euclidean distance from a 10 × 10 toroidal SOM umat trained using linear learning rate decay, radius function and radius decay. Colour blending indicates the RGZ label of the four most dominant labelled RGZ validation images matching each neuron.

Table 3.15: Accuracy and performance metrics of anomaly detection by separating neuron matches sources based on the Euclidean distance of neurons a 10 × 10 toroidal SOM umat at each SD 1-5, trained with a linear learning rate decay, radius function and radius decay

10 × 10 Map Size Neuron Euclidean Distance SD	Euclidean Distance at SD	Accuracy	Precision	Recall	F1 Score
SD1	0.131	0.439	0.402	0.999	0.573
SD2	0.262	0.734	0.590	0.961	**0.731**
SD3	0.393	**0.799**	0.839	0.576	0.683
SD4	0.524	0.677	0.881	0.165	0.278
SD5	0.655	0.633	0.917	0.029	0.057

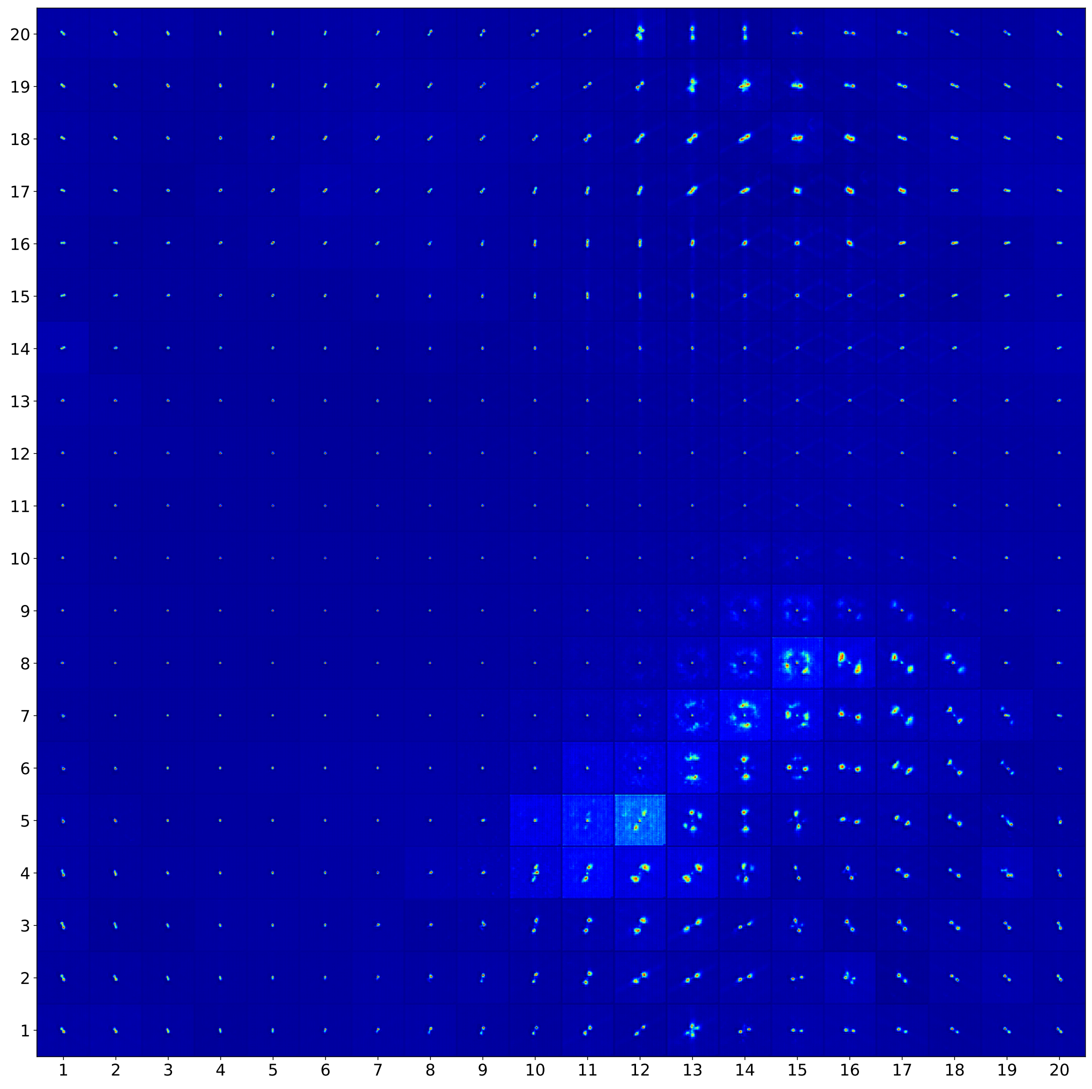

Figure 3.60: 20 × 20 toroidal SOM trained using linear learning rate decay, radius function and radius decay with each neuron displaying false colour decoded neuron weight images

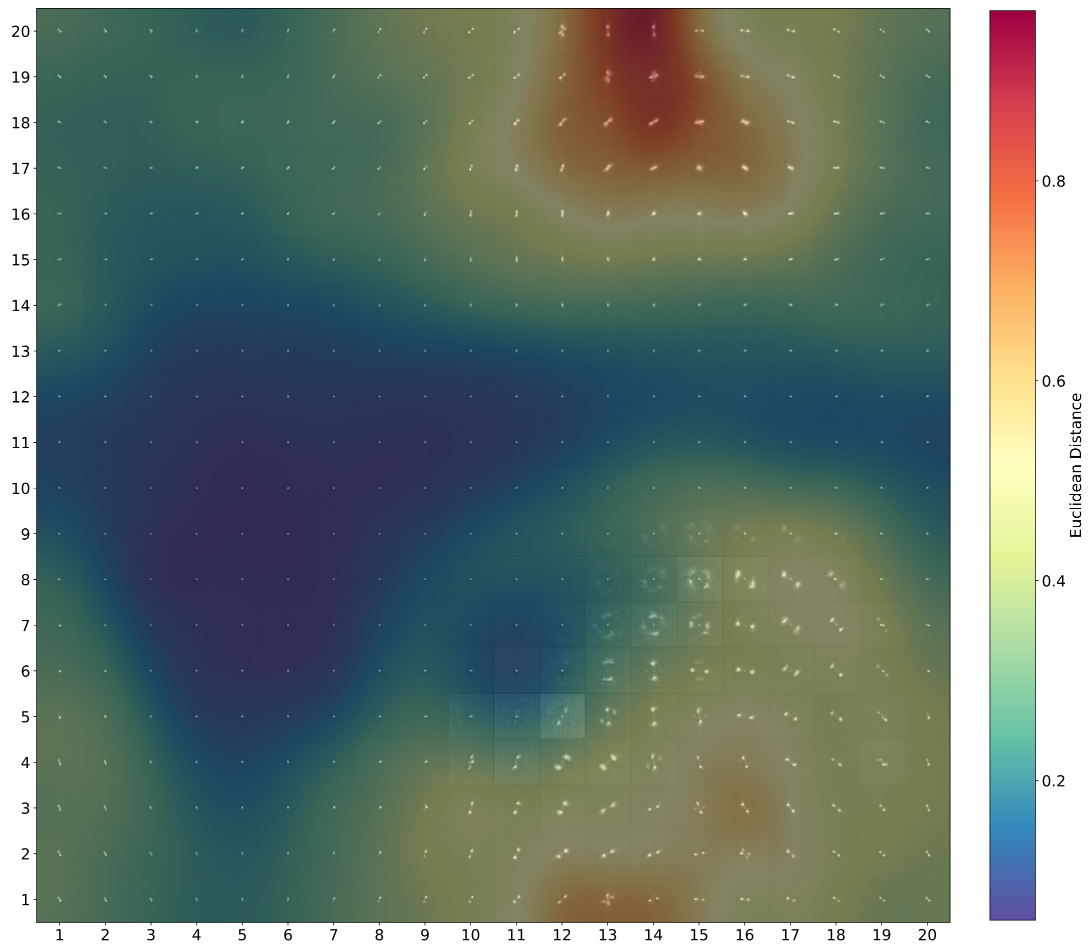

Figure 3.61: 20 × 20 toroidal SOM umat trained using linear learning rate decay, radius function and radius decay with each neuron displaying false colour decoded neuron weight images.

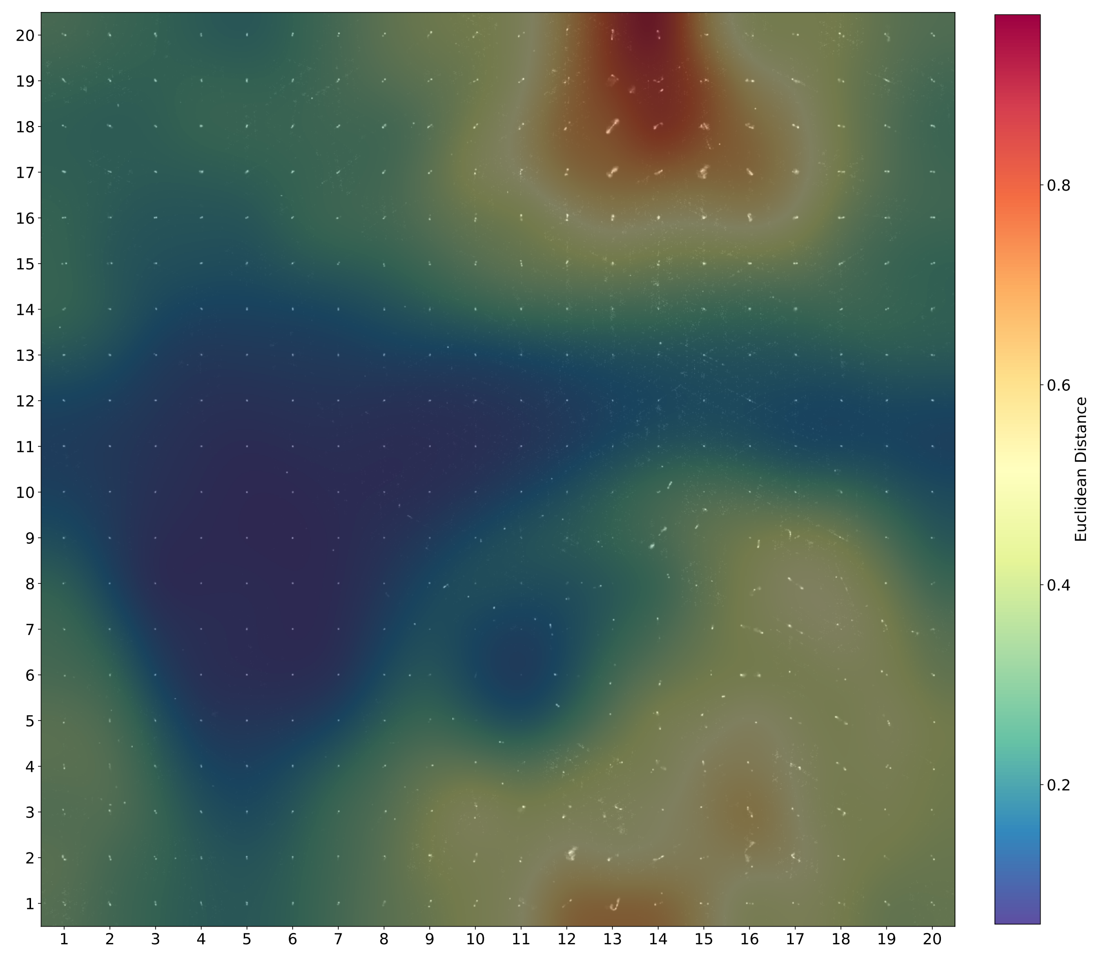

Figure 3.62: 20x20toroidal SOM umat trained using linear learning rate decay, linear radius function and linear radius decay with each neuron displaying the RGZ image with the closest matching (Euclidean distance) latent vector transformation to the learned neuron weight.

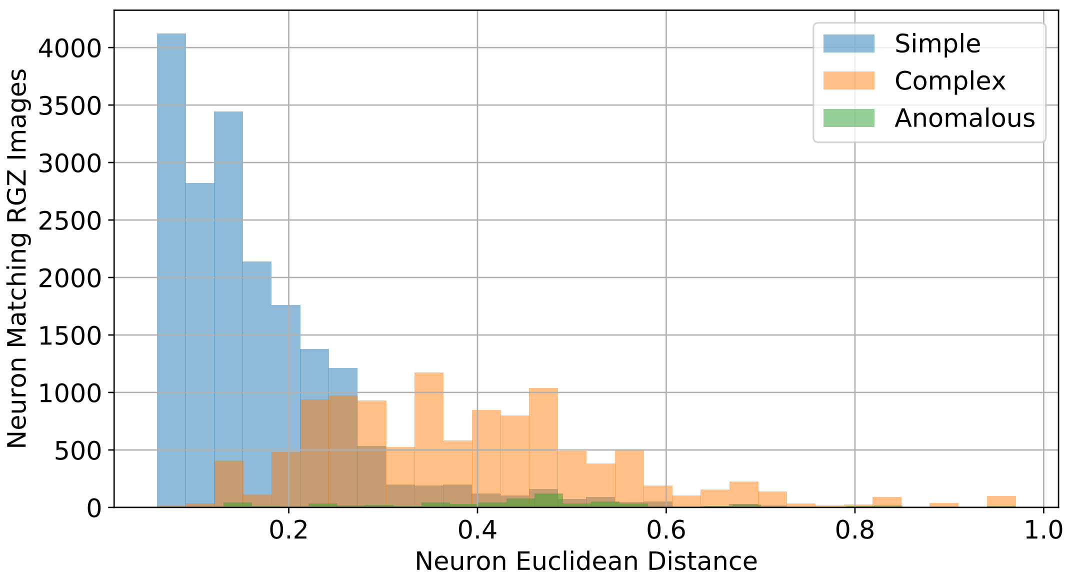

Figure 3.63: Distribution of neuron Euclidean distance from a 20 × 20 toroidal SOM umat trained using linear learning rate decay, radius function and radius decay. Colour blending indicates RGZ label derived complexity and anomalousness of all RGZ validation images matching each neuron.

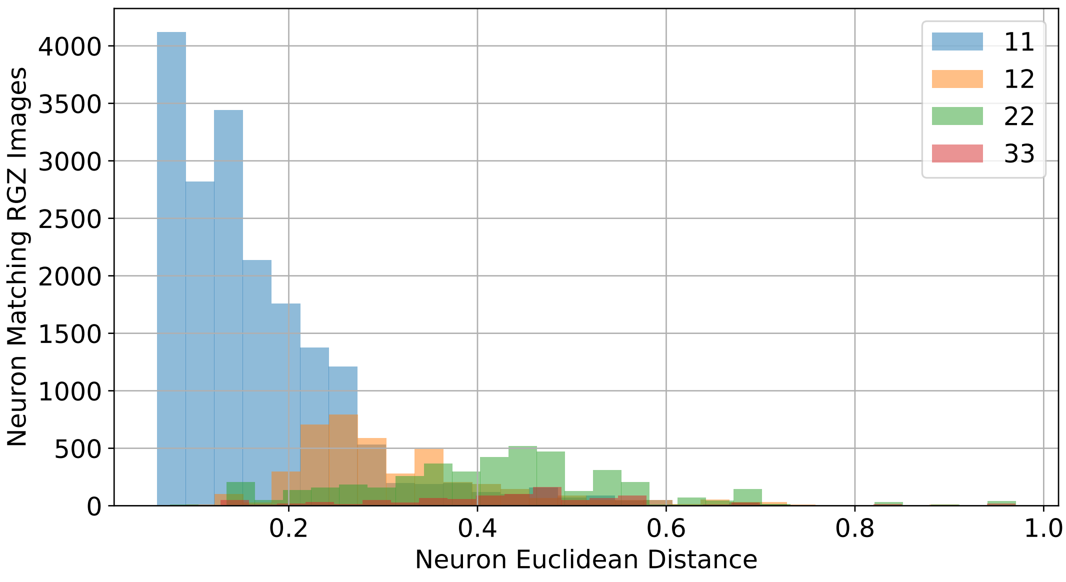

Figure 3.64: Distribution of neuron Euclidean distance from a 20 × 20 toroidal SOM umat trained using linear learning rate decay, radius function and radius decay. Colour blending indicates the RGZ label of the four most dominant labelled RGZ validation images matching each neuron.

Table 3.16: Accuracy and performance metrics of anomaly detection by separating neuron matches sources based on the Euclidean distance of neurons a 20 × 20 toroidal SOM umat at each SD 1-5, trained with a linear learning rate decay, radius function and radius decay

20 × 20 Map Size Neuron Euclidean Distance SD	Euclidean Distance at SD	Accuracy	Precision	Recall	F1 Score
SD1	0.163	0.732	0.589	0.958	**0.729**
SD2	0.326	**0.820**	0.864	0.619	0.721
SD3	0.489	0.691	0.882	0.207	0.335
SD4	0.652	0.648	0.917	0.071	0.132
SD5	0.815	0.630	0.961	0.020	0.039

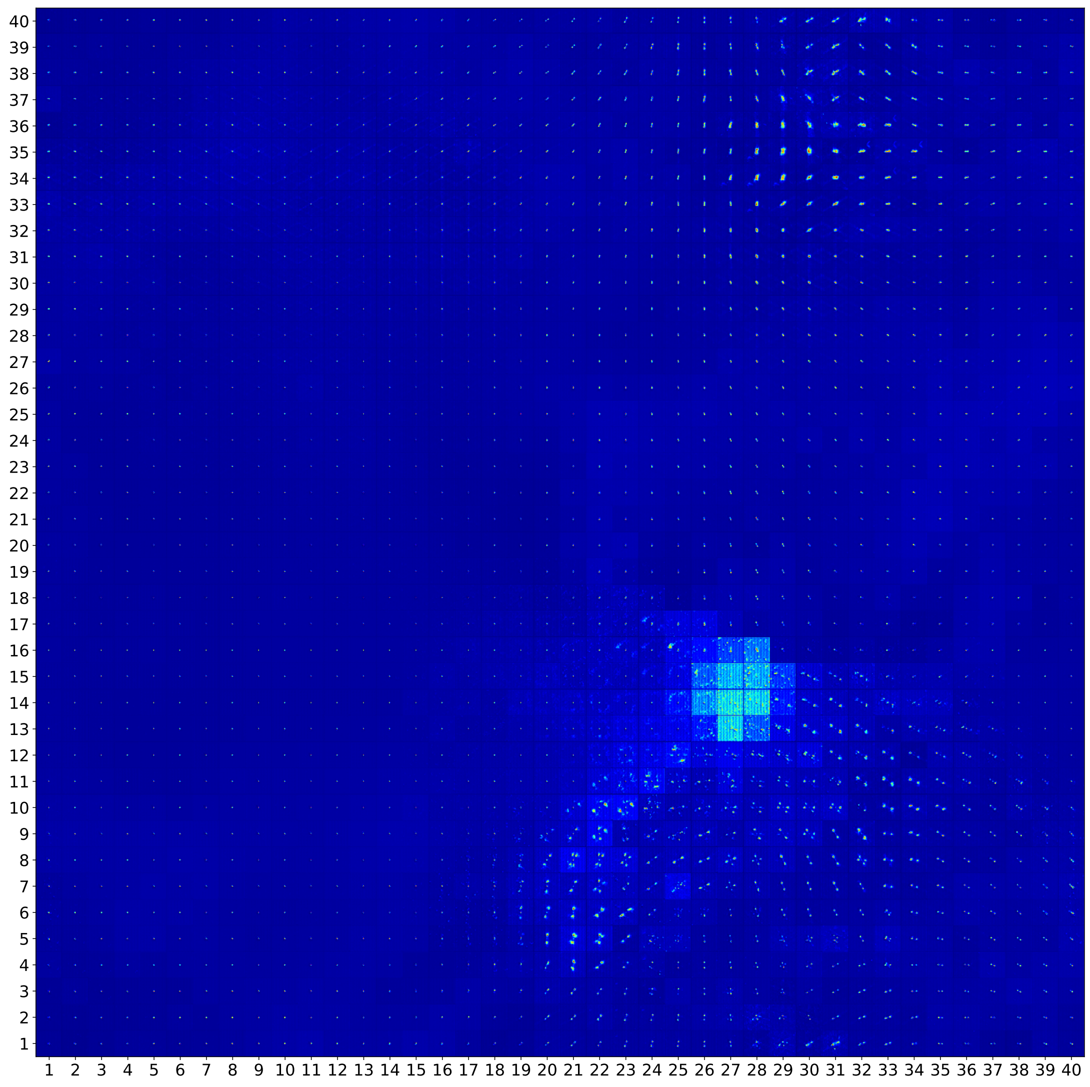

Figure 3.65: 40 × 40 toroidal SOM trained using linear learning rate decay, radius function and radius decay with each neuron displaying false colour decoded neuron weight images

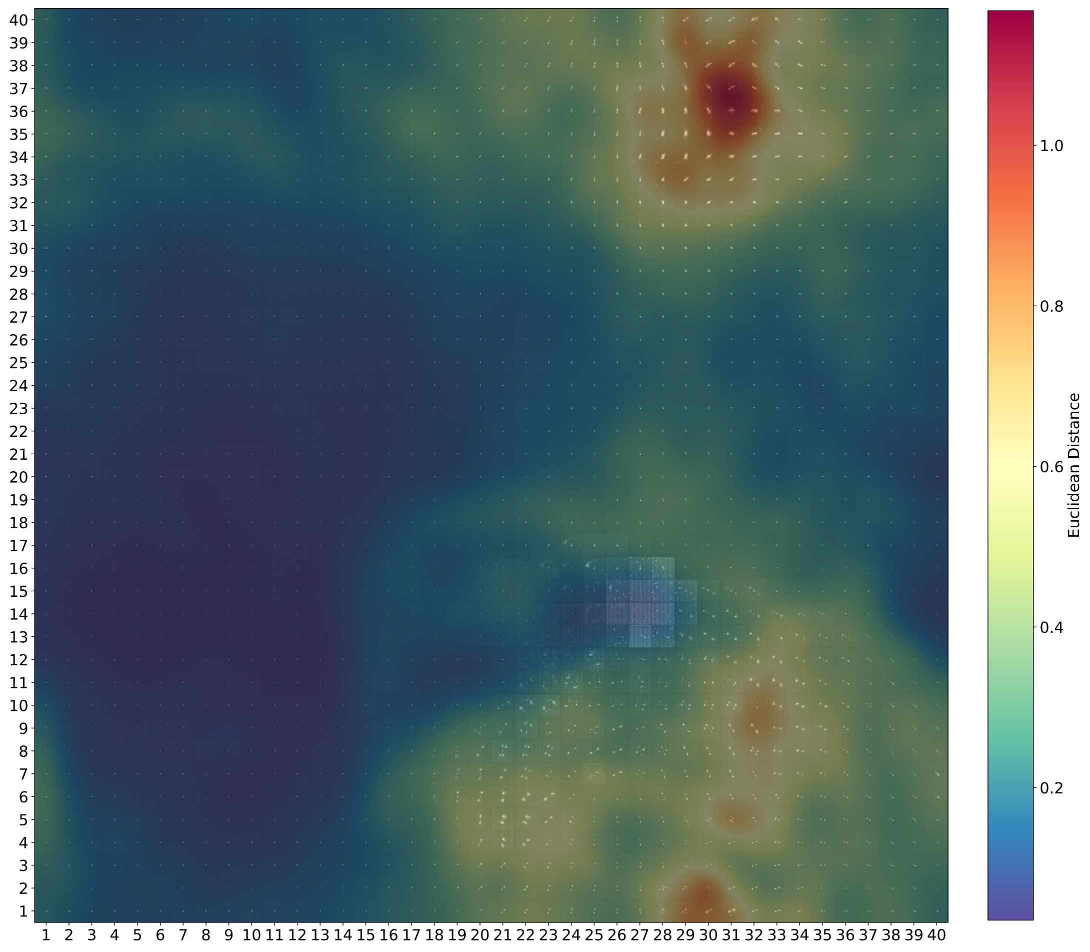

Figure 3.66: 40×40 toroidal SOM umat trained using linear learning rate decay, radius function and radius decay with each neuron displaying false colour decoded neuron weight images.

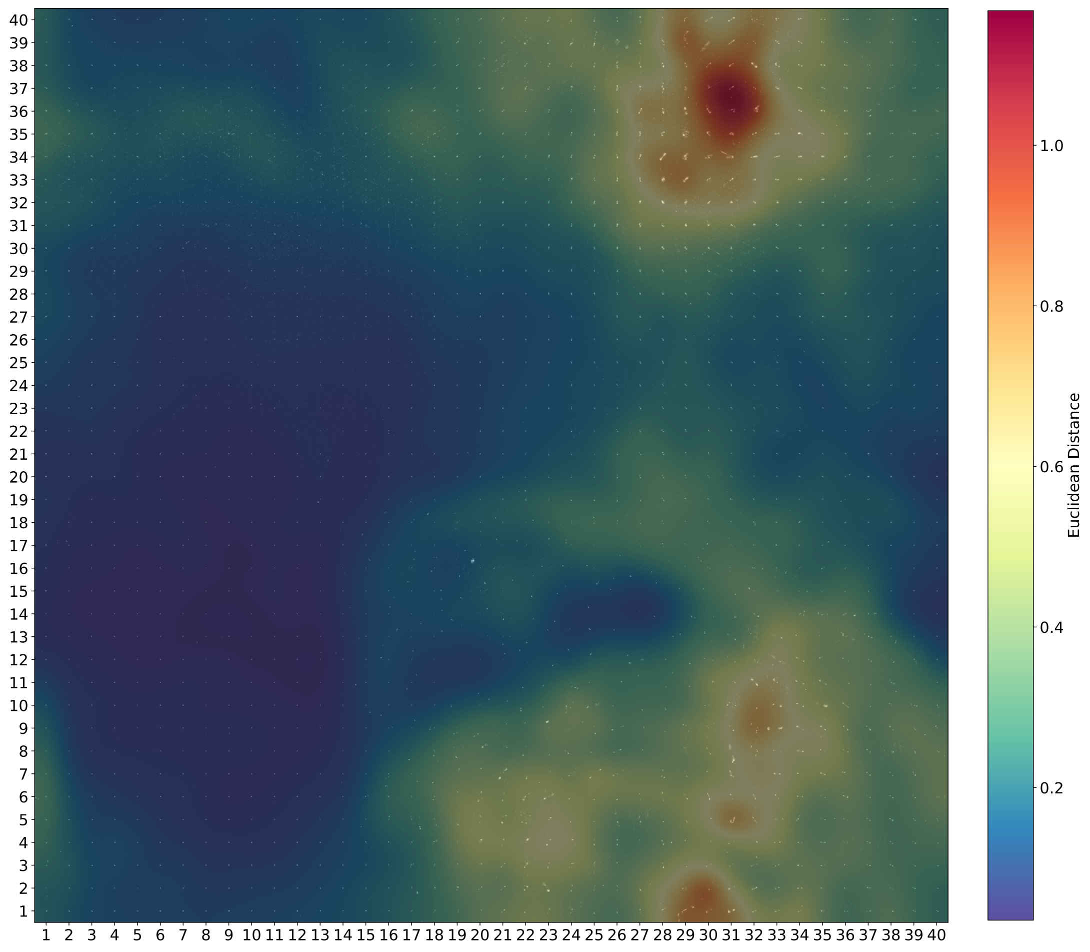

Figure 3.67: 40 × 40 toroidal SOM umat trained using linear learning rate decay, linear radius function and linear radius decay with each neuron displaying the RGZ image with the closest matching (Euclidean distance) latent vector transformation to the learned neuron weight.

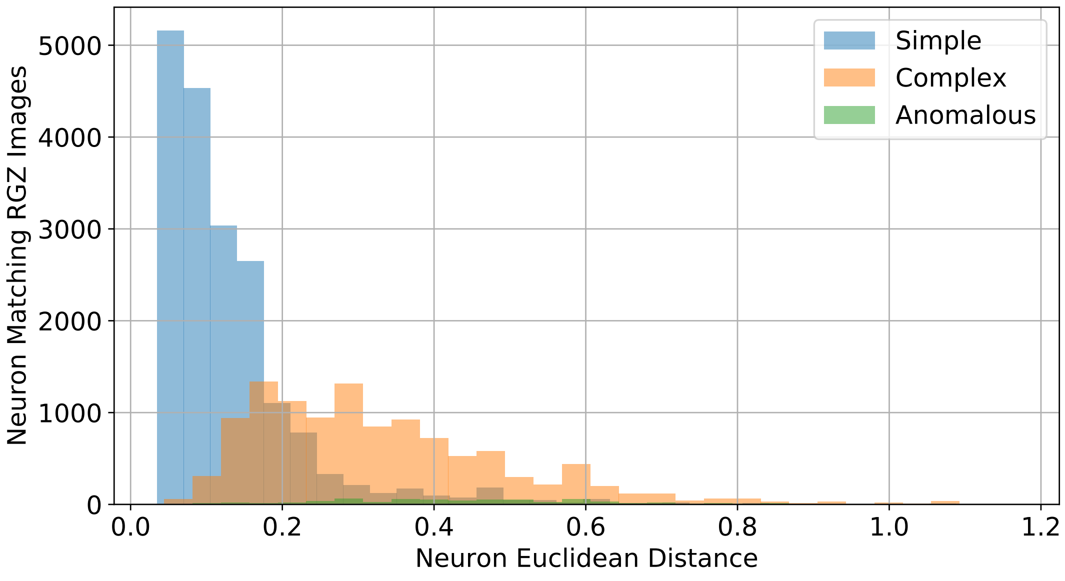

Figure 3.68: Distribution of neuron Euclidean distance from a 40 × 40 toroidal SOM umat trained using linear learning rate decay, radius function and radius decay. Colour blending indicates RGZ label derived complexity and anomalousness of all RGZ validation images matching each neuron.

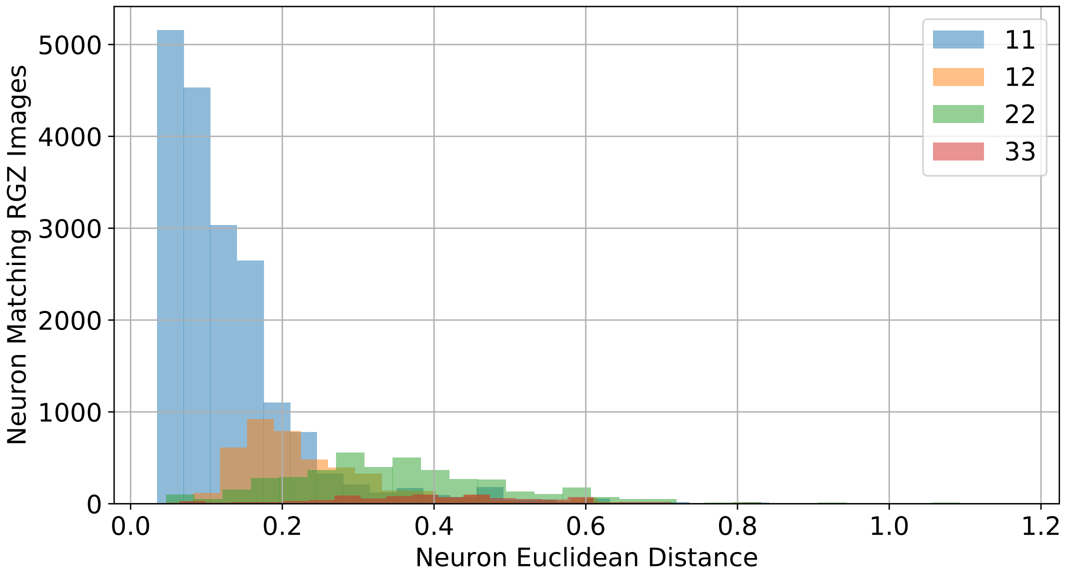

Figure 3.69: Distribution of neuron Euclidean distance from a 40 × 40 toroidal SOM umat trained using linear learning rate decay, radius function and radius decay. Colour blending indicates the RGZ label of the four most dominant labelled RGZ validation images matching each neuron.

Table 3.17: Accuracy and performance metrics of anomaly detection by separating neuron matches sources based on the Euclidean distance of neurons a 40 × 40 toroidal SOM umat at each SD 1-5, trained with a linear learning rate decay, radius function and radius decay

40 × 40 Map Size Neuron Euclidean Distance SD	Euclidean Distance at SD	Accuracy	Precision	Recall	F1 Score
SD1	0.163	**0.814**	0.704	0.873	**0.779**
SD2	0.325	0.755	0.851	0.425	0.567
SD3	0.488	0.671	0.870	0.149	0.255
SD4	0.650	0.636	0.866	0.042	0.080
SD5	0.813	0.628	0.850	0.016	0.031

Table 3.18: Processing time metrics of each toroidal SOM with square 5,10,20 and 40 sizes maps trained with a linear learning rate decay, radius function and radius decay

Map Size	Total Training Time (s)	Average Training Time Per Epoch (s)
5 × 5	3.475	0.290
10 × 10	8.272	0.689
20 × 20	25.528	2.127
40 × 40	83.623	6.969

Table 3.19: Accuracy and performance metrics of anomaly detection of the best SD in each toroidal SOM with square 5,10,20 and 40 sizes maps trained with a linear learning rate decay, radius function and radius decay

Map Size	SD	Euclidean Distance at SD	Accuracy	Precision	Recall	F1 Score
5 × 5	SD3	0.318	0.746	0.603	0.953	0.739
10 × 10	SD2	0.262	0.734	0.590	0.961	0.731
20 × 20	SD1	0.163	0.732	0.589	0.958	0.729
40 × 40	SD1	0.163	**0.814**	0.704	0.873	**0.779**

3.3.3.2 Learning Rate Decay

Outlined in Section 2.5.1, the learning rate decay is the gradual minimisation of the learning rate constant of neuron weights during training. This test shows that the exponential alternative to linear learning rate decay has a minimal effect on SOM training and is robust to changes in learning rates.

In this test, I compared the linear learning rate decay function from Section 3.3.3.1 to the exponential learning rate decay function. I tested the effects of these decay functions on the SOM accuracy for the most accurate SOM map sizes. These sizes are the 10 × 10, as justified in Section 3.3.3.1. The accuracy using each decay function has very little change, with a slight increase in training time. As a result, I use the linear learning rate decay function in all subsequent tests.

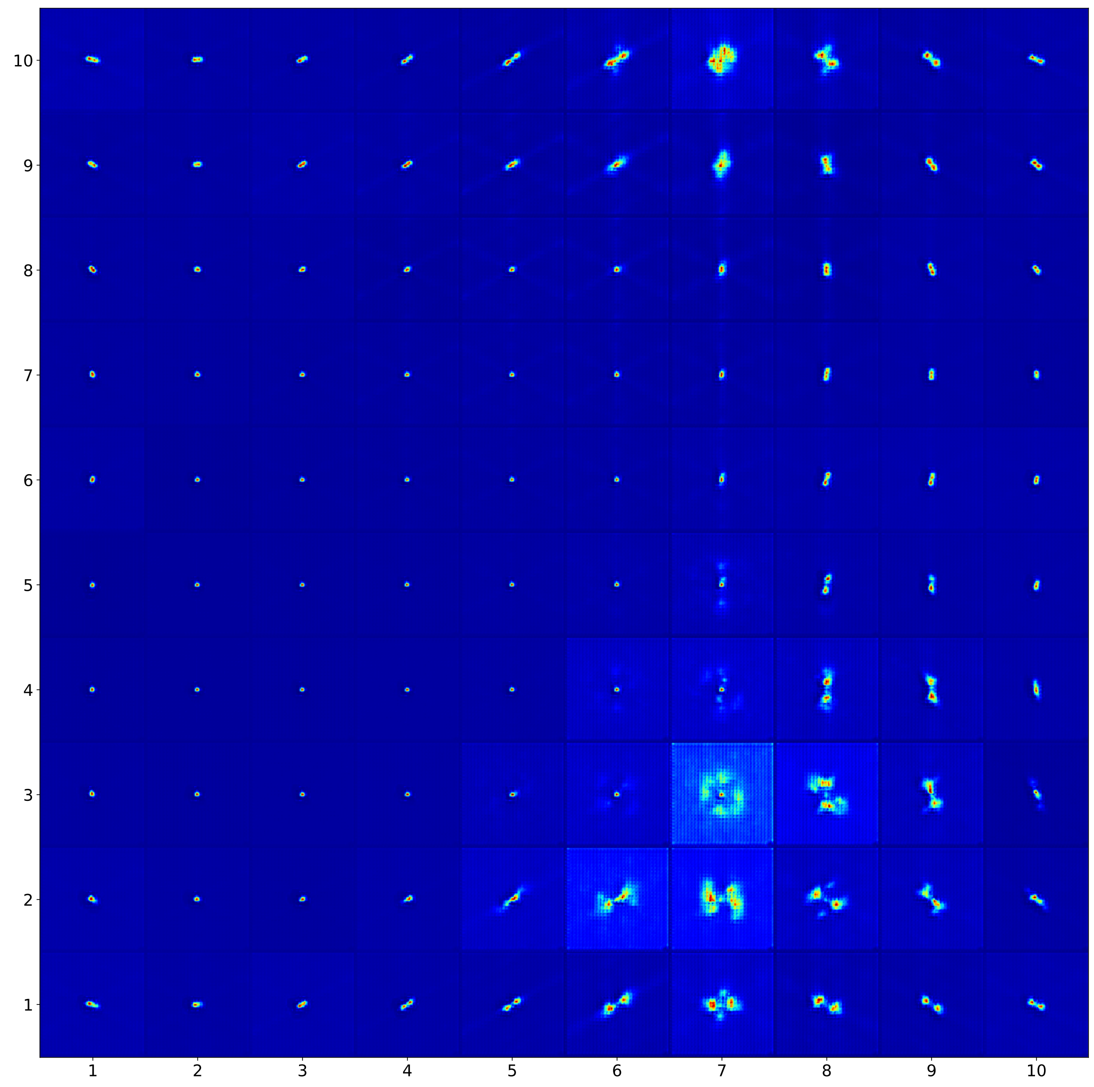

Figure 3.70: 10 × 10 toroidal SOM trained using linear learning rate decay, radius function and radius decay with each neuron displaying false colour decoded neuron weight images

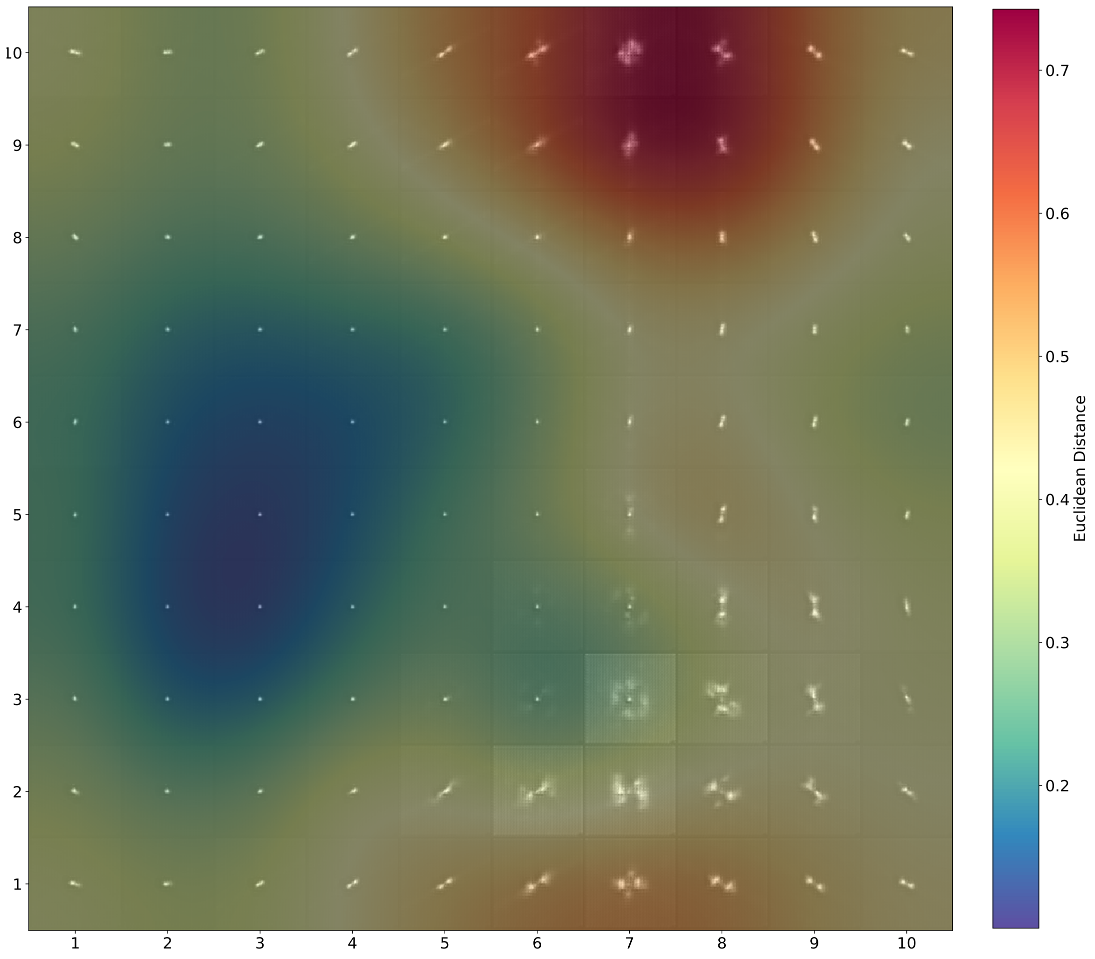

Figure 3.71: 10×10 toroidal SOM umat trained using linear learning rate decay, radius function and radius decay with each neuron displaying false colour decoded neuron weight images.

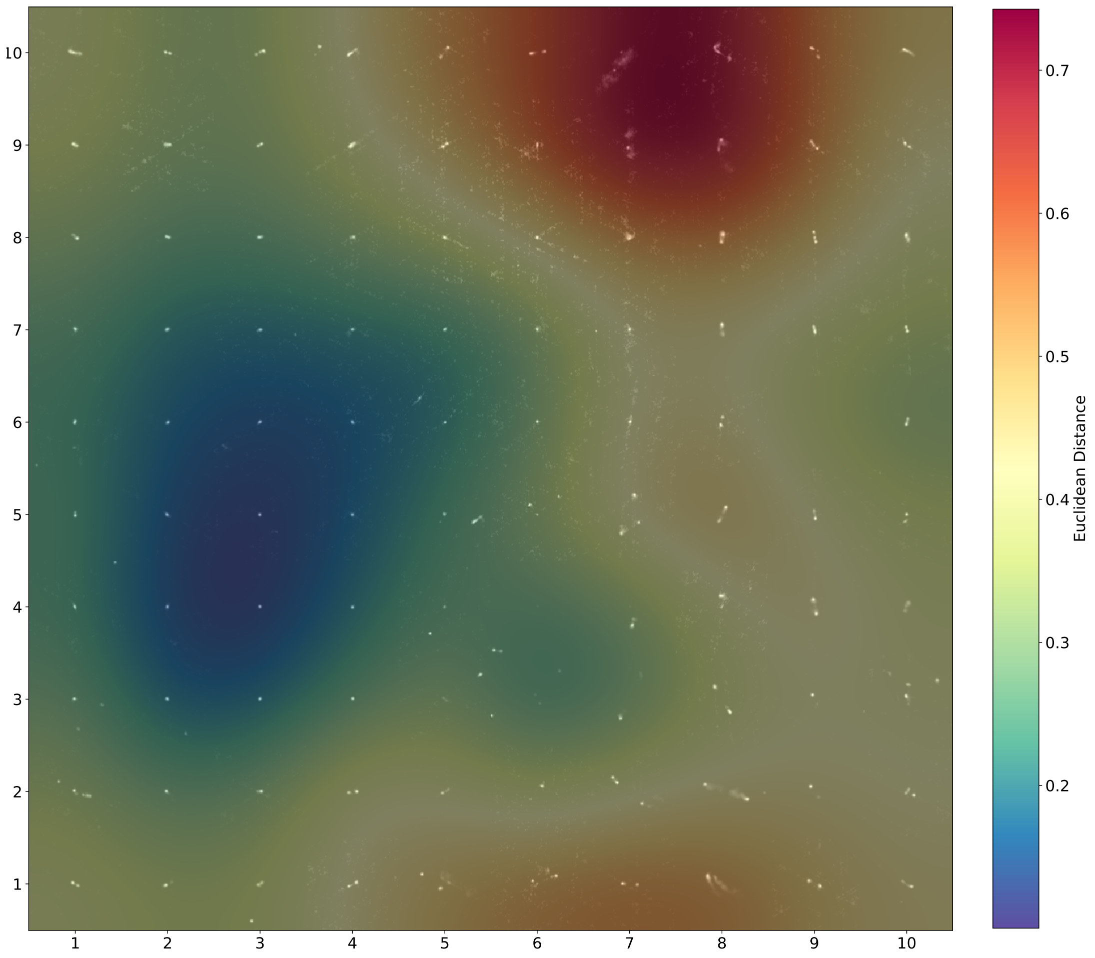

Figure 3.72: 10 × 10 toroidal SOM umat trained using linear learning rate decay, linear radius function and linear radius decay with each neuron displaying the RGZ image with the closest matching (Euclidean distance) latent vector transformation to the learned neuron weight.

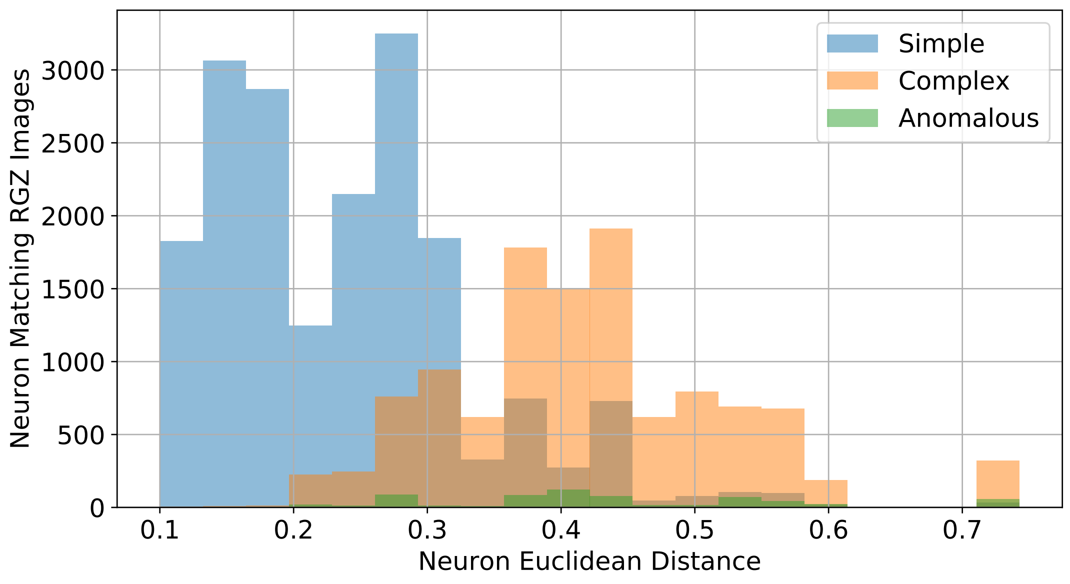

Figure 3.73: Distribution of neuron Euclidean distance from a 10 × 10 toroidal SOM umat trained using linear learning rate decay, radius function and radius decay. Colour blending indicates RGZ label derived complexity and anomalousness of all RGZ validation images matching each neuron.

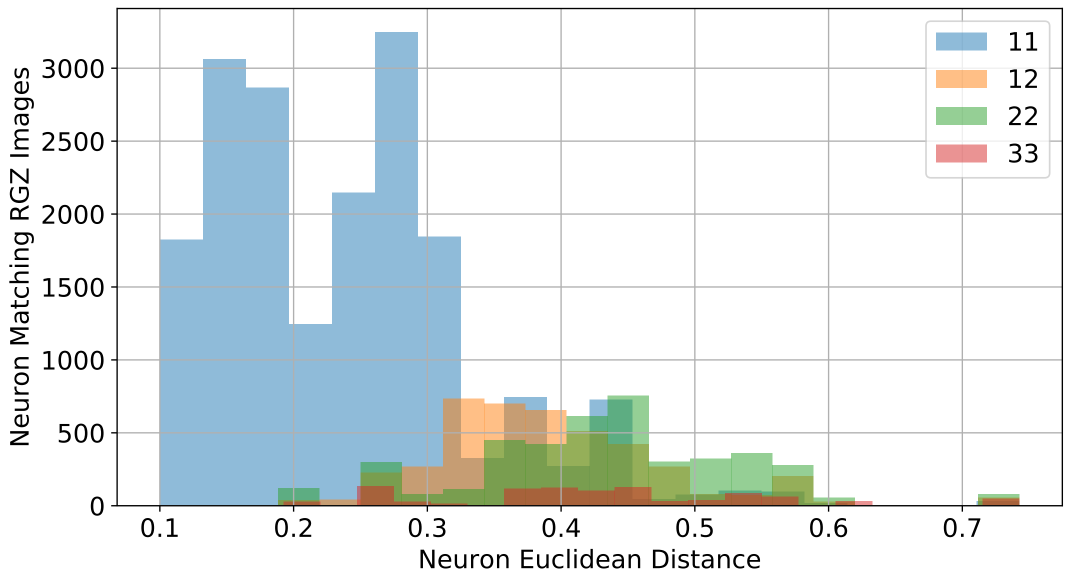

Figure 3.74: Distribution of neuron Euclidean distance from a 10 × 10 toroidal SOM umat trained using exponential learning rate decay, linear radius function and radius decay. Colour blending indicates the RGZ label of the four most dominant labelled RGZ validation images matching each neuron.

Table 3.20: Accuracy and performance metrics of anomaly detection by separating neuron matches sources based on the Euclidean distance of neurons a 10×10 toroidal SOM umat at each SD 1-5, trained with an exponential learning rate decay, linear radius function and radius decay

10 × 10 Map Size Neuron Euclidean Distance SD	Euclidean Distance at SD	Accuracy	Precision	Recall	F1 Score
SD1	0.131	0.437	0.401	0.999	0.572
SD2	0.262	0.732	0.589	0.956	**0.729**
SD3	0.393	**0.798**	0.838	0.575	0.682
SD4	0.525	0.677	0.881	0.166	0.279
SD5	0.656	0.633	0.911	0.028	0.055

Table 3.21: Accuracy and performance metrics of anomaly detection of the best SD in each toroidal SOM with square 10×10 map size trained with a exponential and linear learning rate decay with a linear radius function and radius decay

Map Size and Learning Rate Decay	SD	Euclidean Distance at SD	Accuracy	Precision	Recall	F1 Score
10x10 Linear	SD2	0.262	**0.734**	0.590	0.961	**0.731**
10 × 10 Exponential	SD2	0.262	0.732	0.589	0.956	0.729

Table 3.22: Processing time metrics of each toroidal SOM in each toroidal SOM with square 10×10 map size trained with a exponential and linear learning rate decay with a linear radius function and radius decay

Map Size and Learning Rate Decay	Total Training Time (s)	Average Training Time Per Epoch (s)
10 × 10 Linear	8.272	0.689
10 × 10 Exponential	8.571	0.714

3.3.3.3 Learning Neighbourhood Type and Radius Decay

In Section 2.5.1, I outline the SOM learning neighbourhood types and radius decay functions. These tests show the initial method with linear neighbourhood and radius decay function is the most accurate with the least amount of training time when trained on radio-astronomical latent features vectors. I tested the combined effects of these configurations in four tests, linear neighbourhood with each decay rate and the Gaussian type with each radius decay function. These tests are evaluated by the training time and the accuracy of each SOM.

These tests show a minor increase in accuracy with the Gaussian neighbourhood type. Compared to the linear neighbourhood type, the Gaussian neighbourhood appears to learn fewer but more resolved ‘average rotation neurons’ and higher resolution morphologies. These differences are illustrated in Figure 3.75, compared the results of training with the linear neighbourhood function in Figure 3.55. In addition to the differences in morphology, this radius type produces a broader range of Euclidean distances between 0.1 and 1.2, where the linear radius type is restricted to 0.1 and 0.8. This greater range provides more a defined neuron distance distribution as shown in Figures 3.79 and 3.78, where point sources mostly reside in a broad low distance peak, and more complex morphologies show more defined, and higher mean distances compared to Figures 3.59 and 3.58. This improvement in the neuron distance range and a slight change in neuron morphology has a moderate impact on the performance of the complexity separation, as shown in Table 3.27, and is consequently retained for subsequent tests.

Although the Gaussian neighbourhood type and linear radius decay show a moderate performance increase, I show the exponential radius decay function in both the Gaussian and linear neighbourhood types results in very poor modelling with no complete morphologies or clustered matching RGZ images. These poor results are also reflected in their accompanying neuron Euclidean distance distributions, complexity separation metrics and final comparisons across all tests in Table 3.27.

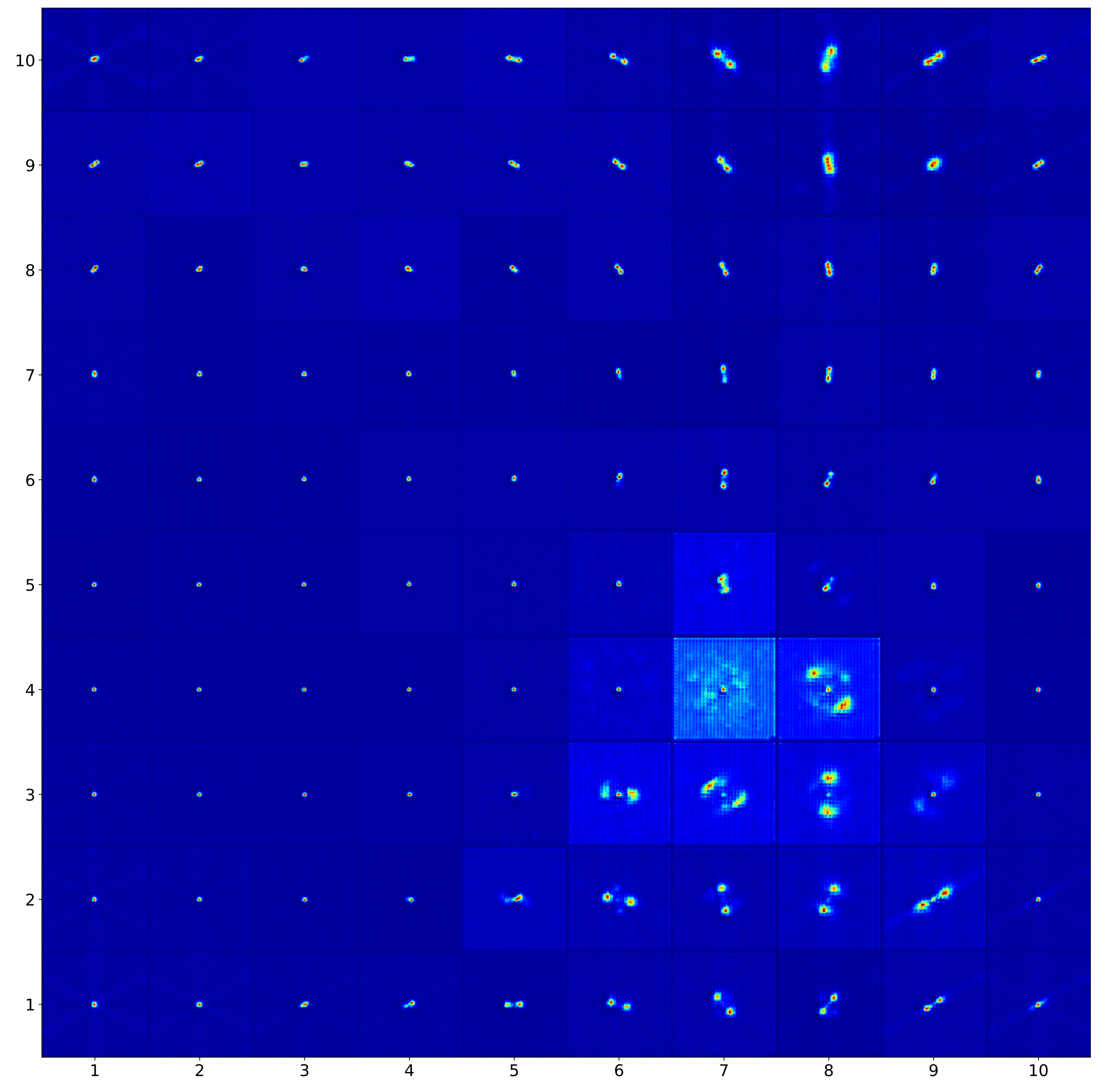

Figure 3.75: 10 × 10 toroidal SOM trained using linear learning rate decay and radius decay and Gaussian radius function, with each neuron displaying false colour decoded neuron weight images

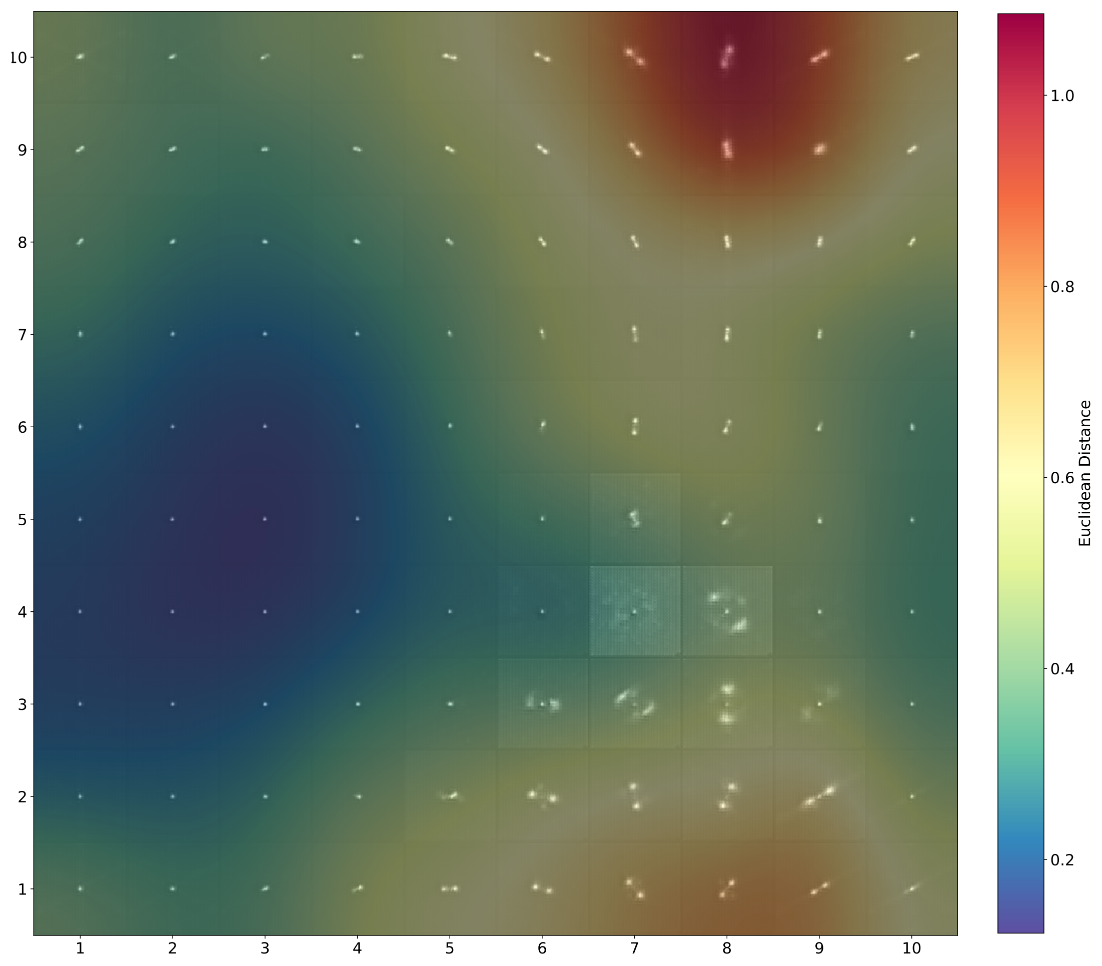

Figure 3.76: 10 × 10 toroidal SOM umat trained using linear learning rate decay and radius decay and Gaussian radius function, with each neuron displaying false colour decoded neuron weight images.

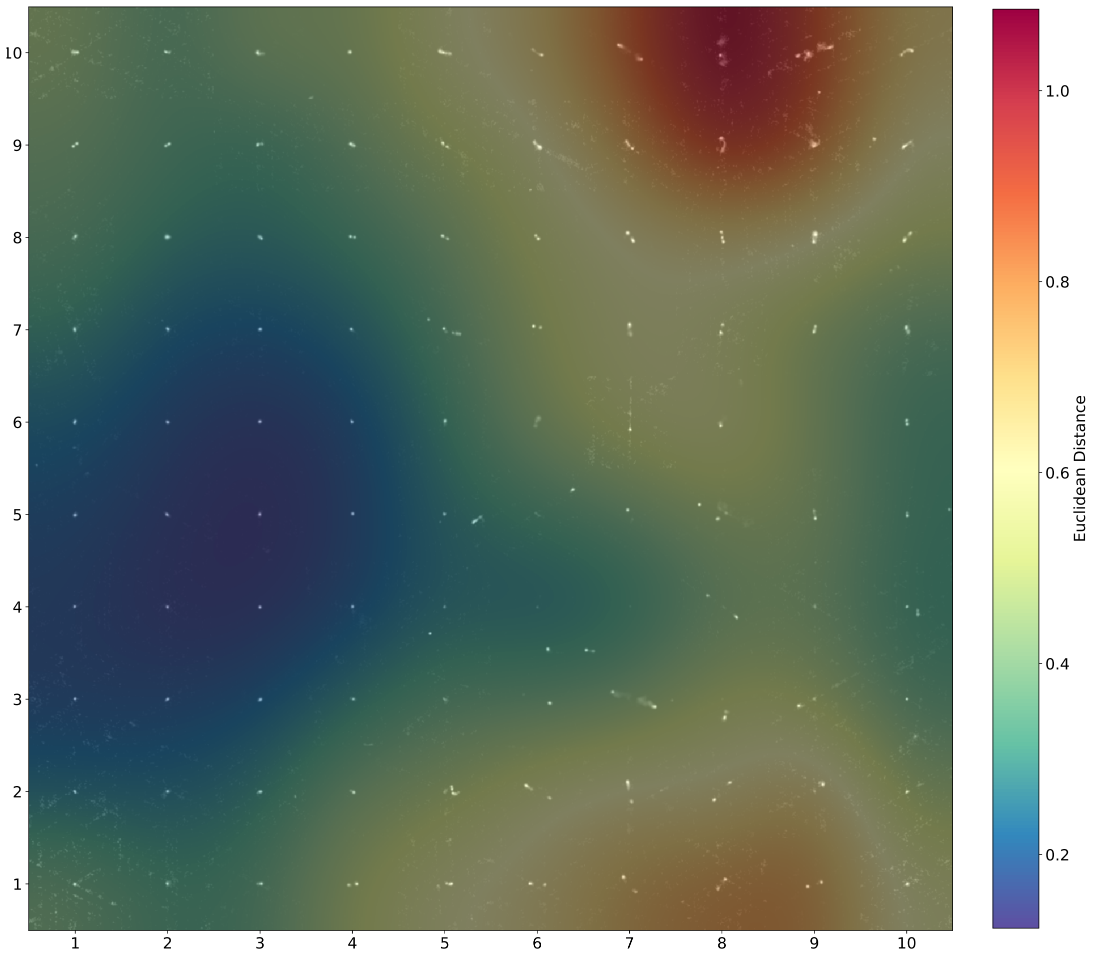

Figure 3.77: 10 × 10 toroidal SOM umat trained using linear learning rate decay, linear learning rate decay and radius decay and Gaussian radius function, with each neuron displaying the RGZ image with the closest matching (Euclidean distance) latent vector transformation to the learned neuron weight.

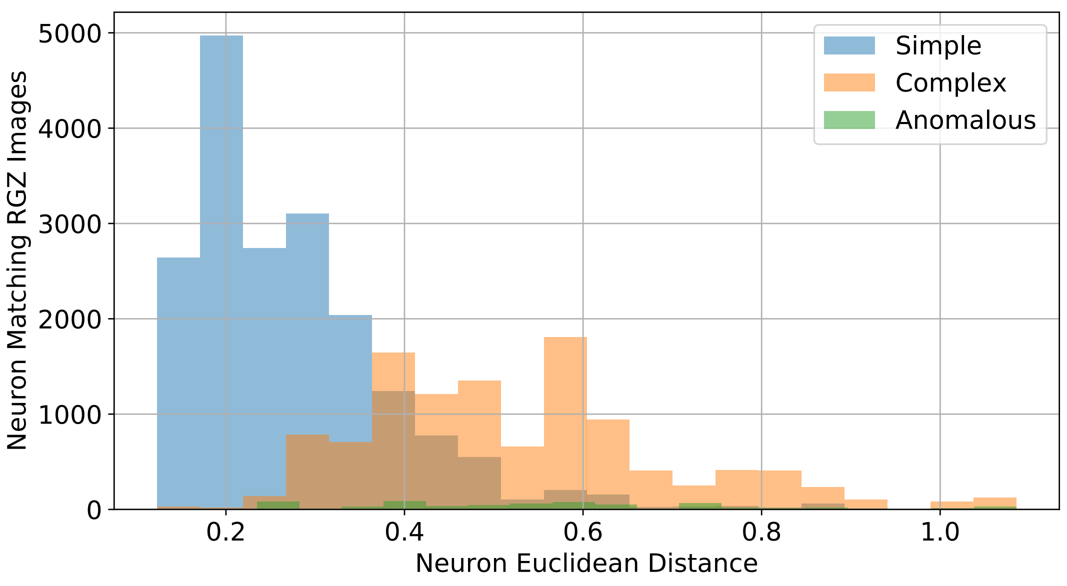

Figure 3.78: Distribution of neuron Euclidean distance from a 10 × 10 toroidal SOM umat trained using linear learning rate decay and radius decay and Gaussian radius function. Colour blending indicates RGZ label derived complexity and anomalousness of all RGZ validation images matching each neuron.

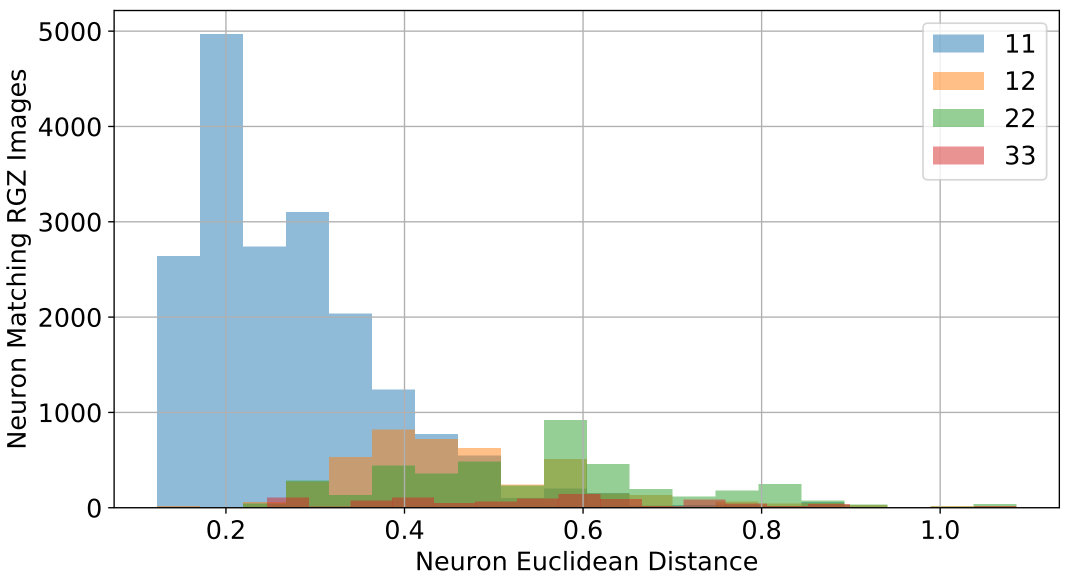

Figure 3.79: Distribution of neuron Euclidean distance from a 10 × 10 toroidal SOM umat trained using linear learning rate decay and radius decay and Gaussian radius function. Colour blending indicates the RGZ label of the four most dominant labelled RGZ validation images matching each neuron.

Table 3.23: Accuracy and performance metrics of anomaly detection by separating neuron matches sources based on the Euclidean distance of neurons a 10×10 toroidal SOM umat at each SD 1-5, trained with a linear learning rate decay and radius decay and Gaussian radius function

10×10 Map Size Neuron Euclidean Distance SD	Euclidean Distance at SD	Accuracy	Precision	Recall	F1 Score
SD1	0.187	0.578	0.471	0.997	0.640
SD2	0.375	**0.833**	0.758	0.818	**0.787**
SD3	0.562	0.758	0.897	0.404	0.557
SD4	0.750	0.664	0.909	0.121	0.213
SD5	0.937	0.630	0.932	0.018	0.036

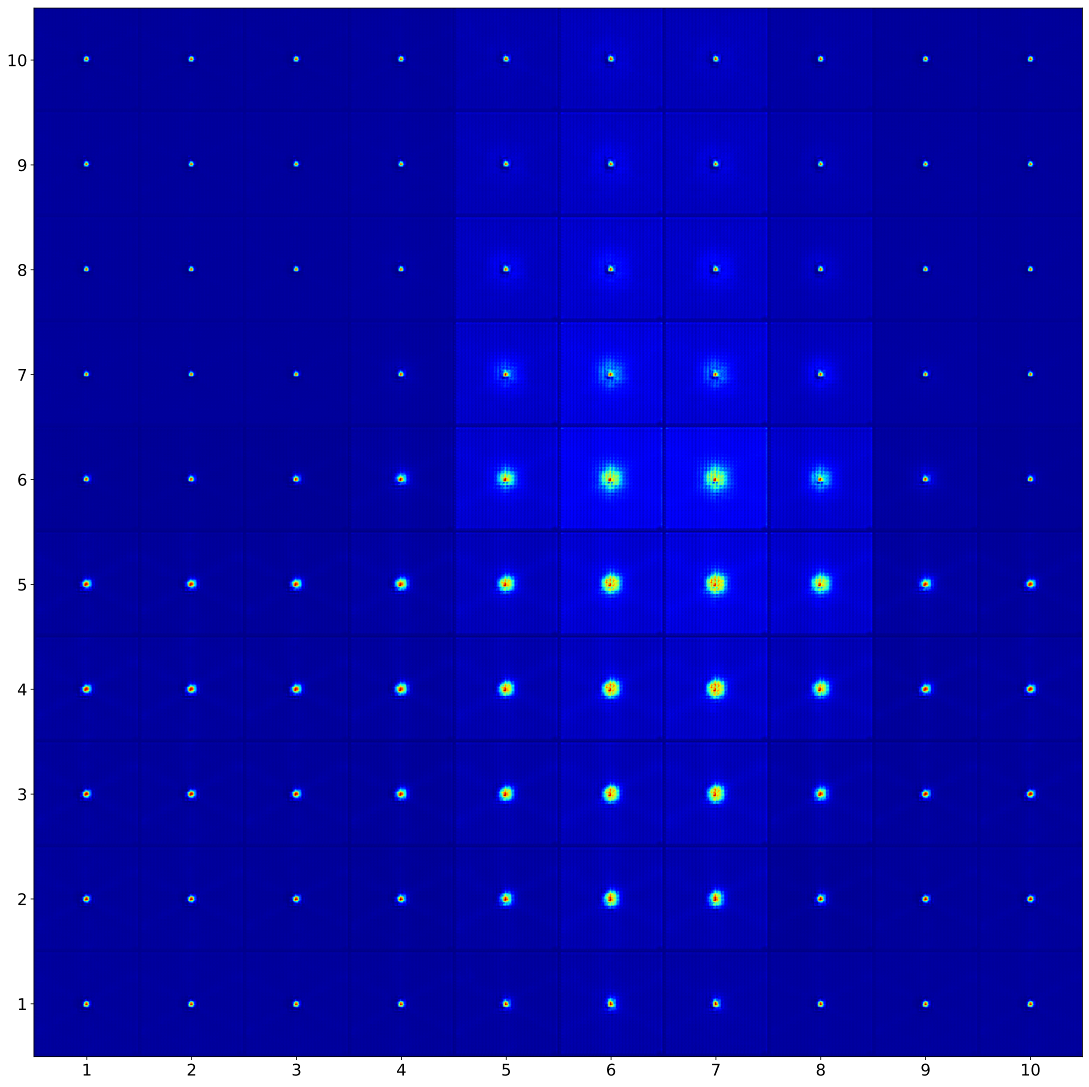

Figure 3.80: 10×10 toroidal SOM trained using linear learning rate decay, radius function and radius decay with each neuron displaying false colour decoded neuron weight images

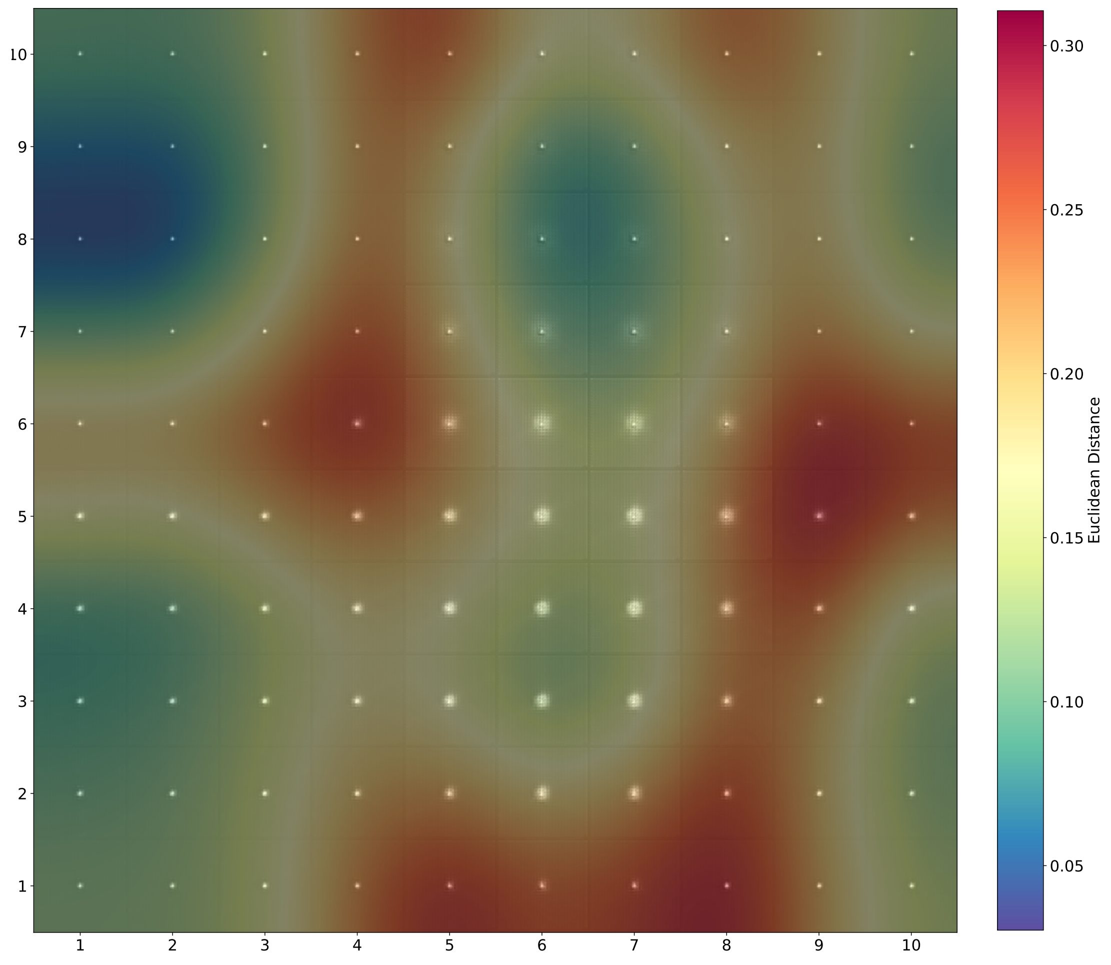

Figure 3.81: 10 × 10 toroidal SOM umat trained using linear learning rate decay, Gaussian radius function and linear radius decay with each neuron displaying false colour decoded neuron weight images.

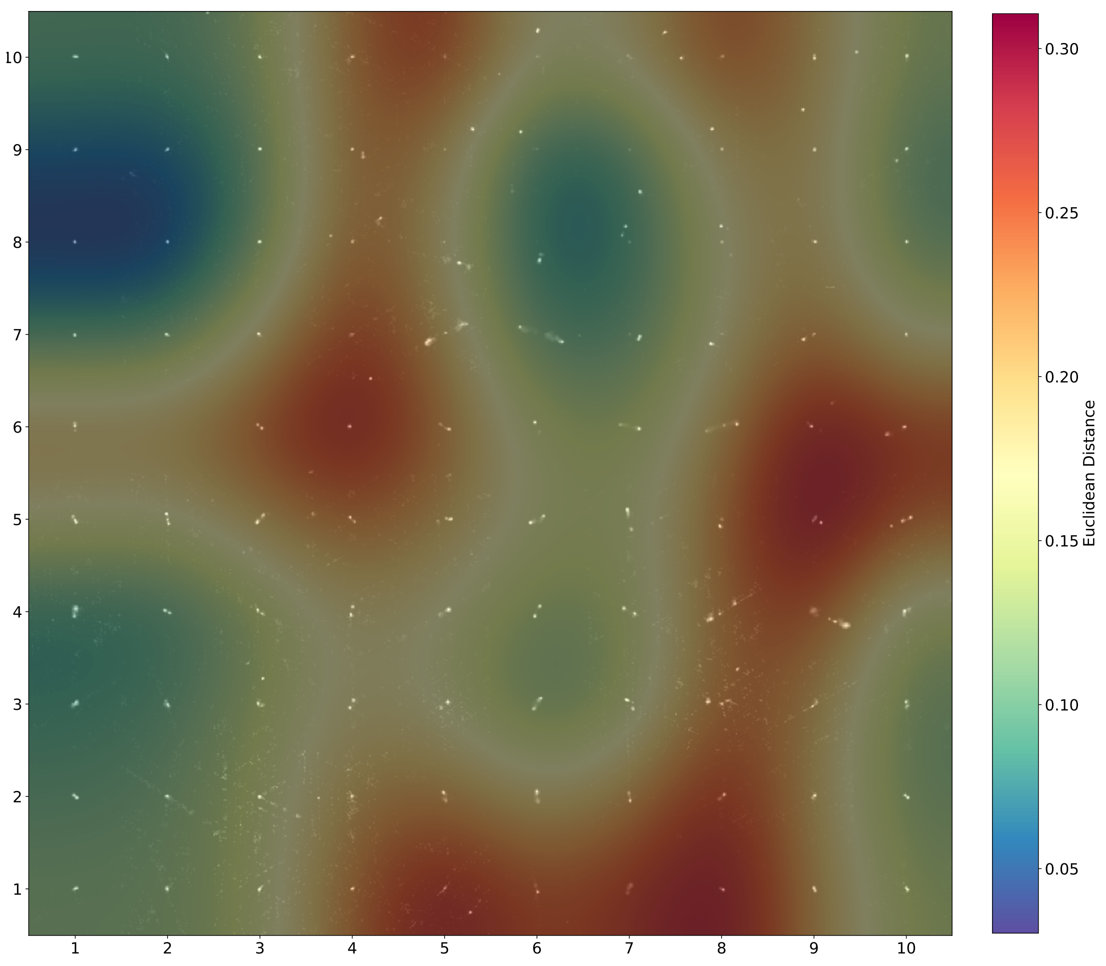

Figure 3.82: 10 × 10 toroidal SOM umat trained using linear learning rate decay, Gaussian radius function and linear radius decay with each neuron displaying the RGZ image with the closest matching (Euclidean distance) latent vector transformation to the learned neuron weight.

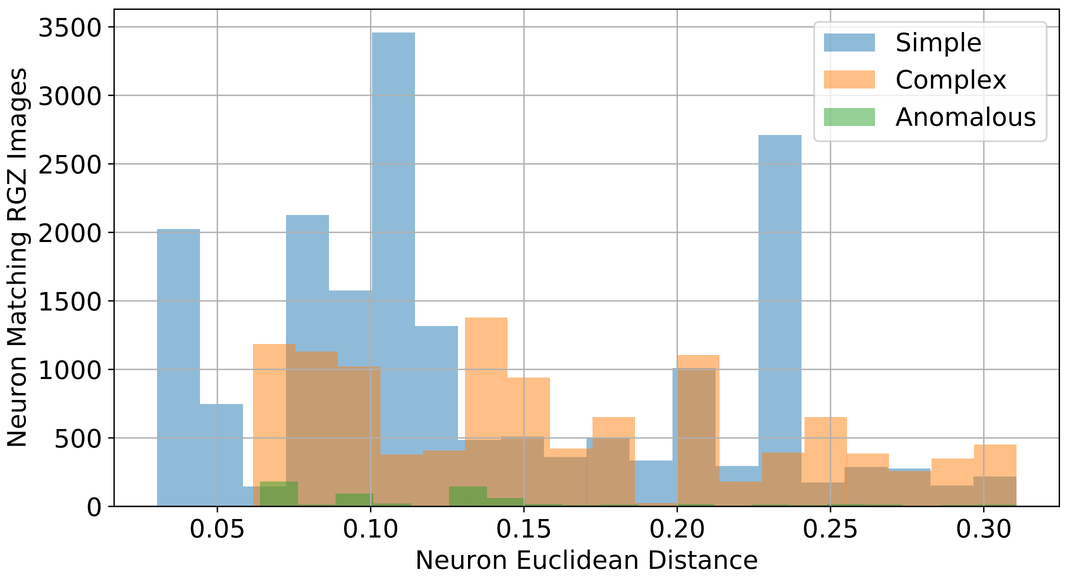

Figure 3.83: Distribution of neuron Euclidean distance from a 10 × 10 toroidal SOM umat trained using linear learning rate decay, Gaussian radius function and linear radius decay. Colour blending indicates RGZ label derived complexity and anomalousness of all RGZ validation images matching each neuron.

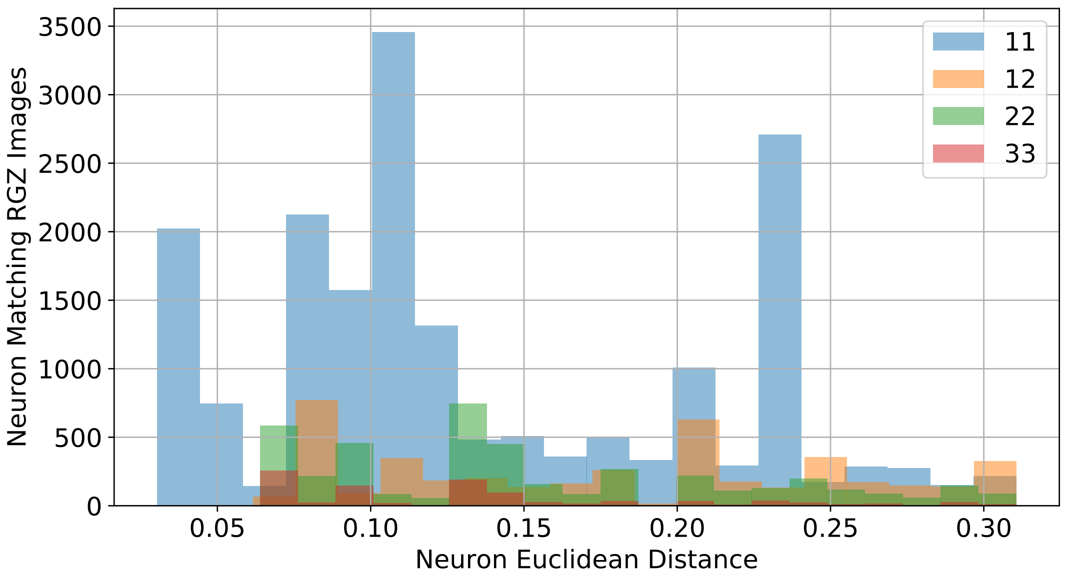

Figure 3.84: Distribution of neuron Euclidean distance from a 10 × 10 toroidal SOM umat trained using linear learning rate decay, Gaussian radius function and linear radius decay. Colour blending indicates the RGZ label of the four most dominant labelled RGZ validation images matching each neuron.

Table 3.24: Accuracy and performance metrics of anomaly detection by separating neuron matches sources based on the Euclidean distance of neurons a 10×10 toroidal SOM umat at each SD 1-5, trained with a linear learning rate decay, Gaussian radius function and linear radius decay

10 × 10 Map Size Neuron Euclidean Distance SD	Euclidean Distance at SD	Accuracy	Precision	Recall	F1 Score
SD1	0.073	0.474	0.409	0.895	**0.562**
SD2	0.146	0.601	0.472	0.513	0.492
SD3	0.219	0.579	0.394	0.220	0.282
SD4	0.292	**0.638**	0.726	0.064	0.118
SD5	0.365	0.623	0.000	0.000	0.000

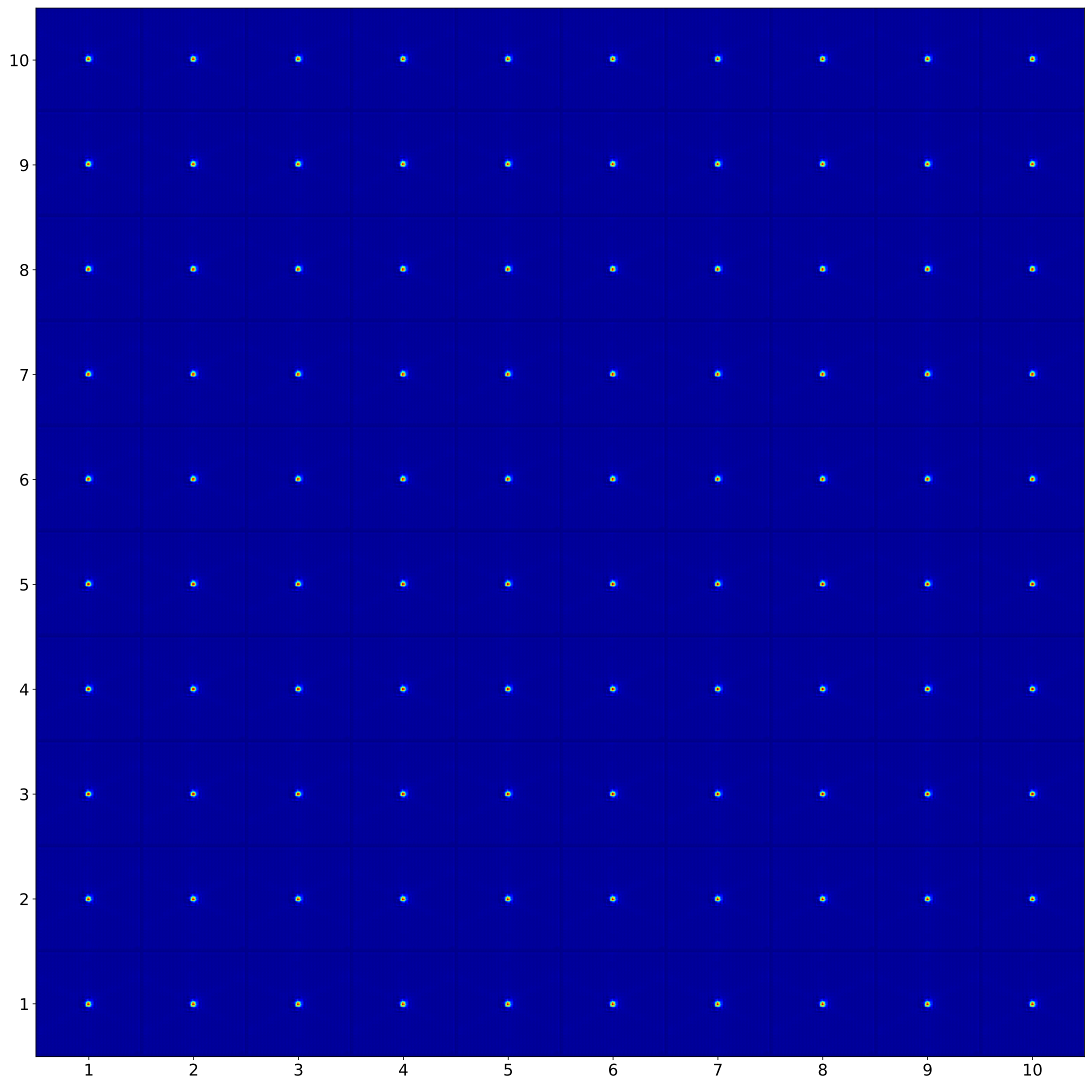

Figure 3.85: 10×10 toroidal SOM trained using linear learning rate decay, radius function and exponential radius decay with each neuron displaying false colour decoded neuron weight images

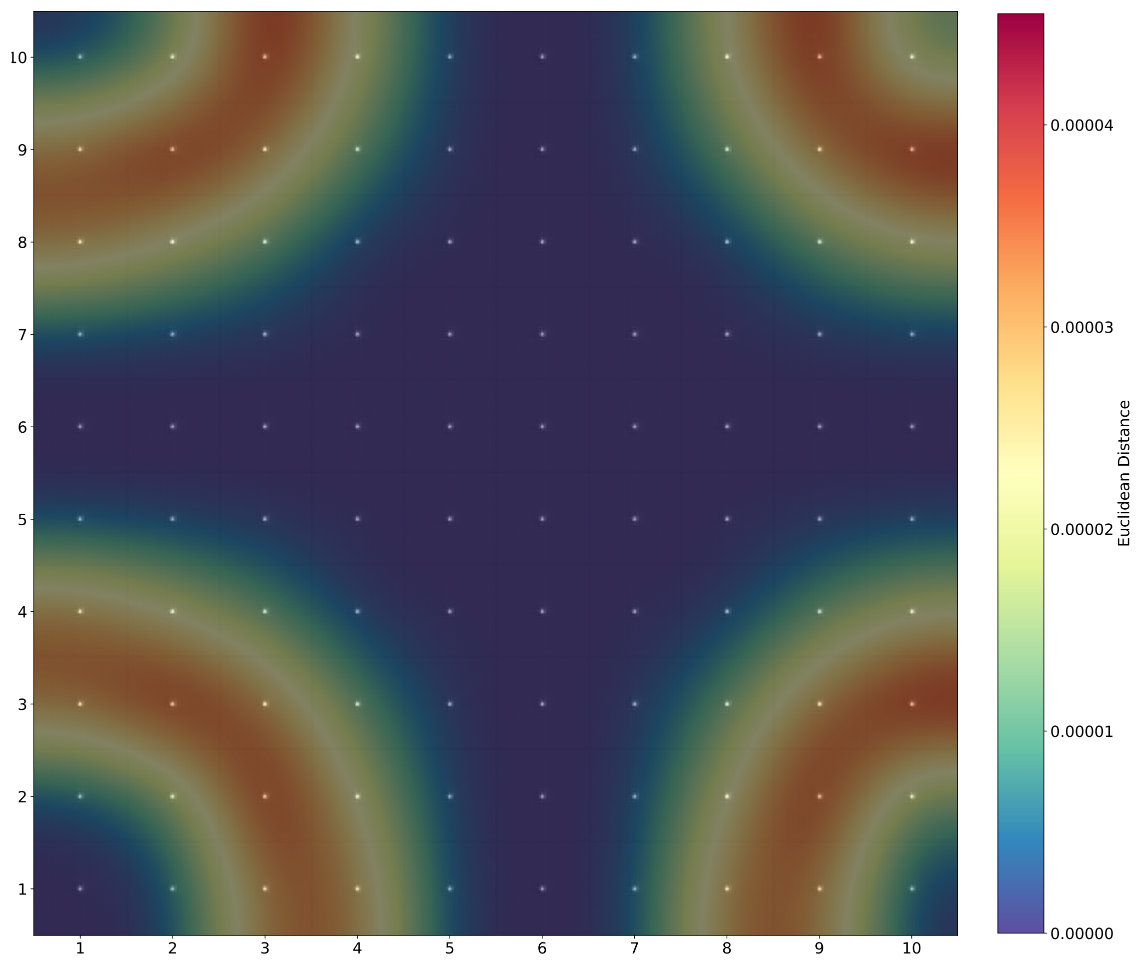

Figure 3.86: 10×10 toroidal SOM umat trained using linear learning rate decay, radius function and exponential radius decay with each neuron displaying false colour decoded neuron weight images.

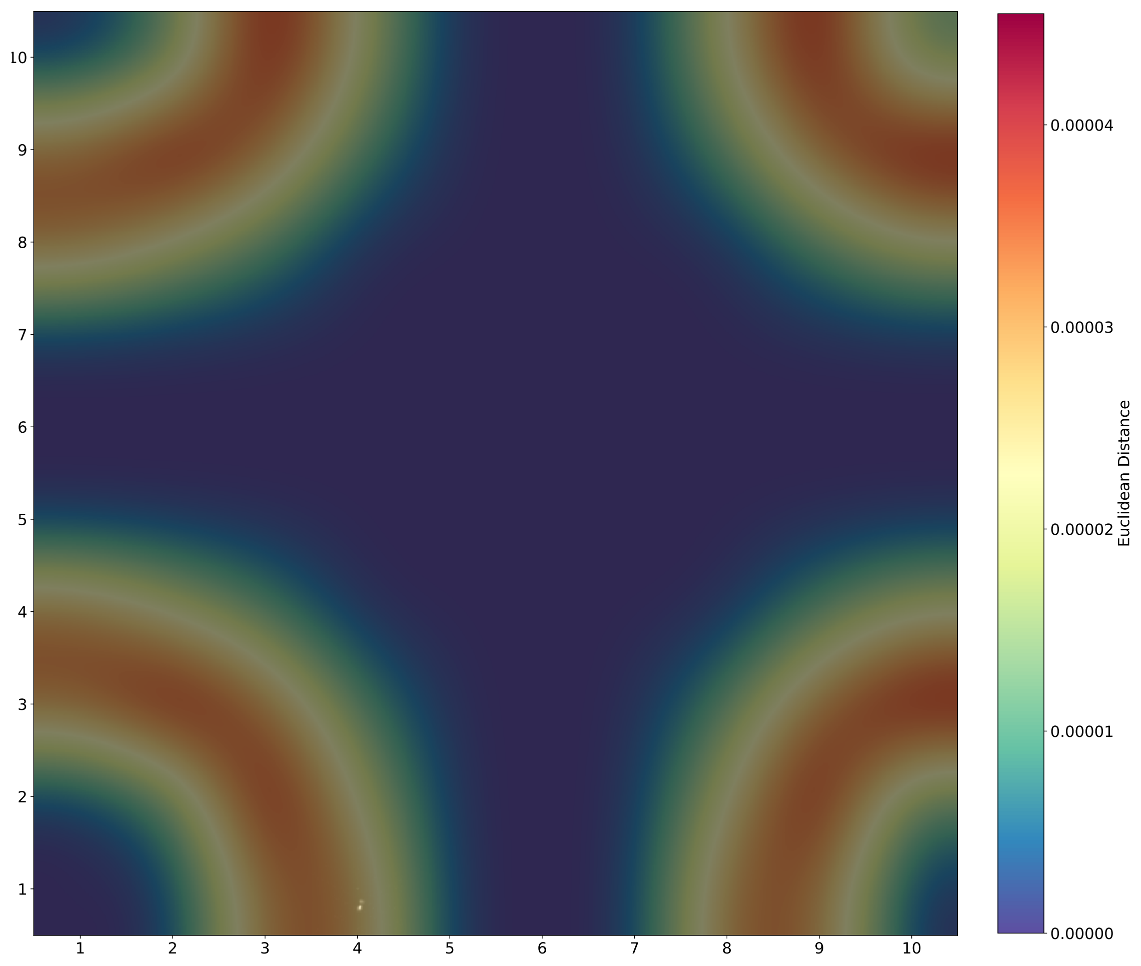

Figure 3.87: 10 × 10 toroidal SOM umat trained using linear learning rate decay, linear radius function and exponential radius decay with each neuron displaying the RGZ image with the closest matching (Euclidean distance) latent vector transformation to the learned neuron weight.

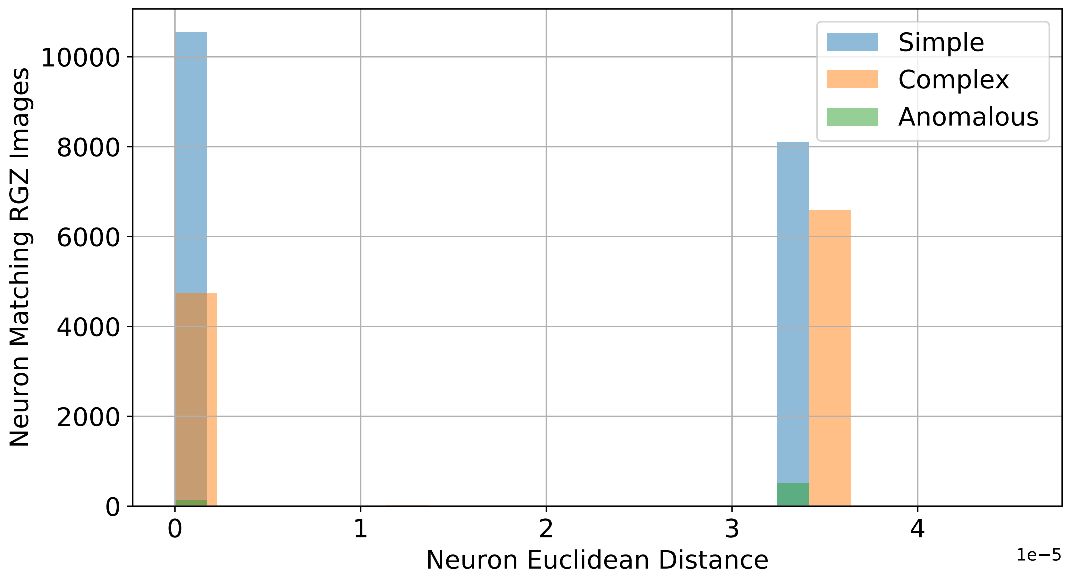

Figure 3.88: Distribution of neuron Euclidean distance from a 10 × 10 toroidal SOM umat trained using linear learning rate decay, radius function and exponential radius decay. Colour blending indicates RGZ label derived complexity and anomalousness of all RGZ validation images matching each neuron.

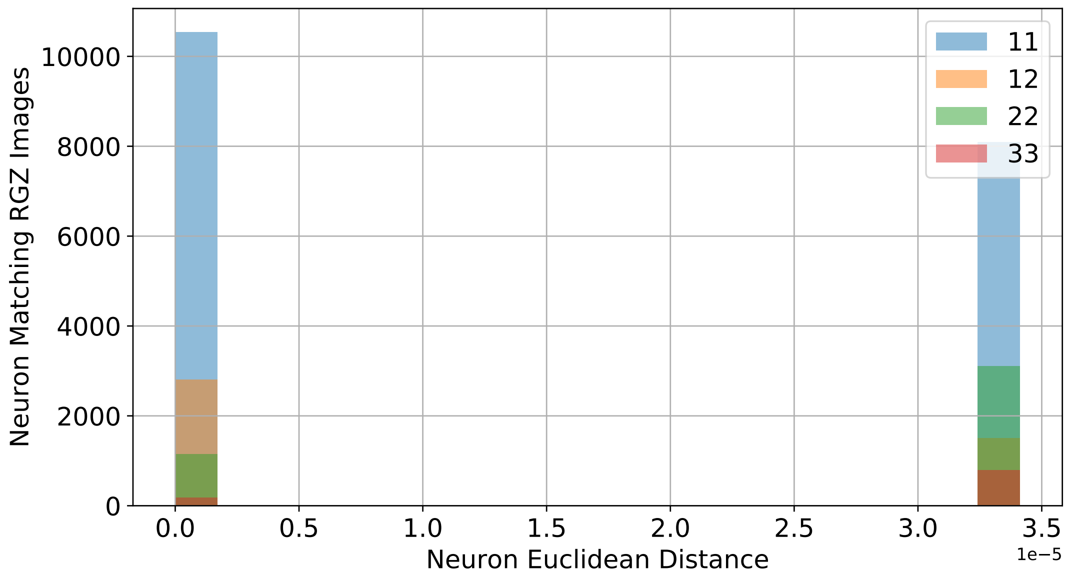

Figure 3.89: Distribution of neuron Euclidean distance from a 10 × 10 toroidal SOM umat trained using linear learning rate decay, radius function and exponential radius decay. Colour blending indicates the RGZ label of the four most dominant labelled RGZ validation images matching each neuron.

Table 3.25: Accuracy and performance metrics of anomaly detection by separating neuron matches sources based on the Euclidean distance of neurons a 10×10 toroidal SOM umat at each SD 1-5, trained with a linear learning rate decay, radius function and exponential radius decay

10×10 Map Size Neuron Euclidean Distance SD	Euclidean Distance at SD	Accuracy	Precision	Recall	F1 Score
SD1	0.0	0.571	0.449	0.581	**0.507**
SD2	0.0	0.571	0.449	0.581	0.506
SD3	0.0	0.621	0.000	0.000	0.000
SD4	0.0	0.621	0.000	0.000	0.000
SD5	0.0	0.621	0.000	0.000	0.000

Table 3.26: Processing time metrics of each toroidal SOM with square 10×10 sized maps trained with combinations of neighbourhood and radius decay functions

Neighbourhood Type	Radius Decay Function	Total Training Time (s)	Average Training Time Per Epoch (s)
Gaussian	Linear	8.571	0.714
Gaussian	Exponential	8.399	0.7
Linear	Exponential	8.058	0.671
Linear	Linear	8.272	0.689

Table 3.27: Accuracy metrics and comparisons of each toroidal SOM with square 10×10 sized maps trained with combinations of neighbourhood and radius decay functions

Neighbourhood and Radius Decay Function	SD and Euclidean Distance	Accuracy	Precision	Recall	F1 Score
Gaussian and Linear	SD2: 0.375	**0.833**	0.758	0.818	**0.787**
Gaussian and Exponential	SD1: 0.073	0.474	0.409	0.895	0.562
Linear and Exponential	SD1: 0.0	0.571	0.449	0.581	0.507
Linear and Linear	SD2: 0.262	0.734	0.590	0.961	0.731

3.3.3.4 Effects of Rotation Dataset Augmentations in Autoencoder Training

In this Section, I tested the effects of training a SOM on the latent vectors of radio astronomical images produced from an autoencoder trained with random rotation dataset augmentations. I conducted these tests using the ideal configurations for the SOM with linear learning rate and radius decay functions and a Gaussian neighbourhood. The results were evaluated similarly to previous tests by morphology and the accuracy of the complexity separation. I also compared these results to the previous

SOM results without augmentations.

The results show that the Gaussian neighbourhood type SOM trained with latent vectors from an autoencoder trained with random rotations produces in larger map sizes produces greater accuracy and F1 scores compared regular trained latent vector training, as summarised in Table 3.31. However, the un-augmented latent vector training produces a consistently higher and consistent F1 score. Additionally, the training time required in both test are almost identical, as shown in Table 3.32. These tests suggest that these different latent vector encodings do not significantly affect the performance of the complexity separation and anomaly detection. The effects of this training are significantly more pronounced when examining the learned morphologies of the map, however.

Despite training only on four different random rotation angles (the rotation augmentation trained autoencoder converged at 4 epochs, detailed in Section 3.2.2.5), the SOM produces maps with weights that appear to have learned sufficient rotational features to remove most of the learned 'rotated average' neuron weights, as seen in all previous decoded neuron test outputs. In these tests however, the SOM trained on rotation trained autoencoder latent vectors displays neuron weights removes most of these rotated features, where only some neurons in larger 20×20 and 40×40 maps such as Figures 3.95 and 3.100, may show an extra rotation in the neuron weight near the location of typical rotated average weights as shown in Figures 3.60 and 3.65. These extra rotations appear as radio-doubles showing additional symmetrically rotated peaks or AGN with extra symmetrically rotated jets. Despite these differences, the same absence of rotated average neuron weights is seen regardless of map size.

Training with these latent vectors appears to produce better defined rotated neuron weights which can be seen in the RGZ image matched umat, where the rotation angle of neuron radio sources is clearly better defined and arranged. As a result, the distribution of neuron Euclidean distance appears more defined in these tests compared to regular latent vector training. As in previous map size tests, increased SOM map sizes result in a smoother evolution of morphologies and a more defined neuron Euclidean distance distribution.

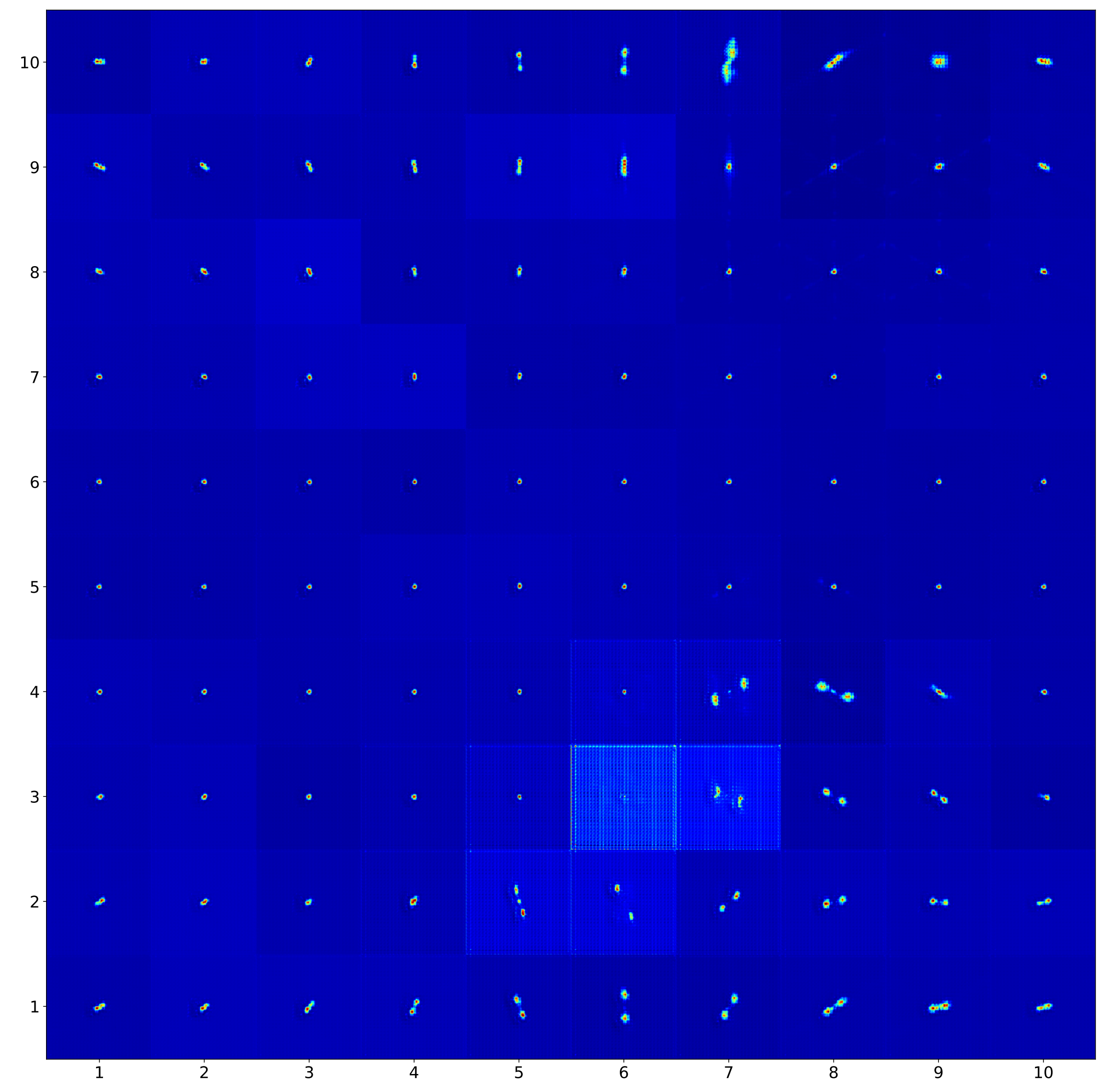

Figure 3.90: 10×10 toroidal SOM with linear learning rate decay and radius decay and Gaussian radius function using RGZ latent vectors produced using an autoencoder with random rotation augmentation training, with decoded neuron weights

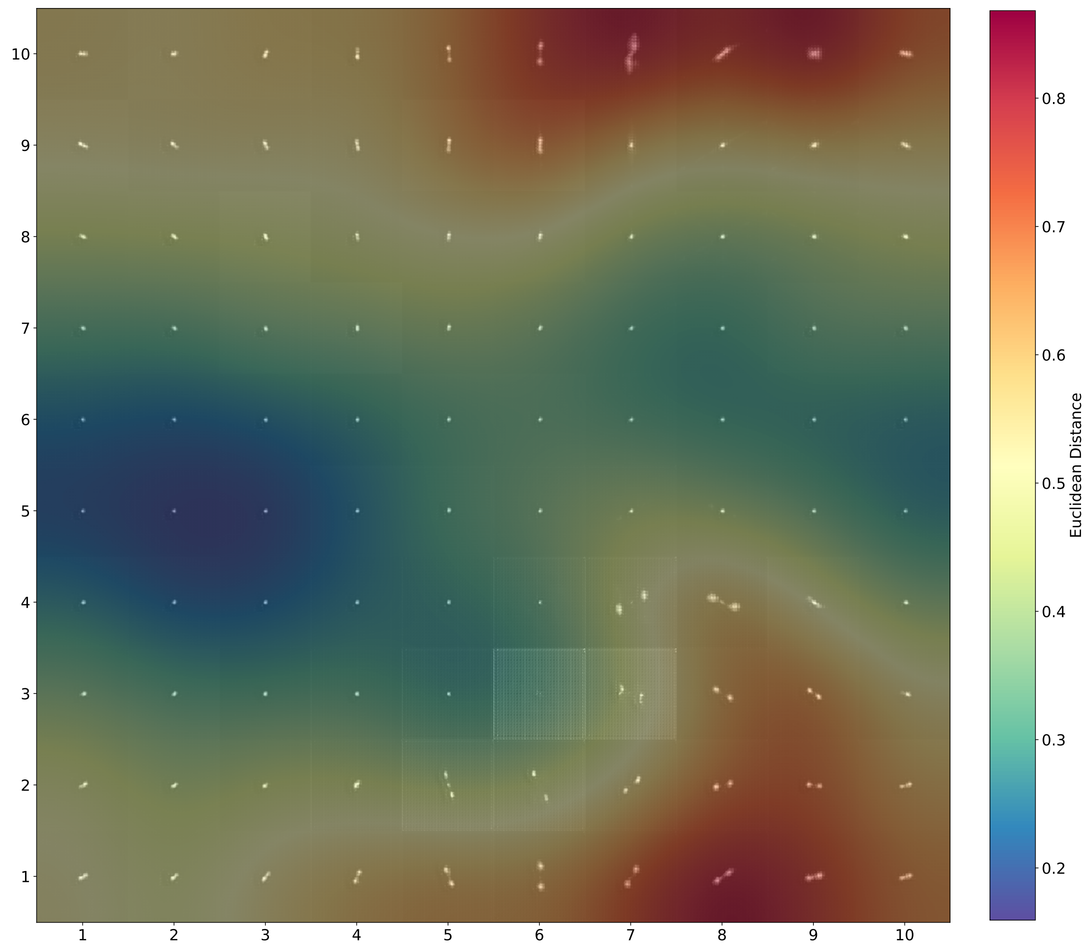

Figure 3.91: 10 × 10 toroidal SOM umat trained with linear learning rate decay and radius decay and Gaussian radius function using RGZ latent vectors produced using an autoencoder with random rotation augmentation training, with decoded neuron weights.

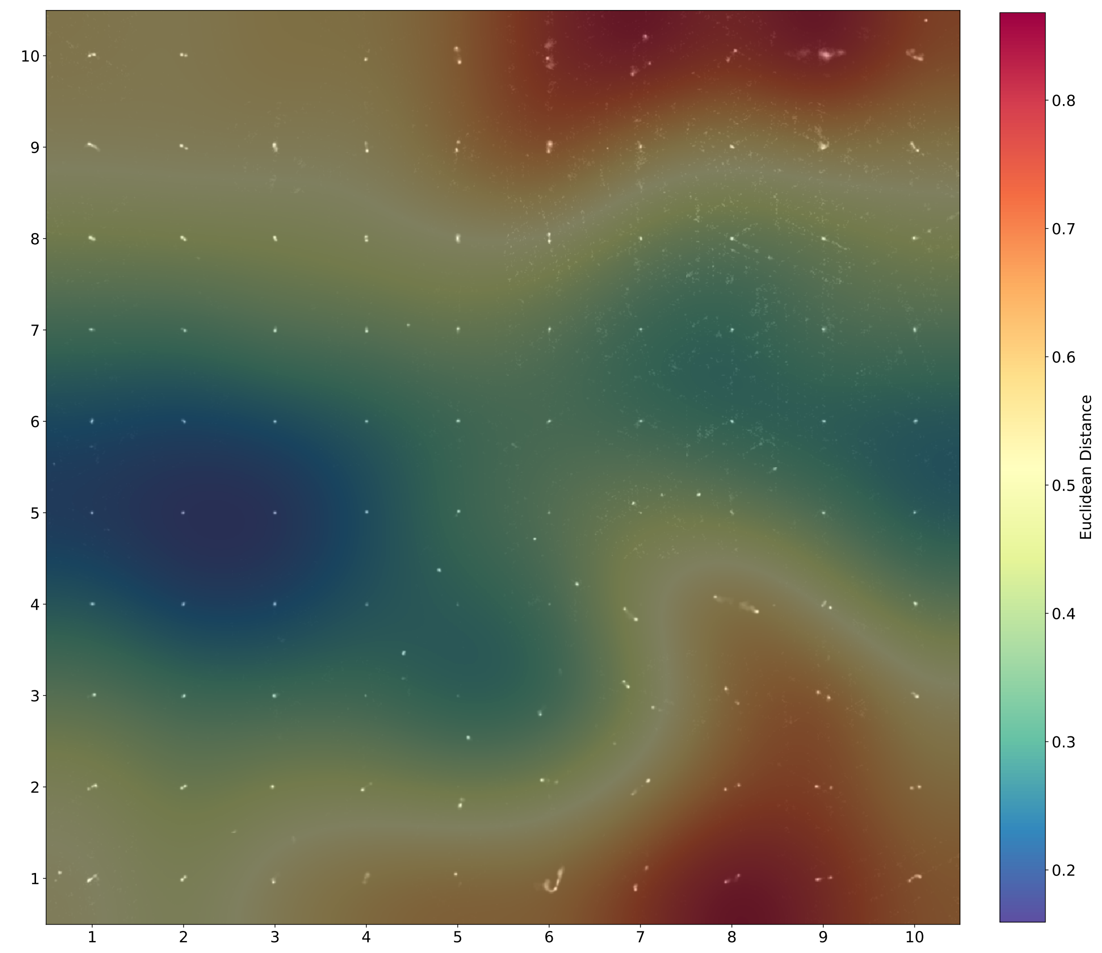

Figure 3.92: 10 × 10 toroidal SOM umat trained using linear learning rate decay, linear learning rate decay and radius decay and Gaussian radius function using RGZ latent vectors produced using an autoencoder with random rotation augmentation training, with each neuron displaying the RGZ image with the closest matching (Euclidean distance) latent vector transformation to the learned neuron weight.

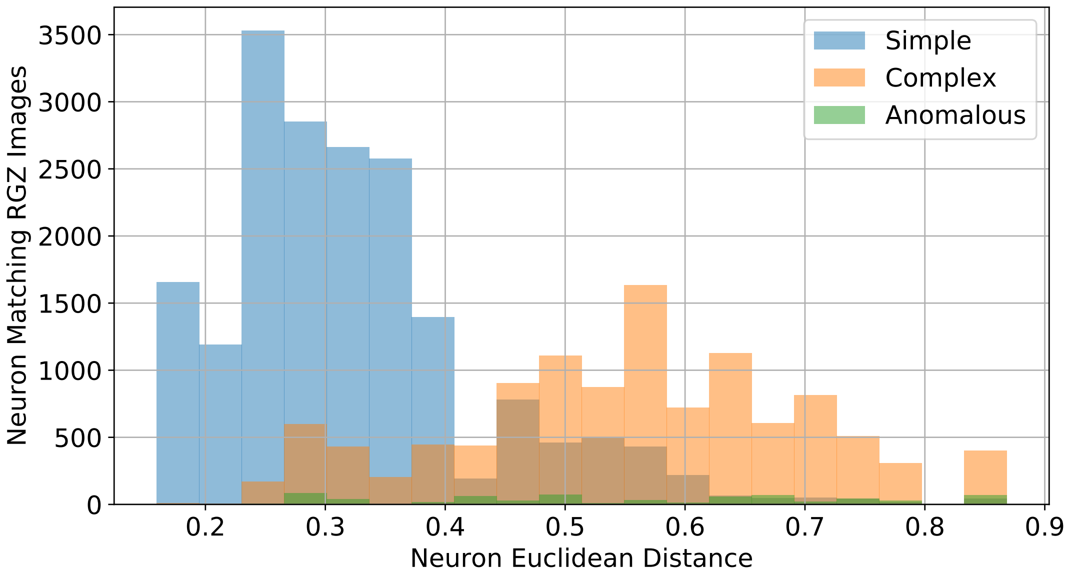

Figure 3.93: Distribution of neuron Euclidean distance from a 10 × 10 toroidal SOM umat trained using linear learning rate decay and radius decay and Gaussian radius function using RGZ latent vectors produced using an autoencoder with random rotation augmentation training. Colour blending indicates RGZ label derived complexity and anomalousness of all RGZ validation images matching each neuron.

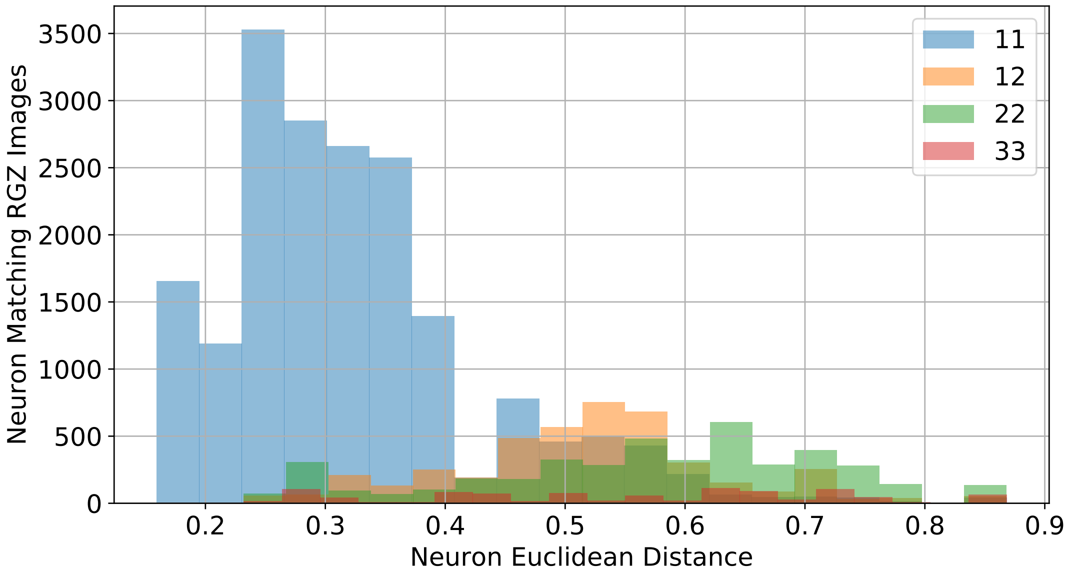

Figure 3.94: Distribution of neuron Euclidean distance from a 10 × 10 toroidal SOM umat trained using linear learning rate decay and radius decay and Gaussian radius function using RGZ latent vectors produced using an autoencoder with random rotation augmentation training. Colour blending indicates the RGZ label of the four most dominant labelled RGZ validation images matching each neuron.

Table 3.28: Accuracy and performance metrics of anomaly detection by separating neuron matches sources based on the Euclidean distance of neurons a 10×10 toroidal SOM umat at each SD 1-5, trained with a linear learning rate decay and radius decay and Gaussian radius function using RGZ latent vectors produced using an autoencoder with random rotation augmentation training

10 × 10 Map Size Neuron Euclidean Distance SD	Euclidean Distance at SD	Accuracy	Precision	Recall	F1 Score
SD1	0.169	0.416	0.392	0.999	0.563
SD2	0.337	0.733	0.597	0.893	0.715
SD3	0.506	**0.812**	0.816	0.648	**0.722**
SD4	0.674	0.693	0.931	0.200	0.329
SD5	0.843	0.635	0.901	0.035	0.068

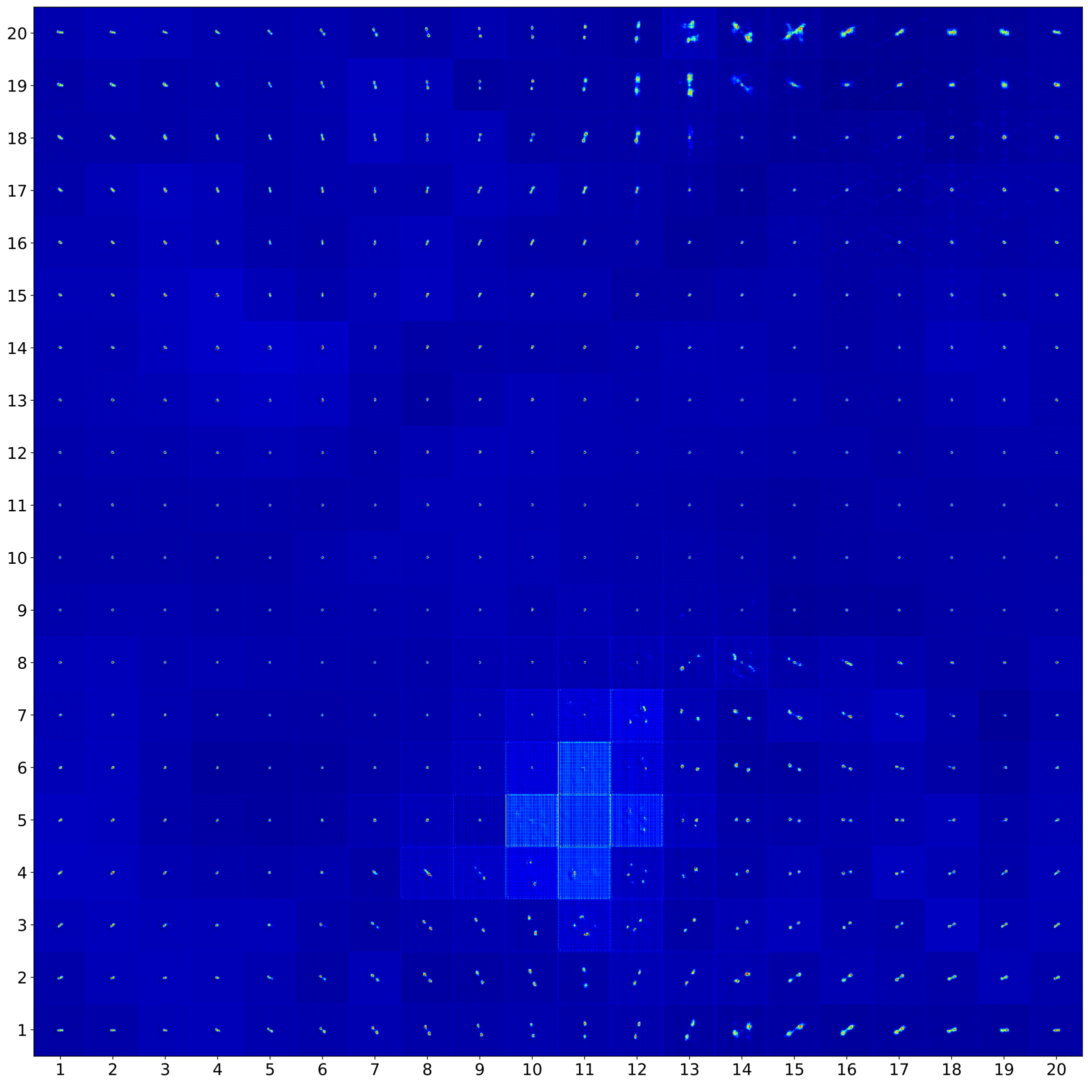

Figure 3.95: 20×20 toroidal SOM with linear learning rate decay and radius decay and Gaussian radius function using RGZ latent vectors produced using an autoencoder with random rotation augmentation training, with decoded neuron weights

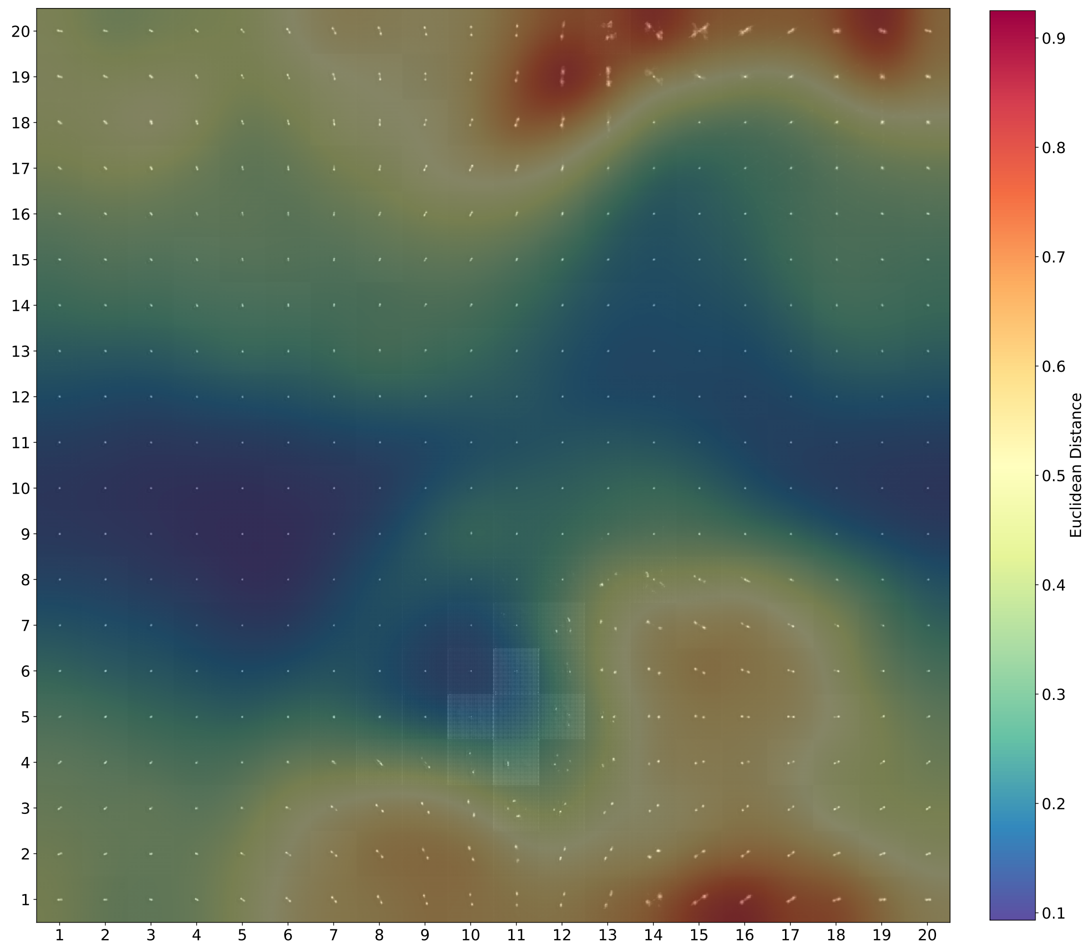

Figure 3.96: 20 × 20 toroidal SOM umat trained with linear learning rate decay and radius decay and Gaussian radius function using RGZ latent vectors produced using an autoencoder with random rotation augmentation training, with decoded neuron weights.

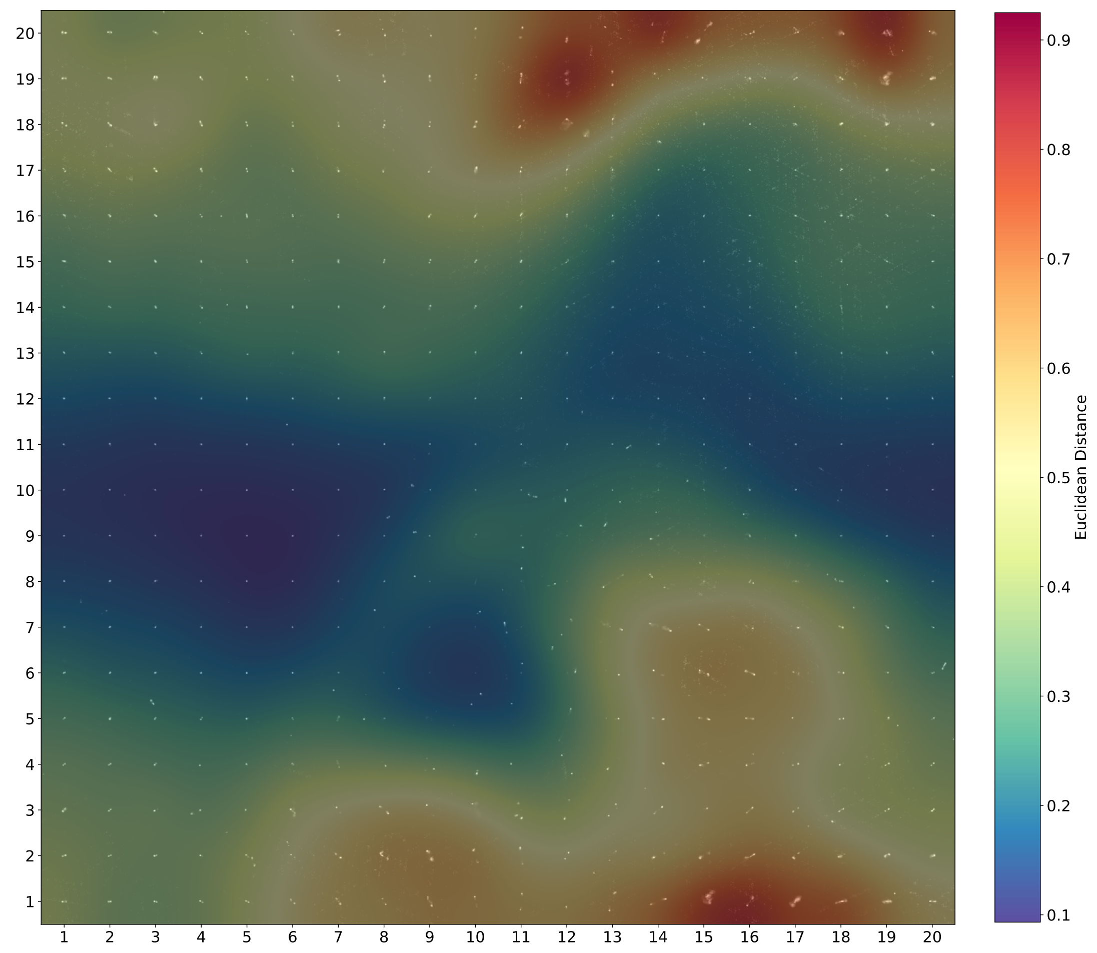

Figure 3.97: 20 × 20 toroidal SOM umat trained using linear learning rate decay, linear learning rate decay and radius decay and Gaussian radius function using RGZ latent vectors produced using an autoencoder with random rotation augmentation training, with each neuron displaying the RGZ image with the closest matching (Euclidean distance) latent vector transformation to the learned neuron weight.

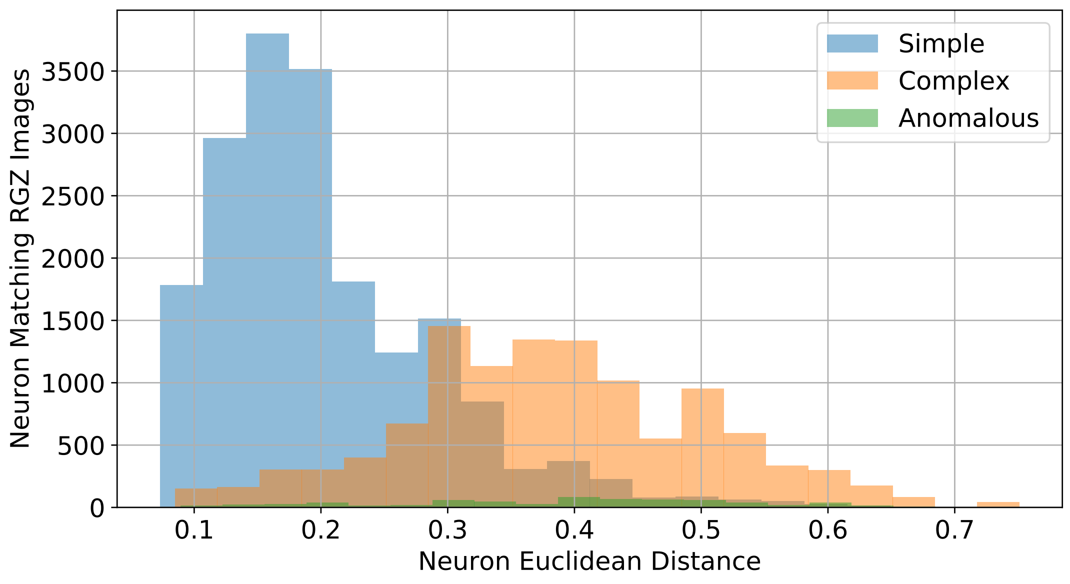

Figure 3.98: Distribution of neuron Euclidean distance from a 20 × 20 toroidal SOM umat trained using linear learning rate decay and radius decay and Gaussian radius function using RGZ latent vectors produced using an autoencoder with random rotation augmentation training. Colour blending indicates RGZ label derived complexity and anomalousness of all RGZ validation images matching each neuron.

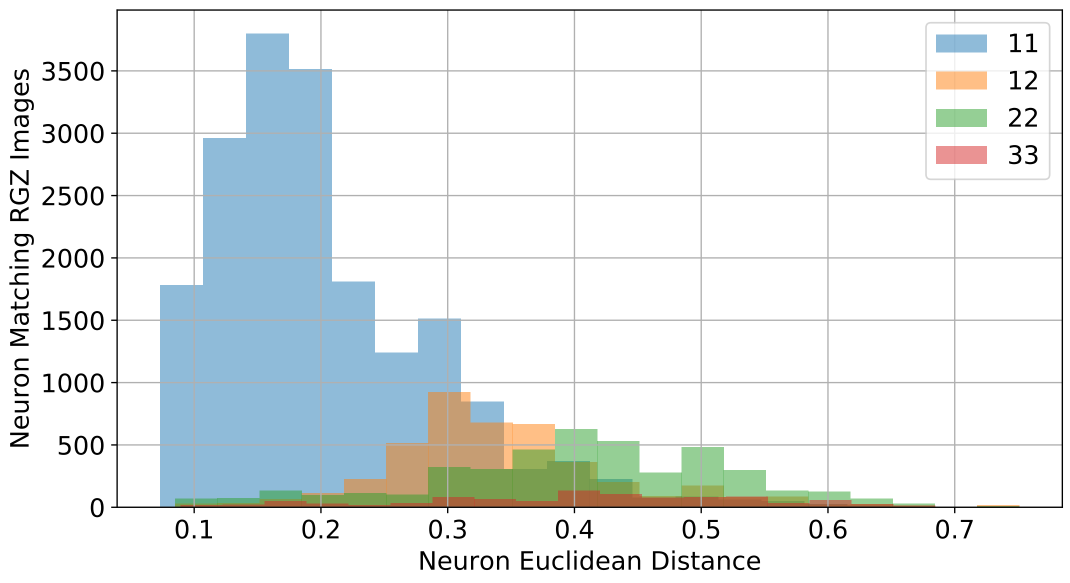

Figure 3.99: Distribution of neuron Euclidean distance from a 20 × 20 toroidal SOM umat trained using linear learning rate decay and radius decay and Gaussian radius function using RGZ latent vectors produced using an autoencoder with random rotation augmentation training. Colour blending indicates the RGZ label of the four most dominant labelled RGZ validation images matching each neuron.

Table 3.29: Accuracy and performance metrics of anomaly detection by separating neuron matches sources based on the Euclidean distance of neurons a 20 × 20 toroidal SOM umat at each SD 1-5, trained with a linear learning rate decay and radius decay and Gaussian radius function using RGZ latent vectors produced using an autoencoder with random rotation augmentation training

10 × 10 Map Size Neuron Euclidean Distance SD	Euclidean Distance at SD	Accuracy	Precision	Recall	F1 Score
SD1	0.134	0.513	0.435	0.980	0.603
SD2	0.268	**0.823**	0.721	0.865	**0.787**
SD3	0.402	0.762	0.872	0.430	0.576
SD4	0.536	0.658	0.911	0.102	0.183
SD5	0.671	0.625	0.917	0.004	0.008

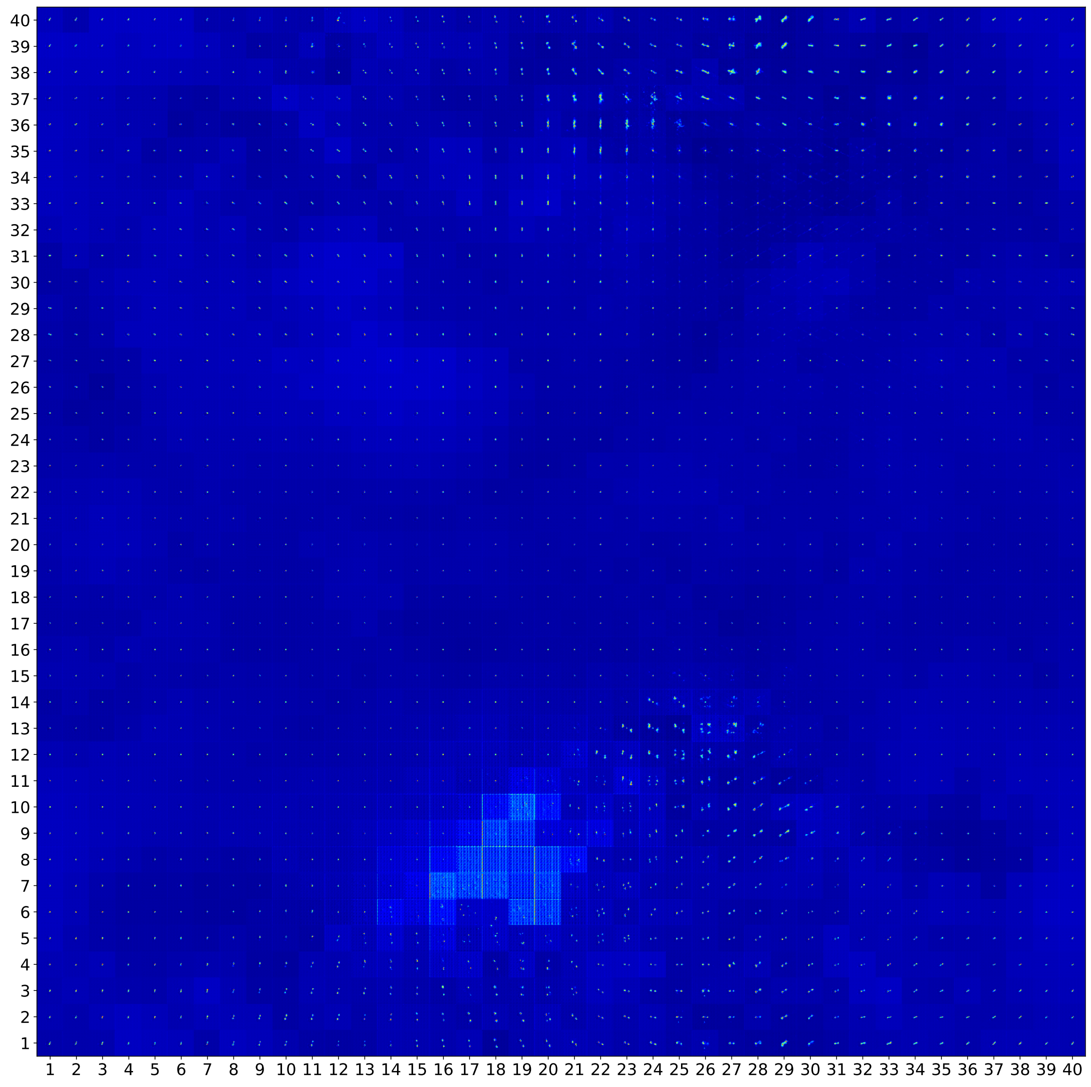

Figure 3.100: 40×40 toroidal SOM with linear learning rate decay and radius decay and Gaussian radius function using RGZ latent vectors produced using an autoencoder with random rotation augmentation training, with decoded neuron weights

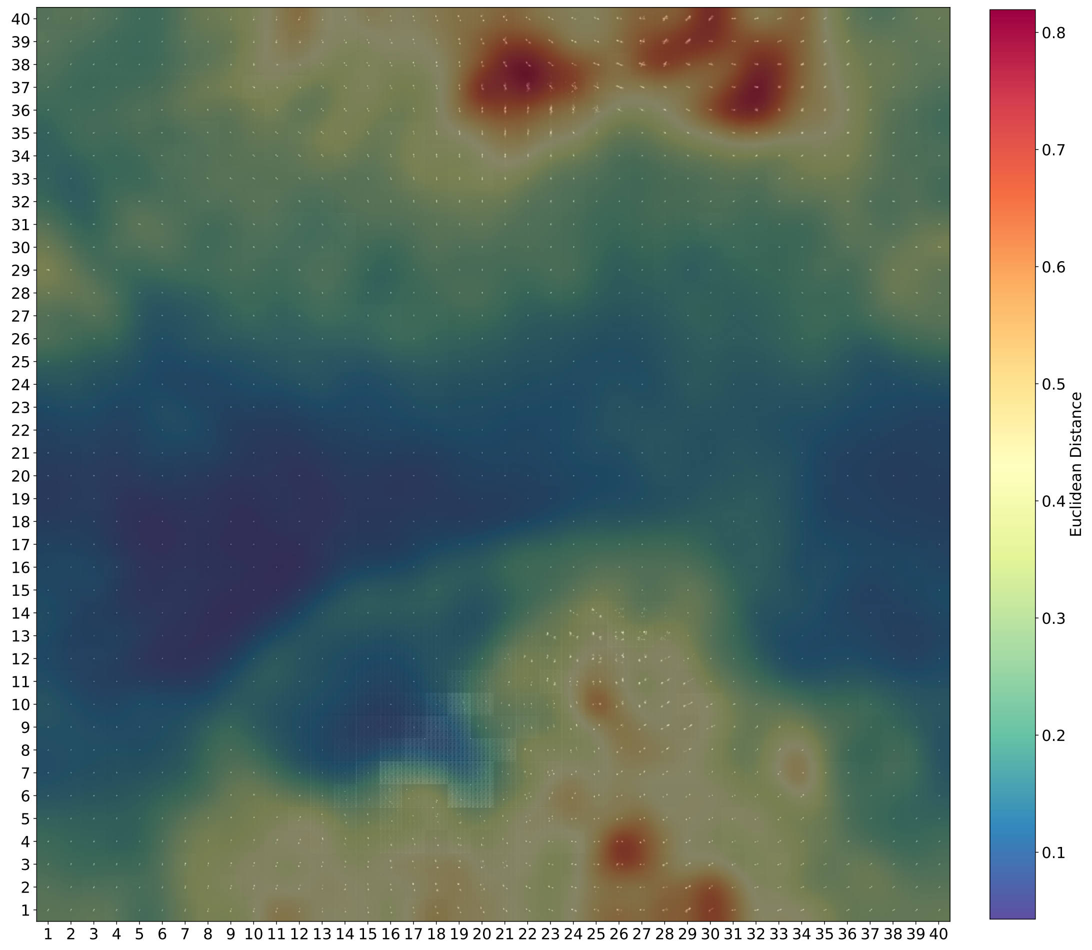

Figure 3.101: 40 × 40 toroidal SOM umat trained with linear learning rate decay and radius decay and Gaussian radius function using RGZ latent vectors produced using an autoencoder with random rotation augmentation training, with decoded neuron weights.

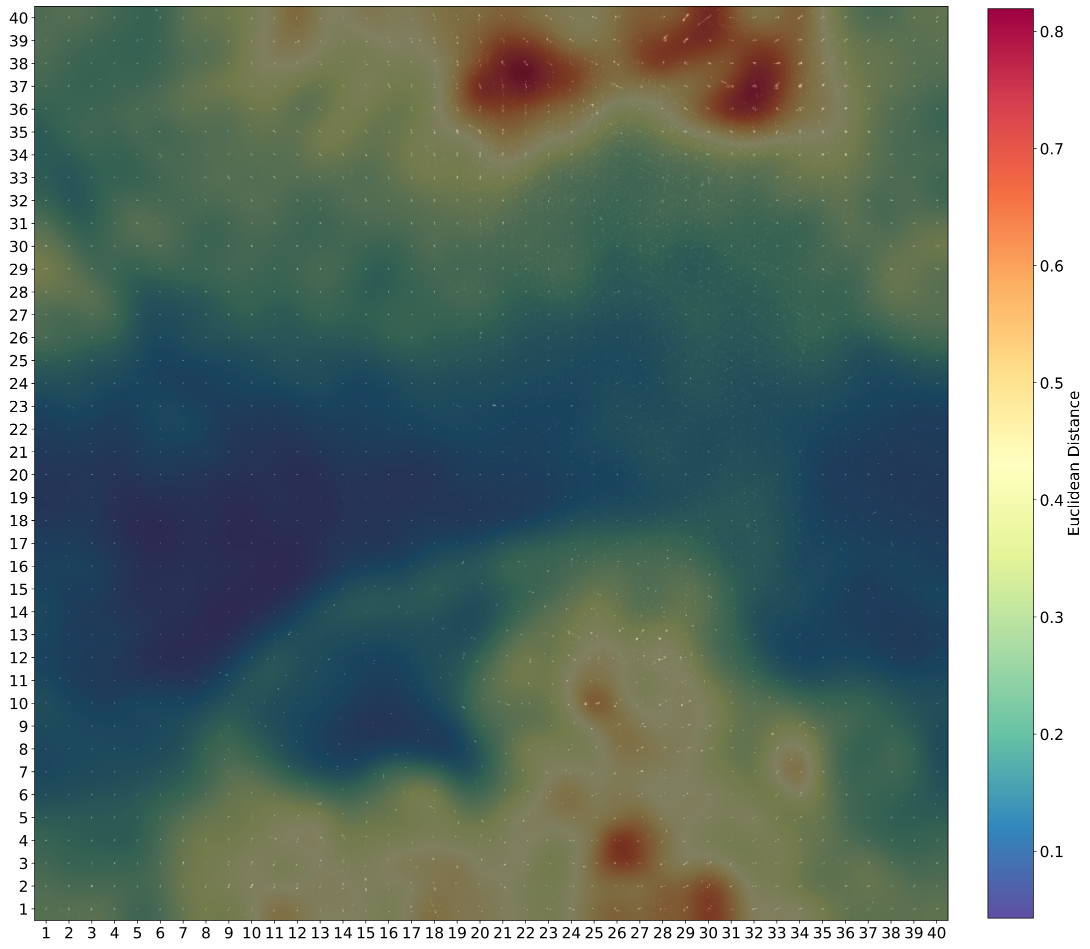

Figure 3.102: 40×40 toroidal SOM umat trained using linear learning rate decay, linear learning rate decay and radius decay and Gaussian radius function using RGZ latent vectors produced using an autoencoder with random rotation augmentation training, with each neuron displaying the RGZ image with the closest matching (Euclidean distance) latent vector transformation to the learned neuron weight.

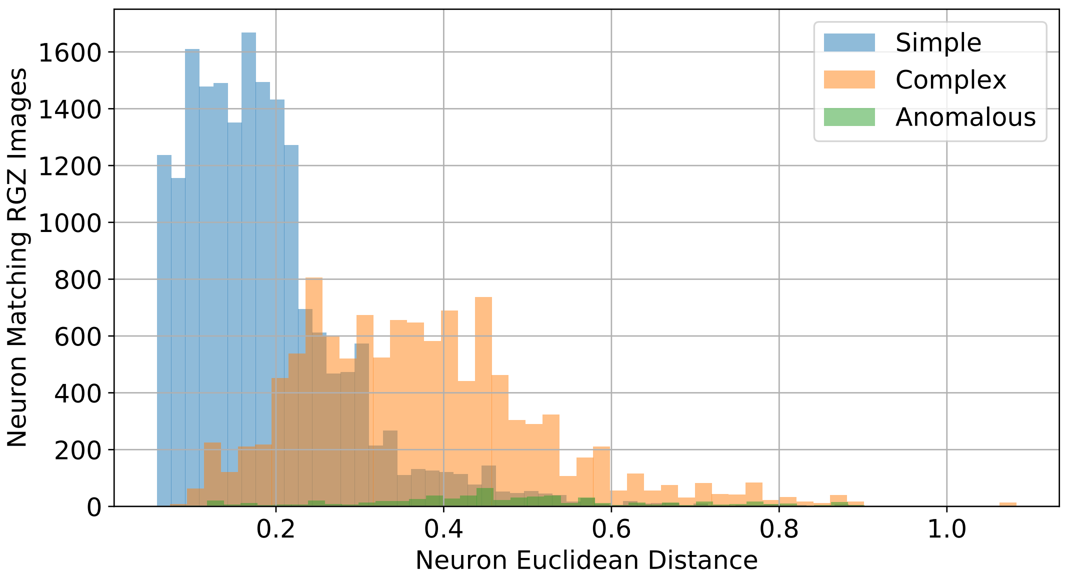

Figure 3.103: Distribution of neuron Euclidean distance from a 40 × 40 toroidal SOM umat trained using linear learning rate decay and radius decay and Gaussian radius function using RGZ latent vectors produced using an autoencoder with random rotation augmentation training. Colour blending indicates RGZ label derived complexity and anomalousness of all RGZ validation images matching each neuron.

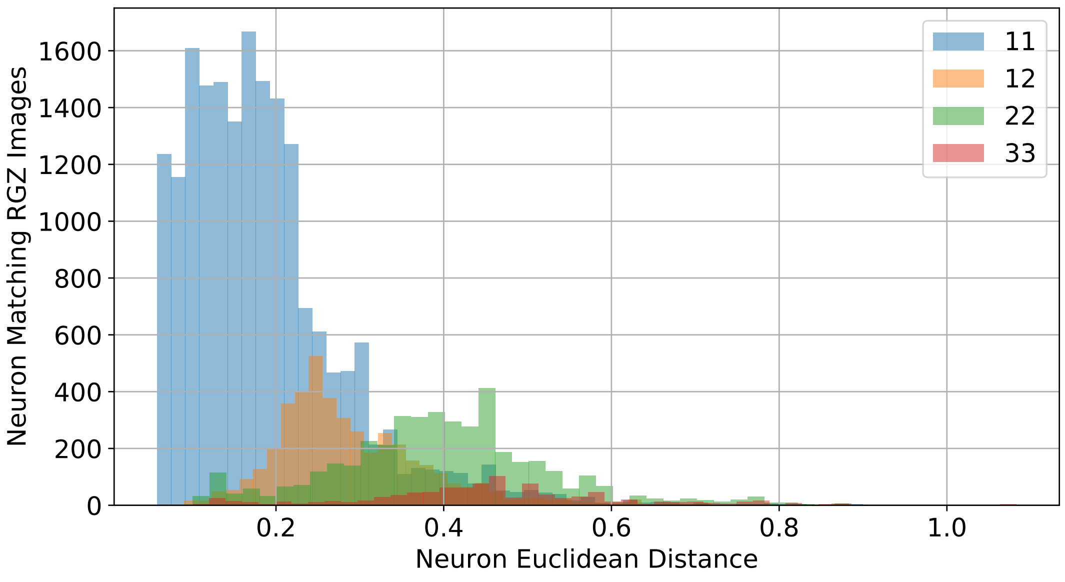

Figure 3.104: Distribution of neuron Euclidean distance from a 40 × 40 toroidal SOM umat trained using linear learning rate decay and radius decay and Gaussian radius function using RGZ latent vectors produced using an autoencoder with random rotation augmentation training. Colour blending indicates the RGZ label of the four most dominant labelled RGZ validation images matching each neuron.

Table 3.30: Accuracy and performance metrics of anomaly detection by separating neuron matches sources based on the Euclidean distance of neurons a 40×40 toroidal SOM umat at each SD 1-5, trained with a linear learning rate decay and radius decay and Gaussian radius function using RGZ latent vectors produced using an autoencoder with random rotation augmentation training

10 × 10 Map Size Neuron Euclidean Distance SD	Euclidean Distance at SD	Accuracy	Precision	Recall	F1 Score
SD1	0.148	0.613	0.493	0.965	0.653
SD2	0.297	**0.804**	0.781	0.666	**0.719**
SD3	0.445	0.707	0.851	0.271	0.411
SD4	0.594	0.645	0.878	0.068	0.127
SD5	0.742	0.631	0.893	0.024	0.048

Table 3.31: Accuracy and performance metrics of anomaly detection by separating neuron matches sources based on the Euclidean distance of neurons on square 10, 20 and 40 toroidal SOM umat at each SD 1-5, trained with a linear learning rate decay and radius decay and Gaussian radius function using RGZ latent vectors produced using an autoencoder with random rotation augmentation training compared to no rotation augmentations

Map Size and Augmentations	Neighbourhood Type	Accuracy	Precision	Recall	F1 Score
Augmented 10 × 10	Gaussian	0.812	0.816	0.648	0.722
Augmented 20 × 20	Gaussian	0.823	0.721	0.865	**0.787**
Augmented 40 × 40	Gaussian	0.804	0.781	0.666	0.719
Un-Augmented 10 × 10	Gaussian	0.833	0.758	0.818	0.787
Un-Augmented 20 × 20	Gaussian	**0.824**	0.841	0.659	0.739
Un-Augmented 40 × 40	Gaussian	0.752	0.611	0.943	0.741

Table 3.32: Processing time metrics of each toroidal SOM with square 5,10,20 and 40 sizes maps trained with a linear learning rate decay and radius decay and Gaussian radius function using RGZ latent vectors produced using an autoencoder with random rotation augmentation training compared to no rotation augmentations

Map Size	Augmentations	Total Training Time (s)	Average Training Time (s)
10 × 10	Augmented	9.457	0.788
20 × 20	Augmented	25.208	2.101
40 × 40	Augmented	91.88	7.657
10 × 10	Un-Augmented	8.571	0.714
20 × 20	Un-Augmented	25.528	2.127
40 × 40	Un-Augmented	92.089	7.674

3.3.3.5 Hierarchical Clustering of Optimised Self-Organised Map Neuron Weights

In this Section I detail the HC of 10×10, 20×20 and 40×40 square toroidal SOMs with the most accurate training configurations from the previous tests in Sections 3.3.3.1 - 3.3.3.3. This configuration is the linear learning rate and radius decay rate function with a Gaussian neighbourhood function. I tested 4,8,16 'K' clusters using the Scikit-Learn K-means HC algorithm. Additionally, I applied the HC to maps trained with latent vectors produced from the autoencoder trained with random rotation augmentations and the optimal training hyper-parameters, as presented in Section 3.3.3.4.

Table 3.73 details the time taken for each test to perform the K-means HC. Processing time across all maps is less than a second in most cases, where all K clusters on 10×10 and 20×20 map sizes require less than half a second to complete and the highest processing time for the K 16 clusters in the 40×40 map is just over 1 second. These figures suggest K-means clustering time increases proportionally with both the allocated number of clusters and the map size. There are no clear differences in the clustering time required for the latent vectors with or without rotation augmentation training.

Test results for the map segmentation using HC are shown for each map size using an image of the SOM grid with decoded neurons weights (described in Section 2.5.3) and map displaying the closest matching RGZ image, with a K-means HC colour coding on each neuron for the assigned K-means cluster ID number (Section 2.5.5). All K-means cluster ID numbers are arbitrary as they are assigned in an unsupervised sense. All associated ID colours are assigned as discrete intervals on the Matplotlib 'jet' colour map to visually differentiate individual clusters IDs. Two tables are included for each test. The first Table describes the division of the map assigned to each cluster and associated entropy statistics. The second Table describes for every cluster, the division of neurons with the label of the closest matching RGZ image to each neuron, the total division of the RGZ images with that label in the cluster and the associated entropy statistics.

The results of the HC are similar to the preliminary HC results of Section 3.3.2, and show that across all map sizes that clustering is closely related to morphology and to a lesser extent, the rotation angle of the source features. Clusters also appear to segment morphologies by their relative complexity. Regarding general clustering quality, all tests show reasonable connectedness with few neuron clusters inter-mixing were only minor issues are present in larger maps such as the 40×40 map, where 16 K clusters are used with some stray neurons caught in the middle of neighbouring clusters (Figures 3.139 and 3.121).

Starting in the 10×10 map without random rotation augmentation latent vectors in Figures 3.105 and 3.106 and Tables 3.33-3.34, the four K clusters appear to produce the following groupings:

- Point sources in K-means cluster 0.
- Compact doubles such as RGZ labelled 22 and 12 sources with K cluster 1.
- K-means cluster 2 separating bright and dense multi-component and anomalous sources with bent-tail morphologies and possible artefacts.
- K-means cluster 3 groups multi-component single peak sources with companions.

As in the preliminary results, there are clear complex, simple and intermediate classes. These simple classes can be seen in Figure 3.106 and Table 3.34 in cluster 0, which contains 73.5% of point sources labels, where 81.8% of the cluster contains closest matching RGZ images with this RGZ 11 label. Intermediate classes such as cluster 1, contain mostly a mix of 'medium' complex sources such as RGZ label 12. While more complex classes such as cluster 2 which contain a 72.9% of all RGZ labelled 22 sources and the most complex with cluster 3 containing 100% of all 13 sources and a mix of other classes with the highest mean entropy.

Similar relationships are shown across all the 20×20 and 40×40 map sizes, but naturally with different assigned ID numbers. Tables 3.40 and 3.46 reinforce these observations with four clusters representing each of these morphologies. As in the map size testing of Section 3.3.3.1, tests demonstrate that with larger maps and neuron counts, more morphologies can be represented and more meaningful clusters extracted with the HC, shown with the presence of more clustered complex 33 RGZ labelled images. Greater representation in these larger maps where the 20×20 and 40×40 map contains classes dominated by point sources, mid complexity and high complexity sources that correlated to the mean entropy.

These findings continue in higher K numbers of 8 and 16, where clusters become increasingly more pure in terms of how homogeneous their associated morphologies appear to be, with minimal mixing and leakage. Detailed in Tables 3.41-3.51 and Figures 3.113-3.122, these relationships become most apparent with many clusters being comprised of high divisions of the closest matching neuron image labels that correlate with the different complexity levels observed in the preliminary results and K 4 cluster tests. The higher 20×20 and 40×40 map sizes with the 8 and 16 clusters once again produce the most effective separations. An example of this can be shown in Figures 3.113 and 3.114, and Table 3.42, where many clusters contain such as the 20×20 map cluster 0 containing the entire population 13 and 33 RGZ labelled sources. This clustering quality is further reinforced by reduced mean entropy in clusters of these sizes and with these large maps.

Only larger maps show greater complexity with more triple classes, and none of the highly anomalous labels appears as first matching images but are clustered none the less. These classes are only shown with the largest 40×40 map where the K-means cluster 14 of the 16 K contains great anomalousness and the rare 44 RGZ label at a population division of only 0.0002 (Section 2.2). The 10×10 map size does not perform as well here with the 8 and 16 K cluster numbers, where some clear morphological clusters are broken into separate K clusters.

Overall, clusters across all 20×20 and 40×40 tests with 8 and 16 K values, effectively separate the following morphologies and image types:

- Point sources (RGZ label 11)
- Compact, multi-peak single component sources (RGZ label 12)
- Highly separated sources or sources with companions or possibly unrelated multiple source images.
- Source with greater than normal noise or PSF side-lobes with point sources and the occasional compact single component multi-peak source.
- Bright and diffuse extended structure with globular and bent tail morphology.
- Highly compact multi-peak and component sources.
- Source with extended emission with large jets and AGN-like morphology.

It is evident in these population division tables the SOM, and the HC are segmenting morphologies and associating relationships not based entirely on the RGZ labels. This is indicated by relatively low populations of RGZ labels in many clusters across all tests, despite the clustered maps showing reasonably clear and logically assigned clusters based on morphology and complexity.

In the HC of maps trained with random rotation trained autoencoder latent vectors (Figures 3.90 - 3.140 and Tables 3.28 - 3.72), there are primarily similar cluster assignments. However, in the HC of the 10×10 map trained with random rotation augmentation latent vectors (Figures 3.123 and 3.124), clusters appear to not group sources as well, with Table 3.53 detailing the point source dominated cluster ID 1 containing 76.2% of the cluster being RGZ label 11, only containing a 56.1% of the 11 labelled sources in dataset, compared to the 81.8% from similar simple clusters without rotation, shown in Table 3.34. Similar results are seen across the 20×20 and 40×40 map sizes and even in higher cluster numbers 8 and 16, wherein Tables 3.58 to 3.72 show that most classes have comparatively lower closest matching neuron label divisions and overall label population divisions. In these tests, rotation changes appear to be taken into account where many clusters are separated more by the rotation angle of the components and peaks within the image rather than morphological differences. Rotated map tests tend to show a more continuous nature that is clustered in a manner more depended on image rotation angle. However, both clustering maps show similar relationships where there are many distinct reconstructions and morphologies separated and associated.

The mean entropy measures across clusters increase with the presence of more complex and anomalous sources, as also observed in the preliminary results. Entropy here describes how 'mixed' the neurons and clusters are where a low entropy indicates a homogeneous neuron or cluster matching label set (detailed further in Section 2.5.6.1). These relationships between entropy and complexity are demonstrated across all test tables where clusters dominated by RGZ labelled point sources 11 have the lowest entropy, while RGZ 12 labelled single component double peak sources show increased entropy and more complex sources such as radio triple RGZ 33 labels with greater still entropy. In the largest, 40×40 map sizes where even anomalous sources are neuron closest matching images with highly anomalous RGZ labelled 44, posses entropy in the highest order. These groupings are seen across all test results in Tables 3.33 - 3.72, which show these entropy distributions, where point sources have the greatest neuron population across the map and the lowest mean entropy. Higher complexity sources, which represent a smaller part of the training set appear to have consistently smaller cluster populations. These relationships are illustrated by the results of all map sizes and latent vector training augmentation tests.

These results demonstrate successfully HC of the complex SOM space. It is clear from these results that the best radio-astronomical image feature and morphology separations can be found in larger map sizes 20×20 and 40×40 with K of 8 and 16 clusters.

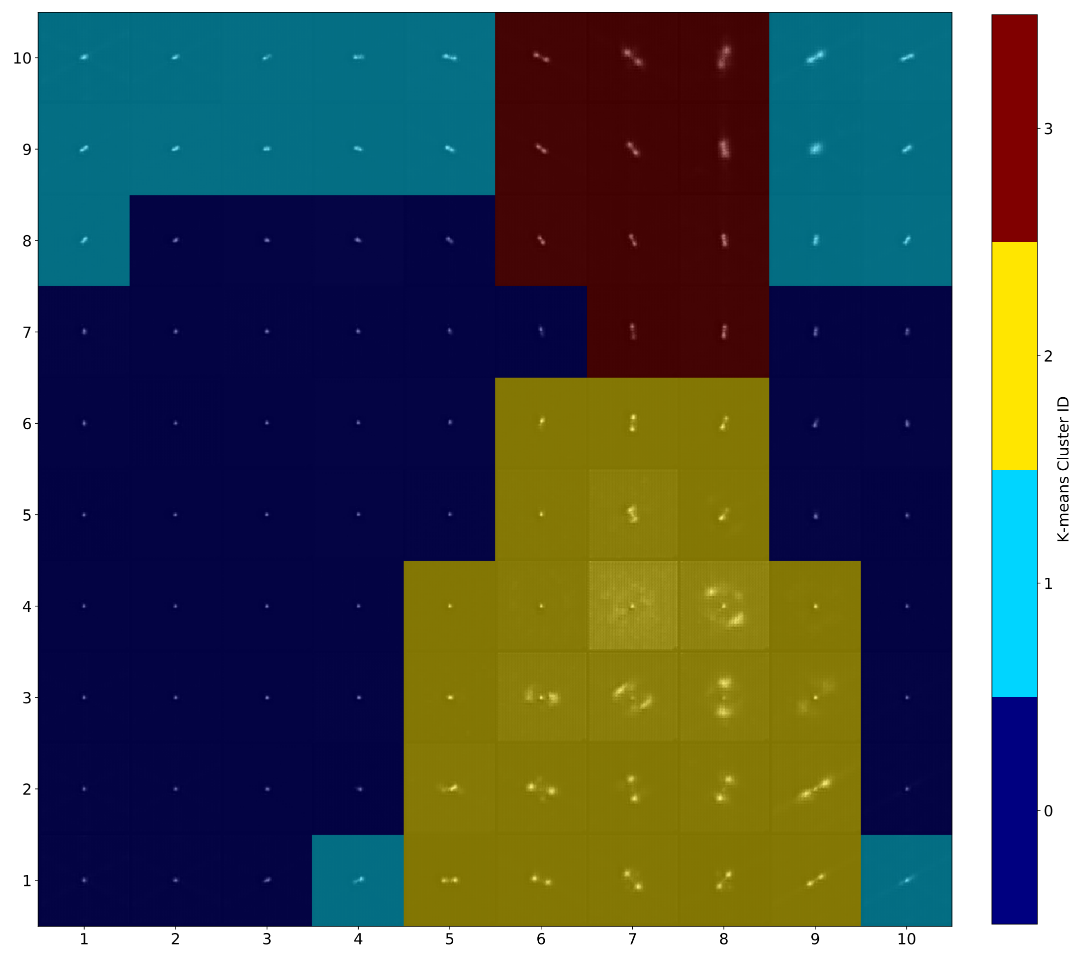

Figure 3.105: 10 × 10 toroidal SOM trained using linear learning rate decay and radius decay, with a Gaussian radius decay, displaying decoded neuron weights with 4 colour coded K-means clusters.

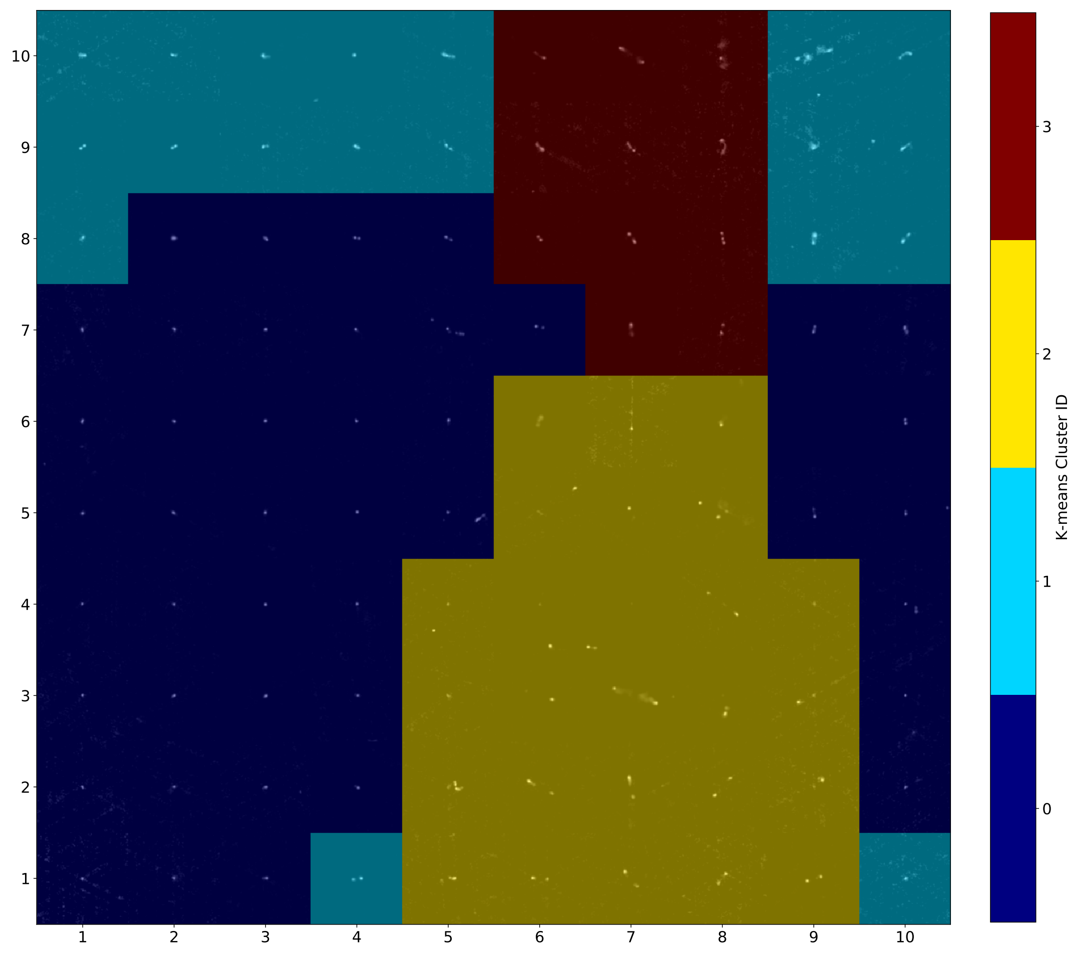

Figure 3.106: 10 × 10 toroidal SOM trained using linear learning rate decay and radius decay, with a Gaussian radius decay, displaying neuron closest matching RGZ images with 4 colour coded K-means clusters.

Table 3.33: Cluster population and entropy statistics for the HC 10 × 10 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay with 4 K clusters.

K-Means Cluster	Cluster Population Over Map	Minimum Entropy	Maximum Entropy	Mean Entropy
0	0.44	0.03	0.52	0.28
1	0.19	0.00	0.73	0.39
2	0.26	0.01	0.91	0.58
3	0.11	0.40	1.00	0.79

Table 3.34: Divisions of clusters based on the label of the closest matching RGZ image, matching clustered label divisions and entropy statistics for each of the 4 K clusters in the HC 10×10 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay

Cluster ID	RGZ Label	Division of Matching Label In Cluster	Division of All Image Labels In Cluster	Mean Entropy	Minimum Entropy	Maximum Entropy
0	11	0.818	0.735	0.25	0.03	0.35
	12	0.159	0.280	0.38	0.24	0.52
	22	0.023	0.040	0.41	0.41	0.41
1	12	0.684	0.520	0.47	0.15	0.70
	11	0.263	0.102	0.12	0.00	0.31
	22	0.053	0.040	0.73	0.73	0.73
2	22	0.692	0.720	0.73	0.38	0.91
	11	0.308	0.163	0.24	0.01	0.64
3	22	0.455	0.200	0.86	0.76	1.00
	12	0.455	0.200	0.69	0.40	0.81
	13	0.091	1.000	0.90	0.90	0.90

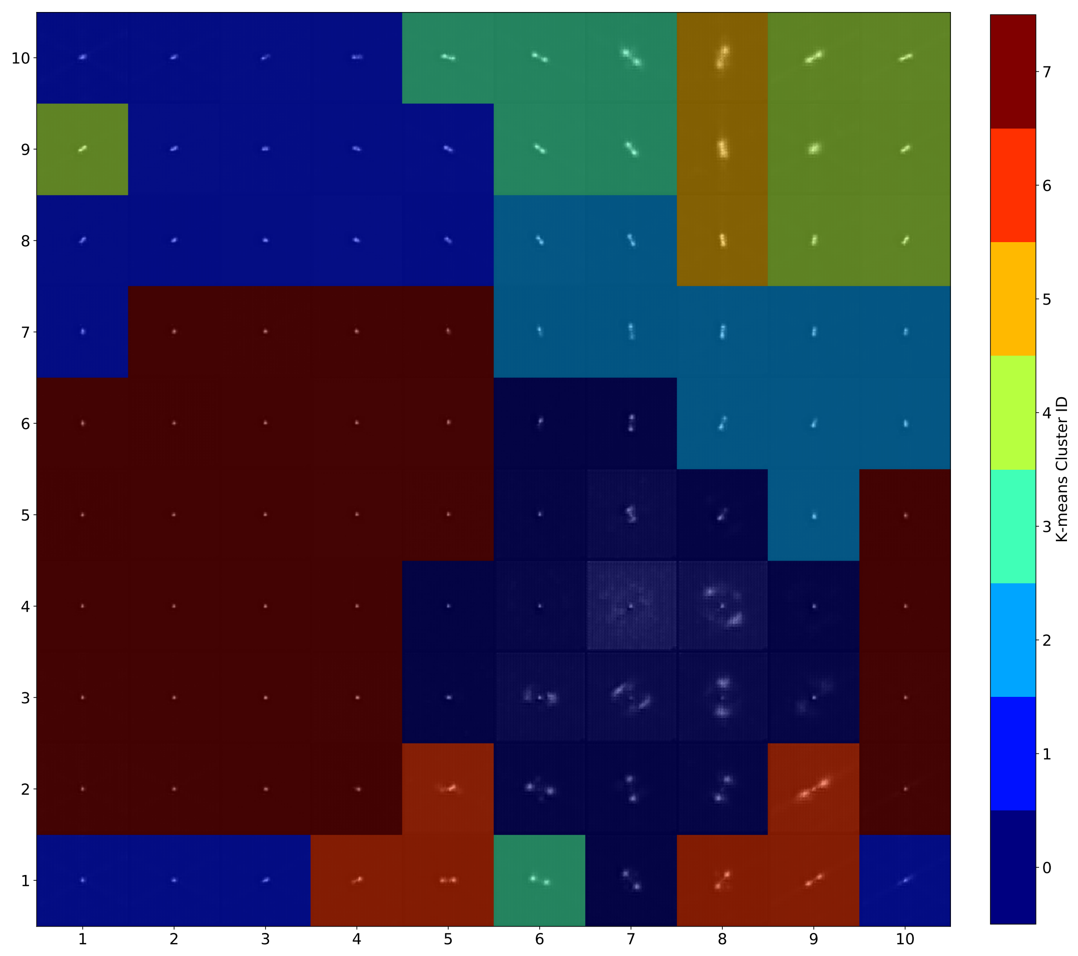

Figure 3.107: 10×10 toroidal SOM trained using linear learning rate decay and radius decay, with a Gaussian radius decay, displaying decoded neuron weights with 8 colour coded K-means clusters.

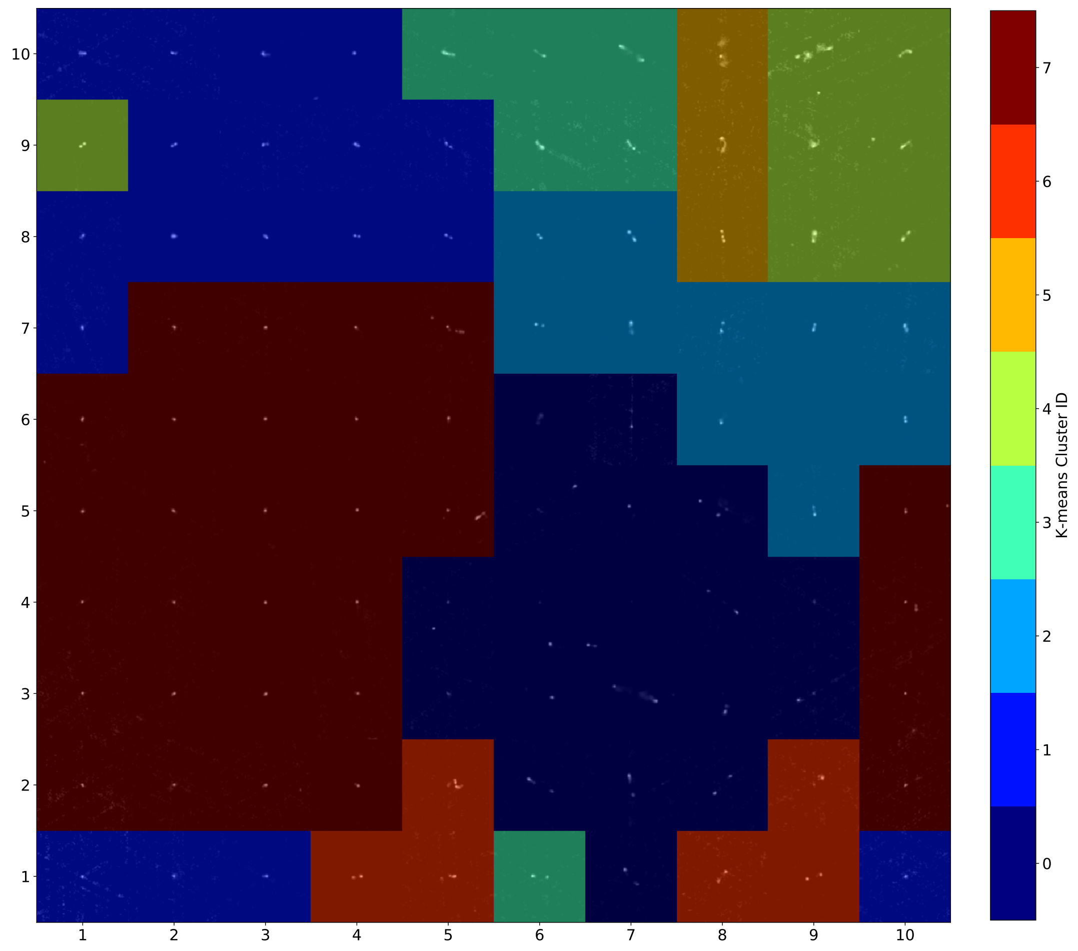

Figure 3.108: 10 × 10 toroidal SOM trained using linear learning rate decay and radius decay, with a Gaussian radius decay, displaying neuron closest matching RGZ images with 8 colour coded K-means clusters.

Table 3.35: Cluster population and entropy statistics for the HC 10 × 10 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay with 8 K clusters.

K-Means Cluster	Cluster Population Over Map	Minimum Entropy	Maximum Entropy	Mean Entropy
0	0.19	0.01	0.89	0.51
1	0.18	0.00	0.62	0.26
2	0.11	0.34	0.84	0.58
3	0.06	0.70	0.89	0.78
4	0.07	0.15	0.73	0.41
5	0.03	0.40	1.00	0.77
6	0.06	0.43	0.91	0.73
7	0.30	0.03	0.35	0.28

Table 3.36: Divisions of clusters based on the label of the closest matching RGZ image, matching clustered label divisions and entropy statistics for each of the 8 K clusters in the HC 10×10 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay

Cluster ID	RGZ Label	Division of Matching Label In Cluster	Division of All Image Labels In Cluster	Mean Entropy	Minimum Entropy	Maximum Entropy
0	11	0.556	0.204	0.12	0.00	0.22
	12	0.444	0.320	0.44	0.30	0.62
1	11	0.967	0.592	0.28	0.03	0.35
	12	0.033	0.040	0.24	0.24	0.24
2	12	0.571	0.160	0.42	0.15	0.62
	11	0.286	0.041	0.23	0.15	0.31
	22	0.143	0.040	0.73	0.73	0.73
3	22	0.833	0.200	0.77	0.43	0.91
	12	0.167	0.040	0.51	0.51	0.51
4	22	0.579	0.440	0.71	0.38	0.89
	11	0.421	0.163	0.24	0.01	0.64
5	22	0.500	0.120	0.81	0.76	0.89
	12	0.500	0.120	0.76	0.70	0.81
6	12	0.636	0.280	0.51	0.34	0.81
	22	0.364	0.160	0.71	0.41	0.84
7	13	0.333	1.000	0.90	0.90	0.90
	22	0.333	0.040	1.00	1.00	1.00
	12	0.333	0.040	0.40	0.40	0.40

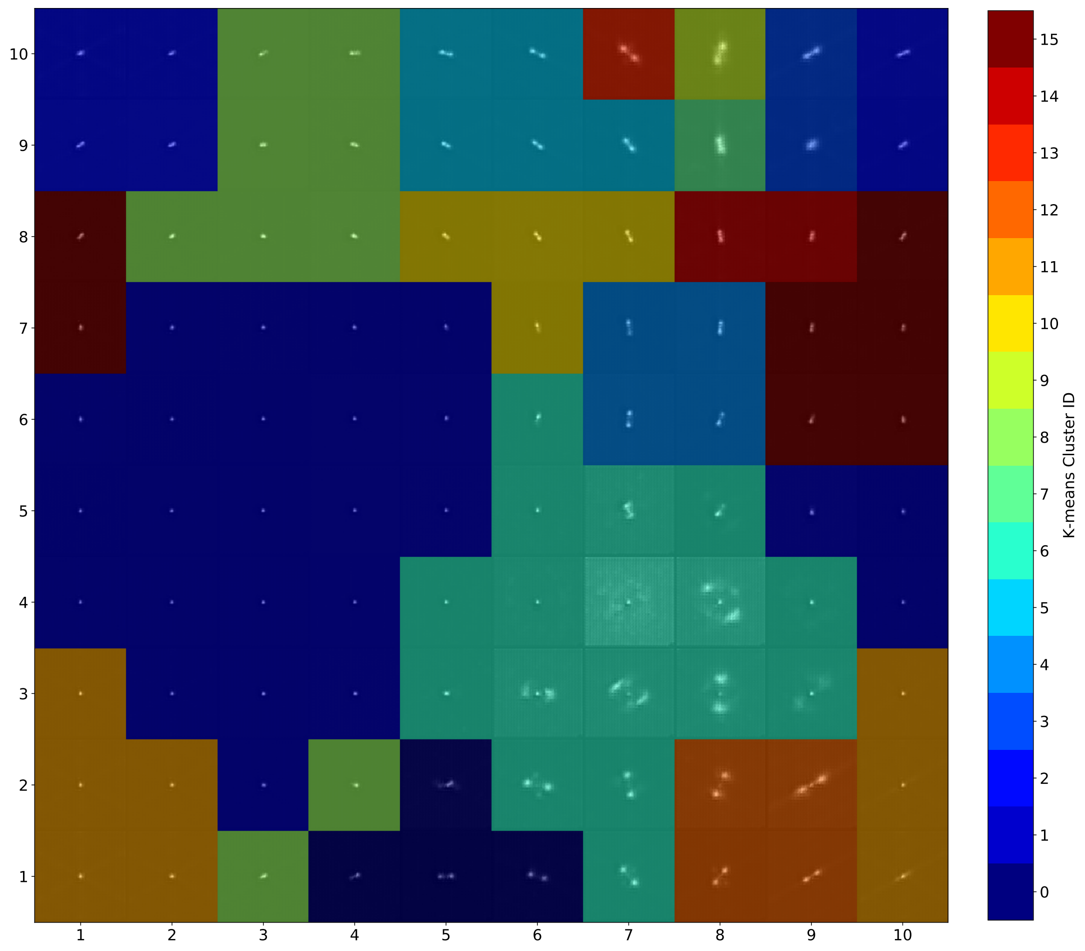

Figure 3.109: 10 × 10 toroidal SOM trained using linear learning rate decay and radius decay, with a Gaussian radius decay, displaying decoded neuron weights with 16 colour coded K-means clusters.

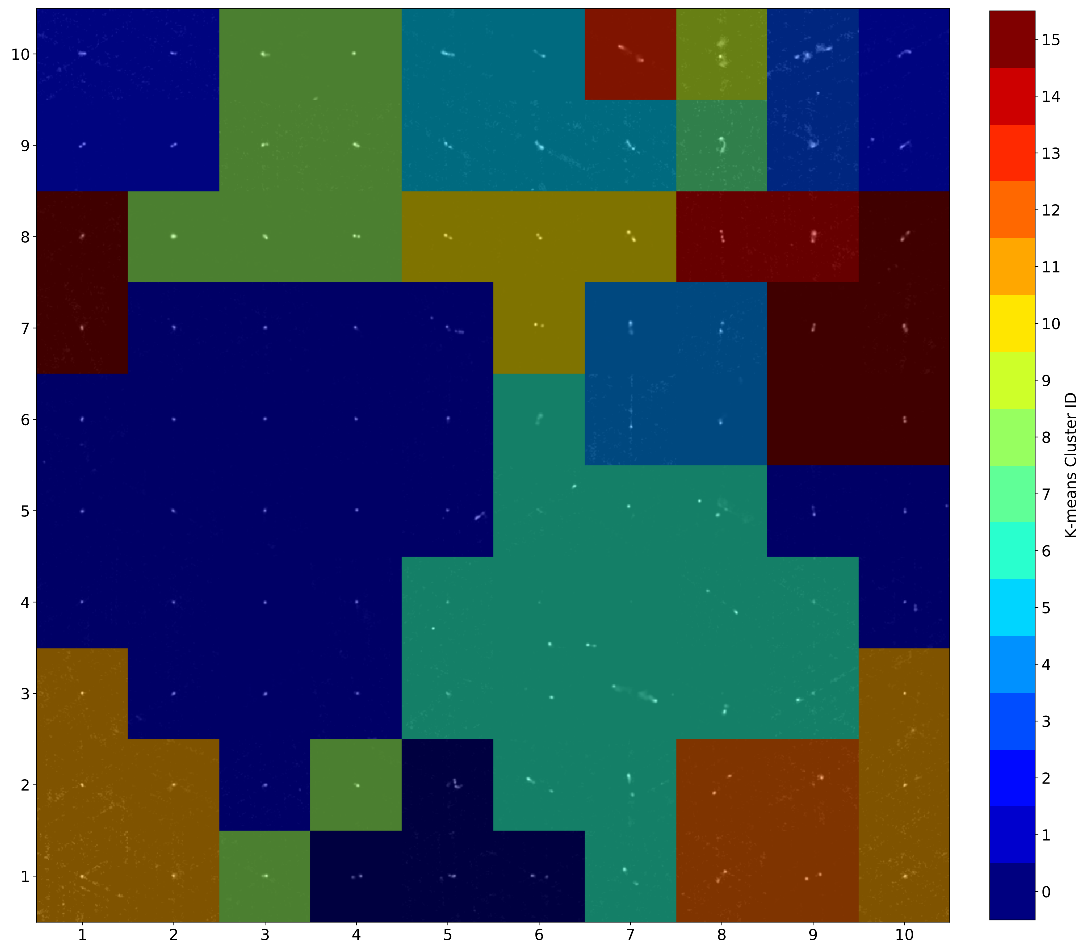

Figure 3.110: 10 × 10 toroidal SOM trained using linear learning rate decay and radius decay, with a Gaussian radius decay, displaying neuron closest matching RGZ images with 16 colour coded K-means clusters.

Table 3.37: Cluster population and entropy statistics for the HC 10×10 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay with 16 K clusters.

K-Means Cluster	Cluster Population Over Map	Minimum Entropy	Maximum Entropy	Mean Entropy
0	0.04	0.43	0.83	0.64
1	0.25	0.03	0.41	0.27
2	0.06	0.00	0.62	0.27
3	0.02	0.31	0.73	0.52
4	0.04	0.74	0.84	0.80
5	0.05	0.52	0.81	0.71
6	0.17	0.01	0.89	0.48
7	0.01	0.90	0.90	0.90
8	0.09	0.04	0.62	0.30
9	0.01	1.00	1.00	1.00
10	0.04	0.33	0.81	0.59
11	0.08	0.10	0.35	0.26
12	0.04	0.79	0.91	0.86
13	0.01	0.89	0.89	0.89
14	0.02	0.15	0.40	0.28
15	0.07	0.12	0.42	0.35

Table 3.38: Divisions of clusters based on the label of the closest matching RGZ image, matching clustered label divisions and entropy statistics for each of the 16 K clusters in the HC 10×10 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay

Cluster ID	RGZ Label	Division of Matching Label In Cluster	Division of All Image Labels In Cluster	Mean Entropy	Minimum Entropy	Maximum Entropy
0	11	1.000	0.163	0.26	0.10	0.35
1	11	0.960	0.490	0.27	0.03	0.35
	22	0.040	0.040	0.41	0.41	0.41
2	12	0.857	0.240	0.39	0.34	0.42
	11	0.143	0.020	0.12	0.12	0.12
3	11	0.500	0.061	0.06	0.00	0.15
	12	0.500	0.120	0.48	0.30	0.62
4	12	0.556	0.200	0.45	0.24	0.62
	11	0.444	0.082	0.12	0.04	0.20
5	22	0.750	0.120	0.68	0.43	0.83
	12	0.250	0.040	0.51	0.51	0.51
6	22	0.529	0.360	0.69	0.38	0.89
	11	0.471	0.163	0.24	0.01	0.64
7	12	1.000	0.160	0.59	0.33	0.81
8	12	0.800	0.160	0.70	0.52	0.81
	22	0.200	0.040	0.76	0.76	0.76
9	22	1.000	0.160	0.80	0.74	0.84
10	22	1.000	0.040	0.89	0.89	0.89
11	22	1.000	0.160	0.86	0.79	0.91
12	12	1.000	0.080	0.28	0.15	0.40
13	13	1.000	1.000	0.90	0.90	0.90
14	22	1.000	0.040	1.00	1.00	1.00
15	11	0.500	0.020	0.31	0.31	0.31
	22	0.500	0.040	0.73	0.73	0.73

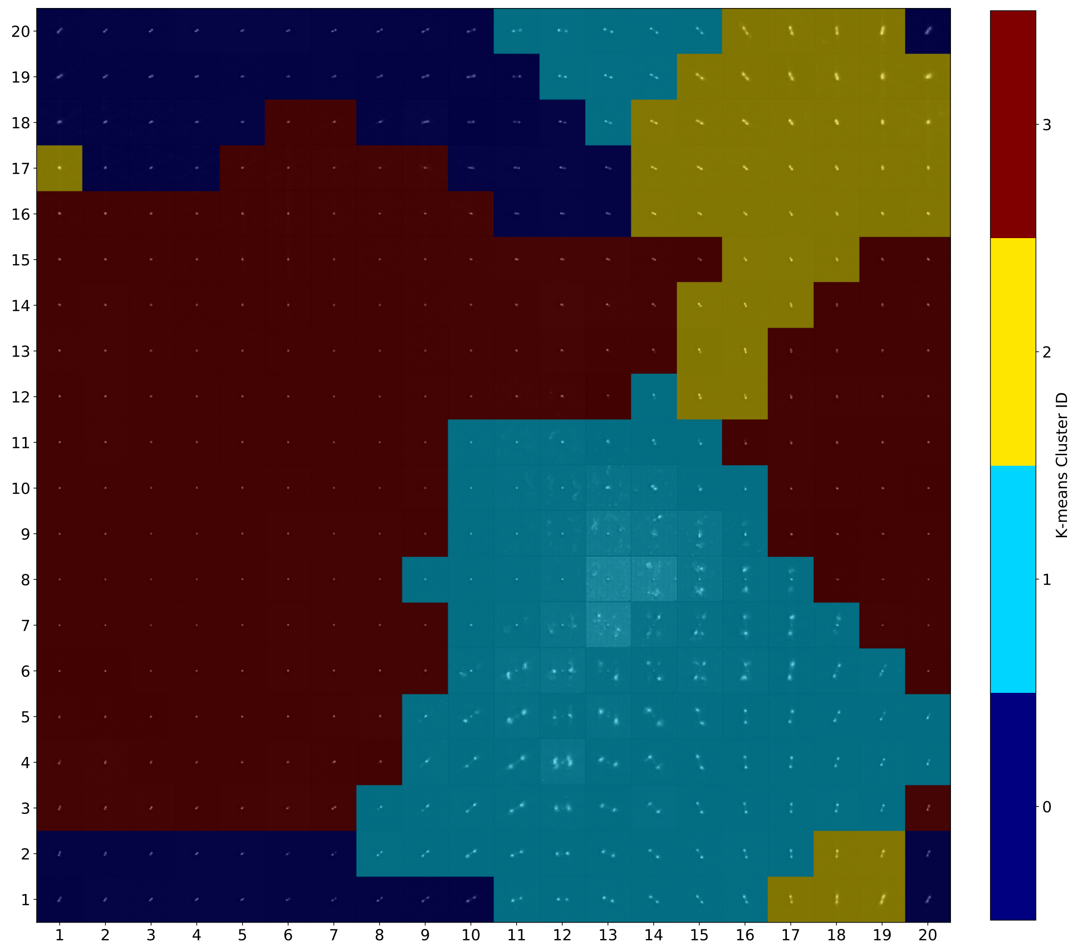

Figure 3.111: 20 × 20 toroidal SOM trained using linear learning rate decay and radius decay, with a Gaussian radius decay, displaying decoded neuron weights with 4 colour coded K-means clusters.

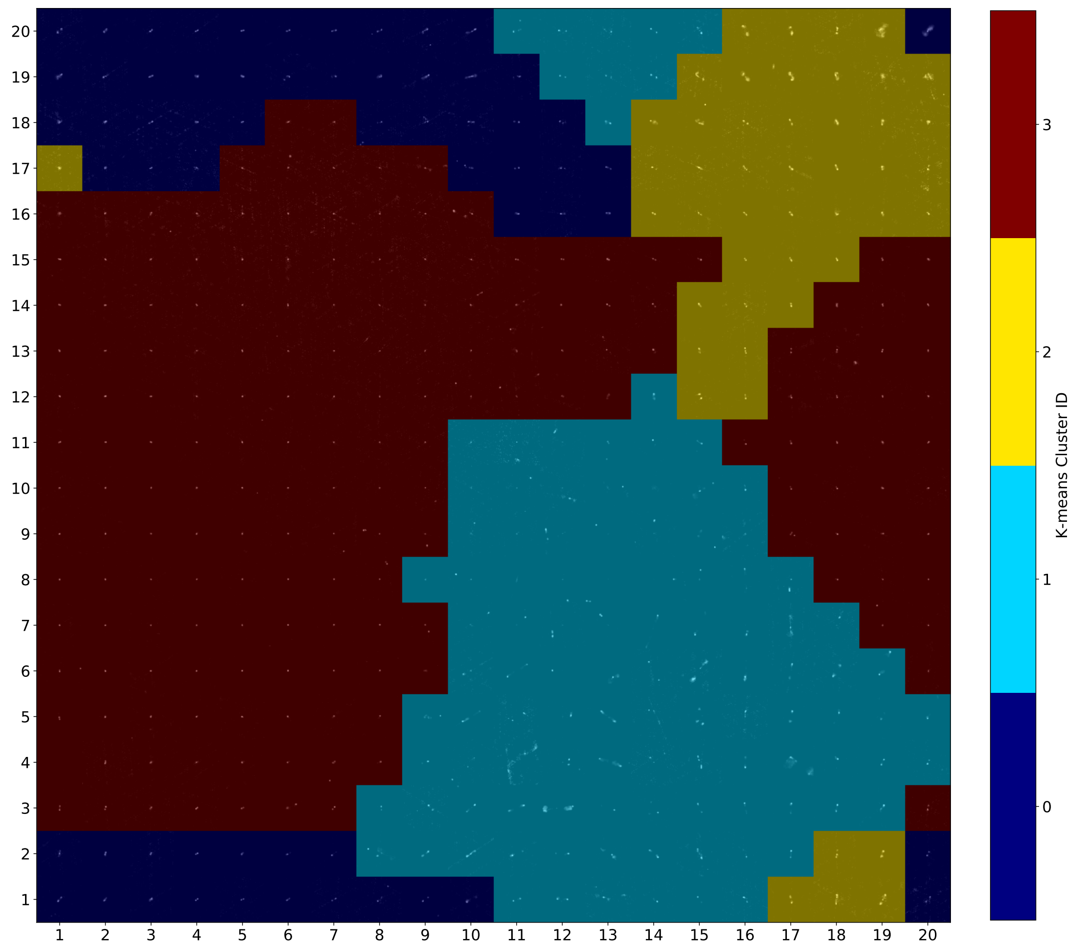

Figure 3.112: 20 × 20 toroidal SOM trained using linear learning rate decay and radius decay, with a Gaussian radius decay, displaying neuron closest matching RGZ images with 4 colour coded K-means clusters.

Table 3.39: Cluster population and entropy statistics for the HC 20 × 20 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay with 4 K clusters.

K-Means Cluster	Cluster Population Over Map	Minimum Entropy	Maximum Entropy	Mean Entropy
0	0.1525	0.01	0.69	0.32
1	0.2750	0.00	1.00	0.36
2	0.1175	0.00	0.93	0.39
3	0.4550	0.00	0.66	0.17

Table 3.40: Divisions of clusters based on the label of the closest matching RGZ image, matching clustered label divisions and entropy statistics for each of the 4 K clusters in the HC 20×20 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay

Cluster ID	RGZ Label	Division of Matching Label In Cluster	Division of All Image Labels In Cluster	Mean Entropy	Minimum Entropy	Maximum Entropy
0	12	0.590	0.404	0.35	0.14	0.69
	22	0.213	0.137	0.44	0.13	0.69
	11	0.197	0.056	0.08	0.01	0.20
1	11	0.857	0.729	0.14	0.00	0.21
	12	0.137	0.281	0.34	0.10	0.66
	22	0.005	0.011	0.30	0.30	0.30
2	12	0.596	0.315	0.40	0.12	0.74
	22	0.213	0.105	0.49	0.34	0.59
	11	0.149	0.033	0.10	0.00	0.27
	13	0.021	1.000	0.93	0.93	0.93
	33	0.021	1.000	0.81	0.81	0.81
3	22	0.645	0.747	0.49	0.17	1.00
	11	0.355	0.182	0.14	0.00	0.39

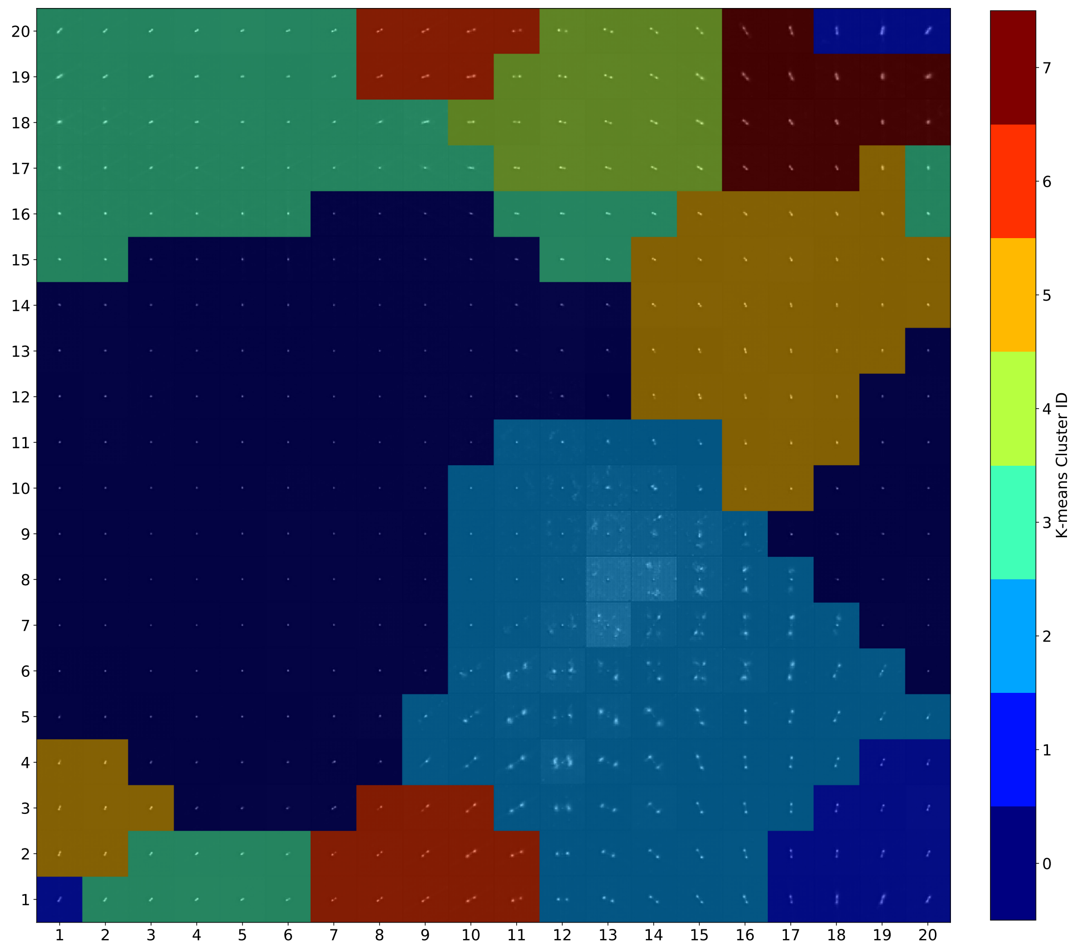

Figure 3.113: 20 × 20 toroidal SOM trained using linear learning rate decay and radius decay, with a Gaussian radius decay, displaying decoded neuron weights with 8 colour coded K-means clusters.

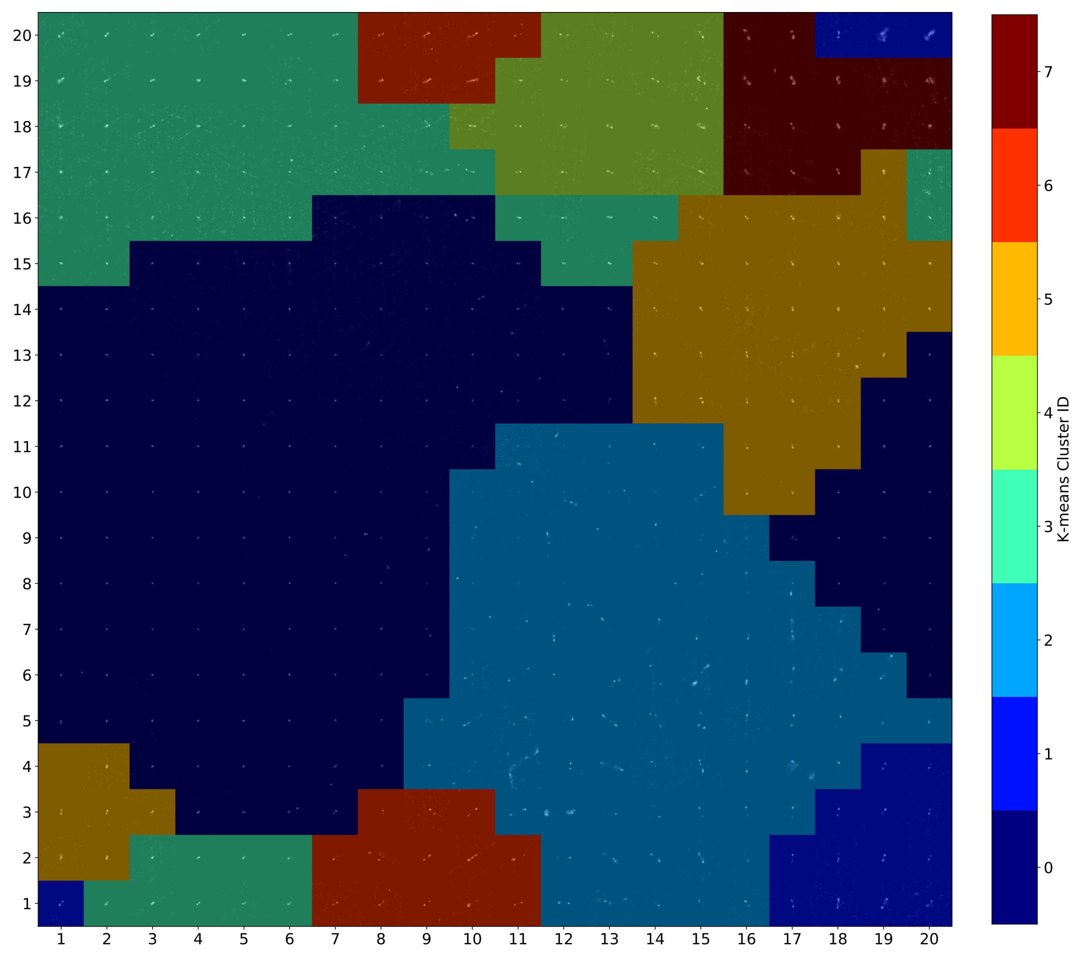

Figure 3.114: 20 × 20 toroidal SOM trained using linear learning rate decay and radius decay, with a Gaussian radius decay, displaying neuron closest matching RGZ images with 8 colour coded K-means clusters.

Table 3.41: Cluster population and entropy statistics for the HC 20 × 20 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay with 8 K clusters.

K-Means Cluster	Cluster Population Over Map	Minimum Entropy	Maximum Entropy	Mean Entropy
0	0.3575	0.00	0.41	0.16
1	0.0425	0.38	0.93	0.54
2	0.2100	0.00	1.00	0.33
3	0.1450	0.00	0.69	0.19
4	0.0500	0.34	0.60	0.46
5	0.1075	0.03	0.66	0.32
6	0.0500	0.27	0.73	0.51
7	0.0375	0.03	0.74	0.38

Table 3.42: Divisions of clusters based on the label of the closest matching RGZ image, matching clustered label divisions and entropy statistics for each of the 8 K clusters in the HC 20×20 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay

Cluster ID	RGZ Label	Division of Matching Label In Cluster	Division of All Image Labels In Cluster	Mean Entropy	Minimum Entropy	Maximum Entropy
0	22	0.647	0.116	0.50	0.39	0.62
	12	0.235	0.045	0.50	0.38	0.57
	13	0.059	1.000	0.93	0.93	0.93
	33	0.059	1.000	0.81	0.81	0.81
1	12	0.628	0.303	0.39	0.16	0.66
	11	0.256	0.051	0.10	0.03	0.21
	22	0.116	0.053	0.40	0.30	0.59
2	11	0.930	0.621	0.15	0.00	0.19
	12	0.070	0.112	0.23	0.10	0.41
3	11	0.517	0.140	0.09	0.00	0.20
	12	0.448	0.292	0.31	0.14	0.69
	22	0.034	0.021	0.15	0.13	0.16
4	22	0.750	0.158	0.52	0.27	0.73
	12	0.250	0.056	0.47	0.38	0.57
5	22	0.560	0.495	0.49	0.17	1.00
	11	0.440	0.173	0.13	0.00	0.39
6	22	0.650	0.137	0.47	0.35	0.60
	12	0.350	0.079	0.44	0.34	0.51
7	12	0.667	0.112	0.40	0.12	0.74
	11	0.200	0.014	0.16	0.03	0.27
	22	0.133	0.021	0.57	0.57	0.58

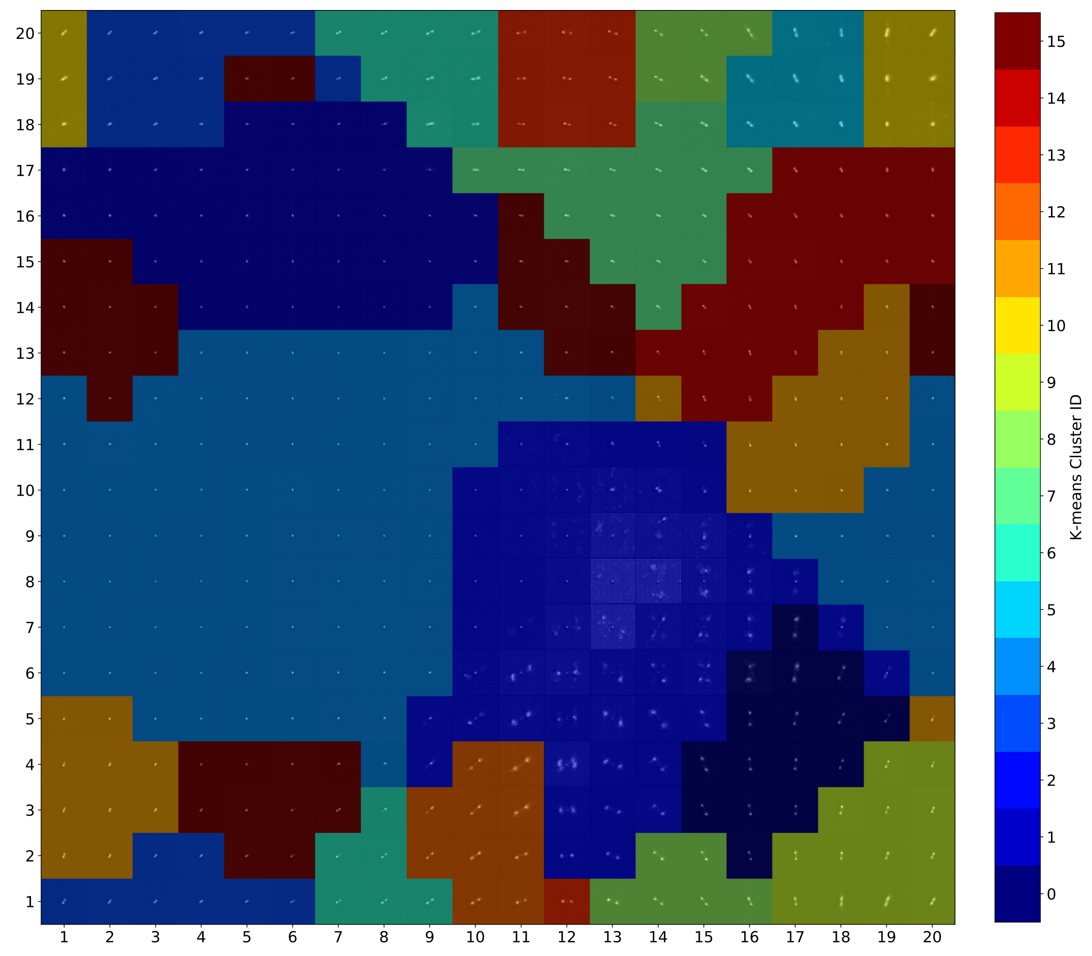

Figure 3.115: 20 × 20 toroidal SOM trained using linear learning rate decay and radius decay, with a Gaussian radius decay, displaying decoded neuron weights with 16 colour coded K-means clusters.

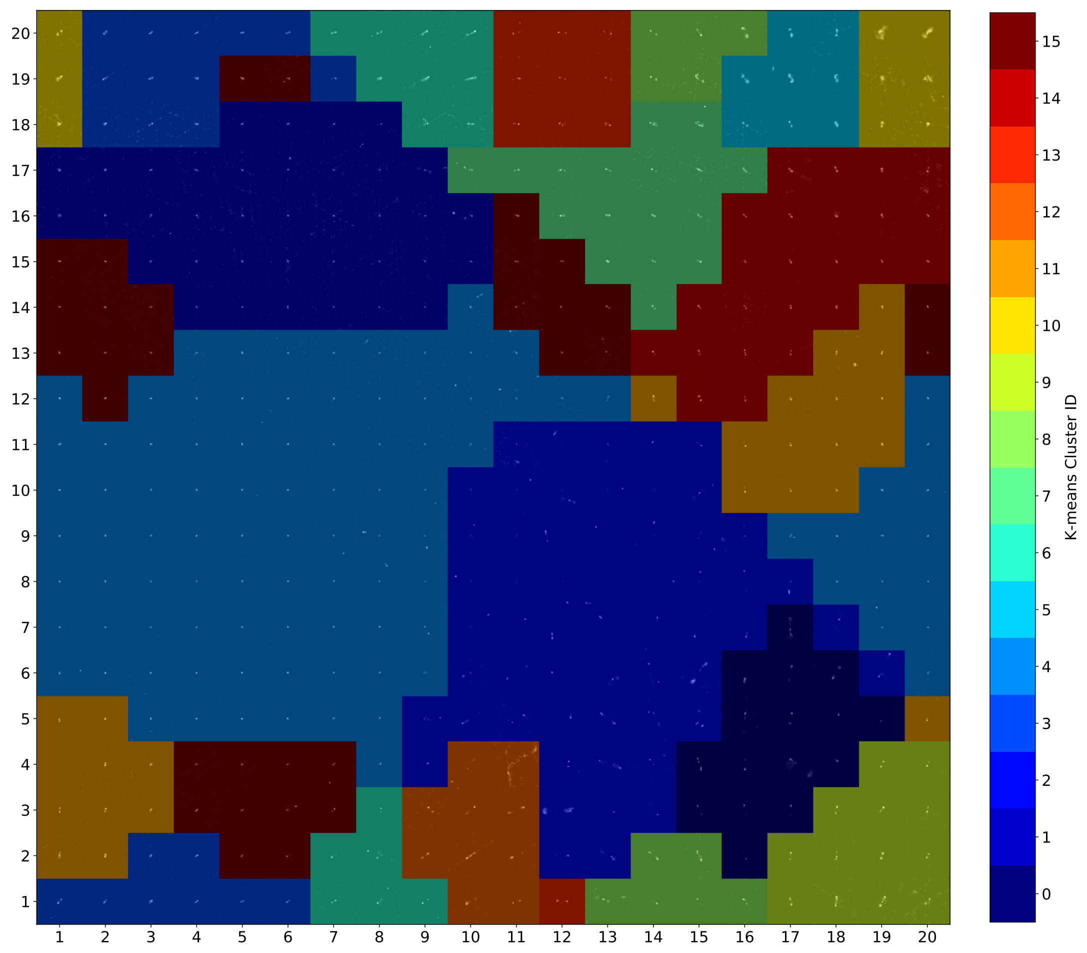

Figure 3.116: 20 × 20 toroidal SOM trained using linear learning rate decay and radius decay, with a Gaussian radius decay, displaying neuron closest matching RGZ images with 16 colour coded K-means clusters.

Table 3.43: Cluster population and entropy statistics for the HC 20×20 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay with 16 K clusters.

K-Means Cluster	Cluster Population Over Map	Minimum Entropy	Maximum Entropy	Mean Entropy
0	0.0400	0.35	1.00	0.56
1	0.0925	0.00	0.20	0.12
2	0.1425	0.00	0.65	0.24
3	0.0500	0.01	0.69	0.24
4	0.2425	0.00	0.28	0.16
5	0.0200	0.25	0.74	0.46
6	0.0375	0.12	0.63	0.46
7	0.0425	0.16	0.66	0.41
8	0.0275	0.35	0.58	0.47
9	0.0325	0.39	0.81	0.53
10	0.0225	0.01	0.93	0.29
11	0.0625	0.03	0.65	0.23
12	0.0250	0.33	0.73	0.54
13	0.0250	0.39	0.60	0.48
14	0.0600	0.00	0.65	0.32
15	0.0775	0.02	0.41	0.17

Table 3.44: Divisions of clusters based on the label of the closest matching RGZ image, matching clustered label divisions and entropy statistics for each of the 16 K clusters in the HC 20×20 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay

Cluster ID	RGZ Label	Division of Matching Label In Cluster	Division of All Image Labels In Cluster	Mean Entropy	Minimum Entropy	Maximum Entropy
0	12	0.700	0.157	0.30	0.14	0.69
	11	0.250	0.023	0.09	0.01	0.14
	22	0.050	0.011	0.13	0.13	0.13
1	11	0.520	0.061	0.10	0.03	0.21
	12	0.320	0.090	0.40	0.25	0.65
	22	0.160	0.042	0.34	0.30	0.44
2	11	0.948	0.430	0.16	0.00	0.19
	12	0.052	0.056	0.18	0.10	0.28
3	11	0.645	0.093	0.11	0.02	0.19
	12	0.355	0.124	0.28	0.19	0.41
4	11	1.000	0.173	0.12	0.00	0.20
5	11	0.444	0.019	0.12	0.01	0.27
	12	0.444	0.045	0.30	0.12	0.56
	13	0.111	1.000	0.93	0.93	0.93
6	22	0.467	0.074	0.50	0.27	0.63
	12	0.467	0.079	0.48	0.38	0.61
	11	0.067	0.005	0.12	0.12	0.12
7	22	1.000	0.105	0.54	0.33	0.73
8	11	0.649	0.173	0.13	0.00	0.39
	22	0.351	0.211	0.43	0.17	0.65
9	12	0.882	0.169	0.43	0.16	0.66
	22	0.118	0.021	0.32	0.16	0.49
10	22	1.000	0.105	0.48	0.39	0.60
11	12	1.000	0.116	0.47	0.35	0.58
12	12	0.708	0.191	0.36	0.16	0.65
	11	0.208	0.023	0.10	0.00	0.15
	22	0.083	0.021	0.46	0.34	0.59
13	22	1.000	0.168	0.56	0.35	1.00
14	12	0.750	0.067	0.45	0.25	0.74
	22	0.250	0.021	0.48	0.40	0.57
15	22	0.769	0.105	0.51	0.39	0.62
	12	0.154	0.022	0.53	0.49	0.57
	33	0.077	1.000	0.81	0.81	0.81

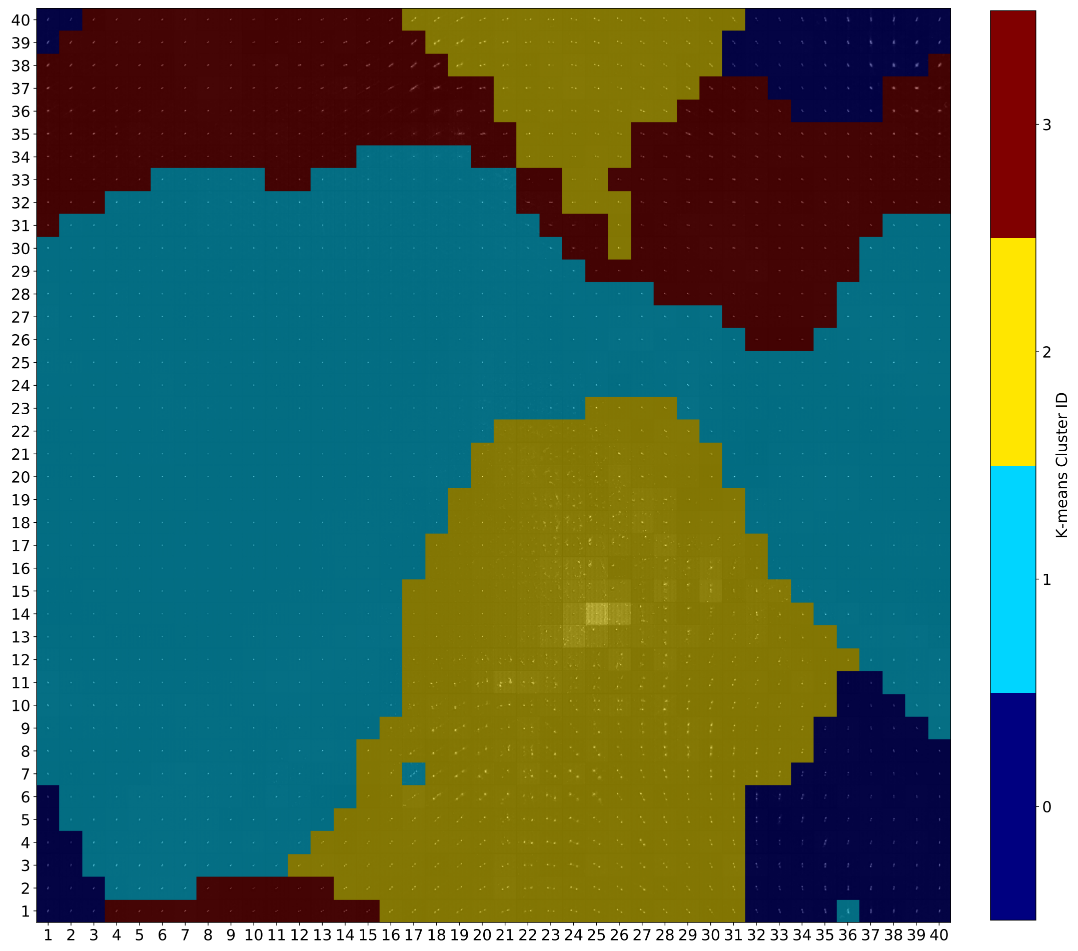

Figure 3.117: 40 × 40 toroidal SOM trained using linear learning rate decay and radius decay, with a Gaussian radius decay, displaying decoded neuron weights with 4 colour coded K-means clusters.

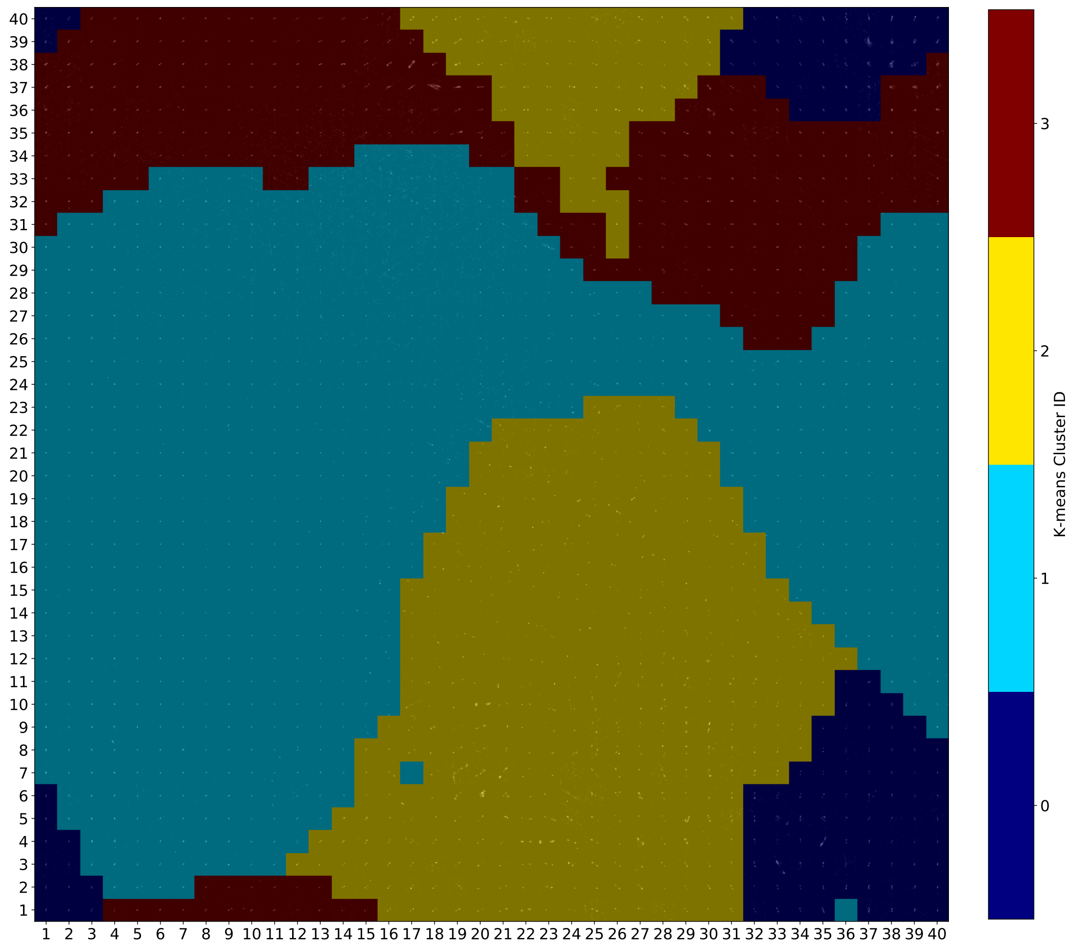

Figure 3.118: 40 × 40 toroidal SOM trained using linear learning rate decay and radius decay, with a Gaussian radius decay, displaying neuron closest matching RGZ images with 4 colour coded K-means clusters.

Table 3.45: Cluster population and entropy statistics for the HC 40 × 40 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay with 4 K clusters.

K-Means Cluster	Cluster Population Over Map	Minimum Entropy	Maximum Entropy	Mean Entropy
0	0.08000	0.02	0.78	0.29
1	0.46625	0.00	0.44	0.10
2	0.27625	0.00	0.77	0.22
3	0.17750	0.00	1.00	0.20

Table 3.46: Divisions of clusters based on the label of the closest matching RGZ image, matching clustered label divisions and entropy statistics for each of the 4 K clusters in the HC 40 × 40 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay

Cluster ID	RGZ Label	Division of Matching Label In Cluster	Division of All Image Labels In Cluster	Mean Entropy	Minimum Entropy	Maximum Entropy
0	12	0.469	0.190	0.26	0.09	0.53
	22	0.375	0.144	0.30	0.14	0.59
	11	0.078	0.011	0.13	0.02	0.42
	13	0.047	0.462	0.58	0.39	0.78
1	11	0.840	0.691	0.08	0.00	0.15
	12	0.145	0.343	0.20	0.06	0.44
2	12	0.454	0.410	0.25	0.09	0.46
	11	0.359	0.112	0.08	0.00	0.55
	22	0.151	0.129	0.28	0.07	0.50
	13	0.021	0.462	0.52	0.32	1.00
3	22	0.525	0.695	0.29	0.08	0.77
	11	0.380	0.185	0.09	0.00	0.29
	12	0.041	0.057	0.26	0.07	0.59
	33	0.038	0.895	0.35	0.21	0.56

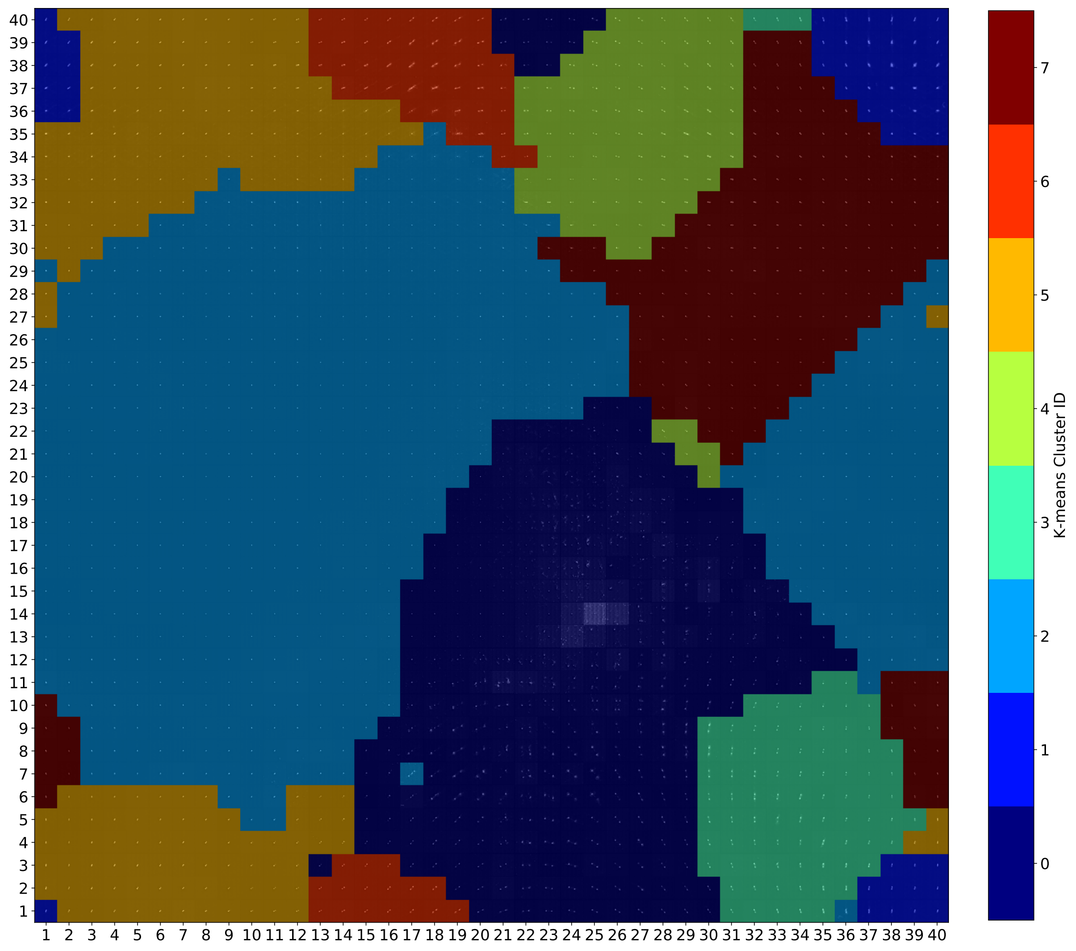

Figure 3.119: 40 × 40 toroidal SOM trained using linear learning rate decay and radius decay, with a Gaussian radius decay, displaying decoded neuron weights with 8 colour coded K-means clusters.

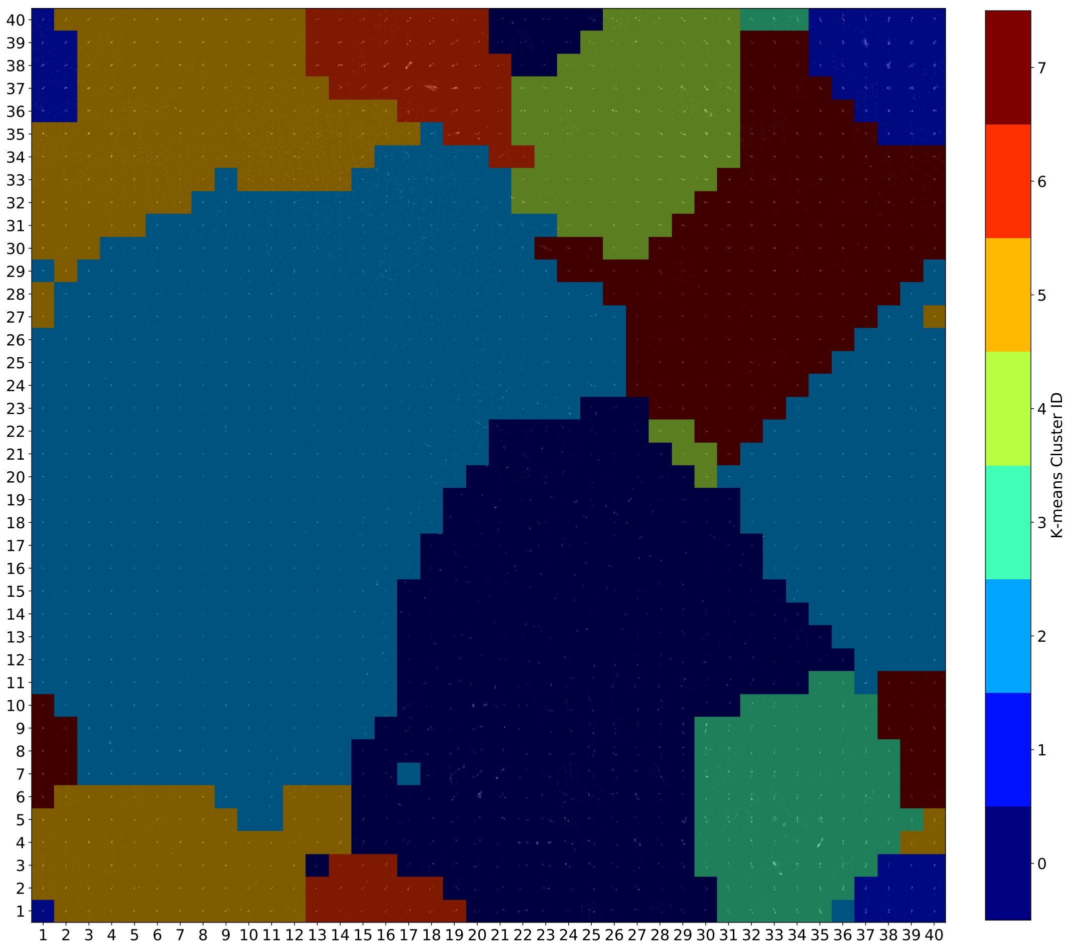

Figure 3.120: 40 × 40 toroidal SOM trained using linear learning rate decay and radius decay, with a Gaussian radius decay, displaying neuron closest matching RGZ images with 8 colour coded K-means clusters.

Table 3.47: Cluster population and entropy statistics for the HC 40 × 40 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay with 8 K clusters.

K-Means Cluster	Cluster Population Over Map	Minimum Entropy	Maximum Entropy	Mean Entropy
0	0.204	0.00	0.77	0.18
1	0.032	0.00	0.78	0.26
2	0.386	0.00	0.34	0.08
3	0.052	0.09	0.74	0.33
4	0.056	0.00	0.61	0.30
5	0.121	0.00	0.53	0.15
6	0.037	0.09	1.00	0.36
7	0.112	0.00	0.50	0.19

Table 3.48: Divisions of clusters based on the label of the closest matching RGZ image, matching clustered label divisions and entropy statistics for each of the 8 K clusters in the HC 40×40 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay

Cluster ID	RGZ Label	Division of Matching Label In Cluster	Division of All Image Labels In Cluster	Mean Entropy	Minimum Entropy	Maximum Entropy
0	12	0.471	0.076	0.25	0.09	0.53
	11	0.275	0.015	0.15	0.00	0.42
	22	0.118	0.018	0.22	0.07	0.32
	13	0.098	0.385	0.56	0.32	0.78
1	11	0.479	0.103	0.06	0.00	0.22
	12	0.443	0.273	0.24	0.08	0.53
	22	0.077	0.045	0.24	0.14	0.31
2	12	0.648	0.368	0.23	0.06	0.44
	11	0.257	0.051	0.06	0.00	0.10
	22	0.089	0.048	0.28	0.17	0.50
3	11	0.940	0.639	0.08	0.00	0.15
	12	0.049	0.095	0.15	0.06	0.34
4	22	0.542	0.096	0.34	0.21	0.47
	12	0.186	0.035	0.31	0.19	0.46
	11	0.102	0.007	0.28	0.09	0.55
	13	0.068	0.308	0.58	0.33	1.00
	33	0.051	0.158	0.50	0.36	0.58
	23	0.034	0.250	0.52	0.50	0.53
5	11	0.508	0.183	0.09	0.00	0.29
	22	0.410	0.401	0.27	0.08	0.77
	33	0.037	0.632	0.33	0.21	0.45
	12	0.037	0.038	0.23	0.07	0.55
6	22	0.674	0.180	0.30	0.11	0.48
	12	0.270	0.076	0.28	0.13	0.59
7	22	0.762	0.192	0.33	0.14	0.59
	12	0.143	0.038	0.29	0.24	0.34
	34	0.024	1.000	0.62	0.51	0.74
	13	0.024	0.154	0.49	0.48	0.50
	33	0.024	0.105	0.34	0.33	0.36

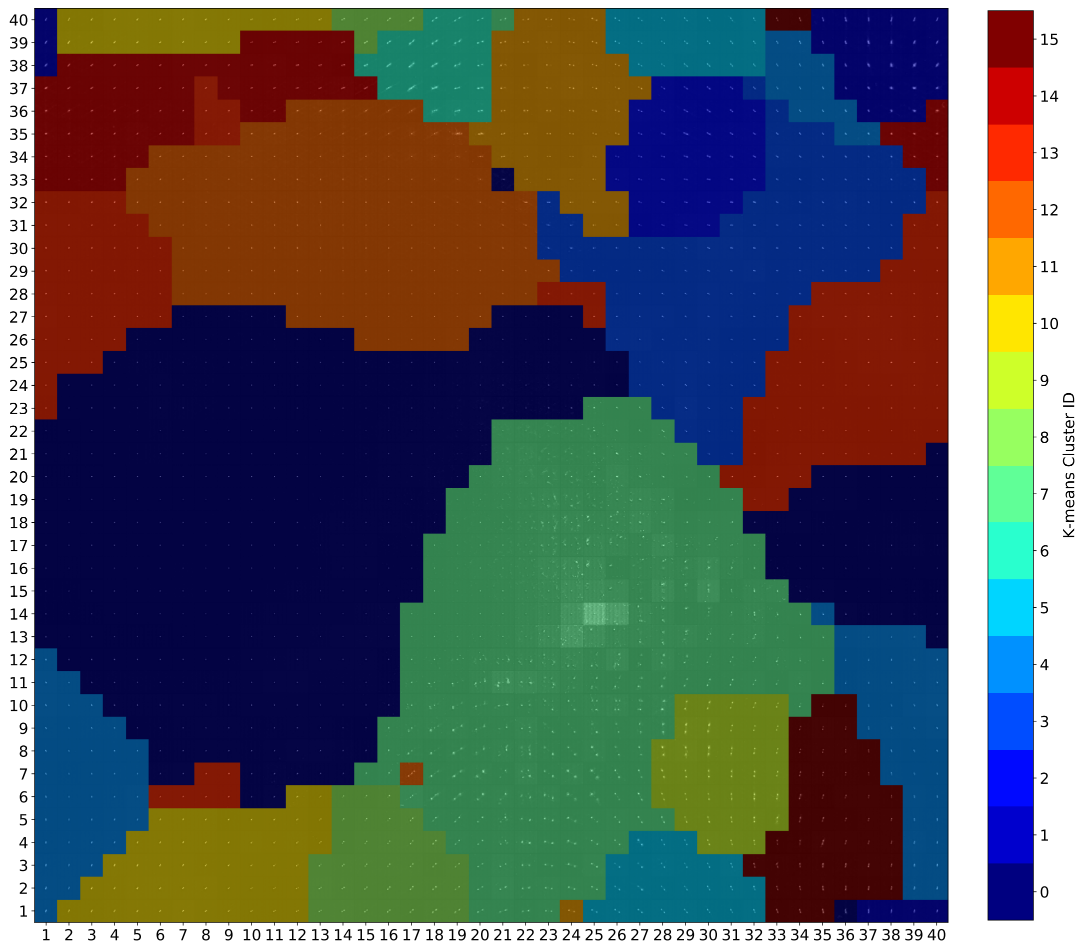

Figure 3.121: 40 × 40 toroidal SOM trained using linear learning rate decay and radius decay, with a Gaussian radius decay, displaying decoded neuron weights with 16 colour coded K-means clusters.

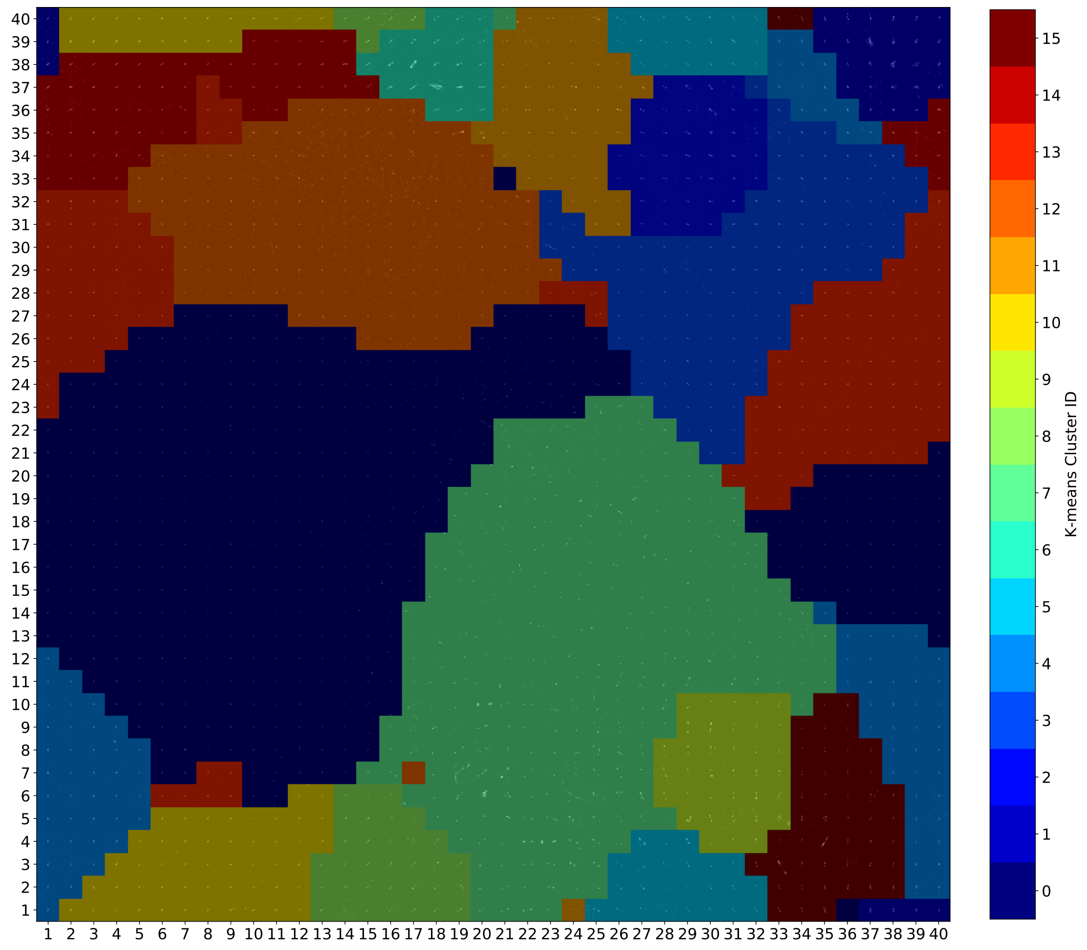

Figure 3.122: 40 × 40 toroidal SOM trained using linear learning rate decay and radius decay, with a Gaussian radius decay, displaying neuron closest matching RGZ images with 16 colour coded K-means clusters.

Table 3.49: Cluster population and entropy statistics for the HC 40×40 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay with 16 K clusters.

K-Means Cluster	Cluster Population Over Map	Minimum Entropy	Maximum Entropy	Mean Entropy
0	0.250	0.00	0.21	0.08
1	0.020	0.03	0.78	0.31
2	0.024	0.06	0.61	0.29
3	0.071	0.00	0.50	0.20
4	0.058	0.01	0.45	0.17
5	0.028	0.14	0.59	0.31
6	0.014	0.28	1.00	0.44
7	0.172	0.00	0.77	0.16
8	0.024	0.12	0.44	0.30
9	0.022	0.18	0.59	0.35
10	0.043	0.01	0.53	0.23
11	0.031	0.11	0.56	0.31
12	0.091	0.00	0.44	0.08
13	0.083	0.00	0.37	0.09
14	0.040	0.00	0.43	0.14
15	0.029	0.14	0.74	0.33

Table 3.50: Divisions of clusters 0-8 based on the label of the closest matching RGZ image, matching clustered label divisions and entropy statistics for each of the 16 K clusters in the HC 40×40 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay

Cluster ID	RGZ Label	Division of Matching Label In Cluster	Division of All Image Labels In Cluster	Mean Entropy	Minimum Entropy	Maximum Entropy
0	12	0.598	0.175	0.21	0.06	0.45
	11	0.250	0.025	0.05	0.01	0.10
	22	0.152	0.042	0.25	0.15	0.44
1	11	0.965	0.426	0.08	0.00	0.15
	12	0.028	0.035	0.15	0.09	0.21
2	11	0.812	0.119	0.07	0.00	0.15
	12	0.180	0.076	0.18	0.08	0.37
3	11	0.516	0.036	0.06	0.00	0.34
	12	0.375	0.076	0.22	0.09	0.43
	22	0.109	0.021	0.24	0.07	0.41
4	12	0.500	0.051	0.26	0.10	0.53
	11	0.219	0.008	0.22	0.03	0.42
	13	0.156	0.385	0.56	0.32	0.78
	22	0.062	0.006	0.25	0.21	0.29
	23	0.031	0.125	0.61	0.61	0.61
	33	0.031	0.053	0.37	0.37	0.37
5	12	0.739	0.162	0.26	0.11	0.53
	11	0.145	0.011	0.06	0.01	0.10
	22	0.116	0.024	0.22	0.14	0.30
6	11	0.986	0.159	0.08	0.00	0.31
7	22	0.895	0.102	0.30	0.12	0.44
	12	0.079	0.010	0.23	0.19	0.30
	33	0.026	0.053	0.36	0.36	0.36
8	11	0.604	0.183	0.09	0.00	0.29
	22	0.313	0.257	0.25	0.08	0.77
	12	0.040	0.035	0.22	0.07	0.55
	33	0.033	0.474	0.32	0.21	0.41

Table 3.51: Divisions of clusters 9-15 based on the label of the closest matching RGZ image, matching clustered label divisions and entropy statistics for each of the 16 K clusters in the HC 40×40 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay

Cluster ID	RGZ Label	Division of Matching Label In Cluster	Division of All Image Labels In Cluster	Mean Entropy	Minimum Entropy	Maximum Entropy
9	22	0.273	0.018	0.37	0.29	0.47
	12	0.227	0.016	0.35	0.29	0.46
	11	0.136	0.003	0.39	0.28	0.55
	13	0.136	0.231	0.59	0.33	1.00
	23	0.091	0.250	0.52	0.50	0.53
	33	0.091	0.105	0.57	0.56	0.58
	14	0.045	0.500	0.54	0.54	0.54
10	22	0.920	0.138	0.30	0.11	0.48
	12	0.060	0.010	0.24	0.13	0.30
	13	0.020	0.077	0.56	0.56	0.56
11	12	0.637	0.229	0.24	0.09	0.44
	11	0.230	0.029	0.06	0.00	0.15
	22	0.124	0.042	0.25	0.17	0.50
12	22	0.795	0.105	0.31	0.14	0.49
	12	0.136	0.019	0.32	0.14	0.59
	33	0.045	0.105	0.38	0.32	0.45
	23	0.023	0.125	0.38	0.38	0.38
13	12	0.564	0.070	0.27	0.15	0.39
	22	0.359	0.042	0.29	0.19	0.39
	11	0.026	0.001	0.06	0.06	0.06
	14	0.026	0.500	0.48	0.48	0.48
	23	0.026	0.125	0.61	0.61	0.61
14	22	0.889	0.096	0.35	0.18	0.59
	33	0.111	0.211	0.32	0.25	0.36
15	22	0.638	0.090	0.31	0.14	0.54
	12	0.255	0.038	0.31	0.25	0.51
	34	0.043	1.000	0.62	0.51	0.74
	13	0.043	0.154	0.49	0.48	0.50
	23	0.021	0.125	0.39	0.39	0.39

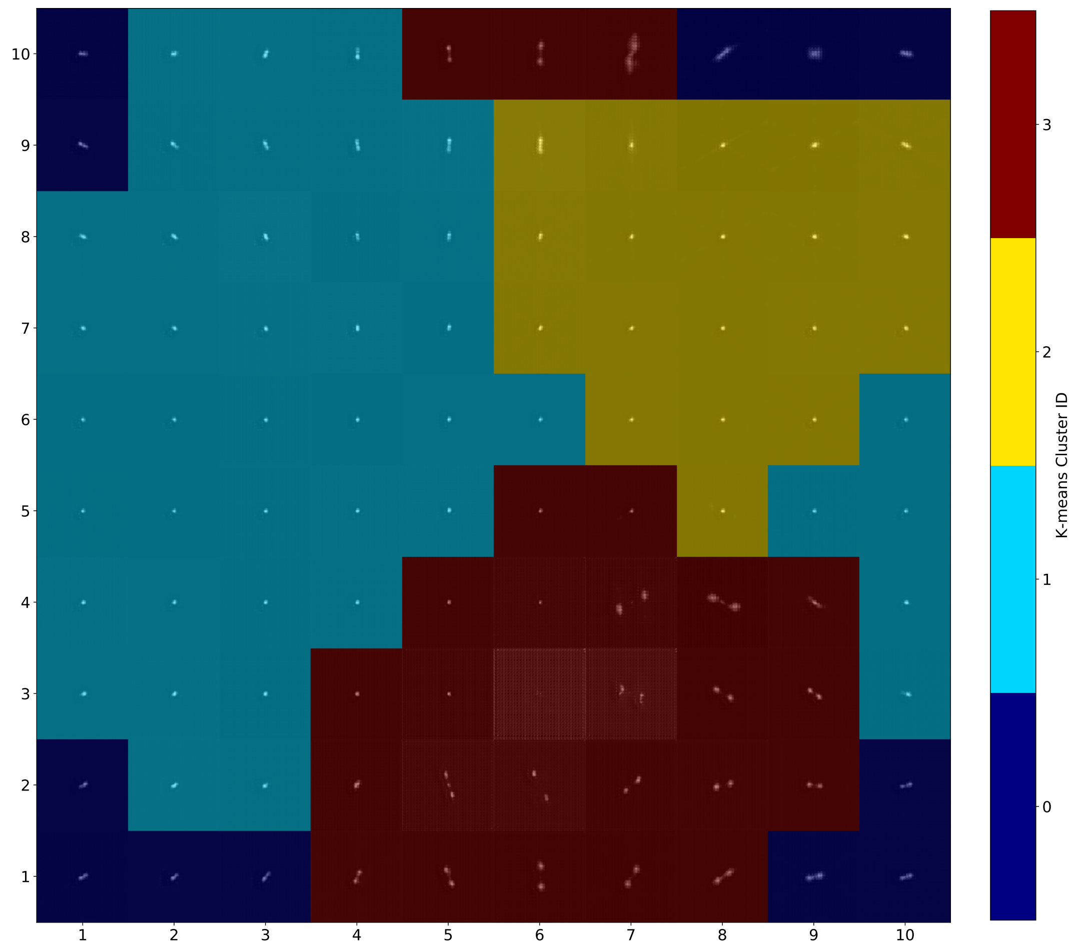

Figure 3.123: 10 × 10 toroidal SOM trained using linear learning rate decay and radius decay, with a Gaussian radius decay, on latent vectors trained with random rotation augmentations. Neurons here display the decoded neuron weights with 4 colour coded K-means clusters.

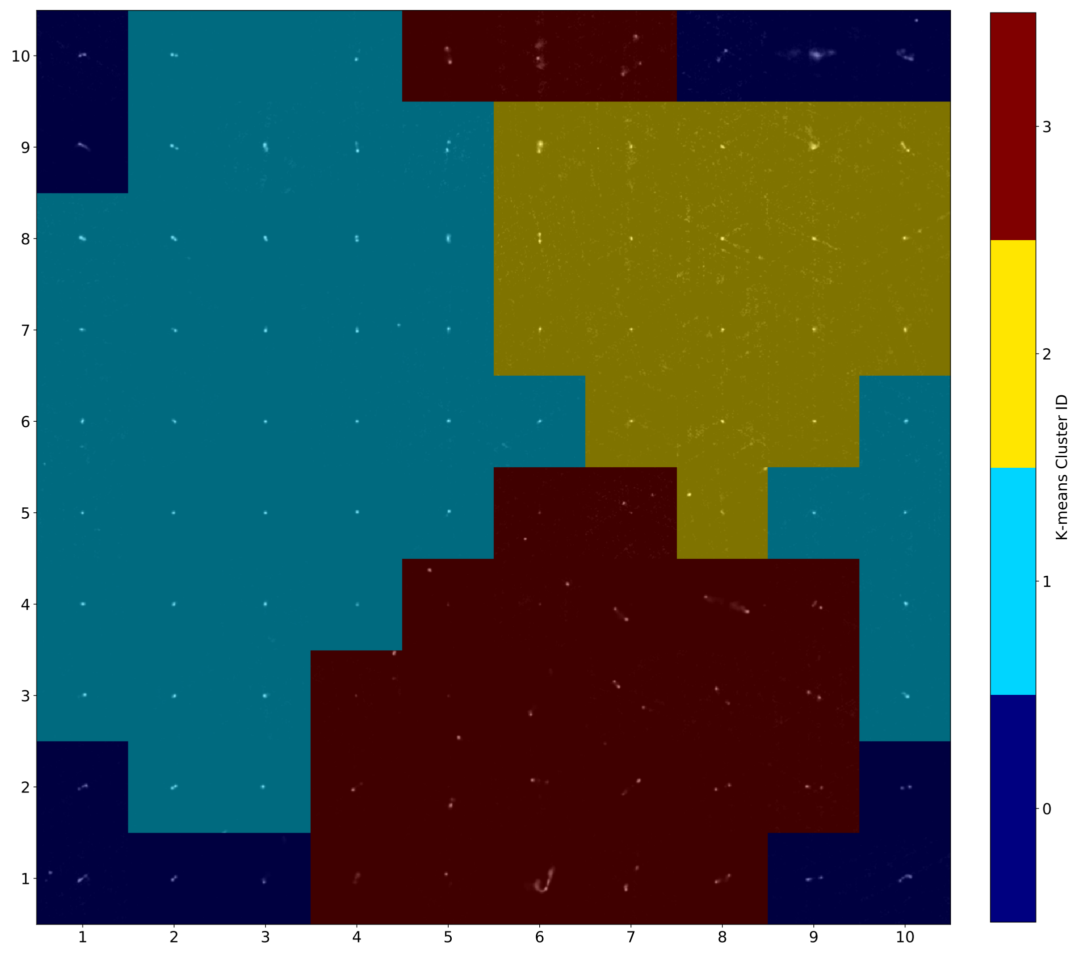

Figure 3.124: 10 × 10 toroidal SOM trained using linear learning rate decay and radius decay, with a Gaussian radius decay, on latent vectors trained with random rotation augmentations. Neurons here display the closest matching RGZ images with 4 colour coded K-means clusters.

Table 3.52: Cluster population and entropy statistics for the HC 10 × 10 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay, on latent vectors trained with random rotation augmentations with 4 K clusters.

K-Means Cluster	Cluster Population Over Map	Minimum Entropy	Maximum Entropy	Mean Entropy
0	0.12	0.31	0.63	0.47
1	0.42	0.00	0.53	0.24
2	0.19	0.00	0.58	0.18
3	0.27	0.03	1.00	0.46

Table 3.53: Divisions of clusters based on the label of the closest matching RGZ image, matching clustered label divisions and entropy statistics for each of the 4 K clusters in the HC 10×10 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay, on latent vectors trained with random rotation augmentations

Cluster ID	RGZ Label	Division of Matching Label In Cluster	Division of All Image Labels In Cluster	Mean Entropy	Minimum Entropy	Maximum Entropy
0	12	0.667	0.471	0.44	0.31	0.57
	22	0.250	0.115	0.61	0.59	0.63
	11	0.083	0.018	0.33	0.33	0.33
1	11	0.762	0.561	0.19	0.00	0.27
	12	0.214	0.529	0.40	0.31	0.53
	22	0.024	0.038	0.38	0.38	0.38
2	22	0.741	0.769	0.58	0.18	1.00
	11	0.259	0.123	0.11	0.03	0.16
3	11	0.895	0.298	0.16	0.00	0.26
	22	0.105	0.077	0.42	0.27	0.58

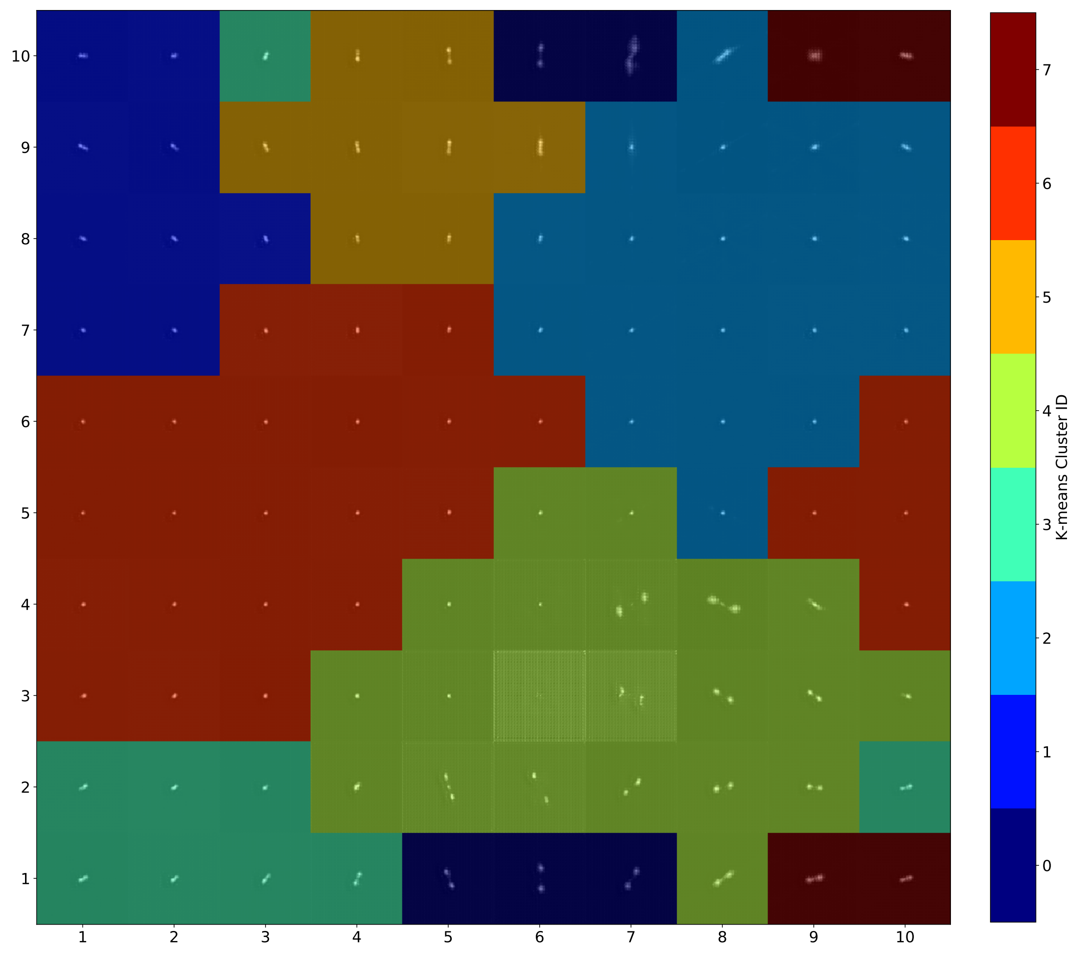

Figure 3.125: 10 × 10 toroidal SOM trained using linear learning rate decay and radius decay, with a Gaussian radius decay, on latent vectors trained with random rotation augmentations. Neurons here display the decoded neuron weights with 8 colour coded K-means clusters.

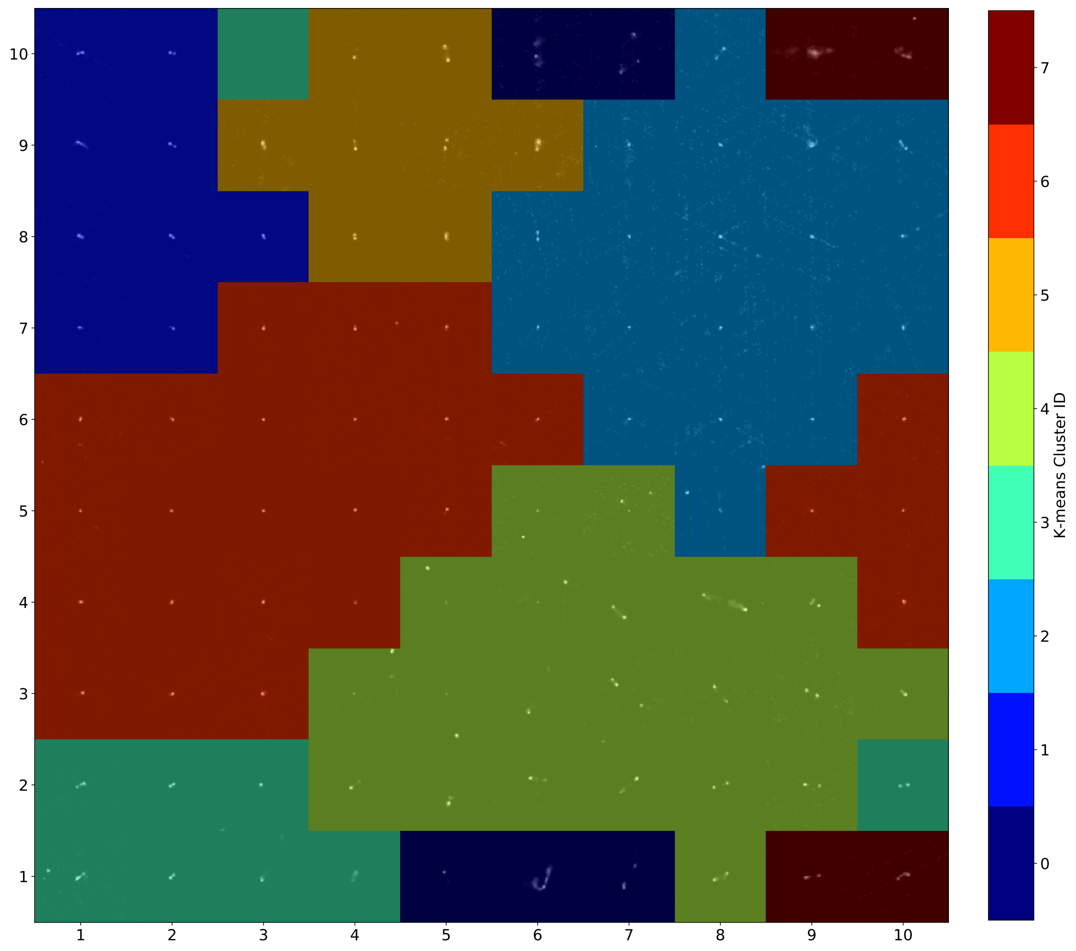

Figure 3.126: 10 × 10 toroidal SOM trained using linear learning rate decay and radius decay, with a Gaussian radius decay, on latent vectors trained with random rotation augmentations. Neurons here display the closest matching RGZ images with 8 colour coded K-means clusters.

Table 3.54: Cluster population and entropy statistics for the HC 10×10 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay, on latent vectors trained with random rotation augmentations with 8 K clusters.

K-Means Cluster	Cluster Population Over Map	Minimum Entropy	Maximum Entropy	Mean Entropy
0	0.05	0.62	1.00	0.73
1	0.09	0.14	0.57	0.30
2	0.19	0.00	0.63	0.19
3	0.09	0.11	0.59	0.38
4	0.21	0.03	0.79	0.38
5	0.08	0.32	0.61	0.46
6	0.25	0.00	0.27	0.19
7	0.04	0.33	0.61	0.45

Table 3.55: Divisions of clusters based on the label of the closest matching RGZ image, matching clustered label divisions and entropy statistics for each of the 8 K clusters in the HC 10×10 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay, on latent vectors trained with random rotation augmentations

Cluster ID	RGZ Label	Division of Matching Label In Cluster	Division of All Image Labels In Cluster	Mean Entropy	Minimum Entropy	Maximum Entropy
0	12	0.556	0.294	0.39	0.31	0.54
	11	0.222	0.035	0.14	0.11	0.16
	22	0.222	0.077	0.57	0.55	0.59
1	11	1.000	0.439	0.19	0.00	0.27
2	11	0.556	0.088	0.18	0.14	0.21
	12	0.444	0.235	0.45	0.33	0.57
3	12	0.625	0.294	0.43	0.32	0.53
	22	0.375	0.115	0.52	0.38	0.61
4	22	0.619	0.500	0.52	0.18	0.79
	11	0.333	0.123	0.11	0.03	0.16
	12	0.048	0.059	0.32	0.32	0.32
5	22	1.000	0.192	0.73	0.62	1.00
6	11	0.895	0.298	0.16	0.00	0.26
	22	0.105	0.077	0.45	0.27	0.63
7	12	0.500	0.118	0.44	0.36	0.51
	11	0.250	0.018	0.33	0.33	0.33
	22	0.250	0.038	0.61	0.61	0.61

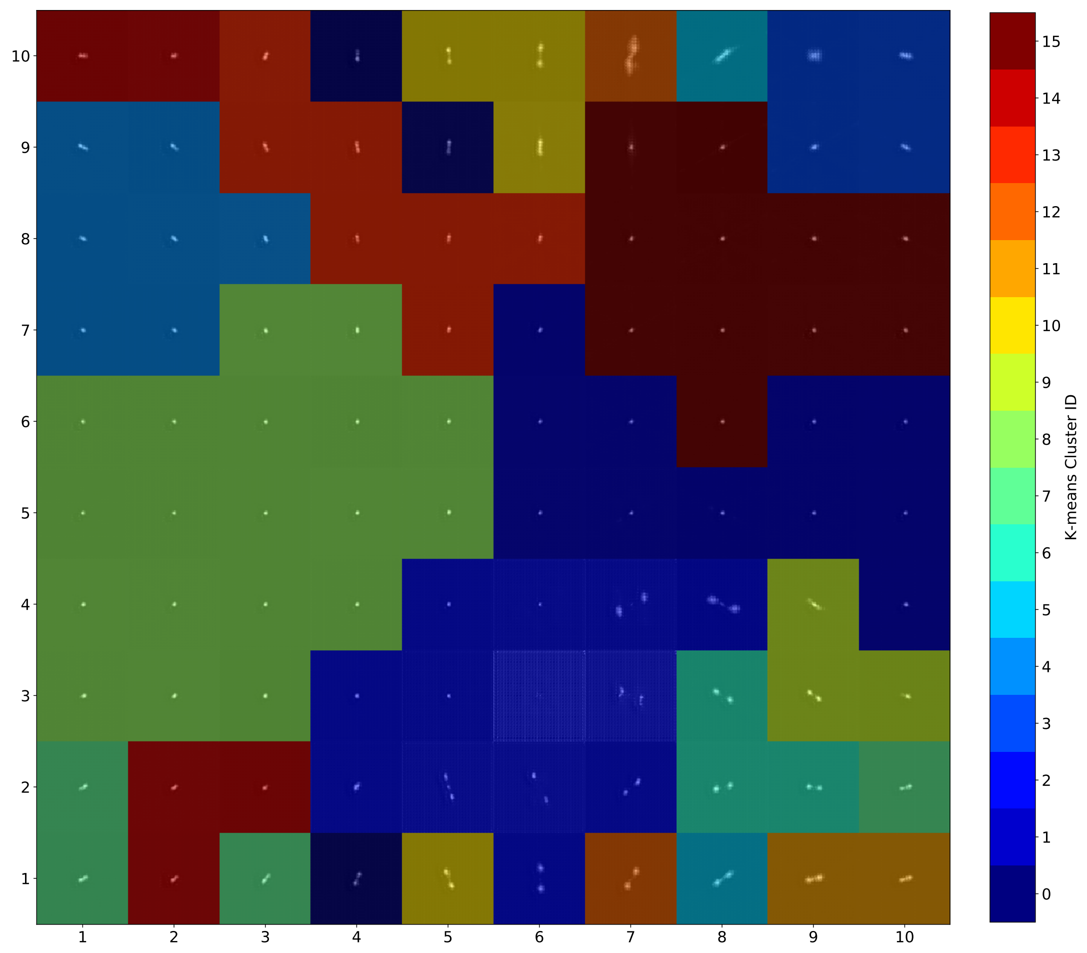

Figure 3.127: 10 × 10 toroidal SOM trained using linear learning rate decay and radius decay, with a Gaussian radius decay, on latent vectors trained with random rotation augmentations. Neurons here display the decoded neuron weights with 16 colour coded K-means clusters.

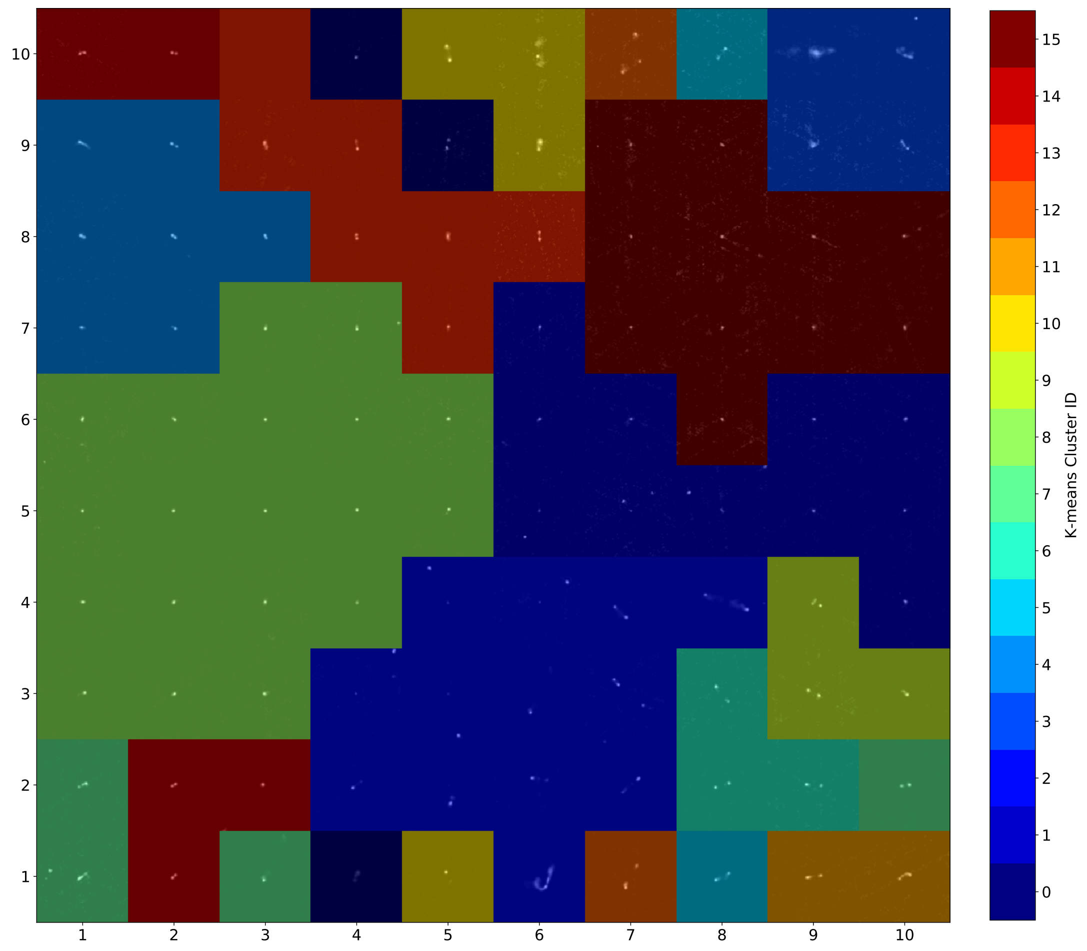

Figure 3.128: 10 × 10 toroidal SOM trained using linear learning rate decay and radius decay, with a Gaussian radius decay, on latent vectors trained with random rotation augmentations. Neurons here display the closest matching RGZ images with 16 colour coded K-means clusters.

Table 3.56: Cluster population and entropy statistics for the HC 10×10 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay, on latent vectors trained with random rotation augmentations with 8 K clusters.

K-Means Cluster	Cluster Population Over Map	Minimum Entropy	Maximum Entropy	Mean Entropy
0	0.03	0.38	0.55	0.49
1	0.11	0.09	0.27	0.20
2	0.13	0.03	0.79	0.36
3	0.04	0.00	0.36	0.24
4	0.07	0.14	0.57	0.30
5	0.02	0.63	0.71	0.67
6	0.03	0.55	0.62	0.60
7	0.04	0.31	0.59	0.44
8	0.19	0.00	0.27	0.19
9	0.03	0.18	0.59	0.36
10	0.04	0.58	0.68	0.62
11	0.02	0.51	0.61	0.56
12	0.02	0.63	1.00	0.82
13	0.07	0.06	0.52	0.30
14	0.05	0.11	0.47	0.26
15	0.11	0.05	0.26	0.17

Table 3.57: Divisions of clusters based on the label of the closest matching RGZ image, matching clustered label divisions and entropy statistics for each of the 16 K clusters in the HC 10×10 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay, on latent vectors trained with random rotation augmentations

Cluster ID	RGZ Label	Division of Matching Label In Cluster	Division of All Image Labels In Cluster	Mean Entropy	Minimum Entropy	Maximum Entropy
0	12	0.750	0.176	0.40	0.31	0.54
	22	0.250	0.038	0.59	0.59	0.59
1	11	1.000	0.333	0.19	0.00	0.27
2	11	0.571	0.070	0.18	0.14	0.21
	12	0.429	0.176	0.47	0.33	0.57
3	11	0.600	0.053	0.15	0.11	0.19
	12	0.400	0.118	0.42	0.38	0.47
4	12	0.714	0.294	0.38	0.31	0.52
	11	0.286	0.035	0.08	0.06	0.09
5	22	0.667	0.077	0.46	0.38	0.55
	12	0.333	0.059	0.53	0.53	0.53
6	22	0.615	0.308	0.53	0.19	0.79
	11	0.385	0.088	0.09	0.03	0.13
7	22	1.000	0.154	0.62	0.58	0.68
8	11	1.000	0.193	0.20	0.09	0.27
9	22	1.000	0.077	0.82	0.63	1.00
10	11	1.000	0.193	0.17	0.05	0.26
11	22	1.000	0.077	0.67	0.63	0.71
12	22	1.000	0.115	0.60	0.55	0.62
13	22	0.500	0.038	0.61	0.61	0.61
	12	0.500	0.059	0.51	0.51	0.51
14	22	0.667	0.077	0.38	0.18	0.59
	12	0.333	0.059	0.32	0.32	0.32
15	11	0.500	0.035	0.16	0.00	0.33
	22	0.250	0.038	0.27	0.27	0.27
	12	0.250	0.059	0.36	0.36	0.36

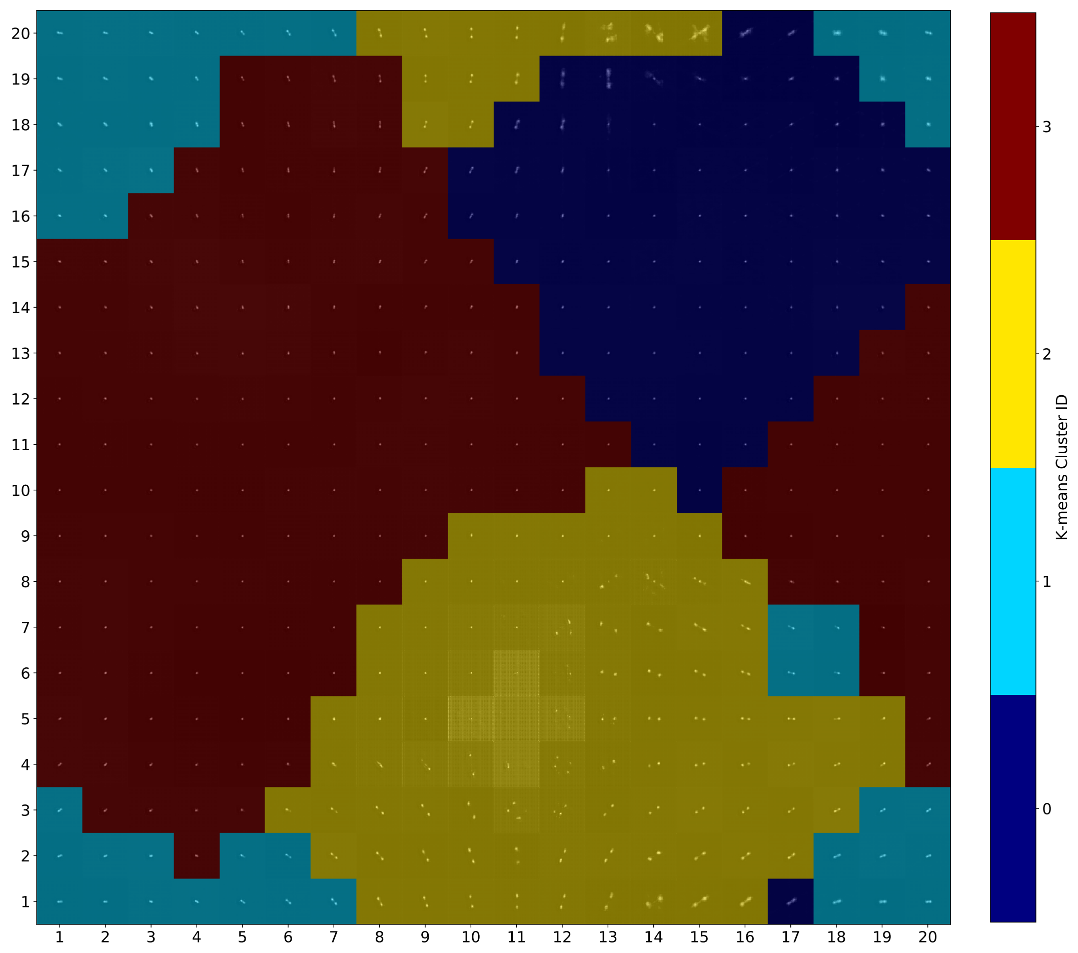

Figure 3.129: 20 × 20 toroidal SOM trained using linear learning rate decay and radius decay, with a Gaussian radius decay, on latent vectors trained with random rotation augmentations. Neurons here display the decoded neuron weights with 4 colour coded K-means clusters.

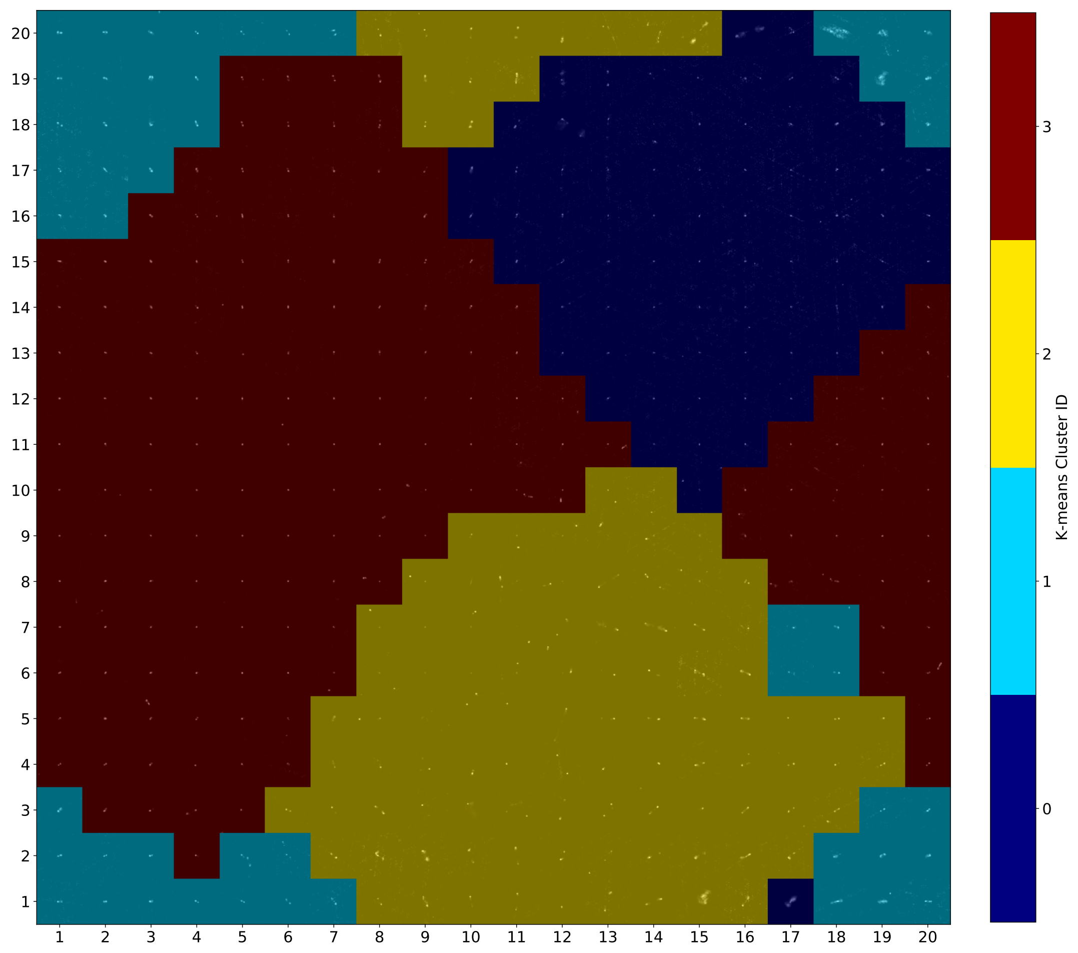

Figure 3.130: 20×20 toroidal SOM trained using linear learning rate decay and radius decay, with a Gaussian radius decay, on latent vectors trained with random rotation augmentations. Neurons here display the closest matching RGZ images with 4 colour coded K-means clusters.

Table 3.58: Cluster population and entropy statistics for the HC 20×20 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay, on latent vectors trained with random rotation augmentations with 4 K clusters.

K-Means Cluster	Cluster Population Over Map	Minimum Entropy	Maximum Entropy	Mean Entropy
0	0.188	0.00	0.93	0.21
1	0.128	0.06	0.73	0.39
2	0.265	0.00	1.00	0.42
3	0.420	0.00	0.76	0.21

Table 3.59: Divisions of clusters based on the label of the closest matching RGZ image, matching clustered label divisions and entropy statistics for each of the 4 K clusters in the HC 20×20 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay, on latent vectors trained with random rotation augmentations

Cluster ID	RGZ Label	Division of Matching Label In Cluster	Division of All Image Labels In Cluster	Mean Entropy	Minimum Entropy	Maximum Entropy
0	12	0.667	0.459	0.40	0.16	0.73
	22	0.176	0.098	0.48	0.30	0.55
	11	0.157	0.035	0.21	0.06	0.50
1	11	0.768	0.561	0.16	0.00	0.22
	12	0.208	0.473	0.35	0.10	0.76
	22	0.024	0.043	0.46	0.26	0.56
2	22	0.642	0.739	0.55	0.22	1.00
	11	0.311	0.143	0.14	0.00	0.41
	33	0.028	0.750	0.72	0.57	0.81
3	11	0.800	0.261	0.14	0.00	0.22
	22	0.147	0.120	0.48	0.28	0.84
	12	0.040	0.041	0.34	0.16	0.51

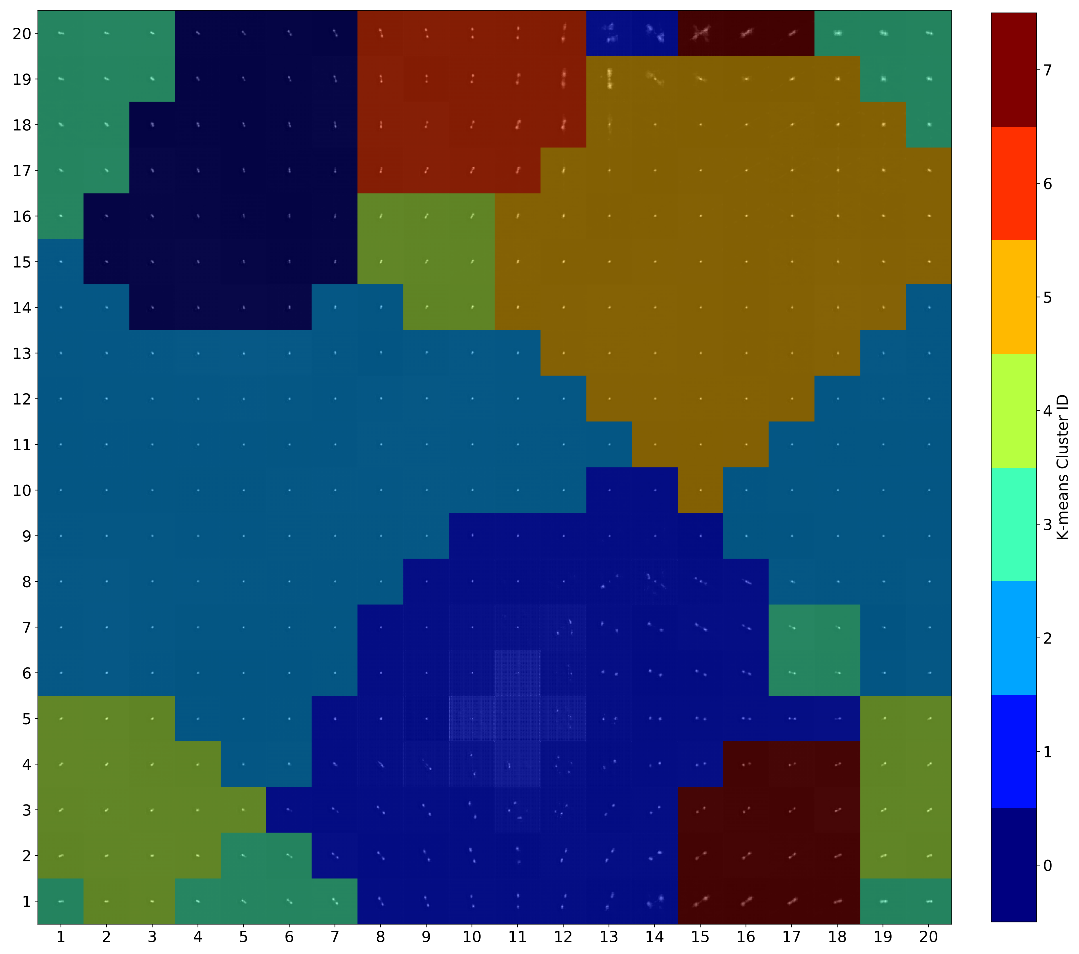

Figure 3.131: 20 × 20 toroidal SOM trained using linear learning rate decay and radius decay, with a Gaussian radius decay, on latent vectors trained with random rotation augmentations. Neurons here display the decoded neuron weights with 8 colour coded K-means clusters.

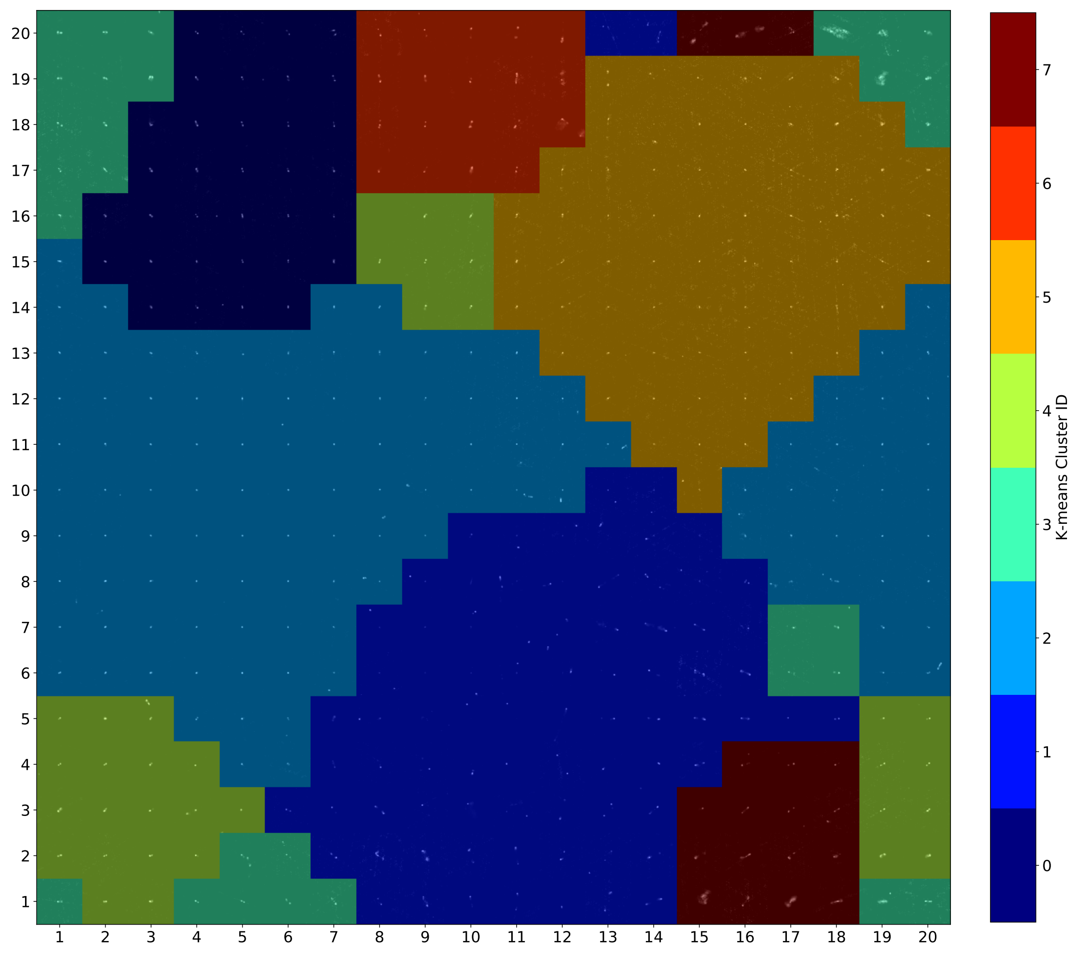

Figure 3.132: 20 × 20 toroidal SOM trained using linear learning rate decay and radius decay, with a Gaussian radius decay, on latent vectors trained with random rotation augmentations. Neurons here display the closest matching RGZ images with 8 colour coded K-means clusters.

Table 3.60: Cluster population and entropy statistics for the HC 20×20 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay, on latent vectors trained with random rotation augmentations with 8 K clusters.

K-Means Cluster	Cluster Population Over Map	Minimum Entropy	Maximum Entropy	Mean Entropy
0	0.085	0.07	0.76	0.31
1	0.202	0.00	0.99	0.37
2	0.292	0.00	0.27	0.17
3	0.075	0.06	0.73	0.39
4	0.085	0.01	0.61	0.31
5	0.168	0.00	0.58	0.16
6	0.048	0.40	1.00	0.60
7	0.045	0.39	0.83	0.60

Table 3.61: Divisions of clusters based on the label of the closest matching RGZ image, matching clustered label divisions and entropy statistics for each of the 8 K clusters in the HC 20×20 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay, on latent vectors trained with random rotation augmentations

Cluster ID	RGZ Label	Division of Matching Label In Cluster	Division of All Image Labels In Cluster	Mean Entropy	Minimum Entropy	Maximum Entropy
0	12	0.633	0.257	0.40	0.16	0.73
	22	0.200	0.065	0.48	0.30	0.55
	11	0.167	0.022	0.24	0.06	0.50
1	12	0.735	0.338	0.37	0.18	0.61
	11	0.206	0.030	0.10	0.01	0.20
	22	0.059	0.022	0.37	0.32	0.42
2	11	0.966	0.491	0.17	0.00	0.22
	12	0.034	0.054	0.18	0.10	0.27
3	12	0.559	0.257	0.39	0.17	0.76
	11	0.324	0.048	0.13	0.07	0.22
	22	0.118	0.043	0.42	0.26	0.51
4	22	0.543	0.478	0.53	0.22	0.99
	11	0.407	0.143	0.14	0.00	0.41
	33	0.025	0.500	0.68	0.57	0.78
	12	0.025	0.027	0.24	0.17	0.30
5	22	0.842	0.174	0.59	0.40	1.00
	12	0.105	0.027	0.48	0.47	0.49
	33	0.053	0.250	0.93	0.93	0.93
6	11	0.910	0.265	0.14	0.00	0.22
	22	0.075	0.054	0.38	0.28	0.58
7	22	0.833	0.163	0.59	0.39	0.83
	12	0.111	0.027	0.56	0.51	0.61
	33	0.056	0.250	0.81	0.81	0.81

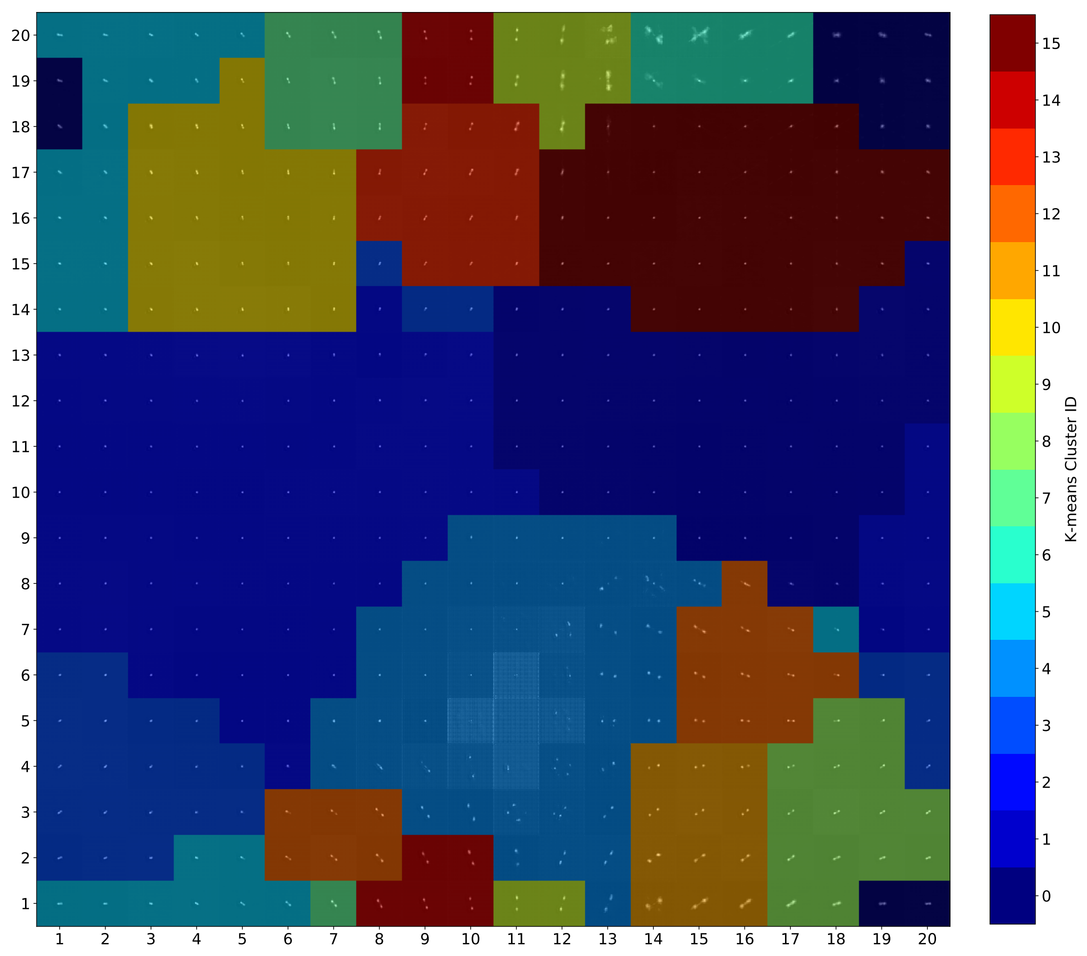

Figure 3.133: 20 × 20 toroidal SOM trained using linear learning rate decay and radius decay, with a Gaussian radius decay, on latent vectors trained with random rotation augmentations. Neurons here display the decoded neuron weights with 16 colour coded K-means clusters.

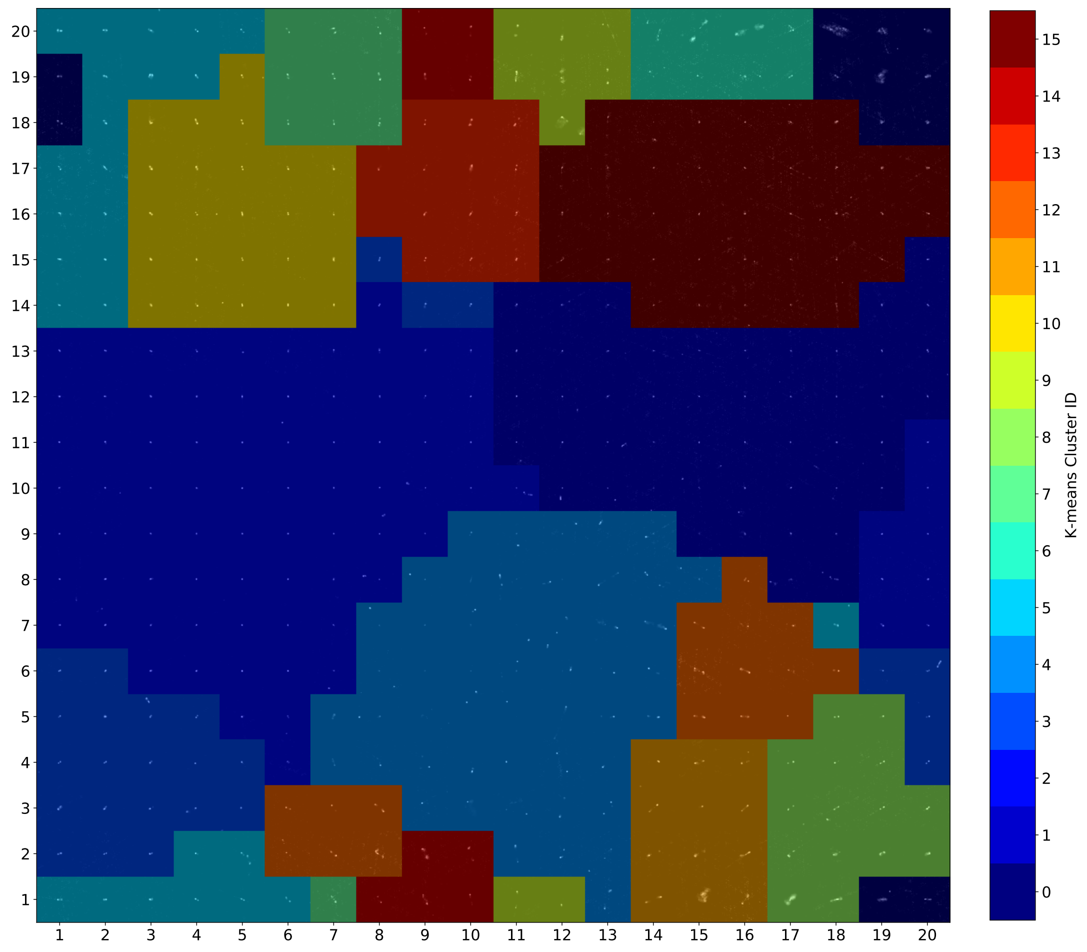

Figure 3.134: 20×20 toroidal SOM trained using linear learning rate decay and radius decay, with a Gaussian radius decay, on latent vectors trained with random rotation augmentations. Neurons here display the closest matching RGZ images with 16 colour coded K-means clusters.

Table 3.62: Cluster population and entropy statistics for the HC 20×20 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay, on latent vectors trained with random rotation augmentations with 16 K clusters.

K-Means Cluster	Cluster Population Over Map	Minimum Entropy	Maximum Entropy	Mean Entropy
0	0.030	0.02	0.50	0.28
1	0.122	0.00	0.22	0.17
2	0.205	0.02	0.22	0.17
3	0.065	0.01	0.59	0.24
4	0.125	0.00	0.70	0.28
5	0.065	0.06	0.73	0.34
6	0.020	0.10	0.99	0.44
7	0.025	0.41	0.68	0.50
8	0.038	0.18	0.64	0.47
9	0.022	0.48	1.00	0.71
10	0.060	0.07	0.76	0.25
11	0.030	0.38	0.88	0.64
12	0.042	0.14	0.67	0.52
13	0.035	0.10	0.73	0.42
14	0.022	0.40	0.68	0.53
15	0.092	0.00	0.36	0.14

Table 3.63: Divisions of clusters based on the label of the closest matching RGZ image, matching clustered label divisions and entropy statistics for each of the 16 K clusters in the HC 20×20 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay, on latent vectors trained with random rotation augmentations

Cluster ID	RGZ Label	Division of Matching Label In Cluster	Division of All Image Labels In Cluster	Mean Entropy	Minimum Entropy	Maximum Entropy
0	12	0.692	0.243	0.43	0.21	0.73
	11	0.269	0.030	0.12	0.06	0.20
	22	0.038	0.011	0.30	0.30	0.30
1	12	0.615	0.216	0.33	0.15	0.59
	11	0.385	0.043	0.09	0.01	0.20
2	11	0.976	0.348	0.17	0.02	0.22
	12	0.024	0.027	0.16	0.10	0.22
3	11	0.500	0.026	0.20	0.02	0.50
	12	0.417	0.068	0.33	0.16	0.45
	22	0.083	0.011	0.49	0.49	0.49
4	12	0.542	0.176	0.35	0.17	0.76
	11	0.417	0.043	0.12	0.07	0.22
	22	0.042	0.011	0.26	0.26	0.26
5	22	0.941	0.174	0.54	0.22	0.67
	11	0.059	0.004	0.14	0.14	0.14
6	22	0.600	0.065	0.54	0.43	0.68
	12	0.400	0.054	0.45	0.41	0.49
7	11	0.580	0.126	0.15	0.00	0.41
	22	0.360	0.196	0.49	0.33	0.70
	12	0.040	0.027	0.24	0.17	0.30
	33	0.020	0.250	0.57	0.57	0.57

Table 3.64: Accuracy and performance metrics of anomaly detection by separating neuron matches sources based on the Euclidean distance of neurons a 10 × 10 toroidal SOM umat at each SD 1-5, trained with a linear learning rate decay, radius function and radius decay

Cluster ID	RGZ Label	Division of Matching Label In Cluster	Division of All Image Labels In Cluster	Mean Entropy	Minimum Entropy	Maximum Entropy
8	22	1.000	0.098	0.53	0.40	0.68
9	12	0.571	0.108	0.38	0.16	0.61
	22	0.357	0.054	0.54	0.47	0.73
	11	0.071	0.004	0.10	0.10	0.10
10	22	0.778	0.076	0.67	0.48	1.00
	33	0.222	0.500	0.86	0.78	0.93
11	11	1.000	0.213	0.17	0.00	0.22
12	11	0.973	0.157	0.13	0.00	0.22
	22	0.027	0.011	0.36	0.36	0.36
13	22	0.917	0.120	0.62	0.38	0.88
	33	0.083	0.250	0.81	0.81	0.81
14	22	0.875	0.076	0.49	0.28	0.99
	11	0.125	0.004	0.10	0.10	0.10
15	22	0.600	0.098	0.48	0.32	0.64
	12	0.400	0.081	0.46	0.18	0.61

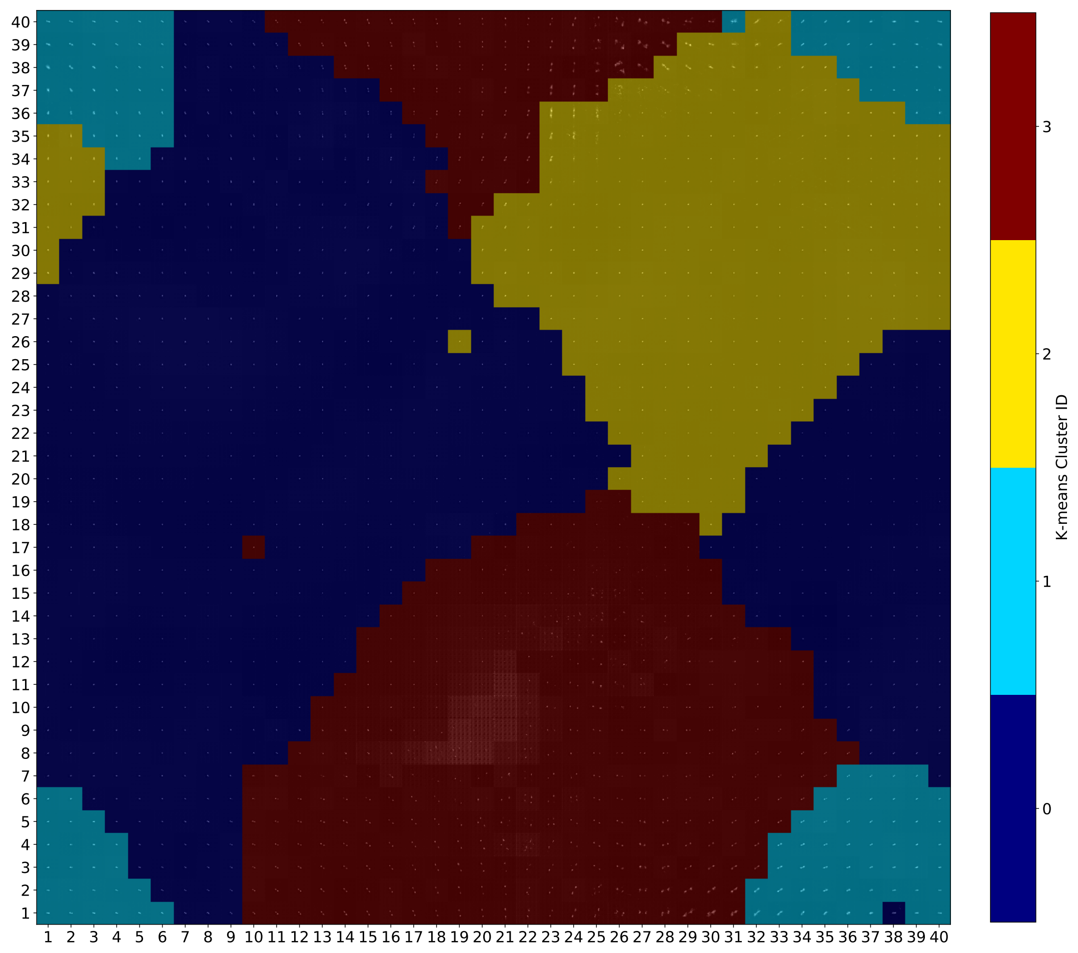

Figure 3.135: 40 × 40 toroidal SOM trained using linear learning rate decay and radius decay, with a Gaussian radius decay, on latent vectors trained with random rotation augmentations. Neurons here display the decoded neuron weights with 4 colour coded K-means clusters.

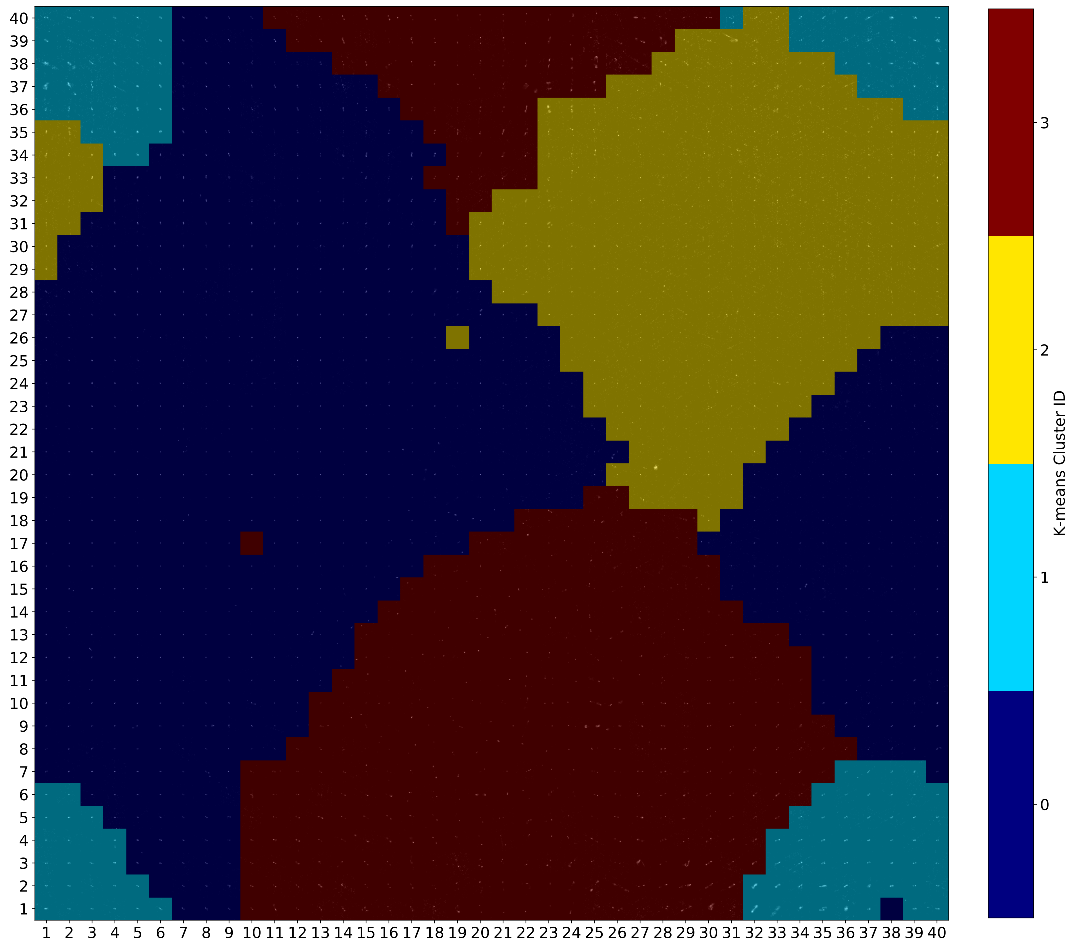

Figure 3.136: 40 × 40 toroidal SOM trained using linear learning rate decay and radius decay, with a Gaussian radius decay, on latent vectors trained with random rotation augmentations. Neurons here display the closest matching RGZ images with 4 colour coded K-means clusters.

Table 3.65: Cluster population and entropy statistics for the HC 40 × 40 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay, on latent vectors trained with random rotation augmentations with 4 K clusters.

K-Means Cluster	Cluster Population Over Map	Minimum Entropy	Maximum Entropy	Mean Entropy
0	0.446	0.00	0.60	0.13
1	0.085	0.01	0.75	0.23
2	0.192	0.00	0.54	0.10
3	0.277	0.00	1.00	0.24

Table 3.66: Divisions of clusters based on the label of the closest matching RGZ image, matching clustered label divisions and entropy statistics for each of the 4 K clusters in the HC 40×40 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay, on latent vectors trained with random rotation augmentations

Cluster ID	RGZ Label	Division of Matching Label In Cluster	Division of All Image Labels In Cluster	Mean Entropy	Minimum Entropy	Maximum Entropy
0	12	0.529	0.263	0.24	0.08	0.54
	11	0.243	0.034	0.11	0.01	0.37
	22	0.176	0.077	0.29	0.13	0.66
	13	0.044	0.600	0.46	0.24	0.75
1	11	0.710	0.518	0.09	0.00	0.36
	12	0.234	0.609	0.24	0.06	0.48
	22	0.052	0.118	0.27	0.07	0.46
2	11	0.883	0.277	0.08	0.00	0.22
	22	0.091	0.089	0.28	0.09	0.54
	12	0.023	0.026	0.10	0.09	0.11
3	22	0.506	0.716	0.31	0.07	0.89
	11	0.377	0.171	0.11	0.00	1.00
	12	0.063	0.102	0.25	0.10	0.45
	33	0.036	0.941	0.43	0.27	0.82

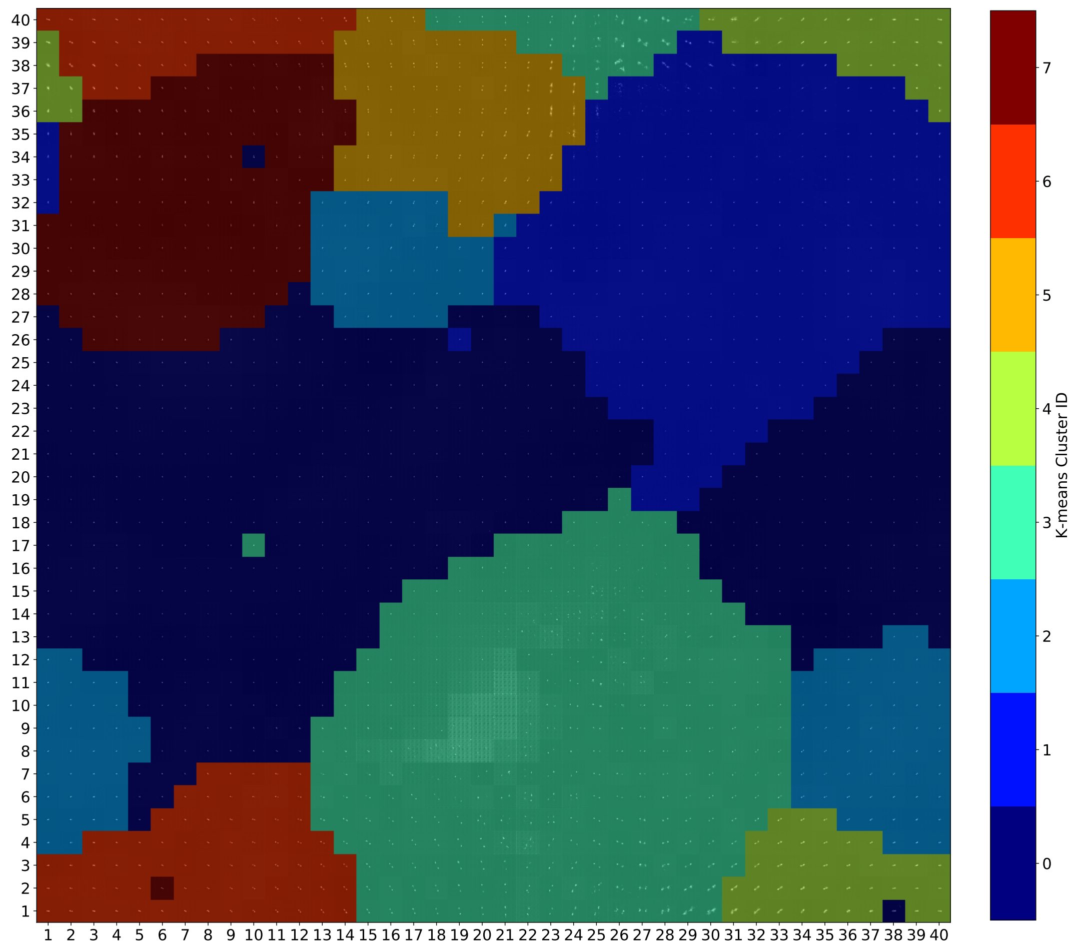

Figure 3.137: 40 × 40 toroidal SOM trained using linear learning rate decay and radius decay, with a Gaussian radius decay, on latent vectors trained with random rotation augmentations. Neurons here display the decoded neuron weights with 8 colour coded K-means clusters.

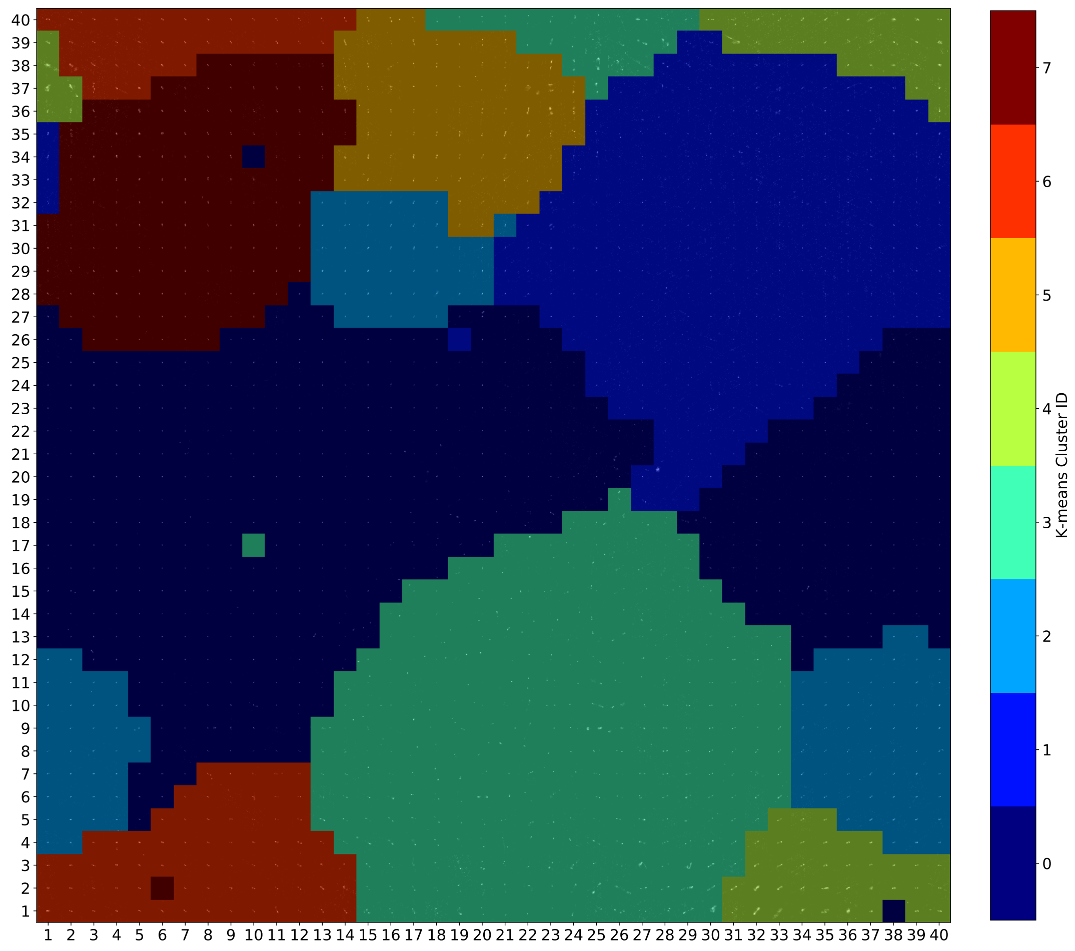

Figure 3.138: 40 × 40 toroidal SOM trained using linear learning rate decay and radius decay, with a Gaussian radius decay, on latent vectors trained with random rotation augmentations. Neurons here display the closest matching RGZ images with 8 colour coded K-means clusters.

Table 3.67: Cluster population and entropy statistics for the HC 40×40 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay, on latent vectors trained with random rotation augmentations with 8 K clusters.

K-Means Cluster	Cluster Population Over Map	Minimum Entropy	Maximum Entropy	Mean Entropy
0	0.298	0.00	0.28	0.09
1	0.169	0.00	0.43	0.09
2	0.084	0.01	0.46	0.19
3	0.206	0.00	1.00	0.22
4	0.045	0.01	0.75	0.28
5	0.049	0.05	0.88	0.33
6	0.066	0.06	0.46	0.26
7	0.085	0.00	0.54	0.18

Table 3.68: Divisions of clusters based on the label of the closest matching RGZ image, matching clustered label divisions and entropy statistics for each of the 8 K clusters in the HC 40 × 40 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay, on latent vectors trained with random rotation augmentations

Cluster ID	RGZ Label	Division of Matching Label In Cluster	Division of All Image Labels In Cluster	Mean Entropy	Minimum Entropy	Maximum Entropy
0	12	0.638	0.245	0.27	0.08	0.46
	22	0.314	0.105	0.29	0.11	0.46
	11	0.048	0.005	0.08	0.06	0.12
1	12	0.619	0.303	0.22	0.06	0.46
	11	0.239	0.033	0.07	0.01	0.19
	22	0.142	0.061	0.25	0.13	0.46
2	11	0.960	0.467	0.09	0.00	0.24
	12	0.038	0.066	0.14	0.06	0.28
3	12	0.478	0.237	0.25	0.08	0.54
	11	0.397	0.055	0.07	0.00	0.36
	22	0.118	0.051	0.25	0.07	0.35
4	11	0.930	0.257	0.08	0.00	0.22
	22	0.056	0.048	0.24	0.09	0.43
5	12	0.361	0.095	0.27	0.09	0.47
	11	0.278	0.020	0.12	0.01	0.37
	22	0.236	0.054	0.34	0.13	0.66
	13	0.083	0.600	0.46	0.24	0.75
	23	0.028	0.500	0.50	0.49	0.52
6	11	0.474	0.160	0.11	0.00	1.00
	22	0.444	0.466	0.31	0.07	0.89
	33	0.046	0.882	0.41	0.27	0.82
	12	0.024	0.029	0.19	0.12	0.28
7	22	0.846	0.211	0.33	0.15	0.54
	12	0.051	0.015	0.30	0.20	0.39
	11	0.038	0.003	0.12	0.05	0.19
	13	0.038	0.300	0.49	0.35	0.60

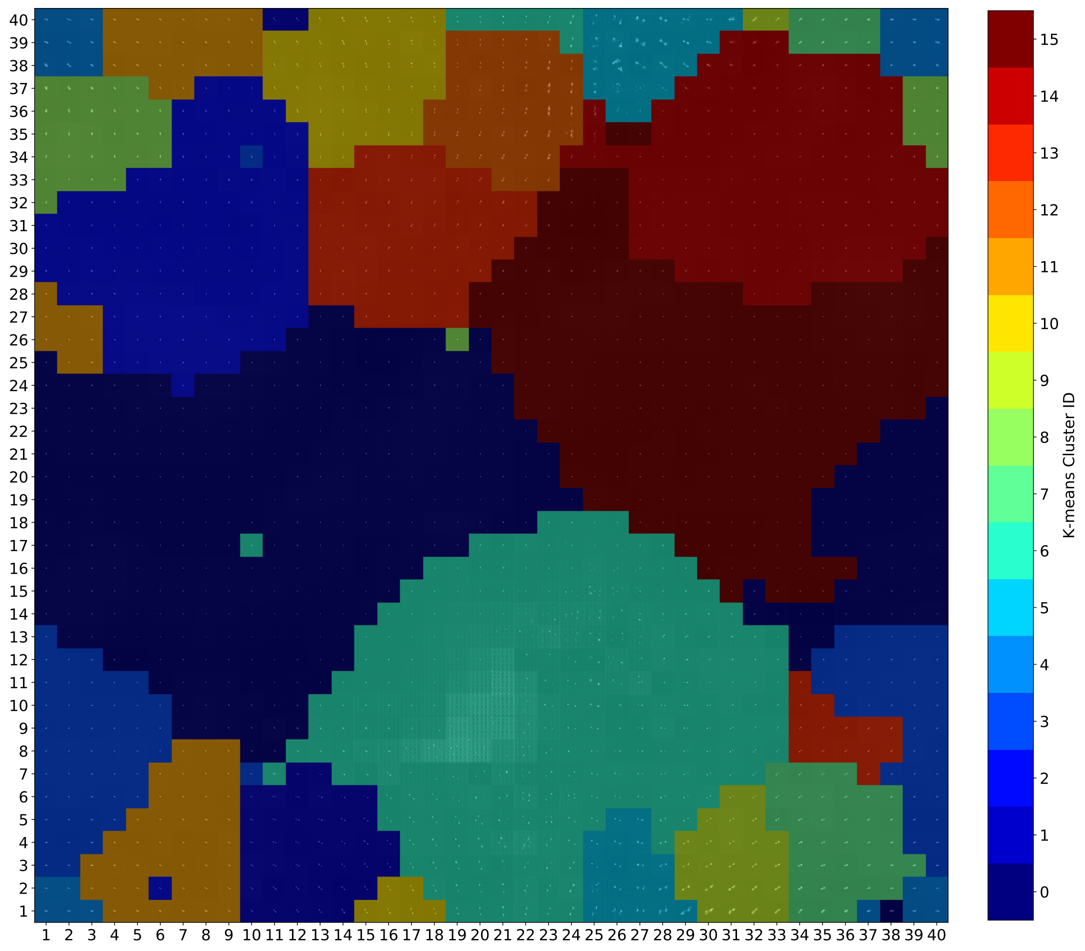

Figure 3.139: 40 × 40 toroidal SOM trained using linear learning rate decay and radius decay, with a Gaussian radius decay, on latent vectors trained with random rotation augmentations. Neurons here display the decoded neuron weights with 16 colour coded K-means clusters.

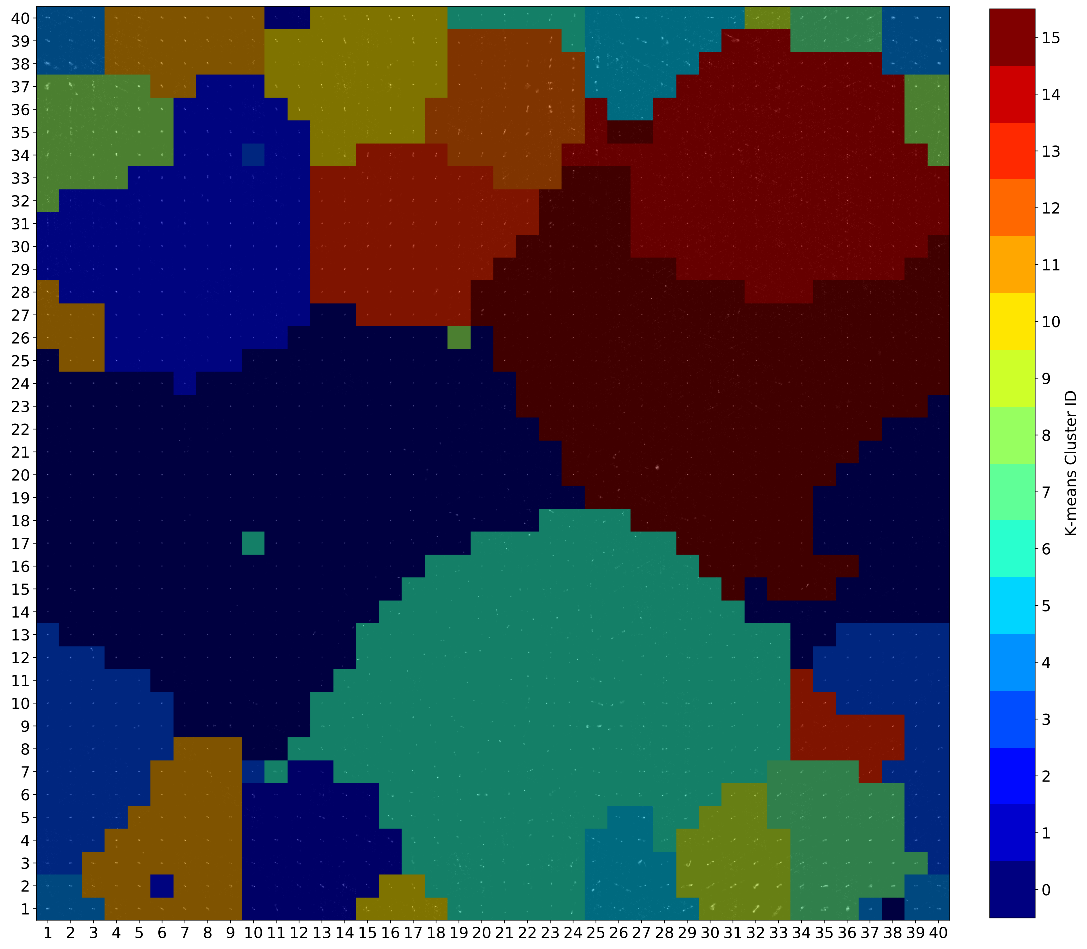

Figure 3.140: 40 × 40 toroidal SOM trained using linear learning rate decay and radius decay, with a Gaussian radius decay, on latent vectors trained with random rotation augmentations. Neurons here display the closest matching RGZ images with 16 colour coded K-means clusters.

Table 3.69: Cluster population and entropy statistics for the HC 40×40 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay, on latent vectors trained with random rotation augmentations with 16 K clusters.

K-Means Cluster	Cluster Population Over Map	Minimum Entropy	Maximum Entropy	Mean Entropy
0	0.214	0.00	0.24	0.09
1	0.026	0.11	0.46	0.30
2	0.069	0.00	0.50	0.15
3	0.052	0.00	0.46	0.14
4	0.018	0.03	0.49	0.24
5	0.026	0.00	1.00	0.37
6	0.164	0.00	0.89	0.18
7	0.027	0.04	0.56	0.27
8	0.022	0.00	0.54	0.16
9	0.016	0.19	0.75	0.41
10	0.031	0.16	0.51	0.31
11	0.046	0.03	0.46	0.23
12	0.024	0.05	0.88	0.35
13	0.047	0.01	0.60	0.24
14	0.081	0.01	0.43	0.09
15	0.138	0.00	0.17	0.09

Table 3.70: Divisions of clusters 0-5 based on the label of the closest matching RGZ image, matching clustered label divisions and entropy statistics for each of the 16 K clusters in the HC 40×40 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay, on latent vectors trained with random rotation augmentations

Cluster ID	RGZ Label	Division of Matching Label In Cluster	Division of All Image Labels In Cluster	Mean Entropy	Minimum Entropy	Maximum Entropy
0	12	0.571	0.058	0.23	0.12	0.35
	11	0.179	0.005	0.12	0.03	0.29
	22	0.143	0.013	0.33	0.28	0.40
	13	0.107	0.300	0.41	0.29	0.49
1	12	0.512	0.157	0.19	0.07	0.46
	11	0.429	0.037	0.07	0.00	0.19
	22	0.060	0.016	0.17	0.13	0.22
2	11	0.974	0.342	0.09	0.00	0.24
	12	0.023	0.029	0.14	0.06	0.21
3	12	0.699	0.186	0.27	0.08	0.46
	11	0.205	0.015	0.08	0.03	0.12
	22	0.096	0.022	0.24	0.17	0.34
4	11	0.473	0.053	0.06	0.00	0.36
	12	0.473	0.190	0.23	0.08	0.48
	22	0.045	0.016	0.21	0.07	0.35
5	11	0.500	0.018	0.10	0.00	0.32
	22	0.250	0.029	0.21	0.09	0.31
	12	0.222	0.029	0.23	0.09	0.54
	13	0.028	0.100	0.24	0.24	0.24

Table 3.71: Divisions of clusters 6-10 based on the label of the closest matching RGZ image, matching clustered label divisions and entropy statistics for each of the 16 K clusters in the HC 40×40 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay, on latent vectors trained with random rotation augmentations

Cluster ID	RGZ Label	Division of Matching Label In Cluster	Division of All Image Labels In Cluster	Mean Entropy	Minimum Entropy	Maximum Entropy
6	22	0.683	0.089	0.30	0.11	0.46
	12	0.317	0.047	0.29	0.18	0.45
7	11	0.573	0.153	0.10	0.00	0.35
	22	0.370	0.310	0.29	0.07	0.89
	33	0.027	0.412	0.32	0.27	0.39
	12	0.027	0.026	0.19	0.12	0.29
8	22	0.837	0.131	0.32	0.16	0.46
	12	0.122	0.022	0.26	0.20	0.31
	13	0.020	0.100	0.51	0.51	0.51
	33	0.020	0.059	0.39	0.39	0.39
9	12	0.560	0.153	0.23	0.06	0.46
	22	0.320	0.077	0.27	0.13	0.39
	11	0.093	0.007	0.07	0.01	0.14
	13	0.027	0.200	0.48	0.35	0.60
10	22	0.846	0.105	0.35	0.15	0.54
	11	0.077	0.003	0.12	0.05	0.19
	44	0.026	0.500	0.88	0.88	0.88
	33	0.026	0.059	0.66	0.66	0.66
	12	0.026	0.004	0.39	0.39	0.39

Table 3.72: Divisions of clusters 11-15 based on the label of the closest matching RGZ image, matching clustered label divisions and entropy statistics for each of the 16 K clusters in the HC 40×40 toroidal SOM umat trained with linear learning rate decay and radius decay with a Gaussian radius decay, on latent vectors trained with random rotation augmentations

Cluster ID	RGZ Label	Division of Matching Label In Cluster	Division of All Image Labels In Cluster	Mean Entropy	Minimum Entropy	Maximum Entropy
11	11	0.991	0.223	0.09	0.00	0.11
12	11	0.961	0.127	0.08	0.01	0.22
	22	0.039	0.016	0.24	0.09	0.43
13	22	0.571	0.077	0.40	0.17	0.67
	11	0.214	0.009	0.18	0.00	1.00
	33	0.167	0.412	0.51	0.33	0.82
	23	0.024	0.250	0.39	0.39	0.39
	24	0.024	1.000	0.62	0.62	0.62
14	22	0.615	0.051	0.37	0.27	0.66
	11	0.115	0.003	0.29	0.19	0.37
	23	0.115	0.750	0.49	0.47	0.52
	44	0.038	0.500	0.70	0.70	0.70
	13	0.038	0.100	0.75	0.75	0.75
	33	0.038	0.059	0.38	0.38	0.38
	12	0.038	0.004	0.47	0.47	0.47
15	12	0.581	0.091	0.27	0.12	0.46
	22	0.302	0.042	0.29	0.13	0.46
	11	0.093	0.004	0.10	0.04	0.17
	13	0.023	0.100	0.56	0.56	0.56

Table 3.73: HC processing time metrics of the toroidal 10, 20 and 40 square SOM with a linear learning rate decay and radius decay and a Gaussian neighbourhood function compared to SOMs training on latent vectors produced by an autoencoder with rotation augmentation training.

Map Size	Latent Vector Training Augmentations	K-Means Clusters	Clustering Time (s)
10x10	None	4	0.042
		8	0.061
		16	0.090
20x20	None	4	0.136
		8	0.192
		16	0.241
40×40	None	4	0.781
		8	0.626
		16	1.180
10×10	Rotation	4	0.050
		8	0.048
		16	0.098
20×20	Rotation	4	0.137
		8	0.186
		16	0.224
40×40	Rotation	4	0.609
		8	0.884
		16	1.136

3.3.4 Overview of Optimised Clustered Self-Organised Map

In the optimised network testing, I trialled four square toroidal SOM sizes; 5×5, 10×10, 20×20 and 40×40. These maps were trained for 12 epochs using 30,000 RGZ images for training and 30,000 RGZ images for validation. Results of this Chapter demonstrate that the maps can successfully visualise similarity arranged latent relationships, features and morphologies of the radio-astronomical images captured in encoded RGZ data, with clear improvement over the preliminary tests. Neuron weight decoding performed well, with the learned weights as latent vectors converted back into the learned RGZ dataset image and showing the true learned features of each neuron's weight vector. The decoded neuron images showed several rotated average morphologies which are mostly removed by training the SOM training on latent vectors produced with an autoencoder trained on random rotations.

Test results indicate that with the initial conditions outlined in Section 3.3.3, a SOM with the following configuration is the most accurate in terms of complexity separation:

- Linear learning rate decay function
- Gaussian learning neighbourhood
- Linear neighbourhood radius decay function
- Trained on latent vector representations without random rotation augmentations

These results suggested that complexity separation accuracy was reasonably robust to changes in map size; however the best radio-astronomical image feature and morphology separations can be found in larger map sizes than the 5×5 with 10×10, 20×20 and 40×40 map sizes. The training time for these maps is negligible, with at most 92 seconds for the largest 40×40 map tests on the hardware specified in Section 2.1. These configurations were used in the subsequent HC tests and demonstrate improvement over the preliminary results and successful clustering of the complex SOM space across all map sizes with less than a second of clustering time in most cases. These tests suggest that HC across all map sizes is closely related to morphology, relative complexity and to a lesser extent, the rotation angle of the source features. The optimal number of clusters appear to be 8 and 16 K clusters. Tests on SOMs trained by latent vectors with autoencoder random rotation training tend to show a more continuous nature evolution of morphologies that are clustered in a manner more depended on image rotation angle. It is evident in these population division tables that the SOM and the HC are segmenting deeper and more complex relationships and morphologies that are not captured by the RGZ labels.

Chapter 4

Discussion

In this thesis, I developed a novel unsupervised clustering and classification system that combines a convolutional autoencoder, Self-Organising Map and Hierarchical Clustering with to investigate and answer the research questions I posed in Section 1.4. Each stage of this developed system is distinct and was evaluated separately across multiple results sections. These results from previous Chapter 3 are discussed in the following sections hierarchically from:

1. The preprocessing methods outlined in Section 2.3 used to clean the radio-astronomical images from the RGZ dataset and allow the system to focus on relevant astronomical features.
2. The convolutional autoencoder of Section 2.4, to compress the RGZ images to a compact 900×1 latent feature vector representation in 0.008425 of a second per image with a training time of 8.81 minutes (un-augmented, full detailed in 3.2.3).
3. The HC SOM of Section 2.5, for accurate complexity separation, clustering and visualisation of the latent relationships and features from within the encoded RGZ data for a 10×10 map after 8.571 seconds of training and an upper-bound of 0.090 seconds of clustering time with 16 K clusters.

Finally, I outline the potential applications of this system to astronomy based on the results and discussion of each system component.

4.1 Preprocessing Discussion

The preprocessing methods I implemented in this ML system were intended to improve the training accuracy of the autoencoder and SOM by focusing on only relevant astronomical features and ease visual interpretation. In this case, I considered filtering successful with the removal of severe noise or side-lobes while retaining astronomical features such as peaks, components and bright diffuse structure. The two methods I trialled were the adaptive thresholding and sigma clipping methods.

I evaluated both preprocessing methods qualitatively by visual inspection. In these tests, I found that the aggressive adaptive thresholding method could clean more background noise than the sigma clipping method. However, I observed that this approach had a negative impact on astronomical features with a tendency to filter out components or create new once by segmenting existing regions. More favourable results were obtained by adopting the traditional sigma clipping techniques used in (Galvin et al., 2018) due to the robust and straightforward nature of segmenting via pixel intensity distribution bins and by taking advantage of the dynamic range found in radio astronomical .FITS images. While the performance of the adaptive masking method would likely be improved if conducted on .FITS images instead of .PNG images, I adopted the simpler, more efficient and commonly applied sigma clipping approach.

This choice is reinforced when considering my aims to keep this system unsupervised and automated with as few manually tunable parameters as possible. Manually tuning preprocessing parameters for new datasets reduces the overall autonomous nature of the system. Manual tuning is reduced in the sigma clipping approach which operates on robust statistically derived intensity thresholds.

Although the final preprocessing solution performed well in radio images, this may not be the case in other wavelengths. Better performing preprocessing solutions may be required in studies with different wavelengths and image features. In situations such as processing the IR wavelength RGZ images, background noise cannot be so easily determined by segmenting the inner 15%, given there are many peaks and diffuse emission outside of this region which would invariably raise the calculated RMS noise, causing the whole image to have a higher overall pixel intensity distribution (histogram). Alternately, I may be able to better measure the RMS by fitting a Gaussian to the distribution of the pixel intensity values, ignoring the outliers. Since the vast majority of pixels are just noise, the real noise is Gaussian, and the peaks and troughs are the outliers.

My results highlight the importance of understanding the characteristics of image data when approaching a problem as I did from a generalist image processing perspective. This is especially the case in this investigation where the best preprocessing solution was a technique already well established in the field. Approaching system development as I did by directly converting all images to .PNG is not a consistently effective solution across all applications, even with image format invariance in mind. This appears to be the case with the radio astronomy where the common .FITS image found in the RGZ DR1 contain a greater range of pixel intensities that may be used for deeper processing than what is found an 8-bit .PNG.

4.2 Autoencoder Discussion

The results of my autoencoder trials addressed my original research question of whether ML dimensionality reduction methods can be used as an information extraction technique and to delineate the complexity and volume of radio-astronomical data in the big-data era (Section 1.4. These results also demonstrated the success of my aim to investigate a convolutional autoencoder in this radio image delineation and feature extraction.

The preliminary results of the autoencoder presented in Section 3.2.1, were a promising first outcome of Ralph et al. (2018) and this thesis. Through these preliminary tests, I demonstrated the convolutional autoencoder as a fast training dimensionality reduction method where the autoencoder produced accurate reconstructions of RGZ images from only a 900×1 array after a considerably short training time of only 2.2 minutes with an average MSE ~ 700.

Although the preliminary autoencoder produced accurate reconstructions, there are some expected minor artefacts. The most noticeable artefact is the blurring found in many of the reconstructed images. This blurring is also often seen as zero value pixel features surrounding image peaks and components. The blurring shape is noticeably square, reflecting the shape and stride of the max-pooling filters and convolutional receptive fields. As discussed in Section 2.4.4.1, I expected this effect due to the coarseness of the filtering (filter size combined with stride) and with the use of zero-padding used in all filtering layers to maintain consistent dimensions. Additional training and optimisation may allow the autoencoder to generalise the dataset image features better to remove this blurring and preserve the exact number of peaks and components.

These results reflected the method only as a proof of concept and to some degree, a lack of refinement. The preliminary results were achieved with training hyper-parameters and autoencoder architecture optimised by simple trial and error. As discussed in Section 2.4.4, this was the overall aim; to prioritise the development time over accuracy as a proof of concept for the method.

As discussed in Section 2.4.4, I augmented these results in the later stages of this investigation using a more refined method that included a better convergence metric and comparison of the autoencoder training and validation error to check model fit quality. Additionally, I show all image reconstruction comparisons with the same representative sample images used throughout this thesis for better comparison between methods (first shown in Section 2.2)

In testing after the submission of Ralph et al. (2018), I optimised all main components of the convolutional autoencoder to improve accuracy and training time. I used a set of initial intuitive conditions and gradually tuned the convolutional and max-pooling receptive field size, the number of convolutions in the encoder and decoder hidden middle layer, and the batch size.

During the first optimisation tests, I found that a filter size of 5×5 with the constant stride of [1x2x2x1] with the initial conditions of Section 2.4.5, was the most ideal to model the radio-astronomical features in the RGZ images. This receptive field size was a trade-off between the prediction accuracy and the time to training and convergence. The 5×5 filter requires ~ 500 seconds more training time than the smallest 2×2 filter, but with a significantly lower error. These differences likely lie in the complexity of the radio-astronomical features in the dataset and the number of trainable parameters required to map them. The test results show the 2×2 filter can reasonably approximate the RGZ images by examining only peak and component features, which requires much less training time by producing highly pixelated reconstructions. The larger 5×5 filter has a greater receptive field size and sufficient trainable parameters to map higher-level relationships that span multiple image features. Consequently, this filter produces less error at the cost of training time. Larger field sizes such as the 7×7 filter operate similarly but are shown to require much greater training time to fully converge on a large set of trainable parameters that incorporates all of the captured features with high fidelity reconstructions.

Minor fluctuations in validation error seen first in this test, across other tests and even between different parameters in the same trial are likely the result of the random sampling during training and validation and do not necessarily indicate instability in the network configuration during the test.

During max-pooling filter size tests, I determined the 2×2 filter was an ideal trade-off between the most accurate configuration with good fidelity in maintaining the number of peaks and components across all samples while still providing some essence of translation and scale invariance. Detailed in Section 2.4.4.2, I aimed to provide scale invariance and boost the inherent translation invariance of the convolutional layers by blurring features with this filter and allow the network to map fewer low scale details and approximate feature positions. This affine invariance cannot be easily seen during autoencoder reconstructions, but in the analysis of the latent vector transformations of the RGZ images in the later SOM training stages. Although these filter size tests showed that a filter of 1×1 has the most accuracy, this filter is merely a perception layer and produces none of the blurring or this accompanying translation invariance. As expected, the blurring effect is proportional to the size of the pooling filter as it is receptive to larger areas of the image. Despite the difference in the filter sizes, I found the training time to convergence between each test had little variation, likely due to the simplicity of the max-pooling operation.

While testing the number of convolutional filters in the central layer of the encoder and decoder, I found most configurations show very similar convergence times and error with greater than 24 filters. The number of filters used in these central hidden layers effectively determine the amount and complexity of the radio-astronomical features mapped by the network. These tests show that at least 24 filters are required to allow the autoencoder to map sufficient radio-astronomical features and high-level relationships to produce accurate reconstructions. The fastest and most accurate configuration was found to be the 32 filters, with the lowest error and reduced convergence time over the original 64 filter configuration. Although the 128 filter configuration does converge after only four epochs, it requires comparatively more time to reach this point and with higher error than the 32 filter test. Despite the higher error, this greater filter count allowed the autoencoder to reconstruct even the central source of the AGN but at a moderate cost of convergence time.

My batch size optimisation tests show that four radio-astronomical images per training batch was ideal regarding training time, accuracy and stability for the current refined architecture. This small batch size provides the autoencoder with a degree of generalisation, where the network can model a wide variety of astronomical features across the dataset. Through my testing, I found that although this size is a significant improvement over the original batch size of 125, it is also prone to instabilities during training. These instabilities are likely the consequence of particularly difficult to map astronomical features in single or recurring batches causing drastic swings in the network trainable parameters and therefore the validation error. This effect is particularly notable in tests in lower batch size tests and to a lesser extent in larger batch sizes. Higher performance with low batch sizes in these tests indicates that additional trials may be required with a lower network learning rate to model the astronomical features. This lower rate would also allow training of larger batch sizes to balance the accuracy and generalisation with increased stability and training time by examining larger sets of astronomical images before weight refinement.

Testing with corruption and random rotations training augmentations produced several interesting results that impacted SOM training deeper in the system. Through these tests, I attempted to investigate the rotational invariance that the autoencoder and SOM may achieve by randomly rotating input images during autoencoder training. As a typical ML problem, rotational variance prevents clustering methods from recognising rotation as a feature distinguished enough to separate it from its class. I sought to determine whether rotational features will be encoded into the latent vector as a result of rotational dataset augmentations. Additionally, I also tested autoencoder denoising to determine whether impulse noise corruption in RGZ images can reduce training time and improve robustness. Corrupted inputs ideally allow the autoencoder to extract more robust and useful features since higher level representations have more stability than the injected random noise (Vincent et al., 2010). In principle, accuracy may be improved by the injection of random noise that introduces enough variance to avoid over-fitting a training set with a slight training time trade-off required to generalise this noise and perform the noise injection.

The results of my testing indicated that the affine and noise injection augmentation training using random rotations and corruption with impulse noise injection in training images provide a slight increase in accuracy and stability with a minor trade-off in training time. Random rotations in the training set provide the greatest stability and the least error after just over 10 minutes for 10,000 images. Findings in these tests suggest that the rotation and corruption augmentation methods are not complementary, where the combination of noise injection and the rotations degrades the accuracy and stability achieved with the rotation augmentations. Performance tests indicate autoencoder reconstructions of RGZ images with the rotation training augmentations are the most accurate with nearly all peaks and components restored. Despite the accuracy increase, the differences between training and validation error are marginally more erratic with the dataset augmentations compared to normal training conditions. Including more training epochs may reveal different or more pronounced effects caused by the rotations by analysing the effect of training on much more overall orientation. Large selections of rotation angle can be seen in systems such as Parallelized rotation and flipping Invariant Kohonen maps (PINK) from (Polsterer et al., 2015), where the SOM is trained on images from the FIRST survey rotated every 2°.

Improvements can be made to these rotation augmentations by injecting image characteristic noise into all NaN pixels in the corners of input images, as described in the sigma clipping preprocessing of Section 2.3. Additionally, sampling a larger window of the original survey image would provide this information. NaN corner regions were not a significant concern, however, given background noise is ideally zero after preprocessing.

Across all tests, there are no significant indications of over-fitting or under-fitting past the first epoch. Seen by the difference between the training and validation error, no significant deviations are observed, except where low training batch sizes result in training instability. These observations indicate that this particular autoencoder configuration and set of training hyper-parameters results in a system that appears robust in nature by its tendency to perform similarly in training and validation. This may indicate that this network can function accurately for different radio-astronomical data with different PSF and instrument properties, or even astronomical images in other wavelengths. The robust nature of this system may be related to how many output parameters the network can regress to in the latent vector. Poor fitting in a supervised CNN, for example, can be seen more clearly given the network is trained to regress neuron outputs to an only a few neurons in the output. While the autoencoder I implemented in this system is regressing images down to 900 neurons and back to the original dimensions. The larger mappable space in the autoencoder may be the cause of such robust modelling.

Throughout this optimisation process, I may have lost a degree of data generalisation. Attempting to process new data using this configuration may not produce the same ideal results if through the optimisation process I configured the autoencoder architecture mostly to model RGZ data specifically. Additionally, changes in input image size will alter the latent vector size and affect network performance and the performance of any subsequent analysis. More testing is required to determine an optimal latent vector size and how the input image may be shaped to produce a more dynamic network for application to other radio-astronomical image data. Additionally, my network may be limited by a potentially oversimplified and imprecise convergence metric. Given I was searching for a compromise between accuracy and training time, I adopted a convergence measure that examines a reasonably short window, since running for a large number of epochs does not align with my aim of a fast training system. Ideally, I would also take into account the convergence of the weights and bias' rather than convergence primarily concerned with accuracy, although it is easier to visually communicate the results of this approach in the reconstruction image output.

Overall, there is significant performance and training time improvement in each optimisation step, with the optimised autoencoder producing accurate reconstructions of radio-astronomical image features with a MSE approximately 25 times lower than the preliminary network error. Improved accuracy in the final network is at the cost of additional training time, however, where the original configuration was trained in 2.2 minutes, and the optimised configuration was trained in 8.8 without random rotation augmentation training and 10.6 minutes with the augmentations. No significant indications of under-fitting or over-fitting were observed in training after the first epoch throughout all trials.

The final un-augmented autoencoder network can encode an image on average, in 0.0084 of a second. This encoding time indicates that on average, the time the average time to encode 30,000 RGZ images for SOM training or validation is 252.75 seconds or 4.21 minutes. The random rotation augmentation results in a total encoding time of 440.25 seconds or 7.34 minutes at an average of 0.0147 of a second per image. The successful implementation of a convolutional autoencoder directly addresses the growing big-data challenges of radio-astronomy, by substantially reducing large volumes of complex radio-astronomical images to a compressed latent feature vector representation with minor computational overheads.

4.3 Self-Organising Map Discussion

The SOM results in Ralph et al. (2018) demonstrate successful implementation of an autoencoder latent vector trained SOM. These results indicate that my novel approach is an efficient unsupervised ML alternative to solving the problems of big-data scale and complexity in modern radio-astronomical data.

Through these results, I was able to determine whether such a SOM can perform practical data exploration by extracting and visualising the dynamic distribution and high-level topological relationships of radio-astronomical images from the RGZ dataset. I show that SOM neurons trained on latent feature vectors can be visualised by reconstruction using the autoencoder that produced the original training latent vectors. I demonstrate that within a single minute, my approach can train a SOM to perform unsupervised complexity separation and HC of RGZ images on a SOM umat for in-depth data exploration. Using this system, I also successfully investigate the nature of rotational invariance in SOM training and the effects of training using latent feature vectors produced from an autoencoder trained with random rotation augmentation.

In the preliminary results of Ralph et al. (2018), the trained SOM produced a umats that clearly shows the morphology distribution of RGZ images. On this map, the Euclidean distance between the learned weight of each neuron and their neighbouring neurons shows images clustered by relative complexity. In high distance regions, anomalous matching images have a latent feature vector with a high to surrounding neighbours which highlights outliers within the RGZ image set. The morphological clusters in this map are continuous and not highly discrete. These clustered regions are sub-clustered by orientation, with similarly oriented objects clustered together with gradual transitions between classes. I expected to see this gradual transition between classes of images given these objects do not have entirely discrete classifications. The central low distance region of the umat contains matching images as single compact sources, which gradually progress in complexity to compact multi-point sources. In these preliminary results, I also demonstrated HC, where I used three the K-means algorithm with three K clusters, 0, 1 and 2, which segment the SOM into three groups of simple, compact multi-feature and highly complex images.

In the final tests after the submission of Ralph et al. (2018), I optimised the final SOM training hyper-parameters and network configuration similar to the autoencoder. In this approach, I used a set of initial conditions used in literature to iteratively determine the ideal map size, learning rate decay function, learning radius and decay function. I evaluated these SOMs by their performance in anomaly detection and tested the effect of training the SOM on RGZ images latent vectors that had been produced by an autoencoder trained with randomly rotated RGZ images. Finally, using the most accurate combination of these, I tested HC of the SOM with the K-means algorithm.

Throughout map size testing, I trialled four toroidal square SOM sizes; 5×5, 10×10, 20×20 and 40×40 for 12 epochs using 30,000 RGZ images for training and 30,000 RGZ images for validation. Across all map sizes, the SOM successfully visualises a wide array of latent radio-astronomical relationships and features of the RGZ data, distributed by their mutual similarity. In these continuous clusters, radio sources gradually develop an impressive array of features and structures that evolve over the surface from simple point sources to highly complex multi-peak, multi-components sources. As in the preliminary tests, high distance regions on the umat highlight the highly complex structure and outliers within the RGZ image set. These results show a gradual transition between image classes given radio-astronomical objects do not have discrete features, but instead continuous unresolved morphology. These learned gradual transitions are ideal as they reflect the true continuous and indiscrete characteristics of radio astronomical image features.

In the final system, I implemented neuron weight decoding using the decoder side of A trained autoencoder. A SOM trained on images alone will produce neurons with weights learned and displayed as images, as seen in cases such as (Polsterer et al., 2015). My neuron weight decoding approach bridged a serious gap found when training a SOM on latent vector representations, where latent vectors are not themselves images and cannot be visually interpreted. This weight decoding method functioned successfully to solve this problem and illustrated the true learned features of each neuron as a reconstructed image representation. This approach satisfied my research aims of displaying the SOM learned relationships contained in the latent vector image representations (Section 1.5).

Decoded neuron images demonstrated an interesting phenomenon not observed in the preliminary results. In these decoded neuron weight maps, the learned latent vectors of the RGZ dataset display a gradually expanding morphology from a circularly distributed set of neuron features, where the centre is a neuron that appears as a rotated average of a radio image morphologies. These ‘rotated’ neuron weights are examples of rotationally invariant neurons, where matching image morphologies would be consistently placed here regardless of the source rotation angle. The rotationally invariant matches are seen in all of the umat matching source overlay images, where the matching images appear somewhat randomly rotated but placed appropriately by morphology. Naturally, no source in the dataset exhibit this strange halo morphology, but their presence suggests that SOM training on autoencoder latent vectors may have some degree of inherent rotational invariance given a small subset of neurons have a matching criterion that ignores source rotation angle.

Overall, results across different map sizes greater than the 5×5 SOM differ only very slightly, indicating the SOM complexity separation performance is reasonably robust to most map size changes. The most accurate map using the initial conditions is the 20×20 map, However, the 10×10 map here with a slightly lower accuracy but the highest F1 score as a more certain performance metric of the classifier. The best performing SD anomaly detection point on the Euclidean distance distribution map is inversely proportional to the map size, where the most accurate SD, draws closer to the point of zero variance given the map size increases the number of distance bins. These trends highlight ideal complexity separation, where simple point sources reside in a broad peak near the lowest distance bins and increasingly complex sources distributed over higher distance regions. Improved results in larger maps are shown across all tests likely due to a larger neuron sample size but at the cost of reduced human-readability and negligible processing time.

Training times for these maps are very low, with less than a minute even for the largest 40×40 SOM size. As expected, the training time for all tests increases proportionally to the map size, given a larger network. Based on these tests I chose the 10×10 SOM for all subsequent trials, as it produces the best F1 score and shows a wide range of RGZ morphologies in a relatively small map that is easy to interpret visually.

Although the system performs well even in this stage, a typical SOM umat with much larger grid sizes will usually show high Euclidean distance regions separating neurons into distinct clusters. In my system, SOM sizes of 10×10-40x40 are not capable of displaying these relationships given their relatively small map size. I sacrifice this functionality much like (Polsterer et al., 2015), for a visually interpretable map that requires little processing time. Distinct umat regions may be possible with larger maps or an emergent SOM, where the grid space contains more neurons than training samples. Despite this slight lack of refinement, optimisation tests on larger maps are unnecessary given I scale the parameters with the map size and I am training on a toroidal SOM space, meaning few cases will see neurons not adequately influenced by different learning rates, neighbourhood functions and decay rates during training. If I was to train for significantly larger maps, then this may become more necessary. As a result, I use only used the ideal map size in the first test for all subsequent tests, except where larger map sizes were required to show more detailed morphologies or to deeper investigate the rotation invariance tests.

Subsequent learning rate tests indicated that the exponential alternative to linear learning rate decay has a minimal effect on SOM training. This finding may be specific to only the latent vector data, as other astronomy-related SOM implementations such as (Geach, 2012) using spectral data, achieved optimal performance using this exponential learning rate decay function. These results suggest that SOM learning is not highly dependent on weight learning scales or decay. This is further reinforced by how robust SOM learning is in experiments with changes made to the map size and the constant learning rate bounds I employ.

Tests conducted neuron learning neighbourhood, and radius decay functions indicated the Gaussian neighbourhood with a linear radius decay function is the most accurate concerning complexity separation and with the lowest training time when trained on radio-astronomical latent features vectors. This Gaussian neighbourhood appears to learn fewer 'rotations' with higher resolution morphologies compared to the linear neighbourhood. In addition to the differences in morphology, this radius type produces a broader range of Euclidean distances with a more defined neuron distance distribution. Output maps show that point sources are mostly residing in a broad low distance region, and more complex morphologies in more defined higher Euclidean distance regions. This broader distribution appears to have had a slight impact on the observed complexity separation performance as a result of the Gaussian neighbourhood function providing a better mapping of the SOM space distance between different radio-morphologies in the latent vector training set.

Tests show the linear neighbourhood results in under-fitted neurons, with all sources as single component sources and halo-like extended structure toward the centre. Correspondingly, the distribution of the umat vary significantly across RGZ labels with matching sources showing with no consistent relationships. Similar results can be seen with the exponential learning rate decay functions, except where the umat shows only modelling around the edge of the learning radius in the initial epoch. Given I use a toroidal map, the first epoch influences neurons with an initial radius of 5 neurons, leading to the higher Euclidean distance regions forming as quarter circles around the corners, which is a circular region centred in the middle of the map with this initial diameter. This poor mapping results in a scattered Euclidean distance distribution. It is clear by these results and poor RGZ image matching on the umat that regardless of the neighbour type the exponential radius decay does not allow the map neurons to completely model the radio source morphologies in the RGZ training and validation sets.

Using the tested ideal training hyper-parameters and system architecture, I tested the effects of training a SOM on the latent vectors of radio astronomical images using an autoencoder trained using random rotation dataset augmentations. These tests show that a Gaussian neighbourhood type SOM trained with latent vectors from an autoencoder trained with random rotations produces the highest accuracy and

F1 score. Despite training only on four different random rotation angles (the rotation augmentation trained autoencoder converged at 4 epochs as outlined in Section 3.2.2.5). The SOM produces maps with weights that appear to have learned sufficient rotational features to remove most of the learned 'rotated average' neuron weights, where only some high distance neurons may show an extra rotation in the neuron weight. Training in this manner results in more discrete neuron weights with a greater emphasis on source rotation angle.

I interpret this reduction in multiple rotation angles as the SOM and autoencoder learning the rotation angle as a dominant feature along with the spatial morphology. By removing the rotated average weight neurons, these maps are fundamentally less rotationally invariant given neurons weights, and matching radio images are arranged and defined more by their rotation angle than in previous tests. Despite becoming more sensitive to the rotation angle, complexity separation performance does not appear to be significantly affected. Additionally, these maps are more natural to visually interpret, with matching RGZ images on the umats exhibiting a smoother distribution of source rotation angles across the SOM surface and with decoded neuron weights resembling more specific radio-astronomical features with less rotated average morphologies.

In the final stages of this thesis, I tested the HC of 10×10, 20×20 and 40×40 square toroidal SOM maps using a linear learning rate and radius decay rate function with a Gaussian neighbourhood function as the most accurate training configurations from earlier sections. I tested 4, 8 and 16 'K' clusters using the Scikit-Learn K-means HC algorithm and maps trained with latent vectors produced from the autoencoder trained with and without random rotation augmentations. The results of the HC were similar to the preliminary results and showed across all map sizes that clustering is closely related to morphology, complexity and a lesser extent, the rotation angle of the source features depending on the autoencoder training.

Concerning general clustering quality, all tests show reasonable connectedness with few neuron clusters inter-mixing. Mixing appears to be only a minor problem present in larger maps. In higher K numbers of 8 and 16, clusters become increasingly purer. Purity here concerns how homogeneous clustered morphologies appear to be, with minimal mixing and leakage. Compared to preliminary results, there is a definite improvement with higher cluster numbers and more distinct separation between different morphological groups. This improvement is shown with the way I display HC, the addition of neuron weight decoding images and in increased completeness across all clustering tests, where many clusters contain large portions of closely related labelled RGZ images. Minimal processing time is seen in the time required to process the K-means HC across all maps. Clustering time here is less than a second in most cases, which increases proportionally with both the allocated number of clusters and the map size and no conclusive differences in the clustering time required for the latent vectors with or without rotation augmentation training.

Similar to previous map size tests, the most meaningful morphological separations appear to be found with the highest map sizes of 20×20 and 40×40 maps and with 8 and 16 K cluster values. Results show the algorithm effectively separates many interesting morphologies where relatively simple clusters are comprised of mostly point sources (RGZ label 11), compact multi-peak single component sources (RGZ label 12), complex sources with highly separated sources or sparse sources with distant companions and sources with higher than average noise or PSF side-lobes features. In these maps, there are also clusters separating the most complex sources, such as bright and diffuse extended structure with globular and bent tail morphology, sources with extended emission with large jets and AGN-like morphology. These classes appear more evident with the largest 40×40 maps where some clusters in the 16 K HC tests contain highly anomalous sources such as the 44 RGZ labelled sources. The 10×10 map size does not perform as well here with the 8 and 16 K cluster, where some clear morphological groups are broken erroneously into separate K clusters. As in the preliminary results, there are definitive simple, complex and intermediate groups divided by the HC.

A balance must be reached with the number of clusters and neurons available in the map. Using too many clusters may cause groupings to split logical classes to satisfy the K value, as seen in the 10×10 tests. Using too few clusters, however, may result in true divisions and relationships on the map being under-represented or not revealed. The main flaw in clustering continuous data such as radio-astronomical features is that SOMs will produce a corresponding continuous output, with morphology clusters gradually evolving over the map. By attempting to find discrete divisions in a continuous manifold is difficult and primarily resides in the realm of regression. This difficulty does highlight success in the SOM given it accurately represents the characteristics of the data as in-discrete, where complex extended radio features are continuous.

Across all tests, there are several evident entropy and map population effects. Most notably, mean cluster entropy values increase with the presence of more complex and anomalous sources. These effects are also observed in the preliminary results, where entropy describes the neuron and cluster label distributions, where a low entropy indicates a homogeneous neuron or cluster matching label set (Section 2.5.6.1). As expected, point sources have the lowest mean entropy due to their simplicity and the greatest neuron population across the map due to the point source bias in the RGZ dataset. Higher complexity sources, represent a smaller part of the training set and appear to have consistently smaller cluster populations, but significantly greater mean entropy due to their complexity. These population and entropy statistics effectively reveal not only the types and complexity of morphologies in the dataset but also effectively describe the original label distributions and biases.

In HC testing of maps trained with random rotation trained autoencoder latent vectors, there are mostly similar cluster assignments. However, in the HC of the 10×10 map trained with random rotation augmentation latent vectors, these clusters appear not to separate source morphology as well as the regular autoencoder training. Similar results are seen across the 20×20 and 40×40 map sizes, where label divisions are not as complete as in the previous tests. In these trials, the issue seems to originate from rotation changes being taken more into account, where many clusters are separated by the rotation angle of the components and peaks within the image. As discussed, effects such as these are an expected downfall of a discrete clustering algorithm such as K-means but is especially difficult in the rotated maps which tend to show a more continuous nature that is also clustered in a manner more depended on image rotation angle. Although rotation angle is not a meaningful property in radio astronomy, this clustering with an increased focus on the rotation angle appears to create a more naturally interpretable SOM. Regardless, clustered maps trained with and without latent vectors trained with autoencoder rotation augmentations successfully show similar relationships where there are many distinct morphological clusters.

The increased rotational dependency observed in these tests raise many questions regarding the true nature of rotational invariance in a SOM. For a neuron to be rotational invariant, morphological features must be the only feature that is clustered by the system. For this to be the case, genuine rotational invariance would result in all neurons on the SOM being mapped with the same position angle, or for each neuron to contain all possible position angles, as the observed rotated average morphology seen in decoded neuron images. These concepts suggest that greater rotational invariance can be found on maps with more of the observed averaged rotation neurons. Moreover, this invariance is diminished when the SOM is trained on latent vectors produced with random rotation augmentation trained latent vectors. To combat these effects, true rotational invariance may be achieved by aligning all images to a common major axis, and greater scale invariance may also be achieved by cropping and enlarging all central components to the same size.

These tests demonstrate the SOM behaviour when applied to a relatively practical radio-astronomical case, where the dataset is unbalanced, with training data containing mostly point sources (RGZ label 11). I tested using the full dataset without any consensus filtering to retain as many 'difficult' to classify images as possible. Consequently, validation in the final stages was likely affected by low consensus scores which potentially indicate images with false or disputed labels. My results still show that the SOM manifold contains regions representing even the most unique and low population source morphologies, despite training with a dataset containing a substantial point source bias. This robust dataset representation further reinforces the practicality of my method as a tool for future surveys.

The difficulty I encountered when trying to use clusters as classifications on a continuous manifold can be seen by both the label cross-overs found between many HC classes and the point made by (Kohonen, 1997); SOMs are not explicitly designed for hard classification. The original principles of SOM learning will not produce highly distinct clusters, but will instead produce what I see in my results; a semantic map of outliers, regions, and morphologies rather than highly distinct groups. These results are not unlike those of PINK in (Galvin et al., 2018) where SOM outputs show a continuous indiscrete range of morphologies. As previously mentioned, it is possible that in exceptionally large SOM these relationships may have enough space to become sufficiently separated for discrete classification.

It is evident in the population division tables of the SOM HC tests that the K-means algorithm is segmenting morphologies and associating relationships not entirely based on the RGZ labels. These segmentations are evident due to the relatively low populations of RGZ labels in many clusters across all tests, despite the clustered maps showing reasonably clear and logically assigned clusters based on morphology and complexity. These results are far from negative, however, as this indicates that the relationships learned by the system are deeper and more true to the genuine nature of the radio-astronomical images rather than the simple peak and component counts of the RGZ label set. Moreover, detection of different labels than the training set is also wholly expected given the system is trained unsupervised.

Unsupervised learning is a highly challenging task where the relationships of an image are automatically extracted without a label set. Whether these features are useful or represent meaning in the image is the essential performance and success of the method. Establishing performance measures in these unsupervised systems is challenging given the relationships, and ‘labels’ learned by unsupervised methods will usually only correlate with those of the training set. As a result, one of the simplest ways to evaluate the accuracy is by visually examining how well neuron clusters resemble the label set or produce distinct and useful morphological groups. Standard clustering performance measures are also difficult to apply here, given the HC appears to separate neurons not only based on the umat Euclidean distance, but also on the radio-astronomical features. Ideally, I would use a statistical means of evaluating the separation of each cluster by quantifying how ‘distant’ cluster members are, or by a purity measure of the features. However, these statistical measures are also not easily applied, given the HC is based on both the umat distance and morphology modelled from the latent vectors. The nature of this clustering means the definition of ‘ideal separations’ across these two spaces is statistically challenging to locate, especially when considering the relationships my method detected are not directly based on the RGZ labels.

As previously mentioned, a significantly larger and more refined umat would likely provide enough information across a higher number of neurons to produce a meaningful statistical measure for assessing the performance of the SOM and HC. Difficulty in producing discrete classification and definitive performance measures is not unique to my system, but a challenge seen in other astronomical image trained SOMs in literature. These obstacles can be seen with the previous example of (Polsterer et al., 2015) and even (Naim et al., 1997), which is in fact titled 'Galaxy morphology without classification...'.

The results of my system highlight the need for more complex labels that better describe the features and morphology of the dataset images. This need is evident by my system extracting morphologies significantly more complex than simple peak and component counts. More useful labels than peak and component counts are required to describe image morphologies that are actually interesting to researchers with such as diffuse, double-lobed, bend-tail and AGN-like. The label set shortcomings can also be seen especially in cases where the RGZ labels do not fully capture the meaning of radio-astronomical images. For example, a RGZ label 33 could describe both a standard radio triple as three simple point sources or an AGN, which are vastly different objects. Additionally, sources with so-called bent-tail morphologies are highly interesting as galaxy cluster tracers but can be given the same label as basic radio double with a 22 annotation. By only learning on human classified labels, these classification methods will only be as good or worse than human classifications. These points suggest that my unsupervised approach may have use in comparing images and associated labels across labelled datasets to determine how well labels truly describe the nature of their associated images. Additionally, the extracted images features and relationships found using my approach may bear insight into more intuitive radio astronomical image categories without the influence of human bias.

The results of this thesis indicate that my system is a robust and practical solution to the current issues of scale and complexity in the big-data era of Radio Astronomy. These results also confirm my research aims and questions (Sections 1.4 and 1.5). Using my approach, I successfully demonstrated delineation of the scale and complexity of radio-astronomical data by mapping image feature morphologies from the RGZ dataset onto a learning manifold. My results show that this SOM space can efficiently and successfully map these radio-astronomical morphologies to a relatively small space comprised of square maps between 10×10 and 40×40 neurons. This efficiency is the result of successfully training the SOM on image data reduced by 1600%, where the SOM processes only 900 elements per image in the autoencoder latent vectors as opposed to the full 14,400 elements of a preprocessed RGZ image. These results suggest that my system may be used as a powerful preprocessor to improve analysis efficiency of a supervised classification or regression method by training on the delineated spaces provided by my method.

The efficiency, effectiveness, and scalability of this unsupervised method presents itself as a useful addition to many survey and observational pipelines while requiring only basic and easily accessible CPU requirements. The speed of this system also challenges the notion of brute forcing big-data problems with increased computational power and GPUs power, rather than searching for a more efficient solution. Through these capabilities, my approach demonstrates itself as a potentially vital instrument in the modern astronomer's tool-kit. The final un-augmented autoencoder network can on average, encode an image, in 0.008425 of a second with the results of the batch size tests, where a training time of 8.81 minutes for 10,000 RGZ images is required. This encoding time indicates that on average, the time to encode 30,000 RGZ images for SOM training or validation is 4.21 minutes. While the random rotation augmentation results in a total encoding time of 7.34 minutes at an average of 0.015 seconds per image. With the most accurate SOM (Table3.31) requiring a training time of only 1.53 minutes with the highest clustering time of 1.136 seconds for 16 classes, the upper-bound on the overall processing time of my system is only 14.55 minutes.

The total training time of this system is competitive with other methods from literature such as (Polsterer et al., 2016). My system produces similar SOM morphologies with square neurons and significantly reduced processing time as python code using a 24 core CPU (Detailed in Section 2.1) requiring 14.55 minutes for 30,000 images, compared to 17 days with 200,000 images using python with an 8 core CPU. The difference in data volume is the likely cause, where even with random rotation autoencoder training, my training latent vectors are only 900 elements per image, opposed to a total of 1,331,280 elements per image including rotations used in (Polsterer et al., 2016). These results also confirm my research aims and questions (Sections 1.4 and 1.5) of developing a system that can solve the current challenges of scale and volume faced by researchers in the big-data era of Astronomy.

4.4 Applications To The Field

Given the results of this thesis, the system I developed has the potential to be implemented in a wide range of big-data radio-astronomical applications where the complexity and scale of collected data render analysis too time-consuming or complicated. Survey science is a particularly relevant used case, where my system can be run on minimal computational resources after a source finder to separate complexity, locate objects of interest and extract a visual summary of a large set of observations that could not be gained through manual analysis.

My approach bears potential for use in sizeable future radio surveys such as EMU which is expected to contain over 70 million radio sources (Norris et al., 2011). The ASKAP processing pipeline (calibration, imaging, source extraction) is expected to produce a data volume in the order of 22 TB per day. The scale and complexity of this data are made all the more difficult when considering the tasks planned for this data, namely, removing artefacts, extracting complex and diffuse sources, merging components into sources, source classification and cross-identification across multi-wavelength observations. An observing campaign the size of EMU requires a system such as this new autoencoder and SOM hybrid which can perform fast automatic complexity separation and data exploration that address many of these problems. Furthermore, anomalous sources and objects of interest that may be located on the SOM surface such as bent-tail galaxy radio morphologies are of particular importance and tracers of galaxy clusters. The presence of SOM regions with neurons clusters showing higher than normal traces of the instrument PSF compared to other clusters also suggests that this method may be used to locate noisy images and possibly artefacts. Additionally, by developing multi-wavelength capabilities into my system, I may be able to address the task of multi-wavelength cross-matching and objectives such as radio-IR host identification.

Potential applications to novelty analysis and complexity separation tie into both SETI and Widefield Outlier Finder (WTF) studies. In these applications, discovering the uncharacteristic in an observation, whether it be an object or process, may indicate inconstancies in astronomical understanding and possibly mark a discovery. Current efforts rely on researchers to make these discoveries while exploring a hypothesis, although most of the significant discoveries in radio-astronomy have been serendipitous (Ekers, 2009), to be unplanned and found instead while pursuing an unrelated hypothesis (Norris, 2017a). Given the unsupervised nature of my system, it has particular use in these applications, where pre-labelled training sets are not available. Instead, these applications require a system such as mine to detect complex and anomalous behaviour or outlier classes not captured in labelled datasets but instead found on their own as an unsupervised ML technique, while making few assumptions about input data and reducing the human bias inherent in training sets (Torralba and Efros, 2011).

The speed of this method holds implications for use on large future surveys, large-scale instruments such as the Square Kilometre Array and in other big data and anomaly detection applications. Additionally, by combining clustering with citizen science projects such as RGZ, greater efficiencies can be achieved with volunteers inspecting only a small sample of objects from each cluster on the SOM surface or being guided by possible morphologies in each cluster. The successful implementation of this system additionally raises questions of the possible application of this approach to other data such as the compression of images obtained in other wavelengths or even spectra and other time-series data. The efficiency obtained by reducing these data volumes could significantly reduce processing time and computation requirements while boosting research output with faster analysis times.

Used cases for this system may also extend to studying pre-labelled data to analyse the existing relationships with labels as a general guide, given my system is unsupervised and can use labels for verification. Moreover, this technique can be used in this capacity as an additional analysis step and an unbiased checksumming method. This extra verification step could be used to display dataset morphologies with their associated labels to see any shortcomings in the way the labels represent the data set, much in the way shown with the results of this thesis.

Chapter 5

Conclusion and Future Work

5.1 Future Work

I plan to improve each of the major and distinct components of this system from the autoencoder, SOM and HC algorithms. Starting with the autoencoder, I aim to conduct additional tests in varying the latent vector sizes and network structure to improve optimisation by locating a better balance of training time to accuracy. I aim to implement multi-wavelength analysis across radio, IR by developing a stacked autoencoder architecture (Vincent et al., 2010) to train latent vectors to encode multi-channel data. The autoencoder also provides many interesting data compression and information extraction methods that I will apply to time-series and spectral data.

As previously discussed, the SOM is continuous. Consequently, the challenge of more definitive and in-depth source classification can be viewed as a regression problem. I aim to continue working in this direction to create a ML regression framework auxiliary to the SOM to regress the continuous SOM morphologies into discrete classes. A system such as a RTF would be particularly useful in this case where it may be trained to map existing RGZ labels to the SOM activation map location probabilities of a target image.

Given the ability of the HC algorithm to segment the SOM surface, I seek to develop a more expansive SOM manifold using a topological SOM, where I will train additional SOMs on the matching neurons contained in each cluster. In this method, individual SOMs may be explicitly trained on the pre-clustered latent vectors to analyse deeper relationships between similar morphologies. Additional variables such as map size can be eliminated and more dynamic relationships examined using a growing SOM, as proposed by (Rauber et al., 2002). This SOM variation dynamically scales the SOM grid based solely on the relationships detected within the dataset. I find this approach very interesting where a significantly larger and more refined umat may contain enough information to bridge the problem faced by current efforts in SOM astronomical morphological analysis, where it is difficult to produce a meaningful statistical measure for assessing the performance of the SOM and HC with current map sizes and shapes. I will also investigate other HC algorithms on the SOM surface and in different learning manifolds to explore other avenues of complexity separation and clustering.

Based on the results of the affine invariance testing, I may be able to achieve true rotation invariance by aligning all images by their features to a common major axis. Additionally, I aim to achieve better scale invariance by cropping and equally enlarging all central components to the bounds of the image. This scaling is a complicated operation as separating related, and unrelated features are difficult especially if no IR morphological information is used for host cross-matching. If these affine transformation methods are successful, however, they would likely provide a significant increase in the overall system performance and significantly contribute to current efforts in affine invariant SOM analysis. Finally, further work must also be done to create a more definitive statistical means of evaluating the SOM and HC clustering performance to more robustly demonstrate the overall performance of the system.

5.2 Conclusion

In this thesis, I developed a novel ML method that takes a new approach to unsupervised clustering and data exploration. Through my investigation, I demonstrated the coupling of SOMs with convolutional autoencoders as a powerful means of clustering and automatic unsupervised data exploration of large radio-astronomical datasets. Thesis directly addresses the challenge of rapidly increasing data scale and complexity in modern big-data radio-astronomy.

In this investigation, I implemented a convolutional autoencoder which performed successful delineation of high volume, complex radio-astronomical data. My system provided rapid feature extraction with dimensionality reduction of these images to a compressed latent feature vector representation. Based on this success, I demonstrated SOM feature extraction and visualisation of the high-level topological relationships found in radio-astronomical latent vector representations of the RGZ dataset.

The results of my investigation demonstrated the capabilities of my method as an accurate and fast analysis method. Within this short processing time, my approach analyses radio-astronomical images by displaying the high-level morphology relationships of RGZ dataset images using Euclidean distance on a SOM umat and in HC with distinct classes of varying complexity. This operation is shown to accurately perform the vital task of complexity separation, whereby astronomers can focus on potential discoveries and the unexplained currently hidden in the modern data deluge. Using this SOM and HC technique, I successfully investigated the nature of rotational invariance in SOM training and the effects of SOM training using latent feature vectors produced from an autoencoder trained with random rotation augmentation.

The speed of this method indicates excellent scalability and holds implications for use on large future surveys, large-scale instruments such as the Square Kilometre Array and in other big-data and complexity analysis applications. Given major astronomical discoveries are unplanned and found in the unexpected, my unsupervised approach is highly desirable, by operating without assumptions and only requiring labelled training data for verification. Through this thesis and Ralph et al. (2018), I contributed to the field by addressing a wide range of modern challenges and limitations using my new approach. Using this powerful new tool, I show the potential of research that can be found in cross-over studies between engineering and astronomy with the generalisable nature of machine learning and image processing in data analysis.

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. (2016). Tensorflow: a system for large-scale machine learning. In *OSDI*, volume 16, pages 265–283.

Abbott, B., Abbott, R., Adhikari, R., Ananyeva, A., Anderson, S., Appert, S., Arai, K., Araya, M., Barayoga, J., Barish, B., et al. (2017). Multi-messenger observations of a binary neutron star merger. *Astrophysical Journal Letters*, 848(2):L12.

Al-Jarrah, O. Y., Yoo, P. D., Muhaidat, S., Karagiannidis, G. K., and Taha, K. (2015). Efficient machine learning for big data: A review. *Big Data Research*, 2(3):87–93.

Alger, M., Banfield, J., Ong, C., Rudnick, L., Wong, O., Wolf, C., Andernach, H., Norris, R., and Shabala, S. (2018). Radio Galaxy Zoo: Machine learning for radio source host galaxy cross-identification. *Monthly Notices of the Royal Astronomical Society*.

Anders, F., Chiappini, C., Santiago, B. X., Matijeviffh, G., Queiroz, A. B., and Steinmetz, M. (2018). Dissecting stellar chemical abundance space with t-SNE. *ArXiv e-prints*, page arXiv:1803.09341.

Andreu, G., Crespo, A., and Valiente, J. (1997). Selecting the toroidal self-organizing feature maps (TSOFM) best organized to object recognition. In *Neural Networks, 1997., International Conference on*, volume 2, pages 1341–1346. IEEE.

Bailey, S., Aragon, C., Romano, R., Thomas, R. C., Weaver, B. A., and Wong, D. (2007). How to Find More Supernovae with Less Work: Object Classification Techniques for Difference Imaging. ApJ, 665:1246–1253.

Baldi, P. (2012). Autoencoders, unsupervised learning, and deep architectures. In *Proceedings of ICML workshop on unsupervised and transfer learning*, pages 37–49.

Ball, N. M. and Brunner, R. J. (2010). Data mining and machine learning in astronomy. *International Journal of Modern Physics D*, 19(07):1049–1106.

Ball, N. M., Brunner, R. J., Myers, A. D., Strand, N. E., Alberts, S. L., Tcheng, D., and Llor, X. (2007). Robust machine learning applied to astronomical data sets. II. Quantifying photometric redshifts for quasars using instance-based learning. *The Astrophysical Journal*, 663(2):774.

Ball, N. M., Brunner, R. J., Myers, A. D., and Tcheng, D. (2006). Robust Machine Learning Applied to Astronomical Data Sets. I. Star- Galaxy Classification of the Sloan Digital Sky Survey DR3 Using Decision Trees. ApJ, 650:497–509.

Ball, N. M., Loveday, J., and Brunner, R. J. (2008). Galaxy colour, morphology and environment in the Sloan Digital Sky Survey. MNRAS, 383:907–922.

Ballard, D. H. (1987). Modular Learning in Neural Networks. In *AAAI*, pages 279–284.

Banfield, J. K., Wong, O., Willett, K. W., Norris, R. P., Rudnick, L., Shabala, S. S., Simmons, B. D., Snyder, C., Garon, A., Seymour, N., et al. (2015). Radio Galaxy Zoo: host galaxies and radio morphologies derived from visual inspection. *Monthly Notices of the Royal Astronomical Society*, 453(3):2326–2340.

Baron, D. and Poznanski, D. (2017). The weirdest SDSS galaxies: results from an outlier detection algorithm. MNRAS, 465:4530–4555.

Becker, R. H., White, R. L., and Helfand, D. J. (1995). The FIRST Survey: Faint Images of the Radio Sky at Twenty Centimeters. ApJ, 450:559.

Bentez, N. (2000). Bayesian Photometric Redshift Estimation. *The Astrophysical Journal*, 536(2):571.

Bezdek, J. C. (1981). Objective function clustering. In *Pattern recognition with fuzzy objective function algorithms*, pages 43–93. Springer.

Boyd, S., Parikh, N., Chu, E., Peleato, B., and Eckstein, J. (2011). Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122.

Bradski, G. (2000). The OpenCV Library. *Dr. Dobb's Journal of Software Tools.*

Breiman, L. (1996). Bagging predictors. *Machine learning*, 24(2):123–140.

Breiman, L. (2001). Random forests. *Machine learning*, 45(1):5–32.

Bundy, K., Ellis, R. S., Conselice, C. J., Taylor, J. E., Cooper, M. C., Willmer, C. N. A., Weiner, B. J., Coil, A. L., Noeske, K. G., and Eisenhardt, P. R. M. (2006). The Mass Assembly History of Field Galaxies: Detection of an Evolving Mass Limit for Star-Forming Galaxies. ApJ, 651:120–141.

Carliles, S., Budavri, T., Heinis, S., Priebe, C., and Szalay, A. S. (2010). Random forests for photometric redshifts. *The Astrophysical Journal*, 712(1):511.

Carrasco Kind, M. and Brunner, R. J. (2013). TPZ: photometric redshift PDFs and ancillary information by using prediction trees and random forests. MNRAS, 432:1483–1501.

Carrasco Kind, M. and Brunner, R. J. (2014). SOMz: photometric redshift PDFs with self-organizing maps and random atlas. *Monthly Notices of the Royal Astronomical Society*, 438(4):3409–3421.

Chandola, V., Banerjee, A., and Kumar, V. (2009). Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):15.

Chattopadhyay, T., Misra, R., Chattopadhyay, A. K., and Naskar, M. (2007). Statistical Evidence for Three Classes of Gamma-Ray Bursts. ApJ, 667:1017–1023.

Chaudhary, V., Bhatia, R., and Ahlawat, A. K. (2015). A Constant Learning Rate Self-Organizing Map (CLRSOM) Learning Algorithm. *J. Inf. Sci. Eng.*, 31(2):387–397.

Christiansen, W. N. (1953). A High-Resolution Aerial for Radio Astronomy. Nature, 171:831–833.

Condon, J. J., Cotton, W. D., Greisen, E. W., Yin, Q. F., Perley, R. A., Taylor, G. B., and Broderick, J. J. (1998). The NRAO VLA Sky Survey. AJ, 115:1693–1716.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3):273–297.

Cover, T. and Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27.

Cutri, R. M. and et al. (2012). VizieR Online Data Catalog: WISE All-Sky Data Release (Cutri+ 2012). *VizieR Online Data Catalog*, page II/311.

Dietterich, T. (1995). Overfitting and Undercomputing in Machine Learning. *ACM Comput. Surv.*, 27(3):326–327.

Dietterich, T. G. (2000). Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pages 1–15. Springer.

Domingos, P. (2012). A Few Useful Things to Know About Machine Learning. *Commun. ACM*, 55(10):78–87.

Dougherty, J., Kohavi, R., and Sahami, M. (1995). Supervised and unsupervised discretization of continuous features. In *Machine Learning Proceedings 1995*, pages 194–202. Elsevier.

Downes, A., Peacock, J., Savage, A., and Carrie, D. (1986). The Parkes selected regions: powerful radio galaxies and quasars at high redshifts. *Monthly Notices of the Royal Astronomical Society*, 218(1):31–62.

Dubath, P., Rimoldini, L., Sveges, M., Blomme, J., Lpez, M., Sarro, L. M., De Ridder, J., Cuypers, J., Guy, L., Lecoeur, I., et al. (2011). Random forest automated supervised classification of Hipparcos periodic variable stars. *Monthly Notices of the Royal Astronomical Society*, 414(3):2602–2617.

Dutta, H., Giannella, C., Borne, K., and Kargupta, H. (2007). Distributed top-k outlier detection from astronomy catalogs using the demac system. In *Proceedings of the 2007 SIAM International Conference on Data Mining*, pages 473–478. SIAM.

Ekers, R. D. (2009). Big and Small. In *Proceedings of the special session Accelerating the Rate of Astronomical Discovery of the 27th IAU General Assembly. August 11-14 2009. Rio de Janeiro, Brazil.*, page 7.

Fadely, R., Hogg, D. W., and Willman, B. (2012). Star-Galaxy Classification in Multi-band Optical Imaging. ApJ, 760:15.

Fan, D., Budavri, T., Norris, R. P., and Hopkins, A. M. (2015). Matching radio catalogues with realistic geometry: application to SWIRE and ATLAS. *Monthly Notices of the Royal Astronomical Society*, 451(2):1299–1305.

Findlay, J. W. (1971). Filled-Aperture Antennas for Radio Astronomy. *Annual Review of Astronomy and Astrophysics*, 9:271.

Flach, P. and Kull, M. (2015). Precision-Recall-Gain Curves: PR Analysis Done Right. In Cortes, C., Lawrence, N. D., Lee, D. D., Sugiyama, M., and Garnett, R., editors, *Advances in Neural Information Processing Systems 28*, pages 838–846. Curran Associates, Inc.

Fritzke, B. (1995). A growing neural gas network learns topologies. In *Advances in neural information processing systems*, pages 625–632.

Galvin, T. J., Huynh, M., Norris, R. P., Wang, R., Hopkins, E., and Polsterer, K. L. (2018). Transferring Knowledge from Radio Galaxy Zoo Data via Rotationally Invariant Self-Organising Maps. *prep.*

Gao, D., Zhang, Y.-X., and Zhao, Y.-H. (2009). Random forest algorithm for classification of multiwavelength data. *Research in Astronomy and Astrophysics*, 9(2):220.

Geach, J. E. (2012). Unsupervised self-organized mapping: a versatile empirical tool for object selection, classification and redshift estimation in large surveys. MNRAS, 419:2633–2645.

Geman, S., Bienenstock, E., and Doursat, R. (1992). Neural networks and the bias/-variance dilemma. *Neural computation*, 4(1):1–58.

Girshick, R., Donahue, J., Darrell, T., and Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680.

Graff, P., Feroz, F., Hobson, M. P., and Lasenby, A. (2014). SkyNet: an efficient and robust neural network training tool for machine learning in astronomy. *Monthly Notices of the Royal Astronomical Society*, 441(2):1741–1759.

Guo, P., Duan, F., Wang, P., Yao, Y., and Xin, X. (2017). Pulsar Candidate Identification with Artificial Intelligence Techniques. *ArXiv e-prints*, page arXiv:1711.10339.

Guo, Y., Liu, Y., Oerlemans, A., Lao, S., Wu, S., and Lew, M. S. (2016). Deep learning for visual understanding: A review. *Neurocomputing*, 187:27–48.

Hao-ran, Q., Ji-ming, L., and Jun-yi, W. (2017). Stacked Denoising Autoencoders Applied to Star/Galaxy Classification. *Chinese Astronomy and Astrophysics*, 41(2):282–292.

Harvey, R. L. (1994). *Neural network principles.* Prentice Hall Englewood Cliffs, NJ.

Heyer, M. H. and Schloerb, F. P. (1997). Application of Principal Component Analysis to Large-Scale Spectral Line Imaging Studies of the Interstellar Medium. *The Astrophysical Journal*, 475(1):173.

Hinton, G. E., Osindero, S., and Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554.

Hinton, G. E. and Salakhutdinov, R. R. (2006). Reducing the Dimensionality of Data with Neural Networks. *Science*, 313(5786):504–507.

Ho, T. K. (1995). Random decision forests. In *Document analysis and recognition, 1995., proceedings of the third international conference on*, volume 1, pages 278–282. IEEE.

Hocking, A., Geach, J. E., Sun, Y., and Davey, N. (2018). An automatic taxonomy of galaxy morphology using unsupervised machine learning. MNRAS, 473:1108–1129.

Hogg, D. W., Casey, A. R., Ness, M., Rix, H.-W., Foreman-Mackey, D., Hasselquist, S., Ho, A. Y. Q., Holtzman, J. A., Majewski, S. R., Martell, S. L., et al. (2016). Chemical Tagging Can Work: Identification of Stellar Phase-space Structures Purely by Chemical-abundance Similarity. ApJ, 833:262.

Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24(6):417.

Hu, W., Singh, R. R. P., and Scalettar, R. T. (2017). Discovering phases, phase transitions, and crossovers through unsupervised machine learning: A critical examination. Phys. Rev. E, 95:062122.

Huertas-Company, M., Rouan, D., Tasca, L., Soucail, G., and Le Fvre, O. (2008). A robust morphological classification of high-redshift galaxies using support vector machines on seeing limited images. I. Method description. A&A, 478:971–980.

Hunter, J. D. (2007). Matplotlib: A 2d graphics environment. *Computing in science & engineering*, 9(3):90–95.

Jansky, K. G. (1933). Radio waves from outside the solar system. *Nature*, 132(3323):66.

Johnston, S., Bailes, M., Bartel, N., Baugh, C., Bietenholz, M., Blake, C., Braun, R., Brown, J., Chatterjee, S., Darling, J., et al. (2007). Science with the Australian square kilometre array pathfinder. *Publications of the Astronomical Society of Australia*, 24(4):174–188.

Jolliffe, I. (2011). Principal component analysis. In *International encyclopedia of statistical science*, pages 1094–1096. Springer.

Jones, E., Oliphant, T., Peterson, P., and others (2001). *SciPy: Open source scientific tools for Python.*

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980.*

Knollmller, J., Frank, P., and Ens slin, T. A. (2018). Separating diffuse from point-like sources - a Bayesian approach. *ArXiv e-prints*, page arXiv:1804.05591.

Kohonen, T. (1997). Exploration of very large databases by self-organizing maps. In *Neural Networks, 1997., International Conference on*, volume 1, pages PL1–PL6. IEEE.

Kononenko, I. (2001). Machine learning for medical diagnosis: history, state of the art and perspective. *Artificial Intelligence in medicine*, 23(1):89–109.

Kotsiantis, S. B., Zaharakis, I., and Pintelas, P. (2007). Supervised machine learning: A review of classification techniques. *Emerging artificial intelligence applications in computer engineering*, 160:3–24.

Kovcs, A. and Szapudi, I. (2015). Star-galaxy separation strategies for WISE-2mass all-sky infrared galaxy catalogues. MNRAS, 448:1305–1313.

Krause, J., Perer, A., and Ng, K. (2016). Interacting with Predictions: Visual Inspection of Black-box Machine Learning Models. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI '16, pages 5686–5697, New York, NY, USA. ACM.

Kremer, J., Stensbo-Smidt, K., Gieseke, F., Steenstrup Pedersen, K., and Igel, C. (2017). Big Universe, Big Data: Machine Learning and Image Analysis for Astronomy. *ArXiv e-prints*, page arXiv:1704.04650.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

Kuo, Y.-H., Li, Z., and Kifer, D. (2018). Detecting Outliers in Data with Correlated Measures. *ArXiv e-prints*.

Langkvist, M., Karlsson, L., and Loutfi, A. (2014). A review of unsupervised feature learning and deep learning for time-series modeling. *Pattern Recognition Letters*, 42:11 – 24.

Larochelle, H., Erhan, D., Courville, A., Bergstra, J., and Bengio, Y. (2007). An empirical evaluation of deep architectures on problems with many factors of variation. In *Proceedings of the 24th international conference on Machine learning*, pages 473–480. ACM.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *nature*, 521(7553):436.

Li, X.-R., Pan, R.-Y., and Duan, F.-Q. (2017). Parameterizing Stellar Spectra Using Deep Neural Networks. *Research in Astronomy and Astrophysics*, 17:036.

Libbrecht, M. W. and Noble, W. S. (2015). Machine learning applications in genetics and genomics. *Nature Reviews Genetics*, 16(6):321.

Liu, Z.-b., Zhou, F.-x., Qin, Z.-t., Luo, X.-g., and Zhang, J. (2018). Classification of stellar spectra with SVM based on within-class scatter and between-class scatter. Ap&SS, 363:140.

Lloyd, S. (1982). Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137.

Lo, K. K., Farrell, S., Murphy, T., and Gaensler, B. M. (2014). Automatic Classification of Time-variable X-Ray Sources. ApJ, 786:20.

LSST Science Collaboration, Abell, P. A., Allison, J., Anderson, S. F., Andrew, J. R., Angel, J. R. P., Armus, L., Arnett, D., Asztalos, S. J., Axelrod, T. S., et al. (2009). LSST Science Book, Version 2.0. *ArXiv e-prints*, page arXiv:0912.0201.

Lukic, V. and Brggen, M. (2017). Galaxy Classifications with Deep Learning. In Brescia, M., Djorgovski, S. G., Feigelson, E. D., Longo, G., and Cavuoti, S., editors, *Astroinformatics*, volume 325, pages 217–220.

Lukic, V., Brggen, M., Banfield, J. K., Wong, O. I., Rudnick, L., Norris, R. P., and Simmons, B. (2018). Radio Galaxy Zoo: compact and extended radio source classification with deep learning. MNRAS, 476:246–260.

Ma, Y., Xiang, Z., Du, Q., and Fan, W. (2018). Effects of user-provided photos on hotel review helpfulness: An analytical approach with deep leaning. *International Journal of Hospitality Management*, 71:120–131.

Maas, A. L., Hannun, A. Y., and Ng, A. Y. (2013). Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml*, volume 30, page 3.

Maaten, L. v. d. and Hinton, G. (2008). Visualizing data using t-SNE. *Journal of machine learning research*, 9(Nov):2579–2605.

MacDonald, D. and Fyfe, C. (2000). The kernel self-organising map. In *Knowledge-Based Intelligent Engineering Systems and Allied Technologies, 2000. Proceedings. Fourth International Conference on*, volume 1, pages 317–320. IEEE.

Mackay, D. J. C. (2003). *Information Theory, Inference and Learning Algorithms.*

MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA.

Malek, K., Solarz, A., Pollo, A., Fritz, A., Garilli, B., Scodeggio, M., Iovino, A., Granett, B., Abbas, U., Adami, C., et al. (2013). The VIMOS Public Extragalactic Redshift Survey (VIPERS)-A support vector machine classification of galaxies, stars, and AGNs. *Astronomy & Astrophysics*, 557:A16.

Marengo, M. and Sanchez, M. C. (2009). A k-NN method to classify rare astronomical sources: Photometric search of brown dwarfs with Spitzer/IRAC. *The Astronomical Journal*, 138(1):63.

Marsland, S., Shapiro, J., and Nehmzow, U. (2002). A self-organising network that grows when required. *Neural networks*, 15(8-9):1041–1058.

Matijeviffh, G., Chiappini, C., Grebel, E. K., Wyse, R., Zwitter, T., Bienayme, O., Bland-Hawthorn, J., Freeman, K. C., Gibson, B. K., Gilmore, G., et al. (2017). Very metal-poor stars observed by the RAVE survey. *Astronomy & Astrophysics*, 603:A19.

McCulloch, W. S. and Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133.

Memisevic, R. and Hinton, G. (2007). Unsupervised learning of image transformations. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE.

Meusinger, H., Schalldach, P., Scholz, R.-D., Newholm, M., de Hoon, A., Kaminsky, B., and others (2012). Unusual quasars from the Sloan Digital Sky Survey selected by means of Kohonen self-organising maps. *Astronomy & Astrophysics*, 541:A77.

Mirabal, N., Nieto, D., and Pardo, S. (2010). The exotic fraction among unassociated Fermi sources. *ArXiv e-prints*, page arXiv:1007.2644.

Mitchell, M. (1998). *An introduction to genetic algorithms.* MIT press.

Molinari, E. and Smareglia, R. (1998). Neural network method for galaxy classification: the luminosity function of E/S0 in clusters. A&A, 330:447–452.

Morales-Luis, A. B., Snchez Almeida, J., Aguerri, J. A. L., and Muoz-Tun, C. (2011). Systematic Search for Extremely Metal-poor Galaxies in the Sloan Digital Sky Survey. ApJ, 743:77.

Morgan, J. N. and Sonquist, J. A. (1963). Problems in the analysis of survey data, and a proposal. *Journal of the American statistical association*, 58(302):415–434.

Morice-Atkinson, X., Hoyle, B., and Bacon, D. (2017). Learning from the machine: interpreting machine learning algorithms for point- and extended- source classification. *ArXiv e-prints*, page arXiv:1712.03970.

Mustafa, M., Bard, D., Bhimji, W., Al-Rfou, R., and Lukifj, Z. (2017). Creating Virtual Universes Using Generative Adversarial Networks. *arXiv preprint arXiv:1706.02390.*

Naim, A., Ratnatunga, K. U., and Griffiths, R. E. (1997). Galaxy morphology without classification: Self-organizing maps. *The Astrophysical Journal Supplement Series*, 111(2):357.

Ng, M. K. and Huang, Z. (1999). Data-mining massive time series astronomical data: challenges, problems and solutions. *Information and Software Technology*, 41(9):545 – 556.

Norris, R. P. (2017a). Discovering the unexpected in astronomical survey data. *Publications of the Astronomical Society of Australia*, 34.

Norris, R. P. (2017b). Extragalactic radio continuum surveys and the transformation of radio astronomy. *Nature Astronomy*, 1(10):671.

Norris, R. P., Hopkins, A. M., Afonso, J., Brown, S., Condon, J. J., Dunne, L., Feain, I., Hollow, R., Jarvis, M., Johnston-Hollitt, M., et al. (2011). EMU: Evolutionary map of the universe. *Publications of the Astronomical Society of Australia*, 28(3):215–248.

Odewahn, S. C. (1995). Automated Classification of Astronomical Images. *Publications of the Astronomical Society of the Pacific*, 107:770.

Oliphant, T. E. (2006). *A guide to NumPy*, volume 1. Trelgol Publishing USA.

Opitz, D. and Maclin, R. (1999). Popular ensemble methods: An empirical study. *Journal of artificial intelligence research*, 11:169–198.

Pan, R.-y. and Li, X.-r. (2017). Stellar Atmospheric Parameterization Based on Deep Learning. Chinese Astron. Astrophys., 41:318–330.

Pascanu, R., Mikolov, T., and Bengio, Y. (2013). On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning*, pages 1310–1318.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Perantonis, S. J. and Lisboa, P. J. (1992). Translation, rotation, and scale invariant pattern recognition by high-order neural networks and moment classifiers. *IEEE Transactions on Neural Networks*, 3(2):241–251.

Poliszczuk, A., Solarz, A., Pollo, A., and NEP-Deep Team (2018). Searching for previously unknown classes of objects in the AKARI-NEP Deep data with fuzzy logic SVM algorithm. In *The Cosmic Wheel and the Legacy of the AKARI archive: from galaxies and stars to planets and life. Edited by Takafumi Ootsubo*, pages 375–378.

Polsterer, K., Gieseke, F. C., Igel, C., Doser, B., and Gianniotis, N. (2016). Parallelized rotation and flipping INvariant Kohonen maps (PINK) on GPUs.

Polsterer, K. L., Gieseke, F., and Igel, C. (2015). Automatic Galaxy Classification via Machine Learning Techniques: Parallelized Rotation/Flipping INvariant Kohonen Maps (PINK). In *Astronomical Data Analysis Software an Systems XXIV (ADASS XXIV)*, volume 495, page 81. San Francisco: ASP.

Pruyt, E., Cunningham, S., Kwakkel, J., and De Bruijn, J. (2014). From data-poor to data-rich: system dynamics in the era of big data. In *32nd International Conference of the System Dynamics Society, Delft, The Netherlands, 20-24 July 2014; Authors version*. The System Dynamics Society.

Radford, A., Metz, L., and Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*.

Ralph, N. O., Norris, R. P., Fang, G., Park, L. A. F., Galvin, T. J., Alger, M., Andernach, H., Lintott, C., Rudnick, L., Shabala, S., et al. (2018). Radio Galaxy Zoo: Unsupervised Clustering of Convolutionally Encoded Radio-astronomical Images. *PASP, submitted*.

Rauber, A., Merkl, D., and Dittenbach, M. (2002). The growing hierarchical self-organizing map: exploratory analysis of high-dimensional data. *IEEE Transactions on Neural Networks*, 13(6):1331–1341.

Ravanbakhsh, S., Lanusse, F., Mandelbaum, R., Schneider, J., and Poczos, B. (2016). Enabling Dark Energy Science with Deep Generative Models of Galaxy Images. *ArXiv e-prints*, page arXiv:1609.05796.

Reber, G. (1940). Notes: Cosmic Static. ApJ, 91:621–624.

Rees, D. E., Lpez Ariste, A., Thatcher, J., and Semel, M. (2000). Fast inversion of spectral lines using principal component analysis. I. Fundamentals. A&A, 355:759–768.

Richards, J. W., Starr, D. L., Butler, N. R., Bloom, J. S., Brewer, J. M., Crellin-Quick, A., Higgins, J., Kennedy, R., and Rischard, M. (2011). On Machine-learned Classification of Variable Stars with Sparse and Noisy Time-series Data. ApJ, 733:10.

Riess, A. G., Filippenko, A. V., Challis, P., Clocchiatti, A., Diercks, A., Garnavich, P. M., Gilliland, R. L., Hogan, C. J., Jha, S., Kirshner, R. P., et al. (1998). Observational Evidence from Supernovae for an Accelerating Universe and a Cosmological Constant. AJ, 116:1009–1038.

Robbins, H. and Monro, S. (1951). A Stochastic Approximation Method, Annals Math. *Statistics*, 22:400–407.

Rodriguez, A. C., Kacprzak, T., Lucchi, A., Amara, A., Sgier, R., Fluri, J., Hofmann, T., and Rfrgier, A. (2018). Fast Cosmic Web Simulations with Generative Adversarial Networks. *ArXiv e-prints*, page arXiv:1801.09070.

Rohlfs, K. and Wilson, T. L. (2013). *Tools of radio astronomy*. Springer Science & Business Media.

Ronen, S., Aragn-Salamanca, A., and Lahav, O. (1999). Principal component analysis of synthetic galaxy spectra. *Monthly Notices of the Royal Astronomical Society*, 303(2):284–296.

Roweis, S. T. and Saul, L. K. (2000). Nonlinear dimensionality reduction by locally linear embedding. *science*, 290(5500):2323–2326.

Ruder, S. (2016). An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*.

Ryle, M. (1952). A new radio interferometer and its application to the observation of weak radio stars. *Proc. R. Soc. Lond. A*, 211(1106):351–375.

Salvato, M., Ilbert, O., and Hoyle, B. (2018). The many flavours of photometric redshifts. *Nature Astronomy*, page 68.

Salzberg, S., Chandar, R., Ford, H., Murthy, S. K., and White, R. (1995). Decision trees for automated identification of cosmic-ray hits in Hubble Space Telescope images. PASP, 107:279–288.

Sanger, T. D. (1989). Optimal unsupervised learning in a single-layer linear feedforward neural network. *Neural networks*, 2(6):459–473.

Schawinski, K., Turp, D., and Zhang, C. (2018). Exploring galaxy evolution with latent space walks. In *American Astronomical Society Meeting Abstracts #231*, volume 231, page 309.01.

Schawinski, K., Zhang, C., Zhang, H., Fowler, L., and Krishnan Santhanam, G. (2017a). *GalaxyGAN: Generative Adversarial Networks for recovery of galaxy features*. Published: Astrophysics Source Code Library.

Schawinski, K., Zhang, C., Zhang, H., Fowler, L., and Santhanam, G. K. (2017b). Generative adversarial networks recover features in astrophysical images of galaxies beyond the deconvolution limit. *Monthly Notices of the Royal Astronomical Society: Letters*, 467(1):L110–L114.

Schleicher, D. G. and Bair, A. (2010). A Compositional Taxonomy of Comets and Narrowband Photometry of EPOXI Target 103p/Hartley 2. In *Bulletin of the American Astronomical Society*, volume 42, page 945.

Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural Networks*, 61:85 – 117.

Sedlmair, M., Tatu, A., Munzner, T., and Tory, M. (2012). A taxonomy of visual cluster separation factors. In *Computer Graphics Forum*, volume 31, pages 1335–1344. Wiley Online Library.

Seymour, N., Dwelly, T., Moss, D., McHardy, I., Zoghbi, A., Rieke, G., Page, M., Hopkins, A., and Loaring, N. (2008). The star formation history of the Universe as revealed by deep radio observations. *Monthly Notices of the Royal Astronomical Society*, 386(3):1695–1708.

Shen, H., George, D., Huerta, E. A., and Zhao, Z. (2017). Denoising Gravitational Waves using Deep Learning with Recurrent Denoising Autoencoders. *ArXiv e-prints*, page arXiv:1711.09919.

Singh, H. P., Gulati, R. K., and Gupta, R. (1998). Stellar spectral classification using principal component analysis and artificial neural networks. *Monthly Notices of the Royal Astronomical Society*, 295(2):312–318.

Singh, S. and Bard, D. (2017). Deep Learning the Universe. In *American Astronomical Society Meeting Abstracts #229*, volume 229, page 430.03.

Smolensky, P. (1986). Information processing in dynamical systems: Foundations of harmony theory. Technical report, COLORADO UNIV AT BOULDER DEPT OF COMPUTER SCIENCE.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958.

Srivastava, N. and Salakhutdinov, R. R. (2012). Multimodal learning with deep boltzmann machines. In *Advances in neural information processing systems*, pages 2222–2230.

Stark, D., Launet, B., Schawinski, K., Zhang, C., Koss, M., Turp, M. D., Sartori, L. F., Zhang, H., Chen, Y., and Weigel, A. K. (2018). psfgan: a generative adversarial network system for separating quasar point sources and host galaxy light. *Monthly Notices of the Royal Astronomical Society*, 477(2):2513–2527.

Storrie-Lombardi, M. C., Irwin, M. J., Hippel, T. v., and Storrie-Lombardi, L. J. (1994). Spectral classification with principal component analysis and artificial neural networks. *Vistas in Astronomy*, 38:331 – IN8.

Suchkov, A. A., Hanisch, R. J., and Margon, B. (2005). A Census of Object Types and Redshift Estimates in the SDSS Photometric Catalog from a Trained Decision Tree Classifier. AJ, 130:2439–2452.

Sutherland, W. and Saunders, W. (1992). On the likelihood ratio for source identification. *Monthly Notices of the Royal Astronomical Society*, 259(3):413–420.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9.

Snchez Almeida, J., Aguerri, J. A. L., Muoz-Tun, C., and de Vicente, A. (2010). Automatic Unsupervised Classification of All Sloan Digital Sky Survey Data Release 7 Galaxy Spectra. ApJ, 714:487–504.

Snchez Almeida, J., Prez-Montero, E., Morales-Luis, A. B., Muoz-Tun, C., Garca-Benito, R., Nuza, S. E., and Kitaura, F. S. (2016). Search for Extremely Metal-poor Galaxies in the Sloan Digital Sky Survey. (II). High Electron Temperature Objects. ApJ, 819:110.

Tagliaferri, R., Longo, G., Andreon, S., Capozziello, S., Donalek, C., and Giordano, G. (2003). Neural Networks for Photometric Redshifts Evaluation. In *Neural Nets. Lecture Notes in Computer Science, Volume 2859. ISBN 978-3-540-20227-1. Springer Berlin Heidelberg, 2003. p. 226-234*, pages 226–234.

Tarter, J. (2001). The Search for Extraterrestrial Intelligence (SETI). *Annual Review of Astronomy and Astrophysics*, 39:511–548.

Tenenbaum, J. B., De Silva, V., and Langford, J. C. (2000). A global geometric framework for nonlinear dimensionality reduction. *science*, 290(5500):2319–2323.

Tishby, N., Pereira, F. C., and Bialek, W. (2000). The information bottleneck method. *arXiv preprint physics/0004057*.

Torniainen, I., Tornikoski, M., Turunen, M., Lainela, M., Lhteenmki, A., Hovatta, T., Mingaliev, M., Aller, M., and Aller, H. (2008). Cluster analyses of gigahertz-peaked spectrum sources with self-organising maps. *Astronomy & Astrophysics*, 482(2):483–498.

Torralba, A. and Efros, A. A. (2011). Unbiased look at dataset bias. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1521–1528. IEEE.

Traven, G., Matijeviffh, G., Zwitter, T., erjal, M., Kos, J., Asplund, M., Bland-Hawthorn, J., Casey, A. R., Silva, G. D., Freeman, K., et al. (2017). The Galah Survey: Classification and Diagnostics with t-SNE Reduction of Spectral Information. *The Astrophysical Journal Supplement Series*, 228(2):24.

VanderPlas, J., Connolly, A. J., Ivezic, Z., and Gray, A. (2012). Introduction to astroML: Machine learning for astrophysics. In *Proceedings of Conference on Intelligent Data Understanding (CIDU*, pages 47–54.

Vincent, P., Larochelle, H., Lajoie, I., Bengio, Y., and Manzagol, P.-A. (2010). Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*, 11(Dec):3371–3408.

Wagstaff, K. L. and Laidler, V. G. (2005). Making the most of missing values: Object clustering with partial data in astronomy. In *Astronomical data analysis software and systems XIV*, volume 347, page 172.

Wang, B., Li, H., and Sun, D. (2014). Social-Ecological patterns of soil heavy metals based on a Self-Organizing Map (SOM): A case study in Beijing, China. *International journal of environmental research and public health*, 11(4):3618–3638.

Weatherill, G. and Burton, P. W. (2009). Delineation of shallow seismic source zones using K-means cluster analysis, with application to the Aegean region. *Geophysical Journal International*, 176:565–588.

Weir, N., Fayyad, U. M., and Djorgovski, S. (1995). Automated Star/Galaxy Classification for Digitized Poss-II. AJ, 109:2401.

Wittek, P., Gao, S. C., Lim, I. S., and Zhao, L. (2013). Somoclu: An efficient parallel library for self-organizing maps. *Journal of Statistical Software, 78, 1 (2017).*

Wong, O. I. (2018). Radio Galaxy Zoo: Data Release 1. *prep.*

Wright, E. L., Eisenhardt, P. R., Mainzer, A. K., Ressler, M. E., Cutri, R. M., Jarrett, T., Kirkpatrick, J. D., Padgett, D., McMillan, R. S., Skrutskie, M., et al. (2010). The Wide-field Infrared Survey Explorer (WISE): mission description and initial on-orbit performance. *The Astronomical Journal*, 140(6):1868.

Wu, C., Wong, O., Rudnick, L., Shabala, S. S., Alger, M. J., Banfield, J. K., Ong, C. S., White, S. V., Garon, A. F., Norris, R. P., et al. (2018). Radio Galaxy Zoo: ClaRAN-A Deep Learning Classifier for Radio Morphologies. *Monthly Notices of the Royal Astronomical Society arXiv:1805.12008.*

Xie, J., Xu, L., and Chen, E. (2012). Image denoising and inpainting with deep neural networks. In *Advances in neural information processing systems*, pages 341–349.

Xu, B., Wang, N., Chen, T., and Li, M. (2015). Empirical evaluation of rectified activations in convolutional network. *arXiv preprint arXiv:1505.00853*.

York, D. G., Adelman, J., Anderson Jr, J. E., Anderson, S. F., Annis, J., Bahcall, N. A., Bakken, J., Barkhouser, R., Bastian, S., Berman, E., et al. (2000). The sloan digital sky survey: Technical summary. *The Astronomical Journal*, 120(3):1579.

Zhang, Y. and Zhao, Y. (2004). Automated clustering algorithms for classification of astronomical objects. A&A, 422:1113–1121.

Zhao, Y. and Zhang, Y. (2008). Comparison of decision tree methods for finding active objects. *Advances in Space Research*, 41(12):1955 – 1959.

Zhao, Z., Zhang, X., and Fang, Y. (2015). Stacked multilayer self-organizing map for background modeling. *IEEE Transactions on Image Processing*, 24(9):2841–2850.

Zhou, L., Pan, S., Wang, J., and Vasilakos, A. V. (2017). Machine learning on big data: Opportunities and challenges. *Neurocomputing*, 237:350 – 361.

Zhu, X. (2006). Semi-supervised learning literature survey. *Computer Science, University of Wisconsin-Madison*, 2(3):4.

Zingales, T. and Waldmann, I. P. (2018). ExoGAN: Retrieving Exoplanetary Atmospheres Using Deep Convolutional Generative Adversarial Networks. *ArXiv e-prints*, page arXiv:1806.02906.

Zitlau, R., Hoyle, B., Paech, K., Weller, J., Rau, M. M., and Seitz, S. (2016). *Monthly Notices of the Royal Astronomical Society*, 460(3):3152–3162.

Appendix A

Submitted Paper

The following paper was submitted to the Machine Intelligence in Astronomy and Astrophysics Special Issue of the Publications of the Astronomical Society of the Pacific on July 24, 2018, and is currently under the process of review. The distribution of work between the authors is summarised as:

- First author Nicholas Ralph (myself) as the primary investigator, developing the software pipeline and writing of the paper.
- Authors Professor Ray Norris and Associate Professor Gu Fang as academic supervisors who provided important technical, field-specific direction and proofing.
- Authors Dr Laurence Park and Dr Timothy Galvin, provided deep contributions to ensuring the paper kept to ML best practices. Dr Galvin provided code and guidance for the preprocessing stages, as indicated in the paper and Section 2.3 of this thesis.
- The remaining authors are the RGZ collaborators. These collaborators provided the DR1 dataset with guidance in its use. Additionally, the collaborators proofread the paper and provide vital detailed feedback throughout the investigation.

Draft version July 14, 2018
Typeset using LaTeX **twocolumn** style in AASTeX62

Radio Galaxy Zoo: Unsupervised Clustering of Convolutionally Encoded Radio-astronomical Images

Nicholas O. Ralph,[1,2] Ray P. Norris,[1,2] Gu Fang,[1] Laurence A. F. Park,[1] Timothy J. Galvin,[3] Matthew Alger,[4,5] Heinz Andernach,[6] Chris Lintott,[7] Lawrence Rudnick,[8] Stanislav Shabala,[9] and O. Ivy Wong[10]

[1] *Western Sydney University, School of Computing, Engineering and Mathematics Locked Bag 1797, Penrith NSW 2751, Australia*
[2] *CSIRO Astronomy and Space Science, Australia Telescope National Facility, PO Box 76, Epping, NSW 1710, Australia*
[3] *CSIRO Astronomy and Space Science Kensington WA 6151, Australia*
[4] *Research School of Astronomy and Astrophysics, The Australian National University, Canberra, ACT 2611, Australia*
[5] *Data61, CSIRO, Canberra, ACT 2601, Australia*
[6] *Departamento de Astronomía, DCNE, Universidad de Guanajuato, Apdo. Postal 144, CP 36000, Guanajuato, Gto., Mexico*
[7] *Department of Physics, University of Oxford, Denys Wilkinson Building, Keble Road, Oxford, OX1 3RH, UK*
[8] *Minnesota Institute for Astrophysics, University of Minnesota*
[9] *University of Tasmania, School of Natural Sciences, Private Bag 37, Hobart, Tasmania 7001, Australia*
[10] *International Centre for Radio Astronomy (ICRAR), M468, The University of Western Australia, 35 Stirling Highway, Crawley, WA 6009, Australia*

ABSTRACT

This paper demonstrates a novel and efficient unsupervised clustering method with the combination of a self-organising map (SOM) and a convolutional autoencoder (CAe). The rapidly increasing volume of radio-astronomical data has increased demand for machine learning methods as solutions to classification and outlier detection. Major astronomical discoveries are unplanned and found in the unexpected (Norris 2017a), making unsupervised machine learning highly desirable by operating without assumptions and labeled training data. Our approach shows SOM training time is drastically reduced and high level features can be clustered by training on autoencoded feature vectors instead of raw images. Our results demonstrate this method is capable of accurately locating outliers on a SOM with neighborhood similarity and k-means clustering of feature complexity. We show this method as a powerful new approach to data exploration by providing a detailed understanding of the morphology and relationships of RGZ dataset image features which can be applied to new radio survey data.

Keywords: Machine Learning, Astronomy, Autoencoder, Self-Organising Map, Unsupervised Clustering

1. INTRODUCTION

Large radio continuum surveys have played a key role in our understanding of the evolution of galaxies (Norris 2017b). Exceptionally large surveys such as the Evolutionary Map of the Universe (Norris et al. 2011, EMU) are expected to detect over 70 million radio sources. The sheer scale and complexity of this data is pushing researchers towards automated techniques such as machine learning with Neural Networks (NN).

NN's are networks of non-linear parameterised functions termed "neurons" that operate as function approx-

imators. Neuron parameters are learned via backpropagation, where weights are updated using gradient descent of a given loss function as a difference measure between input and output prediction.
A NN trained to classify images with a specific orientation and scale will however encounter difficulties when classifying the same training image at an untrained angle or scale (Perantonis & Lisboa 1992). Affine transformations such as rotation, scaling and translation, are a common cause of machine learning prediction errors. A classical solution involves augmenting a training set with random rotations and scaling at the cost of training time. Alternatively, a network can be made invariant to scaling by adding convolutional and max pooling layers. Rotational invariance is more easily solved with the addition of rotated training images.

NN's such as SkyNet (Graff et al. 2014) have accurately classified astronomical data using supervised learning of pre-classified examples. Efforts to use these supervised neural networks have been supported with citizen science projects such as the Radio Galaxy Zoo (RGZ) (Banfield et al. 2015) which has created large labeled datasets of radio sources. This RGZ dataset has been used to successfully train classifiers for source classification (Wu et al. 2018; Lukic et al. 2018) and radio source host galaxy cross identification (Alger et al. 2018). However, this supervised training is not always suitable in outlier detection as it requires a more complete knowledge of all potential classes of new unseen data. Given that most of the major discoveries in astronomy have been unplanned (Norris 2017a), this is a major shortcoming.

Unsupervised learning techniques such as autoencoders (Ae) bridge this gap by working with no assumptions about input data. An autoencoder is an NN variant designed for unsupervised dimensionality reduction. Autoencoders work by extracting and compressing the features of input images into a feature vector (Sanger 1989). The ideal autoencoder is trained to perfectly compress and restore input data with no loss. The layer configuration of an autoencoder (Figure 2) reduces input data to a compact feature representation on the encoding side before returning it to its original form on the decoding side. The layer configuration of the encoder and decoder are usually very similar. Autoencoder loss as the difference between the ground truth and the prediction is derived from the difference between input data and the decoded output. This loss is naturally an indicator of the performance of the network but is also sensitive to differences between an input image and the training set. Since loss is calculated from the input data, a label set is not required and the network can be trained unsupervised. Autoencoders have seen success in many image processing applications with the addition of convolution, max pooling and denoising architecture (Xie et al. 2012).

Abstract relationships and topology in large datasets can easily be interpreted by visualising Ae encoded feature vectors with dimensionality reduction methods such as Principle Component Analysis (PCA) (Hotelling 1933) and T-distributed Stochastic Neighbor Embedding (T-SNE) (Maaten & Hinton 2008) onto a learning manifold. More complex approaches such as Self-Organising maps (SOM) (Kohonen 1997) have also been recognised as powerful unsupervised data exploration tools in astronomy (Polsterer et al. 2015). By adapting to the shape of encoded latent vectors, these SOM's can display various topological relationships and morphology distributions. Moreover, these algorithms have been augmented to produce labeled classification with K-Means clustering (Lloyd 1982).

This paper demonstrates a novel and efficient unsupervised clustering by combining a self-organising map with a convolutional autoencoder. SOM training time is drastically reduced by training on the compressed autoencoder feature vectors of the RGZ image. This method is demonstrated as a powerful data exploration and visualisation tool. This approach shows k-means clustering of trained SOM weights as a method of grouping complexity with accurate outlier detection. We demonstrate our method as a solution to understanding the morphology and relationships of RGZ images that can be applied to unexplored fields for discovery purposes. The use of abstracted image representations as autoencoded feature vectors are a significant novel aspect of our method and offer great advantages compared to prior work analysing complete images alone. As the use of an abstracted representation of the data such as that provided by the auto encoder offer significant advantages, we adopt them here; this is a significant novel aspect of our method compared to prior work.

2. DATA

Radio Galaxy Zoo is a citizen science project for radio image classification by volunteers via web interface (Banfield et al. 2015). The majority of the radio image data in Radio Galaxy Zoo comes from the 1.4 GHz Faint Images of the Radio Sky at Twenty-cm (FIRST) survey catalogue (Becker et al. 1995) version 14 March 2004. FIRST covers over 9000 square degrees of the northern sky down to a 1 σ noise level of 150 μJy beam^{-1} at $5''$ resolution.

Table 1. Dataset labels by population

Class Label	Class Population
11	63.78%
22	14.3%
12	13.86%
33	3.28%
23	1.94%
13	0.78%
44	0.76%
34	0.42%
45	0.2%
24	0.16%
55	0.14%
35	0.08%
14	0.08%
46	0.08%
56	0.04%
16	0.02%
15	0.02%
36	0.02%
57	0.02%
67	0.02%

A random sample of 10,000 FIRST images from the RGZ Data Release 1 catalog is used in this paper (Wong 2018).

Hand labelled RGZ annotations of the dataset contain the number of components for every resolved source in the image (Banfield et al. 2015). These annotations also include the number of peaks above a set threshold within an image. We have encoded these labels as components-peaks, e.g single component with a single peak is 11, two components with two peaks is 22. Table 1 shows that the largest fraction of the dataset contains single component single peak sources.

3. IMAGE PREPROCESSING

Radio images are contaminated by remnants of the instrument's Point Spread Function (PSF). Termed "sidelobes", this contamination is often a major component of the feature space of the RGZ training set. These elements must be removed, as unsupervised clustering methods are initialised with the most dominant features. We found that without proper preprocessing, clustering resulted in two classes: "noisy" and "not noisy", distinguished only by intensity distribution. Early preprocessing methods used in this investigation were effective at removing noise but had a tendency to remove faint sources and produce artifacts. As a result, we adopted the preprocessing method of Galvin et al. (2018) with results shown in Figure 1. This approach corrects blank pixels in images at the edge of the FIRST image mosaic, sigma clips noise and normalises pixel intensity.

1. Blank pixel regions found in images close to the edge of the FIRST mosaic are corrected. This correction replaces these masked values with a random sample of the mean and standard deviation of valid pixels around the outer edge region of the image (assuming a normal distribution). These samples are extracted from the outer regions of the image with few astronomical features to properly sample the background noise.

2. Noise is removed and background flux is corrected with sigma clipping. This operation subtracts the mean background pixel value and scales all pixel intensities below 1 σ to zero.

3. Intensity scaling is applied to normalise the global intensity of each image.

4. All images are additionally cropped for the purposes of this paper to 120x120 from the center to reduce the dataset size while preserving salient features.

4. METHOD

In this section we outline our method of reducing RGZ images with convolutional autoencoding to a compact feature vector for clustering and visualisation using a self-organising map and the T-SNE algorithm. These methods were developed using the Python Language on CPU only with a Intel(R) Xeon(R) CPU E5-2650 v4 at 2.20GHz. The Google Tensorflow Library (Abadi et al. 2016) was used to create the Ae, and Somoclu (Wittek et al. 2013) was used to implement the SOM.

4.1. *Affine Invariant Convolutional Autoencoders*

In our method we extract the latent relationships of RGZ image features using a convolutional autoencoder.

We use a convolutional autoencoder with three convolutional layers. Figure 2 shows the autoencoder architecture with each convolutional layer contained 1, 64 and 1 filters both on the encoder and decoder sides for scale and translation invariance. All convolution layers use a kernel size of 3x3, with a stride of 2x2. The encoder output layer is a max-pooling operation with a kernel size of 3x3 and a stride of 1x1. The decoder input layer restores the latent vector to its dimensions before max pooling with a linear interpolator. A latent vector with a 900x1 shape is the consequence of the number and dimensions of kernels used in the decoder. These dimensions can

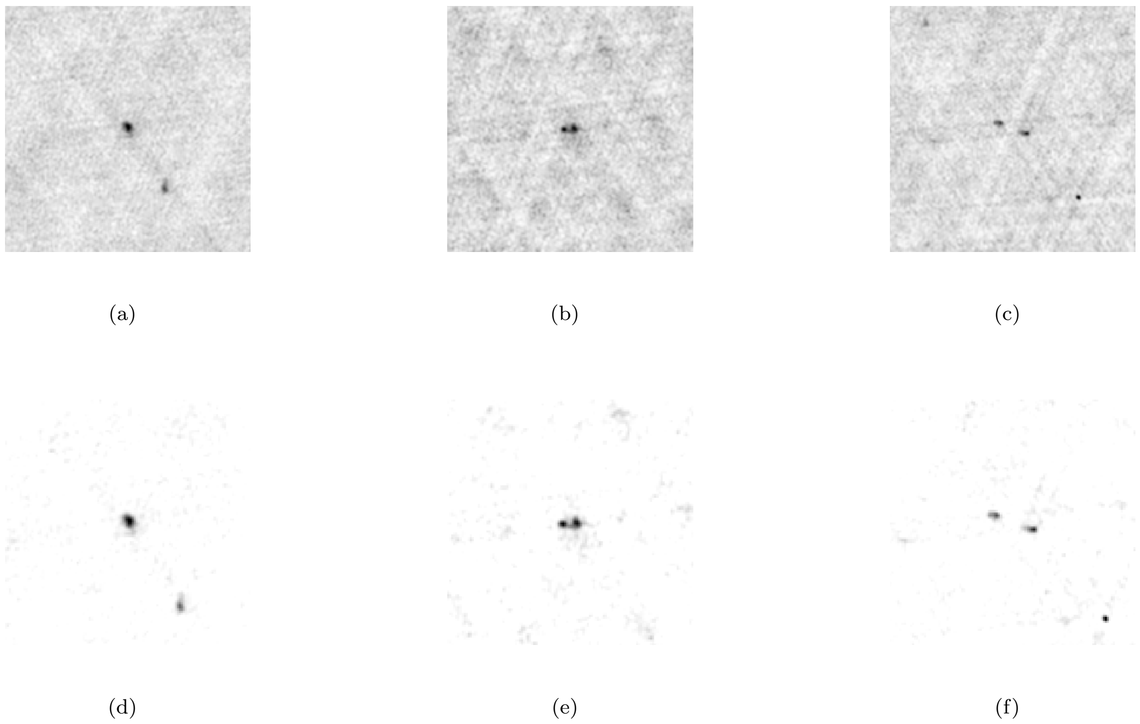

Figure 1. RGZ image preprocessing, (a),(b) and (c) as unprocessed FIRST images. Images (d), (e) and (f) shown as the preprocessing output with noticeable improvement to noise.

be modified by scaling the input image, however it was found that training converged quickly with this 900x1 latent vector. The dimensions of this vector represent a significant reduction to the original image dimensions (120x120, 14400x1) while still containing sufficient free parameters to preserve information for decompression with minimum error.

Rotational invariance is achieved by randomly rotating input images during training. Loss is calculated as the pixel square difference between the autoencoder prediction image and the original rotated input image. This rotational invariance prevents clustering methods from recognising rotation as a feature distinguished enough to separate it from its class. As the autoencoder is still being trained on rotation, these features will still be encoded into the latent vectors but with less weight.

4.2. *Self Organising Maps*

Self-Organising Maps (SOM) are data analysis methods used in unsupervised clustering and data exploration. SOMs create similarity maps or learning manifolds of input data where distinct groups of neurons reflect latent clusters in the data. Neurons on this grid are trained to move toward similar data points on the grid while moving dissimilar neurons apart. A well trained SOM will visualise the dynamic distribution of input data and high-level topological relationships.

The key input parameters for a SOM are the number of neurons expressed as map size, the rate in which neurons move as learning rate and update distance as learning radius. The learning rate and radius is reduced after each iteration (epoch) by decay rates balanced to let all neurons stabilise in optimal time. Our SOM was trained using the following procedure:

1. Initialise grid neurons in the input data feature space by PCA clustering with the SOM as the manifold.

2. Select a unit of data (feature vector in our case) randomly.

3. Locate the Best Matching Unit (BMU) as the 'closest' neuron to the selected data point.
   A common and reliable distance metric used to calculate this is Euclidean distance (Eq 1).

$$d(\vec{x}, \vec{y}) = \sqrt{\sum_{i=1}^{n} (y_i - x_i)^2} \tag{1}$$

4. Move the BMU neuron toward the data point as a function of the distance metric and learning rate.

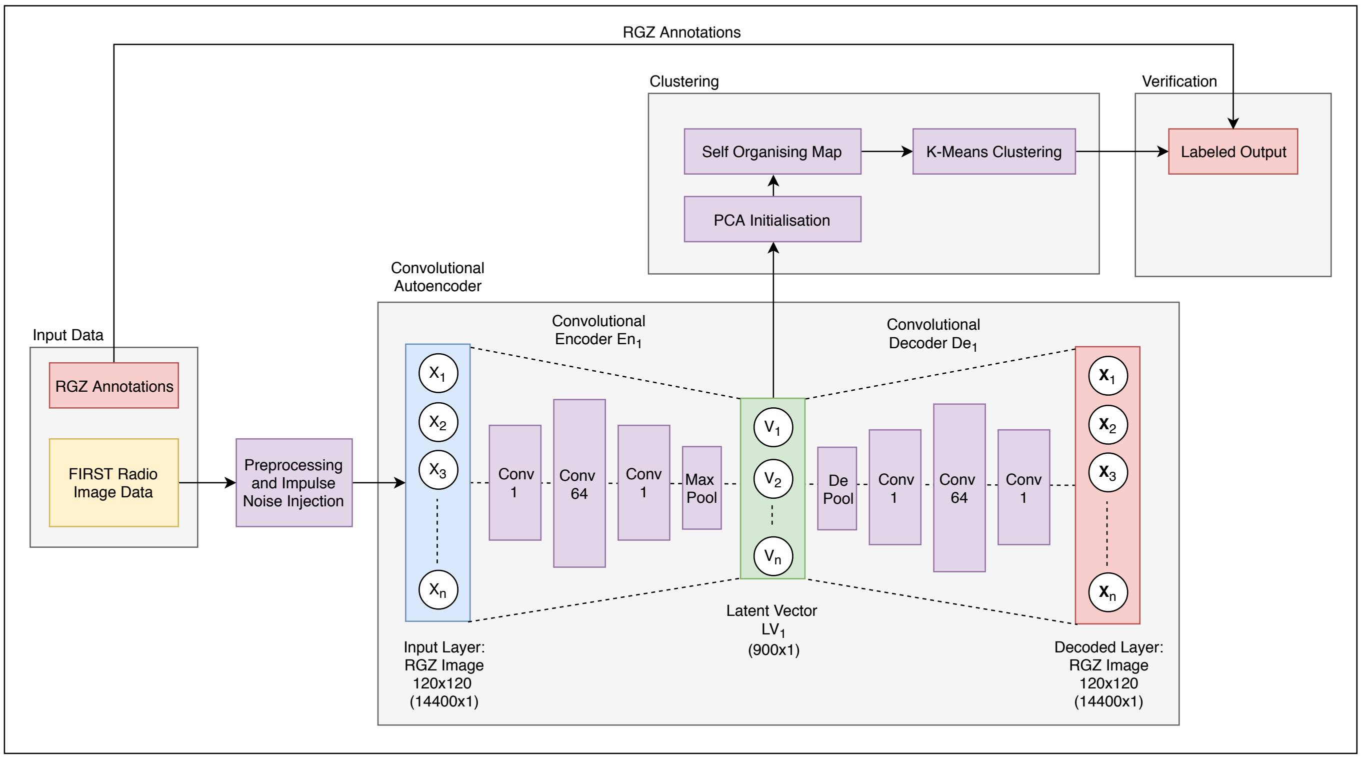
RGZ Annotations
Clustering
Verification
Self Organising Map
K-Means Clustering
Labeled Output
PCA Initialisation
Convolutional Autoencoder
Input Data
RGZ Annotations
FIRST Radio Image Data
Preprocessing and Impulse Noise Injection
Convolutional Encoder En1
Convolutional Decoder De1
Conv 1
Conv 64
Conv 1
Max Pool
De Pool
Conv 1
Conv 64
Conv 1
Latent Vector LV1 (900x1)
Input Layer: RGZ Image 120x120 (14400x1)
Decoded Layer: RGZ Image 120x120 (14400x1)

Figure 2. Overall pipeline configuration with the convolutional autoencoder architecture and self-organising map used in this paper. This network compresses input images with 14400 elements to the latent feature vector with 900 elements for clustering. Encoder and decoder architecture is identical with three convolutional layers at 1, 64 and 1 layers deep selected experimentally. The SOM weights are initialised using PCA and trained on the encoded latent vectors. K-means clustering is applied to SOM weights and labeled for verification to compare the map clusters against ground truth.

5. Move all BMU neurons within the learning radius toward the data point as a function of the learning rate and grid distance.

6. Update learning rate and radius based on respective input decay rates.

7. Iterate until neurons have converged and map is stable

8. Classify neuron distance with k-means clustering. K-means segments neurons on the SOM into k clusters (2 in our case) by its Euclidean distance (Lloyd 1982). This algorithm groups objects by assigning inputs a clusters based on a metric such as Euclidean distance. This is an iterative process where the distance between each cluster is calculated as the average distance of its consistent objects. Input clusters are continually refined based on this distance until the changes in each cluster reach a stop condition. These clusters are discrete, where an object is assigned to only one cluster. We use these two clusters as proxies for complex and simple feature vectors on the SOM. K-means clustering was implemented using the Scikit learn package (Pedregosa et al. 2011).

The SOM output is a unified distance matrix (umat). This umat is visualised as a heat map of the euclidean distance between each neuron and its neighborhood. Labels indicating the annotation of the first BMU, K-means cluster and entropy are also displayed for each neuron. Entropy, $\hat{E}$ is used here as a metric to describe the distribution labels in the BMU's of each neuron.

$$\hat{P}_i = \frac{n_i}{N} \tag{2}$$

$$\hat{E} = \sum_i \hat{P}_i log_2 \hat{P}_i \tag{3}$$

where n_i is the number of class occurrences i and N the total number of occurring classes. A low entropy indicates good consensus where most BMU's have the same label. Conversely, a high entropy indicates the BMU's of a neuron has a wide range of different labels.

5. RESULTS

This section outlines the results and performance of our approach at each stage of the method.

5.1. *Autoencoder Training and Image Reconstruction*

The autoencoder trained on RGZ images demonstrates successful compression and decompression across the dataset. This is demonstrated in Figure 4 where the reconstructed image strongly approximates the input image. From this figure we determine the autoencoder is capable of recognising and preserving enough key image features to successfully predict the original image from the compressed latent vector. The difference images in the figure show the autoencoder loses most fidelity around the edges of regions and reconstructs background noise with low error. Blurring in the reconstructed image has a square kernel shape and is expected due to the shape of the max pooling and convolutional layers. Ae difference images also show the background noise of each image. Additional layers and training may allow the autoencoder to better generalise the dataset image features to remove this blurring and background noise. The average training time of this autoencoder is 1.2 seconds per batch of 128 images. The training time for a full epoch is 2.2 minutes. Figure 3 shows that training converges shortly after 70 epochs.

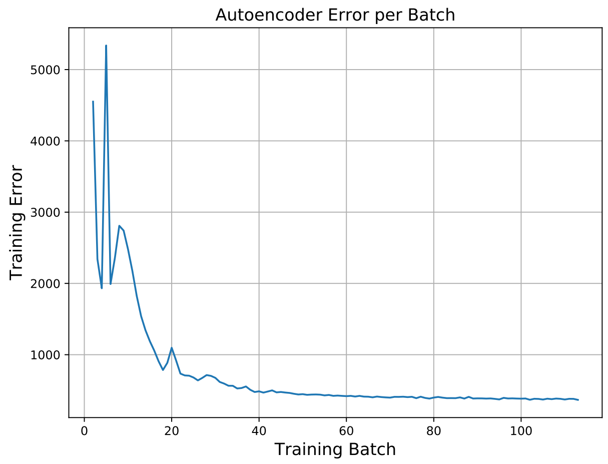

Figure 3. Autoencoder error per batch as mean squared difference between input target image and reconstructed image. Training error converges after 70 epochs. Total training time for a full epoch is 2.2 minutes.

5.2. *Self Organising Map of Image Latent Vectors*

The Self-Organising Map in our method was trained on autoencoder latent vectors. A 20x20 neuron toroidal umat is displayed in Figure 5. In this image we superimpose the first RGZ image BMU of each neuron to visualise the dataset topology and latent relationships learned from the latent vectors. This first BMU of each neuron represents the best matching image in the dataset to the learned weights with subsequent BMU's decreasing in similarity. This map was created in 40 seconds after 100 epochs for an average of 0.36 seconds each. This umat clearly shows the morphology distribution of RGZ images where Euclidean distance between the learned weight of each neuron and their neighboring neurons displayed as a heat map. Images placed in high distance regions have a latent feature vector with a high euclidean distance to surrounding neighbors. This is confirmed with high distance regions highlighting outliers within the RGZ image set, particularly in the case of the upper left regions of the umat with sources showing complex morphology.

The morphological clusters in this map are not highly discrete with neurons essentially representing a probability distribution of latent feature vectors. These clustered regions are sub-clustered by orientation, with similarly oriented objects clustered together with gradual transitions between classes. We expect to see this gradual transition between classes of images given these objects do not have entirely discrete classifications. The central low distance region of the umat contains BMU's as compact single sources. BMU's in this central compact region gradually progress in complexity to compact multi point sources toward outer edges. BMU's on the edges of the map in high distance regions show the greatest complexity. These observations are confirmed by examining the RGZ labels and entropy of each BMU.

5.3. *Verification with RGZ labels*

Figure 6 shows a labeled umat of Figure 5 with RGZ labels of the highest matching BMU, entropy of all neuron BMU's and a k-means cluster number with color coding based on the neuron cluster number. Table 2 details the proportions of RGZ labels expressed as the first BMU across the map with min, max and mean entropy. As expected, the population of labels in the dataset is correlated with the proportion of labels matching the first BMU. Entropy follows a similar trend, however more complex objects such as 14, 45 and 46 showing higher entropies due to their intricacy. This is reinforced by their position on the umat, with these complex high entropy sources residing mostly in neurons with a high distance. Complex objects such as 57 and 67 that make up a small portion of the dataset have no neurons on this map as first BMU's due to their complexity. Larger map sizes can be used to better represent more images as first BMU's by providing neurons with enough space to cluster them.

5.4. *K-Means Clustering of Self Organised Map Neuron Weights*

Three k-means clusters, 0, 1 and 2 are labeled in the center of each neuron in Figure 6. These labels are color coded cyan for cluster 0, red for cluster 1 and white for cluster 2. Table 3 details the proportion of neurons classified into each cluster with the min, max and mean entropy. Tables 4 and 5 list the proportion of first best matching units for each label across the clusters. These tables detail the overall class population of the cluster and the overall proportion of each class's dataset population in the cluster. Additionally, the min, max and mean entropy for each clustered label set is listed.

We observe these classes segment the SOM into three groups of simple, compact multi-feature and highly complex images. Cluster 1 is located in the low distance center of the umat, dominated by simple point sources with 79.13% of all neurons as single peak point sources. This cluster contains 78.37% of all class 11 in the dataset with a very low entropy. The remaining 15.05% of neurons are highly compact 12 point sources with two peaks. Two 33 and 22 sources are likely caught in this cluster due to highly compact morphology. The high proportion of these 11 class sources and highly compact sources indicate that cluster 1 is effective in clustering simple objects. Cluster 2 however contains a mix of sparse and highly complex sources located mostly in the high distance regions of the umat. This cluster is highly interesting and contains 70.18% of all class 22 sources, making up 31.75% of the whole cluster (Table 6. The other dominant class in this cluster are sparse and complex class 11 sources in the upper left region with faint companions and noise. The remaining sources in this cluster include all of the low population outlier 14, 16, 44, 45, 46 and 55 classes in addition to 52.94% of the 23 sources and a majority of the 33 sources at 71.43%. These populations indicate that cluster 2 effectively segments outliers and sources of high interest.

We observe that cluster 0 resides between these two clusters on the umat in medium distance regions with BMU's containing classes that are not too complex for cluster 1 but not compact or simple enough for cluster 2. This is confirmed by Table 5, with highly compact multi-peak and multi-component sources in this cluster. This cluster is comprised 50.0% 12 class sources, 20.59% of 22 sources and a series of 11, 23, 33 and 13 sources. Many of the less dominant sources here are complex enough to remain out of cluster 1 but contain a compact enough morphology to keep them from the sparse and complex cluster 2. These observations indicate that cluster 0 segments medium complexity sources in a manner that allows cluster 1 and 2 to remain dominated by simple and complex sources respectively.

The label cross-over between these classes reinforces the point made by Kohonen (1997); SOM's are not designed specifically for hard classification, as SOMs are trained to produce a probability density function of the input vectors. As a result, k-means clustering of SOM weights will instead produce what we see in our results; a semantic map of outliers, regions, and morphologies, distributed by encoded relationships. These results are not unlike those of PINK in (Galvin et al. 2018) where SOM outputs shows a range of morphologies and RGZ label clustering. However this method also took into account additional channels. Most notably, our method operates significantly faster by processing only 900 elements per image as opposed to full FIRST images.

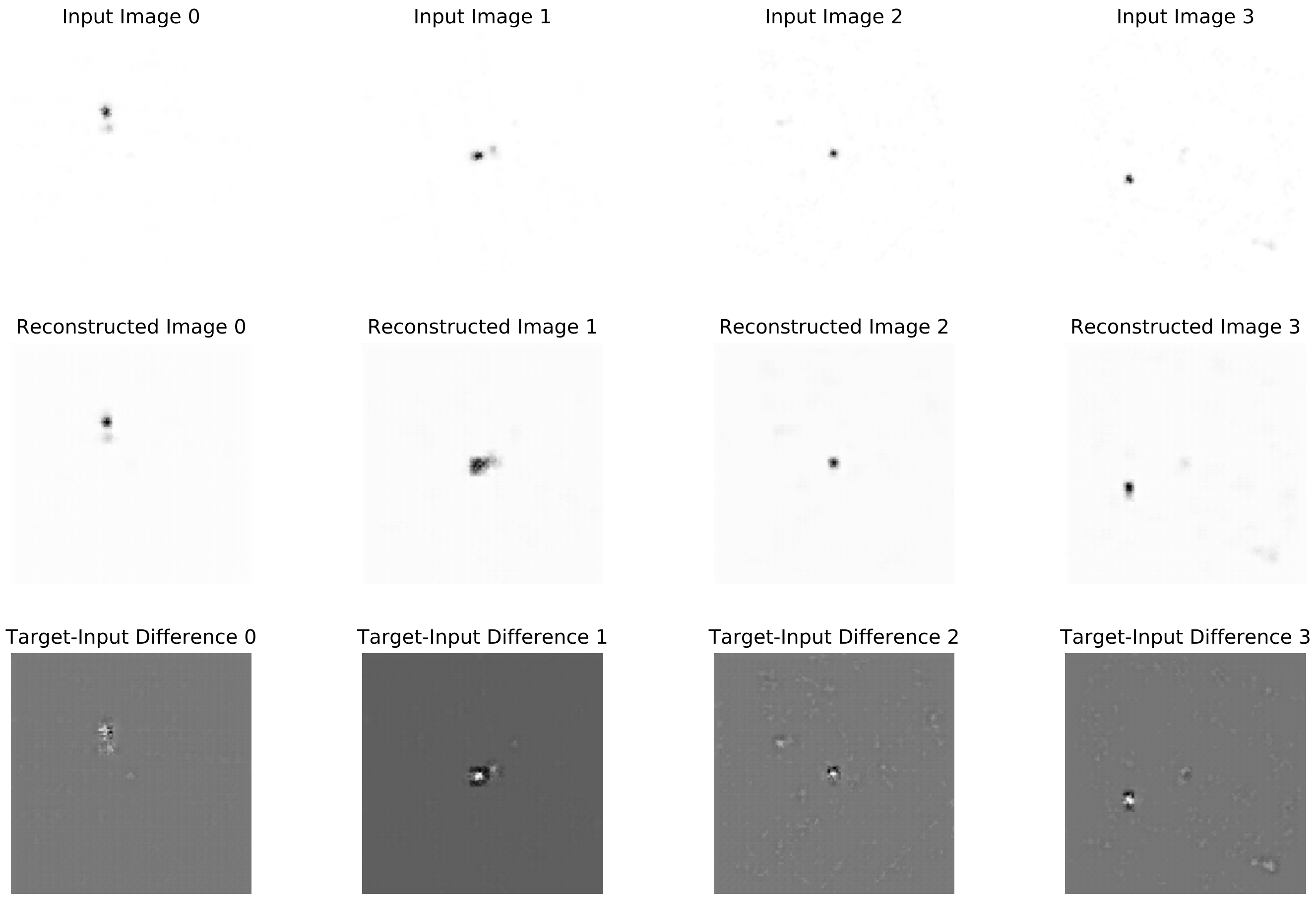

Figure 4. Convolutional Autoencoder prediction of RGZ input images after 20 training epochs. Top row: Original preprocessed input, Middle row: trained autoencoder prediction, Bottom row: Difference image between predicted and original image. .

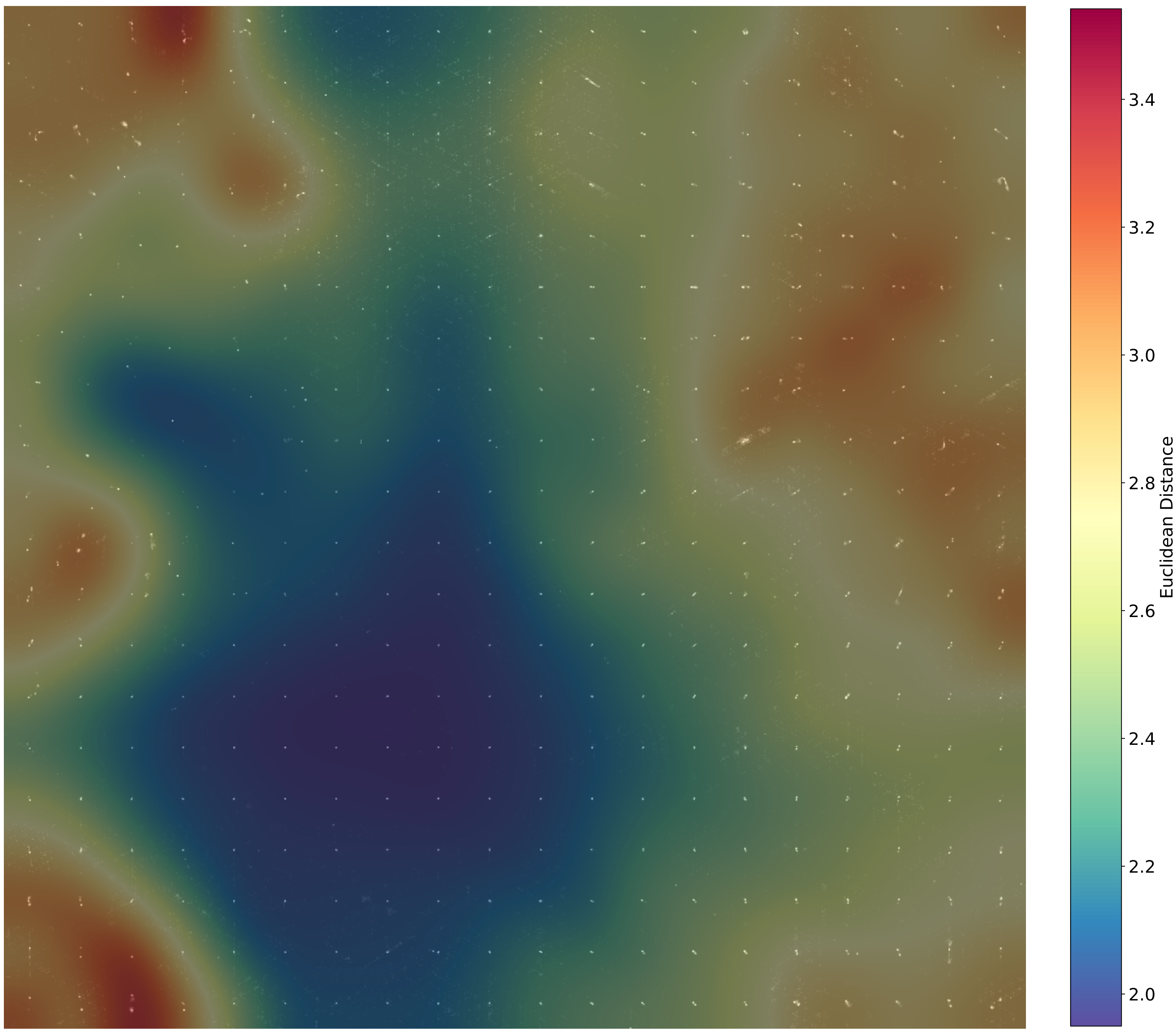

Figure 5. Toroidal umat as SOM output with transparent greyscale BMU FIRST images over a the heat map of the Euclidean distance between neuron weights. This map was created in an average of 0.4 seconds per epoch for 40 seconds after 100 epochs. Sources are softly clustered with an even transition between classes. The central low distance regions contains mostly compact single sources. Progressing to the outer high distance edges are sources with gradually increasing complexity. Outliers and complex sources reside on high distance regions while more common point sources remain in low distance areas.

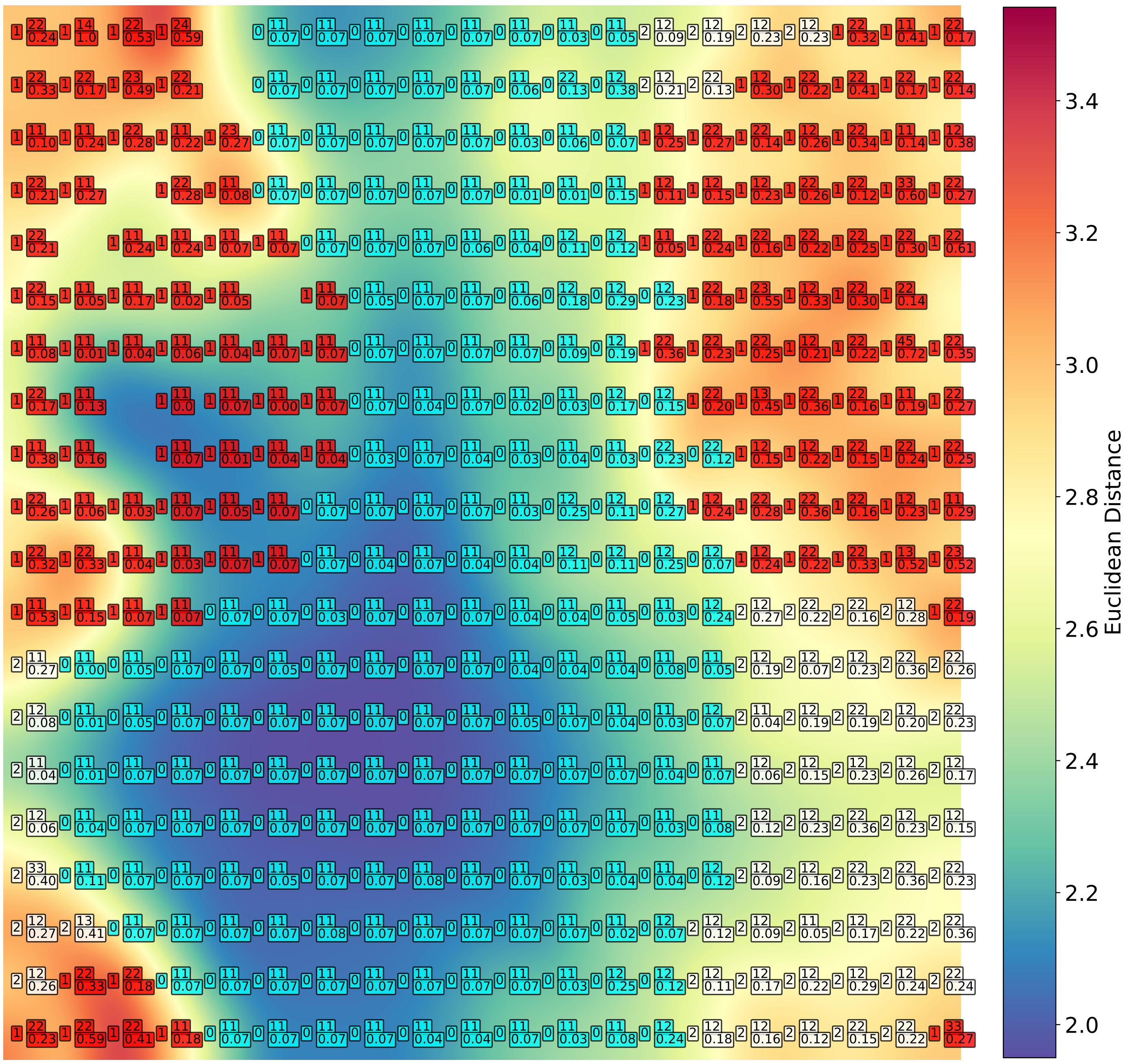

Figure 6. The same SOM as shown in Figure 5 with labels for class (0,1,2) color coded, cyan, red and white respectively. Each neuron is labeled with three numbers, giving the properties of the best matching unit for that neurons. The first (0,1,2) shows the cluster number, the next is a two-digit number representing the labels, and the third is a floating-point number giving the entropy..

Table 2. Proportion of labels expressed as the first BMU across the map with min, max and mean entropy.

Categories	Population	Match Population Over Map	Mean Entropy	Max Entropy	Min Entropy
11	63.78%	52.0%	0.44	0.07	1.52
12	13.86%	18.75%	0.97	0.21	2.27
22	14.3%	14.25%	1.30	0.10	2.57
33	3.28%	5.25%	1.36	0.29	2.29
23	1.94%	4.25%	1.39	0.12	2.25
13	0.78%	0.75%	1.67	1.12	2.40
44	0.76%	0.75%	1.02	0.68	1.60
45	0.2%	0.5%	2.60	2.19	3.00
34	0.42%	0.5%	1.92	1.92	1.92
14	0.08%	0.5%	4.32	3.04	5.61
16	0.02%	0.25%	1.66	1.66	1.66
46	0.08%	0.25%	3.32	3.32	3.32
55	0.14%	0.25%	1.28	1.28	1.28
24	0.16%	0.0%	-	-	-
35	0.08%	0.0%	-	-	-
36	0.02%	0.0%	-	-	-
15	0.02%	0.0%	-	-	-
56	0.04%	0.0%	-	-	-
57	0.02%	0.0%	-	-	-
67	0.02%	0.0%	-	-	-

Table 3. Proportion of k-means clusters across the map with min, max and mean entropy.

K-Means Cluster	Match Population Over Map	Min Entropy	Max Entropy	Mean Entropy
0	17.0%	0.21	2.27	1.15
1	51.5%	0.07	2.18	0.48
2	31.5%	0.10	5.61	1.19

Table 4. K-means Cluster 0: Proportion of BMU's across the cluster, the overall dataset population of each class contained in the cluster with min, max and mean entropy.

Categories	Overall Proportion in Cluster	Proportion of Population in Cluster	Mean Entropy	Min Entropy	Max Entropy
12	50.0%	45.33%	1.14	0.36	2.27
22	20.59%	24.56%	1.32	0.72	1.94
11	11.76%	3.85%	0.60	0.21	1.05
23	11.76%	47.06%	1.33	0.24	2.25
33	2.94%	9.52%	1.68	1.30	2.06
13	1.47%	33.33%	1.12	1.12	1.12

Table 5. K-means Cluster 1: Proportion of BMU's across the cluster, the overall dataset population of each class contained in the cluster with min, max and entropy.

Categories	Overall Proportion in Cluster	Proportion of Population in Cluster	Mean Entropy	Min Entropy	Max Entropy
11	79.13%	78.37%	0.41	0.07	0.75
12	15.05%	41.33%	0.80	0.21	2.18
33	1.94%	19.05%	0.64	0.29	1.28
22	1.46%	5.26%	0.81	0.22	1.27

Table 6. K-means Cluster 2: Proportion of BMU's across the cluster with min, max and mean entropy.

Categories	Overall Proportion in Cluster	Proportion of Population in Cluster	Mean Entropy	Min Entropy	Max Entropy
22	31.75%	70.18%	1.33	0.10	2.57
11	29.37%	17.79%	0.55	0.10	1.52
33	11.9%	71.43%	1.52	0.78	2.29
12	7.94%	13.33%	0.94	0.31	1.31
23	7.14%	52.94%	1.44	0.12	2.01
44	2.38%	100.0%	1.02	0.68	1.60
45	1.59%	100.0%	2.60	2.19	3.00
34	1.59%	100.0%	1.92	1.92	1.92
14	1.59%	100.0%	4.32	3.04	5.61
13	1.59%	66.67%	1.94	1.48	2.40
16	0.79%	100.0%	1.66	1.66	1.66
46	0.79%	100.0%	3.32	3.32	3.32
55	0.79%	100.0%	1.28	1.28	1.28

6. FUTURE WORK

Our plans are to improve each of the three distinct components, the autoencoder, self-organising map and clustering algorithms. By varying the latent vector sizes and structure of the autoencoder we will achieve a better balance of training time to accuracy. Additionally, we will be using a stacked architecture to train latent vectors for multi-channel data. The SOM will be further improved with heat map display of entropy in addition to gathering more performance metrics such as precision and reliability. We will also investigate other clustering algorithms trained in different learning manifolds. We will work to better display the learned weights of each neuron as images similar to other approaches such as PINK (Polsterer et al. 2015) by decoding the learned weights with the Ae.

7. CONCLUSION

We conclude that the coupling of self-organising maps with convolutional autoencoders is an effective method of data exploration and unsupervised clustering of radio-astronomical images. Our approach directly addresses the growing survey processing time and provides a better means to explore large datasets automatically with a total processing time less than 4 minutes for 10,000 images. Our results demonstrate an accurate visualisation of morphology distributions found within the RGZ dataset. Our results show the capabilities of this method in locating outliers as high euclidean distance umat points and in k-means clustering with a distinct class of highly complex sources with low dataset population. By combining clustering with citizen science projects such as Radio Galaxy Zoo, greater efficiencies can be achieved with volunteers inspecting only a small sample of objects from each cluster or being guided by likely morphologies in each cluster. The speed of this method holds implications for use on large future surveys, large-scale instruments such as the Square Kilometre Array and in other big data and anomaly detection applications.

ACKNOWLEDGEMENTS

The authors acknowledge the Radio Galaxy Zoo Project builders and volunteers listed in full at http://rgzauthors.galaxyzoo.org for their contribution to RGZ dataset and labels used in this paper. We also acknowledge the National Radio Astronomy Observatory (NRAO) and the Karl G. Jansky Very Large Array (VLA) as the source of this radio data. Partial support for L.R is provided by the U.S National Science Foundation grant AST17-14205 to the University of Minnesota. H.A benefited from grant DAIP #066/2018 of Universidad de Guanajuato.

REFERENCES

Abadi, M., Barham, P., Chen, J., et al. 2016, in OSDI, Vol. 16, 265–283

Alger, M., Banfield, J., Ong, C., et al. 2018, Monthly Notices of the Royal Astronomical Society

Banfield, J. K., Wong, O., Willett, K. W., et al. 2015, Monthly Notices of the Royal Astronomical Society, 453, 2326

Becker, R. H., White, R. L., & Helfand, D. J. 1995, The Astrophysical Journal, 450, 559

Galvin, T. J., Huynh, M., Norris, R. P., et al. 2018, prep

Graff, P., Feroz, F., Hobson, M. P., & Lasenby, A. 2014, Monthly Notices of the Royal Astronomical Society, 441, 1741

Hotelling, H. 1933, Journal of Educational Psychology, 24, 417

Kohonen, T. 1997, in Neural Networks, 1997., International Conference on, Vol. 1, IEEE, PL1–PL6

Lloyd, S. 1982, IEEE Transactions on Information Theory, 28, 129

Lukic, V., Brüggen, M., Banfield, J., et al. 2018, Monthly Notices of the Royal Astronomical Society, 476, 246

Maaten, L. v. d., & Hinton, G. 2008, Journal of machine learning research, 9, 2579

Norris, R. P. 2017a, Publications of the Astronomical Society of Australia, 34

—. 2017b, Nature Astronomy, 1, 671

Norris, R. P., Hopkins, A. M., Afonso, J., et al. 2011, Publications of the Astronomical Society of Australia, 28, 215

Pedregosa, F., Varoquaux, G., Gramfort, A., et al. 2011, Journal of Machine Learning Research, 12, 2825

Perantonis, S. J., & Lisboa, P. J. 1992, IEEE Transactions on Neural Networks, 3, 241

Polsterer, K. L., Gieseke, F., & Igel, C. 2015, in Astronomical Data Analysis Software an Systems XXIV (ADASS XXIV), Vol. 495, San Francisco: ASP, 81

Sanger, T. D. 1989, Neural networks, 2, 459

Wittek, P., Gao, S. C., Lim, I. S., & Zhao, L. 2013, Journal of Statistical Software, 78, 1 (2017)

Wong, O. I. 2018, prep

Wu, C., Wong, O., Rudnick, L., et al. 2018, Monthly Notices of the Royal Astronomical Society arXiv:1805.12008

Xie, J., Xu, L., & Chen, E. 2012, in Advances in neural information processing systems, 341–349