UNIVERSITY OF GRONINGEN

MASTER'S THESIS

SDR aided Star, Galaxy and QSO Classification

Author:
Marten A. A. LOURENS

Supervisor:
Prof. Dr. Scott C. TRAGER

A thesis submitted in fulfillment of the requirements for the degree of MSc Astronomy: Quantum Universe

in the

Kapteyn Astronomical Institute

university of groningen

faculty of science and engineering

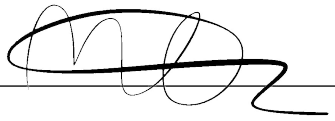
kapteyn astronomical institute

July 18, 2023

Declaration of Authorship

I, Marten A. A. LOURENS, declare that this thesis titled, "SDR aided Star, Galaxy and QSO Classification" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: 

Date: July 18, 2023

UNIVERSITY OF GRONINGEN

Abstract

Faculty of Science and Engineering
Kapteyn Astronomical Institute

MSc Astronomy: Quantum Universe

SDR aided Star, Galaxy and QSO Classification

by Marten A. A. LOURENS

This thesis explores the use of broadband colors to classify stars, galaxies and QSOs. Specifically, the focus is on the application of sharpened dimensionality reduction (SDR)-aided classification, which aims to enhance cluster separation in the projections of high-dimensional data clusters to allow for better classification performance. Based on a qualitative and quantitative analysis of the embeddings produced by SDR, I find that SDR consistently produces projections with a high degree of cluster separation. A number of projection performance metrics are used to evaluate the performance of SDR. These are the trustworthiness, continuity, Jaccard similarity coefficient, Shepard goodness, distance consistency, distribution consistency and neighborhood hit metrics. I also study the oversegmentation feature of SDR projections. These reveal physical information, which allows one to understand the structure of the high-dimensional broadband color data in greater detail. Furthermore, I employ a scalable and out-of-sample (OOS) capable approach, called SDR-NNP, which uses a neural network to reproduce SDR projections. Various classifiers are used to label stars, galaxies and QSOs based on the embeddings yielded by SDR-NNP. A comparison with HDBSCAN-based classification shows similar performance, but SDR-aided classification offers advantages in terms of scalability and interpretability. However, HDBSCAN does not require labeled data for classification. Overall, this thesis demonstrates the potential of SDR-aided classification to provide an accurate and insightful classification of astronomical objects based on their broadband colors.

Acknowledgements

I would first like to thank my thesis supervisor Prof. Dr. Scott C. Trager for is advice, guidance and feedback. Whenever I had a question about my research or writing I could always send him an email or schedule a meeting.

I would also like to thank Dr. Youngjoo Kim for her input and advice regarding the use of sharpened dimensionality reduction in my research.

I am also grateful to Prof. Dr. Alexandru C. Telea for finding time to discuss issues the I had regarding the values of the Jaccard similarity coefficient and trying to find reasons to alleviate these.

I would also like to extend my sincere thanks to Dennis Koopmans for his friendship, help and support throughout my years of study and during my master research.

Furthermore, I would be at remiss not including my parents and sister for their unwavering support throughout my years of study and through the process of researching and writing this thesis.

Lastly, these acknowledgements would not be complete without acknowledging the research output of the various institutions, observatories, consortia and surveys my research is based on:

Funding for SDSS-III has been provided by the Alfred P. Sloan Foundation, the Participating Institutions, the National Science Foundation, and the U.S. Department of Energy Office of Science. The SDSS-III web site is http://www.sdss3.org/.

SDSS-III is managed by the Astrophysical Research Consortium for the Participating Institutions of the SDSS-III Collaboration including the University of Arizona, the Brazilian Participation Group, Brookhaven National Laboratory, Carnegie Mellon University, University of Florida, the French Participation Group, the German Participation Group, Harvard University, the Instituto de Astrofisica de Canarias, the Michigan State/Notre Dame/JINA Participation Group, Johns Hopkins University, Lawrence Berkeley National Laboratory, Max Planck Institute for Astrophysics, Max Planck Institute for Extraterrestrial Physics, New Mexico State University, New York University, Ohio State University, Pennsylvania State University, University of Portsmouth, Princeton University, the Spanish Participation Group, University of Tokyo, University of Utah, Vanderbilt University, University of Virginia, University of Washington, and Yale University.

GAMA is a joint European-Australasian project based around a spectroscopic campaign using the Anglo-Australian Telescope. The GAMA input catalogue is based on data taken from the Sloan Digital Sky Survey and the UKIRT Infrared Deep Sky Survey. Complementary imaging of the GAMA regions is being obtained by a number of independent survey programmes including GALEX MIS, VST KiDS, VISTA VIKING, WISE, Herschel-ATLAS, GMRT and ASKAP providing UV to radio coverage. GAMA is funded by the STFC (UK), the ARC (Australia), the AAO, and the participating institutions. The GAMA website is http://www.gama-survey.org/.

This thesis uses data from the VIMOS Public Extragalactic Redshift Survey (VIPERS). VIPERS has been performed using the ESO Very Large Telescope, under the "Large

Programme" 182.A-0886. The participating institutions and funding agencies are listed at http://vipers.inaf.it.

This research uses data from the VIMOS VLT Deep Survey, obtained from the VVDS database operated by Cesam, Laboratoire d'Astrophysique de Marseille, France.

Funding for PRIMUS is provided by NSF (AST-0607701, AST-0908246, AST-0908442, AST-0908354) and NASA (Spitzer-1356708, 08-ADP08-0019, NNX09AC95G).

This publication makes use of data products from the Wide-field Infrared Survey Explorer, which is a joint project of the University of California, Los Angeles, and the Jet Propulsion Laboratory/California Institute of Technology, funded by the National Aeronautics and Space Administration.

Based on observations collected at the European Southern Observatory under ESO programmes 179.A-2004 and 179.A-2006.

Based on observations obtained with MegaPrime/MegaCam, a joint project of CFHT and CEA/IRFU, at the Canada-France-Hawaii Telescope (CFHT) which is operated by the National Research Council (NRC) of Canada, the Institut National des Science de l'Univers of the Centre National de la Recherche Scientifique (CNRS) of France, and the University of Hawaii. This work is based in part on data products produced at Terapix available at the Canadian Astronomy Data Centre as part of the Canada-France-Hawaii Telescope Legacy Survey, a collaborative project of NRC and CNRS.

Based on data products from observations made with ESO Telescopes at the La Silla Paranal Observatory under programme IDs 177.A-3016, 177.A-3017 and 177.A-3018, and on data products produced by Target/OmegaCEN, INAF-OACN, INAF-OAPD and the KiDS production team, on behalf of the KiDS consortium. OmegaCEN and the KiDS production team acknowledge support by NOVA and NWO-M grants. Members of INAF-OAPD and INAF-OACN also acknowledge the support from the Department of Physics & Astronomy of the University of Padova, and of the Department of Physics of Univ. Federico II (Naples).

This work has made use of data from the European Space Agency (ESA) mission *Gaia* (https://www.cosmos.esa.int/gaia), processed by the *Gaia* Data Processing and Analysis Consortium (DPAC, https://www.cosmos.esa.int/web/gaia/dpac/consortium). Funding for the DPAC has been provided by national institutions, in particular the institutions participating in the *Gaia* Multilateral Agreement.

Funding for the SDSS and SDSS-II has been provided by the Alfred P. Sloan Foundation, the Participating Institutions, the National Science Foundation, the U.S. Department of Energy, the National Aeronautics and Space Administration, the Japanese Monbukagakusho, the Max Planck Society, and the Higher Education Funding Council for England. The SDSS Web Site is http://www.sdss.org/.

The SDSS is managed by the Astrophysical Research Consortium for the Participating Institutions. The Participating Institutions are the American Museum of Natural History, Astrophysical Institute Potsdam, University of Basel, University of Cambridge, Case Western Reserve University, University of Chicago, Drexel University, Fermilab, the Institute for Advanced Study, the Japan Participation Group,

Johns Hopkins University, the Joint Institute for Nuclear Astrophysics, the Kavli Institute for Particle Astrophysics and Cosmology, the Korean Scientist Group, the Chinese Academy of Sciences (LAMOST), Los Alamos National Laboratory, the Max-Planck-Institute for Astronomy (MPIA), the Max-Planck-Institute for Astrophysics (MPA), New Mexico State University, Ohio State University, University of Pittsburgh, University of Portsmouth, Princeton University, the United States Naval Observatory, and the University of Washington.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Source detection and taxonomy of celestial objects are key steps in any astronomical analysis. Examples of this include the classification of stars based on their spectral characteristics in the O to M Harvard spectral classes (Cannon and Pickering, 1912), the categorization of galaxy morphologies by following the Hubble sequence (Hubble, 1926) and the identification of quasi-stellar objects (QSOs), also known as quasars. The first QSOs were discovered in the 1950's as point-like radio sources with no optical counterpart. In 1963, Schmidt (1963) found a faint optical counterpart for the radio source 3C 273 and obtained an optical spectrum with strange emission lines which he identified as hydrogen spectral lines with a redshift of 0.158. This suggested that the source thus far assumed to be a star was receding with an enormous velocity of 47400 km/s. Today these objects are known to be extremely luminous objects inhabiting the nuclei of distant active galaxies with velocities largely driven by cosmological expansion and believed to be powered by the accretion onto a supermassive black hole.

With the advent of multi-wavelength surveys, many sophisticated color selection criteria have been developed to isolate stars, galaxies and active galactic nuclei (AGN) of which QSOs are a subclass. However, the ever-increasing volume of these surveys requires new automated methods to classify such objects, as illustrated by e.g., Dubath et al. (2016). Employing machine-learning methods in the data processing pipelines of these surveys offers a viable solution to a range of problems, which include photometric redshift estimation and the classification of celestial objects.

Broadly there are two different classes of machine-learning methods that can be distinguished by the strategy employed during the training phase. The first class is supervised learning, which uses a training set including input features, e.g., multi-wavelength colors, and output labels, e.g., the class of the astronomical object, to learn the underlying correlations between the input features and output labels. A well-known application of this is the stellarity parameter in Source Extractor, which uses a neural network trained to determine whether a source is a star based on the extend of sources in astronomical images (Bertin and Arnouts, 1996). Other applications of supervised learning methods include the use of decision trees (see e.g., Vasconcellos et al. 2011; Clarke et al. 2020). The second class is unsupervised learning, which searches for data clusters in the feature space and assigns labels based on the clusters points belong to. There exist many different algorithms for unsupervised learning, with varying numbers of hyperparameters, making some harder to tune than others. One of those algorithms is HDBSCAN, Hierarchical Density-Based Spatial Clustering of Applications with Noise. This algorithm has been applied by Logan and Fotopoulou (2020) (LF20) to the CPz dataset (Fotopoulou and Paltani, 2018) (FP18) consisting of "a representative population of spectroscopically observed, stars, galaxies and QSO selected on the basis of their complete photometric coverage in the optical, near infrared, and mid-infrared wavelengths" (Logan and

Fotopoulou, 2020) to perform star, galaxy, QSO classification.

Since I will be comparing the results presented in this work with the classification results obtained by LF20 through the use of HDBSCAN, I now give a brief description of the HDBSCAN algorithm. HDBSCAN (Campello, Moulavi, and Sander, 2013) is an extension of the previous DBSCAN algorithm (Ester et al., 1996), which assigned clusters based on the distance between points and a minimum number of objects in each cluster. Both algorithms define distances between two points as the *mutual reachability distance*. The mutual reachability distance between points a and b is defined as follows:

$$d_{\text{mreach},k}(a,b) = \max\{\text{core}_k(a), \text{core}_k(b), d(a,b)\} \tag{1.1}$$

where $\text{core}_k(\cdot)$ is the core distance defined as the distance to the k^{th} nearest neighbor of a point. This distance metric ensures that sparser points (which are generally noise) are viewed as being spaced further from higher density regions which are the clusters one is interested in. This increases the separation between the data and the noise. Next, both algorithms consider the mutual reachability distance between each set of points to define a weighted graph where each edge is weighted by the mutual reachability distance. By removing edges in this graph when their weights are above a predefined threshold one can start disconnecting the graph into smaller connected components. However, determining the connected components for such a graph is expensive since for n data points there are $n(n-1)$ edges. Therefore, HDBSCAN and DBSCAN employ a more efficient approach by constructing a graph using Prim's algorithm (Prim, 1957). The graph is built one edge at a time, always adding the edge with the lowest weight connecting two vertices that are not yet part of the same graph. The result is a *minimum spanning tree*. Given this minimum spanning tree, the next step converts this tree into a hierarchy of connected components by sorting the edges of the tree in increasing order of mutual reachability distance and iterating through this list of edges, defining a new combined cluster for each edge. The DBSCAN algorithm ends here and requires the user to define a distance scale in this hierarchy at which the set of clusters will be determined. On the other hand, HDBSCAN condenses the minimum spanning tree using a parameter specifying the minimum cluster size. This ensures the tree will only contain branches that have a predefined minimum number of samples. The final step HDBSCAN takes is choosing which clusters to label. This is done by looking at the stability of each cluster in the tree. The cluster stability is defined to be $\sum_{p\in\text{cluster}}(\lambda_p - \lambda_{\text{birth}})$, where $\lambda = 1/\text{distance}$, λ_{birth} corresponds to the λ value when the cluster split off from the original cluster and λ_p corresponds to the λ value when the point p fell out of the cluster. Only clusters with the highest stability that are not children of one another are labeled.[1]

The aim of this thesis is to demonstrate that broadband colors can be used to classify stars, galaxies and QSOs. To perform this classification one can use two-dimensional projections of the high-dimensional set of broadband colors. However, it is often challenging to distinguish high-dimensional data clusters in the 2D projection. To mittigate this issue Kim et al. (2022b) has proposed a method called sharpened dimensionality reduction (SDR), which sharpens the data clusters present in the high-dimensional dataset before projecting it using conventional dimensionality reduction (DR) methods. In this work I show whether SDR can be used to aid the classification of stars, galaxies and QSOs based on the two-dimensional projections

[1] This explanation was adapted from https://hdbscan.readthedocs.io/en/0.8.18/how_hdbscan_works.html.

of high-dimensional sets of broadband colors. In Chapter 2 I present the datasets that I use for this classification process. Chapter 3 introduces various quality metrics that I use to evaluate the performance of various projection methods and classifiers used in this thesis. In Chapter 4 I discuss the process of SDR, explain the various dimensionality reduction (DR) methods which I use in conjunction with the sharpening step and discuss various optimization metrics that I used to optimize the hyperparameters of SDR. SDR-NNP (SDR through Neural Network Projections) is introduced in Chapter 5 as a way to make SDR more scalable and to give it out-of-sample (OOS) capability. Chapter 6 presents various classifiers that I use to perform star, galaxy and QSO classification based on the projections yielded by SDR-NNP and various consolidation methods that can be used to consolidate the results of various classifiers. Each of these chapters contains a results section in which I discuss the results of the methods employed in the respective chapter. In Chapter 7 I zoom in on the results of a sharpened LMDS projection. Specifically, I look at the various subclusters present in the projection and determine whether these convey anything meaningful. Finally, in Chapters 8 and 9 I give a brief summary of what I have discussed and look at some important conclusions. In addition, I compare the results obtained through SDR aided classification with the results obtained through HDBSCAN by LF20.

Chapter 2

The Dataset

The dataset used in this work for SDR aided classification is the CPz catalog, first introduced by FP18 and revised by LF20 to include unsupervised star, galaxy and QSO classification results from HDBSCAN.[1] The original purpose of the CPz catalog was to perform classification-aided photometric redshift (z) estimation, hence the name. The catalog consists of a set of spectroscopically observed sources from different surveys spanning a combined redshift range of $z \in [0-4]$ (see Figure 2a of Fotopoulou and Paltani (2018)). The spectroscopic surveys included in CPz are

- SDSS DR12 (Alam et al., 2015);
- GAMA DR2 (Liske et al., 2015);
- VIPERS DR1 (Garilli et al., 2014);
- VVDS DR2 (Le Fèvre et al., 2013);
- PRIMUS DR1 (Coil et al., 2011; Cool et al., 2013); and
- 6dF DR3 (Jones et al., 2004; Jones et al., 2009).

The combined sample was filtered by FP18 such that it only included sources of highest spectroscopic redshift quality. This was done by only keeping sources with zflag ≥ 3 for the GAMA and 6dF surveys, ZWARNING $= 0$ for SDSS and zflag $= XX3$ or $XX4$ (with $X \in \{0,1,2\}$) for VIPERS and VVDS. Figure 2a in Fotopoulou and Paltani (2018) shows the distributions of spectroscopic redshifts for the various surveys. From this plot one can see that SDSS samples dominate the dataset, especially at higher redshifts.
Subsequently, FP18 matched the remaining spectroscopic sources to photometric detections by various surveys within an angular radius of $1''$. The photometric filters used by each of the photometric surveys cover different parts of the electromagnetic spectrum resulting in a combined wavelength coverage ranging all the way from the mid-IR to the UV. The photometric surveys included in CPz are

- the WISE ALLWISE data release (Wright et al., 2010; Mainzer et al., 2011; Cutri et al., 2013) using the mid-IR $W1$ and $W2$ filters ($W1_{\mathrm{lim,AB}} = 20.3$ (Fotopoulou and Paltani, 2018)[2]);
- the first cycle of ESO near-IR Public VISTA surveys (Arnaboldi et al., 2007) using the near-IR Z, Y, J, H and K_s filters:

[1] The revised catalog is available at the CDS through https://cdsarc.u-strasbg.fr/viz-bin/cat/J/A+A/633/A154.

[2] The Explanatory Supplement to the AllWISE Data Release Products (Cutri et al., 2013) states that the $W1$ band has an average flux limit away from the Galactic plane of 54 μJy at an SNR of 5, which corresponds to an AB magnitude limit of 19.6.

  - VIKING ($J_{\text{lim,AB}} = 22.1$ at 5σ, PI W. Sutherland) and
  - VIDEO ($J_{\text{lim,AB}} = 24.5$ at 5σ, PI M. Jarvis).

- SDSS DR12 (Alam et al., 2015) using the optical u, g, r, i and z filters ($i_{\text{lim,AB}} = 21.3$ (95% completeness limit for point sources)[3]);
- CFHTLS-T0007 Wide using the optical u^*, g', r', i' and z' filters ($i'_{\text{lim,AB}} = 24.8$ (80% completeness limit)) (Hudelot et al., 2013). These filters are similar to those used by SDSS;
- KiDS DR2 (de Jong et al., 2015) using the u, g, r and i filters ($i_{\text{lim,AB}} = 24.2$ (Fotopoulou and Paltani, 2018)[4]) similar to the ones used by SDSS; and
- the GALEX AIS GR6/7 data release (Bianchi, Conti, and Shiao, 2014) using the NUV and FUV filters ($\text{NUV}_{\text{lim,AB}} = 20.5$ (Fotopoulou and Paltani, 2018)[5]).

All optical photometric measurements were corrected for Galactic extinction using Schlegel maps of Galactic absorption (Schlegel, Finkbeiner, and Davis, 1998) and the Cardelli law for the Milky Way (Cardelli, Clayton, and Mathis, 1989). The GALEX filters were omitted from the catalog used by LF20. Therefore, for a good comparison between classification using HDBSCAN and SDR aided classification, they are also not used in this work.

In this work, I apply supervised-learning techniques to train classifiers to label sources based on their location in the projection space provided by the SDR method. Therefore, I require ground-truth class labels to train and validate the performance of these classifiers. The class labels are provided by the CPz dataset used by LF20. The class labels were assigned either automatically, in the case of SDSS spectra, or manually, in the case of VIPERS and VVDS. A breakdown of the different labels is shown in Table 1 of LF20. In 52 percent of cases the spectrum had the class label UNKNOWN. Therefore LF20 chose to label these samples as STAR whenever $z < 0.0015$ and the remaining samples as GAL (i.e., galaxy). Sources labeled as AGN (active galactic nucleus) were omitted from the final dataset. After these changes and removals, LF20 ended up with a dataset comprised of in total 48686 sources of which 7731 were labeled as STAR, 36763 were labeled as GAL and 4192 were labeled as QSO.

The CPz dataset consists of both total and $3''$ aperture magnitudes in the u, g, r, i, z, Y, H, J, K bands and total magnitudes in the $W1$ and $W2$ bands. Each of these magnitudes can be combined into colors by subtracting one from the other resulting in a total number of $\frac{(2\times9+2)(2\times9+2-1)}{2} = 190$ unique colours. This is bound to introduce correlations apart from any correlations inherent to the photometric data itself. Generally, machine learning algorithms are very sensitive to the presence of correlations in the input data. Therefore, it is important to remove these correlations as much as possible and end up with a set of most informative colors. This process is called feature selection. LF20 attempted to achieve this by constructing multiple random forest (RF) classifiers for each binary classification problem (i.e. STAR/non-STAR, GAL/non-GAL and QSO/non-QSO). The resulting RFs were used to obtain a ranked list of colors in order of importance to the classification problem. Additionally, LF20 constructed RF classifiers for the multiple labeling setup in which

[3] See https://www.sdss4.org/dr12/scope/.

[4] The publication for the first and second data releases of KiDS (de Jong et al., 2015) reports a magnitude limit of $i_{\text{lim,AB}} = 23.8$ at 5σ in a $2''$ aperture.

[5] Bianchi, Conti, and Shiao (2014) report a survey depth of 20.8 for the NUV filter.

TABLE 2.1: Lists of attributes used for SDR aided star, galaxy and QSO classification.

Attribute list	**Colors**				
CPz STAR	$K - Y_3$	$K - J_3$	$K - z_3$	$K - H_3$	$J_3 - K_3$
	$Y_3 - K_3$	$J_3 - W1$	$Y_3 - W1$	$J - K$	$H_3 - K_3$
	$H_3 - W1$	$Y - K$	$H - Y_3$	$Y_3 - W2$	$J_3 - W2$
	$i - g_3$	$z_3 - W1$	$z_3 - K_3$	$z - u_3$	$H - J_3$
CPz GAL	$g - J$	$Y - W1$	$J_3 - W1$	$Y_3 - W1$	$J_3 - W2$
	$H_3 - W2$	$Y_3 - W2$	$z_3 - W2$	$K - J_3$	$H_3 - W1$
	$z_3 - W1$	$K - H_3$	$H - W2$	$K - W2$	$W1 - W2$
	$i - W2$	$g - K$	$g - H$	$i - W1$	$r - H$
	$g_3 - i_3$	$r - z_3$	$r - i$	$r_3 - i_3$	$K_3 - W2$
	$r - z$	$r - Y_3$	$H - J_3$	$i - u_3$	
CPz QSO	$J_3 - W1$	$Y_3 - W1$	$J_3 - W2$	$H_3 - W2$	$Y_3 - W2$
	$z_3 - W2$	$K - J_3$	$H_3 - W1$	$z_3 - W1$	$K - H_3$
	$H - W2$	$K - W2$	$W1 - W2$	$g - J$	$i - W2$
	$g - K$	$g - H$	$i - W1$	$r - H$	$g_3 - i_3$
	$r - z_3$	$r - i$	$r_3 - i_3$	$K_3 - W2$	$r - z$
	$r - Y_3$	$H - J_3$	$i - u_3$		
CPz ALL	$K - Y_3$	$K - J_3$	$K - H_3$	$J_3 - W1$	$J_3 - K_3$
	$Y_3 - W1$	$H_3 - W1$	$H_3 - K_3$	$J - K$	$Y_3 - K_3$
CPz SDSS	*All unique combinations of SDSS ugriz* $3''$ *aperture and total magnitudes.*				

all STAR, GAL and QSO labels were assigned at the same time to obtain a similar ranked list of colors. The top ten of this list will be referred to as "CPz ALL" in this work (see Table 2.1). The top ten of the different color lists are given in Table 2 of LF20.

These feature sets still possess significant correlation between different attributes (see Figure 2 of LF20). This correlation arises because RFs only look at individual attributes at each point in the decision tree, making these classifiers insensitive to correlations between different attributes. Therefore, LF20 decided to combine these lists of important colors with those obtained from the A, B and C RF classifiers in FP18, which were significantly less correlated, to generate numerous attribute sets. A grid search was performed over these sets to find an optimal set for each binary classification problem.

I use the same sets of attributes in my work. These attribute sets are referred to as *best_star_colours*, *best_gal_colours* and *best_qso_colours* in Table 3 of LF20. For clarity, I have also listed them here in Table 2.1 and from here onward I refer to them as "CPz STAR", "CPz GAL" and "CPz QSO", respectively. In addition to the CPz STAR, CPz GAL, CPz QSO and CPz ALL color sets, I also test SDR on a set of $\frac{(2\times5)(2\times5-1)}{2} = 45$ unique combinations of SDSS *ugriz* $3''$ aperture and total magnitudes to evaluate the effect of feature selection on the performance of this method. Hereafter this set will be referred to as "CPz SDSS".

Chapter 3

Performance Metrics

In this chapter I discuss several metrics that can be used to evaluate the performance of the projection techniques and classifiers used to perform star, galaxy and QSO classification. The different projection techniques and classifiers I use are discussed further in Chapters 4, 5 and 6. This chapter is comprised of two sections. In the first section I discuss various projection performance metrics and explain what metric I use to decide on the optimum set of parameters for each projection technique. In the second section I present several classification performance metrics which are used to evaluate the performance of different classifiers.

3.1 Projection Performance Metrics

In this thesis I distinguish between three different classes of scalar metrics. The first class of scalar metrics are *local neighborhood metrics*. These metrics compare the neighborhoods of samples in both the feature space and the projection space and quantify whether various local neighborhood relations are preserved in the projection. In this section I discuss the following three local neighborhood metrics:

- trustworthiness;
- continuity; and
- Jaccard similarity coefficient.

The second class of scalar metrics are *distance preservation metrics*. These metrics quantify the preservation of pointwise distances in the projection space with respect to the feature space. In this section I discuss the following distance preservation metrics:

- normalized stress; and
- Shepard goodness.

The last class of scalar metrics are *cluster separation metrics* or *class separation metrics*. These metrics are used to quantify the degree of separation between clusters of different overall class label in the projection space. I discuss the following cluster separation metrics:

- distance consistency;
- distribution consistency; and
- neighborhood hit.

The latter two metrics can also be categorized as *purity metrics* since they both measure the average purity of a class label in the neighborhood sets in the projection space.

Each of these cluster separation metrics arises from the concept of class consistency. According to Sips et al. (2009) a projection is consistent with the **class structure** (i.e., a set of class labels assigned to each sample in a dataset) of a dataset (D) when points in the neighborhood of a point $\mathbf{x}_i \in D$ have the same class label. Sips et al. (2009) used this concept to develop metrics to measure the class consistency of different visualizations of high dimensional datasets: the distance consistency and distribution consistency metrics.

3.1.1 Local Neighborhood Metrics

Trustworthiness

The trustworthiness metric was first introduced by Venna and Kaski (2001) and is defined as follows:

$$M_t(k) = 1 - \frac{2}{Nk(2N - 3k - 1)} \sum_{i=1}^{N} \sum_{j \in \mathcal{U}_i^k} (r(i,j) - k) \tag{3.1}$$

In this definition, N is the number of samples in the dataset D and k is the number of nearest neighbors to consider.[1] The set $\mathcal{U}_i^k$ consists of the k nearest neighbors of sample i in the projection that are *not* among the k nearest neighbors of i in the feature space. The quantity $r(i,j)$ specifies the rank of the point j when data vectors are ordered based on their Euclidean distance to point i in the feature space. All of this implies that the second term in equation (3.1) quantifies the **proportion of false neighbors** and punishes based on how far they are out of the set of nearest neighbors in the feature space (in terms of rank). When the trustworthiness is close to one, the second term in equation (3.1) is close to zero and there are very few false neighbors in the projection. Conversely, when the trustworthiness is close to zero, the second term is close to one and there are many false neighbors in the projection.

Continuity

The continuity metric is closely related to the trustworthiness metric and was also introduced by Venna and Kaski (2001). In fact one could compute the continuity metric by swapping D and $P(D)$ in the trustworthiness metric, i.e., the data samples embedded in the feature space and projection space, respectively. The continuity metric is defined as follows:

$$M_c(k) = 1 - \frac{2}{Nk(2N - 3k - 1)} \sum_{i=1}^{N} \sum_{j \in \mathcal{V}_i^k} (\hat{r}(i,j) - k) \tag{3.2}$$

where N and k are defined as before. The set $\mathcal{V}_i^k$ consists of the k nearest neighbors of sample i in the feature space that are not among the k nearest neighbors after the projection. The quantity $\hat{r}(i,j)$ specifies the rank of the point j when data vectors are ordered based on their Euclidean distance to point i in the projection space. Thus, the second term in equation (3.2) quantifies the **proportion of missing neighbors**

[1] k should always be smaller than $N/2$ for the metric to be properly normalized.

after the projection and penalizes based on how far they are out of the set of nearest neighbors after the projection. When the continuity is close to one the second term in equation (3.2) is close to zero meaning there are few missing neighbors in the projection. Conversely, when the continuity is close to zero the second term in equation (3.2) is close to one indicating that there are many missing neighbors in the projection.

Jaccard similarity coefficient

The Jaccard similarity coefficient metric is based on the *coefficient de communauté florale* (translation: coefficient of the floral community) developed by Paul Jaccard (Jaccard, 1902). It quantifies the **average fraction of overlap** between the k nearest neighbor sets in the feature space and the projection space by averaging over all samples. The functional definition reads as follows:

$$M_J(k) = \frac{1}{N}\sum_{i=1}^{N}\frac{\left|\mathcal{N}_i^k \cap \mathcal{M}_i^k\right|}{\left|\mathcal{N}_i^k \cup \mathcal{M}_i^k\right|} \tag{3.3}$$

In this definition, N is the number of samples in the dataset and k is the number of nearest neighbors to consider. The set $\mathcal{N}_i^k$ consists of the k nearest neighbors of sample i in the feature space. The set $\mathcal{M}_i^k$ consists of the k nearest neighbors of sample i in the projection space. The Jaccard similarity coefficient metric is normalized between zero and one. This follows from the following reasoning. When there is complete overlap between the different nearest neighbor sets for each sample i, the cardinality of the intersection and union of the two sets will be equal and the summation will equate N. This makes the Jaccard similarity coefficient metric equal to one. Conversely, when the two nearest neighbor sets are disjoint for each sample i, the cardinality of the intersection of the two sets will be zero and hence the Jaccard similarity coefficient metric will evaluate to zero as well.

3.1.2 Distance Preservation Metrics

Normalized Stress

The normalized stress metric quantifies the **mismatch** between pointwise distances in the feature space and the projection space (Espadoto et al., 2021). The functional definition reads as follows (Espadoto et al., 2021):

$$M_\sigma = \frac{\sum_{i=1}^{N}\sum_{j=1}^{N}\left(\Delta^n(\mathbf{x}_i,\mathbf{x}_j) - \Delta^m(P(\mathbf{x}_i), P(\mathbf{x}_j))\right)^2}{\sum_{i=1}^{N}\sum_{j=1}^{N}\Delta^n(\mathbf{x}_i,\mathbf{x}_j)^2} \tag{3.4}$$

In this definition, $P(\cdot)$ is a function that projects a n-dimensional feature vector $\mathbf{x}_i$ onto an m-dimensional space and N is the total number of samples in the dataset. The function $\Delta^k(\mathbf{y}_i, \mathbf{y}_j)$ returns the distance between points i and j in a k-dimensional space. The choice of distance metrics can be arbitrary; typically Euclidean distances are used for both the feature space and the projection space.

It is important to note that this metric, whilst being called "normalized", is not strictly limited to the range $[0, 1]$. For dimensionality reduction methods that do not preserve pointwise distances this metric can attain any value between zero and infinity.

Shepard Goodness

The Shepard goodness is defined to be the Spearman rank correlation of the Shepard diagram. Generally, a Shepard diagram is a scatter plot showing two measurements of distances between objects, where one distance is the true distance and the other is the distance in some other representation of the objects (*A Dictionary of Statistics* 2008). In our case that other representation would be the embedding of the high dimensional data in the low dimensional space through a projection technique. Thus our Shepard diagram reads $\left(\left\|\mathbf{x}_i - \mathbf{x}_j\right\|, \left\|P(\mathbf{x}_i) - P(\mathbf{x}_j)\right\|\right)$. The Shepard goodness is then

$$M_S = \frac{\mathrm{Cov}\left(R\left(\left\|\mathbf{x}_i - \mathbf{x}_j\right\|\right), R\left(\left\|P(\mathbf{x}_i) - P(\mathbf{x}_j)\right\|\right)\right)}{\sigma_{R(\left\|\mathbf{x}_i - \mathbf{x}_j\right\|)}\sigma_{R(\left\|P(\mathbf{x}_i) - P(\mathbf{x}_j)\right\|)}} \tag{3.5}$$

with $R(X)$ denoting the ranking of the variables X. The Shepard goodness metric can attain values from -1 to 1.

To better understand the behavior of this metric we identify three unique cases. First we consider the case in which the metric yields a value of -1. This indicates a negative correlation between pointwise distances in the feature and projection space. That is, points that were close together in the feature space were placed far apart in the projection space and, conversely, points that were far apart in the feature space were placed close together in the projection space. In practice, however, the Shepard goodness metric never attains negative values since the objective function of many projection techniques simply does not favor it. Next we consider the case in which the metric is zero. In this case there is no correlation between the distances in the feature space and the projection space. We can imagine this as a Shepard diagram where all points are scattered around randomly without any clear structure. In this case we can say that the projection technique did a bad job at capturing the pointwise distances in the feature space. This makes the projection useless for probing relationships between different samples in the feature space. Finally we consider the case in which the Shepard goodness is 1. This indicates a positive correlation between pointwise distances in the feature and projection space. This is the desirable case. There is an increasing monotonic relation between distances in the feature space and those in the projection space. In other words, the structure of the data in the feature space is well preserved by the projection technique.

3.1.3 Cluster Separation Metrics

Distance Consistency

The formulation of the distance consistency metric was motivated by Sips et al. (2009) through the observation that clustering algorithms, e.g., k-means, partition the space in k convex clusters such that the square distance of all cluster members belonging to the centroid of that cluster is minimal.

The distance consistency metric attempts to measure class consistency by measuring the **centroid distance** for each data point. The centroid distance was defined by Sips et al. (2009) as follows:

Definition 3.1.1. Let $\mathrm{centr}(c_j)$ be the centroid of all data points in the dataset D with class label c_j out of a set of m class labels and let $\mathbf{x} \in D$ be a sample with class label c_i. The distance from $\mathbf{x}$ to $\mathrm{centr}(c_i)$ is given by $d(\mathbf{x}, \mathrm{centr}(c_i))$. The **centroid distance**

of $\mathbf{x}$ is defined as follows:

$$\mathrm{CD}(\mathbf{x}, \mathrm{centr}(c_i)) = \begin{cases} 1 & d(\mathbf{x}, \mathrm{centr}(c_i)) < d(\mathbf{x}, \mathrm{centr}(c_j)) \forall j \in \{0, 1, \ldots, m\} \wedge j \neq i \\ 0 & \text{otherwise} \end{cases} \tag{3.6}$$

Thus the centroid distance is 1 when a data point $\mathbf{x}$ with class label c_i is closest to its own class centroid and 0 otherwise. The **distance consistency** metric leverages this property and is defined using the classification error of class members when using the centroid distance. It was defined by Sips et al. (2009) as follows:

Definition 3.1.2. Let $\mathrm{centr}(c_j)$ be the centroid of all data points in the dataset D with class label c_j and let $\mathrm{clabel}(\mathbf{x})$ be the class label of a datapoint $\mathbf{x} \in D$. Then the distance consistency can be computed as follows:

$$M_{\mathrm{DC}} = 1 - \frac{|\{\mathbf{x} \in D : \mathrm{CD}(\mathbf{x}, \mathrm{centr}(\mathrm{clabel}(\mathbf{x}))) \neq 1\}|}{N} \tag{3.7}$$

where $|\cdot|$ indicates cardinality.

Note that the distance consistency metric is 1 when all data points are correctly classified and 0 when all are misclassified.

Distribution Consistency

An alternative way of measuring class consistency is by using entropy as a measure of class purity. For each data point $\mathbf{x} \in D$ one can compute the entropy in the distribution of m class labels among its k nearest neighbors. We define $p_{c_i}(\mathbf{x})$ to be the number of data points of class c_i in the nearest neighbor set of point $\mathbf{x}$. The Shannon entropy for each data point then reads as follows:

$$H(\mathbf{x}, k) = -\sum_{i=0}^{m} \frac{p_{c_i}}{\sum_{i=0}^{m} p_{c_i}} \log_2 \left(\frac{p_{c_i}}{\sum_{i=0}^{m} p_{c_i}} \right) \tag{3.8}$$

The entropy is zero when all its neighbors have the same class label and will be $\log_2(m)$, when all m classes are mixed equally in its neighborhood. The distribution consistency is then defined by summing over all data points and normalizing such that all values will be between zero and one:

$$\begin{aligned} M_{\mathrm{DC}}(k) &= 1 - \frac{1}{N \log_2(m)} \sum_{\mathbf{x} \in D} H(\mathbf{x}, k) \\ &= 1 + \frac{1}{N \log_2(m)} \sum_{\mathbf{x} \in D} \sum_{i=0}^{m} \frac{p_{c_i}}{\sum_{i=0}^{m} p_{c_i}} \log_2 \left(\frac{p_{c_i}}{\sum_{i=0}^{m} p_{c_i}} \right) \end{aligned} \tag{3.9}$$

Note that the distribution consistency is zero when all m classes are mixed equally in the neighborhood of each point and that it will be one when all m classes are well separated in the projection.

This definition of distribution consistency is different from that introduced by Sips et al. (2009) in that it computes the entropy using the distribution of class labels in the k-nearest-neighbor set of each sample. Instead the definition of distribution consistency introduced by Sips et al. (2009) uses a kernel of width σ used to integrate over the projection space by selecting different regions over which to compute the entropy. The reason I choose to deviate from this approach is that this method is

TABLE 3.1: Confusion matrix illustration.

		Predicted	
		Positive	Negative
True	Positive	TP (True Positive)	FN (False Negative)
	Negative	FP (False Positive)	TN (True Negative)

highly sensitive to the choice of kernel width, which determines the typical size of regions over which the different classes should not be mixed. In contrast, the k-nearest-neighbor approach puts a constraint on the minimum number of samples that are part of a region with uniform class label and is much more adaptive, as it works well in both sparse and dense regions.

Neighborhood Hit

The neighborhood hit metric is defined as the average over all fractions of k nearest neighbors for each point i that have the same class label as i. Formally it is defined as follows:

$$M_{NH}(k) = \frac{1}{kN} \sum_{i=1}^{N} \left| \left\{ j \in \mathcal{N}_i^k : c_j = c_i \right\} \right| \tag{3.10}$$

In this equation $|\cdot|$ denotes the cardinality of a set, $\mathcal{N}_i^k$ is the set of nearest neighbors of point i in the projection space and c_i denotes the class label of a point i. A metric value of one implies that data with different labels are well separated whereas a value of zero means that labels are not properly separated in the projection.

It is important to note that all of these cluster separation metrics are only relevant when data is labeled and labels are assigned accurately in line with data clusters present in the high-dimensional space.

3.2 Classification Performance Metrics

In order to define classification performance metrics we need to make a distinction between binary classifiers and multi-label classifiers. Binary classifiers are those that only distinguish between two populations. For example, star and non-star or positive and negative. In contrast, multi-label classifiers can distinguish between multiple populations. For example, $\{\text{class1}, \text{class2}, \ldots, \text{class } m\}$ or in our case, {star, galaxy, QSO}. For either of these two classes of classifiers one can construct a confusion matrix from which our performance metrics can be derived in a straightforward manner. A confusion matrix is a matrix representing the counts of predicted versus actual values (see Table 3.1 for an example of a confusion matrix for a binary classification problem).

In the next subsections I give the formal definitions of the classification performance metrics that are used in this work. Naturally, there are many more classification performance metrics that can be derived directly from the confusion matrix. However, I believe that the metrics presented here are the most relevant.

3.2.1 Accuracy

The accuracy of a classifier can be derived by dividing the number of correct predictions by the total number of predictions. For a binary classifier we have the following formula:

$$M_{\text{accuracy}} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{3.11}$$

where TP, TN, FP and FN indicate the number of true positives, true negatives, false positives and false negatives, respectively. For a multi-class classifier, the accuracy can be used to give an idea of the average performance over all classes.

3.2.2 Precision

The precision is the fraction of correct positive predictions. This metric is also known as the positive predictive power or, in astronomy, as the purity. For a binary classifier its formal definition reads:

$$M_{\text{precision}} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{3.12}$$

For a multi-class classifier we can have multiple values for the precision depending on which class is referred to as the "positive" class and which classes are referred to as the "negative" classes. The precision gives one an idea of the power of the classifier to provide correct predictions for a single class.

3.2.3 Recall

The recall, also known as sensitivity, hit rate and true positive rate, is defined as the fraction of truly positive predictions. It is defined as follows:

$$M_{\text{recall}} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3.13}$$

Similarly to the precision a multi-class classifier can have multiple values for recall depending on which class is referred to as the "positive" class. The term most often used for recall in astronomy is *completeness* which is perhaps more informative on its definition. The recall represents the fraction of samples of a specific true class that are not wrongly classified, i.e. how "complete" the set of correctly classified samples is.

3.2.4 F1 score

Usually, one may be more interested in an equal trade-off between precision and recall. A high precision is important to be confident about the reliability of the classifier, however, this should not come at the cost of losing many of the samples to other classes due to misclassification. In order to evaluate the trade-off between precision and recall one can use the F1 score, which is the harmonic mean of precision and recall:

$$M_{\text{F1}} = 2 \cdot \frac{M_{\text{precision}} \cdot M_{\text{recall}}}{M_{\text{precision}} + M_{\text{recall}}} \tag{3.14}$$

Note that, since multi-class classifiers can have multiple values for precision and recall we can also have multiple values for the F1 score depending on which class is referred to as the "positive" class in the definition of the precision and recall.

Chapter 4

Sharpened Dimensionality Reduction

In this chapter I discuss the process of sharpened dimensionality reduction (SDR), which was proposed by Kim et al. (2022b) to tackle the problem of distinguishing high-dimensional data clusters in a 2D projection.[1] The method was shown to yield better cluster separation in the projection than DR methods with no sharpening and scales computationally well with large high-dimensional datasets (Kim et al., 2022b).

In the first section I explain the local gradient clustering (LGC) step proposed by Kim et al. (2022b) to sharpen the data in the high dimensional feature space. The second section gives an overview of the dimensionality reduction (DR) algorithms that I have combined with the LGC step to perform sharpened dimensionality reduction. In the final section of this chapter I analyze the results of applying different SDR algorithms on the different CPz datasets presented in Chapter 2 and discuss how I tuned the hyperparameters of the various algorithms.

4.1 Local Gradient Clustering

The method proposed by Kim et al. (2022b) consists of two separate steps: local gradient clustering (LGC) and dimensionality reduction (DR) using any dimensionality reduction technique of choice. The goal of the LGC step is to precondition the high-dimensional dataset allowing the DR method to provide better cluster separation. Mean shift-based methods are ideal for this LGC step, since they allow to enhance overdensities in the high dimensional data space. In other work, mean shift-based methods have been used, usually in combination with DR methods, to cluster data by determining the cluster modes present in the data.

A recent application of mean-shift methods is gravitational clustering (GC), which was proposed by Binder, Muma, and Zoubir (2018) to adaptively estimate a time-varying number of clusters based on a set of feature vectors by modeling each vector to exert a gravitational force on so-called "mobile mass units" that are injected at each time interval. Over time this force will cause these mobile mass units to gravitate towards regions of high density providing estimates of the cluster modes present in the data.

Other methods do not model the dataset as a gravitational system. Instead, these methods focus on estimating the sample density by constructing a kernel density estimate and computing its gradient. The stationary points of this gradient where the curvature is negative then constitute to a cluster mode (Cheng, 1995).

[1] In Kim et al. (2022b) the method is called "High-Dimensional Sharpened Dimensionality Reduction" (HD-SDR).

The local gradient clustering technique proposed by Kim et al. (2022b) is inspired by the latter approach.[2] The sample density is estimated by constructing a kernel density estimate (KDE):

$$\hat{\rho}(\mathbf{x}_i) = \sum_{\mathbf{x}_j \in \mathcal{N}_i^k} \mathcal{K}\left(\frac{\|\mathbf{x}_i - \mathbf{x}_j\|}{h_i}\right). \tag{4.1}$$

The kernel ($\mathcal{K}(\cdot)$) used is a parabolic kernel called the Epanechnikov kernel, which according to Epanechnikov (1969) is the best choice of kernel in a mean-squared error (MSE) sense. The parameter h_i specifies the bandwidth of the kernel at position $\mathbf{x}_i$ and is defined to be the distance to the k^{th} nearest neighbor of point $\mathbf{x}_i$. This ensures the KDE is insensitive to the cluster scale. The set $\mathcal{N}_i^k$ is the set of k nearest neighbors around the point $\mathbf{x}_i$. The advantage of using only the k nearest neighbors to estimate the KDE is that it accelerates the density estimation from a computational point of view.[3] After estimating the local density $\hat{\rho}$ for $\mathbf{x}_i$, the sample can be shifted along the density gradient in the direction of higher density using the update rule

$$\mathbf{x}_i' = \mathbf{x}_i + \alpha \frac{\nabla \hat{\rho}(\mathbf{x}_i)}{\max\left(\|\nabla \hat{\rho}(\mathbf{x}_i)\|, \epsilon\right)}, \tag{4.2}$$

where $\alpha \geq 0$ is the learning rate and $\epsilon = 10^{-5}$ is a regularization parameter for regions with low density gradient such that points don't shoot off to infinity. This update rule is applied to all samples separately for a total of T iterations. After each iteration the KDE of equation (4.1) is recomputed.

As can be seen, the sharpening step has three free parameters that can be altered to improve the performance of the SDR algorithm. These are the learning rate (α), the number of nearest neighbors (k) used to make a local density estimate and the maximum number of iterations (T) for which we integrate. Following a qualitative analysis by Kim et al. (2022b), each of these parameters has a different effect on the cluster separation in the projection yielded by the DR step of the SDR algorithm:

- the learning rate controls the *size* of the shifts taken by the LGC technique. This can affect the degree of segmentation. Setting α to a value that is too small can result in oversegmentation. Conversely, when α is too large samples can overshoot the cluster mode resulting in them to become more scattered. In the most extreme cases this can result in a lesser degree of cluster separation instead of more (Kim et al., 2022b);

- the number of nearest neighbors controls the *locality* of the shifts taken by the LGC algorithm. Similarly to α, making k too small can lead to oversegmentation. On the other hand, making k too large can significantly increase the amount of time it takes for the algorithm to come up with a density estimate. Everything considered, Kim et al. (2022b) determined that the parameter k is of lesser importance than α since k may not significantly affect segmentation without an appropriate value of α (Kim et al., 2022b); and

- the maximum number of iterations (T), controls the *degree of cluster separation*. When T is too small, samples will only shift by a few steps resulting in only

[2] The code written by Kim et al. (2022b) in C++ is available on GitHub by following this link: https://youngjookim.github.io/sdr/about/.

[3] The k-nearest-neighbor sets are computed using the C++ header-only library nanoflann (see https://jlblancoc.github.io/nanoflann/index.html) which implements a $\mathcal{O}(Nn\log(n))$ algorithm based on KD-trees.

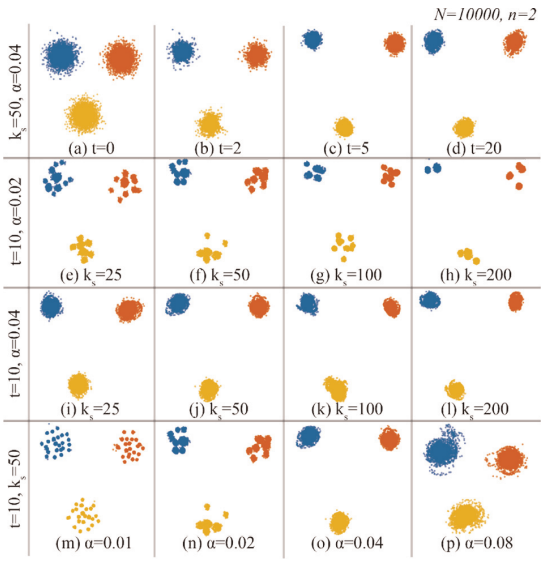


FIGURE 4.1: Figure showing the effects of varying the values of the different parameters used by LGC. The data consists of three clusters, color coded based on their ground-truth class label, with samples drawn from a two-dimensional Gaussian distribution.(Image credit: Kim et al. 2022b)

a small difference from the original data. Kim et al. (2022b) found that setting $T = 10$ is enough to achieve decent cluster separation for both Gaussian as well as non-Gaussian synthetic data whilst being computationally viable (increasing T adds to the computation time). Kim et al. (2022b) also found that varying T by a factor of two does not significantly change the outcome.

These results are summarized in Figure 4.1, taken from Kim et al. (2022b).

4.2 Dimensionality Reduction Methods

In this work I combine the sharpening step introduced in the previous section with various dimensionality reduction techniques. The majority of the DR techniques I have tested are part of Tapkee (Lisitsyn, Widmer, and Garcia, 2013), which supports many common dimensionality-reduction techniques. The DR techniques from this library that I have tested are Landmark Multidimensional Scaling (LMDS) (De Silva and Tenenbaum, 2004; Cox and Cox, 2008), Neighborhood Preserving Embedding (NPE) (He et al., 2005), *t*-Distributed Stochastic Neighbor Embedding (*t*-SNE) (Maaten and Hinton, 2008), Locally Linear Embedding (LLE) (Roweis and Saul, 2000), Laplacian Eigenmaps (Belkin and Niyogi, 2001), Linear Local Tangent Space Alignment (Linear LTSA) (Zhang et al., 2007), Hessian Locally Linear Embedding (HLLE) (Donoho and Grimes, 2003), Manifold Sculpting (Gashler, Ventura, and Martinez, 2007) and Landmark Isomap (Silva and Tenenbaum, 2002). Additionally, I have tested LLE, HLLE and Local Tangent Space Alignment (LTSA) (Zhang and Zha, 2004) of scikit-learn (Pedregosa et al., 2011) and Uniform Manifold Approximation and Projection (UMAP) (McInnes, Healy, and Melville, 2020).

These projection techniques can be categorized according to eight traits that non-specialist users can consider when selecting a particular technique. These are listed in Espadoto et al. (2021). Five of these traits are relevant to consider in this work.

- Linearity: A projection can be either linear or nonlinear. Linear projection techniques are easier to understand but do not perform well for data distributions that follow a high-dimensional, nonlinear manifold. In such cases nonlinear projections perform better.
- Neighborhood: Projection techniques either preserve either local or global neighborhoods. Local neighborhood methods only preserve inter-point distances of points that are in each other's local neighborhood, which can result in better cluster separation. Contrarily, global neighborhood methods try to preserve all inter-point distances which leads to a more accurate representation of the higher-dimensional data. However, this can lead to a lower degree of cluster separation.
- Computational complexity: This is the algorithmic complexity of a projection technique in big-O notation, as function of the number of samples (N) and the number of dimensions (n). Algorithms with lower computational complexity take less time and are therefore better suited for e.g., grid searches or interactive visual exploration.
- Out-of-sample (OOS) capability: The ability of a projection technique to extrapolate to new data based on previous training. This is particularly useful when one wants to use a projection technique as part of a classification pipeline but does not want to go through the costly training step every time new samples need to be classified.
- Determinism: Espadoto et al. (2021) defined this trait as the ability of a projection technique "to reproduce its results regardless of random seed initialization". This implies that many projection techniques will be regarded as non-deterministic even though many of those do provide the ability to set a random seed. Whenever a random seed can be set I have used 42 to make my results reproducible.

I have summarized the five traits of each of the dimensionality reduction techniques I tested in Table 4.1. Whilst I have tried both Tapkee as well as scikit-learn implementations of LLE and HLLE their time complexities are roughly the same with the exception of the formulation of $C(n)$ since Tapkee uses a cover tree and scikit-learn uses a ball tree to do k-nearest-neighbor searches. It is also worth noting that no valid time complexity is known for manifold sculpting since it is an iterative algorithm (Gashler, Ventura, and Martinez, 2007). We can already tell from the time complexities that for datasets with a large number of samples DR methods like, e.g., LMDS, t-SNE and UMAP should scale particularly well.

4.3 Results

In the previous section I presented several DR methods. We have also seen from Table 4.1 that each of these projection techniques have different traits. Espadoto et al. (2021) has shown in his quantitative survey of dimension reduction techniques that depending on these traits and the traits of a given dataset some DR techniques may work better than others. Therefore, mirroring the work of Espadoto et al. (2021), I choose to rank the projection techniques presented in the previous section based on a metric that captures the quality of each projection. The quality metric I use is

TABLE 4.1: Relevant traits for the various projection methods tested in this work. In the table N is the total number of samples in the dataset, n is the input dimensionality, m is the output dimensionality, λ is the number of landmark points, k is the number of nearest neighbors and C is a cost value which is sometimes associated with n.

DR method	Linearity	Neighborhood	Computational Complexity	OOS capability	Determinism
LMDS	linear	global	$\mathcal{O}(C\lambda N + m\lambda N + \lambda^3)$	no	no
Landmark Isomap	nonlinear	global	$\mathcal{O}(C(n)\lambda \log(\lambda) + \lambda^2(k + \log(\lambda)) + C\lambda N + m\lambda N + \lambda^3)$	no	no
NPE	linear	local	$\mathcal{O}(C(n)N \log(N) + nNk^3 + mN^2)$	no	yes
LLE	nonlinear	local	$\mathcal{O}(C(n)N \log(N) + nNk^3 + mN^2)$	no	yes
Hessian LLE	nonlinear	local	$\mathcal{O}(C(n)N \log(N) + nNk^3 + m^6N + mN^2)$	no	yes
Laplacian Eigenmaps	nonlinear	local	$\mathcal{O}(C(n)N \log(N) + nNk^3 + mN^2)$	no	yes
Linear LTSA	linear	local	$\mathcal{O}(C(n)N \log(N) + nNk^3 + mk^2 + mN^2)$	no	yes
LTSA	nonlinear	local	$\mathcal{O}(C(n)N \log(N) + nNk^3 + mk^2 + mN^2)$	no	yes
Manifold Sculpting	nonlinear	local	-	no	no
t-SNE	nonlinear	local	$\mathcal{O}(mN \log(N))$	no	no
UMAP	nonlinear	local	$\mathcal{O}(N^{1.14} + kN)$	yes	no

an aggregate of the projection performance metrics discussed in Chapter 2:

$$M_{\text{total}}(k) = \frac{1}{4}\left(M_t(k) + M_c(k) + M_{NH}(k) + M_S\right) \tag{4.3}$$

This metric captures both the ability of the projection technique to provide an accurate representation of the high dimensional data and the degree of cluster separation in the projection. Whilst Espadoto et al. (2021) also used the normalized stress metric in his quantitative DR technique survey, I omit it. This is because the normalized stress is unbounded from above for DR techniques that globally scale pointwise distances (see Section 3.1.2). Examples of such methods are UMAP and t-SNE.

The results of optimizing the hyperparameters of each DR method applied to the CPz STAR dataset with respect to this metric and their respective parameter grids are summarized in Tables 4.2 and 4.3, respectively. I note that these results are obtained by selecting a subset of 10000 samples from the 48686 data points in the CPz STAR dataset. Firstly, using Table 4.2 we rule out the use of Linear LTSA, HLLE and LTSA. Secondly, the Manifold Sculpting and Landmark Isomap methods did not finish in a reasonable amount of time. Hence we do not have any results for those. Finally, of the remaining methods LMDS, UMAP, NPE and t-SNE run fastest. Therefore, I only focus on those for the remainder of this thesis.

From the results in Table 4.2 one can derive a number of additional conclusions. Firstly, t-SNE performs best in terms of trustworthiness followed by UMAP and LMDS. This means that t-SNE has the lowest fraction of "false neighbors" in the projection. Comparing with Table 4.1 we can say that nonlinear projection techniques (i.e., t-SNE and UMAP) perform better than linear projection techniques (i.e., LMDS and NPE) in terms of trustworthiness. Secondly, in terms of continuity UMAP scores best closely followed by LMDS and t-SNE. From the continuity values one can tell that the projections yielded by each of these methods exhibit a small fraction of "missing neighbors". It is no surprise that LMDS, UMAP and t-SNE perform well in terms of trustworthiness and continuity since these methods should preserve neighborhood relations. Thirdly, the Shepard goodness shows us how well pointwise distances are preserved while allowing these distances to be monotonically scaled. From Table 4.2 one can tell that LMDS best preserves pointwise distances followed by NPE and UMAP. This suggests that NPE and UMAP, even though they are local neighborhood methods (i.e., they tend to only preserve local distances), preserve pointwise distances globally for the CPz STAR dataset.

TABLE 4.2: DR techniques optimized and ranked with respect to the total metric (4.3) using 10000 samples from the CPz STAR dataset. The metrics were computed using $k = 500$.

DR method	Backend	Trustworthiness	Continuity	Neighborhood Hit	Shepard Goodness	Total	Best Parameter Set
LMDS	Tapkee	0.953557	0.992009	0.958214	0.97754	0.97033	landmark_ratio: 0.035
UMAP	umap-learn	0.965203	0.993656	0.967671	0.922242	0.962193	metric: 'euclidean', min_dist: 0.5, num_neighbors: 280, umap_init: 'spectral'
NPE	Tapkee	0.941537	0.98492	0.956514	0.937994	0.955241	num_neighbors: 80
t-SNE	Tapkee	0.988591	0.990558	0.972014	0.862936	0.953525	sne_perplexity: 160, sne_theta: 0.5
LLE	Tapkee	0.956839	0.989021	0.954571	0.874767	0.9438	num_neighbors: 60
LLE	scikit-learn	0.956822	0.989021	0.954529	0.874767	0.943785	num_neighbors: 60
Laplacian Eigenmaps	Tapkee	0.953845	0.978639	0.9697	0.776664	0.919712	gaussian_kernel_width: 1.0, num_neighbors: 10
Linear LTSA	Tapkee	0.646507	0.858524	0.705	0.189983	0.600004	num_neighbors: 260
HLLE	Tapkee	0.598806	0.683654	0.639443	0.285113	0.551754	num_neighbors: 60
LTSA	scikit-learn	0.486421	0.50664	0.4689	0.032393	0.373588	num_neighbors: 20
HLLE	scikit-learn	0.486334	0.506643	0.468671	0.032353	0.3735	num_neighbors: 20
Manifold Sculpting	Tapkee	-	-	-	-	-	
Landmark Isomap	Tapkee	-	-	-	-	-	

TABLE 4.3: Parameter grids used for optimizing the various DR methods listed in Table 4.2.

DR method	Parameter Grid
LMDS	landmark_ratio: [0.005, 0.01, 0.015, 0.02, 0.025, 0.03, 0.035, 0.04, 0.045, 0.05, 0.055, 0.06, 0.065, 0.07, 0.075, 0.08, 0.085, 0.09, 0.095, 0.1]
UMAP	num_neighbors: [20,40,60,80,100,120,140,160,180,200,220,240,260,280,300], min_dist: [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0], metric: ["euclidean"], umap_init: ["spectral"]
NPE	num_neighbors: [20,40,60,80,100,120,140,160,180,200]
t-SNE	sne_perplexity: [20,40,60,80,100,120,140,160,180,200], sne_theta: [0.5]
LLE	num_neighbors: [20,40,60,80,100,120,140,160,180,200]
Laplacian Eigenmaps	num_neighbors: [5,10,15,20,25,30,35,40], gaussian_kernel_width: [1.0]
Linear LTSA	num_neighbors: [20,40,60,80,100,120,140,160,180,200,220,240,260]
HLLE	num_neighbors: [20,40,60,80,100,120,140,160,180,200]
LTSA	num_neighbors: [20,40,60,80,100,120,140,160,180,200,220,240,260]
Landmark Isomap	num_neighbors: [5,10,20,40,80], landmark_ratio: [0.005, 0.01, 0.015, 0.02, 0.025, 0.03, 0.035, 0.04, 0.045, 0.05, 0.055, 0.06, 0.065, 0.07, 0.075, 0.08, 0.085, 0.09, 0.095, 0.1]

We now examine the neighborhood hit metric. This metric measures the class consistency of the projection, i.e., the degree of separation of data clusters that are uniform in class label. In terms of neighborhood hit *t*-SNE performs best followed by UMAP, LMDS and NPE. LMDS and NPE have similar values. From Table 4.2 one can see that the LMDS, UMAP, NPE and *t*-SNE projections all have rather high values for neighborhood hit. This suggests that given the projection one should be able to label the dataset with high accuracy. This begs the question whether we can improve class consistency even further by instead optimizing LMDS, *t*-SNE, NPE and UMAP with respect to one of the Cluster Separation Metrics. To achieve this I use the distribution consistency metric (Eq. (3.9)). By experimenting with the different class consistency metrics on different synthetic datasets, it becomes evident that some metrics behave better than others. When different data clusters, uniform in class label, are completely separated, all metrics converge to 1 (see Figures 4.2a and 4.2b). However, when clusters completely overlap, the degree of cluster separation should be zero. Examining Figure 4.2c we see that this is far from true for the neighborhood hit metric. In fact, the neighborhood hit metric converges to a value of 0.5 and grossly *overestimates* the degree of cluster separation. In contrast, the distance consistency and distribution consistency metrics seem to do a much better job at measuring the degree of cluster separation. Now we examine the case where one of the clusters is located in the concave region of another non-convex cluster (Figure 4.2d). In this case, the centroids of the different clusters, marked by red crosses, start to overlap. However, the clusters themselves do not overlap. This results in the distance consistency metric *underestimating* the degree of cluster separation. We also see that the neighborhood hit and distribution consistency metrics are much more robust in this case. Overall, we conclude from this small experiment that the distribution consistency metric seems to work best in terms of versatility and is therefore my metric of choice to evaluate the degree of cluster separation.

The projection performance metric results of the CPz STAR dataset obtained when optimizing the hyperparameters of LMDS, UMAP, *t*-SNE and NPE with respect to the distribution consistency metric are presented in Table 4.4. The parameter grids used for the optimization are listed in Table 4.5. The same parameter grids have also been used to optimize the DR methods for the other datasets. I plot the 2D projections yielded by each of these DR methods in Figures 4.3, 4.4, 4.5 and 4.6, respectively. Note that these results were computed using a subset of 10000 samples randomly selected from the full *projected* dataset consisting of 48686 data points. The reason behind this is that computing a distance matrix for the full dataset would require too much memory resources.

Tables 4.6 and 4.7 show the same results for the CPz GAL and CPz QSO datasets, respectively. The results for the CPz GAL and CPz QSO datasets are presented in Appendix A. Visual inspection of results for the CPz ALL and CPz SDSS datasets reveal that their projections show very little cluster separation (see Figures A.9 and A.10 in Appendix A). Especially for the linear projection techniques (i.e., LMDS and NPE), I observe significant mixing between the different classes. Therefore, I omit these datasets from this work as their projections are clearly not suitable for classification.

Comparing the metric results in Tables 4.2 and 4.4, one can make the following observations. Firstly, we observe only marginal differences between the trustworthiness, continuity and neighborhood hit metrics. This suggests that the average fractions of false, missing and same class neighbors has not changed much. Furthermore, these differences could mostly be due to the way the metrics were computed. To obtain the results in Table 4.2 I project a random subset of 10000 samples from the

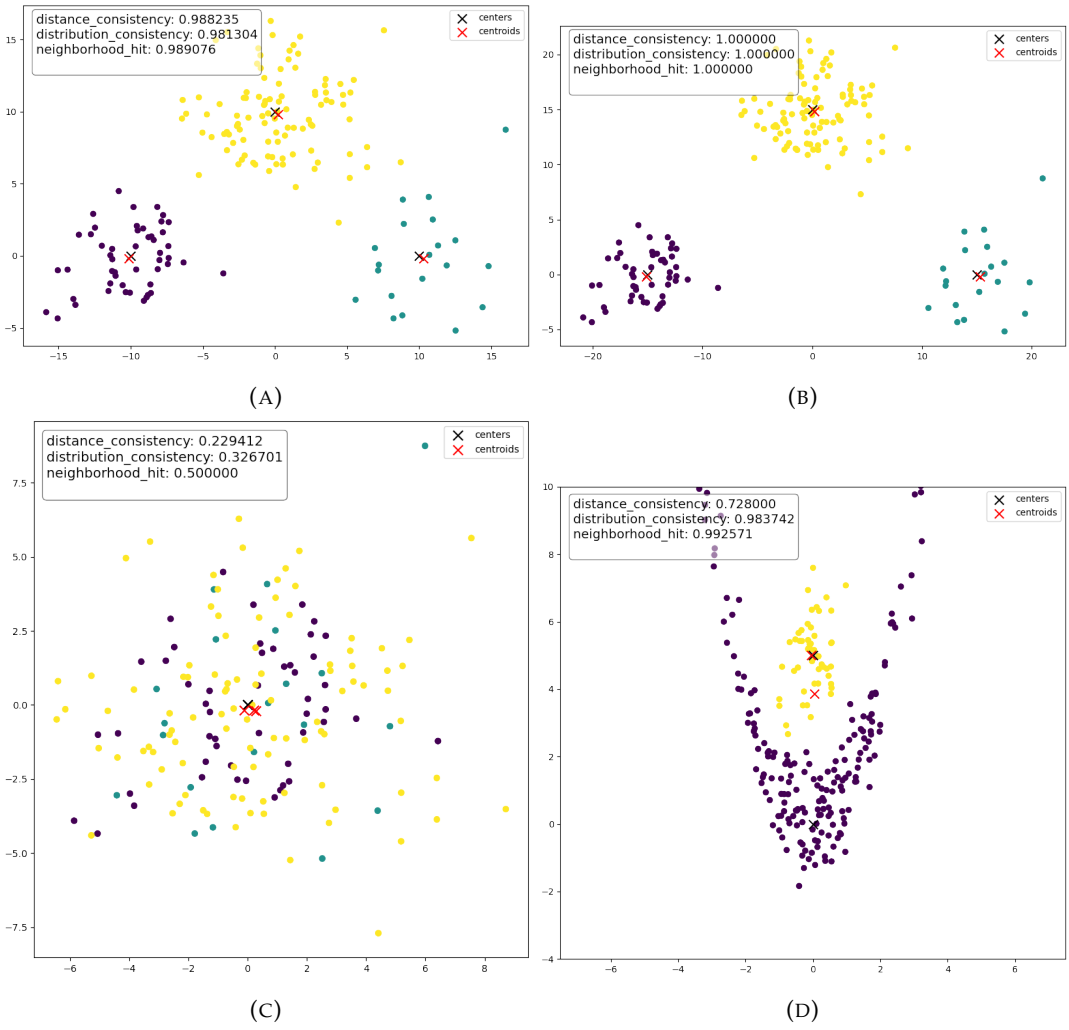


FIGURE 4.2: Demonstration of the ability of the different class consistency metrics presented in Section 3.1.3 to estimate the degree of cluster separation in a synthetic dataset. Figures (4.2a–4.2c) show data drawn from three two-dimensional Gaussian distributions. In the case of Figure 4.2d, both class samples were drawn from a Gaussian distribution, but one had its y coordinates transformed according to the quadratic formula $y' = x^2 + y$ to generate a non-convex cluster.

original dataset (random seed 42) and compute the metrics for the obtained results. Contrarily, the results in Table 4.4 are obtained by first projecting the full dataset of 48686 and subsequently selecting a random subset of 10000 samples (random seed 42) to compute the various projection performance metrics. Secondly, we observe the Shepard Goodness to be significantly lower. This implies that pointwise distances up to a monotonic scaling relation are less well preserved by the projection technique. Overall, LMDS and NPE still seem to perform reasonably well. Furthermore, it is surprising to see that for the NPE and t-SNE methods an increase in the number of nearest neighbors or perplexity, which should improve the preservation of global structure of the dataset, degraded the Shepard goodness of the projection. Table 4.4 also contains some additional projection performance metrics. These are the Jaccard similarity coefficient, distance consistency and distribution consistency. Each of these metrics were introduced in Section 3.1. It is remarkable to see that even

though the average fraction of false neighbors and missing neighbors in the projection is low the Jaccard similarity coefficient still tells us that the average proportion of overlap between k the nearest neighbor sets in the high dimensional space and the projection is low. The distance consistency and distribution consistency metrics demonstrate that the projections exhibit good class separation. This is also apparent from Figures 4.3, 4.4, 4.5 and 4.6. It is good to note the large discrepancy between the two values of distance consistency and distribution consistency for the t-SNE projection. Visually inspecting the t-SNE projection (Figure 4.5), one notices that the different classes are well-separated in the projection. However, the centroid of the QSO class is located within a concave region of the GAL cluster. Furthermore, the STAR cluster is segmented into two separate clusters, with one part located in another concave region of the GAL cluster. This may have caused the distance consistency metric to underestimate the degree of cluster separation, which explains the discrepancy. In conclusion, comparing the results for the neighborhood hit metric in Table 4.4 with those presented in Table 4.2, we notice that optimizing DR techniques with respect to the distribution consistency metric instead of the total metric (equation (4.3)) does not significantly improve cluster separation. Despite this, since our objective is a high degree of cluster separation, I use the parameter sets obtained through distribution consistency optimization for the remainder of this work.

I proceeded in a similar fashion as DR optimization to find the optimal parameter sets for the sharpening step of SDR. I use the following parameter grid for optimizing LGC:

- $\alpha = [0.005, 0.01, 0.015, 0.02, 0.025, 0.03, 0.035, 0.04, 0.045, 0.05, 0.055, 0.06]$
- $k = [25, 75, 125, 175, 225, 275, 325]$
- $T = [10, 15, 20]$

The same parameter grid is used for the CPz STAR, CPz GAL and CPz QSO datasets. The best configuration is found by finding the parameter set that yields a projection with the highest distribution consistency. This ensures a high degree of cluster separation in the projection. The resulting parameter sets, distribution consistency values and projections for the CPz STAR dataset are shown in Figures 4.7, 4.8, 4.9 and 4.10. The results for the remaining datasets are given in Appendix A. Tables 4.8, 4.9 and 4.10 report the values of the projection performance metrics when applying each optimized SDR technique to the different datasets. Looking at the projections, we observe a higher degree of segmentation in the SDR projections when compared with their DR counterparts. In fact, SDR seems to have adversely enhanced the oversegmentation features usually present in t-SNE projections. I investigate these oversegmentation features further in Chapter 7. Furthermore, one can definitely observe a higher degree of cluster separation which is also reflected quantitatively when comparing the values of the distribution consistency metric with the results obtained for DR. Lastly, we can deduce from the plots that LMDS and NPE show the greatest improvement in terms of cluster separation when compared to UMAP and t-SNE.

TABLE 4.4: DR techniques optimized and ranked with respect to the distribution consistency metric using 10000 samples from the projected CPz STAR dataset. The metrics were computed using $k = 500$.

DR method	Trustworthiness	Continuity	Jaccard Similarity Coefficient	Shepard Goodness	Neighborhood Hit	Distance Consistency	Distribution Consistency	Best Parameter Set
UMAP	0.9454	0.9610	0.4359	0.7737	0.9638	0.8547	0.9245	metric: "euclidean", min_dist: 0.1, num_neighbors: 20, umap_init: "spectral"
LMDS	0.9245	0.9679	0.3810	0.9132	0.9451	0.9581	0.8986	landmark_ratio: 0.08
NPE	0.9004	0.9363	0.3070	0.8548	0.9209	0.8191	0.8614	num_neighbors: 140
t-SNE	0.9190	0.9242	0.3997	0.4555	0.9286	0.6996	0.8556	sne_perplexity: 200, sne_theta: 0.5

TABLE 4.5: Parameter grids used for optimizing UMAP, LMDS, *t*-SNE and NPE.

DR method	Parameter Grid
UMAP	num_neighbors: [20,40,60,80,100,120,140,160,180,200,220,240,260,280,300], min_dist: [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0], metric: ["euclidean"], umap_init: ["spectral"]
LMDS	landmark_ratio: [0.005, 0.01, 0.015, 0.02, 0.025, 0.03, 0.035, 0.04, 0.045, 0.05, 0.055, 0.06, 0.065, 0.07, 0.075, 0.08, 0.085, 0.09, 0.095, 0.1]
t-SNE	sne_perplexity: [20,40,60,80,100,120,140,160,180,200], sne_theta: [0.5]
NPE	num_neighbors: [20,40,60,80,100,120,140,160,180,200]

TABLE 4.6: DR techniques optimized and ranked with respect to the distribution consistency metric using 10000 samples from the projected CPz GAL dataset. The metrics were computed using $k = 500$.

DR method	Trustworthiness	Continuity	Jaccard Similarity Coefficient	Shepard Goodness	Neighborhood Hit	Distance Consistency	Distribution Consistency	Best Parameter Set
UMAP	0.9400	0.9616	0.4066	0.8150	0.9739	0.8752	0.9450	metric: "euclidean", min_dist: 0.1, num_neighbors: 80, umap_init: "spectral"
LMDS	0.9391	0.9675	0.3956	0.9158	0.9477	0.9498	0.9096	landmark_ratio: 0.08
t-SNE	0.9466	0.9518	0.4226	0.6999	0.9450	0.7486	0.8920	sne_perplexity: 180, sne_theta: 0.5
NPE	0.8954	0.9372	0.2805	0.8562	0.9167	0.8428	0.8655	num_neighbors: 180

TABLE 4.7: DR techniques optimized and ranked with respect to the distribution consistency metric using 10000 samples from the projected CPz QSO dataset. The metrics were computed using $k = 500$.

DR method	Trustworthiness	Continuity	Jaccard Similarity Coefficient	Shepard Goodness	Neighborhood Hit	Distance Consistency	Distribution Consistency	Best Parameter Set
UMAP	0.9381	0.9595	0.4129	0.8007	0.9746	0.8904	0.9465	metric: "euclidean", min_dist: 0.1, num_neighbors: 40, umap_init: "spectral"
LMDS	0.9312	0.9629	0.3813	0.9021	0.9493	0.9540	0.9111	landmark_ratio: 0.04
t-SNE	0.9426	0.9499	0.4214	0.6865	0.9424	0.7535	0.8860	sne_perplexity: 180, sne_theta: 0.5
NPE	0.8551	0.9145	0.2569	0.7388	0.8916	0.7801	0.8404	num_neighbors: 80

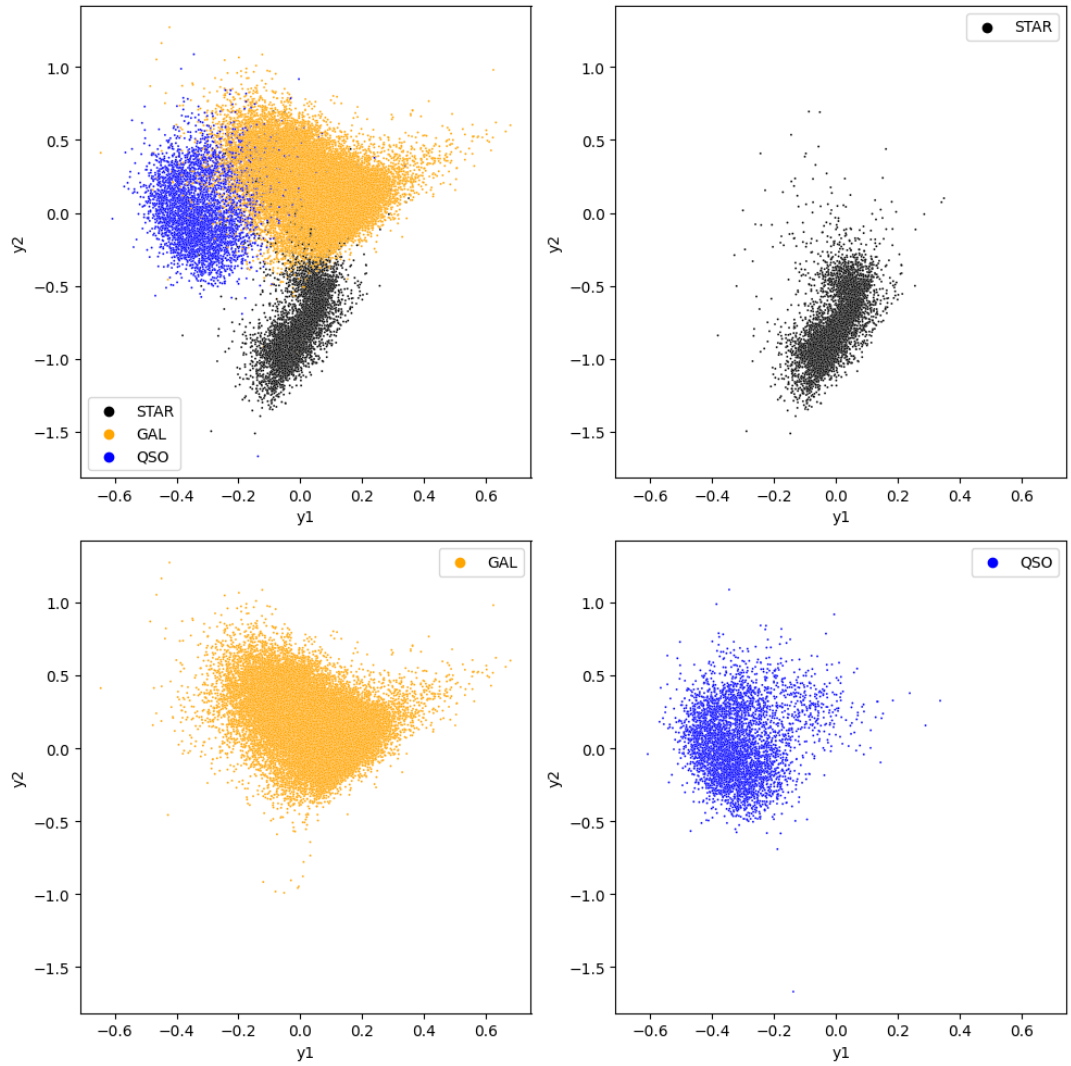


FIGURE 4.3: Plots showing the maximum distribution consistency LMDS projection ($M_{DC} = 0.8986$ with a landmark ratio of 0.08) of the CPz STAR dataset. Samples are colored according to the labeling provided by the CPz dataset.

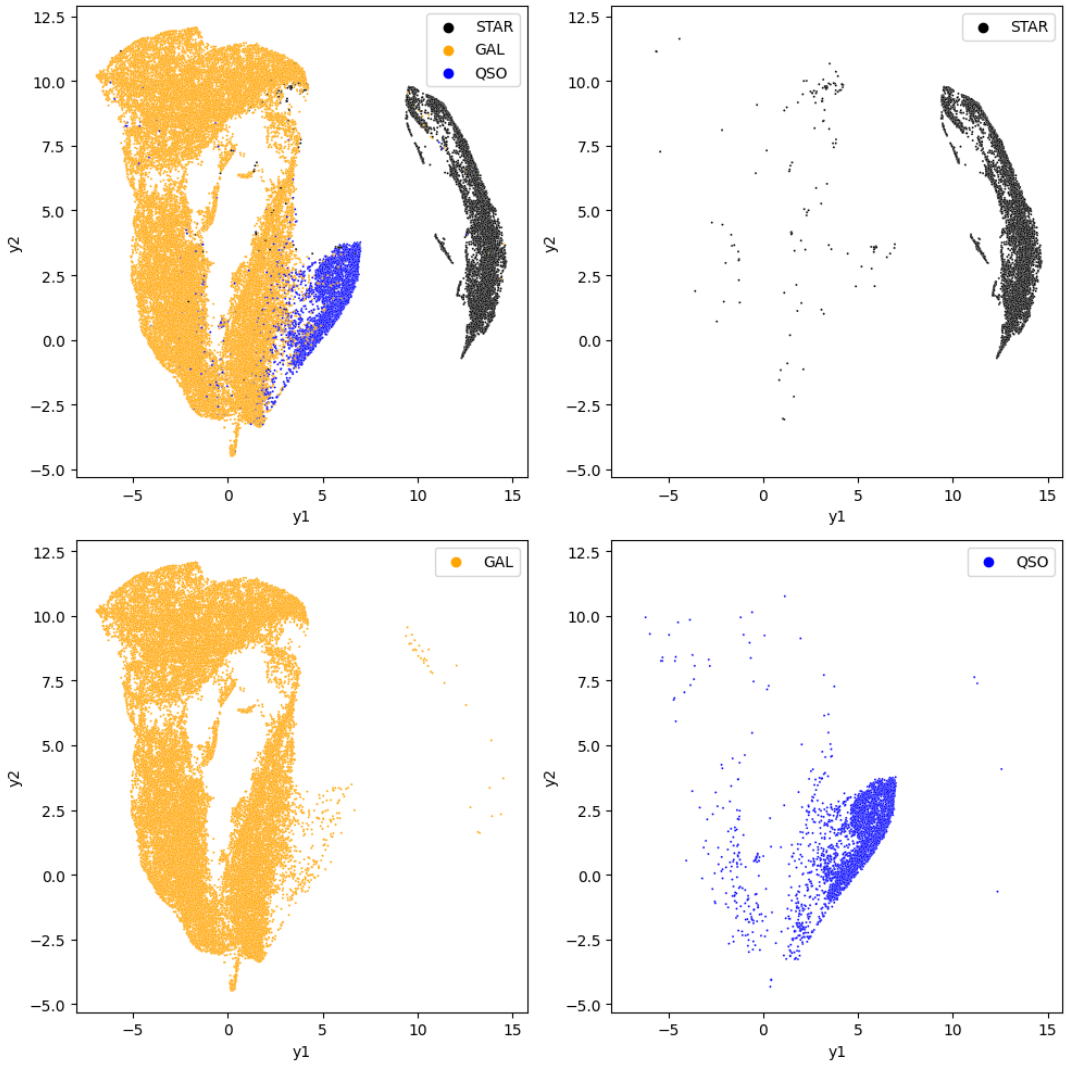


FIGURE 4.4: Plots showing the maximum distribution consistency UMAP projection ($M_{DC} = 0.9245$ with ("metric": "euclidean", "min_dist": 0.1, "num_neighbors": 20, "umap_init": "spectral")) of the CPz STAR dataset. Samples are colored according to the labeling provided by the CPz dataset.

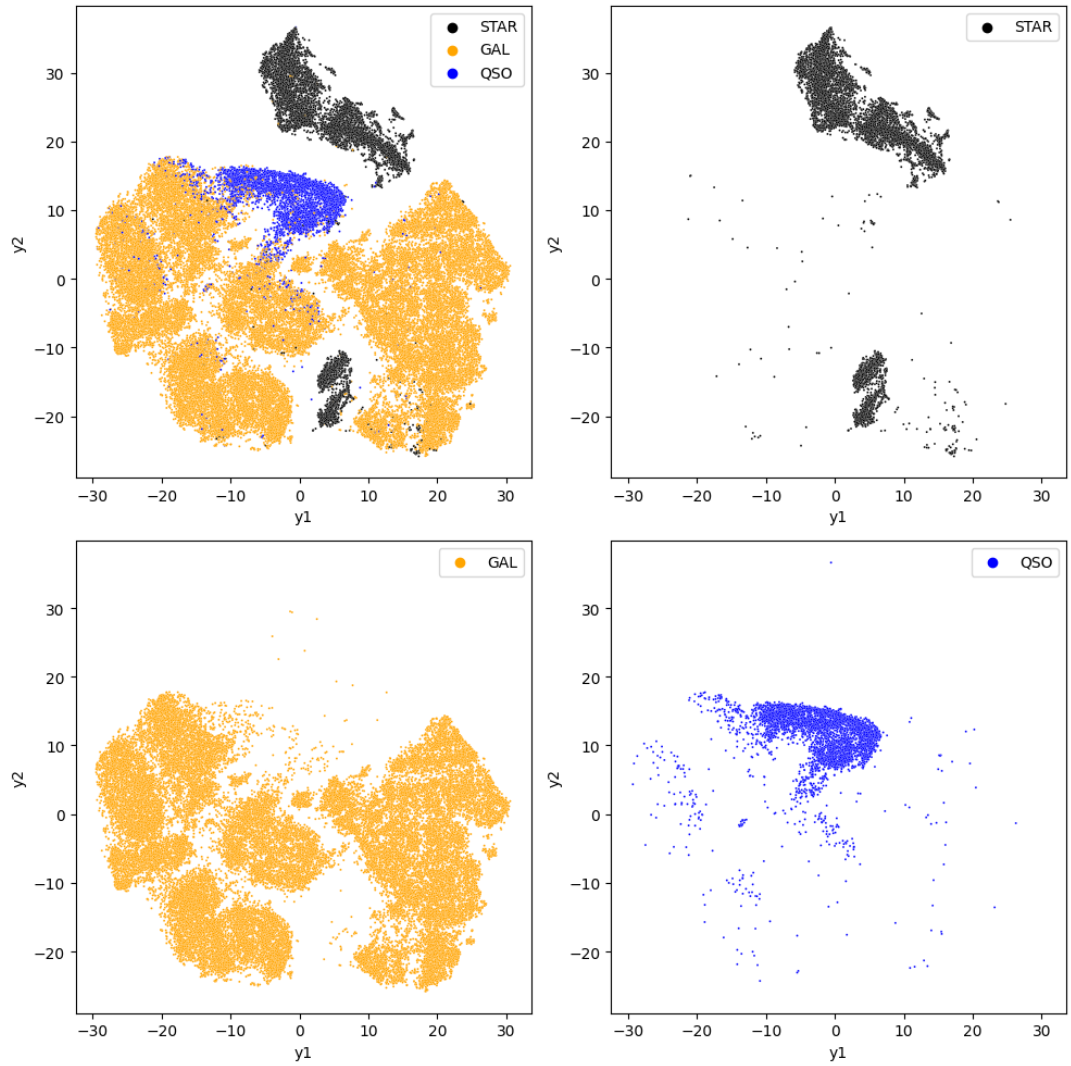


FIGURE 4.5: Plots showing the maximum distribution consistency *t*-SNE projection ($M_{DC} = 0.8556$ with ("sne_perplexity": 200, "sne_theta": 0.5)) of the CPz STAR dataset. Samples are colored according to the labeling provided by the CPz dataset.

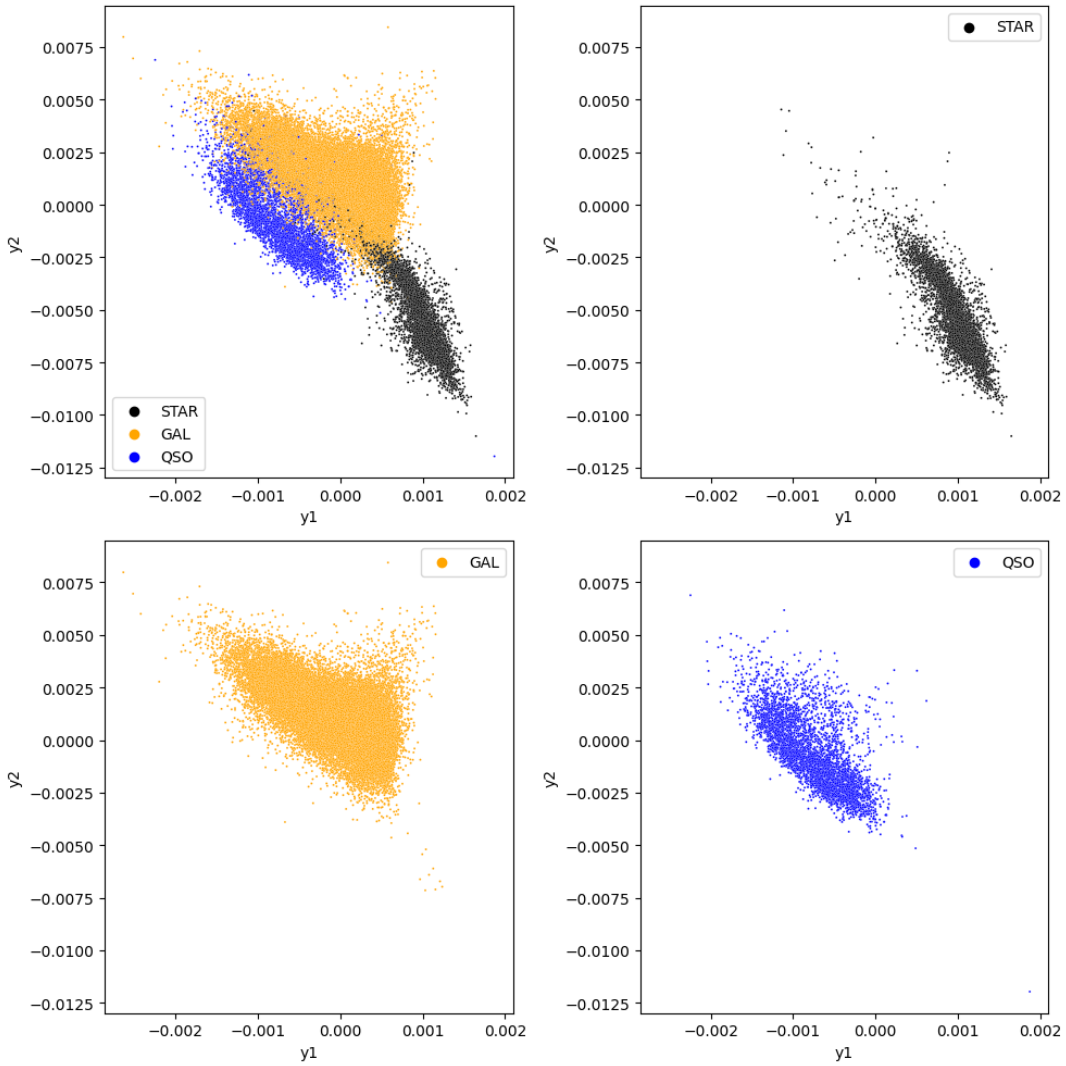


FIGURE 4.6: Plots showing the maximum distribution consistency NPE projection ($M_{DC} = 0.8614$ with 140 nearest neighbors) of the CPz STAR dataset. Samples are colored according to the labeling provided by the CPz dataset.

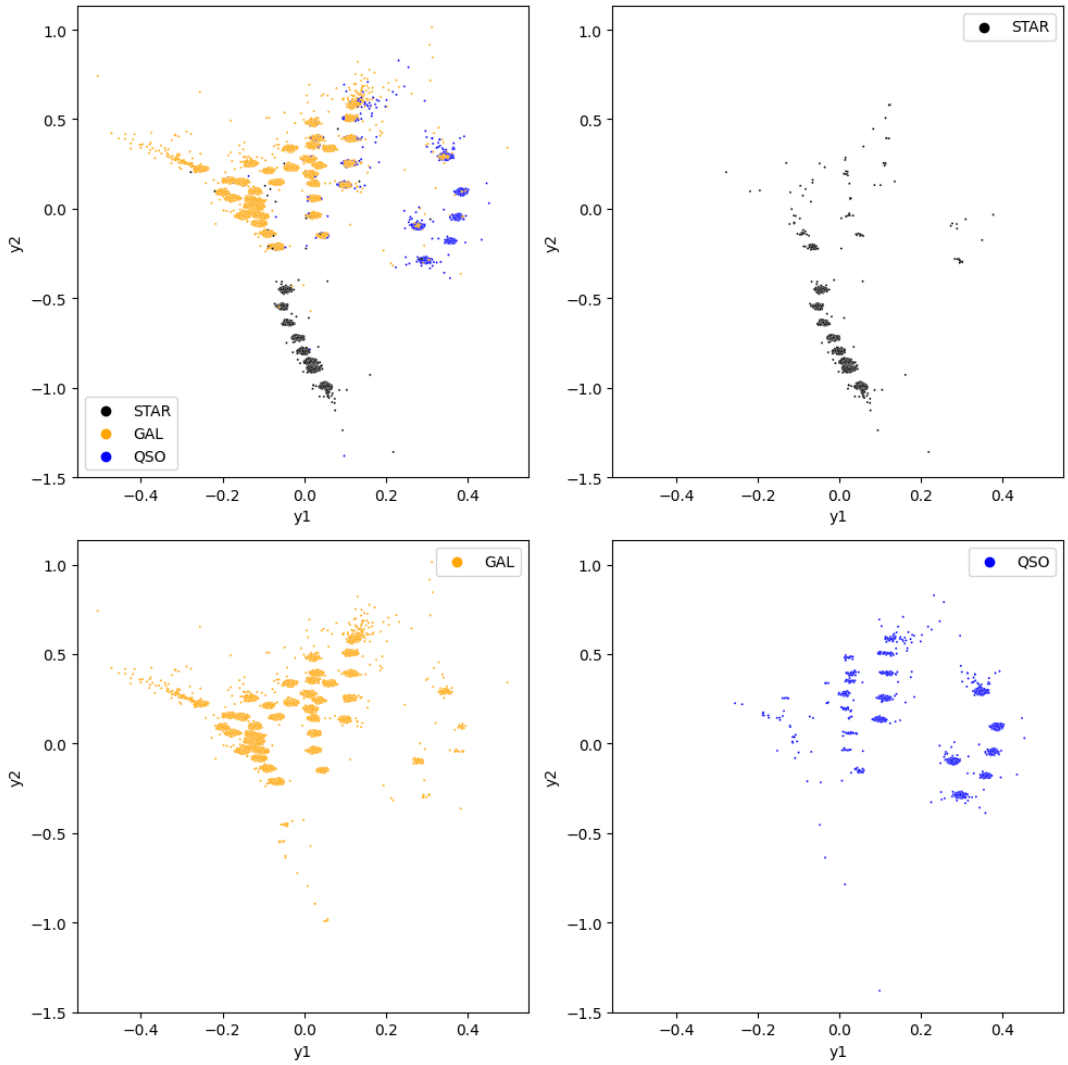


FIGURE 4.7: Plots showing the maximum distribution consistency sharpened LMDS projection ($M_{DC} = 0.9366$ with $(\alpha = 0.03, k = 325, T = 10)$ and a landmark ratio of 0.08) of the CPz STAR dataset. Samples are colored according to the labeling provided by the CPz dataset.

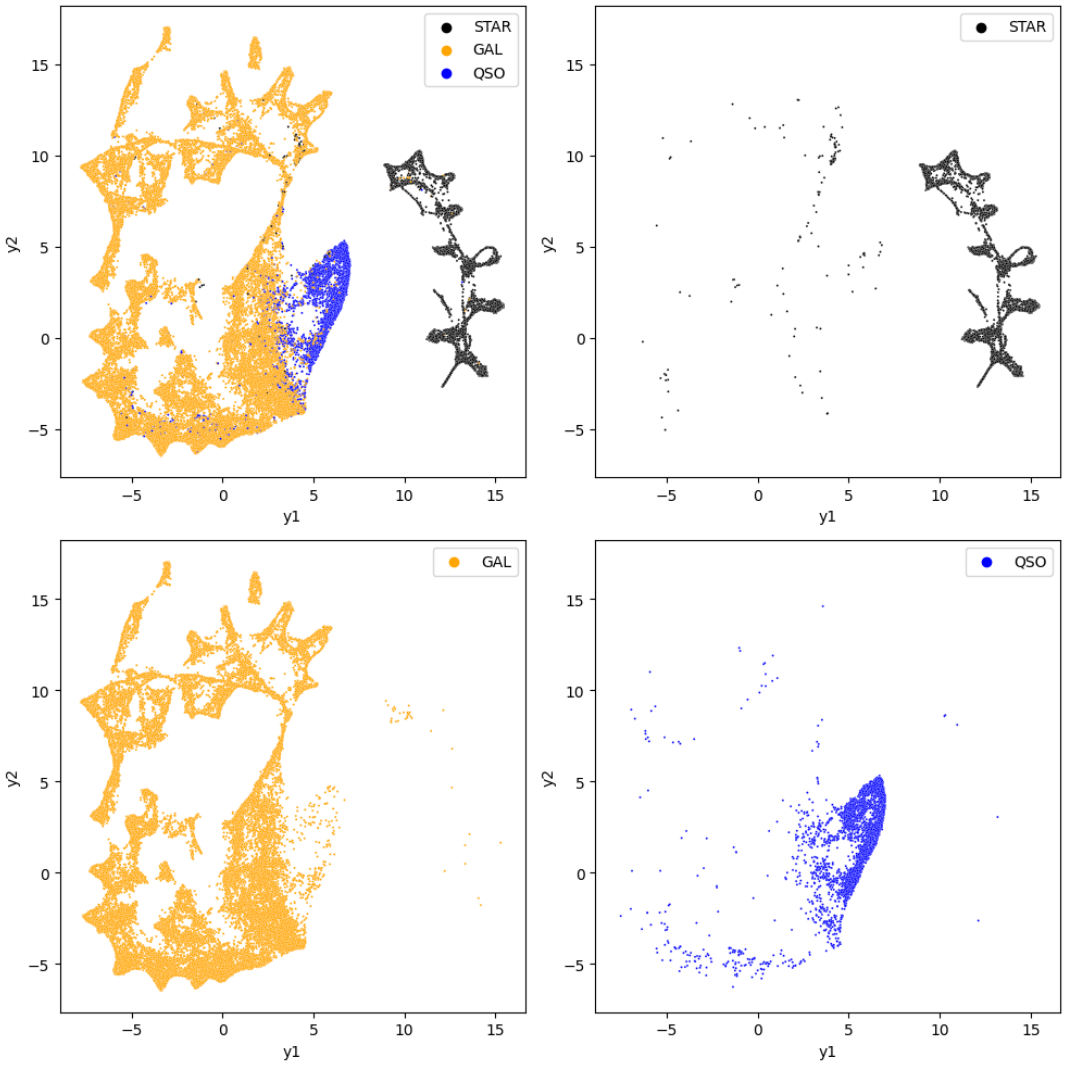


FIGURE 4.8: Plots showing the maximum distribution consistency sharpened UMAP projection ($M_{DC} = 0.9250$ with $(\alpha = 0.005, k = 275, T = 10)$ and ("metric": "euclidean", "min_dist": 0.1, "num_neighbors": 20, "umap_init": "spectral")) of the CPz STAR dataset. Samples are colored according to the labeling provided by the CPz dataset.

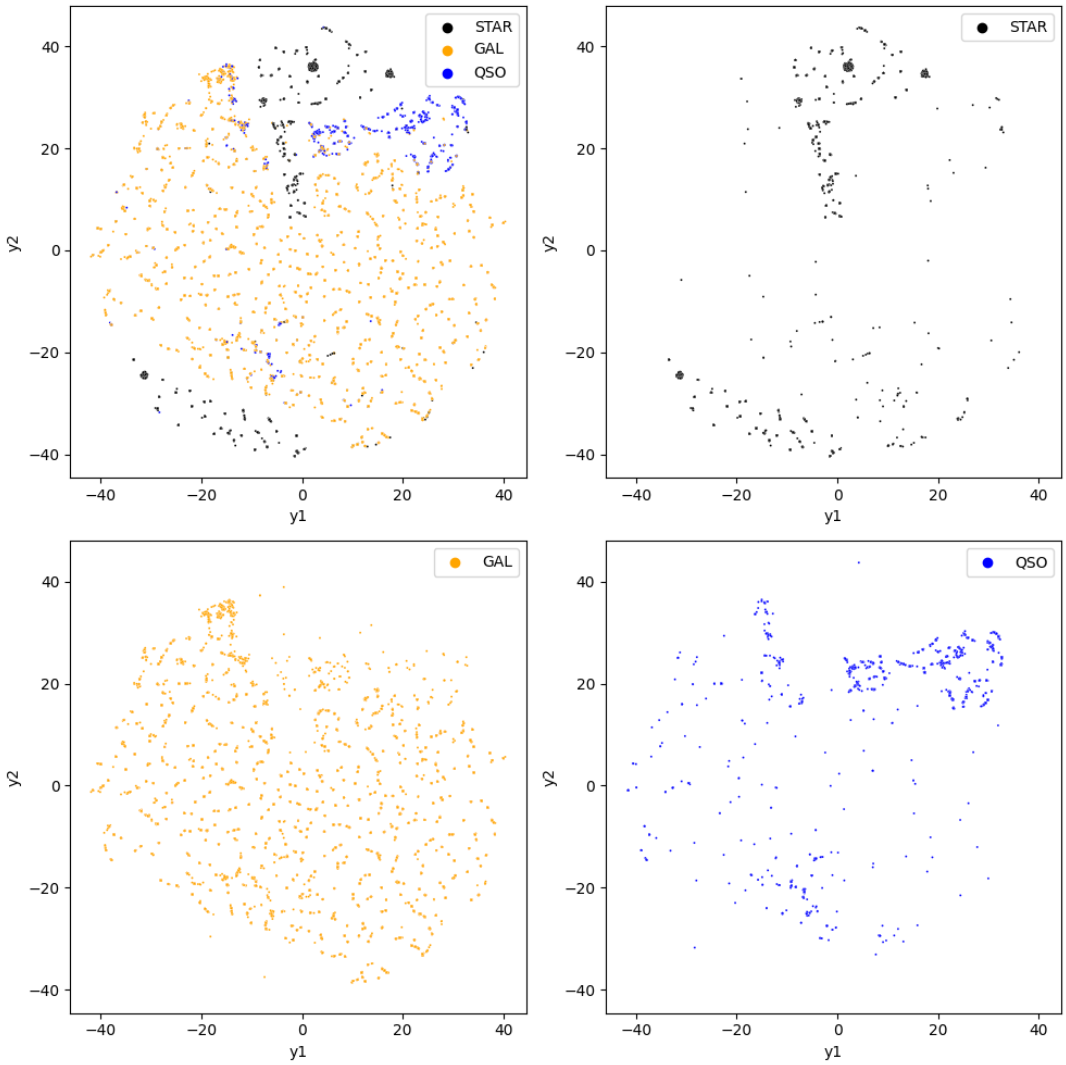


FIGURE 4.9: Plots showing the maximum distribution consistency sharpened t-SNE projection ($M_{DC} = 0.9255$ with $(\alpha = 0.01, k = 25, T = 15)$ and ("sne_perplexity": 200, "sne_theta": 0.5)) of the CPz STAR dataset. Samples are colored according to the labeling provided by the CPz dataset.

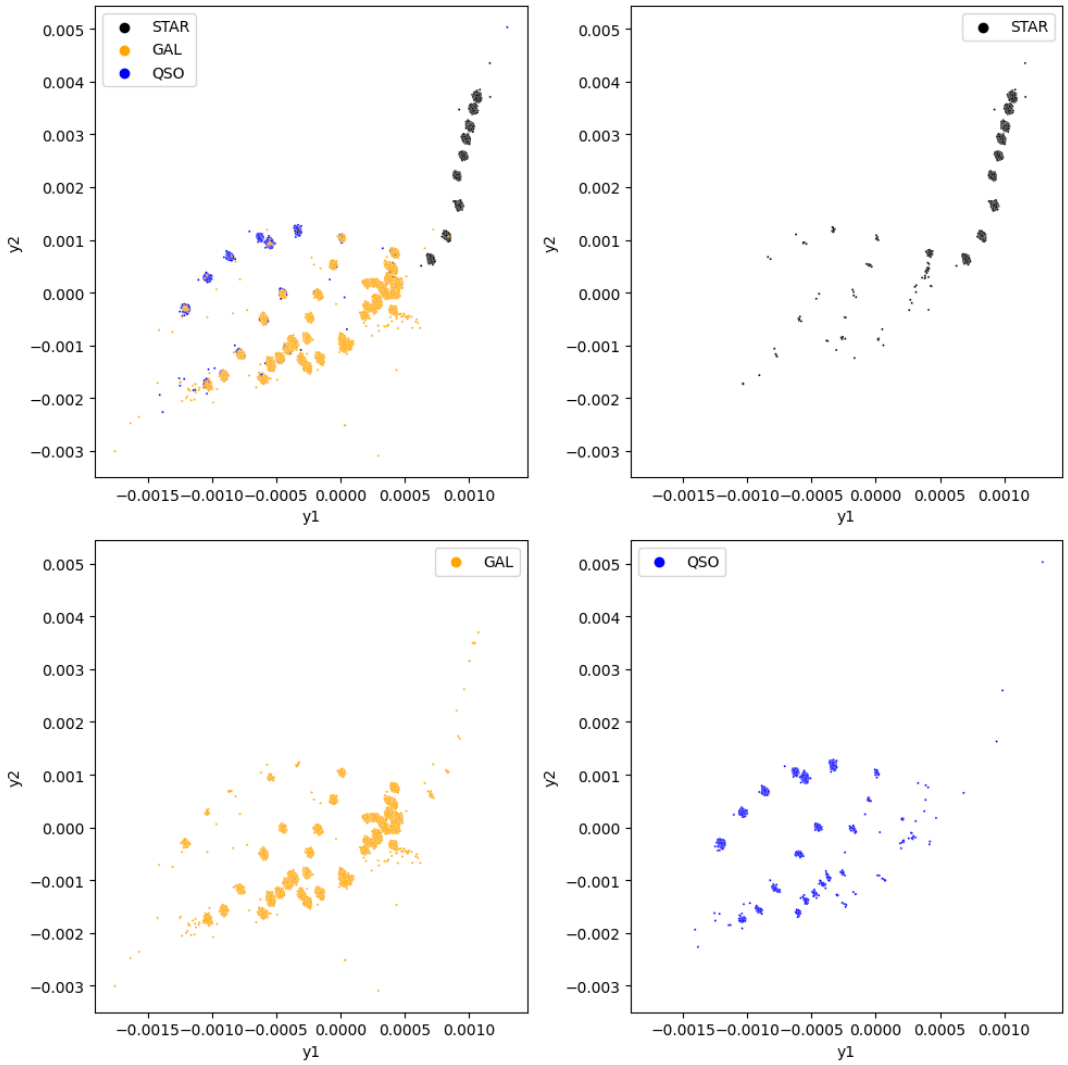


FIGURE 4.10: Plots showing the maximum distribution consistency sharpened NPE projection ($M_{DC} = 0.9279$ with $(\alpha = 0.02, k = 325, T = 20)$ and 140 nearest neighbors) of the CPz STAR dataset. Samples are colored according to the labeling provided by the CPz dataset.

TABLE 4.8: SDR techniques optimized and ranked with respect to the distribution consistency metric using 10000 samples from the projected CPz STAR dataset. The metrics were computed using $k = 500$.

SDR method	Trustworthiness	Continuity	Jaccard Similarity Coefficient	Shepard Goodness	Neighborhood Hit	Distance Consistency	Distribution Consistency	Best LGC Parameter Set
SLMDS	0.9201	0.9484	0.3402	0.8895	0.9573	0.9722	0.9149	'T': 10, 'alpha': 0.03, 'k': 325
SUMAP	0.9262	0.9426	0.3814	0.5676	0.9584	0.8101	0.9105	'T': 10, 'alpha': 0.005, 'k': 275
SNPE	0.9184	0.9376	0.3186	0.8481	0.9350	0.8990	0.8850	'T': 20, 'alpha': 0.02, 'k': 325
S*t*-SNE	0.8418	0.8806	0.2971	0.3764	0.8537	0.5082	0.7254	'T': 15, 'alpha': 0.01, 'k': 25

TABLE 4.9: SDR techniques optimized and ranked with respect to the distribution consistency metric using 10000 samples from the projected CPz GAL dataset. The metrics were computed using $k = 500$.

SDR method	Trustworthiness	Continuity	Jaccard Similarity Coefficient	Shepard Goodness	Neighborhood Hit	Distance Consistency	Distribution Consistency	Best LGC Parameter Set
SUMAP	0.9444	0.9492	0.3894	0.7320	0.9678	0.8206	0.9325	'T': 10, 'alpha': 0.005, 'k': 125
SLMDS	0.9347	0.9482	0.3458	0.8824	0.9592	0.9787	0.9233	'T': 15, 'alpha': 0.02, 'k': 325
SNPE	0.9201	0.9425	0.3120	0.8629	0.9489	0.9384	0.9107	'T': 15, 'alpha': 0.02, 'k': 325
S*t*-SNE	0.9135	0.9235	0.3487	0.5928	0.9139	0.4880	0.8237	'T': 10, 'alpha': 0.005, 'k': 25

TABLE 4.10: SDR techniques optimized and ranked with respect to the distribution consistency metric using 10000 samples from the projected CPz QSO dataset. The metrics were computed using $k = 500$.

SDR method	Trustworthiness	Continuity	Jaccard Similarity Coefficient	Shepard Goodness	Neighborhood Hit	Distance Consistency	Distribution Consistency	Best LGC Parameter Set
SUMAP	0.9422	0.9466	0.3821	0.7140	0.9696	0.8386	0.9372	'T': 10, 'alpha': 0.005, 'k': 225
SLMDS	0.9275	0.9458	0.3425	0.8702	0.9634	0.9792	0.9325	'T': 20, 'alpha': 0.015, 'k': 275
SNPE	0.9141	0.9359	0.3092	0.8310	0.9446	0.8296	0.9007	'T': 10, 'alpha': 0.03, 'k': 325
S*t*-SNE	0.8759	0.9122	0.3212	0.5447	0.8844	0.5101	0.7747	'T': 10, 'alpha': 0.01, 'k': 25

Chapter 5

Neural Network Projection

In the previous chapter I present the sharpened dimensionality reduction (SDR) method based on work by Kim et al. (2022b) and some results using 3 different datasets and the LMDS, UMAP, tSNE and NPE dimensionality reduction (DR) methods. As illustrated both qualitatively by various plots of the projections and quantitatively by the distribution consistency metric, the SDR technique is able to achieve good cluster separation for the CPz STAR, CPz GAL and CPz QSO datasets. DR methods such as LMDS and NPE especially show significant improvements.

Despite this success SDR has some drawbacks. One is the fact that SDR doesn't have out-of-sample (OOS) ability. This trait was introduced in Section 4.2 as "the ability of a projection technique to extrapolate to new data based on previous training". The lack of OOS capability makes SDR particularly computationally unscalable when new data arrives, since SDR would have to generate an new projection for an ever increasing dataset. To mitigate this issue Kim et al. (2022a) introduced SDR-NNP, "Sharpened Dimensionality Reduction with Neural Network Projection". This method leverages the scalability, ease of use and OOS capability of neural networks by training a deep neural network to reproduce a projection maintaining the high degree of cluster separation provided by SDR. They showed that SDR-NNP is able to consistently produce projections with a high degree of cluster separation.

In this chapter I describe the architecture of the neural network I have trained to obtain SDR-NNP projections akin to the ones produced by SDR. I then describe the procedure used to optimize the network's parameters. Finally, I present the training and testing results for the CPz STAR dataset to demonstrate the capability of SDR-NNP to maintain a high degree of cluster separation.

5.1 Architecture & Optimization

The architecture of the deep neural network used for SDR-NNP is shown in Figure 5.1. The network architecture is fairly straightforward and was implemented using TensorFlow (Abadi et al., 2015). The network consists of an input and output layer with a number of hidden fully connected blocks in between. I refer to these blocks as "dense blocks". The idea behind the dense blocks is that they gradually step down the number of data dimensions. To achieve this the dimensionality of each block (called "units" by TensorFlow) halves every two layers. Furthermore, the second dense block has a dimensionality that is 3/4 that of the first dense block, truncated to the nearest integer. Additionally, I give the first dense block a dimensionality that is equal to the dimensionality of the input layer. The idea behind this is that this allows the first dense block to transform the input data non-linearly in the high dimensional space before projecting.

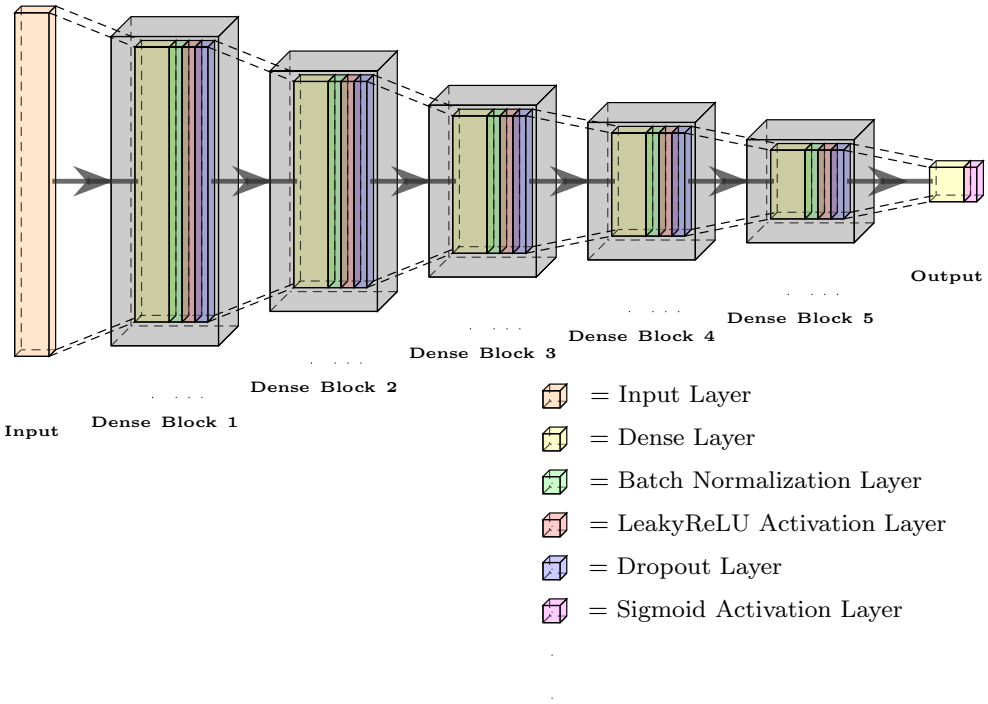


FIGURE 5.1: Architecture of the deep neural network used for SDR-NNP.

Each dense block consists of a number of layers. The first layer is the dense layer, which is a regular fully connected layer, applied with bias and linear activation. This layer applies a matrix-vector product between a weights matrix and the input vector and adds a bias to each output dimension:

$$\mathbf{y}_i = W\mathbf{x}_i + \mathbf{b}. \tag{5.1}$$

Essentially, each entry of the weights matrix resembles the weight corresponding to an edge in the fully connected graph connecting each node in the preceding layer to each node in the current layer. The entries of the weights matrix and bias vector are parameters that can be tuned by the optimization procedure.

The next layer in the architecture is a batch normalization layer. This layer ensures that the mean output of the dense block stays close to 0 and that the standard deviation stays close to 1 for each batch of data. This makes training of neural networks more stable allowing for higher learning rates making optimization faster (Ioffe and Szegedy, 2015). The reason behind this is still not fully understood. The behavior of the batch normalization layer during training and inference is different. During training the batch normalization layer returns the following:

$$y_{ij} = \frac{\gamma\left(x_{ij} - \mu_j\right)}{\sqrt{\sigma_j^2 + \epsilon}} + \beta, \tag{5.2}$$

where γ and β are trainable scaling and offset parameters and $\epsilon = 0.0001$ is a regularization parameter. Furthermore μ_j and σ_j^2 are the mean and variance of the batch corresponding to each node j. To be able to apply batch normalization during inference, the batch normalization layer also computes a moving mean and variance for

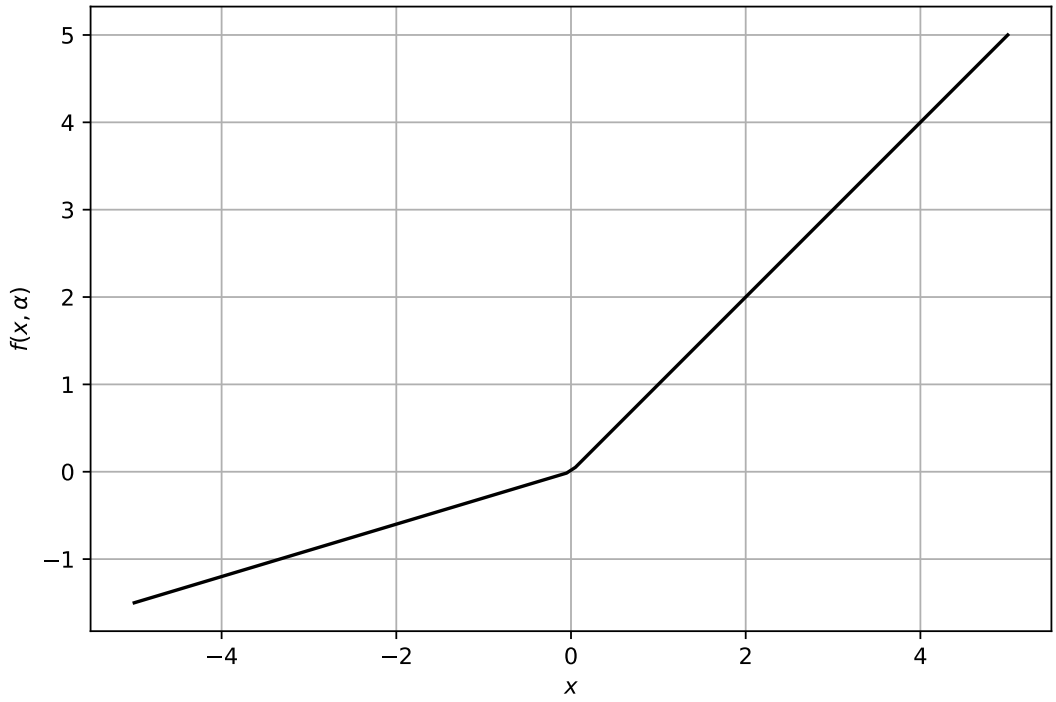


FIGURE 5.2: A plot of the LeakyReLU function used as an activation function in the dense blocks in Figure 5.1.

each node j during training. This is done through the following set of update rules:

$$\hat{\mu}_j = \hat{\mu}_j \cdot \text{momentum} + \mu_j \cdot (1 - \text{momentum}) \text{ and}$$
$$\hat{\sigma}_j = \hat{\sigma}_j \cdot \text{momentum} + \sigma_j \cdot (1 - \text{momentum}).$$

where the momentum is a hyperparameter set to be 0.6 in all of the models. The obtained moving means ($\hat{\mu}_j$) and variances ($\hat{\sigma}_j^2$) are used to apply batch normalization during inference:

$$y_{ij} = \frac{\gamma \left(x_{ij} - \hat{\mu}_j \right)}{\sqrt{\hat{\sigma}_j^2 + \epsilon}} + \beta. \tag{5.3}$$

The output of each batch normalization layer subsequently passes through an activation layer with a LeakyReLU (Leaky Rectified Linear Unit) activation function:

$$f(x) = \begin{cases} \alpha \cdot x & \text{if } x < 0 \\ x & \text{if } x \geq 0, \end{cases} \tag{5.4}$$

where x is the output of a single node in the preceding batch normalization layer and α is a hyperparameter. I use the default specified by TensorFlow, 0.3. An illustration of the LeakyReLU activation function is shown in Figure 5.2. This activation function introduces non-linearity while still allowing for a small gradient whenever the output of a neuron becomes negative. The latter is important to prevent neurons to become inactive once the output becomes negative at some point during the training procedure. This effect is known as the "dying ReLU problem" and occurs when using the ReLU activation function, which has zero gradient for negative values.

The final layer of the dense block is the dropout layer, which is only active during training. This layer randomly sets input units to zero at a specified "dropout rate".

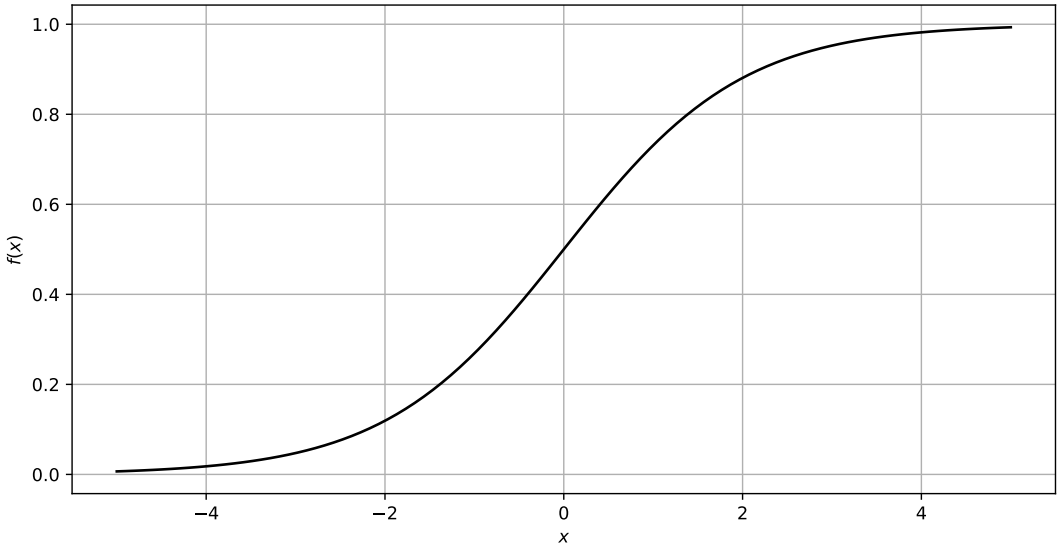


FIGURE 5.3: A plot of the sigmoid function used as an activation function in the output layer in Figure 5.1.

The remaining non-zero units are then scaled by a factor $\frac{1}{1-\text{rate}}$ to ensure that the sum over all units remains the same. The effect that this layer has on the training procedure is that it prevents overfitting on the training data. However, this comes at a disadvantage, namely that the higher the dropout rate, the slower the optimization. In addition, the randomness of the output can make it impossible for the network to properly tune the parameters and find a good model. In practice I find that turning off the dropout layer (i.e., setting the dropout rate to zero) improves the performance of the SDR-NNP model while still showing no sign of overfitting. Therefore, I omit this dropout layer from the models presented in this work.

Examining Figure 5.1 one may notice that the output layer uses a different activation than the activations used in the dense blocks. This is because I want the output of the neural network to be scaled between 0 and 1 for easy comparison with the target projection, that is, the projections produced by SDR, which I also scale to be between 0 and 1. This so-called sigmoid activation function is given in Figure 5.3.

I now explain the optimization procedure. This procedure starts by splitting the full dataset projected by SDR into a training and test set (20% of the data was included in the test set). The training set is used to train the neural network, i.e., to optimize the network's model parameters (weights, biases, etc.) while the test set is used to validate the neural network performance and to ensure that it generalizes properly to unseen data. There are far more galaxies in the dataset compared to stars and QSOs. Therefore, I split the training and test sets such that the relative fractions of stars, galaxies and QSOs is preserved.

One needs to define an "objective function" to optimize any neural network. In machine learning this function is also referred to as a "cost function" or "loss function". In this case we need to formulate a loss function which punishes based on the offset between the true locations and the neural network generated positions in the projection. To this end one can envisage computing the average or median offset in a neural network generated projection. The offsets can be either computed as squared or absolute distances. After experimenting with different loss functions it turned out that the mean absolute error (MAE) loss function provided by TensorFlow yielded

the best optimization results:

$$\text{loss} = \frac{1}{N}\sum_{i=0}^{N-1}\frac{1}{n}\sum_{j=0}^{d-1}\left|\mathbf{y}_{\text{true}} - \mathbf{y}_{\text{predicted}}\right|_{ij}, \tag{5.5}$$

with N being the total number of samples in the batch used for optimization, n being the number of dimensions of the output layer (i.e., two in our case), $\mathbf{y}_{\text{true}}$ being the 2D vector marking the true location of the training sample as given by the SDR technique and $\mathbf{y}_{\text{predicted}}$ being the predicted position vector generated by the neural network.

To find the set of model parameters with the lowest value for the loss function neural networks use gradient descent techniques which are iterative methods to find local minima. One of the most popular gradient descent methods in deep learning is the Adam optimization algorithm (Kingma and Ba, 2014). Attractive features of Adam include its computational scalability and stability. That is, Adam performs well for large datasets with a high number of dimensions in the presence of noise and requires little hyperparameter tuning. In addition, it often performs better than other optimization algorithms such as AdaGrad and RMSProp; see Kingma and Ba (2014) for more detailed information on the Adam optimization algorithm. After some experimenting with the various hyperparameters I have chosen to keep the defaults provided by TensorFlow, as they yielded the best results (i.e., a learning rate of $\alpha = 0.001$, $\beta_1 = 0.9$ and $\beta_2 = 0.999$).

5.2 Results

In the previous section I have discussed the different hyperparameters of the neural network and how I split datasets into training and test sets. In this section I present the results of training the neural network architecture presented in the previous section to reproduce the SDR projections obtained through sharpened LMDS, UMAP, tSNE and NPE. See Figures 5.4, 5.5, 5.6 and 5.7 respectively. As in the previous chapter, the results for the CPz GAL and CPz QSO datasets are provided in the Appendix (see Appendix B).

I let the optimization procedure go through 20000 training epochs to tune the model parameters of the neural network. To prevent overfitting on the training set, I use cross validation at each epoch, where 25% of the training set was set aside to form a validation set. Aside from ensuring that the training set is different at each epoch, making it harder for the neural network to overfit on the training set, this also allows computing a validation loss at each epoch. This validation loss can be plotted together with the training loss to validate whether or not the neural network overfits on the training set. When this happens, the validation loss will start to increase relative to the training loss. Examining, for example, Figure 5.4 one can observe that this is not the case. Note that I make a distinction between the training loss and the inferential training loss. This distinction is made because the batch normalization layers in the neural network behave differently during training and during inference. In addition to looking at the validation loss, one can also compare the final training loss to the test loss (also given in the Figures). The test losses are always close to the values for the training loss which implies that the trained neural networks generalize well to unseen data.

Comparing the SDR embedding to the SDR-NNP embeddings in the various figures, one may notice a number of points. Firstly, the SDR-NNP embeddings only

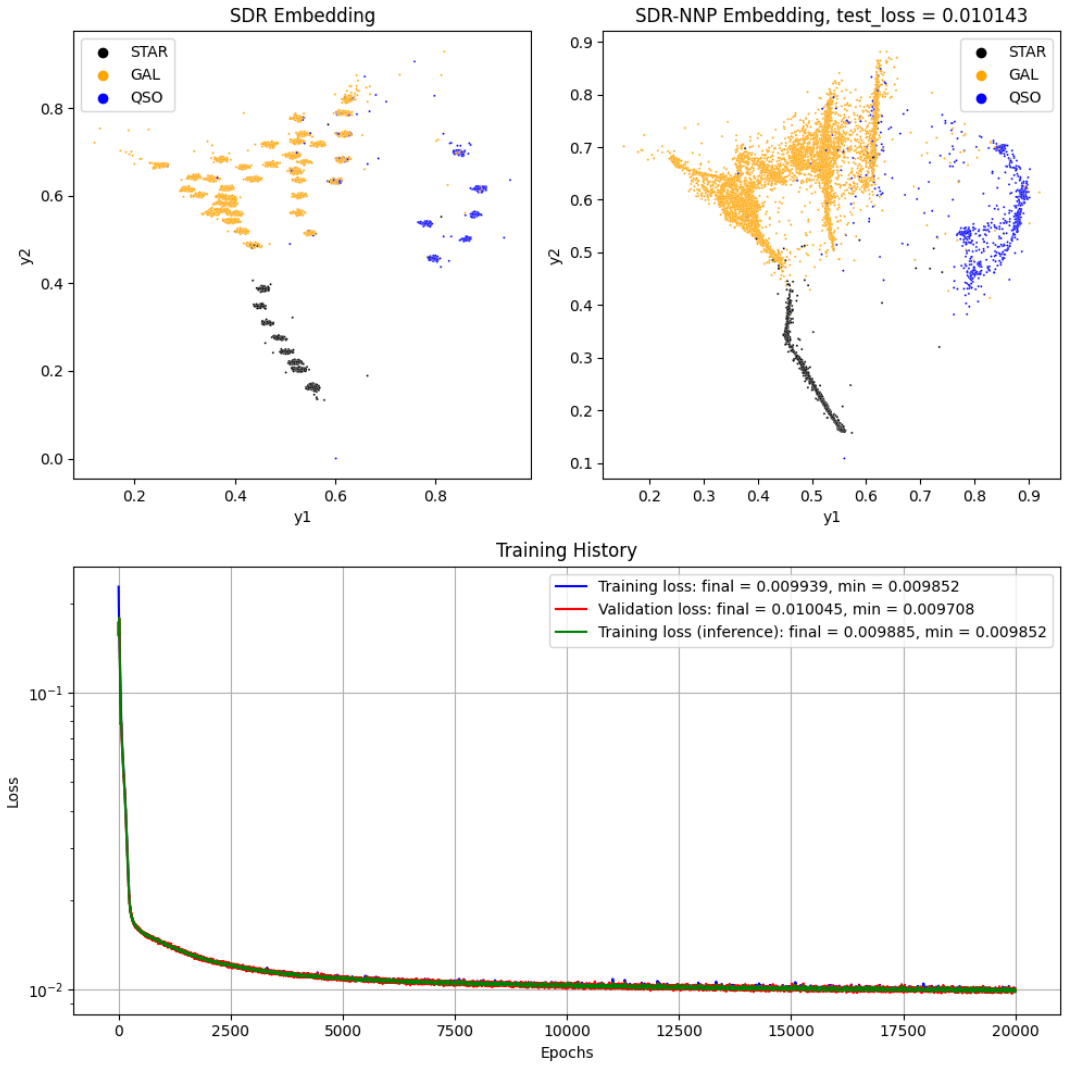


FIGURE 5.4: NNP testing and training results for sharpened LMDS optimized for the CPz STAR dataset.

seem to be able to reproduce the large-scale structure of the SDR embedding. That is, small clusters and filaments are not properly reproduced. This is not necessarily a problem for us, since the star, galaxy and QSO cluster still appear to be well-separated in the projection, which is what is required for classification. Secondly, SDR-NNP seems to remove a lot of the oversegmentation present in the SDR embeddings, making the projection appear more continuous. In Chapter 7 I investigate whether this oversegmentation is a result of the structure of the high-dimensional data or whether it is a feature introduced by the sharpening step of SDR.

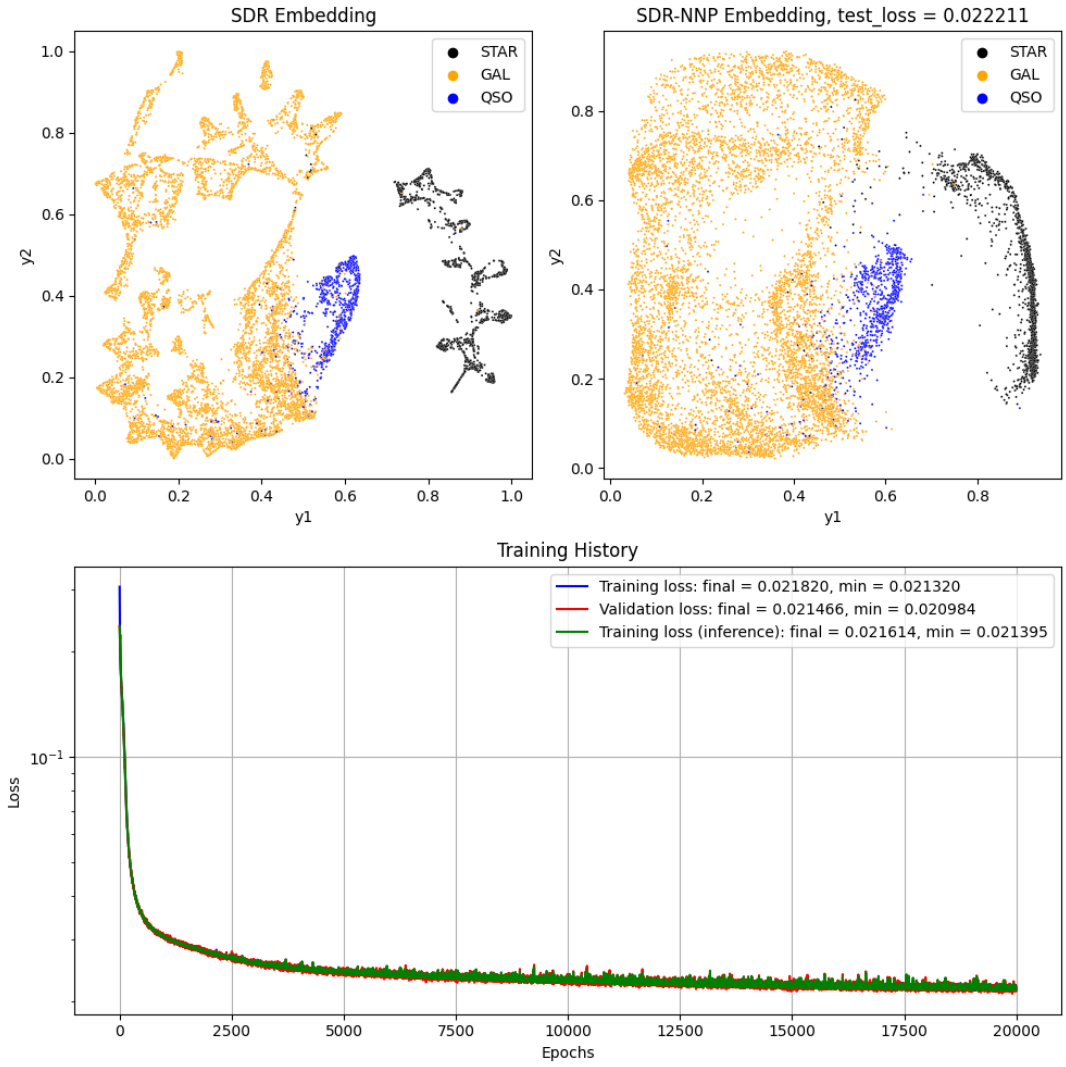


FIGURE 5.5: NNP testing and training results for sharpened UMAP optimized for the CPz STAR dataset.

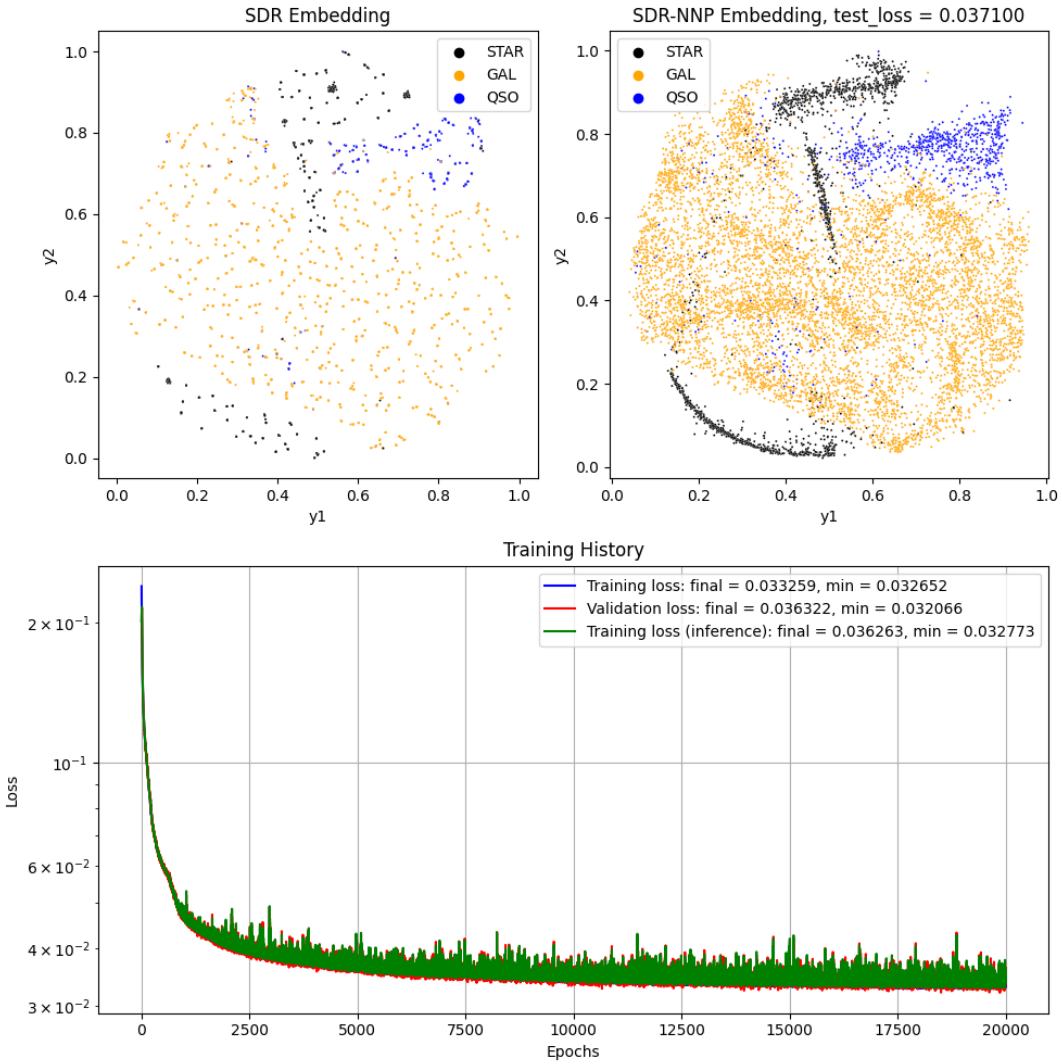


FIGURE 5.6: NNP testing and training results for sharpened tSNE optimized for the CPz STAR dataset.

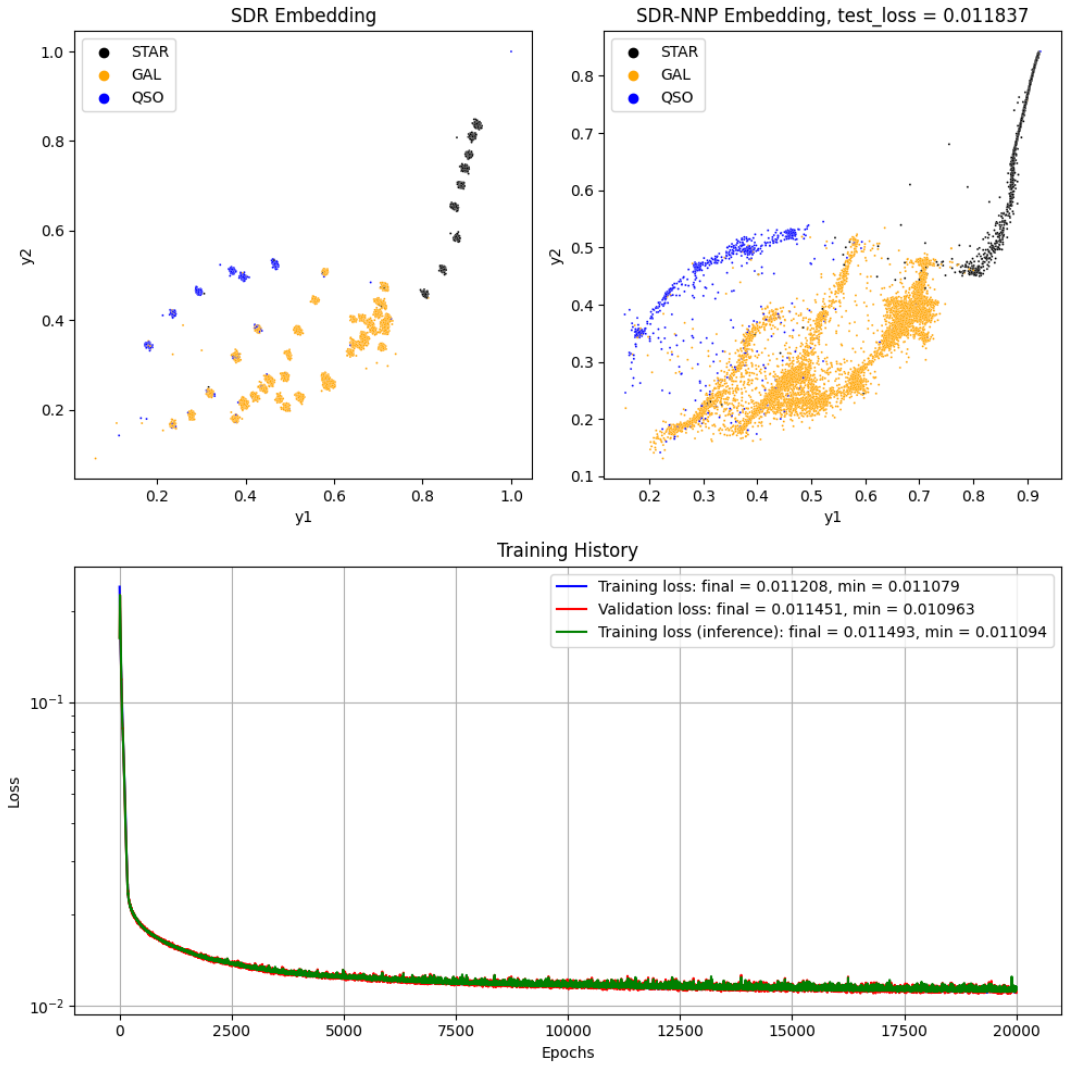


FIGURE 5.7: NNP testing and training results for sharpened NPE optimized for the CPz STAR dataset.

Chapter 6

Classification

In this chapter I present and test various classifiers to perform star, galaxy and QSO classification using the projection results yielded by the various SDR-NNP models presented in Chapter 5. In addition, I present methods to consolidate the classification results from the different datasets, i.e., CPz STAR, CPz GAL and CPz QSO, in the hope of improving the classification results. The feature sets of these datasets are defined by LF20 to maximize the performance of each binary classification task, i.e., star/non-star, galaxy/non-galaxy and QSO/non-QSO.

This chapter starts with a section introducing the various classification algorithms that I use, followed by a section outlining the methods I employ to consolidate the CPz STAR, CPz GAL and CPz QSO results. The last section of this chapter presents the classification results.

6.1 Classifiers

In this work I use four different types of classifiers. These are

- the *k*-nearest neighbors vote based classifier (KNNC);
- the Support Vector Machine Classifier (SVMC), based on LIBSVM (Chang and Lin, 2011);
- the Multi-layer Perceptron Classifier (MLPC), a deep-neural-network classifier; and
- the XGBoost Classifier (XGBC) (Chen and Guestrin, 2016), a tree-based classifier.

I have used the scikit-learn implementations by Pedregosa et al. (2011) for each of these classifiers except for XGBoost.

KNNC classifies new samples based on the majority class of the *k*-nearest neighbor set of the new sample. In addition to predicting the class label, KNNC is also able to yield estimates of class probabilities by computing the fractions of samples in the *k*-nearest neighbor set corresponding to each class.

SVMC as implemented by Pedregosa et al. (2011) is based on the C-Support Vector Classification formulation of LIBSVM (Chang and Lin, 2011). Support Vector Machines (SVMs) are supervised-learning models that try to find a model that assigns samples to one of two classes. The model is designed in such a way that it attempts to maximize the "margin" between training samples of two classes in some mapping whilst subject to the constraint of classifying everything (mostly) correctly. A more formal definition of this optimization problem is given in Chang and Lin (2011). In the following I give a more elaborate explanation of C-Support Vector Classification,

as presented in Chang and Lin (2011). Suppose we are given a set of l training vectors $\mathbf{x}_i \in \mathbb{R}^n$ (in some n dimensional space). Each of these vectors is associated to one of two classes. Let us indicate these classes by the "indicator" vector $\mathbf{y} \in \mathbb{R}^l$ such that $y_i \in \{-1, +1\}$ (note that this means that each class is represented by either a $-$ or a $+$). Now let us suppose that we want to define a hyperplane that separates these two classes. This hyperplane can be parametrized as follows:

$$y = \mathbf{w}^T\mathbf{x} + \mathbf{b}, \tag{6.1}$$

where $\mathbf{w}^T$ represents the transpose of the parameter vector $\mathbf{w}$ representing the slope of the plane in each of the coordinate directions and $\mathbf{b}$ being the bias vector which moves the plane away from the origin. As mentioned earlier SVMs try to position this plane such that the "margin" is maximized, which ensures that the surface defined by $\mathbf{w}^T\mathbf{x} + \mathbf{b} = \mathbf{0}$ has the largest separation from either of the two classes. This situation is indicated pictorially in Figure 6.1a. To obtain the ideal margin (m) we need to maximize the distance from $\mathbf{q}_1$ to $\mathbf{q}_2$, i.e., we need to find the parameters $\mathbf{w}$ and $\mathbf{b}$ for which $m = \|\mathbf{q}_1 - \mathbf{q}_2\|$ is maximal. This is achieved when

$$\begin{cases} y_1 &= \mathbf{w}^T\mathbf{q}_1 + \mathbf{b} = +1 \\ y_2 &= \mathbf{w}^T\mathbf{q}_2 + \mathbf{b} = -1. \end{cases} \tag{6.2}$$

Subtracting the above equations and reshuffling one obtains $\mathbf{w}^T(\mathbf{q}_1 - \mathbf{q}_2) = 2$. One can now divide both sides by $\|\mathbf{w}\|$, and noting that $\frac{\mathbf{w}^T}{\|\mathbf{w}\|}$ is a unit vector that is always parallel to $\mathbf{q}_1 - \mathbf{q}_2$, one obtains the following condition:

$$m \equiv \|\mathbf{q}_1 - \mathbf{q}_2\| = \frac{2}{\|\mathbf{w}\|}. \tag{6.3}$$

This is the equation the SVM tries to maximize. Identically one can try to minimize $\frac{\mathbf{w}^T\mathbf{w}}{2}$, yielding the following optimization problem:

$$\begin{aligned} \min_{\mathbf{w}} \quad & \frac{\mathbf{w}^T\mathbf{w}}{2}, \\ \text{subject to} \quad & y_i\left(\mathbf{w}^T\mathbf{x}_i + \mathbf{b}\right) \geq 1, \quad i = 1, \ldots, l. \end{aligned} \tag{6.4}$$

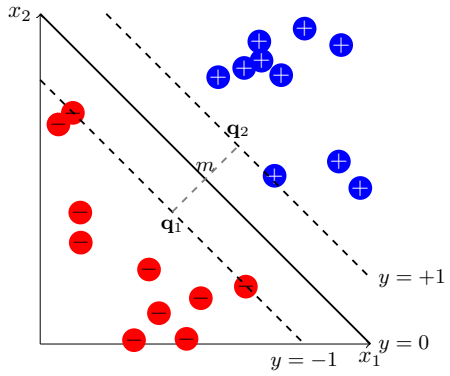
The constraint in the above definition ensures that everything is classified correctly since a mismatch between the sign of the true label (y_i) and the predicted label ($\mathbf{w}^T\mathbf{x}_i + \mathbf{b}$) of the i^{th} sample will lead to the left side of the inequality becoming negative. Due to the potentially high dimensionality of the vector variable $\mathbf{w}$, as it is intrinsically linked to the dimensionality of $\mathbf{x}_i$, it is usually more convenient to solve the following dual problem:

$$\begin{aligned} \operatorname*{argmin}_{\boldsymbol{\alpha}} \quad & W(\boldsymbol{\alpha}) = \min_{\boldsymbol{\alpha}} \left(\frac{1}{2}\sum_{ij} \alpha_i\alpha_j y_i y_j \mathbf{x}_i^T\mathbf{x}_j - \sum_i \alpha_i \right). \\ \text{Subject to} \quad & \alpha_i \geq 0, \quad \sum_i \alpha_i y_i = 0, \quad i, j = 1, \ldots, l, \end{aligned} \tag{6.5}$$

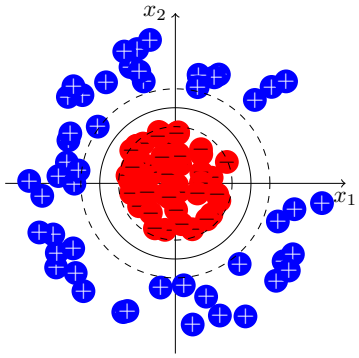
where $\mathbf{w}$ is given by

$$\mathbf{w} \equiv \sum_i y_i\alpha_i\mathbf{x}_i. \tag{6.6}$$

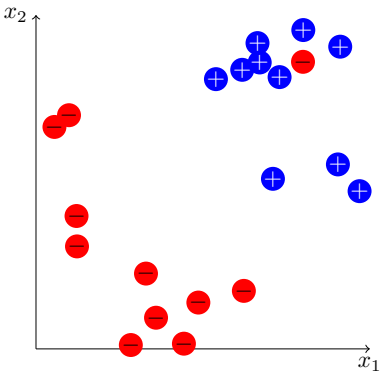
Once having obtained the values for $\mathbf{w}$, one can obtain $\mathbf{b}$ by plugging in known



(A) Illustration of the maximum margin hyperplane of an SVM separating the samples of two linearly married classes indicated by pluses and minuses. The margin is indicated by m and can be computed by calculating the distance between the points $\mathbf{q}_1$ and $\mathbf{q}_2$.



(B) Illustration of two classes that are non-linearly married.



(C) Illustration of two classes with an outlier.

FIGURE 6.1

values for $\mathbf{x}$ and y in equation (6.1) and solving for $\mathbf{b}$. The optimization problem formulated in terms of $\boldsymbol{\alpha}$ by equations (6.5) and $\mathbf{w}$ allows one to observe some interesting properties. Firstly, only samples for which the value of α_i is large contribute to $\mathbf{w}$. In practice it turns out that the samples which contribute most to $\mathbf{w}$ are the samples that lie closest to the margin. The coordinate vectors of these samples are therefore called "support vectors" (whence the name "Support Vector Machine"). Secondly, the optimization problem expressed by equation (6.5) incorporates a sense of similarity in the first term. Coordinate vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ that lie in the same direction contribute more to $W(\boldsymbol{\alpha})$ than coordinate vectors that do not point in the same direction unless they are of opposite label (i.e., $\operatorname{sgn}(y_i) \neq \operatorname{sgn}(y_j)$). This ensures that α_i should be larger when a point is close to samples of opposite label, as this will most effectively reduce the value of $W(\boldsymbol{\alpha})$.

Obviously the SVM introduced above can only separate classes that are linearly married (see, e.g., Figure 6.1a), where "married" means that one can draw a hyperplane between the two classes that separates them from each other. To make the SVM more generally applicable, e.g., to the case portrayed in Figure 6.1b, one may introduce a nonlinear mapping $\phi(\mathbf{x_i})$ that maps each sample $\mathbf{x_i}$ to a higher dimensional space. To accommodate this change of coordinates one can simply replace the values of $\mathbf{x}$ in equations (6.5) and (6.6) by $\phi(\mathbf{x})$. Since the mapping $\phi(\mathbf{x})$ can yield coordinate vectors of an arbitrary high dimension it is usually infeasible to compute

$\phi(\mathbf{x})$. Therefore, SVMs adopt something called the "kernel trick". This allows one to compute the kernel defined as $K(\mathbf{x}_i, \mathbf{x}_j) \equiv \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$, which is a scalar function, instead of $\phi(\mathbf{x}_i)$ and $\phi(\mathbf{x}_j)$ separately. Substituting for $\mathbf{x}$ in equations (6.5) and (6.6) and using the definition for the kernel function, one obtains the following optimization problem (for brevity I have decided to use vector notation instead):

$$\begin{aligned} \underset{\boldsymbol{\alpha}}{\text{argmin}} \quad & W(\boldsymbol{\alpha}) = \min_{\boldsymbol{\alpha}} \left(\frac{1}{2} \boldsymbol{\alpha}^T Q \boldsymbol{\alpha} - \mathbf{e}^T \boldsymbol{\alpha} \right) \\ \text{with} \quad & \mathbf{e}^T = (1, \ldots, 1), \quad Q_{ij} = y_i y_j K(\mathbf{x}_i, \mathbf{x}_j), \\ & K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j), \\ \text{subject to} \quad & \alpha_i \geq 0, \quad \mathbf{y}^T \boldsymbol{\alpha} = 0, \end{aligned} \tag{6.7}$$

and $\mathbf{w}$ is given by

$$\mathbf{w} \equiv \sum_i y_i \alpha_i \phi(\mathbf{x}_i). \tag{6.8}$$

One actually does not need to know the explicit form of $\phi(\mathbf{x})$ since $\mathbf{w}$ does not need to be computed directly. To obtain $\mathbf{b}$ one can simply rewrite $y = \mathbf{w}^T \phi(\mathbf{x}) + \mathbf{b} \implies \mathbf{b} = y - \mathbf{w}^T \phi(\mathbf{x})$ (where $\mathbf{x}$ and y are assumed to be known) in a way that uses the kernel instead by using equation (6.8):

$$\mathbf{b} = y - \sum_i y_i \alpha_i \phi(\mathbf{x}_i)^T \phi(\mathbf{x}) \equiv y - \sum_i y_i \alpha_i K(\mathbf{x}_i, \mathbf{x}) \tag{6.9}$$

and use:

$$\text{sgn}(\mathbf{w}^T \phi(\mathbf{x}) + \mathbf{b}) \equiv \text{sgn} \left(\sum_i y_i \alpha_i K(\mathbf{x}_i, \mathbf{x}) + \mathbf{b} \right) \tag{6.10}$$

to classify any new data point $\mathbf{x}$.

To illustrate what a kernel function could look like, let us consider the following example. We transform the coordinates $\mathbf{q}^T = (q_1, q_2)$ according to the mapping $\phi(\mathbf{q})^T = \left(q_1^2, q_2^2, \sqrt{2} q_1 q_2 \right)$. To construct the kernel function corresponding to this particular mapping, we need to compute $K(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x})^T \phi(\mathbf{y})$. Plugging in our expression for $\phi(\mathbf{q})$ yields the following:

$$\begin{aligned} K(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x})^T \phi(\mathbf{y}) &= x_1^2 y_1^2 + x_2^2 y_2^2 + 2 x_1 x_2 y_1 y_2 \\ &= (x_1 y_1 + x_2 y_2)^2 = (\mathbf{x}^T \mathbf{y})^2. \end{aligned}$$

Note that this kernel function looks very similar to what we used in the linear case, with the exception of the square. This kernel is a special case of the family of kernels which are known as the polynomial kernels, i.e., $K(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^T \mathbf{y} + c_0)^d$, where d is the degree of the polynomial and c_0 is an arbitrary coefficient. There exist many more such kernels; those I use for SVM classification are

- a linear kernel, i.e., $K(\mathbf{x}, \mathbf{y}) = \mathbf{x}^T \mathbf{y}$;
- a polynomial kernel of degree 3, i.e., $K(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^T \mathbf{y})^3$; and
- the radial basis function, i.e., $K(\mathbf{x}, \mathbf{y}) = \exp\left(-\gamma \left\| \mathbf{x} - \mathbf{y} \right\|^2 \right)$, with γ being equal to the inverse of the variance in the training set multiplied by the total number of dimensions.

Until this point we have only looked at the case where there is a way to find a hyperplane that separates both classes completely. In real world datasets this is not always possible, as there can be outliers; as an example see Figure 6.1c. In this case one wants to separate the two classes with the minimum amount of error. To achieve this one needs to add a penalty term to the optimization problem. scikit-learn (Pedregosa et al., 2011) uses a squared L2 penalty for this with a regularization parameter $C > 0$ (I used its default value of $C = 1$).

The Multi-layer Perceptron Classifier (MLPC) of scikit-learn (Pedregosa et al., 2011) is a deep-neural-network classifier. As I use this classifier for multi-class classification, the output activation of this neural network is "softmax":

$$f(\mathbf{x})_i = \frac{e^{x_i}}{\sum_j x_j} \tag{6.11}$$

The sum in the denominator ensures that the output is normalized over all class labels such that the outputs of the neural network can be used as probabilities. Furthermore, I have chosen to use three dense hidden layers with sizes $[20, 10, 5]$ and ReLU activation (i.e., $f(x) = \max(0, x)$). For the optimization I use the Adam optimizer with a maximum number of 1000 iterations, `early_stopping=True` and a validation fraction of 25%.

The XGBoost ("Extreme Gradient Boosting") Classifier (Chen and Guestrin, 2016) is a random forest (RF) classifier. The main difference that sets apart XGBoost from other RF classifiers is that it does not generate multiple decision trees *independently* using a subset of the training data which are then combined together using e.g., a majority vote or by averaging to yield a prediction. Instead XGBoost uses a process called *boosting*. This boosting process generates several models sequentially with each subsequent model trying to correct the errors of the preceding model. The boosting algorithm used by XGBoost is called *gradient boosting* which uses the residual errors of the preceding model as labels for the subsequent model. The result of this process is an ensemble classifier that classifies with a higher accuracy than individual models. The version of XGBoost used in this work is version 1.5.2, as this version still supports automatic label-encoding similar to the label encoder used by scikit-learn (i.e., `sklearn.preprocessing.LabelEncoder`). Using this classifier I keep all the hyperparameters of the XGBoost classifier to their default, since I did not have the time nor expertise to set these myself.

6.2 Consolidation

In this section I discuss the different consolidation schemes I employ to consolidate the results from the classification results obtained using the CPz STAR, CPz GAL and CPz QSO datasets. These consolidation methods are the lowest-entropy method, the average-probability method, the alternative method and majority vote. The first two methods use *class probabilities* yielded by the classifiers, whereas the latter two methods use the *class labels* yielded by the classifiers to consolidate the results. The first two methods are far more versatile, as they allow the user to set thresholds, which can be used to reject classification results that have a low level of certainty. This approach can artificially boost the precision of the classifier at the cost of a lower completeness.

6.2.1 Lowest-Entropy Method

Entropy can be used to quantify the degree of uncertainty in a probability distribution. To illustrate, let us say we have an object that we want to assign to one of the three classes A, B and C. The classifier is able to assign a probability of the object belonging to either of these three classes, i.e., p_A, p_B and p_C, respectively. These probabilities are normalized such that $p_A + p_B + p_C = 1$. Let us consider the first case where we have a uniform distribution of probabilities, i.e., $p_A = p_B = p_C = \frac{1}{3}$. This distribution is the least informative, i.e., given no extra information there is no way of assigning the object to any of the three classes without being biased. We could say that this classifier is unfit to classify this particular object. Now let us consider another classifier which gives us the following distribution of probabilities: $p_A = 1$ and $p_B = p_C = 0$. This classifier is confident that the object belongs to class A. Comparing the results of the first classifier to those of the second classifier, we are more inclined to believe the latter one, provided that the second classifier is not inherently biased. We can quantify this degree of belief in either of the two classifiers by using the Shannon entropy, which is formulated as follows:

$$H(X) = -\sum_{i=1}^{K} p_i \ln(p_i), \tag{6.12}$$

where X is a random variable, p_i is the probability of an object belonging to the i^{th} class and $i = 1, \ldots, K$. This formula expresses the average uncertainty in the distribution of possible outcomes. In the cases we considered, the entropy would amount to either $H_1(X) = \ln(3)$ or $H_2(X) = 0$. Thus the classifier with the lowest entropy is the classifier that is most confident about its classification result. One can leverage these results to consolidate a multitude of classification results by only selecting the one which has the lowest entropy in the distribution of class probabilities. In addition one can set an upper limit for the entropy that can be used to filter out samples for which no good classification result exists, at least not within the bounds of the specified threshold.

6.2.2 Average-Probability Method

This consolidation method simply averages the class probability distributions yielded by various classifiers to obtain a single probability distribution. This distribution can be used to make a prediction of the most likely class label corresponding to any given object. In addition one can set a probability threshold. If the maximum probability in the class probability distribution of a given sample drops below this threshold one can assign it to the post-consolidation outlier class. One can show that the consolidated distribution is still normalized. Let us suppose we have two distributions, i.e., (p_A, p_B, p_C) and (q_A, q_B, q_C) such that $p_A + p_B + p_C = 1$ and $q_A + q_B + q_C = 1$. Averaging the probabilities over the classes in either of the two distributions one obtains the following consolidated distribution: $(r_1, r_2, r_3) = \left(\frac{p_A+q_A}{2}, \frac{p_B+q_B}{2}, \frac{p_C+q_C}{2}\right)$, which is still normalized to 1, i.e., $r_1 + r_2 + r_3 = \frac{1}{2}(p_A + p_B + p_C + q_A + q_B + q_C) = 1$.

6.2.3 Alternative Method and Majority Vote

Both of these methods use class labels instead of probabilities. Suppose we have multiple classifiers each yielding various labels then there are two ways we can consolidate these results, i.e., the alternative method and majority vote. In both cases

we need to count the occurrence of different class labels. The alternative method basically assigns any sample for which there is a disagreement between the different classifiers to the post-consolidation outlier class. The majority vote method is less strict and only assigns samples to the post-consolidation outlier class when the vote is indecisive, i.e., there exists an equal split. In all the other cases the sample will be assigned to the class with the largest number of occurrences.

6.3 Results

Before consolidating the classification results yielded by the different datasets, let us have a look at how the different classifiers, i.e., KNN, Support Vector classifier (SVC), Neural Network Classifier (NNC) and the XGBoost Classifier (XGBC), perform. Figure 6.5 shows plots demonstrating the classification performance of each of these classifiers on the different projections obtained through the different sharpened LMDS-NNP (SLMDS-NNP), sharpened UMAP-NNP (SUMAP-NNP), sharpened *t*-SNE-NNP (StSNE-NNP), sharpened NPE-NNP (SNPE-NNP) models we trained previously. In addition to the aforementioned classifiers I also use a dummy classifier that assigns class labels randomly to obtain a baseline above which useful classifiers should lie. The results for the CPz GAL and CPz QSO datasets are provided as supplemental material in Appendix C. From the plot showing the recall in Figure 6.5 it is clear that the SVC linear classifier applied to the StSNE-NNP performs worse than the dummy classifier in terms of classifying stars. Furthermore, we can tell from these plots that the StSNE-NNP projection is generally unfit for any of the aforementioned classifiers to do classification when compared to performance of classifiers utilizing the other projection methods. The classifiers using either SLMDS-NNP, SUMAP-NNP or SNPE-NNP are generally on a par with each other in terms of precision, recall and F1 score, with the exception of the QSO classification task. In this case SNPE-NNP is the best in terms of precision and F1 score, and SLMDS-NNP is the best in terms of recall. Overall, SLMDS-NNP, SNPE-NNP and SUMAP-NNP in combination with either KNN, SVC_RBF, NNC or XGBC seem to yield the best results. In terms of runtime, KNN is the most scalable of these methods.

Taking into account the different neural network projection models and the different classifiers we can build a total of 24 unique classifiers for each dataset (discarding the dummy classifier). Presenting the decision boundaries and confusion matrices for each of these classifiers would take up too much space. Therefore I decided to only present the most important ones. The decision boundaries and confusion matrix of the sharpened LMDS-NNP based KNN classifier are given in Figure 6.2. Similar plots are provided in Appendix C for the CPz GAL and CPz QSO datasets. The results of these classifiers will be used in consolidation. Additionally, Figures 6.3 and 6.4 show the decision boundaries for the sharpened tSNE-NNP based SVC LINEAR and SVC POLY classifiers, respectively. These plots demonstrate that support vector classifiers using either linear or polynomial (degree 3) kernels do not perform well when classes are scattered or embedded into concave regions of another class. Other classifiers like KNN are much more robust against these kind of features. This is also evident from the performance metrics presented in Figure 6.5 where the support vector classifiers are usually outperformed by KNN, XGBC and NNC (especially for the sharpened tSNE-NNP embedding).

The results of consolidating the classifications yielded by the sharpened LMDS-NNP-based KNN classifiers applied to the CPz STAR, GAL and QSO datasets using

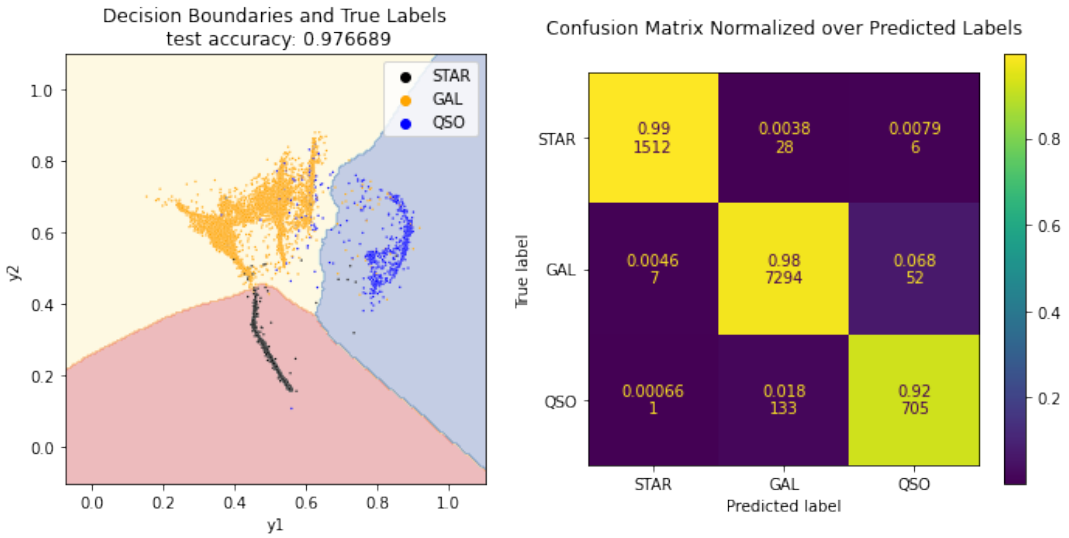


FIGURE 6.2: Figure showing the decision boundaries of the sharpened LMDS-NNP based KNN classifier of the CPz STAR dataset and its confusion matrix.

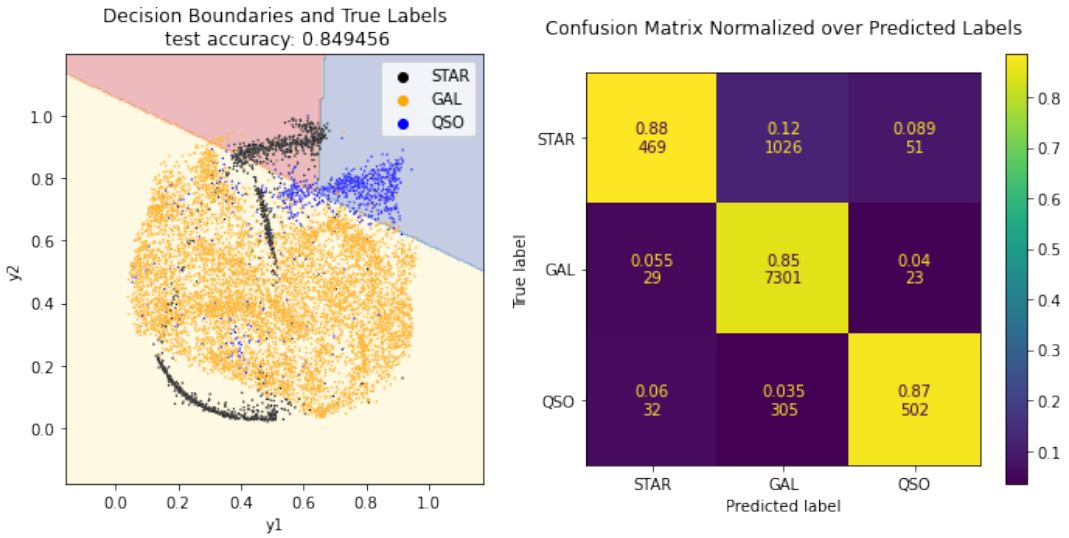


FIGURE 6.3: Figure showing the decision boundaries of the sharpened tSNE-NNP based linear SVM classifier of the CPz STAR dataset and its confusion matrix.

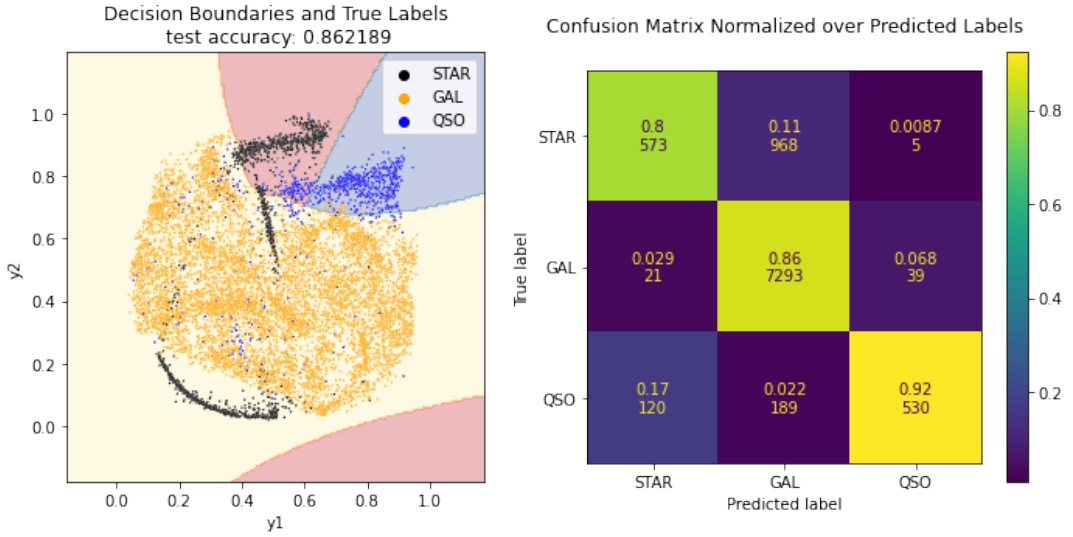


FIGURE 6.4: Figure showing the decision boundaries of the sharpened tSNE-NNP based polynomial SVM classifier (degree 3) of the CPz STAR dataset and its confusion matrix.

the various methods presented in Section 6.2 are shown in Figures 6.6, 6.7, 6.8 and 6.9. From these results we can make the following observations:

- First, the lowest-entropy consolidation method assigned most of the samples to the post-consolidation outlier class. This is mainly due to the chosen post-consolidation entropy threshold of 0.1, indicated by the red dashed line in Figure 6.6b. This threshold can easily be shifted to allow more samples to be classified instead of being assigned to the post-consolidation outlier class. Figure 6.6c clearly shows that most of these outliers lie along the edges where different clusters meet, which is as expected. From Figure 6.6a one can see that the post-consolidation outlier class is mostly dominated by galaxies. This is probably due to the fact that galaxies constitute roughly 76% of the full dataset. The large number of QSOs assigned to the outlier class is perhaps more interesting, since those only constitute roughly 9% of the full dataset; there will be more discussion of this issue in Chapter 8. From the entropy distributions in Figure 6.6b, one can see that the lowest-entropy method achieved the desired effect of finding the set of predicted probabilities with the lowest entropy. This is evident from the post-consolidation entropy distribution being more sharply peaked toward zero. The gray dashed lines in Figure 6.6b indicate the entropy corresponding to the case in which a sample can be assigned to two of the three classes with an equal probability of $1/2$, i.e., $H(X) = \log(2)$. The entropy threshold should preferably be lower than this value. Furthermore, the gray dotted lines in the figure indicate the case in which a sample can be assigned to either of the three classes with equal probability, i.e., $H(X) = \log(3)$. The classification performance metrics listed in Table 6.1 clearly show that while the lowest entropy method has lower values for accuracy and recall, it achieves a higher precision for the QSO class. This higher precision is due to the threshold, which ensures fewer sources are misclassified by assigning them to the outlier class. This feature artificially boosts the precision for the QSO class at the cost of a lower recall.

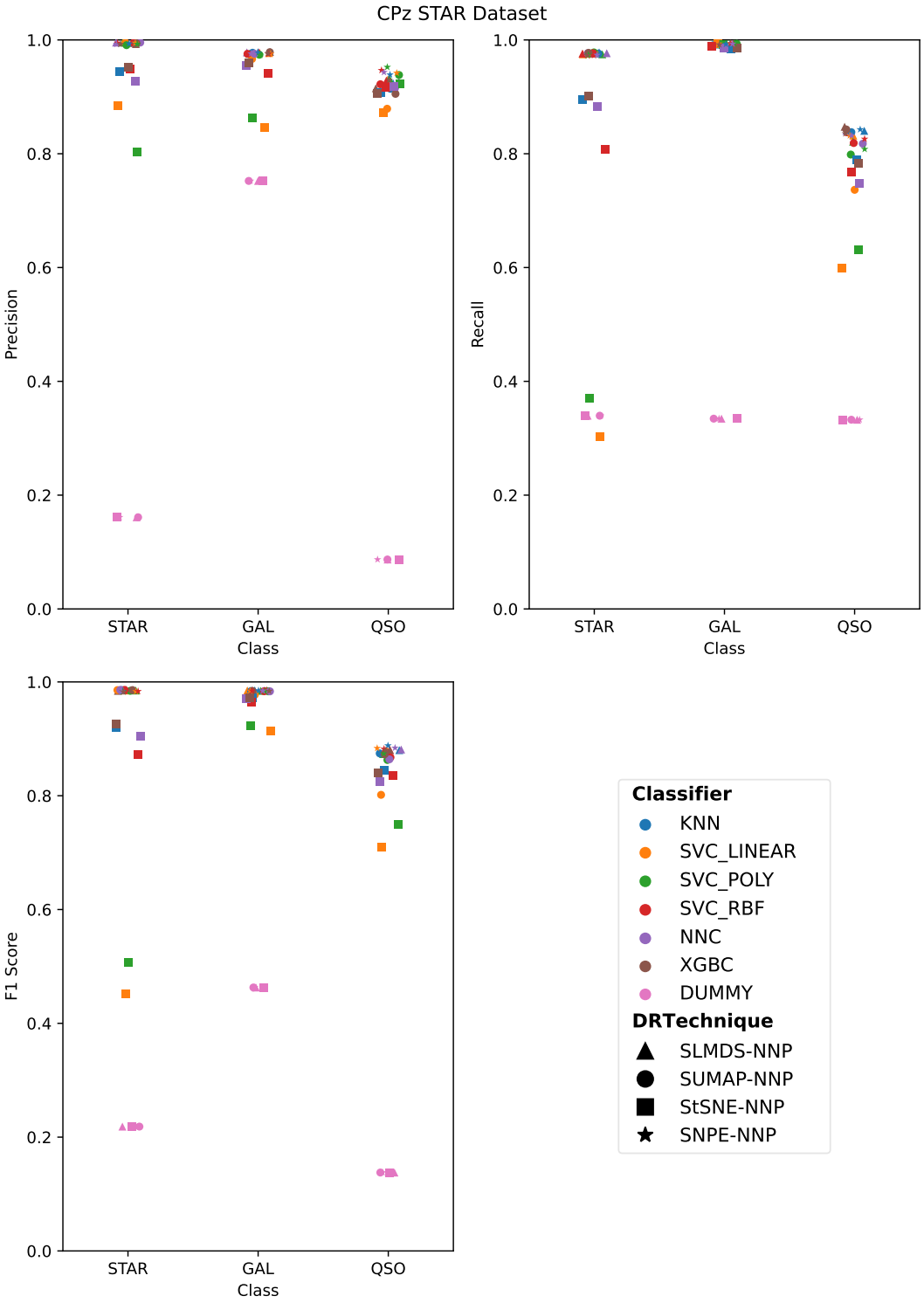


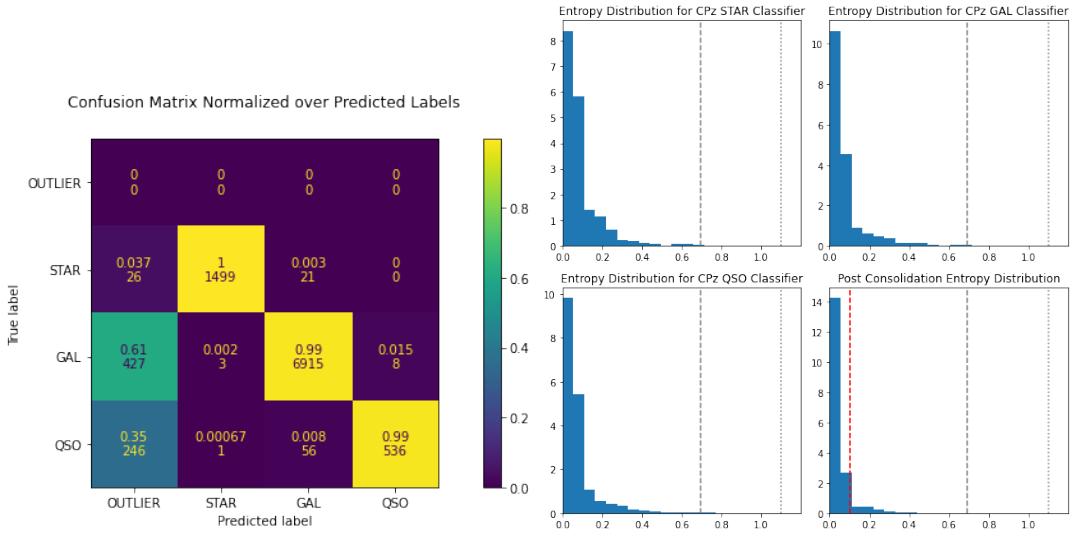
FIGURE 6.5: Various plots demonstrating the classification performance using the CPz STAR dataset in terms of precision, recall and F1 score for various combinations of DR technique and classifier. Note, the "DUMMY" classifier assigns classes randomly and gives a baseline above which any useful classifier should lie.

TABLE 6.1: Post-consolidation performance.

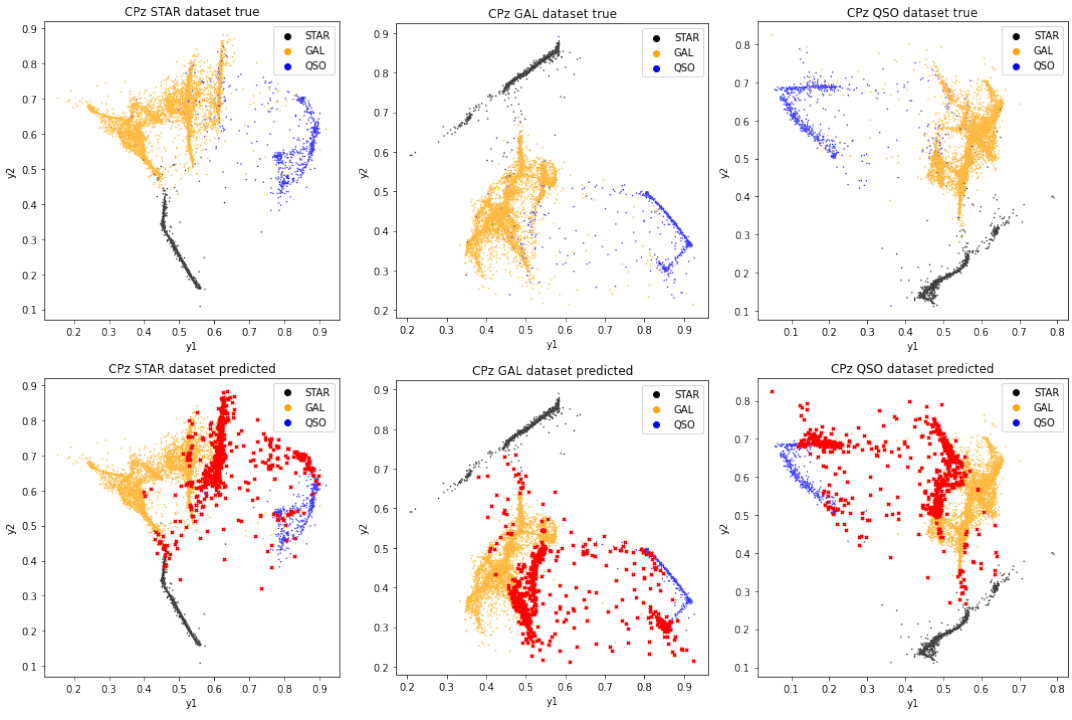
Consolidation Method	Accuracy	Class	Precision	Recall	F1 Score
Lowest Entropy	0.9191	STAR	0.9973	0.9696	0.9833
		GAL	0.9890	0.9404	0.9641
		QSO	0.9853	0.6389	0.7751
Average Probability	0.9793	STAR	0.9974	0.9761	0.9866
		GAL	0.9800	0.9932	0.9866
		QSO	0.9378	0.8629	0.8988
Alternative	0.9720	STAR	0.9973	0.9664	0.9816
		GAL	0.9843	0.9893	0.9868
		QSO	0.9561	0.8308	0.8890
Majority Vote	0.9795	STAR	0.9974	0.9754	0.9863
		GAL	0.9804	0.9931	0.9867
		QSO	0.9357	0.8677	0.9004

- Second, the average-probability method (with a maximum probability threshold of 0.5) assigns only one sample to the outlier class (see Figure 6.7). Similarly to the lowest-entropy method, this threshold can be adjusted to boost the precision of the classifier. Looking at Table 6.1, this method performs similarly to the majority-vote method in terms of classification performance, given the specified threshold.
- Finally, the alternative method seems to assign many samples to the post-consolidation outlier class, as evident from Figure 6.8. Identically to the lowest-entropy method, most of these outliers seem to be located on the interface between the different clusters. In this case the outliers seem to be spread more evenly over the different true classes when compared to the lowest entropy method results.

In conclusion, the alternative and majority vote consolidation methods leave no control in terms of which samples get assigned to the post-consolidation outlier class. This is unfavorable in cases where one wants a high precision at the cost of losing some samples. Furthermore, none of the consolidation methods seem to be able to reproduce the true labels in regions where the data clusters are mixed. I further discuss this in the Discussion.

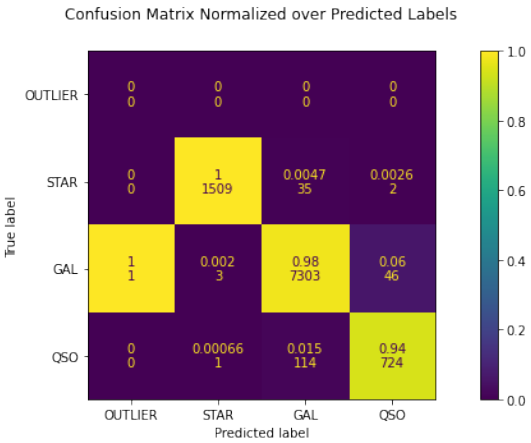


(A) Confusion Matrix (B) Entropy Distributions

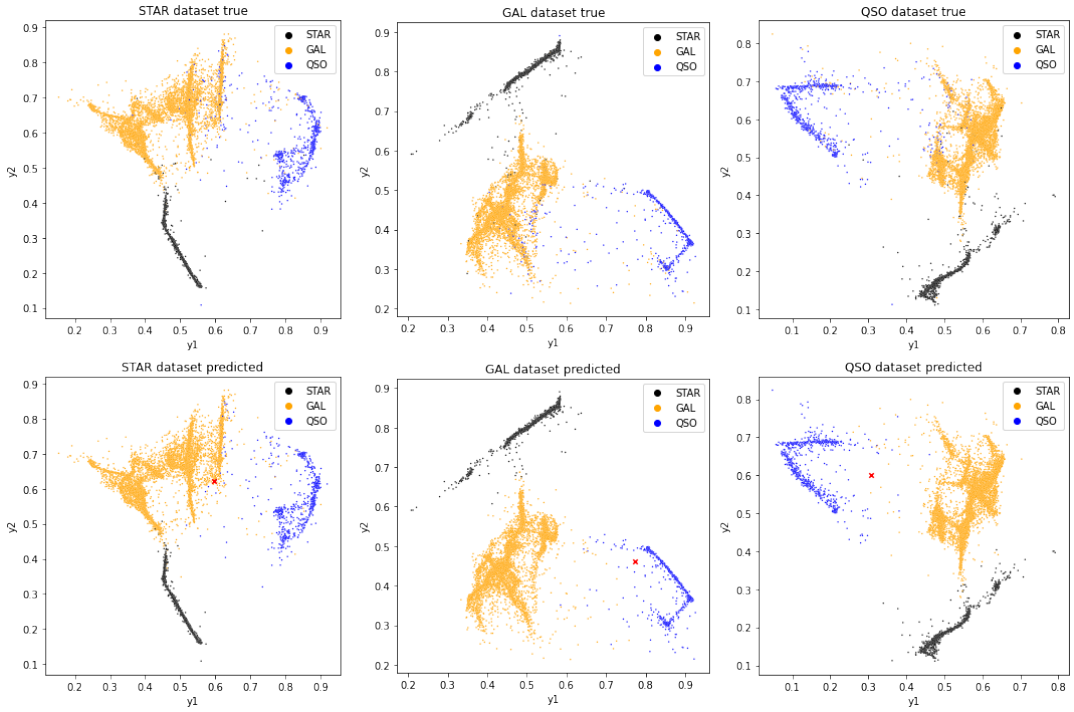


(C) Label Comparison Plots

FIGURE 6.6: Lowest entropy consolidation results.

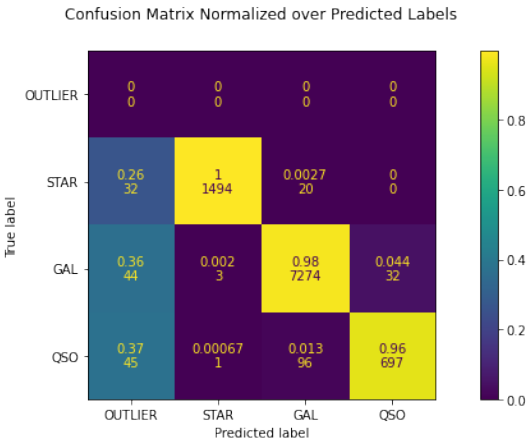


(A) Confusion Matrix

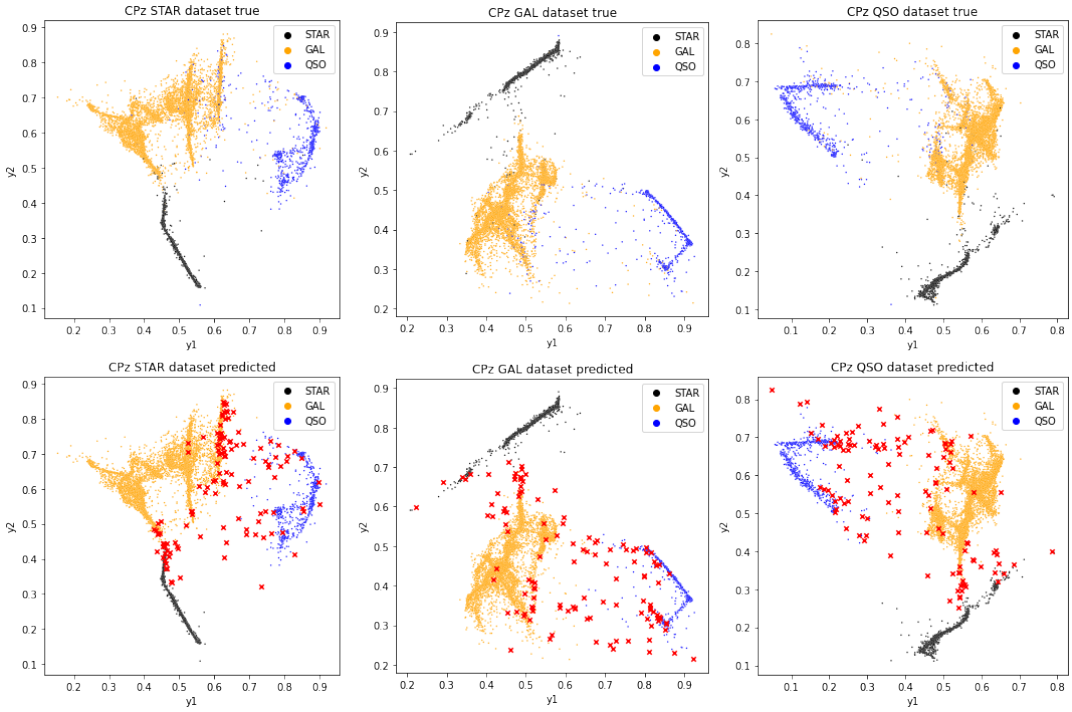


(B) Label Comparison Plots

FIGURE 6.7: Average probability consolidation results.

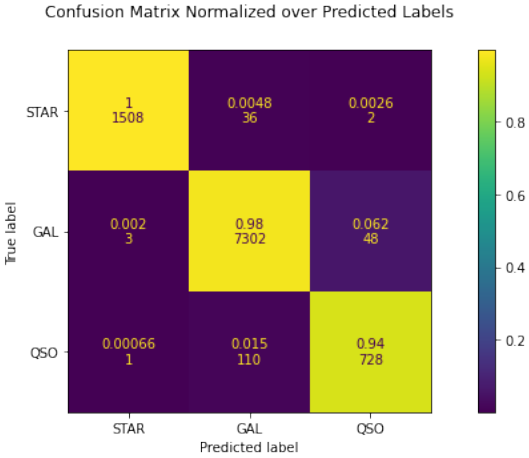


(A) Confusion Matrix

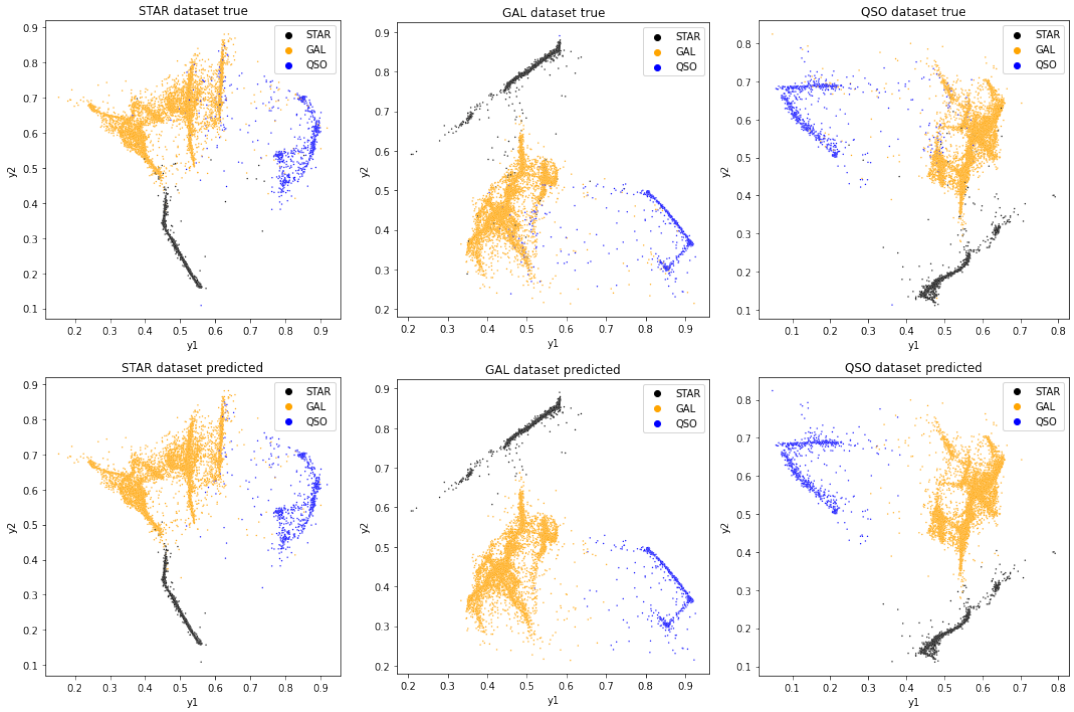


(B) Label Comparison Plots

FIGURE 6.8: Alternative method consolidation results.



(A) Confusion Matrix



(B) Label Comparison Plots

FIGURE 6.9: Majority vote consolidation results.

Chapter 7

Applications

In the previous chapters I have shown the steps involved in building an SDR-NNP aided classifier for star, galaxy and QSO classification. In this chapter I investigate whether the SDR projections convey any extra information besides samples belonging to one of the star, galaxy and QSO classes. In Chapter 4 we have seen that many of the SDR projections manifest some degree of oversegmentation with respect to the star, galaxy and QSO classes. Many of the labeled data points are divided into small subclusters, which gives rise to the question whether those subclusters convey relevant information or whether these are just a feature caused by the sharpening step. In this chapter I try to answer this question by focusing on the sharpened LMDS projection shown in Figure 4.7.

Each of the different sections in this chapter focus on different subsets of the dataset. In Section 7.1 I investigate the oversegmentation of the stellar data by looking at its relation to color, magnitude, effective temperature (T_{eff}) and surface gravity ($\log(g)$). In Section 7.2 I investigate the oversegmentation of the galaxy data by looking at its relation to morphological type, star formation rate (SFR) and redshift. And finally in Section 7.3 I investigate the oversegmentation of the QSO data by looking at its relation to redshift.

7.1 Stellar data

I begin by plotting color–magnitude diagrams (CMDs) to understand whether the stellar subclusters in the projection are physical clumps or an unphysical oversegmentation. The CMDs for the sample are shown in Figure 7.1 for $(g - r, g)$ and $(g_3 - r_3, g_3)$. The eight different stellar subclusters are color-coded. The CMDs appear very smeared out in the y – luminosity – direction. This is because the CPz dataset only contains apparent magnitudes, which are distance dependent. However, from the $(g_3 - r_3, g_3)$ diagram, which uses 3″ aperture magnitudes, we can already see that the stars in each of the different subclusters seems to have a different range of colors and hence different effective temperatures, since there is a (nearly) one-to-one relation between color and effective temperature.

By cross-matching with the astrophysical parameters dataset of Gaia DR3 (Gaia Collaboration et al., 2016; Gaia Collaboration et al., 2022), generated from Gaia data using the GSP-Phot module in the Apsis (Astrophysical parameters inference system) pipeline (Creevey et al., 2022), we can visualize whether the stellar subclusters present in Figure 4.7 convey any astrophysical information. From the Hertzsprung–Russell (HR) diagram and effective temperature versus surface gravity plots in Figure 7.2, we can see that the subclusters not only convey temperature information but also the spectral type of the stars by noticing the shift in temperature of stars

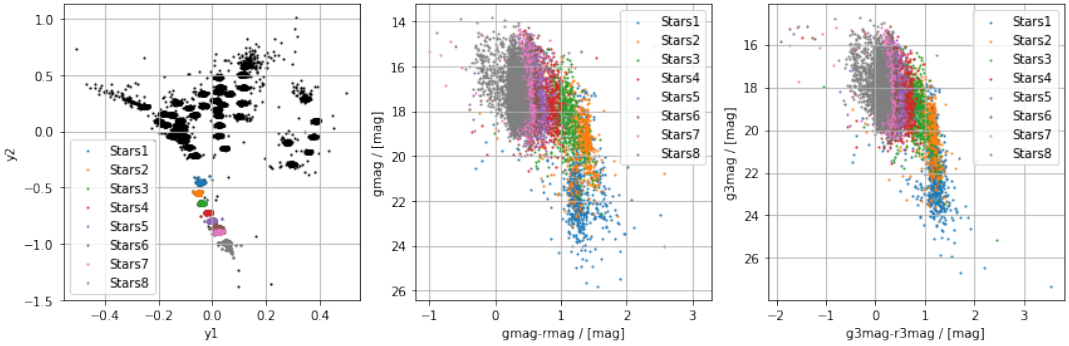


FIGURE 7.1: Color–magnitude diagrams (CMDs) of the various subclusters within the stellar class, color-coded by subclump. The left-most plot shows how the clusters were color coded. The middle plot shows the CMD using total magnitudes. The right-most plot uses 3″-aperture magnitudes.

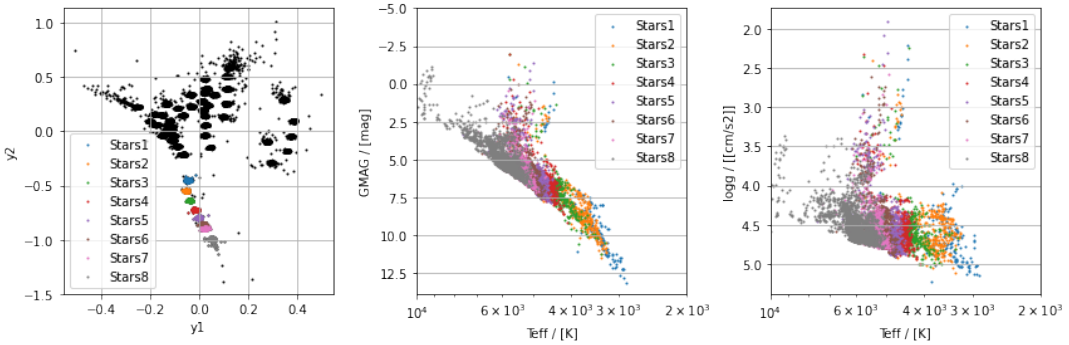


FIGURE 7.2: Hertzsprung-Russell (middle) and effective temperature versus surface gravity (right) diagrams. The color coding for the different stellar subclusters is provided by the left-most plot.

within the same subcluster as we move up in magnitude and down in surface gravity. The vertical branch starting around 6000 K in both of these plots is the giant branch. Since LMDS is a distance-preserving dimensionality-reduction technique, we may also note from Figure 4.7 that stars with spectral type M have colors that most closely match those of galaxies. I discuss this further in Section 7.2.

The CM diagrams, HR diagram and the effective temperature versus surface gravity diagram all seem to show that the sharpening step in SDR has oversegmented the stellar data, since the distribution of broadband colors and effective temperatures shown in Figure 7.2 form a continuum. However, this exercise shows that the projection still retains important physical information, even if it is (perhaps unnecessarily) oversegmented.

7.2 Galaxy data

Next I try to understand the distribution of galaxies in the SDR projection. To get information on morphological types of the galaxies in the sample, I cross-matched the galaxies in the CPz dataset using their angular positions with the Galaxy Zoo 1 (GZ1) data release (Lintott et al., 2008; Lintott et al., 2010). The GZ1 data contains morphological classifications of nearly 900000 galaxies from SDSS data release 6 and 7 (Adelman-McCarthy et al., 2008; Abazajian et al., 2009) classified by hundreds of

thousands of volunteers. The task of each volunteer was to classify each of object into one of six categories. The categories are

1. elliptical (likely also includes lenticular (S0) galaxies);
2. clockwise spiral galaxies;
3. anti-clockwise spiral galaxies;
4. some other kind of spiral galaxy (e.g. edge-on);
5. star or Unknown; and
6. merger.

The votes for each object were subsequently combined into fractions which can be used for further study.

Furthermore, Lintott et al. (2010) used the techniques described by Bamford et al. (2009) to remove the bias introduced by the survey limits of SDSS. The survey limits can cause small, faint or distant galaxies to be misclassified as elliptical galaxies due to spiral arms not being visible in SDSS images. To alleviate this effect, Bamford et al. (2009) devised a technique to estimate this and correct for it by assuming the morphological fraction within bins of fixed galaxy size and luminosity to be constant in redshift. Since redshift is a required parameter for this technique objects needed to be spectroscopically observed by SDSS. Lintott et al. (2010) supplemented the redshifts provided by SDSS DR6 with those provided by DR7 which meant that 92% of the objects in the main galaxy sample of GZ1 had spectroscopic redshifts. For the sake of completeness I now briefly outline the debiasing technique described in Appendix A of Bamford et al. (2009). The debiasing procedure starts with the assumptions that the fraction of early-type to spiral galaxies in the size versus luminosity space is constant in redshift, and that low-redshift samples should be least biased by the survey limits of SDSS. Therefore, Bamford et al. (2009) divided the full sample in bins of similar luminosity, size and redshift. Subsequently for each bin in the luminosity versus size space, one can find the lowest redshift bin containing at least 30 galaxies and assume that using this bin one can compute the "true" early-type to spiral galaxy ratio. Using this result, one can fit an empirical function to the early-type to spiral ratio as a function of luminosity and size. The function proposed by Bamford et al. (2009) is given by equation (7.1):

$$\frac{n_{\mathrm{el}}}{n_{\mathrm{sp}}} = \frac{p_1}{1+\exp\left(\frac{s_1(R_{50})-M_r}{s_2(R_{50})}\right)} + p_2, \tag{7.1}$$

$$\begin{aligned} \text{with} \quad & s_1\left(R_{50}\right) = q_1^{-\left(q_2+q_3 R_{50}^{q_4}\right)} + q_5 \\ \text{and} \quad & s_2\left(R_{50}\right) = r_1 + r_2\left(s_1\left(R_{50}\right) - q_5\right). \end{aligned}$$

Here M_r is the absolute r-band magnitude of the galaxy, R_{50} is the radius containing 50% of the Petrosian flux of the galaxy and $\{p_1, p_2, q_1, q_2, q_3, q_4, q_5, r_1, r_2\}$ are free parameters. Using the baseline estimates for the early-type to spiral ratios, one can define the following correction factor:

$$C(M_r, R_{50}, z) = \log_{10}\left(\frac{\left\langle n_{\mathrm{el}}/n_{\mathrm{sp}}\right\rangle_{\mathrm{base}}}{\left\langle n_{\mathrm{el}}/n_{\mathrm{sp}}\right\rangle_{\mathrm{raw}}}\right), \tag{7.2}$$

where the angular brackets indicate taking the average over the bins in the (M_r, R_{50}, z) space. With this correction factor one can compute the bias-adjusted early-type to spiral ratio from the raw vote shares for each galaxy expressed as fractions (p):

$$\left(\frac{p_{el}}{p_{sp}}\right)_{\text{adj}} = \left(\frac{p_{el}}{p_{sp}}\right)_{\text{raw}} 10^{-C(M_r, R_{50}, z)}. \tag{7.3}$$

The individual bias adjusted fractions, i.e., $p_{\text{el,adj}}$ and $p_{\text{sp,adj}}$ can be computed using the following formulae:

$$\begin{aligned} p_{\text{el,adj}} &= \frac{p_{\text{el,adj}}}{p_{\text{el,adj}} + p_{\text{sp,adj}} + p_{\text{x,adj}}} = \frac{1}{1 + 1 \Big/ \left(\frac{p_{\text{el}}}{p_{\text{sp}}}\right)_{\text{adj}} + \left(\frac{p_{\text{x}}}{p_{\text{el}}}\right)_{\text{adj}}} \\ &= \frac{1}{1 + 1 \Big/ \left(\frac{p_{\text{el}}}{p_{\text{sp}}}\right)_{\text{adj}} + \left(\frac{p_{\text{x}}}{p_{\text{el}}}\right)_{\text{raw}}} \end{aligned} \tag{7.4}$$

$$\begin{aligned} p_{\text{sp, adj}} &= \frac{p_{\text{sp,adj}}}{p_{\text{el,adj}} + p_{\text{sp,adj}} + p_{\text{x,adj}}} = \frac{1}{\left(\frac{p_{\text{el}}}{p_{\text{sp}}}\right)_{\text{adj}} + 1 + \left(\frac{p_{\text{x}}}{p_{\text{sp}}}\right)_{\text{adj}}} \\ &= \frac{1}{\left(\frac{p_{\text{el}}}{p_{\text{sp}}}\right)_{\text{adj}} + 1 + \left(\frac{p_{\text{x}}}{p_{\text{sp}}}\right)_{\text{raw}}} \end{aligned} \tag{7.5}$$

where $p_{\text{x}} = 1 - p_{\text{el}} - p_{\text{sp}}$ and is the fraction of objects assigned to categories 5 and 6 listed above. This debiasing procedure has a number of caveats, as mentioned by Lintott et al. (2010). It requires a homogeneous distribution of a substantial amount of galaxies, i.e., at least 30, to be present in each bin in the (M_r, R_{50}) space at low redshifts. This limits the debiasing procedure to objects with reliable r-band magnitudes, redshifts between 0.001 and 0.25 and absolute magnitudes and sizes that are not outliers from the normal galaxy distribution.

The cross-matched debiased Galaxy Zoo classifications are given in Figure 7.3. The classifications are color-coded as fractions of objects assigned to any of the spiral (p_{sp}), elliptical (p_{el}) or otherwise (i.e. star, merger or unknown) p_{x} categories. For convenience I also include a plot showing the redshifts of the various objects. These redshifts can be used to verify the validity of the classifications. From Figure 7.3 there is no indication that many of the subclusters within the galaxy class produced by SDR convey anything meaningful about morphological type of the galaxies. However, there seems to be a subcluster that contains predominantly spiral galaxies, as indicated in Figure 7.3. We can also note that there appears to be a redshift gradient, with many of the low-redshift galaxies situated on the left side of the main galaxy cluster.

In addition to inspecting the morphologies of the galaxies in the various galaxy subclusters produced by SDR, I examine the star-formation rate (SFR), stellar mass and dust luminosity of the various galaxies in the CPz dataset. I obtain these parameters by cross matching galaxies in the CPz dataset with a catalog of stellar masses and star formation rates by Chang et al. (2015). The specific SFRs (i.e., the SFR per unit stellar mass of the galaxy), stellar masses and dust luminosities in this dataset were obtained by fitting SEDs to the optical and mid IR spectra obtained by SDSS and WISE, respectively, and are displayed in Figure 7.4. The stellar masses (top-right panel) vary slightly from subcluster to subcluster. The dust luminosity plot (bottom-left) shows a slight gradient. Galaxies with a low dust luminosity are closer to stars

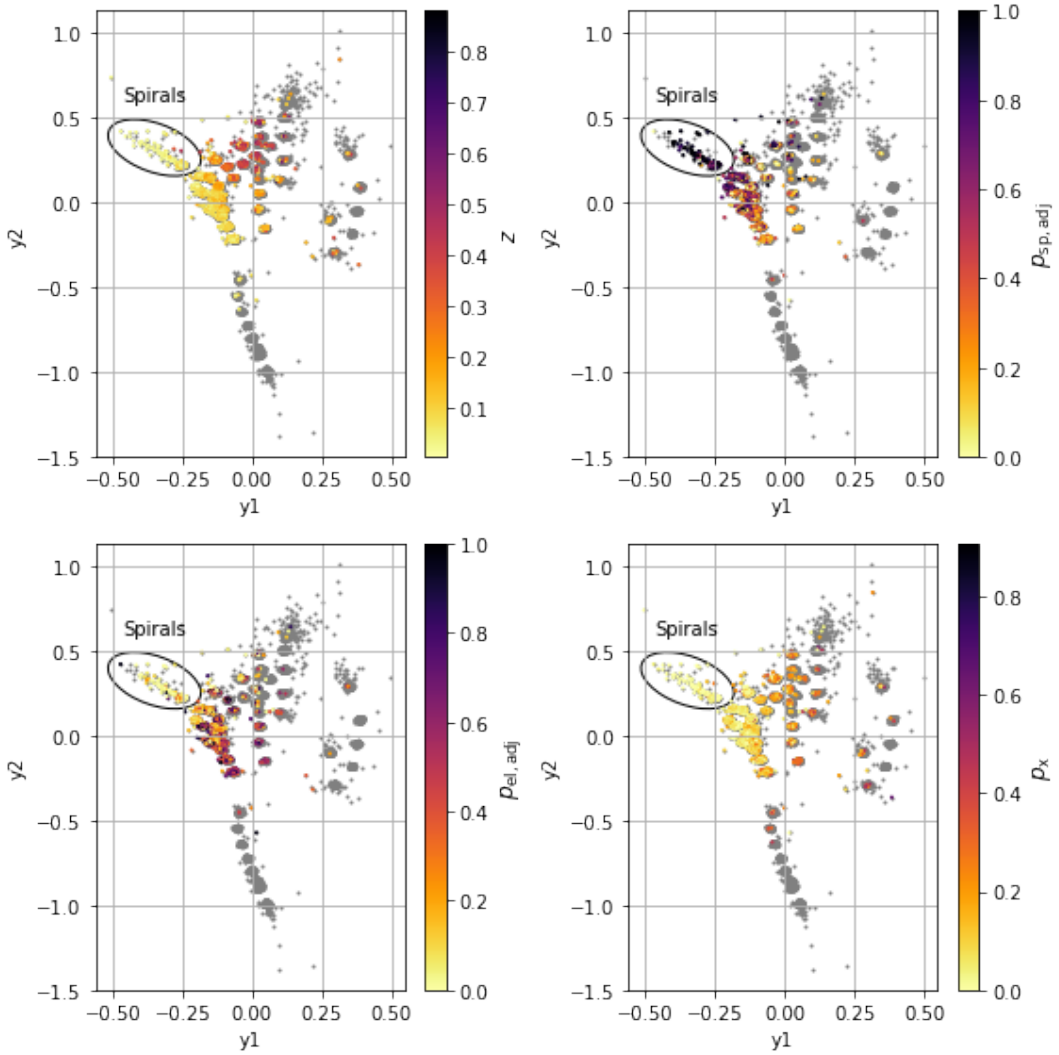


FIGURE 7.3: Plots of the the CPz STAR dataset projected using sharpened LMDS cross-matched with the Galaxy Zoo 1 (GZ1) classifications. The classifications are color-coded and given as debiased vote fractions following the debiasing technique of Bamford et al. (2009). The top-left plot indicates the redshift of the galaxies in GZ1. For reliable debiasing of the classifications, the redshift should be in the range of $0.001 - 0.25$ (Lintott et al., 2010).

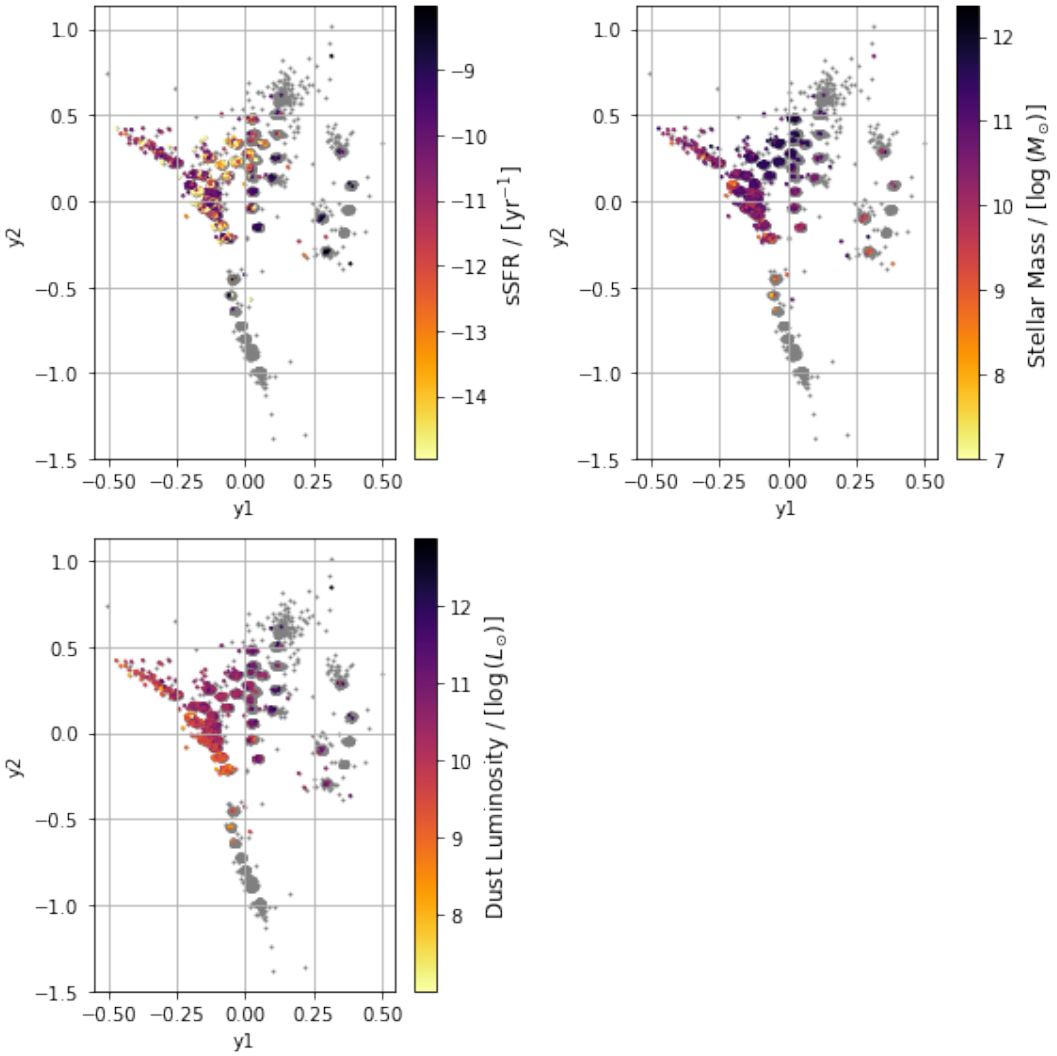


FIGURE 7.4: Plots of the CPz STAR dataset projected using sharpened LMDS cross-matched with a catalog of star formation rates, stellar masses and dust luminosity of various galaxies composed by Chang et al. (2015).

in the SDR projection. There is no coherent structure in the distribution of specific SFRs (top-left) in the projection.

A striking observation that arises when comparing Figure 7.3 with Figure 7.2 is that M stars most closely resemble elliptical galaxies. A potential explanation for this will be given in the Discussion. This observation is motivated by the fact that LMDS is a distance-preserving dimensionality-reduction method. This implies that the 2D projection should preserve the distances between the color coordinates in the high dimensional space. Coupled with the sharpening step discussed in Section 4.1 I have shown that sharpened LMDS still preserves distances reasonably well. This is evident from the value for the Shepard Goodness given in Table 4.8. In addition we can visually inspect the Shepard diagram given in Figure 7.5 to confirm that, while scaled, the point-wise distances are relatively well-conserved. The properties of these ellipticals are summarized in Figure 7.6. In this figure I show histograms of the morphological classes, redshifts, specific SFR, stellar mass and dust luminosity

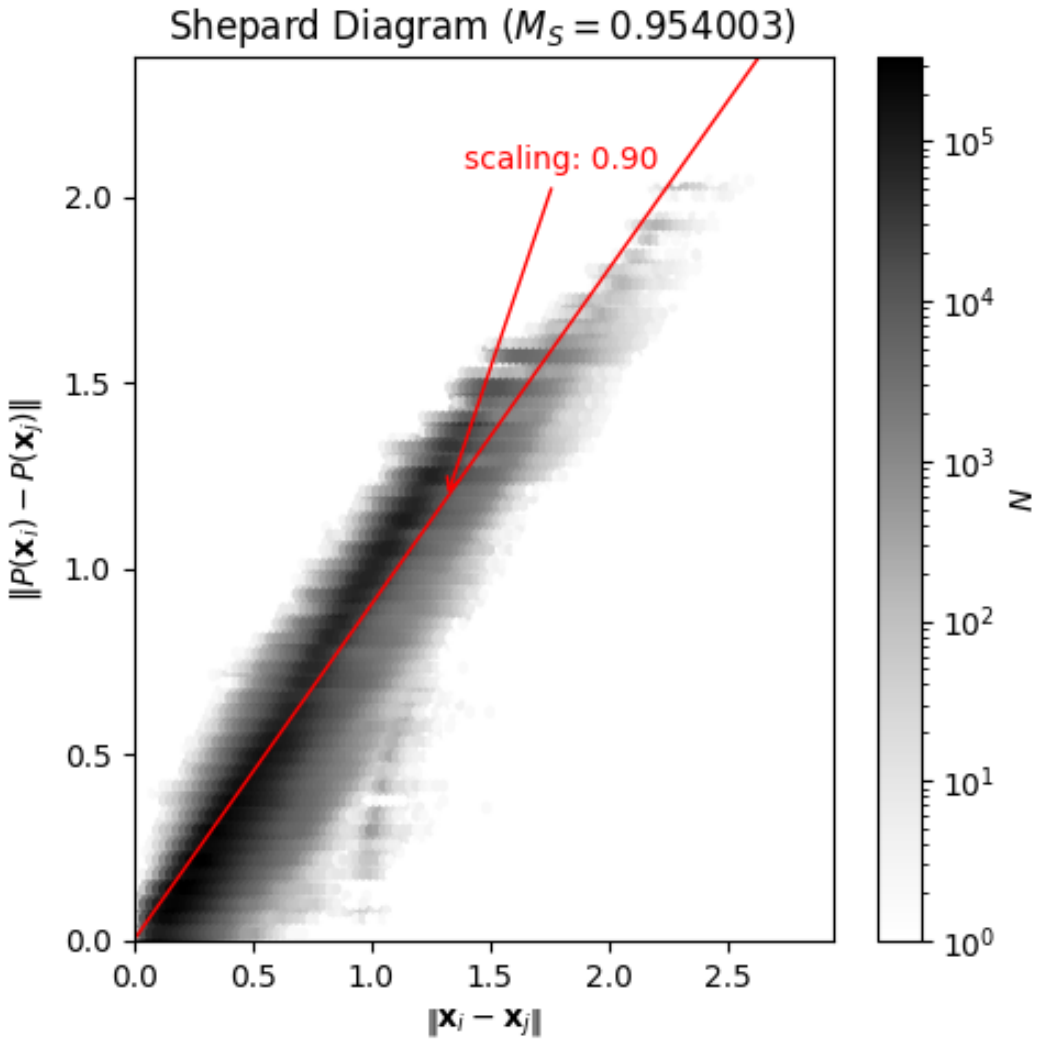


FIGURE 7.5: Shepard diagram of the CPz STAR dataset when projected using sharpened LMDS.

for the galaxy subcluster that is closest to the M star subcluster. From these histograms it is evident that most of these galaxies are ellipticals with a redshift of around 0.08. Furthermore, they have a specific SFR lower than 10^{-11} yr^{-1}, i.e., they are quiescent. Additionally, they have stellar masses between $10^{10} M_{\odot}$ and $10^{11} M_{\odot}$, which is typical for elliptical galaxies, and a relatively low dust luminosity of around $10^{9} L_{\odot}$.

7.3 QSO data

For the QSO sample I only inspect the redshifts, as shown in Figure 7.7. In the top-left plot in this figure, there is a redshift gradient with most of the low-redshift QSO's ($z \lesssim 1$) overlapping with the galaxy cluster. Additionally, the shift in redshift between the various QSO subclusters can be further visualized in the bottom-left histogram using the color-coding presented in the top-right plot. In other words the subclusters in the projection clearly separate various QSO samples in terms of redshift.

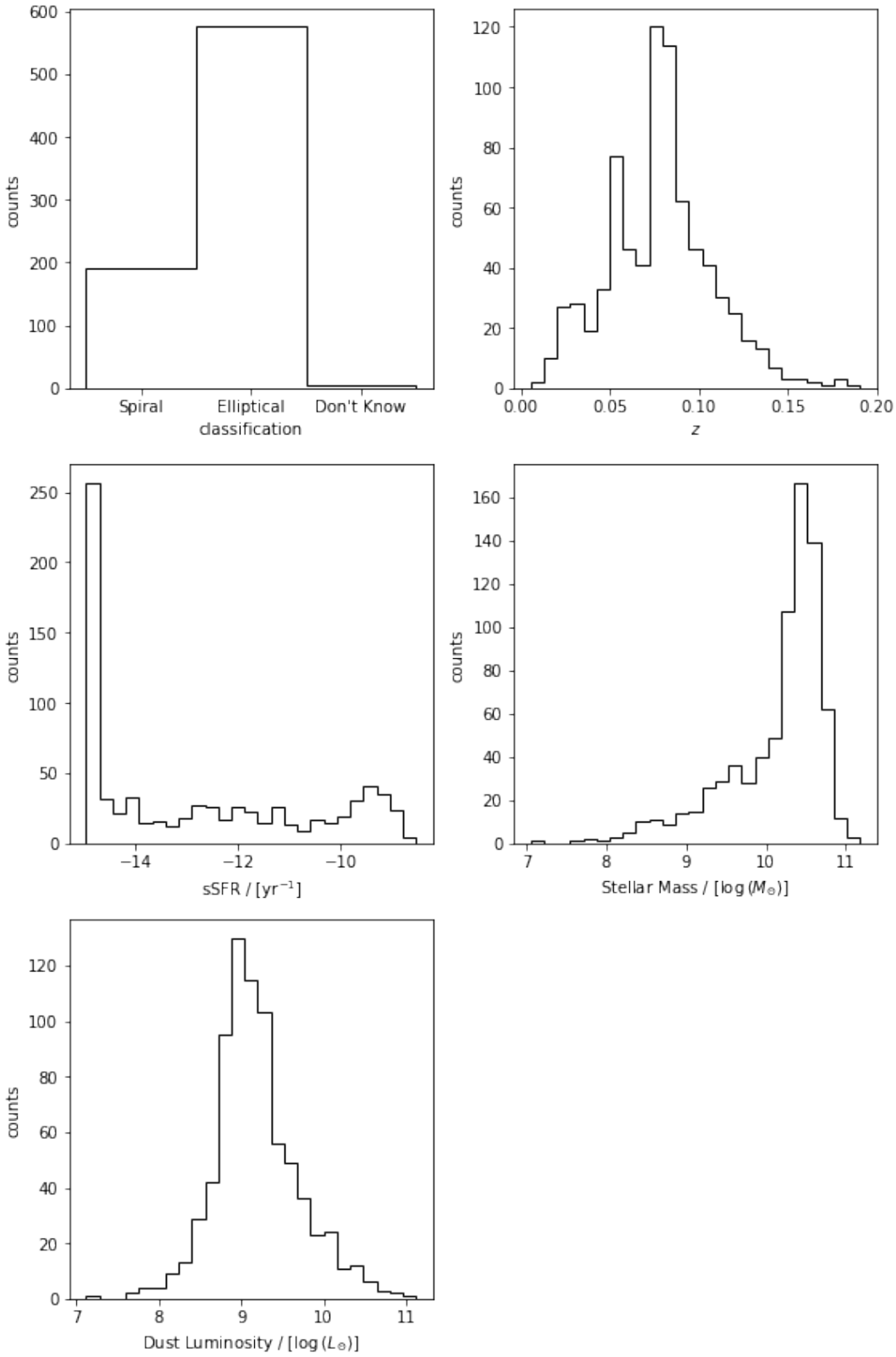


FIGURE 7.6: Histograms summarizing the properties of the galaxies which show most resemblance to M stars in terms of color. The histograms are comprised only of galaxy samples part of the galaxy subcluster that is closest to the subcluster containing M stars in Figure 7.2.

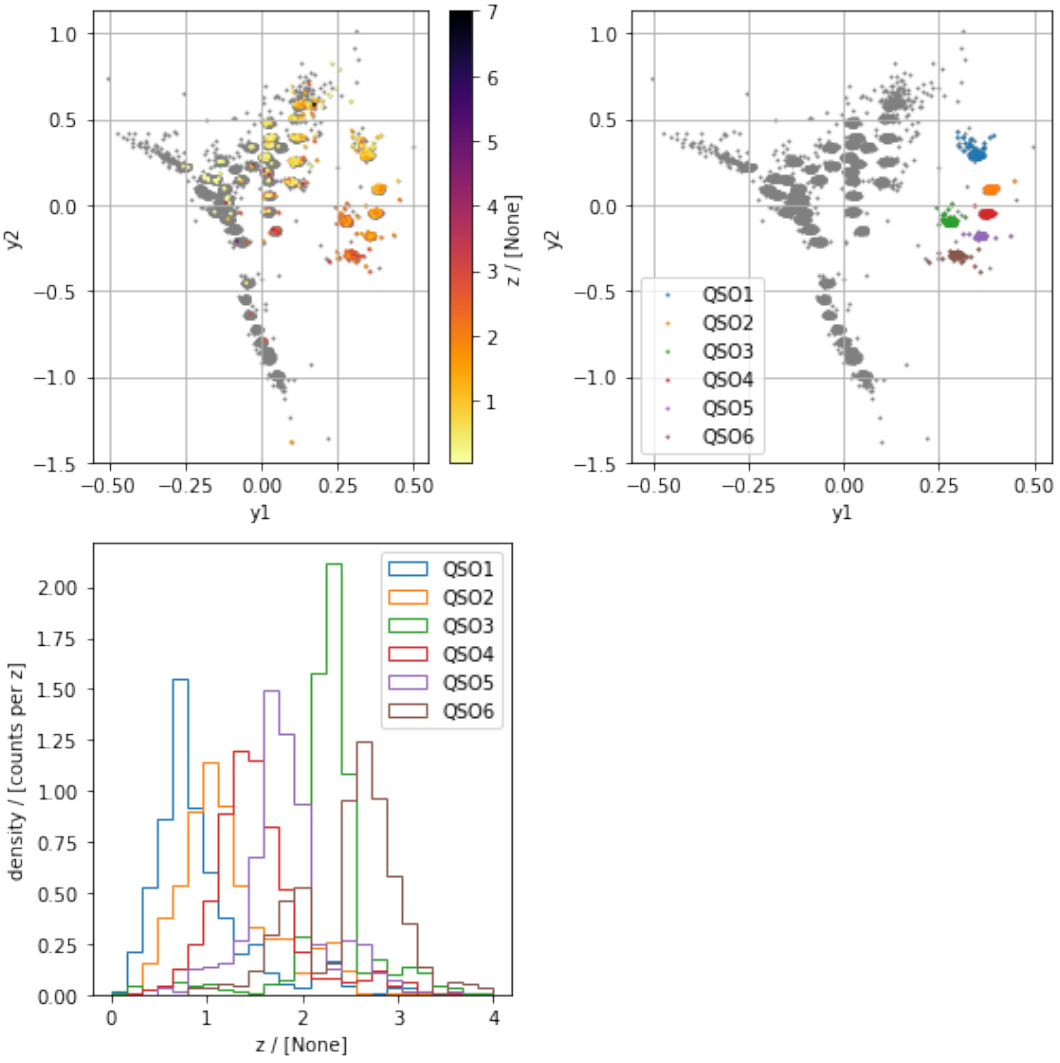


FIGURE 7.7: The top-left plot shows a scatter plot of the CPz STAR dataset projected using sharpened LMDS with the redshifts of the various QSO's color coded. In the top-right plot I show the color coding of the various QSO subclusters which is used in the bottom-left histogram.

Chapter 8

Discussion

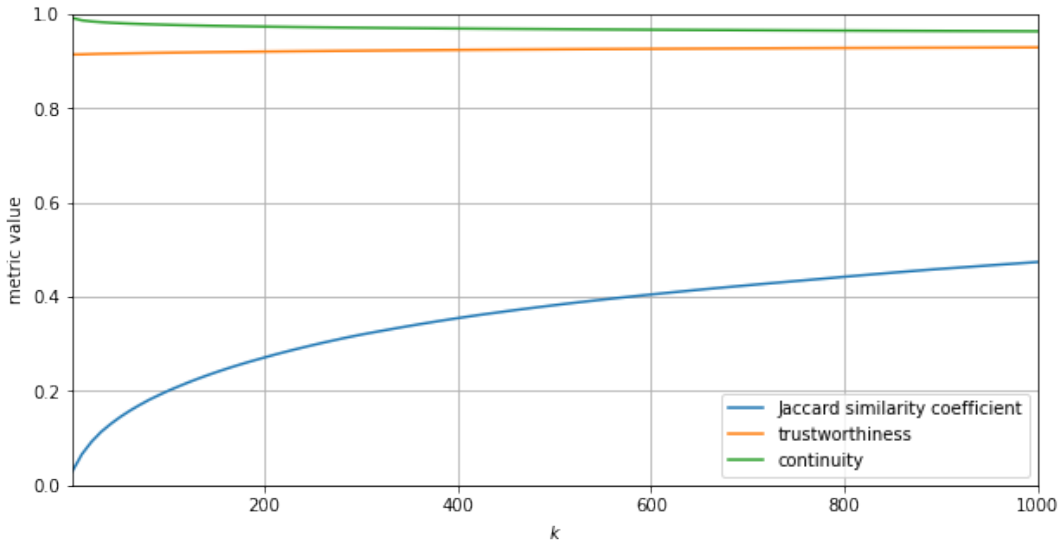
In this thesis I try to answer the question whether broadband colors can be used to classify stars, galaxies and QSOs. Specifically, I look at the merit of sharpened dimensionality reduction (SDR) aided classification to achieve this. The idea behind this approach was to precondition the high-dimensional data, consisting of broadband colors, to sharpen high-dimensional data clusters to enhance cluster separation in the projection. The enhanced cluster separation should aid classifiers to label stars, galaxies and QSOs with higher precision. To answer this question I have mainly looked at three datasets. These datasets I name CPz STAR, CPz GAL and CPz QSO. The suffixes, i.e., STAR, GAL and QSO, reflect the fact that each of these datasets were designed by LF20 for three separate binary classification problems, i.e., star/non-star, galaxy/non-galaxy and QSO/non-QSO; for more details see Chapter 3.

In Chapter 4 I show that SDR can be used to consistently produce projections with high cluster separation. To quantitatively evaluate the degree of cluster separation, I have made use of the distribution consistency metric defined in Chapter 3. Particularly, linear DR methods such as LMDS and NPE showed the greatest improvement in terms of cluster separation when compared to UMAP and t-SNE. This is in line with what was found by Kim et al. (2022b) for other datasets. In addition to looking at cluster separation metrics, I have also looked at local neighborhood metrics and distance preservation metrics to further evaluate the performance of the projections. The results for the trustworthiness and continuity metrics (see, e.g., Table 4.8), which quantify the average proportion of false and missing neighbors in the projection, show that, while the sharpening step disturbs the structure of the high dimensional data, neighborhood relations are still well-preserved. This is in stark contrast with the results of the Jaccard similarity coefficient, which quantifies the average fraction of overlap between the sets of nearest neighbors in the feature and projection space. The Jaccard similarity coefficient values seem to indicate a very poor agreement between the neighborhood relations in the high-dimension feature space and the low-dimension projection space. The Venn diagrams of the different sets involved in the computation of the trustworthiness, continuity and Jaccard similarity coefficient metrics are given in Figure 8.2. From this Figure it is already clear that the Jaccard similarity coefficient and the trustworthiness and continuity metrics reflect different things about the neighborhoods of samples in the feature and projection spaces. However, one would expect that with very few samples in the sets $\mathcal{V}_i^k$ and $\mathcal{U}_i^k$, the sets $\mathcal{N}_i^k$ and $\mathcal{M}_i^k$ should almost entirely overlap. This is not reflected by the metrics, however. That is, I obtained high values for the continuity and trustworthiness metrics and a low value for the Jaccard similarity coefficient. The only difference between these metrics is that the continuity and the trustworthiness metrics are based on rankings instead of counting the number of samples in either $\mathcal{V}_i^k$ or $\mathcal{U}_i^k$. That is, the Jaccard similarity coefficient only checks whether both sets $\mathcal{N}_i^k$ and

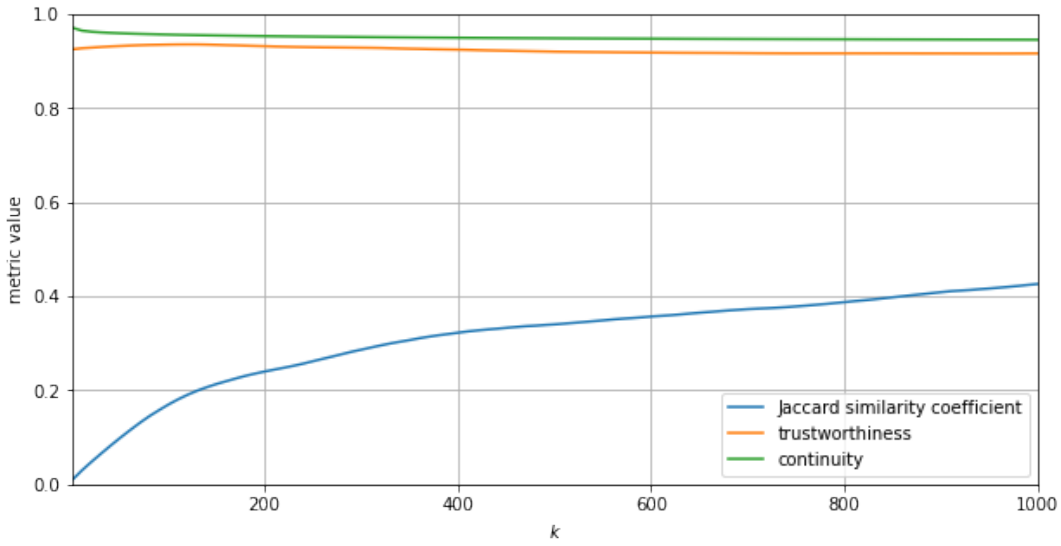
$\mathcal{M}_i^k$ are similar regardless of how distant or close the false neighbors ($\mathcal{V}_i^k$) or missing neighbors ($\mathcal{U}_i^k$) are on average with respect to any sample. This could mean that the Jaccard similarity coefficient is more sensitive to k, especially for low values, when compared to the trustworthiness and continuity metrics. To explore this effect, I have plotted the Jaccard similarity coefficient, trustworthiness and continuity metrics as a function of k for the LMDS and sharpened LMDS projections of CPz STAR dataset in Figure 8.1. I have used values of k ranging from 1 to 1001 in steps of 10. The Figure demonstrates that whereas the values for the trustworthiness and continuity metrics seem to be consistent for any value of k, the Jaccard metric is not. For both LMDS and SLMDS, the Jaccard metric turns out to be an increasing monotonic function starting from roughly 0 and slowly increasing towards 1. In the case of LMDS, the function even seems to follow a power law with a slope of roughly 0.4. Compared to the trustworthiness and continuity metrics, which both have consistently high values close to 1, the Jaccard similarity coefficient does not seem to be a particularly good metric to evaluate neighborhood preservation for the datasets and projection techniques considered in this work. Further research is needed to show why this is. As a start one may want to look at the effect of the normalization term, i.e., the cardinality of the union $\mathcal{N}_i^k \cup \mathcal{M}_i^k$, in the definition for the Jaccard similarity coefficient. This denominator namely includes not only the true neighbors in the feature space but also the false neighbors present in the projection whilst the numerator contains only the number of true neighbors present in both. This can drag down the value of the Jaccard similarity coefficient.

The SDR projections derived from the CPz STAR dataset presented in Chapter 4 all show similar features. Usually the star and galaxy classes are well-separated, whereas some subclusters of the QSO and galaxy tend to mix. This feature is not just present in the projections derived from the CPz STAR dataset but also in those derived from the CPz GAL and CPz QSO datasets presented in Appendix A. This mixing could in part be explained by the following argument. In their paper about unsupervised star, galaxy and QSO classification using HDBSCAN, LF20 explain that in 52% of the case the spectra of the objects in the CPz sample had class label "UNKNOWN". Therefore, they decided to assign labels to these samples according to the spectroscopic redshift of each object. Samples with a redshift less than 0.0015 were assigned to the star class, while samples with a higher redshift were assigned to the galaxy class. This implies that stars moving away from us with a velocity higher than $450\,\mathrm{km\,s^{-1}}$ would be labeled as a galaxy. Furthermore, galaxies moving towards us or away from us with a velocity less than $450\,\mathrm{km\,s^{-1}}$ would be labeled as stars. This mislabeling can appear as a mixing between stars and galaxies in the projections presented in this work. Additionally, the mixing of the galaxy and QSO classes in the projections might be explained by some QSOs having been inaccurately labeled as galaxies. This may also explain the relatively high number of post-consolidation outliers when compared to the galaxies after applying lowest-entropy consolidation (see Figure 6.6). This indicates that classifiers are uncertain about which class label to assign in regions where galaxies are mixed with QSOs.

In Chapter 5 I discuss a way to make SDR more scalable and to allow for out-of-sample (OOS) capability. This allows SDR-aided classification to be applied quickly and makes it suitable for large catalogs and on-the-fly classification. My approach is to train a neural network to reproduce the projections produced by SDR. Adhering to the naming introduced by Kim et al. (2022a), this projection technique is called SDR-NNP. Some of the SDR-NNP projections I produce using a test set comprising 20% of the full dataset and various DR methods are shown in Figures 5.4, 5.5, 5.6 and 5.7. None of these projections reproduce the exact structure of the SDR embedding,



(A)



(B)

FIGURE 8.1: The Jaccard similarity coefficient, trustworthiness and continuity metrics plotted as a function of k. The values were derived by computing either of these metrics for the projections obtained from the CPz STAR dataset through LMDS (Figure 8.1a) and sharpened LMDS (Figure 8.1b).

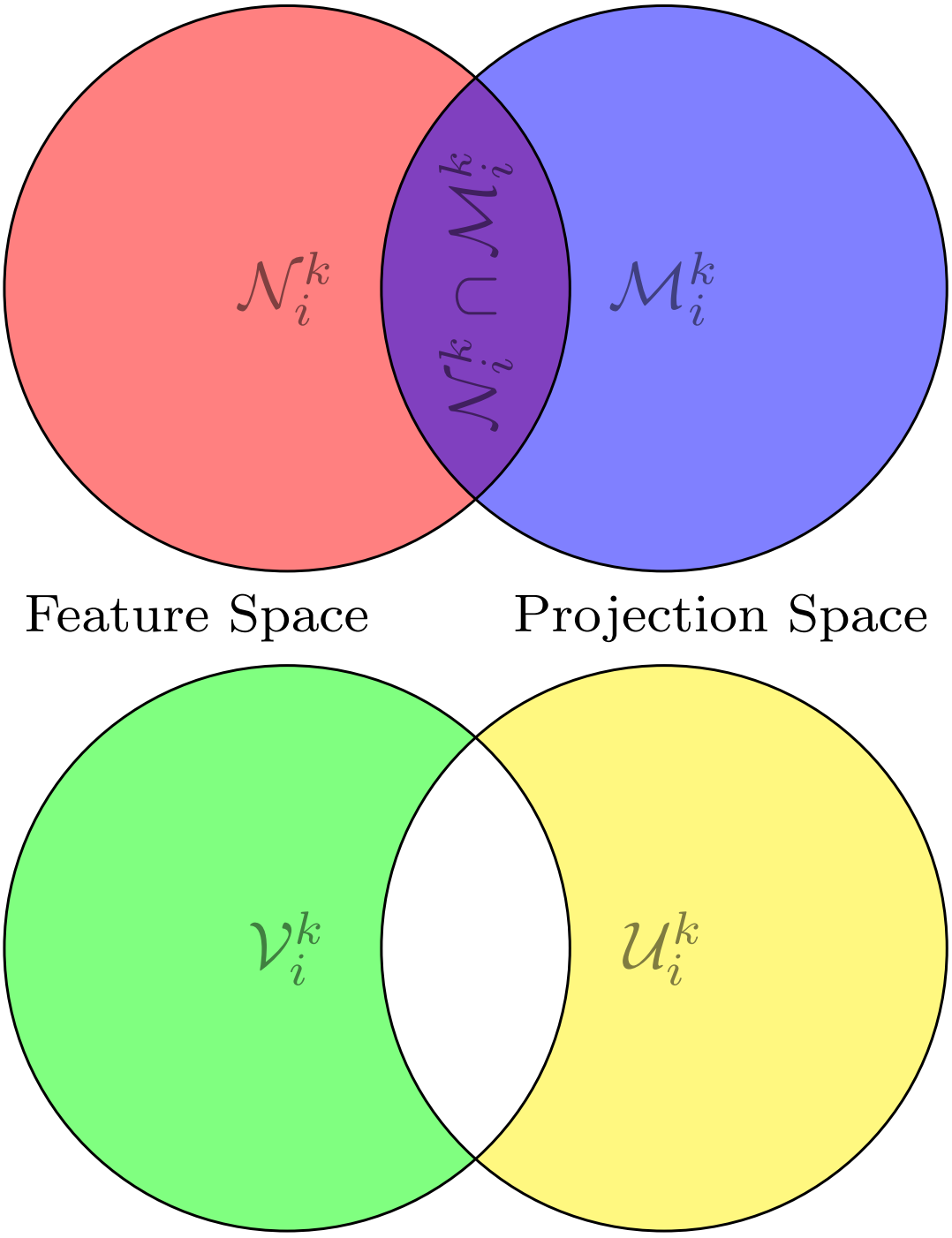


FIGURE 8.2: Venn diagrams of the sets involved in the computation of the trustworthiness, continuity and Jaccard similarity coefficient metrics presented in Chapter 3. The sets $\mathcal{N}_i^k$ and $\mathcal{M}_i^k$ are the k nearest neighbor sets of sample i in the feature and the projection space, respectively. Both of these sets are used for computing the Jaccard similarity coefficient. The set $\mathcal{V}_i^k$ contain the k nearest neighbors of sample i in the feature space that are not in its k-nearest-neighbor set in the projection used for computing the continuity. The set $\mathcal{U}_i^k$ consists of the k nearest neighbors of sample i in the projection space that are not in its k nearest neighbor set in the feature space used for computing the trustworthiness.

i.e., its subclusters. However, the star, galaxy and QSO clusters still appear well-separated, as needed for precise classification. When needed this can be improved upon in later work by allowing for larger more complex neural networks than that presented in this work, because, inspecting the training history in these figures, one may conclude that there is little more to gain from training for more epochs.

In Chapter 6 I show the results of various classifiers (see Figure 6.5). These classifiers included the k-nearest-neighbors classifier, various support-vector classifiers (i.e., using linear, third-degree polynomial and radial basis function kernels), a neural-network classifier and the XGBoost classifier. In addition, I show the results of using various consolidation methods to consolidate the classification results derived from the CPz STAR, CPz GAL and CPz QSO datasets. These consolidation methods include the lowest-entropy method, average-probability method, alternative method and majority vote. Among these methods the lowest-entropy and the average-probability methods yield the most control over the final classification. Respectively, these methods allow the user to define post-consolidation entropy and probability thresholds which determine the samples that are assigned to the post-consolidation "OUTLIER" class. By adjusting this threshold, one can improve the precision of the classification at the cost of a lower completeness (i.e., recall). The lowest-entropy and the average-probability methods have both strengths and weaknesses. The lowest-entropy method does not account for identical probability distributions that lead to different classifications, e.g. $\{p_i\} = \{0.5, 0.3, 0.2\}$ and $\{p_i\} = \{0.3, 0.5, 0.2\}$. On the other hand, the average-probability method is sensitive to outliers, i.e., classifiers that yield a significantly different prediction compared to the others.

In Chapter 7 I look in more detail at the sharpened-LMDS projection derived from the CPz STAR dataset. In many instances I conclude that the subclusters present in the projection are an oversegmentation feature caused by the LGC step of SDR. However, these subclusters do give us additional physical insights. First, using astrophysical parameter data from Gaia DR3 (Creevey et al., 2022), I demonstrated in Chapter 7 that many of the stellar subclusters present in the projection produced by sharpened LMDS convey relevant physical information by plotting HR and effective temperature versus surface gravity diagrams. These diagrams show that each of the stellar subclusters is comprised of stars with a different effective temperature and spectral class. Second, using these subclusters and given the fact that LMDS is a distance-preserving method I determine that M stars and quiescent elliptical galaxies closely resemble each other in terms of color (see the end of Section 7.2). This raises the question as to why this is the case. Looking at Table 2.1 we observe that many of the colors in the CPz STAR dataset are comprised of near-infrared (NIR) broadband magnitudes. Therefore, the projection likely reflects mostly NIR color relations. Additionally, the elliptical galaxies we are comparing to M stars are mostly quiescent. Therefore, it might be suitable to assume that these galaxies consist mostly of old stellar populations. From Figure 14 in Verro et al. (2022), which provides the contribution of RGB stars and TP-AGB stars to the K-band luminosity in various single stellar population (SSP) models using the X-shooter Spectral Library, we see that for old stellar populations ($\gtrsim 2\,\mathrm{Gyr}$) the K-band luminosity is mostly dominated by red giant branch (RGB) stars. RGB stars are mainly comprised of K stars and M stars which might explain why these elliptical galaxies closely resemble M stars in terms of their NIR colors and why they are placed closely together in the sharpened LMDS projection. Third, we observe that different galaxy and QSO subclusters have different average redshifts (see Figures 7.3 and 7.7). This suggests that the projections may allow astronomers to obtain rough redshift estimates for

objects based on their location in the projection. Furthermore, we observe gradients in both stellar mass and dust luminosity for the galaxies in Figure 7.4. Based on this observation one could try to find estimates for the stellar mass and dust luminosity of galaxies based on their location in the projection space. Any of these issues could be topics of future research. All of these observations demonstrate that the unnecessary oversegmentation present in the projections retain physical information and allow us to study the structure of projections and hence also the higher dimensional color data in greater detail.

In the Introduction I discuss the work of LF20, in which they used an unsupervised clustering algorithm called HDBSCAN to perform star, galaxy and QSO classification. The classifications they obtained using various consolidation methods were included in the CPz catalog.[1] To compare their results with those I obtained in Chapter 6, I select the same subset as the one I use for testing in Chapter 6. The classification performance metrics computed using the labels obtained by LF20 using the alternative, optimal and highest probability consolidation methods are presented in Table 8.1. The exact description of each of these consolidation methods can be found in LF20. It is however important to note that the alternative consolidation method used by LF20 is different from that used in this work. This is because LF20 tried to consolidate three binary classifiers, whereas I consolidate three multi-class classifiers. Comparing Tables 8.1 and 6.1, I find that the performance of both HDBSCAN and SDR-aided classification is similar across all the different performance metrics. It is best to compare the alternative and optimal-method results in Table 8.1 with the alternative and majority-vote method results in Table 6.1 as the other methods are sensitive to changes in threshold. In the case of the highest-probability method used by LF20, the results might differ by changing the number of catalog realizations, the choice of distribution and the threshold for sigma. Now that we have seen that both classification methods behave similarly, what are the advantages and disadvantages of SDR-aided classification when compared to using HDBSCAN? First, SDR-aided classification has out-of-sample (OOS) capability through the use of SDR-NNP models, which makes it more scalable than HDBSCAN. HDBSCAN needs to be rerun on the full dataset every time new data becomes available. Additionally, one can apply SDR on a small representative subset of the full dataset and train an SDR-NNP model to project the rest of the data. Second, we have seen that SDR-aided classification is less of a "black-box", as it is a supervised-learning method. This allows the user to inspect the decision boundaries in the projections and determine whether everything works correctly. In addition, as I show in Chapter 4, one can validate whether the SDR projections are accurate by computing various projection performance metrics. Lastly, HDBSCAN does not require the data to be labeled to perform classification, as it is an unsupervised-clustering algorithm. This is an advantage, as one does not need a labeled dataset to train the classifier.

[1]The revised catalog is available at the CDS through https://cdsarc.u-strasbg.fr/viz-bin/cat/J/A+A/633/A154.

TABLE 8.1: Post-consolidation performance of HDBSCAN.

Consolidation Method	**Accuracy**	**Class**	**Precision**	**Recall**	**F1 Score**
Alternative	0.9758	STAR	0.9980	0.9702	0.9839
		GAL	0.9825	0.9902	0.9863
		QSO	0.9487	0.8594	0.9018
Optimal	0.9771	STAR	0.9974	0.9748	0.9859
		GAL	0.9825	0.9902	0.9863
		QSO	0.9429	0.8665	0.9031
Highest-probability	0.9778	STAR	0.9974	0.9748	0.9859
		GAL	0.9823	0.9909	0.9866
		QSO	0.9419	0.8689	0.9039

Chapter 9

Conclusion

This research aims to show whether stars, galaxies and QSOs can be classified using broadband colors with the help of sharpened dimensionality reduction (SDR). Based on a quantitative and qualitative analysis of the embeddings produced by SDR, I demonstrate that SDR consistently produces projections with a high degree of cluster separation. Additionally, I show that using these projections stars, galaxies and QSOs can be classified with high accuracy, precision and recall. Furthermore, I show that these results are on a par with those of an unsupervised-clustering algorithm called HDBSCAN (hierarchical density-based spatial clustering of applications with noise) which has been used previously by LF20 for this classification task. Furthermore, when comparing SDR-NNP (sharpened dimensionality reduction through neural-network projections) aided classification to HDBSCAN. I conclude that SDR-NNP is the desired method for on-the-fly classification and classification of large datasets, as SDR-NNP has out-of-sample (OOS) capability. In addition, SDR aided classification is less of a "black-box" method as it allows the user to inspect the projection along with its decision boundaries. A limitation of SDR-aided classification as it is presented in this work is that it requires a training set with known class labels.

Inspecting the projections yielded by sharpened LMDS (landmark multi-dimensional scaling) in more detail I find that many of the small subclusters present in the projection are likely oversegmentation features caused by LGC (local gradient clustering), the sharpening algorithm employed by SDR. These subclusters do, however, give more insight on the structure of the projection, which also demonstrates that one of the strengths of SDR-aided classification is that it allows for data exploration.

In this work I have mainly focused on three datasets mostly comprised of different NIR broadband colors which were all derived from the CPz dataset composed by FP18 and revised by LF20. Therefore, future work may want to see how SDR-NNP performs on different datasets containing for example a set of optical broadband colors. In addition, one may want to investigate how SDR-NNP-aided classification can be applied to individual large astronomical surveys which usually only have a specific set of filters.

In conclusion, this work shows that SDR-aided classification can be used to classify stars, galaxies and QSOs based on their broadband colors with high accuracy, precision and recall. Furthermore, I show that this method can be applied in a scalable way by training a neural network to reproduce the projections yielded by SDR in linear time, making it suitable for classifying objects in large astronomical datasets. The Python wrapper of the SDR code written in C++ by Kim et al. (2022b) named `pySDR`[1], as well as a Python module containing a pipeline for applying all the steps presented in this thesis, i.e., finding the best SDR parameters, training an

[1] `pySDR`: https://gitlab.astro.rug.nl/lourens/pySDR.

SDR-NNP model and classifying objects based on the SDR-NNP projections using different classifiers, called `SHARC`[2], are available on GitLab.

[2] `SHARC`: https://gitlab.astro.rug.nl/lourens/SHARC.

Appendix A

Supplemental SDR Results

A.1 DR Optimization Results

This section presents several figures of the LMDS, UMAP, tSNE and NPE projection results obtained for the CPz GAL and CPz QSO datasets when optimizing the hyperparameters of these methods with respect to the distribution consistency metric (3.9).

A.1.1 CPz GAL Results

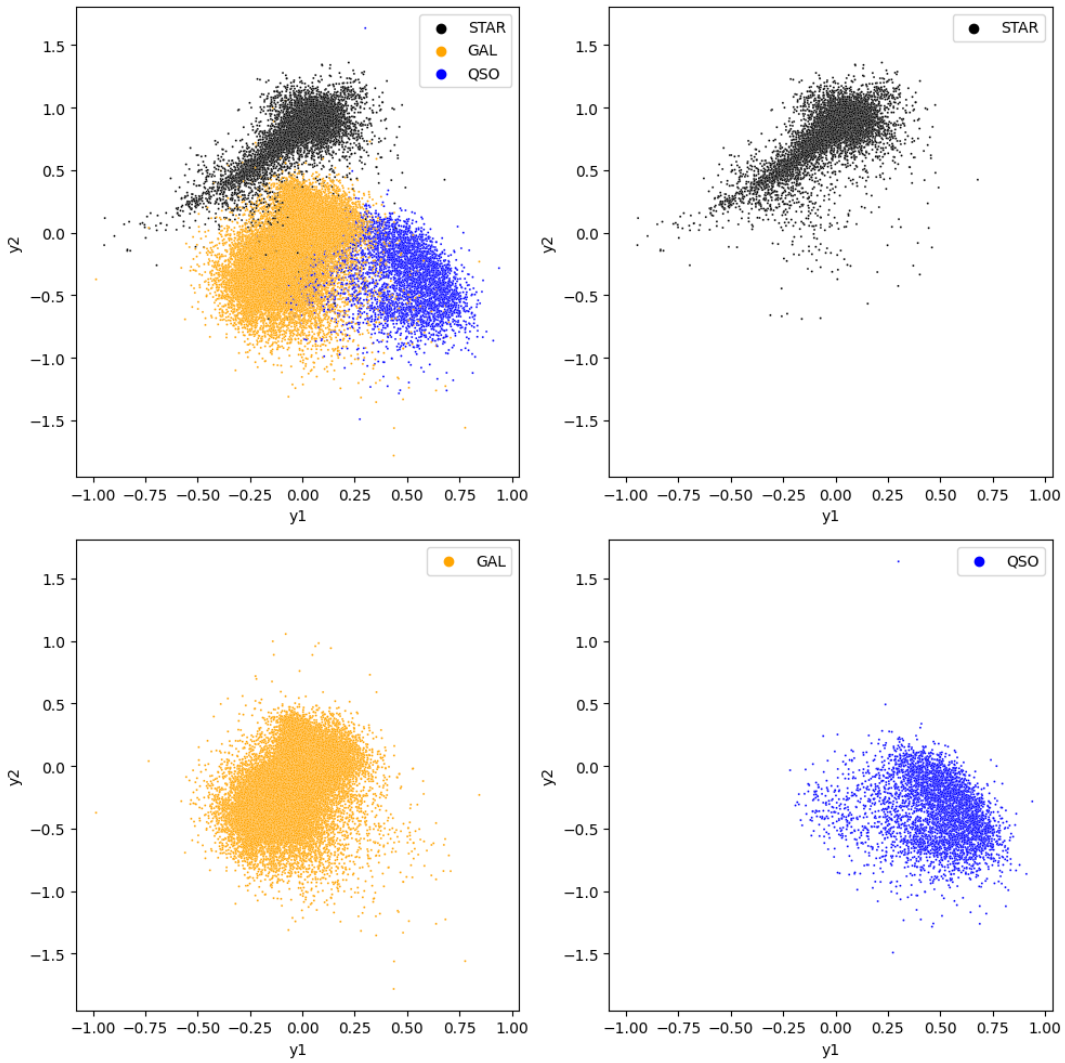


FIGURE A.1: The maximum distribution consistency LMDS projection ($M_{DC} = 0.9096$ with a landmark ratio of 0.08) of the CPz GAL dataset. Samples are colored according to the labeling provided by the CPz dataset.

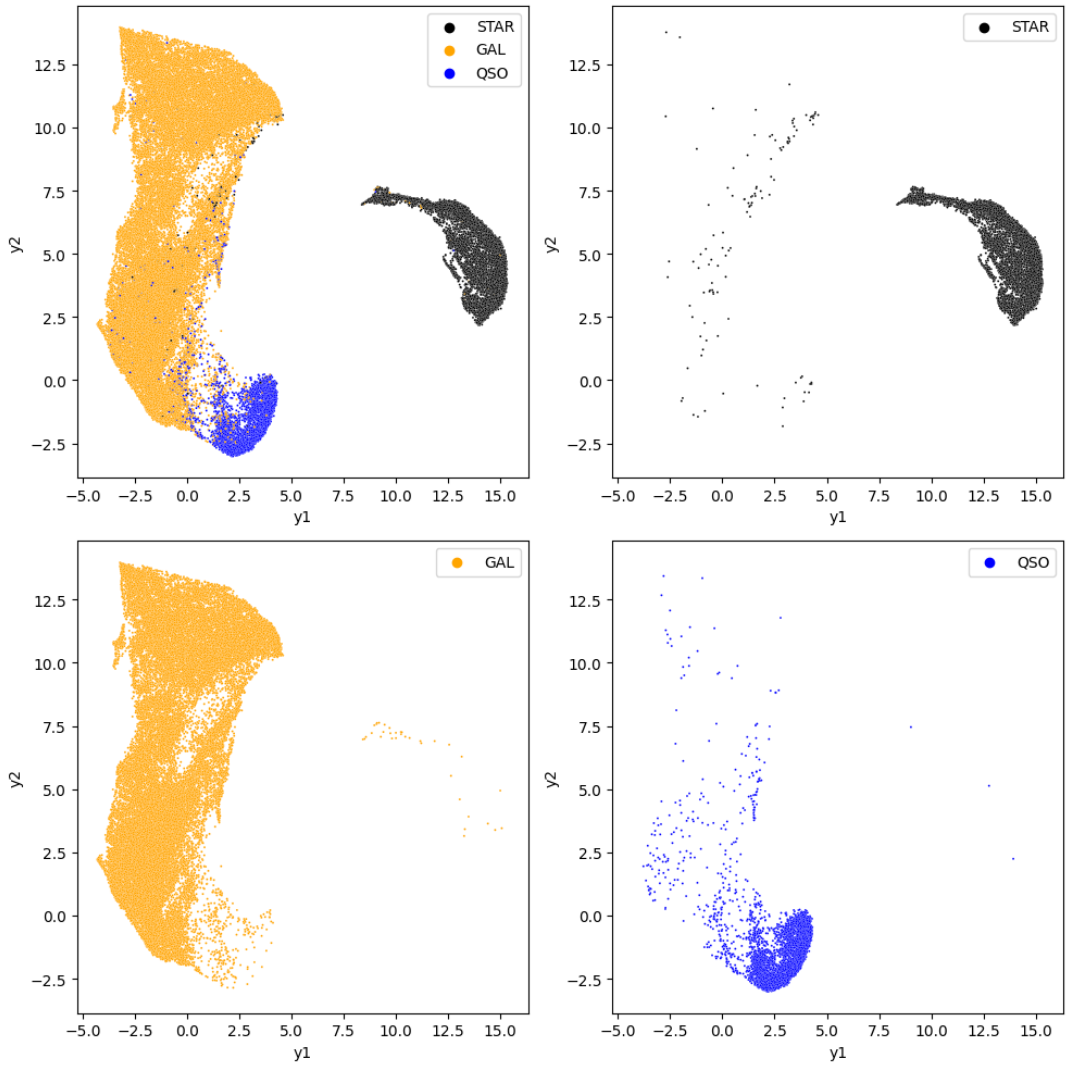


FIGURE A.2: The maximum distribution consistency UMAP projection ($M_{DC} = 0.9450$ with ("metric": "euclidean", "min_dist": 0.1, "num_neighbors": 80, "umap_init": "spectral")) of the CPz GAL dataset. Samples are colored according to the labeling provided by the CPz dataset.

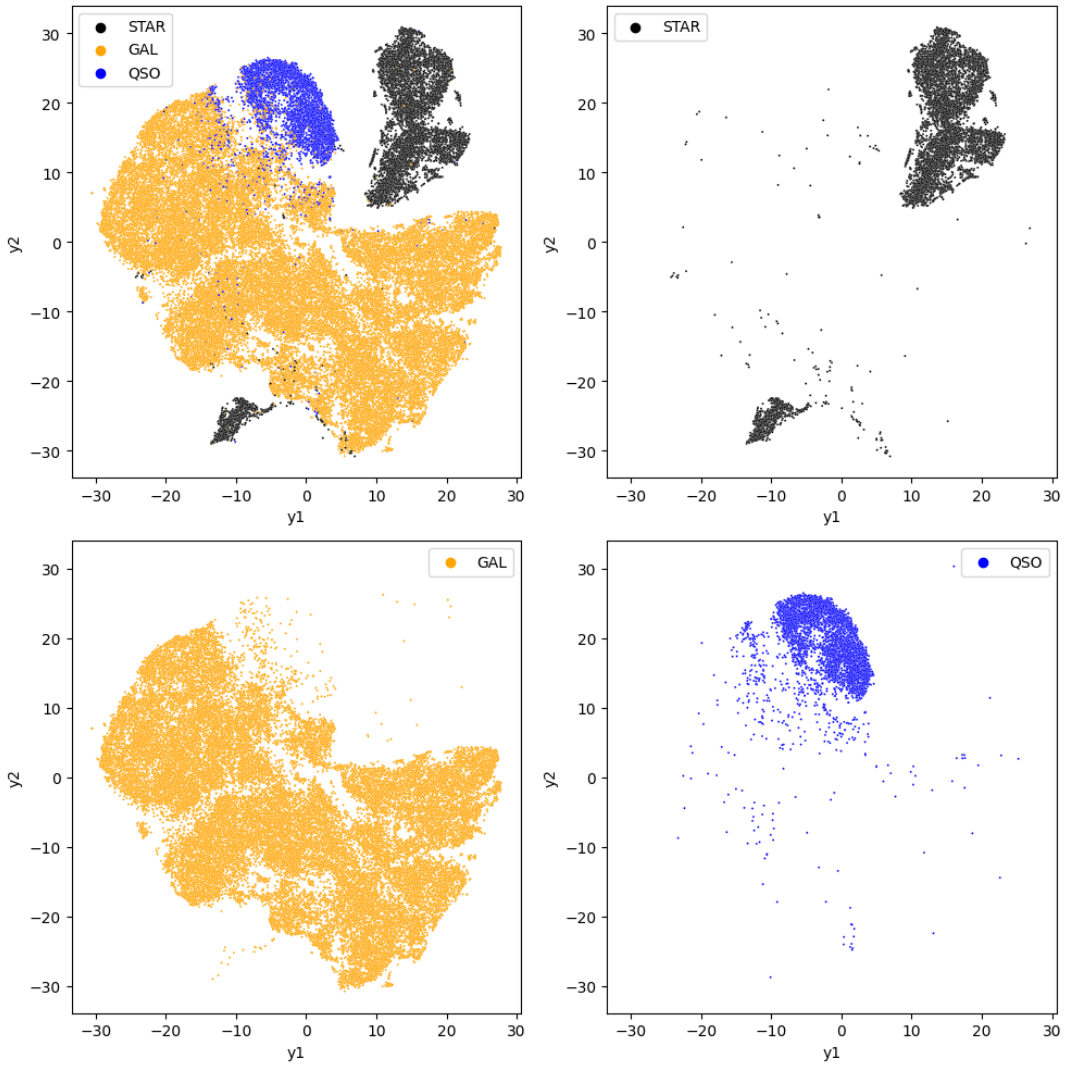


FIGURE A.3: The maximum distribution consistency tSNE projection ($M_{DC} = 0.8920$ with ("sne_perplexity": 180, "sne_theta": 0.5)) of the CPz GAL dataset. Samples are colored according to the labeling provided by the CPz dataset.

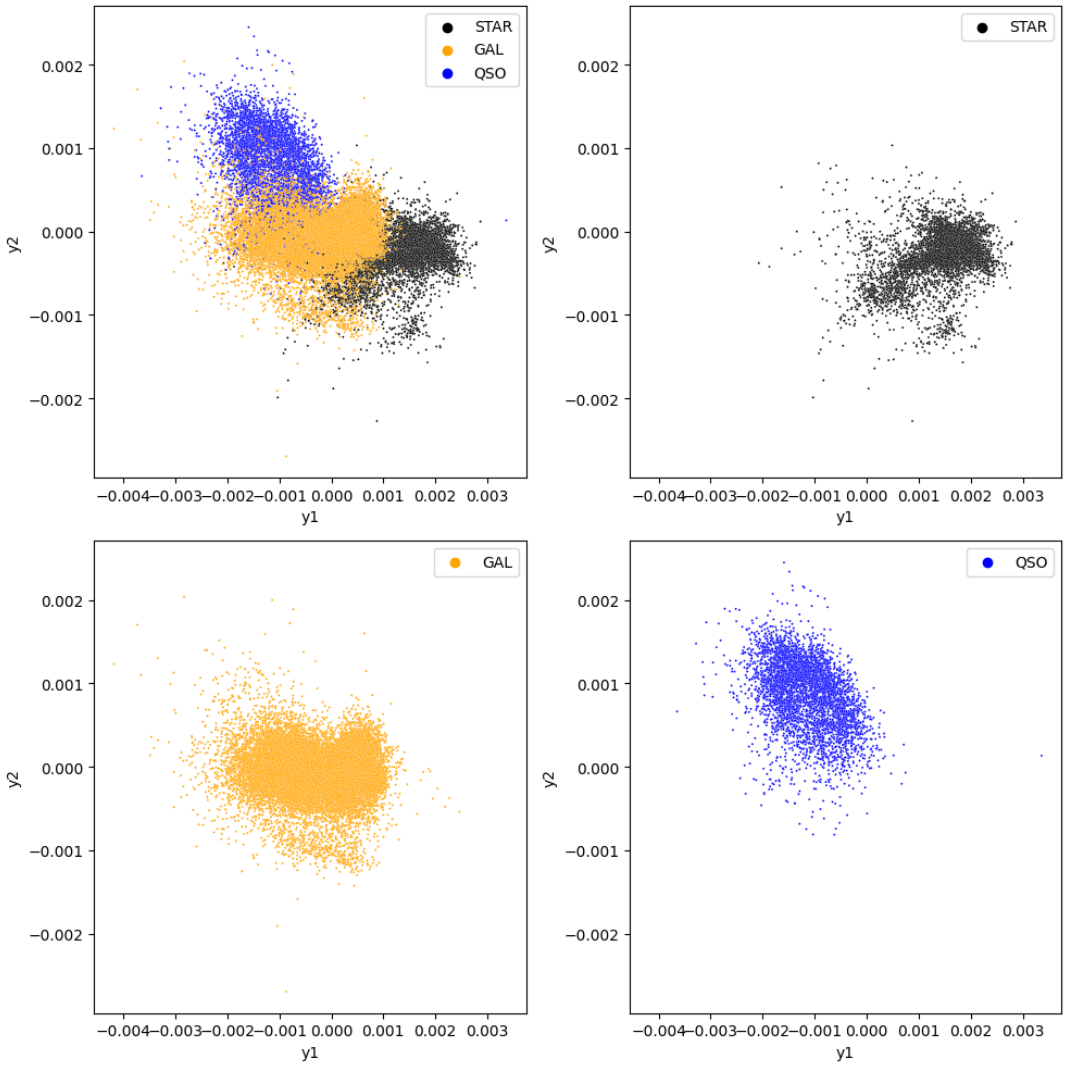


FIGURE A.4: The maximum distribution consistency NPE projection ($M_{DC} = 0.8655$ with 180 nearest neighbors) of the CPz GAL dataset. Samples are colored according to the labeling provided by the CPz dataset.

A.1.2 CPz QSO Results

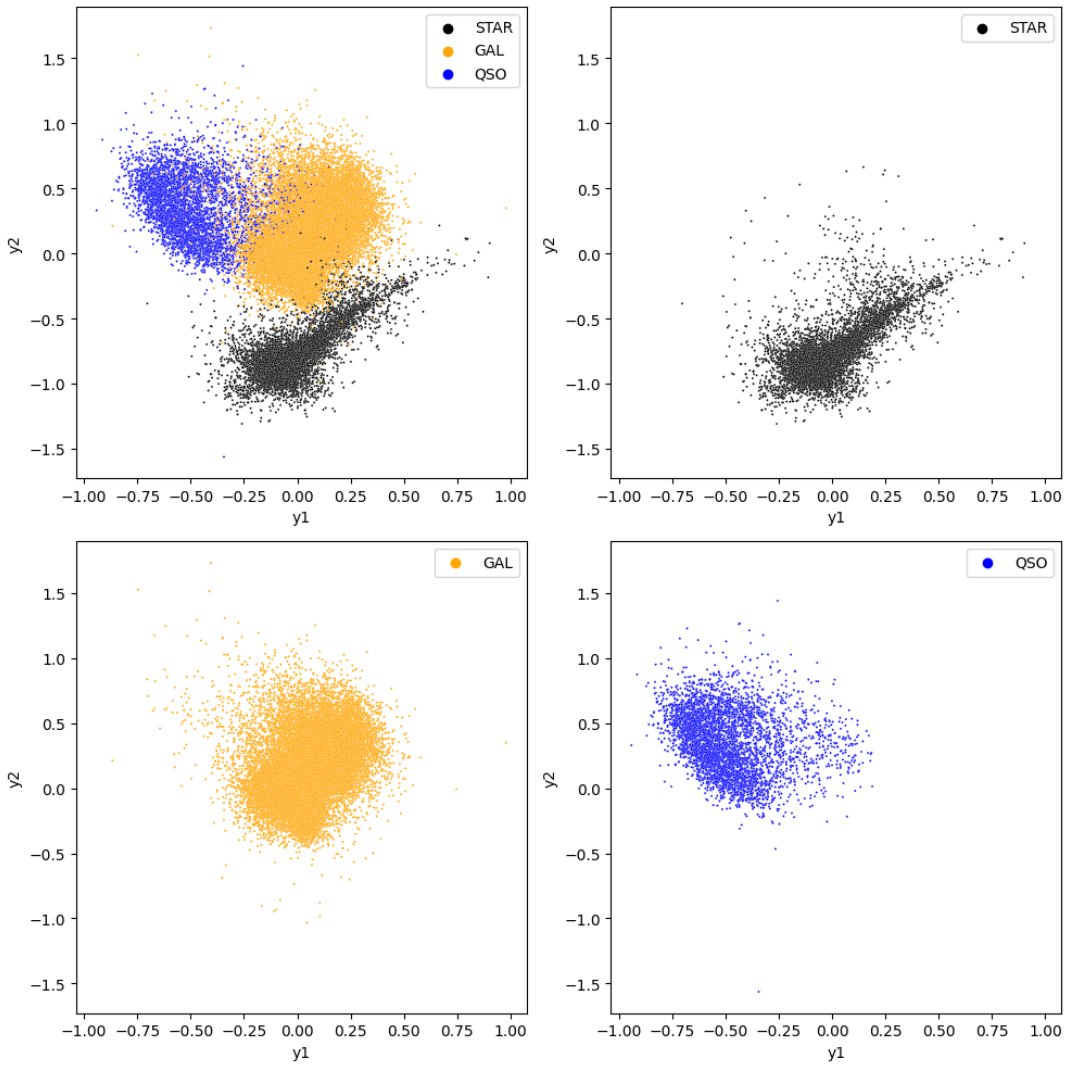


FIGURE A.5: The maximum distribution consistency LMDS projection ($M_{DC} = 0.9111$ with a landmark ratio of 0.04) of the CPz QSO dataset. Samples are colored according to the labeling provided by the CPz dataset.

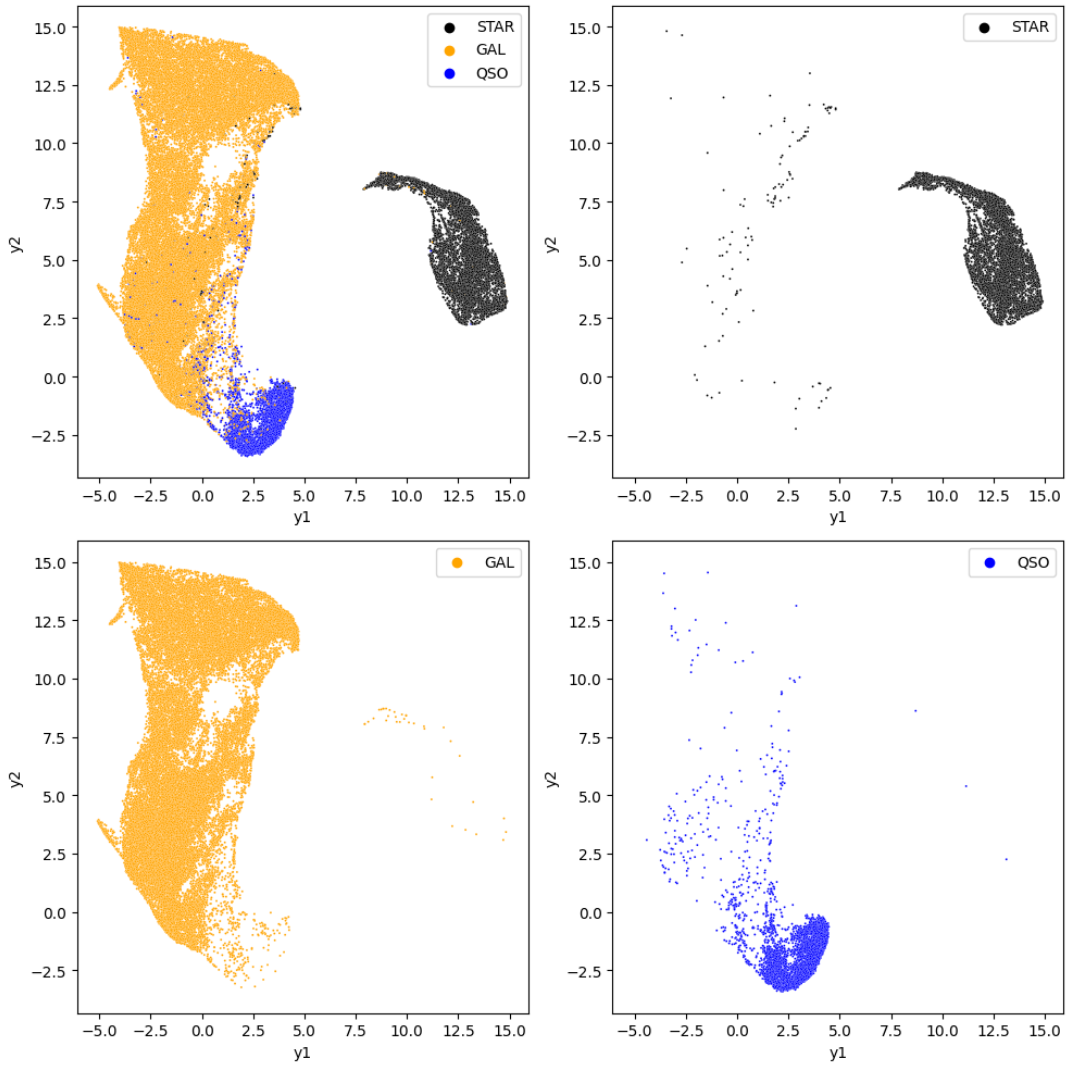


FIGURE A.6: The maximum distribution consistency UMAP projection ($M_{DC} = 0.9465$ with ("metric": "euclidean", "min_dist": 0.1, "num_neighbors": 40, "umap_init": "spectral")) of the CPz QSO dataset. Samples are colored according to the labeling provided by the CPz dataset.

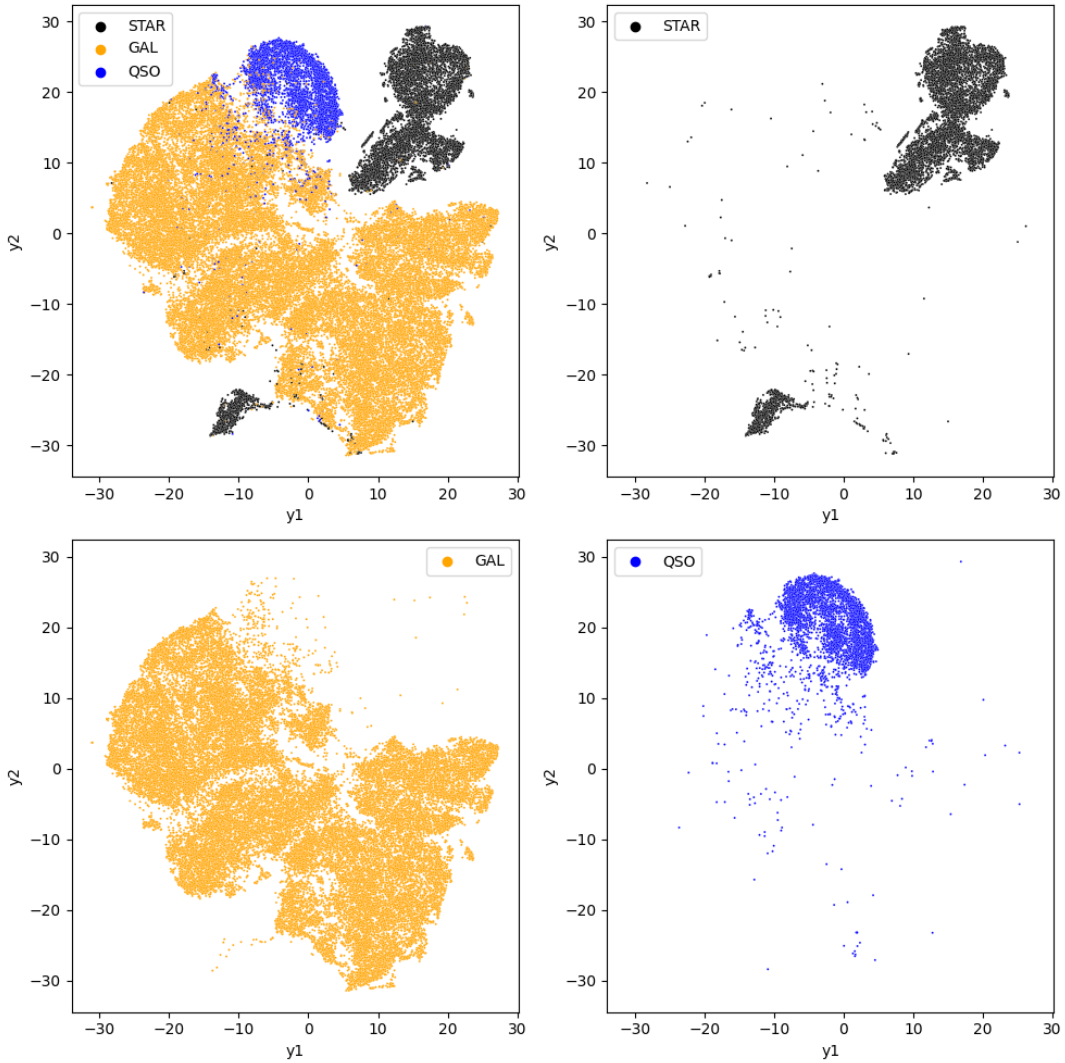


FIGURE A.7: The maximum distribution consistency tSNE projection ($M_{DC} = 0.8860$ with ("sne_perplexity": 180, "sne_theta": 0.5)) of the CPz QSO dataset. Samples are colored according to the labeling provided by the CPz dataset.

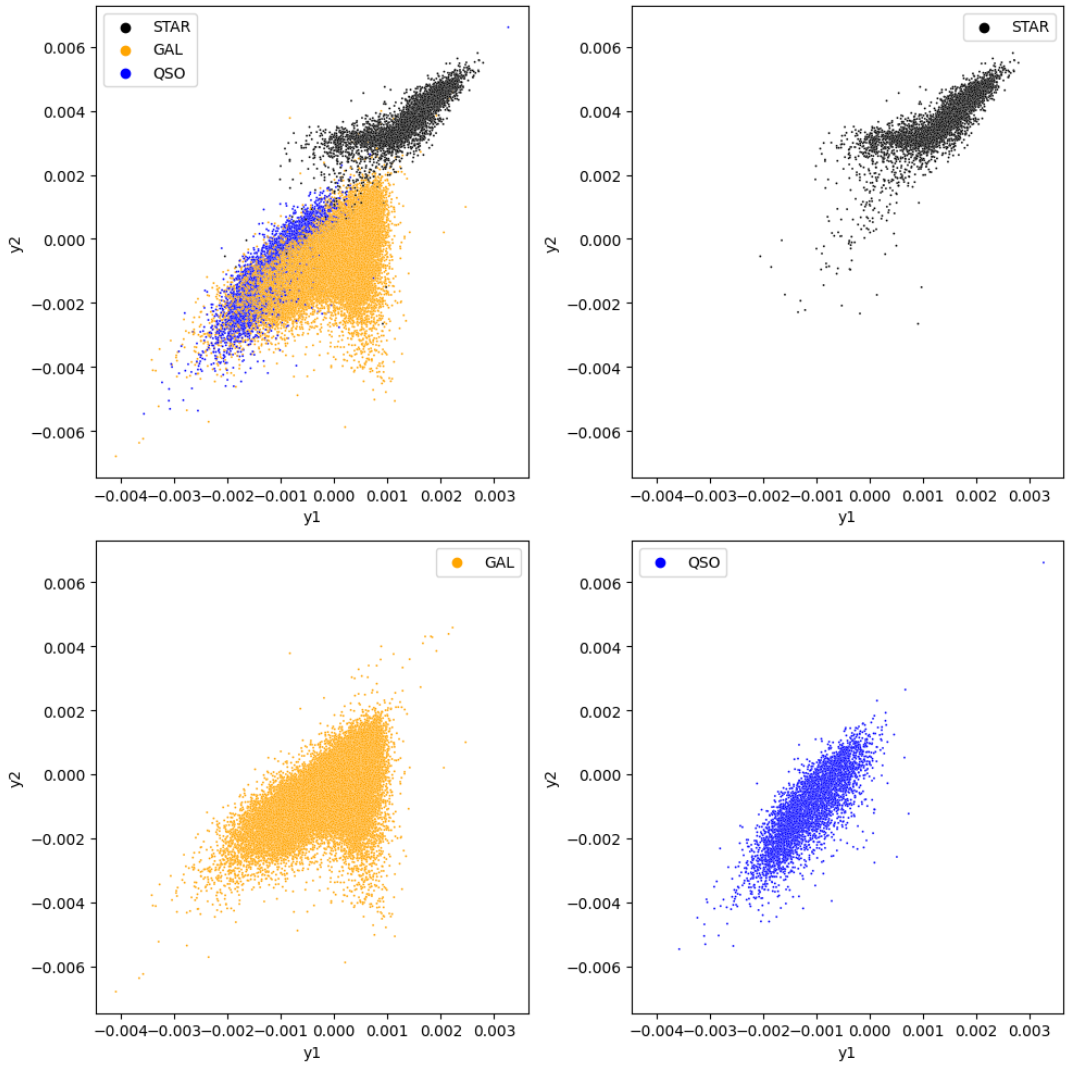
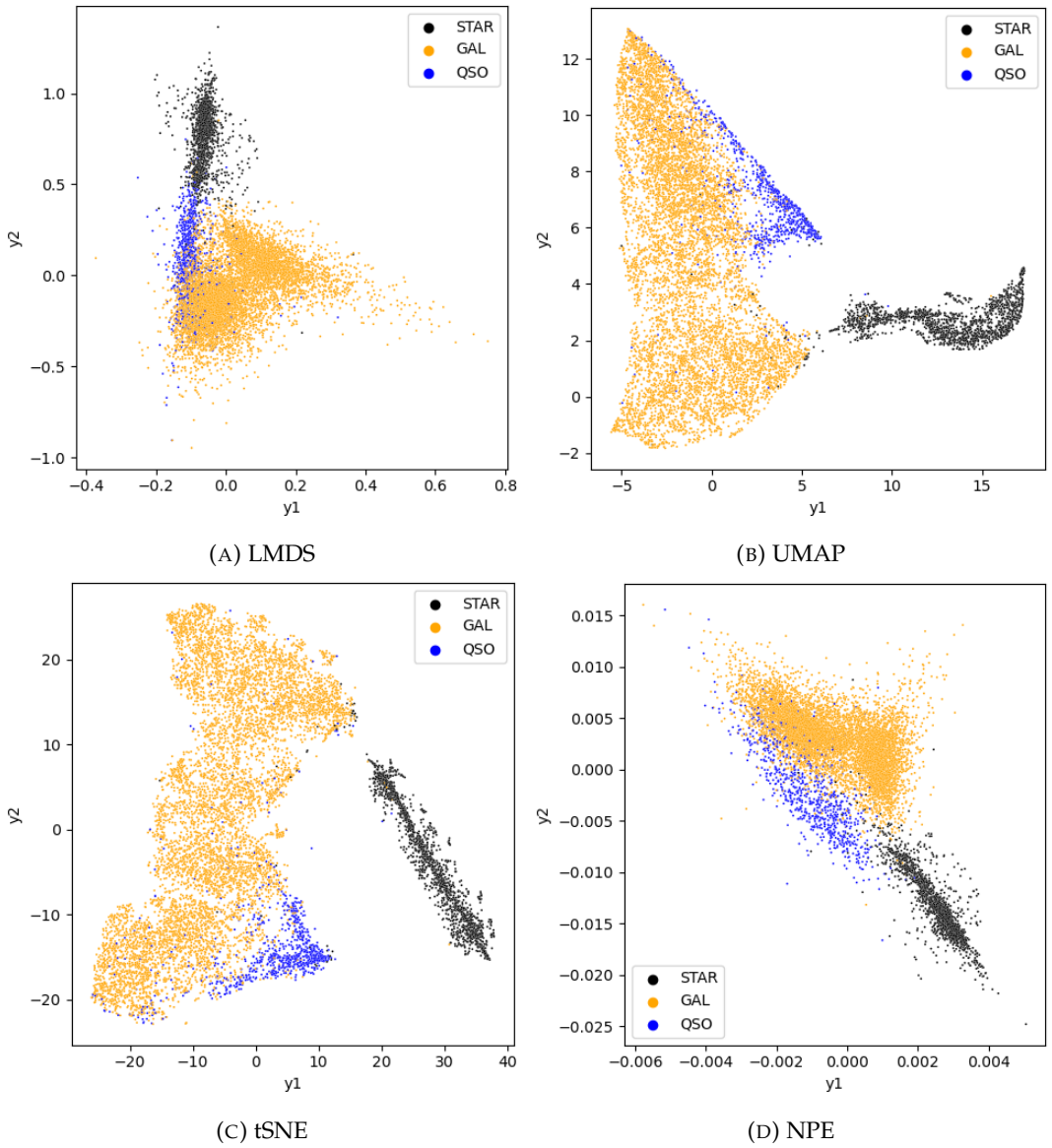


FIGURE A.8: The maximum distribution consistency NPE projection ($M_{DC} = 0.8404$ with 80 nearest neighbors) of the CPz QSO dataset. Samples are colored according to the labeling provided by the CPz dataset.

A.1.3 CPz ALL Results



(A) LMDS

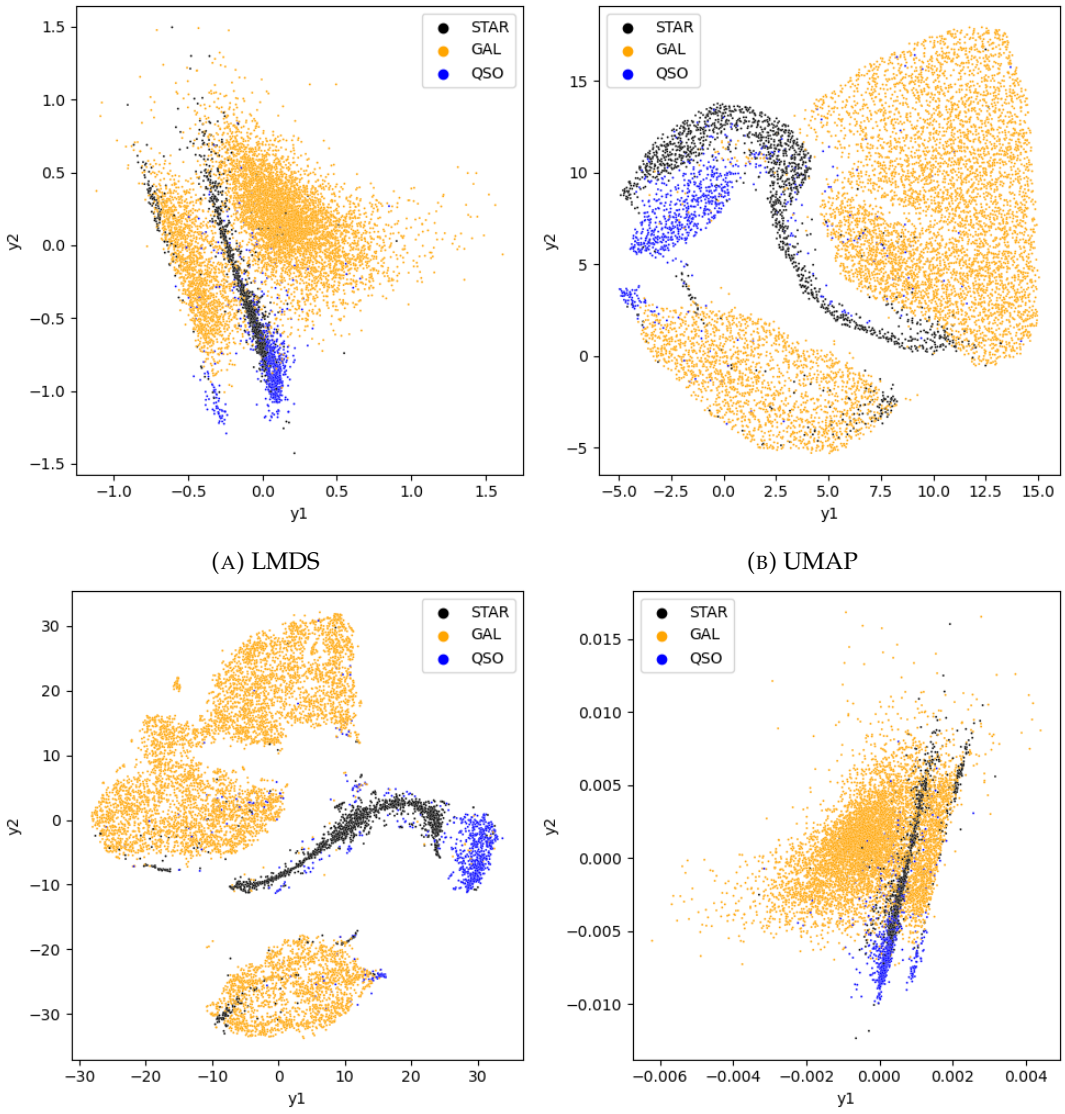
(B) UMAP

(C) tSNE

(D) NPE

FIGURE A.9: CPz ALL results of optimizing LMDS, UMAP, tSNE and NPE with respect to the composite metric given by equation (4.3).

A.1.4 CPz SDSS Results



(A) LMDS

(B) UMAP

(C) tSNE

(D) NPE

FIGURE A.10: CPz SDSS results of optimizing LMDS, UMAP, tSNE and NPE with respect to the composite metric given by equation (4.3).

A.2 LGC Optimization Results

This section presents several figures of the sharpened LMDS, UMAP, tSNE and NPE projection results obtained for the CPz GAL and CPz QSO datasets when optimizing the LGC hyperparameters respect to the distribution consistency metric ((3.9)) and using the best parameter sets found when doing DR optimization.

A.2.1 CPz GAL Results

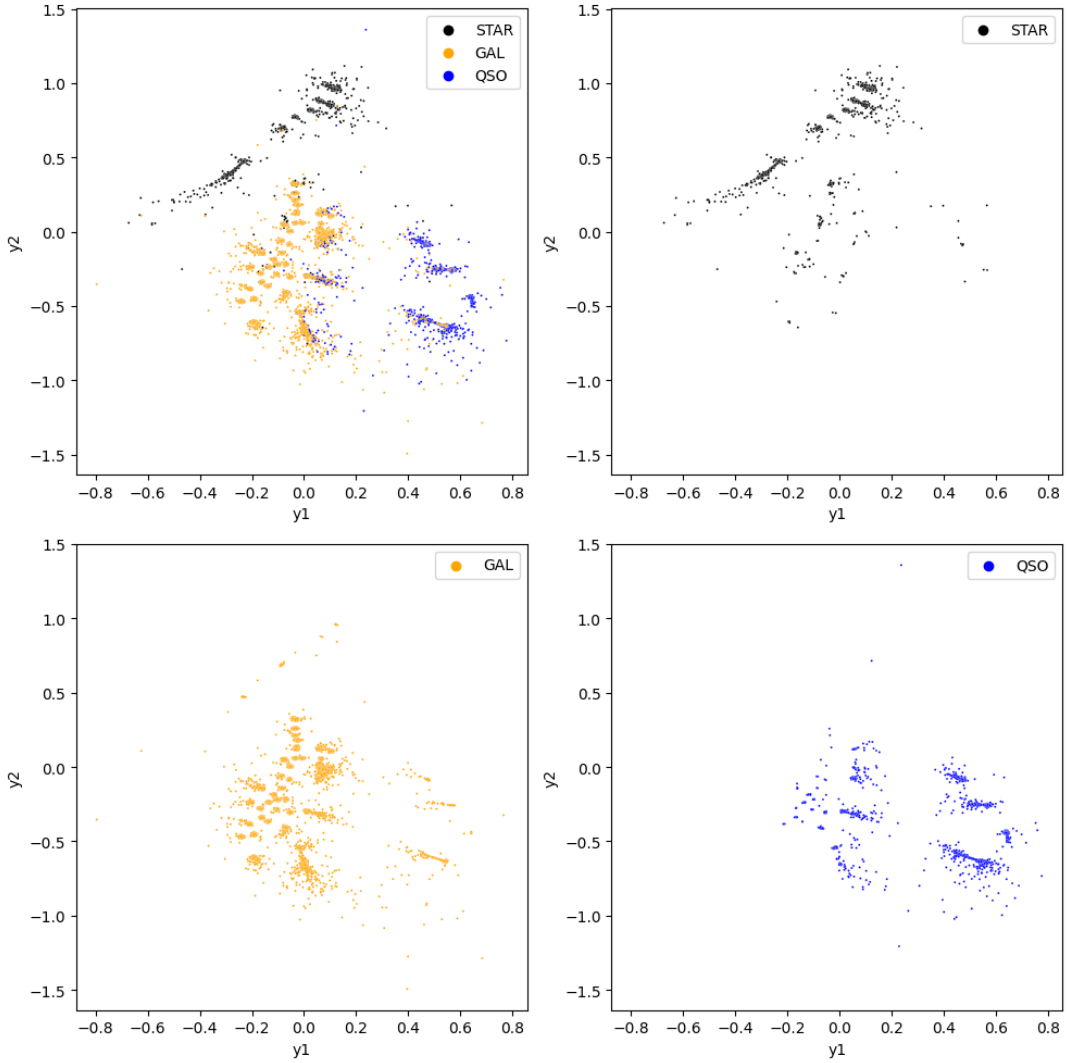


FIGURE A.11: The maximum distribution consistency sharpened LMDS projection ($M_{DC} = 0.9501$ with $(\alpha = 0.02, k = 325, T = 15)$ and a landmark ratio of 0.08) of the CPz GAL dataset. Samples are colored according to the labeling provided by the CPz dataset.

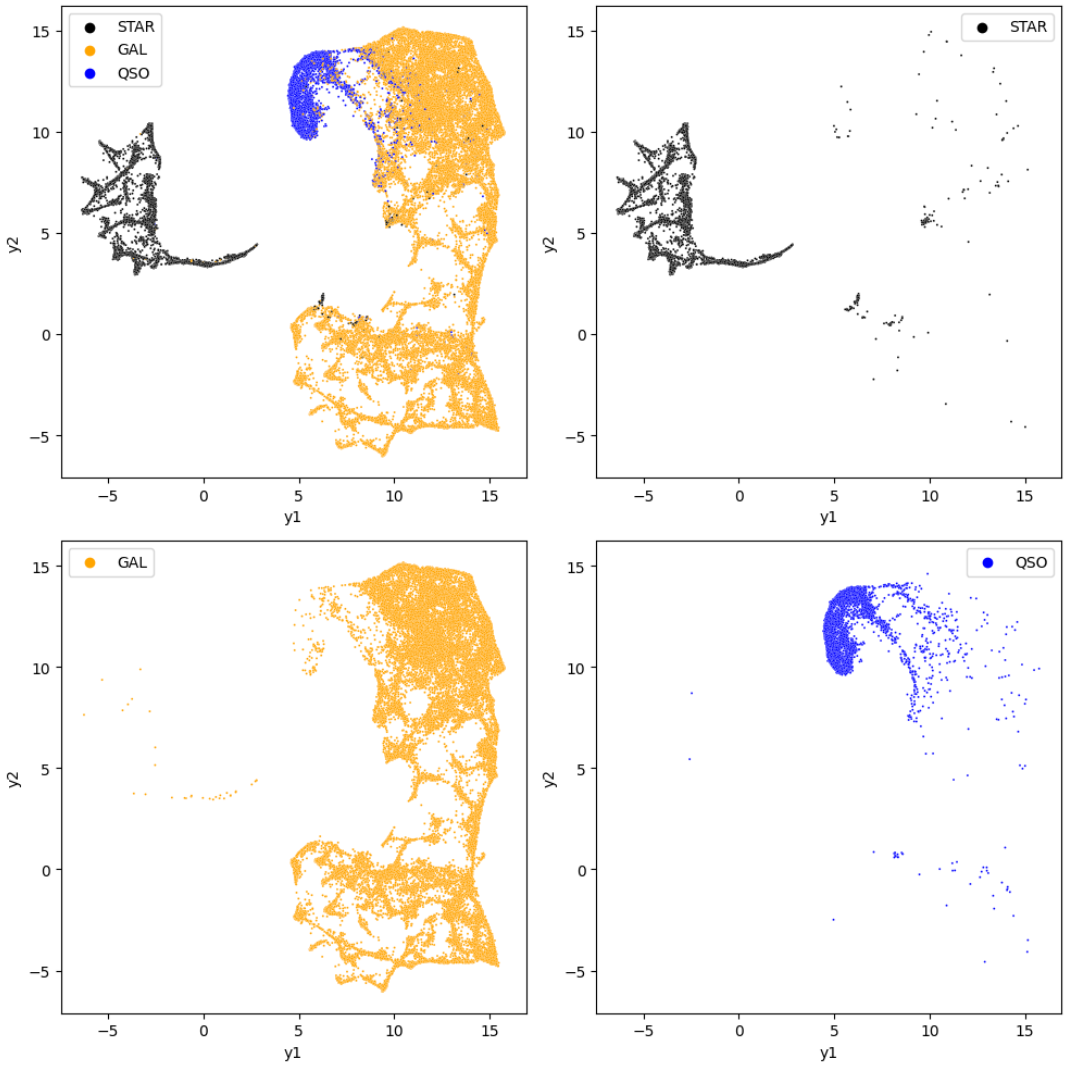


FIGURE A.12: The maximum distribution consistency sharpened UMAP projection ($M_{DC} = 0.9378$ with $(\alpha = 0.005, k = 125, T = 10)$ and ("metric": "euclidean", "min_dist": 0.1, "num_neighbors": 80, "umap_init": "spectral")) of the CPz GAL dataset. Samples are colored according to the labeling provided by the CPz dataset.

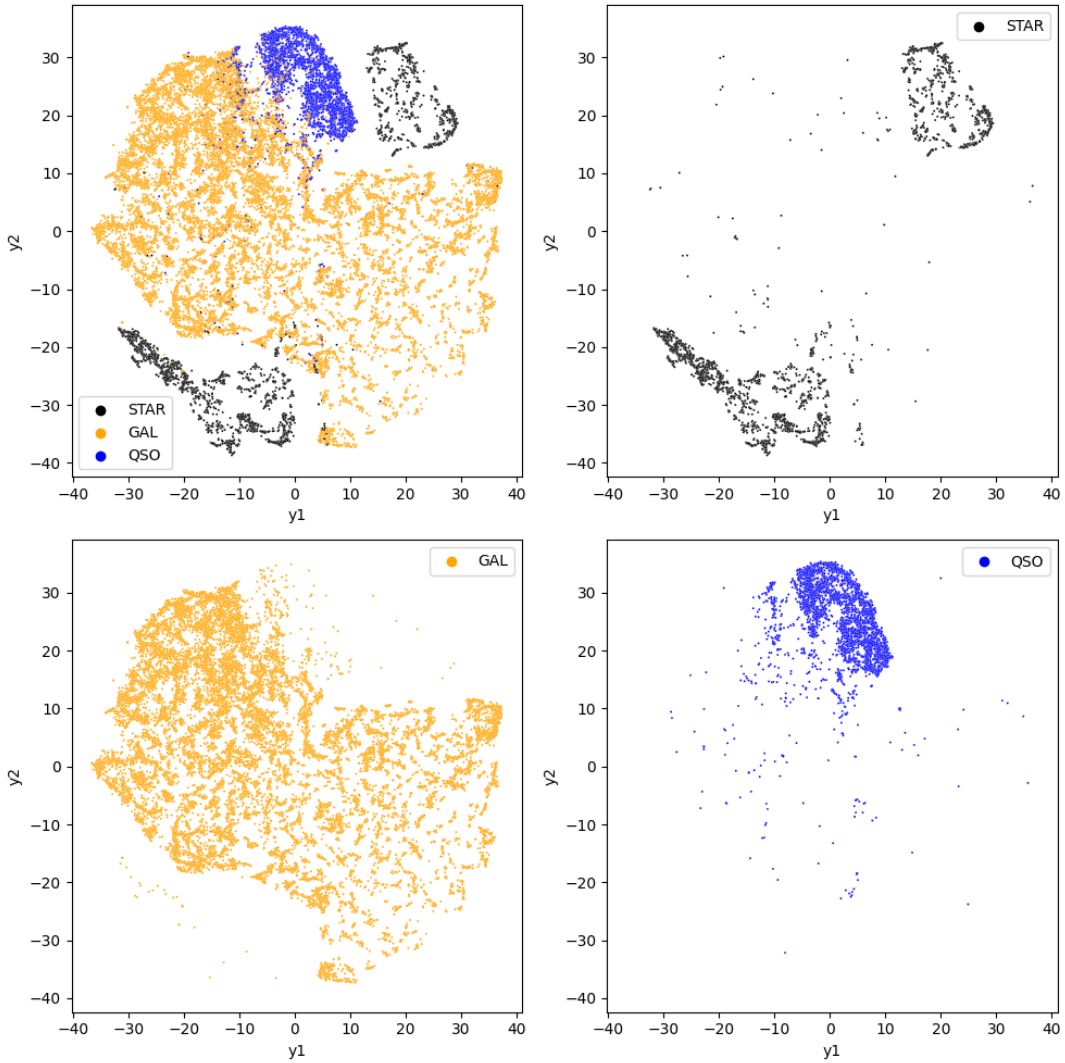


FIGURE A.13: The maximum distribution consistency sharpened tSNE projection ($M_{DC} = 0.9382$ with $(\alpha = 0.005, k = 25, T = 10)$ and ("sne_perplexity": 180, "sne_theta": 0.5)) of the CPz GAL dataset. Samples are colored according to the labeling provided by the CPz dataset.

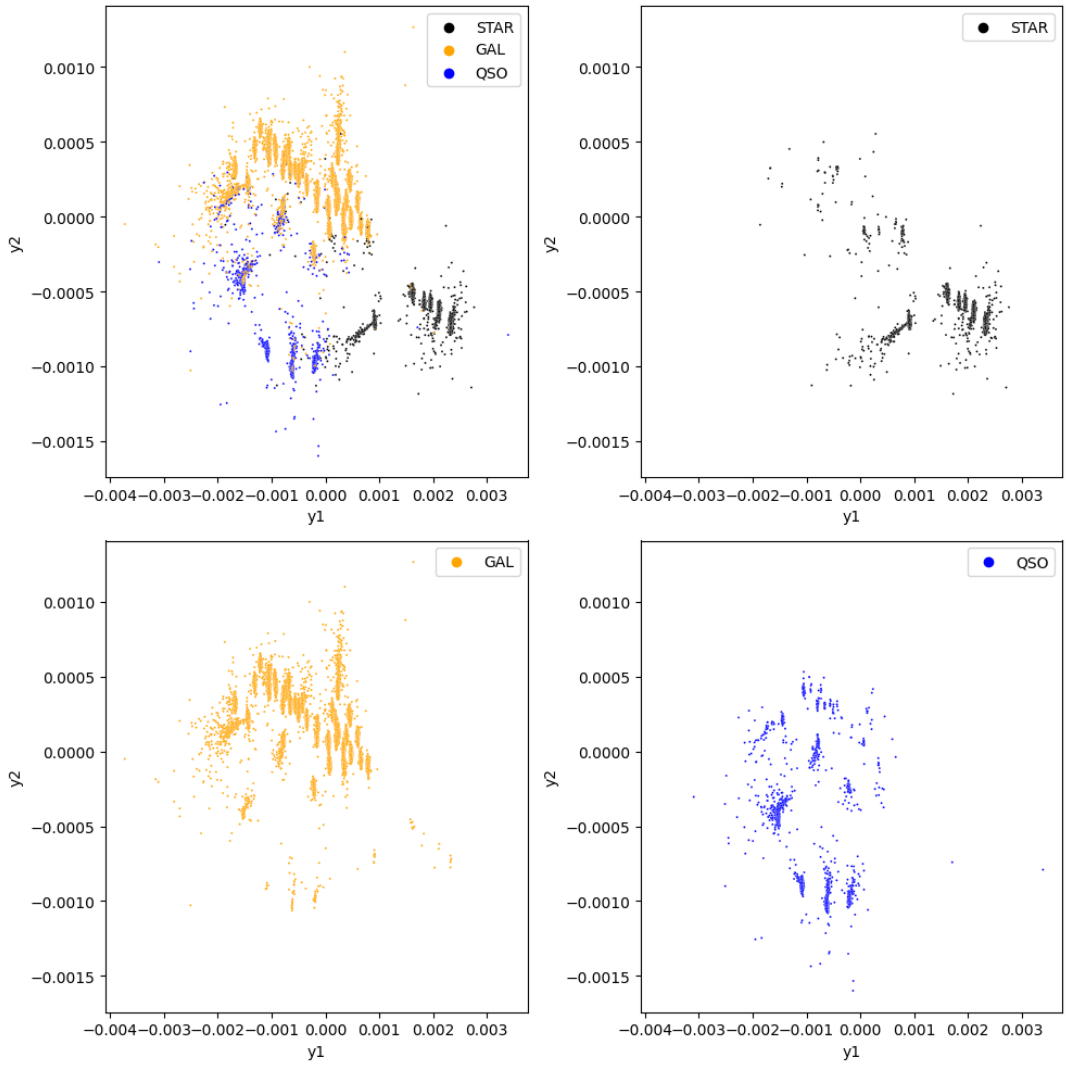


FIGURE A.14: The maximum distribution consistency sharpened NPE projection ($M_{DC} = 0.9416$ with $(\alpha = 0.02, k = 325, T = 15)$ and 180 nearest neighbors) of the CPz GAL dataset. Samples are colored according to the labeling provided by the CPz dataset.

A.2.2 CPz QSO Results

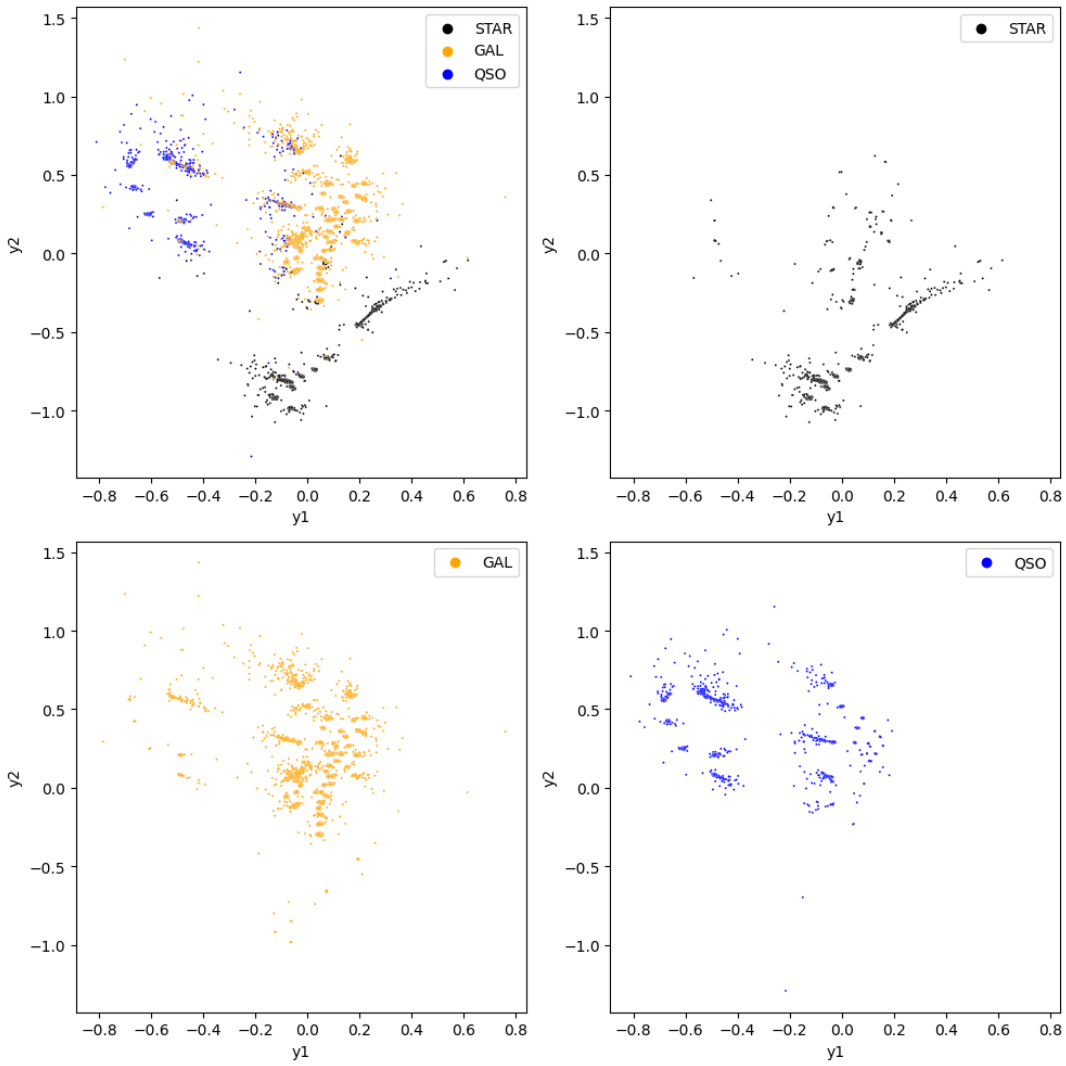


FIGURE A.15: The maximum distribution consistency sharpened LMDS projection ($M_{DC} = 0.9480$ with $(\alpha = 0.015, k = 275, T = 20)$ and a landmark ratio of 0.04) of the CPz QSO dataset. Samples are colored according to the labeling provided by the CPz dataset.

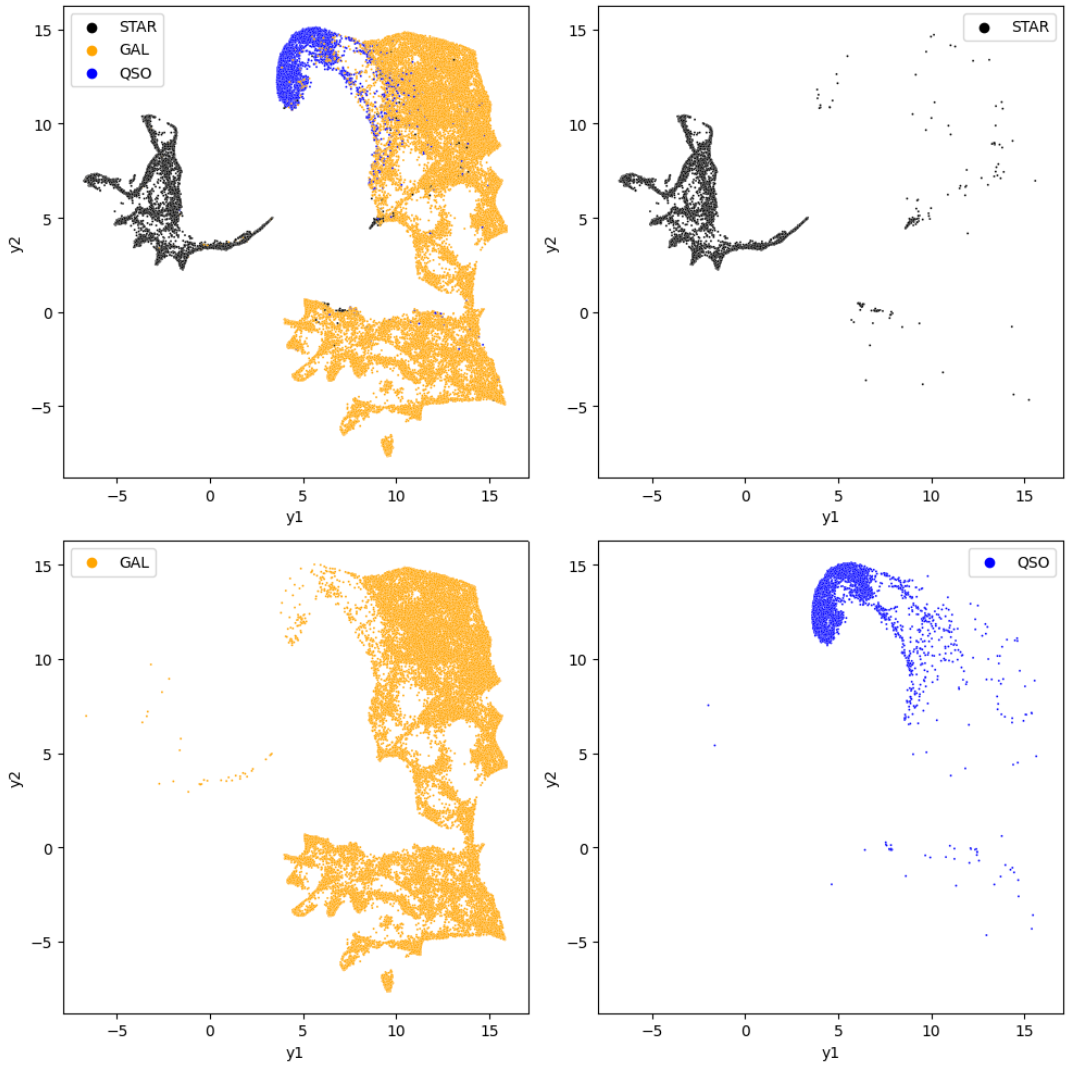


FIGURE A.16: The maximum distribution consistency sharpened UMAP projection ($M_{DC} = 0.9390$ with $(\alpha = 0.005, k = 225, T = 10)$ and ("metric": "euclidean", "min_dist": 0.1, "num_neighbors": 40, "umap_init": "spectral")) of the CPz QSO dataset. Samples are colored according to the labeling provided by the CPz dataset.

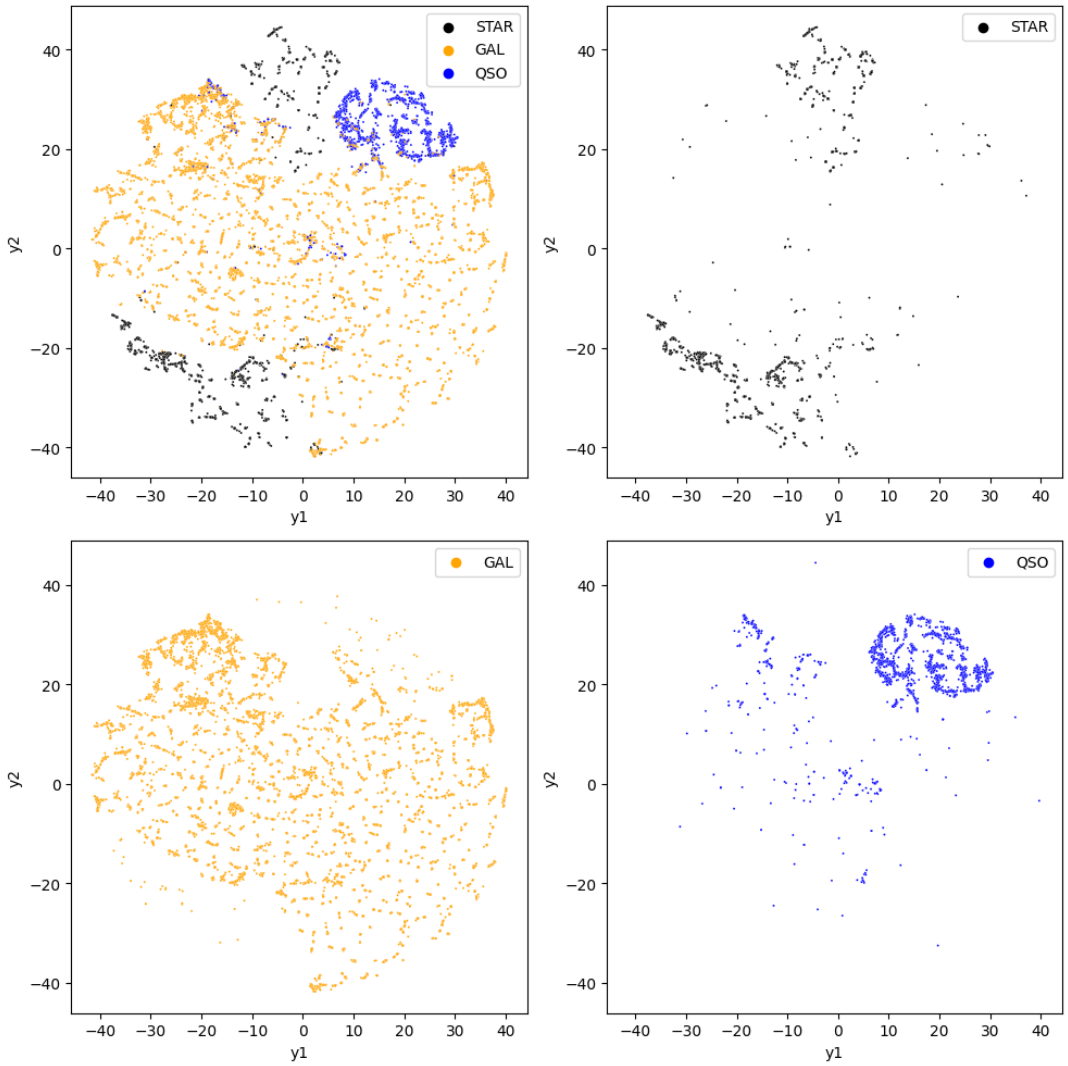


FIGURE A.17: The maximum distribution consistency sharpened tSNE projection ($M_{DC} = 0.9378$ with $(\alpha = 0.01, k = 25, T = 10)$ and ("sne_perplexity": 180, "sne_theta": 0.5)) of the CPz QSO dataset. Samples are colored according to the labeling provided by the CPz dataset.

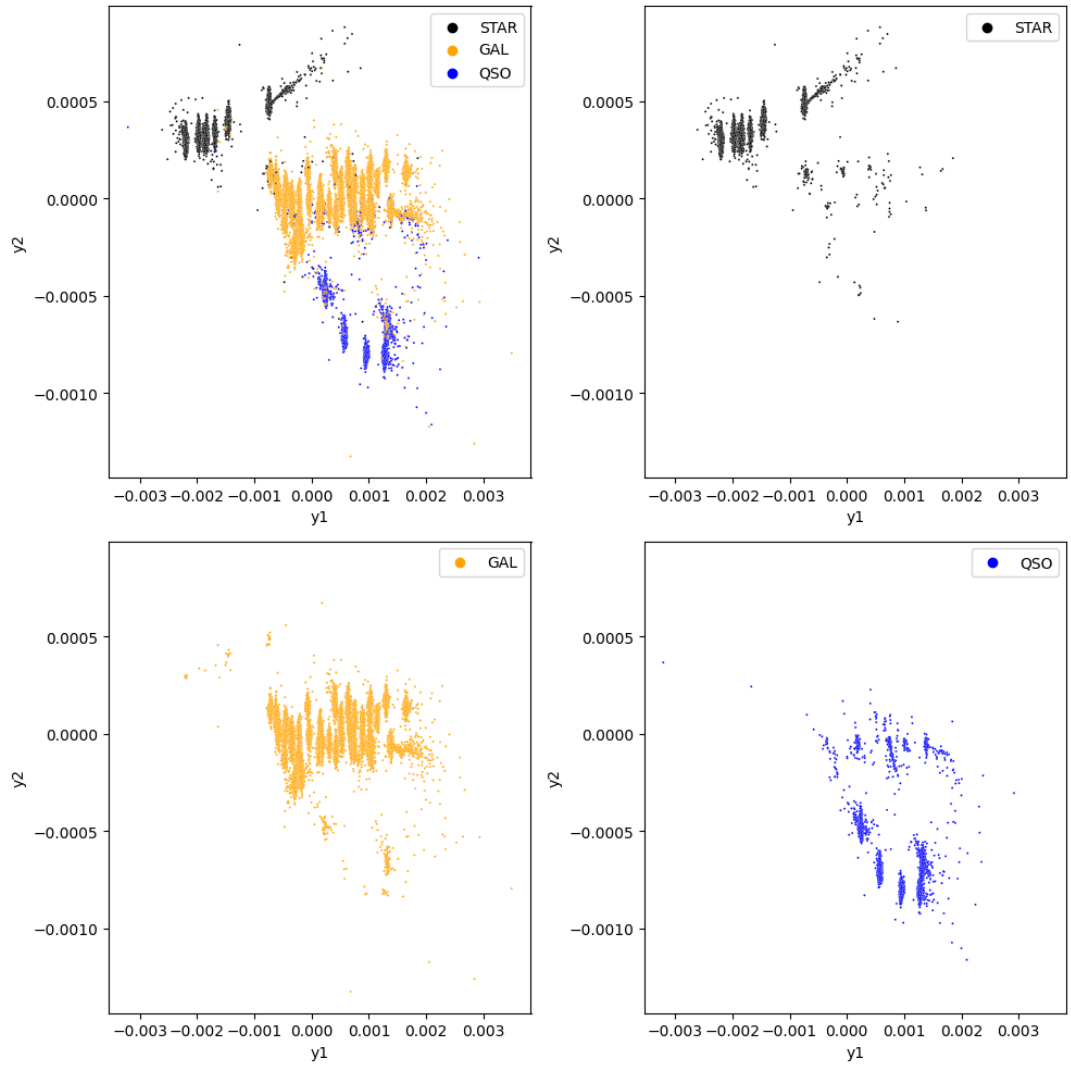


FIGURE A.18: The maximum distribution consistency sharpened NPE projection ($M_{DC} = 0.9389$ with $(\alpha = 0.03, k = 325, T = 10)$ and 80 nearest neighbors) of the CPz QSO dataset. Samples are colored according to the labeling provided by the CPz dataset.

Appendix B

Supplemental SDR-NNP Results

B.1 CPz GAL results

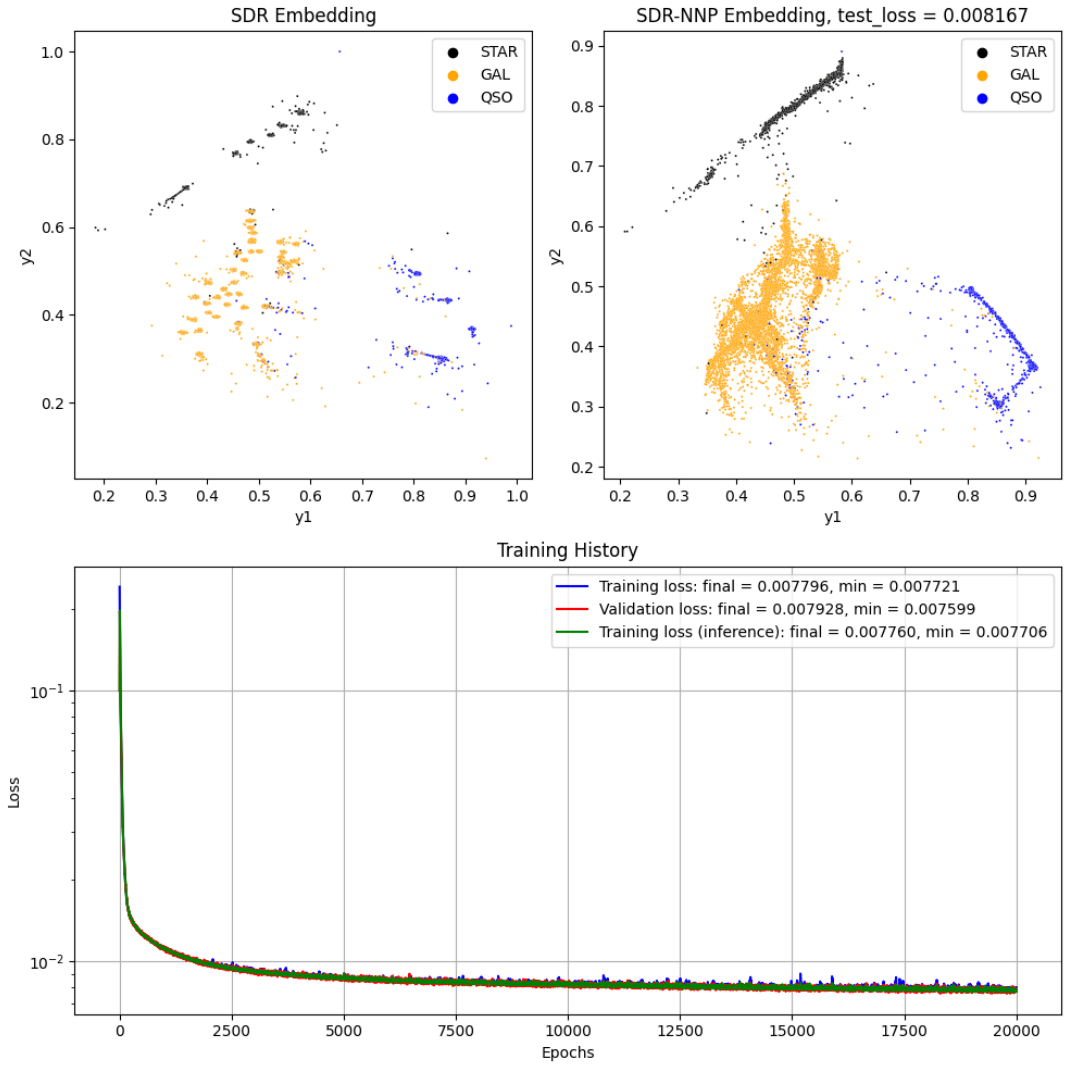


FIGURE B.1: NNP testing and training results for sharpened LMDS optimized for the CPz GAL dataset.

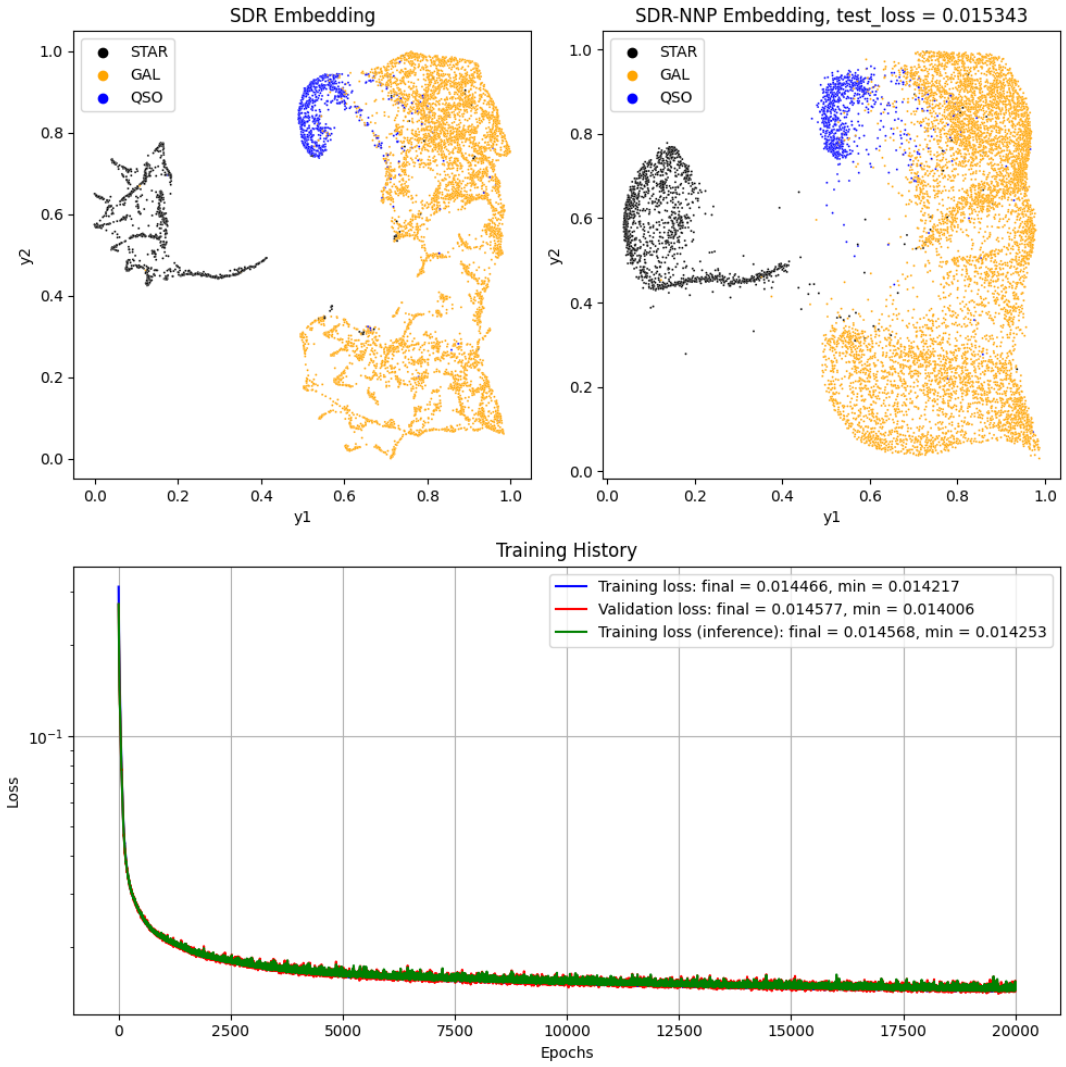


FIGURE B.2: NNP testing and training results for sharpened UMAP optimized for the CPz GAL dataset.

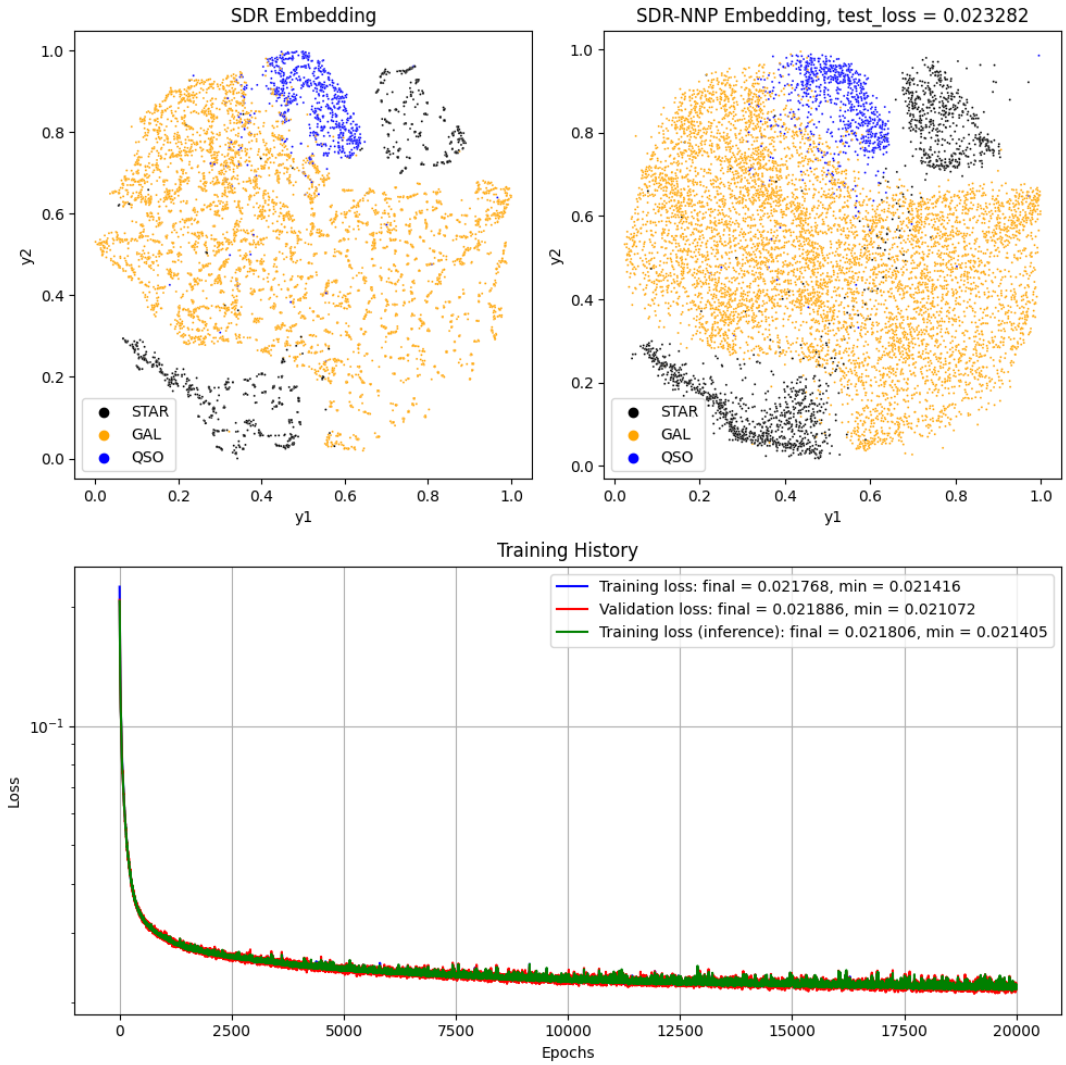


FIGURE B.3: NNP testing and training results for sharpened tSNE optimized for the CPz GAL dataset.

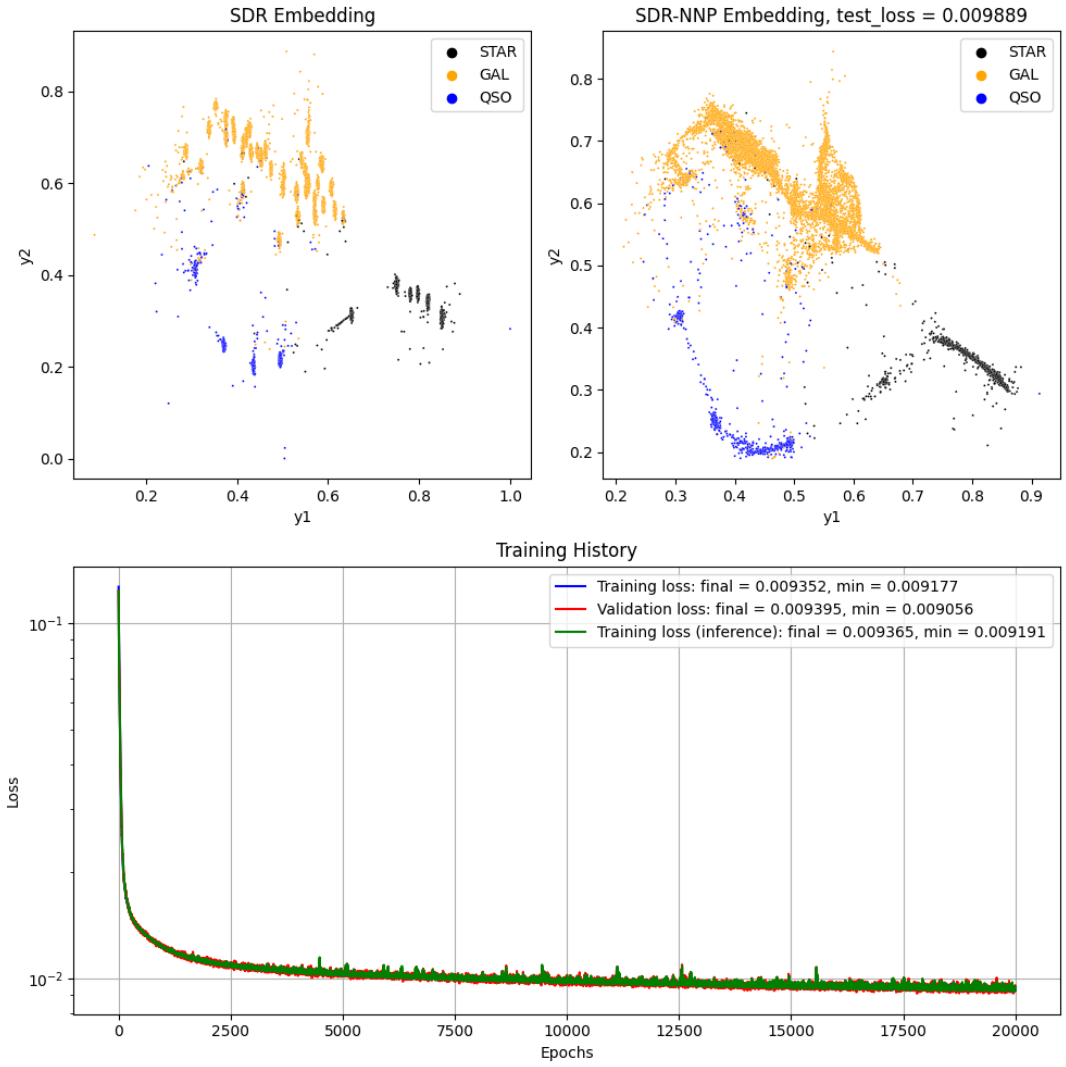


FIGURE B.4: NNP testing and training results for sharpened NPE optimized for the CPz GAL dataset.

B.2 CPz QSO results

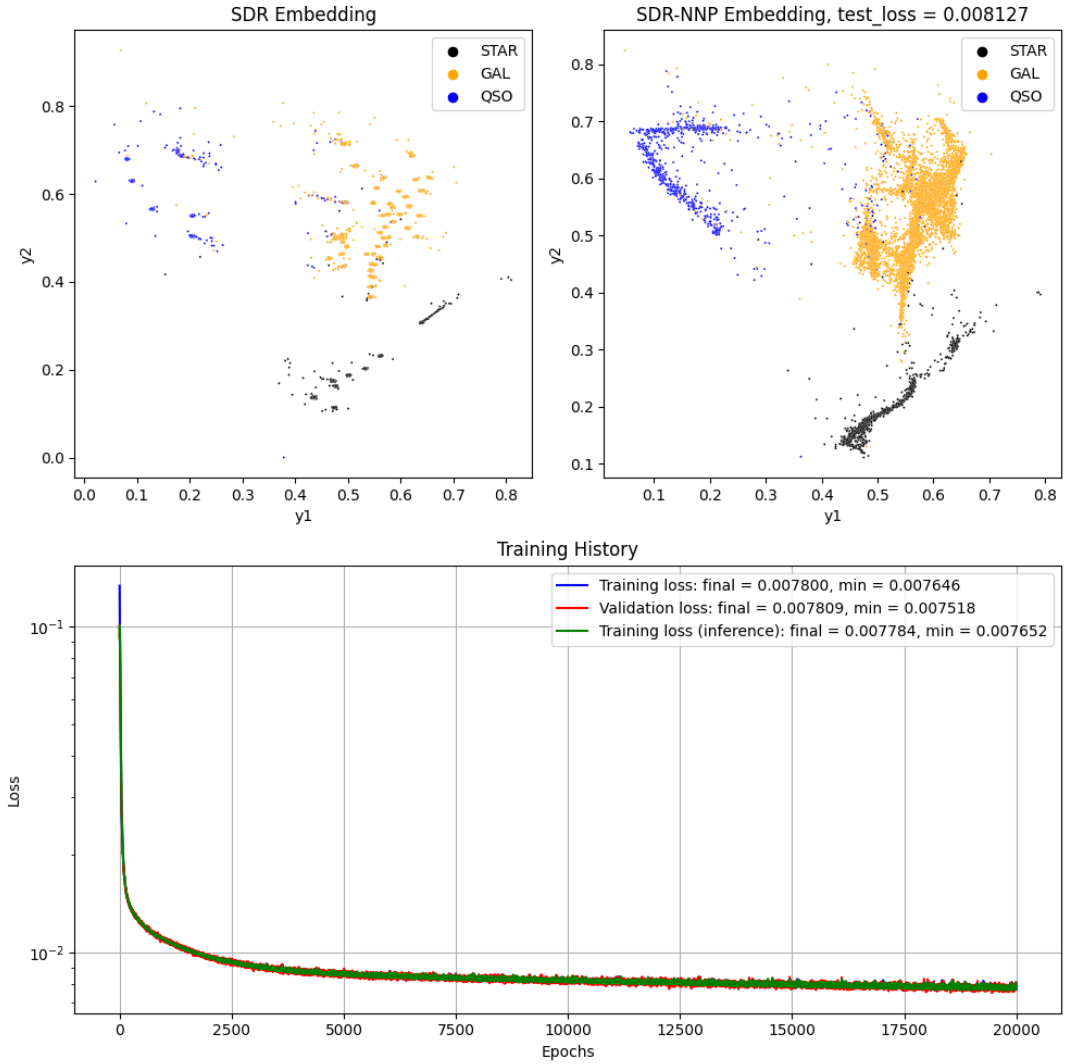


FIGURE B.5: NNP testing and training results for sharpened LMDS optimized for the CPz QSO dataset.

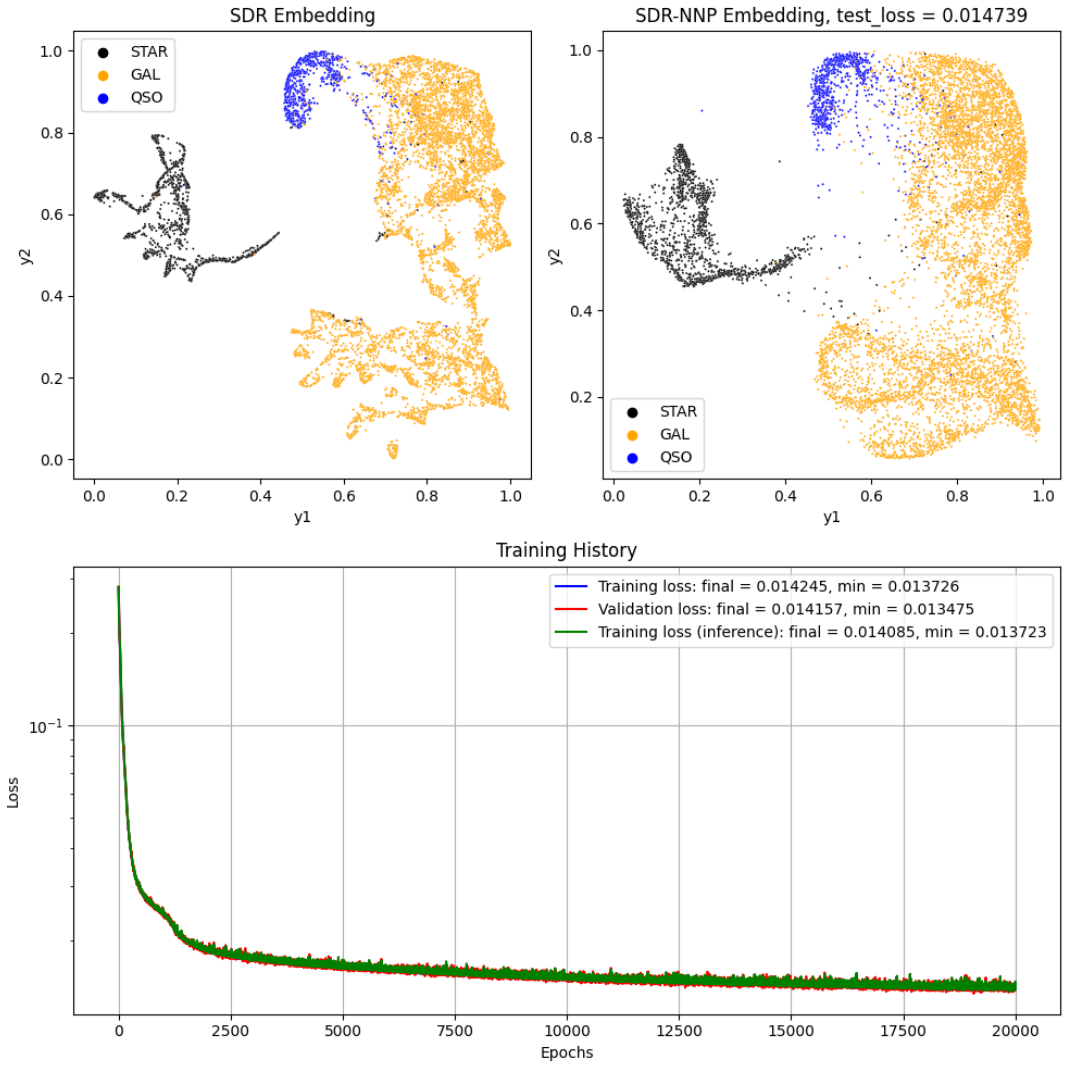


FIGURE B.6: NNP testing and training results for sharpened UMAP optimized for the CPz QSO dataset.

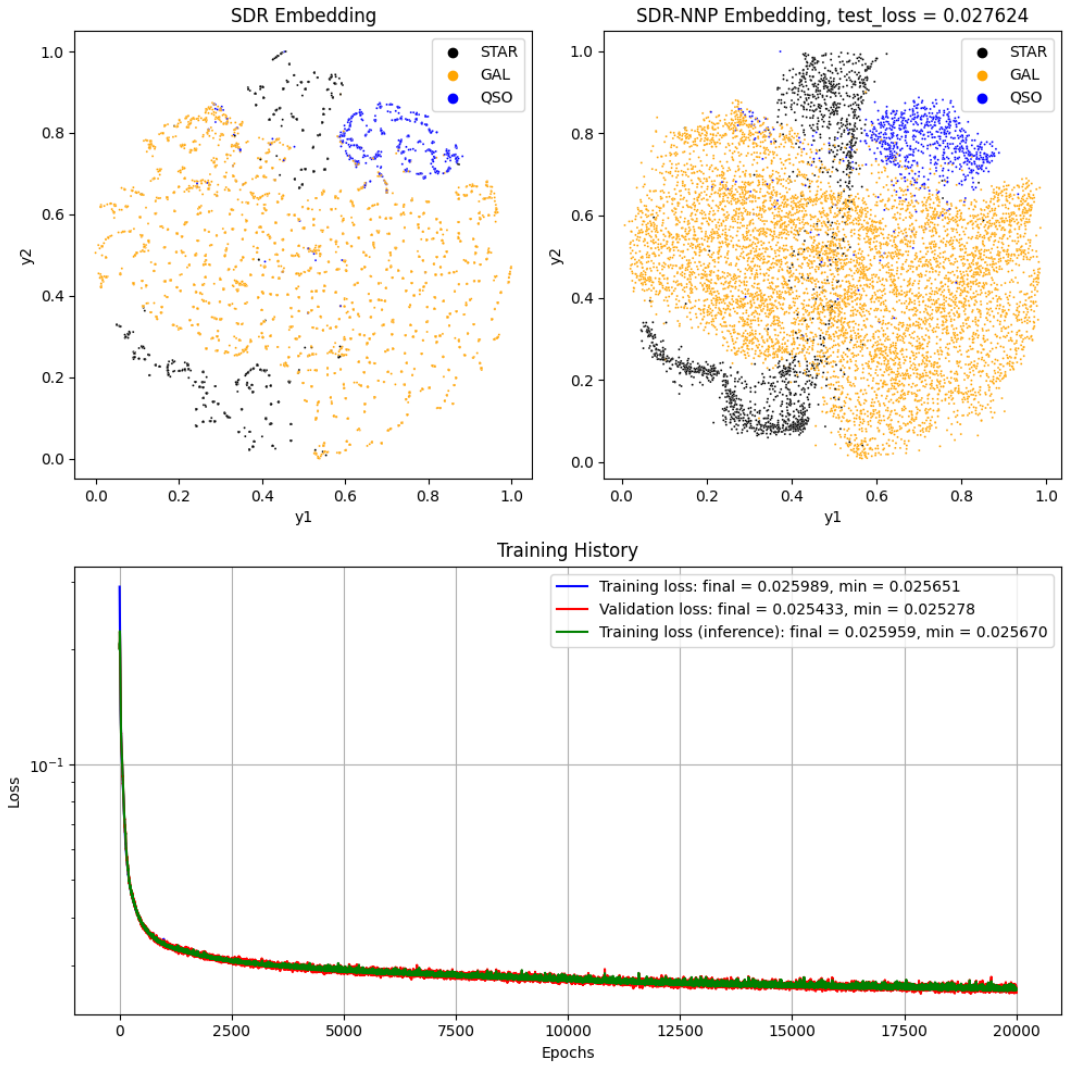


FIGURE B.7: NNP testing and training results for sharpened tSNE optimized for the CPz QSO dataset.

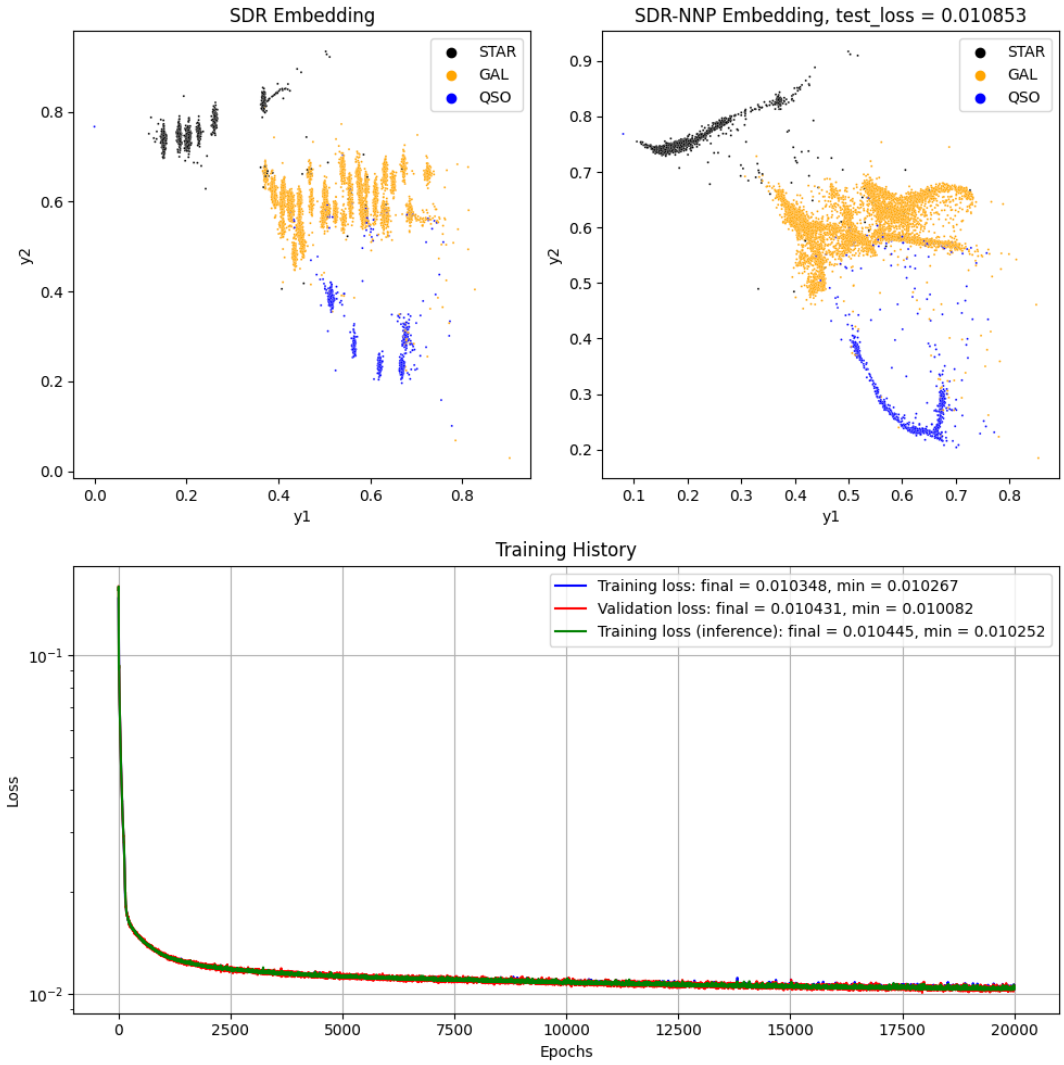


FIGURE B.8: NNP testing and training results for sharpened NPE optimized for the CPz QSO dataset.

Appendix C

Supplemental Classification Performance Results

C.1 CPz GAL results

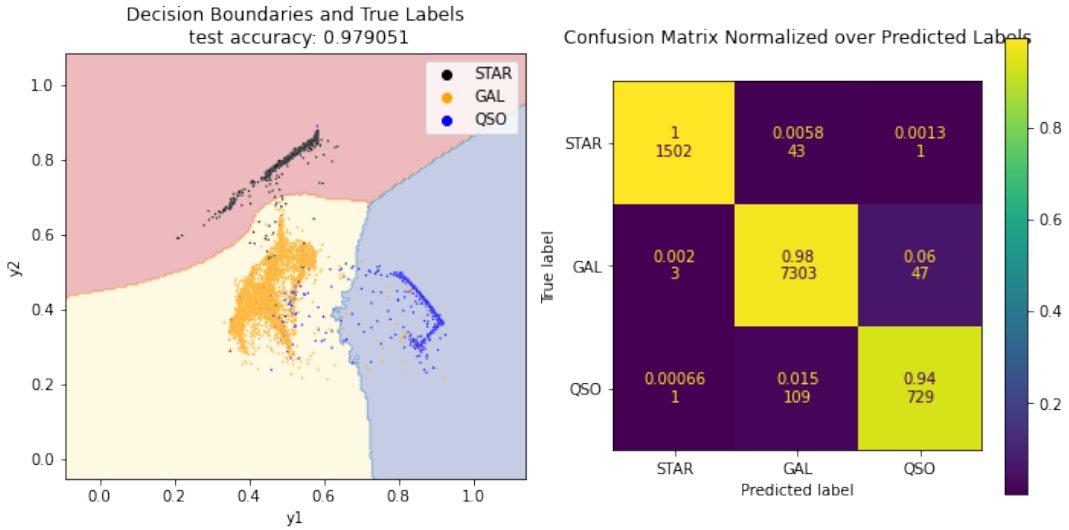


FIGURE C.1: The decision boundaries of the sharpened LMDS-NNP based KNN classifier of the CPz GAL dataset and its confusion matrix.

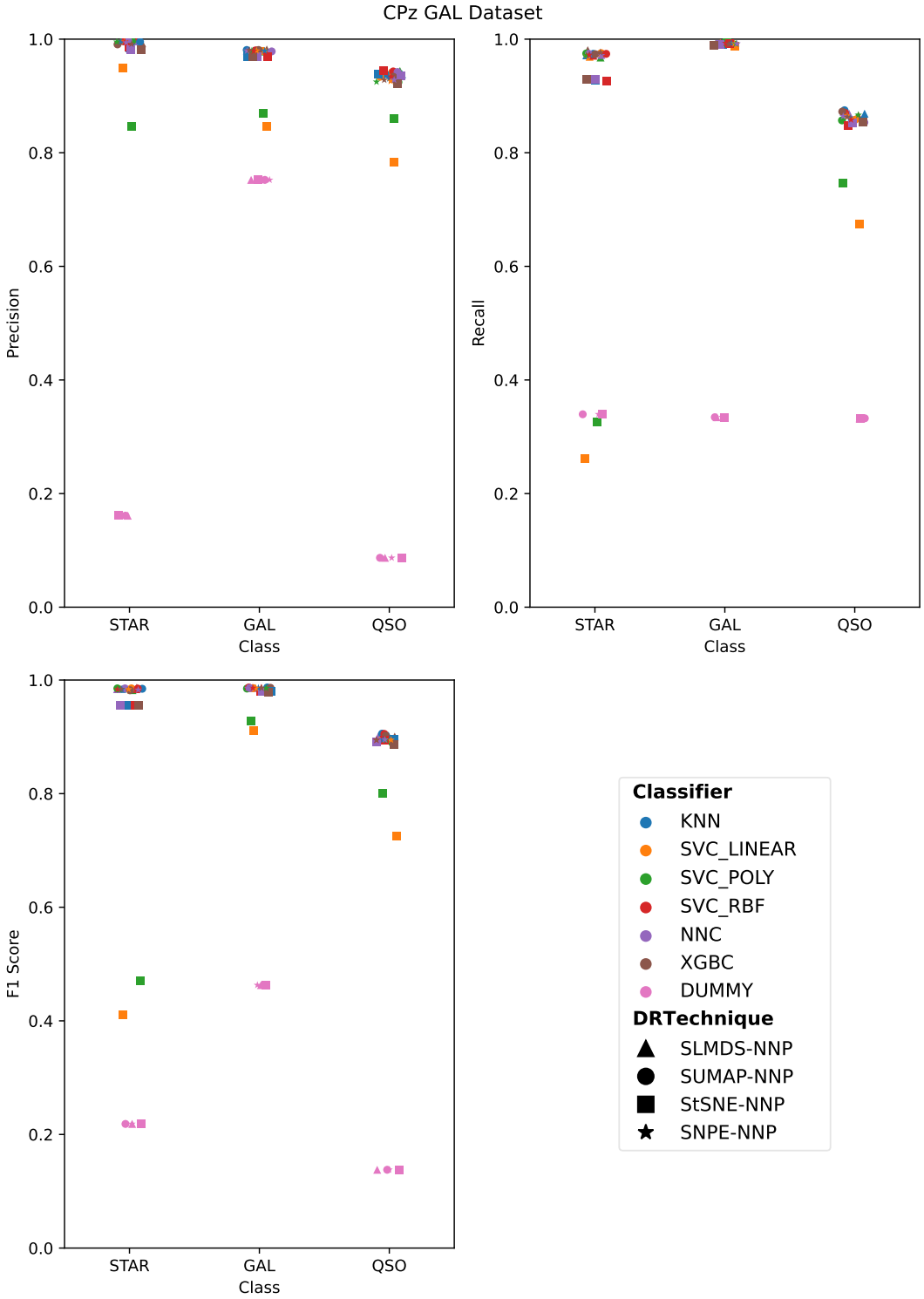


FIGURE C.2: The classification performance using the CPz GAL dataset in terms of precision, recall and F1 score for various combinations of DR technique and classifier. Note, the "DUMMY" classifier assigns classes randomly and gives a baseline above which any useful classifier should lie.

C.2 CPz QSO results

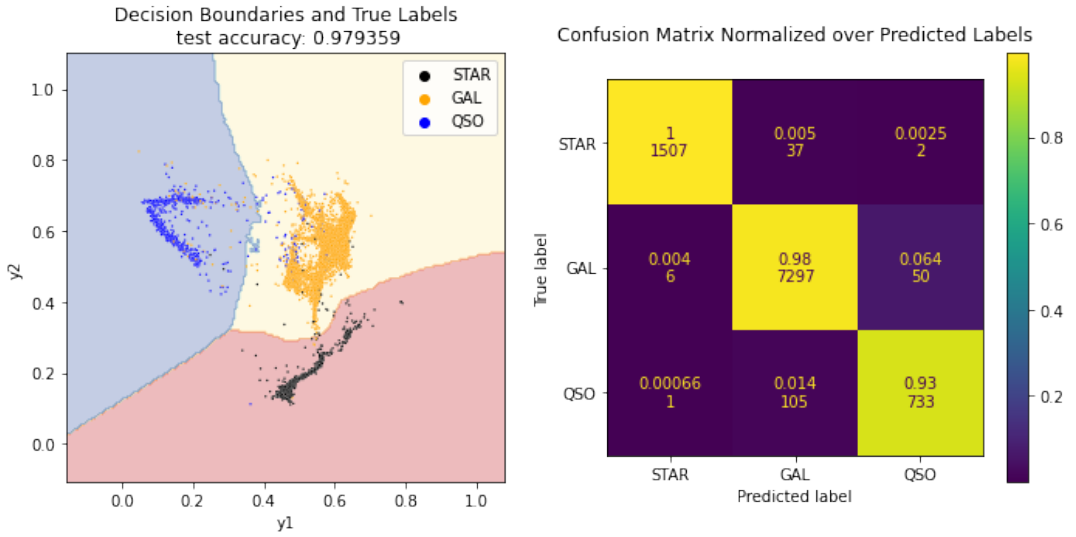


FIGURE C.3: The decision boundaries of the sharpened LMDS-NNP based KNN classifier of the CPz QSO dataset and its confusion matrix.

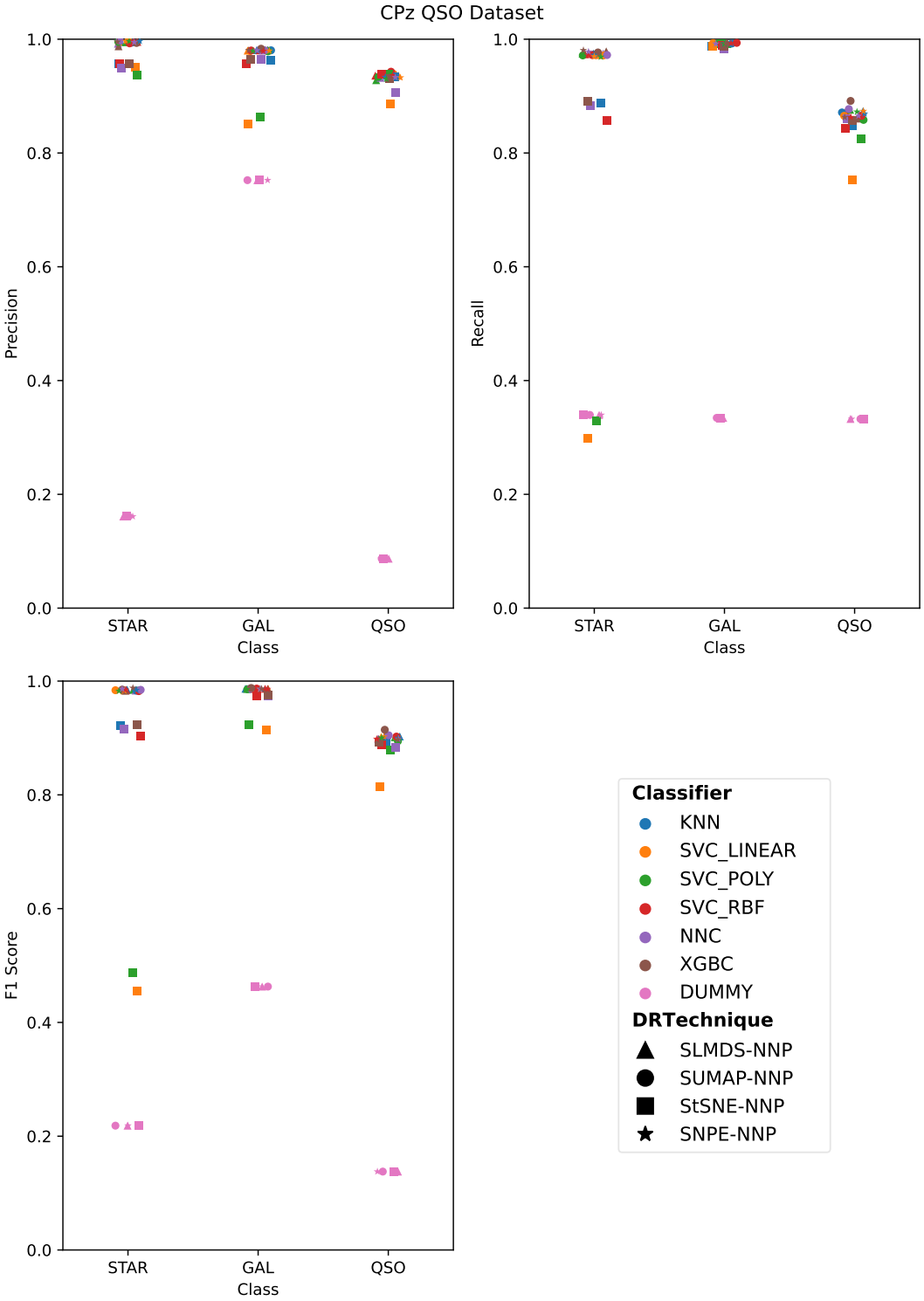


FIGURE C.4: The classification performance using the CPz QSO dataset in terms of precision, recall and F1 score for various combinations of DR technique and classifier. Note, the "DUMMY" classifier assigns classes randomly and gives a baseline above which any useful classifier should lie.

Bibliography

A Dictionary of Statistics (2008). Oxford University Press. ISBN: 9780191726866. DOI: 10.1093/acref/9780199541454.001.0001. URL: https://www.oxfordreference.com/view/10.1093/acref/9780199541454.001.0001/acref-9780199541454.

Abadi, Martín et al. (2015). *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. URL: https://www.tensorflow.org/.

Abazajian, Kevork N. et al. (May 2009). "THE SEVENTH DATA RELEASE OF THE SLOAN DIGITAL SKY SURVEY". In: *The Astrophysical Journal Supplement Series* 182.2, p. 543. DOI: 10.1088/0067-0049/182/2/543. URL: https://dx.doi.org/10.1088/0067-0049/182/2/543.

Adelman-McCarthy, Jennifer K. et al. (Apr. 2008). "The Sixth Data Release of the Sloan Digital Sky Survey". In: *The Astrophysical Journal Supplement Series* 175.2, p. 297. DOI: 10.1086/524984. URL: https://dx.doi.org/10.1086/524984.

Alam, Shadab et al. (July 2015). "THE ELEVENTH AND TWELFTH DATA RELEASES OF THE SLOAN DIGITAL SKY SURVEY: FINAL DATA FROM SDSS-III". In: *The Astrophysical Journal Supplement Series* 219.1, p. 12. DOI: 10.1088/0067-0049/219/1/12. URL: https://dx.doi.org/10.1088/0067-0049/219/1/12.

Arnaboldi, M. et al. (Mar. 2007). "ESO Public Surveys with the VST and VISTA". In: *The Messenger* 127, p. 28.

Bamford, Steven P. et al. (Feb. 2009). "Galaxy Zoo: the dependence of morphology and colour on environment*". In: *Monthly Notices of the Royal Astronomical Society* 393.4, pp. 1324–1352. ISSN: 0035-8711. DOI: 10.1111/j.1365-2966.2008.14252.x. eprint: https://academic.oup.com/mnras/article-pdf/393/4/1324/17320867/mnras0393-1324.pdf. URL: https://doi.org/10.1111/j.1365-2966.2008.14252.x.

Belkin, Mikhail and Partha Niyogi (2001). "Laplacian Eigenmaps and Spectral Techniques for Embedding and Clustering". In: *Advances in Neural Information Processing Systems*. Ed. by T. Dietterich, S. Becker, and Z. Ghahramani. Vol. 14. MIT Press. URL: https://proceedings.neurips.cc/paper_files/paper/2001/file/f106b7f99d2cb30c3db1c3cc0fde9ccb-Paper.pdf.

Bertin, E. and S. Arnouts (June 1996). "SExtractor: Software for source extraction." In: A&AS 117, pp. 393–404. DOI: 10.1051/aas:1996164.

Bianchi, Luciana, Alberto Conti, and Bernie Shiao (2014). "The ultraviolet sky: An overview from the GALEX surveys". In: *Advances in Space Research* 53.6. Stars, Galaxies and Star Formation History in the UV, pp. 900–912. ISSN: 0273-1177. DOI: https://doi.org/10.1016/j.asr.2013.07.045. URL: https://www.sciencedirect.com/science/article/pii/S0273117713004742.

Binder, Patricia, Michael Muma, and Abdelhak M. Zoubir (2018). "Gravitational Clustering: A simple, robust and adaptive approach for distributed networks". In: *Signal Processing* 149, pp. 36–48. ISSN: 0165-1684. DOI: 10.1016/j.sigpro.2018.02.034. URL: https://www.sciencedirect.com/science/article/pii/S0165168418300902.

Campello, Ricardo J. G. B., Davoud Moulavi, and Joerg Sander (2013). "Density-Based Clustering Based on Hierarchical Density Estimates". In: *Advances in Knowledge Discovery and Data Mining*. Ed. by Jian Pei et al. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 160–172. ISBN: 978-3-642-37456-2.

Cannon, Annie Jump and Edward Charles Pickering (Jan. 1912). "Classification of 1,688 southern stars by means of their spectra". In: *Annals of Harvard College Observatory* 56.5, pp. 115–164.

Cardelli, Jason A., Geoffrey C. Clayton, and John S. Mathis (Oct. 1989). "The Relationship between Infrared, Optical, and Ultraviolet Extinction". In: ApJ 345, p. 245. DOI: 10.1086/167900.

Chang, Chih-Chung and Chih-Jen Lin (May 2011). "LIBSVM: A Library for Support Vector Machines". In: *ACM Trans. Intell. Syst. Technol.* 2.3. ISSN: 2157-6904. DOI: 10.1145/1961189.1961199. URL: https://doi.org/10.1145/1961189.1961199.

Chang, Yu-Yen et al. (July 2015). "STELLAR MASSES AND STAR FORMATION RATES FOR 1 M GALAXIES FROM SDSS+WISE". In: *The Astrophysical Journal Supplement Series* 219.1, p. 8. DOI: 10.1088/0067-0049/219/1/8. URL: https://dx.doi.org/10.1088/0067-0049/219/1/8.

Chen, Tianqi and Carlos Guestrin (2016). "XGBoost: A Scalable Tree Boosting System". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '16. San Francisco, California, USA: ACM, pp. 785–794. ISBN: 978-1-4503-4232-2. DOI: 10.1145/2939672.2939785. URL: http://doi.acm.org/10.1145/2939672.2939785.

Cheng, Yizong (1995). "Mean shift, mode seeking, and clustering". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 17.8, pp. 790–799. DOI: 10.1109/34.400568.

Clarke, A. O. et al. (2020). "Identifying galaxies, quasars, and stars with machine learning: A new catalogue of classifications for 111 million SDSS sources without spectra". In: *A&A* 639, A84. DOI: 10.1051/0004-6361/201936770. URL: https://doi.org/10.1051/0004-6361/201936770.

Coil, Alison L. et al. (Oct. 2011). "THE PRISM MULTI-OBJECT SURVEY (PRIMUS). I. SURVEY OVERVIEW AND CHARACTERISTICS". In: *The Astrophysical Journal* 741.1, p. 8. DOI: 10.1088/0004-637X/741/1/8. URL: https://dx.doi.org/10.1088/0004-637X/741/1/8.

Cool, Richard J. et al. (Apr. 2013). "THE PRISM MULTI-OBJECT SURVEY (PRIMUS). II. DATA REDUCTION AND REDSHIFT FITTING". In: *The Astrophysical Journal* 767.2, p. 118. DOI: 10.1088/0004-637X/767/2/118. URL: https://dx.doi.org/10.1088/0004-637X/767/2/118.

Cox, Michael A. A. and Trevor F. Cox (2008). "Multidimensional Scaling". In: *Handbook of Data Visualization*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 315–347. ISBN: 978-3-540-33037-0. DOI: 10.1007/978-3-540-33037-0_14. URL: https://doi.org/10.1007/978-3-540-33037-0_14.

Creevey, O. L. et al. (June 2022). "Gaia Data Release 3: Astrophysical parameters inference system (Apsis) I – methods and content overview". In: *arXiv e-prints*, arXiv:2206.05864, arXiv:2206.05864. DOI: 10.48550/arXiv.2206.05864. arXiv: 2206.05864 [astro-ph.GA].

Cutri, R. M. et al. (Nov. 2013). *Explanatory Supplement to the AllWISE Data Release Products*. Explanatory Supplement to the AllWISE Data Release Products, by R. M. Cutri et al.

de Jong, Jelte T. A. et al. (2015). "The first and second data releases of the Kilo-Degree Survey". In: *A&A* 582, A62. DOI: 10.1051/0004-6361/201526601. URL: https://doi.org/10.1051/0004-6361/201526601.

De Silva, Vin and Joshua B Tenenbaum (2004). *Sparse multidimensional scaling using landmark points*. Tech. rep. technical report, Stanford University.

Donoho, David L. and Carrie Grimes (2003). "Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data". In: *Proceedings of the National Academy of Sciences* 100.10, pp. 5591–5596. DOI: 10.1073/pnas.1031596100. eprint: https://www.pnas.org/doi/pdf/10.1073/pnas.1031596100. URL: https://www.pnas.org/doi/abs/10.1073/pnas.1031596100.

Dubath, Pierre et al. (2016). "The Euclid Data Processing Challenges". In: *Proceedings of the International Astronomical Union* 12.S325, 73–82. DOI: 10.1017/S1743921317001521.

Epanechnikov, V. A. (1969). "Non-Parametric Estimation of a Multivariate Probability Density". In: *Theory of Probability & Its Applications* 14.1, pp. 153–158. DOI: 10.1137/1114019. eprint: https://doi.org/10.1137/1114019. URL: https://doi.org/10.1137/1114019.

Espadoto, Mateus et al. (2021). "Toward a Quantitative Survey of Dimension Reduction Techniques". In: *IEEE Transactions on Visualization and Computer Graphics* 27.3, pp. 2153–2173. DOI: 10.1109/TVCG.2019.2944182.

Ester, Martin et al. (1996). "A density-based algorithm for discovering clusters in large spatial databases with noise". In: *kdd*. Vol. 96. 34, pp. 226–231.

Fotopoulou, S. and S. Paltani (2018). "CPz: Classification-aided photometric-redshift estimation". In: *A&A* 619, A14. DOI: 10.1051/0004-6361/201730763. URL: https://doi.org/10.1051/0004-6361/201730763.

Gaia Collaboration et al. (2016). "The Gaia mission". In: *A&A* 595, A1. DOI: 10.1051/0004-6361/201629272. URL: https://doi.org/10.1051/0004-6361/201629272.

Gaia Collaboration et al. (July 2022). "Gaia Data Release 3: Summary of the content and survey properties". In: *arXiv e-prints*, arXiv:2208.00211, arXiv:2208.00211. DOI: 10.48550/arXiv.2208.00211. arXiv: 2208.00211 [astro-ph.GA].

Garilli, B. et al. (2014). "The VIMOS Public Extragalactic Survey (VIPERS) - First Data Release of 57 spectroscopic measurements". In: *A&A* 562, A23. DOI: 10.1051/0004-6361/201322790. URL: https://doi.org/10.1051/0004-6361/201322790.

Gashler, Michael, Dan Ventura, and Tony Martinez (2007). "Iterative Non-linear Dimensionality Reduction with Manifold Sculpting". In: *Advances in Neural Information Processing Systems*. Ed. by J. Platt et al. Vol. 20. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper_files/paper/2007/file/c06d06da9666a219db15cf575aff2824-Paper.pdf.

He, Xiaofei et al. (2005). "Neighborhood preserving embedding". In: *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*. Vol. 2, 1208–1213 Vol. 2. DOI: 10.1109/ICCV.2005.167.

Hubble, E. P. (Dec. 1926). "Extragalactic nebulae." In: ApJ 64, pp. 321–369. DOI: 10.1086/143018.

Hudelot, P. et al. (2013). *The CFHTLS Survey (T0007 release)*. Version Version 21-Apr-2016 (last modified).

Ioffe, Sergey and Christian Szegedy (2015). "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". In: *CoRR* abs/1502.03167. arXiv: 1502.03167. URL: http://arxiv.org/abs/1502.03167.

Jaccard, Paul (Jan. 1902). "Lois de distribution florale dans la zone alpine". In: *Bulletin de la Société vaudoise des sciences naturelles* 38, pp. 69–130. DOI: 10.5169/seals-266762.

Jones, D. Heath et al. (Dec. 2004). "The 6dF Galaxy Survey: samples, observational techniques and the first data release". In: *Monthly Notices of the Royal Astronomical*

Society 355.3, pp. 747–763. ISSN: 0035-8711. DOI: 10.1111/j.1365-2966.2004.08353.x. eprint: https://academic.oup.com/mnras/article-pdf/355/3/747/2797044/355-3-747.pdf. URL: https://doi.org/10.1111/j.1365-2966.2004.08353.x.

Jones, D. Heath et al. (Oct. 2009). "The 6dF Galaxy Survey: final redshift release (DR3) and southern large-scale structures". In: *Monthly Notices of the Royal Astronomical Society* 399.2, pp. 683–698. ISSN: 0035-8711. DOI: 10.1111/j.1365-2966.2009.15338.x. eprint: https://academic.oup.com/mnras/article-pdf/399/2/683/3630880/mnras0399-0683.pdf. URL: https://doi.org/10.1111/j.1365-2966.2009.15338.x.

Kim, Youngjoo et al. (Feb. 2022a). "SDR-NNP: Sharpened Dimensionality Reduction with Neural Networks". English. In: *Proceedings of the 17th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*. Vol. 3. SciTePress, pp. 63–76. DOI: 10.5220/0010820900003124.

Kim, Youngjoo et al. (2022b). "Visual cluster separation using high-dimensional sharpened dimensionality reduction". In: *Information Visualization* 21.3, pp. 197–219. DOI: 10.1177/14738716221086589. eprint: https://doi.org/10.1177/14738716221086589. URL: https://doi.org/10.1177/14738716221086589.

Kingma, Diederik P. and Jimmy Ba (Dec. 2014). "Adam: A Method for Stochastic Optimization". In: *arXiv e-prints*, arXiv:1412.6980, arXiv:1412.6980. DOI: 10.48550/arXiv.1412.6980. arXiv: 1412.6980 [cs.LG].

Le Fèvre, O. et al. (2013). "The VIMOS VLT Deep Survey final data release: a spectroscopic sample of 35 galaxies and AGN out to z 6.7 selected with 17.5 <= iAB <= 24.75". In: *A&A* 559, A14. DOI: 10.1051/0004-6361/201322179. URL: https://doi.org/10.1051/0004-6361/201322179.

Lintott, Chris et al. (Dec. 2010). "Galaxy Zoo 1: data release of morphological classifications for nearly 900 000 galaxies*". In: *Monthly Notices of the Royal Astronomical Society* 410.1, pp. 166–178. ISSN: 0035-8711. DOI: 10.1111/j.1365-2966.2010.17432.x. eprint: https://academic.oup.com/mnras/article-pdf/410/1/166/18442057/mnras0410-0166.pdf. URL: https://doi.org/10.1111/j.1365-2966.2010.17432.x.

Lintott, Chris J. et al. (Sept. 2008). "Galaxy Zoo: morphologies derived from visual inspection of galaxies from the Sloan Digital Sky Survey*". In: *Monthly Notices of the Royal Astronomical Society* 389.3, pp. 1179–1189. ISSN: 0035-8711. DOI: 10.1111/j.1365-2966.2008.13689.x. eprint: https://academic.oup.com/mnras/article-pdf/389/3/1179/3325962/mnras0389-1179.pdf. URL: https://doi.org/10.1111/j.1365-2966.2008.13689.x.

Lisitsyn, Sergey, Christian Widmer, and Fernando J. Iglesias Garcia (2013). "Tapkee: An Efficient Dimension Reduction Library". In: *Journal of Machine Learning Research* 14.72, pp. 2355–2359. URL: http://jmlr.org/papers/v14/lisitsyn13a.html.

Liske, J. et al. (July 2015). "Galaxy And Mass Assembly (GAMA): end of survey report and data release 2". In: *Monthly Notices of the Royal Astronomical Society* 452.2, pp. 2087–2126. ISSN: 0035-8711. DOI: 10.1093/mnras/stv1436. eprint: https://academic.oup.com/mnras/article-pdf/452/2/2087/18508439/stv1436.pdf. URL: https://doi.org/10.1093/mnras/stv1436.

Logan, C. H. A. and S. Fotopoulou (2020). "Unsupervised star, galaxy, QSO classification - Application of HDBSCAN". In: *A&A* 633, A154. DOI: 10.1051/0004-6361/201936648. URL: https://doi.org/10.1051/0004-6361/201936648.

Maaten, Laurens van der and Geoffrey Hinton (2008). "Visualizing Data using t-SNE". In: *Journal of Machine Learning Research* 9.86, pp. 2579–2605. URL: http://jmlr.org/papers/v9/vandermaaten08a.html.

Mainzer, A. et al. (Mar. 2011). "PRELIMINARY RESULTS FROM NEOWISE: AN ENHANCEMENT TO THE WIDE-FIELD INFRARED SURVEY EXPLORER FOR SOLAR SYSTEM SCIENCE". In: *The Astrophysical Journal* 731.1, p. 53. DOI: 10.1088/0004-637X/731/1/53. URL: https://dx.doi.org/10.1088/0004-637X/731/1/53.

McInnes, Leland, John Healy, and James Melville (2020). *UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction*. arXiv: 1802.03426 [stat.ML].

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

Prim, R. C. (1957). "Shortest connection networks and some generalizations". In: *The Bell System Technical Journal* 36.6, pp. 1389–1401. DOI: 10.1002/j.1538-7305.1957.tb01515.x.

Roweis, Sam T. and Lawrence K. Saul (2000). "Nonlinear Dimensionality Reduction by Locally Linear Embedding". In: *Science* 290.5500, pp. 2323–2326. DOI: 10.1126/science.290.5500.2323. eprint: https://www.science.org/doi/pdf/10.1126/science.290.5500.2323. URL: https://www.science.org/doi/abs/10.1126/science.290.5500.2323.

Schlegel, David J., Douglas P. Finkbeiner, and Marc Davis (June 1998). "Maps of Dust Infrared Emission for Use in Estimation of Reddening and Cosmic Microwave Background Radiation Foregrounds". In: *The Astrophysical Journal* 500.2, p. 525. DOI: 10.1086/305772. URL: https://dx.doi.org/10.1086/305772.

Schmidt, M. (Mar. 1963). "3C 273 : A Star-Like Object with Large Red-Shift". In: Nature 197.4872, p. 1040. DOI: 10.1038/1971040a0.

Silva, Vin and Joshua Tenenbaum (2002). "Global Versus Local Methods in Nonlinear Dimensionality Reduction". In: *Advances in Neural Information Processing Systems*. Ed. by S. Becker, S. Thrun, and K. Obermayer. Vol. 15. MIT Press. URL: https://proceedings.neurips.cc/paper_files/paper/2002/file/5d6646aad9bcc0be55b2c82f69750387-Paper.pdf.

Sips, Mike et al. (2009). "Selecting good views of high-dimensional data using class consistency". In: *Computer Graphics Forum* 28.3, pp. 831–838. DOI: 10.1111/j.1467-8659.2009.01467.x. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1467-8659.2009.01467.x. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-8659.2009.01467.x.

Vasconcellos, E. C. et al. (June 2011). "Decision Tree Classifiers for Star/Galaxy Separation". In: AJ 141.6, 189, p. 189. DOI: 10.1088/0004-6256/141/6/189. arXiv: 1011.1951 [astro-ph.CO].

Venna, Jarkko and Samuel Kaski (2001). "Neighborhood Preservation in Nonlinear Projection Methods: An Experimental Study". In: *Artificial Neural Networks — ICANN 2001*. Ed. by Georg Dorffner, Horst Bischof, and Kurt Hornik. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 485–491. ISBN: 978-3-540-44668-2. DOI: 10.1007/3-540-44668-0_68.

Verro, K. et al. (2022). "Modelling simple stellar populations in the near-ultraviolet to near-infrared with the X-shooter Spectral Library (XSL)". In: *A&A* 661, A50. DOI: 10.1051/0004-6361/202142387. URL: https://doi.org/10.1051/0004-6361/202142387.

Wright, Edward L. et al. (Nov. 2010). "THE WIDE-FIELD INFRARED SURVEY EXPLORER (WISE): MISSION DESCRIPTION AND INITIAL ON-ORBIT PERFORMANCE". In: *The Astronomical Journal* 140.6, p. 1868. DOI: 10.1088/0004-6256/140/6/1868. URL: https://dx.doi.org/10.1088/0004-6256/140/6/1868.

Zhang, Tianhao et al. (2007). "Linear local tangent space alignment and application to face recognition". In: *Neurocomputing* 70.7. Advances in Computational Intelligence and Learning, pp. 1547–1553. ISSN: 0925-2312. DOI: 10.1016/j.neucom.2006.11.007. URL: https://www.sciencedirect.com/science/article/pii/S0925231206004577.

Zhang, Zhenyue and Hongyuan Zha (2004). "Principal Manifolds and Nonlinear Dimensionality Reduction via Tangent Space Alignment". In: *SIAM Journal on Scientific Computing* 26.1, pp. 313–338. DOI: 10.1137/S1064827502419154. eprint: https://doi.org/10.1137/S1064827502419154. URL: https://doi.org/10.1137/S1064827502419154.