Evaluating the Efficacy of GAN-generated Synthetic Data in Enhancing CNN Performance for Imbalanced Classification of Astronomical Objects

Abhishek Mandloi

A dissertation submitted in partial fulfilment of the requirements of

Technological University Dublin for the degree of

M.Sc. in Computer Science (Data Science)

05 January 2024

Declaration

I certify that this dissertation which I now submit for examination for the award of MSc in Computer Science (Data Science), is entirely my own work and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the text of my work.

This dissertation was prepared according to the regulations for postgraduate study of the Technological University Dublin and has not been submitted in whole or part for an award in any other Institute or University.

The work reported on in this dissertation conforms to the principles and requirements of the Institute's guidelines for ethics in research.

Signed: Abhishek Mandloi

Date: 05 January 2024

Abstract

Astronomy, a field rich in data, faces significant challenges in accurately classifying celestial objects, particularly quasars, which are notoriously difficult to identify due to their rare occurrence and similarity to other objects. Addressing this challenge, this research focuses on the prevalent issue of class imbalance in astronomical datasets, specifically within the Sloan Digital Sky Survey (SDSS) DR18 dataset. The study delves into the realm of data augmentation, utilizing Generative Adversarial Networks (GANs) to balance the dataset and potentially improve the performance of Convolutional Neural Networks (CNNs) for classifying quasars, galaxies, and stars. By comparing a base CNN model trained on the original dataset with models trained on datasets augmented using various GAN architectures, the research aims to enhance the recognition of underrepresented quasars. The models are evaluated using accuracy, precision, recall, F1 score, and false negative rate, along with statistical testing through McNemar's test. Contrary to expectations, the augmented models did not outperform the base model, underscoring the challenges in generating synthetic data that adequately capture the complexities of astronomical data. This finding highlights the need for further advancements in GAN architectures and deep learning models specifically designed for astronomical data, offering significant insights and directions for future research in this field.

Keywords: SDSS, Quasars, Data Augmentation, GAN, CNN

Acknowledgments

I would like to express my sincere thanks to my supervisor, Brendan Tierney, for his assistance and motivation during the course of this dissertation.

I would also like to express gratitude to Dr. Robert Ross, who helped me formulate my research.

And a special thanks to my friends and family for their endless support.

Contents

List of Figures

List of Tables

List of Acronyms

SDSS	Sloan Digital Sky Survey
DR	Data Release
FNR	False Negative Rate
FPR	False Positive Rate
CNN	Convolutional Neural Network
GAN	Generative Adversial Network
vGAN	Vanilla Generative Adaptive Network
cGAN	Conditional Generative Adaptive Network
SMOTE	Synthetic Minority Oversampling Technique
QSO	Quasi Stellar Object (Quasar)
AGN	Active Galactic Nuclei
RF	Random Forest
SVM	Support Vector Machine
KNN	K-Nearest Neighbors
BOSS	Baryon Oscillation Spectroscopic Survey
GALEX	Galaxy Evolution Explorer
UKIDSS	UK Infrared Telescope Infrared Deep Sky Survey
WISE	Widefield Infrared Survey Explorer
BASS	Beijing-Arizona Sky Survey
LAMOST	Large Sky Area Multi-object Fiber Spectroscopic Telescope
XGBoost	Extreme Gradient Boosting
MLP	Multi Layer Perceptron

Chapter 1

Introduction

1.1 Background

In recent years, astronomy has made significant progress in analyzing vast amounts of observational data, particularly in classifying celestial objects like quasars, galaxies, and stars. Quasars, known as quasi-stellar radio sources, are highly luminous and distant astronomical objects powered by supermassive black holes at galaxy centers.

Machine learning techniques have been widely utilized to handle the complexity and volume of astronomical data. However, class imbalance in these datasets presents challenges, leading to biased predictions and reduced performance for underrepresented classes. The SDSS DR18 dataset, derived from the Sloan Digital Sky Survey, is extensively used in astronomy and suffers from a class imbalance issue, particularly with fewer instances of the quasar class.

Identifying quasars is challenging due to their rarity and resemblance to very bright stars or galaxies. Their distinct features, such as strong emission lines and non-stellar spectra, help differentiate them. Nevertheless, their extreme brightness often causes misclassification. The class imbalance results in the model prioritizing the majority class and struggling to identify quasar instances accurately, leading to a high False Negative Rate (FNR).

Reducing the FNR in astronomy is crucial to avoid overlooking significant celestial objects or events, ensuring comprehensive data for scientific breakthroughs and a deeper understanding of the universe. Addressing the FNR helps maintain accurate observations, preventing the oversight of critical phenomena and facilitating meaningful discoveries.

Exploring the application of neural networks, which have been underutilized in this context, is a promising avenue. By leveraging machine learning and tackling class imbalance issues, this research aims to enhance classification performance for rare astronomical objects. This research's scope is focused on developing a robust technique for reducing the FNR for the classification of quasars in the SDSS DR18 dataset using Deep Learning algorithms and data augmentation techniques and verifying if the introduction of augmented data for underrepresented classes can improve the classification performance of the model.

1.2 Research Project/Problem

This research tackles the problem of accurately classifying celestial objects, particularly quasars, in astronomy. The dataset being utilized suffers from a class imbalance issue, with fewer instances of the quasar class. This leads to biased predictions and reduced performance for the underrepresented class. Moreover, previous research has used older dataset releases, potentially limiting the generalizability of findings. This research aims to address these limitations by exploring data augmentation techniques and the application of neural networks. The goal is to improve the classification performance by reducing the False Negative Rate for the underrepresented class.

1.2.1 Research Question

Does the performance of a CNN statistically significantly improve when trained with GAN-generated synthetic data for underrepresented classes generated by multiple types of GANs compared to a CNN trained solely on the original imbalanced SDSS DR18 dataset of quasars, galaxies, and stars?

1.2.2 Research Hypothesis

The null hypothesis (H0) states that if synthetic data generated by multiple types of GANs for underrepresented classes `<Z>` is used to augment the SDSS DR18 dataset `<D>`, then the performance of a CNN `<X>` when tested with McNemar's test is statistically significantly same or less (p-value `>= 0.05`) than the performance of a CNN `<Xb>` trained on the original data, in terms of Precision, Recall, F1 score, Accuracy and False Negative Rate.

The alternate hypothesis (HA) posits that if synthetic data generated by multiple types of GANs for underrepresented classes `<Z>` is used to augment SDSS DR18 dataset `<D>` then the performance of a CNN `<X>` when tested with McNemar's test is statistically significantly improved (p-value `< 0.05`) than the performance of a CNN `<Xb>` trained on same input data without such augmentation, in terms of Precision, Recall, F1 score, Accuracy and False Negative Rate.

The hypothesis further specifies that the statistical significance will be determined by achieving a p-value of less than 0.05, indicating that any observed improvement would not occur by chance. By exploring the impact of synthetic data generated by various types of GANs on classification performance, the study aims to evaluate the potential effectiveness of this augmentation technique in enhancing the accuracy and reliability of the CNN classifier in classifying celestial objects within the field of astronomy.

The hypothesis is dissected into smaller components to provide a more detailed breakdown:

1. Is the synthetic data generated by the GANs significantly similar to the real data?

2. How does the use of synthetic data impact the performance of the CNN model `<X>`?

3. Does the utilization of various GAN types have a discernible impact on the performance? If yes, which GAN type yields the most significant performance

enhancement for the CNN <X> classifier?

4. Has the False Negative Rate for classifying Quasars improved, as it is one of the major goals of the research?

1.3 Research Objectives

The main objective of the research is to evaluate the performance of a Convolutional Neural Network (CNN) trained on synthetic data generated by multiple types of GANs in addressing the classification imbalance in the SDSS DR18 dataset. The primary focus of research is to use SDSS DR18 dataset to classify ”class” target variable which is a three class target, however the emphasis is on identifying the quasar class objects which are an underrepresented class amongst star and galaxy class objects.

The first general objective is data acquisition and pre-processing. In this stage, the SDSS DR18 dataset will be downloaded from the SDSS portal for offline manipulation. The dataset will undergo thorough pre-processing, including the identification and removal of null values with low cardinality, examination of feature formats and observation of feature correlations using visualization techniques. The pre-processed dataset, denoted as <J>, will be ready for further analysis.

The second general objective is to develop an augmentation pipeline, denoted as <Z>, using multiple types of GAN to generate synthetic data. The GANs are trained using the original SDSS DR18 dataset, focusing on minority classes. The generator network will create synthetic samples that resemble the real data, while the discriminator network will distinguish between real and synthetic data. The trained GANs will generate synthetic data by inputting random noise vectors into the generator network. The synthetic data will be compared with the original data to verify if it is similar to real data and then will be combined with the original dataset to create an augmented dataset <AD>. The quality of the synthetic data will be assessed using visualizations and using a Kolmogorov–Smirnov test to ensure its effectiveness.

The third general objective focuses on classifier training and evaluation. Training, test, and validation sets will be defined using repeated Monte Carlo sampling to ensure robustness. This process will be performed separately on preprocessed `<J>` and augmented datasets `<AD>`. The base CNN model denoted as `<Xb>`, will be trained on the training set of `<J>`. Simultaneously, the augmented CNN model, denoted as `<X>`, will be trained on the training set of `<AD>`. The training will be tuned using the respective validation sets to prevent overfitting. The performance of both models will be then evaluated on respective test sets using multiple evaluation metrics: Precision (to verify low FPR), Recall (to verify low FNR), F1-score (to verify balance between FPR and FNR), and accuracy as an extra metric to provide insights about correct classification rate. The metrics from all the classifiers will be compared to verify if there is any improvement in the performance of the X model compared to the Xb model.

The fourth general objective involves statistical analysis and drawing conclusions. This is an important step since if there is performance improvement in the X model, it needs to be verified whether this improvement is statistically significant or not. A McNemar's test will be conducted to compare the results of all the models to assess the significance of the results and validity of the hypothesis testing. If the p-value is less than `0.05%`, the results will be considered statistically significant.

The results obtained will be thoroughly analyzed, and ultimately, conclusions will be drawn regarding the effectiveness of using synthetic data generated by various GANs in improving classification performance.

1.4 Research Methodologies

The data for this research is secondary data since it is acquired from the Sloan Digital Sky Survey data archive. The dataset comprises 100,000 records extracted from the SDSS's Data Release 18. Each record is characterized by 42 distinct attributes and includes a single classification column that categorizes the record into one of three classes: Star, Galaxy, or Quasar.

The research being conducted is quantitative type, since it involves analysis of numerical data to verify the hypothesis. The features used for the classification include attributes like Right Ascension and Declination, Photometric Bands, Petrosian Radii, Petrosian Fluxes, and Petrosian half-light radii across five photometric bands. Additionally, it comprises measurements of object magnitudes obtained using the Point Spread Function (PSF) within the same five photometric bands, as well as information about the axis ratio derived from exponential fits to the light profiles of celestial objects observed in those photometric bands.

In this study, a deductive reasoning approach is followed, commencing with the theory that a CNN will perform better when supplied with augmented data for underrepresented classes generated by various GANs for classifying astronomical objects, thus addressing the issue of class imbalance. The primary focus is on improving the classification of the Quasar class by mitigating the high False Negative Rate. Then, a hypothesis will be formed which will be followed by an experiment to analyze the available data. Later on, a statistical analysis will be performed to verify the significance of the initial hypothesis and to decide whether it should be accepted or rejected.

In essence, this research undertook a well-structured and evidence-based methodology to investigate a particular research question and evaluate the hypothesis. This approach entailed the utilization of both pre-existing secondary data as well as extensive statistical analysis to derive findings and arrive at a definitive conclusion.

1.5 Scope and Limitations

The research assumes that the SDSS DR18 dataset is representative of the broader population of celestial objects and can serve as a reliable source for training and testing the models. Additionally, the study assumes the availability of accurate and reliable labels for the celestial objects in the dataset, as mislabeling could introduce errors in the model training and evaluation processes.

One of the main limitations is the class imbalance issue within the dataset, particularly with a smaller number of instances for the quasar class compared to other classes. This unavailability of sufficient data may affect the model's ability to accurately identify and classify quasars, leading to a higher FNR.

The study's delimitations include focusing primarily on improving the quasar classification performance within the context of the SDSS DR18 dataset. While other types of celestial objects will be included in the research, they may not receive as much emphasis or dedicated analysis compared to quasars. Additionally, the study delimits itself to the training and evaluation of only CNN classifiers, where augmented classifiers utilize synthetic data generated by various GANs as input, while the base model is trained solely on the original imbalanced dataset.

1.6 Document Outline

This document is organized into five chapters.

The 'Introduction' chapter lays down the foundation by providing background information about the astronomy domain and the challenges with classifying astronomical objects. This leads to the identification of the research question, and a hypothesis is formed to answer it. Then, the general and specific objectives of the research are outlined before the discussion of research methodologies. The chapter concludes by specifying the scope and limitations of the research and outlining the structure of the document.

The 'Literature Review' chapter delves into the existing body of knowledge, commencing with the discussion of the contributions made by the catalog papers, which are released along with every data release of the SDSS dataset. Then, machine learning and deep learning approaches followed by various researchers are discussed before moving on to the discussion of a combination of different approaches and some unorthodox approaches to solving the problem. Later on, the chapter summary is presented, and gaps in the literature are identified, which motivated this research.

The 'Experiment Design and Methodology' chapter details the research approach, starting with the data collection and preprocessing section, where the focus is on the data extraction process and the rationale is provided behind feature selection. Then, the hardware and software configuration of the experiment is discussed in the Experiment Setup subsection, enabling other researchers to reenact the experimental environment to repeat the research. Further, the data augmentation technique, i.e., various types of GANs and their training and architecture, is discussed along with the architecture and training of base and augmented classifier models before discussing the evaluation metrics used to evaluate their performance. The chapter concludes with the Hypothesis Testing subsection, which explores various statistical tests to corroborate the alternate hypothesis.

The 'Results, Evaluation and Discussion' explains the experiment undertaken and presents the findings. It includes Synthetic Data Quality Evaluation, where the results of the K-S test are discussed for fake data generated by various techniques. Later, the classifier model's evaluation results are discussed. Further on, McNemar's test evaluation is discussed to establish the statistical significance of improvement in performance. To conclude, the chapter summarizes the results and discusses their interpretation.

Finally, the 'Conclusion' chapter wraps up the document by summarizing the research, the key findings, and the contributions made by this research. It also discusses the limitations of this research and gives a direction for future research to overcome them.

Together, these chapters offer a comprehensive view of the research journey and its significance.

Chapter 2

Review of Existing Literature

This literature review is organized into six major sections. The first five sections focus on the contributions of catalog papers, machine learning, and deep learning approaches, the amalgamation of multiple techniques, and alternative research methodologies adopted by scientists. The last section summarizes the gaps in research, discusses the motivation behind the study, and presents the research question that guides this experiment.

2.1 Contributions of Catalog Papers

In this section, the invaluable contributions made by catalog papers will be delved into, which are traditionally a part of SDSS data releases. These papers serve as foundational references, providing comprehensive insights into the properties, classifications, and coordinates of celestial objects captured by the SDSS.

Pâris et al. (2017) presented the SDSS Data Release 12 Quasar catalog (DR12Q) from the Baryon Oscillation Spectroscopic Survey (BOSS) of the SDSS-III providing comprehensive information on quasars targeted during the survey and confirmed via visual inspection of the spectra. The catalog identifies 29,580 broad absorption line quasars and provides redshifts, FWHMs, and characteristics of quasars, as well as photometry, optical morphology, and selection criteria, along with information on optical variability and emission properties from other surveys.

Pâris et al. (2018) presented the SDSS Data Release 14 Quasar catalog (DR14Q) from the extended BOSS (eBOSS) of the SDSS-IV which includes spectroscopically targeted quasar candidates confirmed as quasars, with redshift measurements and various emission and absorption features. The catalog contains 526,356 quasars, with 144,046 new discoveries, detected over 9,376 deg2, providing a significant increase in the number of known quasars. The catalog combines automated procedures and visual inspection to confirm quasars, providing a reliable identification process while contributing to the ongoing efforts to study the luminosity function of quasars and their clustering at moderate scales.

Lyke et al. (2020) presented the final catalog of quasars from the SDSS-IV DR16 of the eBOSS which includes the largest selection of spectroscopically confirmed quasars to date, with a total of 750,414 quasars, including 225,082 new quasars appearing in an SDSS data release for the first time, as well as known quasars from previous SDSS releases. The catalog is estimated to be 99.8% complete with 0.3% to 1.3% contamination. Automated and visual inspection methods were used to identify and determine redshift information for the quasars, and additional redshifts were derived via principal component analysis and emission lines. The catalog also includes multi-wavelength data for the quasars from various sources such as Galaxy Evolution Explorer (GALEX), UK Infrared Telescope Infrared Deep Sky Survey (UKIDSS), Widefield Infrared Survey Explorer (WISE), FIRST, ROSAT/2RXS, XMM-Newton, and Gaia, enabling researchers to study quasars across different wavelengths.

Almeida et al. (2023) presented the eighteenth data release of the SDSS, which is the first release for SDSS-V. It introduces three primary scientific programs: Milky Way Mapper (MWM), Black Hole Mapper (BHM), and Local Volume Mapper (LVM). Additionally, the data release includes new SDSS spectra and supplemental information for X-ray sources identified by eROSITA. The release includes extensive targeting information for the BHM program, comprising input catalogs and selection functions. The BHM program focuses on selecting specific sets of eligible targets for observations that may have been previously observed by single or multiple catalogs and will emphasize the astrophysics of quasars. This includes studies of black hole masses, binarity, accretion, events, broad line region (BLR) dynamics, outflows, and more.

2.2 Machine Learning Methods

Within the realm of astronomical research, the application of machine learning methods has emerged as a powerful tool for automating the classification of celestial objects. This section explores the diverse array of machine-learning techniques employed by researchers to categorize stars, galaxies, and quasars based on the vast datasets generated by the SDSS.

Researchers have leveraged variations of Support Vector Machines (SVM) to achieve this goal. In the study conducted by Herle, Channegowda, and Prabhu (2020), a Linear Support Vector Machine (LSVM) along with Ensemble Bagged Trees (EBT) was used to classify quasars in SDSS DR14. They also succeeded in reducing the False Negative Rate (FNR) by 10x by applying a Learning from Mistakes methodology, however it ended up increasing the False Positive Rate (FPR) to 46.94%. Despite this, they yielded a notable improvement in sensitivity of more than 5% for identifying quasars.

In a similar vein, Z. Li et al. (2017) decided to use Kernel Support Vector Machines (K-SVM) to classify quasars based on the spectral data from SDSS. The researchers improved the parameters selection process, and an accuracy of 94.0503% was achieved for classifying objects in the SDSS dataset.

Meanwhile, Peng, Zhang, Zhao, and Wu (2012) proposed a classification system using SVM classifiers constructed using the SVM light code by Joachims (2013) to select quasar candidates from large sky survey projects such as SDSS, UKIDSS, and GALEX. The SVM classification system achieves an efficiency of 93.21% and a completeness of 97.49% when predicting quasar candidates in the test set.

Moreover, SVMs are not the only machine-learning techniques employed in the classification of astronomical objects. For instance, L. Li, Zhang, and Zhao (2008) delved into the utilization of the KNN algorithm for automated classification of multi-wavelength astronomical objects, which showed to have a running speed that is rather fast and a classification accuracy of up to 97.73% for discriminating active objects from stars and normal galaxies.

Additionally, Tu, Wei, and Ai (2015) discusses the implementation of the local mean-based k-nearest neighbor (LMKNN) method which classifies objects based on spectral data from the Large Sky Area Multi-object Fiber Spectroscopic Telescope-DR1 (LAMOST) survey. The LMKNN method selects k nearest neighbors from each class of training samples and computes the mean vectors of these neighbors to classify a sample based on the distance to the mean vector. The experimental results show that LMKNN performs better or at least as well as KNN and SVM in terms of correct classification rates, achieving rates as high as 98.97% for galaxies and 97.58% for QSOs, with an average correct classification rate of 98.33%.

Furthermore, Gao, Zhang, and Zhao (2008) compared the performance of the k-dimensional tree(kd-tree) and SVM for separating quasars from stars in SDSS and Two Micron All Sky Survey (2MASS) catalogs. The study finds that SVMs show slightly higher accuracy, but kd-tree requires less computation time, and the results suggested that using four colors (u-g, g-r, r-i, i-z) and r magnitude based on SDSS model magnitudes yields the highest accuracy for classification.

Continuing the exploration of machine learning techniques for the classification of astronomical objects, the focus is now shifted to research papers that utilize tree-based algorithms. In their research, Pichara, Protopapas, Kim, Marquette, and Tisserand (2012) harnessed the power of a boosted Random Forest (RF) classifier to identify quasars in the EROS-2 and MACHO datasets. They incorporated variability features, such as auto-regressive model parameters, and identified 1160 quasar candidates in MACHO and 2551 in EROS-2. Comparing these candidates to a list of known strong candidates, they achieved 74% matches for MACHO and 40% for EROS-2, with the difference attributed to EROS-2's shallower survey and lower signal-to-noise ratio.

Transitioning to the study by Clarke, Scaife, Greenhalgh, and Griguta (2020), they used SDSS and WISE photometry data to train an optimized random forest classifier. This resulted in a new catalog containing 50.4 million galaxies, 2.1 million quasars, and 58.8 million stars. They applied a non-linear dimension reduction technique called

Uniform Manifold Approximation and Projection (UMAP) to visualize data separation in two dimensions. The paper also addresses class imbalance and transfer learning challenges, especially concerning fainter sources, such as stars being misclassified as quasars, which can introduce uncertainties in the classification results.

In a related context, Carrasco et al. (2015) used RF to construct a catalog of quasar candidates from Red-Sequence Cluster Survey 2 (RCS-2) point sources using SDSS spectroscopically-confirmed stars and quasars and achieved precision of 89.5% and recall of 88.4% which further refined by incorporating additional information from NUV GALEX resulting in improved precision and recall of 97.0% and 97.5% for the GALEX and when WISE data are included, precision and recall increased to 99.3% and 99.1%.

Furthermore, C. Li et al. (2021) linked the Beijing-Arizona Sky Survey (BASS) DR3 with spectral databases from SDSS and LAMOST to determine the spectroscopic classes of known samples. They then cross-referenced this data with the ALLWISE database to gather optical and infrared information. Utilizing the Extreme Gradient Boosting (XGBoost) algorithm, they created classifiers for binary and multiclass classification based on the optical and infrared data, achieving an accuracy exceeding 90.0%. As a result, the catalog includes 12,375,838 stars, 18,606,073 galaxies, and 798,928 quasar candidates determined from the classification results.

In a complementary context, Zhang (2022) directed their attention to implementing a highly time-efficient machine learning algorithm. They employed the XGBoost algorithm and harnessed automatic tuning technologies, such as Bayes searching and Bayesian optimization, to enhance the efficiency of hyperparameter tuning. Their concerted efforts culminated in an impressive 99.39% overall classification accuracy.

Similarly, Golob, Sawicki, Goulding, and Coupon (2021) outlines a pipeline using Gradient Boosted Trees (GBT) for star, galaxy, and AGN classification on the CLAUDS+HSC-SSP dataset. It attains high accuracy in binary classification (star/galaxy) with an AUC of 0.9974 and impressive sample purity of 99.7% and completeness of 99.8% for galaxies. The model demonstrates generalizability to fainter objects, although its effectiveness varies for Type I and Type II Active Galactic Nucleus.

Furthermore, in a different study, Franco-Arcega, Flores-Flores, and Gabbasov (2013) employed the ParDTLT algorithm, a parallel and incremental decision tree technique, for classifying SDSS astronomical objects. It also utilized the Multilayer Perceptron classifier, a neural network model with multiple layers of weights. Experiments involved an SDSS dataset and ten-fold cross-validation, highlighting the significance of the ”u” attribute for effective object classification.

Additionally, Peng, Zhang, and Zhao (2010) compared three Linear Discriminant Analysis (LDA), KD-tree, and SVMs for classifying celestial objects based on SDSS photometric data from SDSS DR7, using six performance metrics, including positive precision, positive recall, negative precision, negative recall, accuracy, and geometric mean. SVMs perform the best, with all six metrics surpassing 99.00%, while KD-tree also performs well with all metrics over 97.00%. In contrast, LDA exhibits poorer performance, with an accuracy of positive prediction at only 85.98%.

In another notable study, Viquar, Basak, Dasgupta, Agrawal, and Saha (2018) presented the results of various automated classification methods for distinguishing stars from quasars in SDSS DR6 and DR7, providing a critical review of existing approaches and identifying the pitfalls in those approaches based on the nature of the data used for the study. The research highlights the efficacy of asymmetric AdaBoost as a machine-learning method for classifying photometric data.

2.3 Deep Learning Approaches

The advent of deep learning has revolutionized the field of astronomical object classification. In this section, the cutting-edge deep learning approaches harnessed to extract intricate features and patterns from astronomical data are investigated. These methods have propelled the ability to discern celestial objects with unprecedented accuracy and efficiency.

To start with, Brescia, Cavuoti, and Longo (2015) employed the Multi-Layer Perceptron with Quasi-Newton Algorithm (MLPQNA) method to classify objects in SDSS DR10, focusing on distinguishing galaxies, quasars, and stars. MLPQNA effectively separates quasars, achieving a 91.31% overall efficiency and 95% quasar purity. The study produces a catalog of around 3.6 million quasar/AGN candidates, with half a million robust candidates. It also discusses using optical colors for AGN identification and the challenges in differentiating AGN-hosting galaxies.

Furthermore, He et al. (2021) proposed a deep learning source detection network based on the YOLO v4 object detection framework to detect sources and a deep learning classification network called APSCnet for classifying sources. Their detection network achieves an 88.02 mAP score at IOU=0.5, while APSCnet demonstrates high precision and recall for quasars, stars, and galaxies across different magnitude ranges. Notably, for quasars, it achieves 84.1% precision at 93.2% recall (magnitudes 14-25) and 96.6% precision at 94.7% recall (magnitudes ¡20).

Moreover, Jingyi, Li, Chengjin, Jiaqi, and Zengjun (2018) used a heterogeneous kernel CNN model on the SDSS dataset based on AlexNet architecture and achieved 98% overall accuracy due to CNN's excellent feature extraction capabilities.

Similarly, Carrasco-Davis et al. (2019) proposed a recurrent CNN model which is trained using synthetic image sequences that simulate real-world conditions and achieved comparable performance to a light curve RF classifier when tested on real data from the HITS survey and the recall improved from 85% to 94% after performing fine tuning with 10 real samples per class.

Likewise, González, Muñoz, and Hernández (2018) used a combination of CNN, YOLO, and DARKNET framework, incorporating data augmentation techniques, to train models for automatic detection and classification of galaxies using data augmentation which demonstrated this method yields better results compared to methods based on manual feature engineering and SVMs when training datasets are large such as SDSS, Galaxy Zoo, Next Generation Virgo (NGVS) and Fornax (NGFS) surveys. The processing times achieved are impressively fast, 50 milliseconds for an SDSS image and less than 3 seconds for a DECam image using a high-end Nvidia GPU card.

Additionally, Pasquet-Itam and Pasquet (2018) introduced a CNN approach for classifying and detecting quasars in SDSS Stripe 82 with accuracy reaching 91.2%, outperforming other classifiers. When combined with an RF, the overall performance improved even further with a precision of 0.99 for a recall of 0.90.

2.3.0.1 Use of GANs

GANs have been regularly used by researchers for the purpose of generating fake data to resolve the class imbalance problem.

As Goodfellow et al. (2014) elucidated in their foundational paper on Generative Adversarial Networks, 'In the proposed Generative Adversarial Networks framework, simultaneously two models will be trained: a generative model G that captures the data distribution, and a discriminative model D that estimates the probability that a sample came from the training data rather than G. The training procedure for G is to maximize the probability of D making a mistake. This framework corresponds to a minimax two-player game. In the space of arbitrary functions G and D, a unique solution exists, with G recovering the training data distribution and D equal to 1/2 everywhere.'

To put this in layman's terms, the Vanilla GAN, which is the most basic form of GAN, consists of two primary components: a generator and a discriminator. The generator's role is to create synthetic data samples, and the discriminator's role is to evaluate these samples against real data.

However, in a traditional GAN, there is no way to include any conditional arguments on the basis of which the augmented data is generated. Hence, for Vanilla GAN, extra data, a pre-processing step had to be done where a subset of the dataset was created with only QSO class instances to train the model. Similar issues have been identified by researchers in the past for generating data for a classification problem (Sagong, Shin, Yeo, Park, & Ko, 2020).

To overcome this issue, Mirza and Osindero (2014) described another architecture for GAN where a condition of class labels was introduced. Vega-Márquez, Rubio-Escudero, Riquelme, and Nepomuceno-Chamorro (2019) experimented with a Conditional GAN (cGAN) to generate new fake data for a numerical dataset that was very close to real data. They used a cGAN with a similar architecture to that of a vanilla GAN, however only a condition is added as a part of the input.

Additionally, Ger, Jambunath, and Klabjan (2019) introduced an interesting idea of the use of a GAN with an autoencoder component (GAN-AE) for generating synthetic data, particularly for sequential data and labels. The study highlights how augmenting datasets with GAN-AE synthetic data improves model performance, especially in scenarios with imbalanced data.

2.4 Amalgamation of Multiple Techniques

Recognizing the multifaceted nature of astronomical object classification, researchers have increasingly turned to hybrid approaches. This section highlights the innovative methods that amalgamate various classification techniques, often combining machine learning, deep learning, and traditional astrophysical insights to achieve more robust and nuanced results.

The first such technique reviewed is from Makhija, Saha, Basak, and Das (2019), they focused on the problem of classifying matched sources in the GALEX and SDSS catalogs into stars and quasars based on color-color plots where there is no clear linear/non-linear boundary separating the two entities. The authors explore the efficacy of neural network-based classification techniques along with the ensemble classifiers. They use objects with associated spectroscopic information as the training set and build GAN and RF ensemble classifiers to classify photometric samples without spectroscopic labels. The correctness of the classifiers is evaluated by reporting the accuracy and other performance metrics, which show a reasonably satisfactory range of `91% to 100%`.

Furthermore, the research done by Agarwal (2023) builds on the Fermi 4LAC-DR3 catalog and addresses the challenge of categorizing unassociated blazars lacking optical spectral information. It employs high-precision machine-learning algorithms, including random forest, logistic regression, XGBoost, CatBoost, and neural networks, to classify blazars of unknown type into BL Lac objects and FSRQs. By combining results from all models, the study achieves highly accurate and robust predictions, with a classification metric area under the curve ¿0.96.

In a related context, Chuntama, Techa-Angkoon, Suwannajak, Panyangam, and Tanakul (2020) applied multiclass classification techniques to categorize astronomical objects in galaxy M81 using machine learning. Researchers used data from the Canada-France-Hawaii Telescope (CFHT) archive and classified objects into five categories: star, globular cluster, rounded galaxy, elongated galaxy, and fuzzy object. They investigated seven classification techniques, including RF, MLP, Weightless neural network (WiSARD), Weka deep learning, Logistic Regression, SVM, and Multiclass Classifier, with RF and MLP delivering the highest performance, making them the optimal models for object classification in the CFHT data of galaxy M81.

Similarly, Omat, Otey, and Al-Mousa (2022) focused on using multiple models: D-Trees, KNN, Multinomial Logistic Classification, MLP, Naïve Bayes Classifier, SVM, RF, and Soft Voting Classifier, to classify instances in SDSS DR17 as galaxies, quasars, or stars. The Random Forest model performed the best with 98% accuracy, correctly classifying all instances labeled as stars, while the Naïve Bayes Classifier had the lowest accuracy at 91%.

2.5 Alternative Approaches

In the pursuit of enhancing the comprehension of celestial objects, this section explores alternative approaches that diverge from conventional machine learning and deep learning methodologies. These approaches encompass unique strategies, such as deblending methods, the use of hyperplanes, and some task-specific algorithms.

To commence the discussion, Flesch (2021) focused on the identification, confusion, and blending concealment in the SDSS-DR16 Quasar catalogs. The research presented the discovery of 40 previously undeclared quasars that were concealed or confused with other objects due to incomplete de-blending. It also identifies 82 entries in the SDSS-DR16Q main quasar catalog that are shown to be non-quasars, some of which are also due to incomplete de-blending.

In another research, Khramtsov and Akhmetov (2018) presented a new fully-automatic classification model for selecting extragalactic objects within astronomical photometric catalogs, which follows three main procedures, i.e., data representation, creation of feature space, building a hypersurface for outlier detection, and constructing a hyperplane to separate extragalactic from galactic objects. They demonstrated the application of the model through the creation of a photometric catalog of 38 million extragalactic objects, identified in the WISE and Pan-STARRS catalogs cross-matched with each other.

Moreover, Peters et al. (2015) explored combining color and variability data to detect quasars in optical surveys. They employed a Bayesian quasar selection algorithm on SDSS Stripe 82, achieving high completeness and efficiency when using variability, colors, or both. The quasar sample aligns with spectroscopic findings, enabling the creation of a luminosity function.

2.6 Summary

In this section, the wealth of knowledge derived from the reviewed literature has been summarized comprehensively. To begin with, the literature survey discussed the contribution of catalog papers which are traditionally published with every release of the SDSS data.

Later, the section discussed various Machine Learning methods used by researchers, such as variations of SVMs, tree-based methods such as Dtree and RF, KNNs, XGBoost, and AdaBoost. The researchers also used multiple datasets in some instances, using data from various sky surveys, including BOSS, GALEX, WISE, and many more. Some of the researchers explicitly highlighted the challenges presented by class imbalance causing a high misclassification rate.

Further, other researchers' approaches to using Deep Learning methods were discussed. These approaches leverage robust algorithms like MLP and CNN which were fine-tuned using architectures such as AlexNET and frameworks like YOLO and DARKNET. However, the research done on the utilization of neural networks for this problem has been very limited.

In the subsequent section, researchers' approach of amalgamation of various techniques to utilize different types of classifiers to yield the best results by combining their abilities, such as an ensemble of various classifiers and combining results of various classifiers, were discussed.

To conclude, alternate unorthodox approaches followed by researchers, such as de-blending and a combination of various features such as color and variability, were discussed.

In summary, the literature survey has laid a strong foundation by highlighting the key findings, methods, and approaches from previous research. It has provided a comprehensive overview of the landscape and the gaps in the existing knowledge. The following sections will address these gaps and present the research framework for exploring new horizons in classifying celestial objects within the SDSS dataset.

2.7 Gaps in Literature

Within the research literature, several crucial aspects concerning the dataset have been brought to light. Notably, an evident class imbalance issue is identified in the dataset, as reported by (Herle et al., 2020; Pichara et al., 2012; Omat et al., 2022). This imbalance primarily stems from a substantial disparity in the number of instances among the different classes, with the quasar class being significantly underrepresented. Consequently, this imbalance poses formidable challenges in achieving accurate classification and introduces the potential for biased model outcomes. One of the notable consequences of class imbalance is the misclassification of the minority class instances. It's important to recognize that this issue often remains concealed since the overall classifier performance metrics can appear satisfactory, mainly due to the metrics being skewed by the outcomes of the majority class. Therefore, the research emphasizes the importance of addressing this class imbalance problem to ensure more equitable and precise classification across all classes in the dataset.

Past research endeavors have predominantly relied on earlier releases of the dataset, as evidenced by (Brescia et al., 2015; Flesch, 2021; Franco-Arcega et al., 2013; He et al., 2021; Herle et al., 2020; Jingyi et al., 2018; Khramtsov & Akhmetov, 2018; Zhang, 2022; Peters et al., 2015; Peng et al., 2012, 2010). This reliance on outdated data versions, however, poses limitations to the generalizability of the findings. Notably, these earlier dataset releases exhibited an inherent scarcity of instances within the underrepresented classes, thereby hindering classifiers from effectively discerning the underlying patterns associated with these specific instances.

In contrast, the latest dataset release offers a promising outlook, characterized by a more substantial representation of the previously underrepresented classes. This updated dataset empowers classifiers with a better opportunity to gain a comprehensive understanding of the intricate patterns associated with these underrepresented instances. As a result, it significantly enhances the classifiers' potential for robust and reliable learning, consequently contributing to more accurate and generalizable outcomes in classification tasks. This evolution highlights the critical importance of leveraging the most current dataset releases to enhance the overall quality and applicability of research findings.

Furthermore, it's worth noting that the existing studies reviewed do not employ data augmentation techniques to specifically tackle the issue of class imbalance and enhance the classification of minority classes, as highlighted in the works by (Herle et al., 2020; Jingyi et al., 2018; Omat et al., 2022). In these research efforts, the emphasis has primarily been on other aspects of deep learning and data analysis, rather than addressing the class imbalance problem.

The dataset's substantial size, boasting a vast collection of over 100,000 instances, inherently signifies the potential suitability of neural networks for effective classification tasks. Existing research, as evidenced by (Franco-Arcega et al., 2013; Herle et al., 2020; C. Li et al., 2021; Z. Li et al., 2017; Makhija et al., 2019; Omat et al., 2022; Peng et al., 2012; Tu et al., 2015; Khramtsov & Akhmetov, 2018; Zhang, 2022; Bai, Liu, Wang, & Yang, 2018; Golob et al., 2021; Peng et al., 2010; Clarke et al., 2020; Carrasco et al., 2015), and others, has indeed acknowledged the dataset's vastness and the associated potential for leveraging neural networks in the classification process. However, despite this recognition, it is notable that the comprehensive exploration and exploitation of neural networks within this dataset's context were relatively limited.

This under-utilization of neural networks is particularly noteworthy given the dataset's immense scale and complexity. The vast volume of instances within the dataset provides ample opportunities for neural networks to excel in uncovering intricate and latent patterns that may remain elusive to more conventional approaches. The limited exploration of neural networks in previous research underscores the untapped potential and promising avenues that future studies can explore to harness the full benefits of these powerful algorithms for the classification of celestial objects in the dataset.

Chapter 3

Experiment design and methodology

This section provides a detailed overview of the design of the experiment, which will be conducted to test the research hypothesis. It involves the theoretical design of the experiment that will be conducted along with the methodology followed.

The chapter is divided into several parts, beginning with the Data Collection and Data Preprocessing sub-sections, which discuss the collection and preprocessing of data. The focus of this section is to point out the process of data extraction from the source, the rationale behind choosing particular features, and the various types of preprocessing steps performed on the data.

To successfully conduct an experiment, an effective experiment setup is necessary, which is discussed in the Experiment Setup section. The software and hardware setup for the experiment, along with configuration details, is explored.

As the main aim of the research is to evaluate the efficacy of various GAN-based Data Augmentation Techniques, this section will center on the comparative analysis of these GAN models, not only against each other but also against the SMOTE technique as the baseline. Further on, the architecture and training of base and augmented CNN models are also discussed in this section

To set a benchmark for comparison, the SMOTE technique is employed as a baseline model. This approach will be used to assess the efficacy of the GANs in generating useful synthetic data for training purposes.

Following the establishment of the base model, various GAN architectures are trained. Each GAN, once trained, is utilized to generate synthetic data, resulting in augmented datasets. These augmented datasets are then used to train the 'Augmented CNN' - a model with an architecture akin to the base CNN but trained on the data enriched by the GAN-generated samples.

The experiment's success depends on the meticulous choice of the Evaluation Metrics. In this section, various evaluation metrics have been explored, and their alignment with the research objectives has been established.

The experiment's goal is to validate the hypothesis's veracity, which is performed by Hypothesis Testing. This section entails the execution of a diverse array of statistical tests to corroborate the alternate hypothesis.

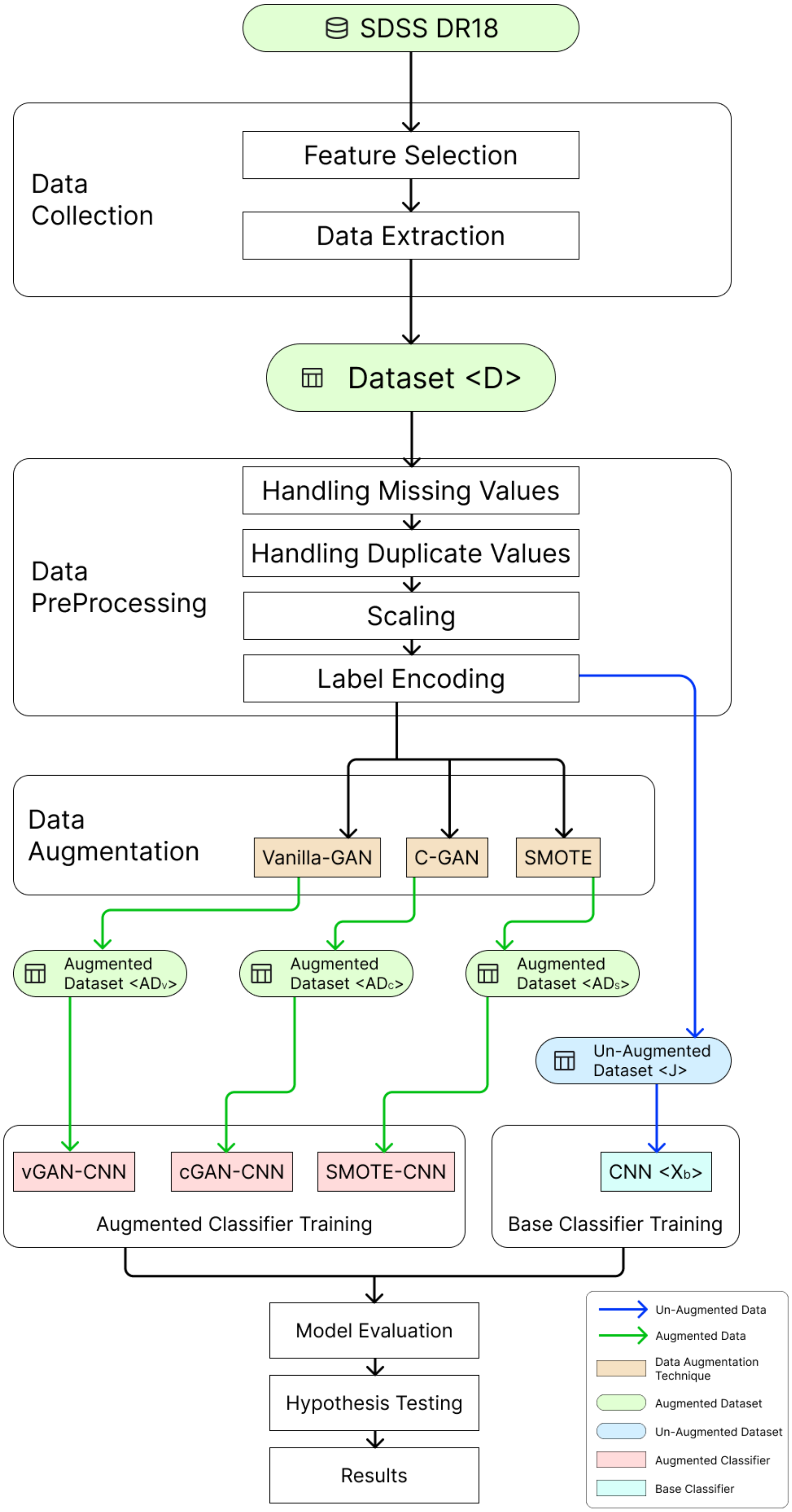


Figure 3.1: Overview of Research Design

3.1 Data Collection

This section discusses the methods and processes involved in collecting the data that form the basis of the research. Data collection is a fundamental step in the research journey, and it influences the quality and reliability of the findings. This section explores how the data was sourced and the tools and techniques used.

As mentioned earlier, the data is sourced from SDSS DR 18, which is publicly available on the SDSS SciServer. The CAS job functionality was used to extract the data from the database using below SQL query:

```
SELECT TOP 100000 p.objid, p.ra, p.dec, p.u, p.g, p.r, p.i, p.z,
p.petroRad_u, p.petroRad_g, p.petroRad_i, p.petroRad_r, p.petroRad_z,
p.petroFlux_u, p.petroFlux_g, p.petroFlux_i, p.petroFlux_r, p.petroFlux_z,
p.petroR50_u, p.petroR50_g, p.petroR50_i, p.petroR50_r, p.petroR50_z,
p.psfMag_u, p.psfMag_r, p.psfMag_g, p.psfMag_i, p.psfMag_z,
p.expAB_u, p.expAB_g, p.expAB_r, p.expAB_i, p.expAB_z,
s.specobjid, s.class, s.z as redshift
into mydb.MyTable from PhotoObj AS p
JOIN SpecObj AS s ON s.bestobjid = p.objid
WHERE
p.u BETWEEN 0 AND 19.6
AND g BETWEEN 0 AND 20
```

The choice of features to be selected was based on reviewed literature. This query extracts the top 100,000 results from all the results returned and selects the following features:

1. **objid and specobjid** - These object identifier columns are just a categorical column that identifies the object by tagging it with a unique ID.

2. **ra** - This column has the right ascension details of the object. Right ascension is

similar to longitude on Earth and is used to specify a celestial object's position in space.

3. **dec** - This column has the declination details of the object. Declination is similar to the concept of latitude on Earth and is used to specify a celestial object's position in space.

4. **u, g, r, i, and z** - These bands represent the five different filters or passbands used for imaging celestial objects in SDSS. The alphabets 'u', 'g', 'r', 'i', and 'z' in SDSS denote different photometric bands capturing ultraviolet, green, red, infrared, and near-infrared wavelengths, respectively. They are used to capture images and measure the brightness of objects at different wavelengths within the optical spectrum. The numerical value represents the measured magnitudes of celestial objects in those specific filters. By observing celestial objects in multiple filters, astronomers can gain insights into the objects' spectra, temperatures, and distances.

5. `petroFlux_u`, `petroFlux_g`, `petroFlux_r`, `petroFlux_i`, and `petroFlux_z` - This is Petrosian Flux for the five photometric bands u, g, r, i, and z respectively. These characteristics encompass the aggregate radiance emitted by celestial objects, encompassing their luminosity, coloration, and spectral energy distribution.

6. `petroRad_u`, `petroRad_g`, `petroRad_r`, `petroRad_i`, and `petroRad_z` - This is Petrosian Radii for the five photometric bands u, g, r, i and z respectively. The Petrosian radius is the distance from a galaxy's center where the ratio of local surface brightness to average surface brightness reaches a predetermined value. Local surface brightness refers to a specific region's brightness, while average surface brightness is the mean brightness over the entire object's surface. These parameters are vital for characterizing celestial objects, aiding in the study of their morphologies, sizes, and evolution.

7. `petroR50_u`, `petroR50_g`, `petroR50_r`, `petroR50_i`, and `petroR50_z` - This is

Petrosian half-light radii for the five photometric bands u, g, r, i, and z respectively. PetroR50 represents the radius where half of a celestial object's emitted light is contained within the Petrosian aperture, defined by the Petrosian radius. This aperture efficiently captures a significant portion of an object's total light while reducing interference from neighboring sources or background noise. These parameters are essential for sizing celestial objects in SDSS data and understanding their size variations across different wavelengths.

8. `psfMag_u`, `psfMag_g`, `psfMag_r`, `psfMag_i`, and `psfMag_z` - This is magnitudes of objects measured using the Point Spread Function (PSF) in the five photometric bands u, g, r, i, and z respectively. The PSF is a model for how an imaging system responds to point sources of light. While most celestial objects are extended, the PSF approximates their brightness as if they were point sources. psfMag values in the SDSS data represent the magnitudes of objects as measured using the PSF. Magnitude is a logarithmic scale for quantifying brightness, with lower values indicating brighter objects. This helps characterize the brightness and colors of stars, galaxies, and other astronomical objects.

9. `expAB_u`, `expAB_g`, `expAB_r`, `expAB_i`, and `expAB_z` - This is axis ratio of exponential fits to the light profile of celestial objects observed in the five photometric bands u, g, r, i, and z respectively. This characterizes the shapes of celestial objects, aiding in the understanding of morphological features, particularly in galaxies, across various SDSS bands.

10. **redshift** - This is the final redshift of the celestial object, which is a measure of how much the light from a celestial object has been shifted toward longer wavelengths due to the Doppler effect.

11. **class** - This column is used to categorize the observed objects into one of three classes: Star, Galaxy, or Quasar. This is the target variable.

3.2 Data Preprocessing

Data preprocessing plays a foundational role in ensuring the accuracy and reliability of the results. Here, the steps and techniques employed to clean, transform, and prepare the data for subsequent analysis will be detailed.

To commence with, data will be checked for any missing values. If any missing values are found, then depending on the ratio of the missing values to total data is less than or more than 5%, they will be dealt with. Since the data pool is already large enough, the removal of such a minor portion (up to 5%) of the dataset will not have any severe impact on the experiment outcome. If the missing values are more than 5%, then they'll be imputed with the mean value of the column.

Next, a check for duplicate values will be conducted to identify and eliminate any redundancies. Duplicate data points can impact the experiment's results, potentially biasing model training, especially when duplicates predominantly belong to a single class.

Furthermore, histograms and box plots will be used to visualize the data to get a better understanding of data distribution. This will be followed by data divided for model training into input features 'X' and target feature 'y' after which the data will be shuffled to remove any inherent patterns in the dataset.

Once the data is shuffled, the data will be split the data into train, validate, and test sets with a ratio of 70:20:10, respectively. Then, the input features will be scaled so the features with huge differences in scales don't impact the model training.

Later, the target feature will be encoded to transform it into a numerical feature, as most algorithms can work with numerical features only.

3.3 Experiment Setup

In this section, the essential software and hardware components that constitute the foundation of the experiment will be discussed. The experiment is conducted on a MacBook Pro equipped with an Apple M1 Pro 10-core processor, a 16-core GPU, and

32 GB of RAM. Python will be the programming language utilized for this research as it offers the necessary tools through modules such as Tensorflow and Keras. These modules provide a comprehensive range of functionalities and libraries required for implementing the GAN and CNN models and conducting the necessary data analysis. By utilizing Python, its extensive ecosystem can be taken advantage of, including various data manipulation, preprocessing, visualizing, and modeling libraries such as Pandas, Numpy, Seaborn, Matplotlib, and many more, enabling efficient implementation and experimentation of research objectives.

3.4 Data Augmentation Technique

In this section, the heart of the research is delved into by exploring various Generative Adversarial Networks (GAN) architectures which are instrumental in generating synthetic data for tabular datasets. The training process, architectural choices, and the capabilities of GANs in enhancing the dataset will be discussed in this section.

The various GAN architectures employed in this experiment are a Vanilla GAN and a Conditional GAN (cGAN).

Additionally, to establish a benchmark for the experimental evaluation, the Synthetic Minority Over-sampling Technique (SMOTE) is utilized as the baseline augmented data generation model. By comparing the performance of GAN-generated datasets against those augmented with SMOTE, the aim is to assess the efficacy and potential advantages of using advanced generative models like GANs in various data augmentation scenarios.

3.4.1 SMOTE

Synthetic Minority Over-sampling Technique (SMOTE), a novel method developed by Chawla, Bowyer, Hall, and Kegelmeyer (2002) to solve class imbalance problem by generating synthetic samples from the minority class, thereby enriching the dataset without merely replicating existing samples.

3.4.1.1 SMOTE Application

The SMOTE technique is applied in the following manner:

- Configuration: The number of k-neighbors is set to 3, and the random seed is set to 42.
- Resampling: The method `fit_resample` is utilized to oversample the minority class in the training dataset.
- Data Concatenation and Shuffling: Synthetic samples from the minority class are concatenated with the original training data. The combined dataset is then shuffled to ensure randomness.

3.4.1.2 Smote Augmented CNN Training

Furthermore, once the generation of the augmented dataset is completed, the CNN training phase starts, which is referred to as SMOTE-CNN further on.

3.4.1.2.1 Architecture of SMOTE-CNN

The CNN architecture is as follows:

- Function Description: The CNN model is defined using a function `cnn_model` which constructs a Convolutional Neural Network with various layers stacked sequentially.
- Layers and Activation Function: The model is constructed using the Sequential model from Keras, where layers are arranged in a linear fashion. The architecture consists of alternating Conv1D and MaxPooling1D layers for feature extraction, followed by a GlobalAveragePooling1D layer to reduce dimensionality. It concludes with a Dense layer for complex feature learning and a Dropout layer to prevent overfitting. All the layers utilize the ReLU (Rectified Linear Unit) activation function.

- Output Layer: The final Dense layer serves as the output layer of the network. The softmax activation function is used to output a probability distribution over the classes.

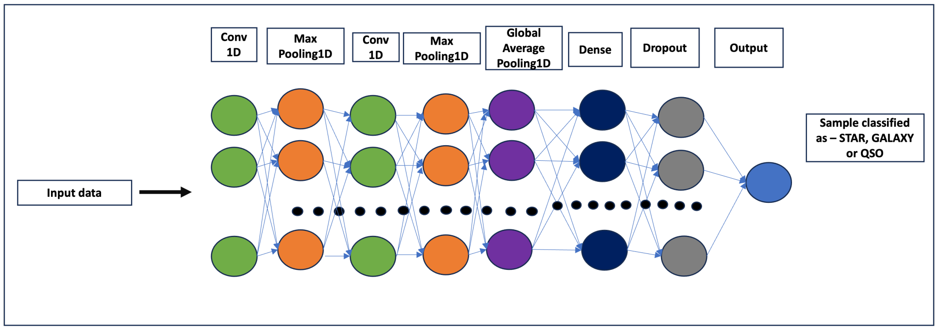


Figure 3.2: Architecture of SMOTE-CNN

3.4.1.2.2 Hyperparameters of SMOTE-CNN

Variable	**Value**
Neural Network Architecture	Convolutional Neural Network (CNN)
Layer Configuration	
Convolutional Layers	2 layers with filter sizes: 64 and 128
Kernel Size	3 for each Conv1D layer
Pooling Layers	2 MaxPooling1D layers with pool size: 2
Global Average Pooling	1 GlobalAveragePooling1D layer
Dense Layers	1 Dense layer with 128 units
Dropout	0.5 for Dropout layer
Activation Function - Conv1D	ReLU for Conv1D layers
Activation Function - Dense	ReLU for first Dense layer, Softmax for output layer
Compilation and Training	
Loss Function	Categorical Crossentropy
Optimizer	Adam
Learning Rate	0.001
Metrics	Accuracy, Precision, Recall, Custom F1 Score, FNR
Batch Size	32
Epochs	10
Number of Classes	3

Table 3.1: CNN's Hyperparameters

3.4.1.2.3 Training Process

This section details the mechanisms and steps involved in training the SMOTE-CNN.

Compilation: The model is compiled with the Adam optimizer with a learning rate of 0.001 and using `categorical_crossentropy` as loss function of, which is suitable for multi-class classification tasks. The metrics for evaluation include accuracy, precision, recall, F1 score, and the False Negative Rate (FNR).

Training: The model will be trained for 10 epochs with a batch size of 32, using both the training data and validation data. The history of the training process is recorded, allowing analysis of performance metrics over epochs.

Evaluation: To rigorously assess the performance of the trained CNN, a Monte Carlo simulation-based evaluation will be performed. This approach provides a robust statistical analysis of the model's predictive capabilities across multiple metrics.

3.4.2 Vanilla GAN (vGAN)

As described earlier, a GAN's architecture is divided between its two components, the Generator and the Discriminator.

3.4.2.1 Architecture of the Vanilla GAN

The Generator architecture is as follows:

- Function Description: The generator model is defined using a function `build_generator`, which dynamically constructs a neural network based on a given layer configuration.

- Layers and Activation Functions: The model is constructed using the Sequential model from Keras, where layers are arranged in a linear fashion. In the generator, each Dense (fully connected) layer is followed by a Dropout layer. The Dense layers utilize ReLU activation functions. The Dropout layers are introduced to mitigate overfitting by randomly setting a fraction of the input units to zero during training.

- Output Layer: The final layer of the generator is a Dense layer with a linear activation function, designed to output data with 33 features, corresponding to the dimensionality of the input dataset.

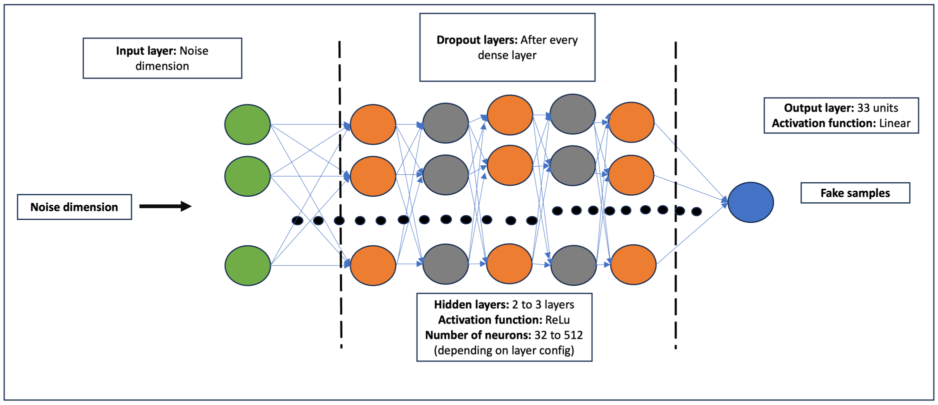

Input layer: Noise dimension
Dropout layers: After every dense layer
Output layer: 33 units
Activation function: Linear
Noise dimension
Fake samples
Hidden layers: 2 to 3 layers
Activation function: ReLu
Number of neurons: 32 to 512
(depending on layer config)

Figure 3.3: Architecture of Vanilla GAN Generator

The Discriminator architecture is as follows:

- Function Description: Similarly, the discriminator model is constructed using a `build_discriminator` function.
- Layer Composition: It also adopts a Sequential model with multiple Dense layers followed by a Dropout layer. Each of these layers employs a ReLU activation function.
- Output layer: The last layer of the discriminator is a single-unit Dense layer with a sigmoid activation function, signifying its role in binary classification to distinguish between real and fake data.

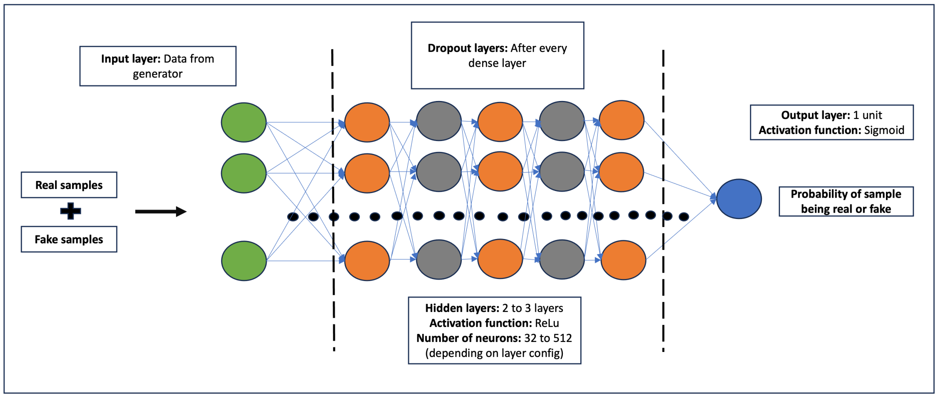

Input layer: Data from generator
Dropout layers: After every dense layer
Output layer: 1 unit
Activation function: Sigmoid
Real samples
Fake samples
Probability of sample being real or fake
Hidden layers: 2 to 3 layers
Activation function: ReLu
Number of neurons: 32 to 512
(depending on layer config)

Figure 3.4: Architecture of Vanilla GAN Discriminator

3.4.2.1.1 Hyperparameters of Vanilla GAN

Variable	**Value**
Neural Network Architecture	Vanilla Generative Adversarial Network (GAN)
Generator	
Hidden Layers	Configurations: (32, 64), (64, 128), (64, 128, 256), (128, 256, 512)
Dropout	0.5 for each layer
Activation Function - Hidden	ReLU
Activation Function - Output	Linear
Discriminator	
Hidden Layers	Configurations: (32, 64), (64, 128), (64, 128, 256), (128, 256, 512)
Dropout	0.5 for each layer
Activation Function - Hidden	ReLU
Activation Function - Output	Sigmoid
Loss Function	Binary Crossentropy
Optimizer	Adam
Learning Rate	0.0002
Optimizer's Hyperparameters	$\beta_1 = 0.5$
Training Parameters	
Noise Dimension	100
Batch Size	32
Epochs	10
Number of Classes	3

Table 3.2: Vanilla GAN's hyperparameters

3.4.2.2 Training Process

This section details the mechanisms and steps involved in training both the generator and discriminator within the GAN architecture. Initially, separate models for the generator and discriminator are constructed.

Building the Models: Initially, separate models for the generator and discriminator are constructed. The generator is responsible for creating data that mimics the real data distribution, while the discriminator's role is to distinguish between the generated data and actual data.

- **Generator**: The `build_generator` function is utilized to create the generator model. This model takes a noise vector as input and outputs data that resembles the real dataset.
- **Discriminator**: Conversely, the `build_discriminator` function constructs the discriminator model, which classifies the input data as real or fake.

These models are then combined to form the GAN, where the generator's output is directly fed into the discriminator. During the combined GAN training, the discriminator's weights are frozen when training the generator. This ensures that only the generator learns from the adversarial process at this stage.

The training occurs in two steps – first, the discriminator is trained on real and generated data, then the generator is trained based on the discriminator's feedback. This process is repeated iteratively. The ultimate goal is to reach a point where the generator produces data so convincing that the discriminator is unable to differentiate it from real data, essentially classifying generated data as real as often as it does with actual data.

Layer Configuration: The performance of a GAN is highly influenced by the number and arrangement of layers in both the generator and discriminator. To identify the most optimal layer configuration, a systematic and iterative approach to test various layer configurations is employed.

Within the loop iterating over these configurations, each model is trained, and its performance is evaluated. If a particular configuration yields a better MSE than what we've observed so far, it's marked as the new global best model. Upon completing the training for all configurations, the generator is restored to the state of the best-performing model. This approach ensures that the generator used in subsequent tasks is the one that demonstrated the highest level of performance across all the configurations that were tested.

Compilation and Optimization: Both models are compiled using the Adam optimizer, with a learning rate of 0.0002 and a beta value of 0.5. This configuration is chosen due to its effectiveness in balancing fast convergence and stability in training GANs. The loss function employed is binary cross-entropy, a standard choice for binary classification problems like ours, where the discriminator classifies data as real or fake.

Training Loop: The training loop involves alternating between training the discriminator and the generator:

- **Discriminator Training**: For each batch, the discriminator is trained on a mixture of real data from the training set and fake data generated by the generator. Through this step, the discriminator learns how to differentiate between real and generated data.
- **Generator Training**: In this phase, the generator is trained to produce data that the discriminator classifies as real. It's a key step in improving the generator's ability to create convincing data.

During training to prevent Discriminator from becoming overpowering, randomness is introduced in the labels. A matrix of random values between 0 and 0.05 is generated which is then added to the labels. This results in the real data labels becoming slightly less than 1 and the generated data labels becoming slightly more than 0, hence preventing Discriminator from becoming overconfident. Ultimately, it helps prevent overfitting and stabilizes the overall GAN training.

Validation and Early Stopping: After each training epoch, the model's performance is evaluated using a Monte Carlo approach. This involves generating multiple sets of data, calculating the MSE for each against the validation set, and then computing the mean and standard deviation of these MSE values.

Then the best mean MSE and best weights are updated whenever the mean MSE of the current epoch is lower than the current best mean MSE. This ensures that the generator weights that yield the best average performance are saved.

The early stopping mechanism is implemented to check if the current epoch number is more than patience epochs away from the best epoch. The best epoch is updated whenever a new best mean MSE is found. This ensures that the training is halted once the improvement in the generator's performance plateaus, effectively preventing overfitting.

At the end of training (either after completing all epochs or due to early stopping), the generator's weights are set to the best weights, ensuring that the model reflects the best performance observed during training.

3.4.2.3 Generation and Augmentation of Fake Data

The trained generator from GAN training is used to generate fake data for the QSO class, which is then augmented to the original dataset.

- **Setup**: The process is initiated by defining two key parameters i.e., the total number of augmented datasets to be generated and the number of synthetic instances added in each iteration. It is set up to generate 5 distinct datasets with an increment of 5000 instances of QSO class in each iteration.

- **Synthetic Data Generation**: Utilizing the trained GAN, specifically its generator component, synthetic samples resembling real data were produced. For each iteration, a noise vector following a standard normal distribution, with dimensions based on the required number of fake instances and the GAN's noise dimension was generated.

- **Transformation into DataFrames:** The GAN's output was then structured into a DataFrame. Each column in this DataFrame represented a feature of the dataset, mirroring the structure of the real data. Importantly, all generated instances were labeled as belonging to the 'QSO' class.

- **Augmentation Process**: In each iteration, the newly created DataFrame of synthetic data is concatenated with the original dataset. This step blends the real and artificial data seamlessly. The augmentation process is iterative. In each subsequent dataset, an additional increment number of synthetic instances is added. This approach resulted in a series of datasets, each with a progressively larger proportion of synthetic data.

- **Enhanced Datasets**: At the conclusion of the process, five distinct datasets were obtained, each augmented with a varying number of GAN-generated samples. This method effectively increased the volume and diversity of the data available for training the machine learning models.

3.4.2.4 Kolmogorov-Smirnov Test

A Kolmogorov-Smirnov test is performed to statistically compare the distributions of features in the original dataset with those in the augmented datasets. The KS test statistic measures the maximum difference between the cumulative distribution functions of two samples. A higher value indicates a greater divergence between the distributions.

To implement the test, `'ks_2samp'` function from `'scipy.stats'` module has been utilized. The test was conducted independently for each feature in the dataset and was applied iteratively across each augmented dataset.

3.4.2.5 Vanilla GAN Augmented CNN

Furthermore, once the augmented datasets are ready, then the CNN will be trained on the augmented datasets iteratively. This CNN will be hereon referred to as vGAN-CNN.

3.4.2.5.1 Architecture of Vanilla GAN Augmented CNN

- Function Description: The `cnn_model` function constructs the CNN with various layers.

- Layers and activation function: Similar to the previous SMOTE-CNN implementation, the model is constructed using the Sequential model from Keras with the architecture consisting of alternating Conv1D and MaxPooling1D, followed by a GlobalAveragePooling1D. It concludes with a Dense layer and a Dropout layer with all layers utilizing the ReLU activation function.

- Output Layer: The final dense layer, which serves as an output layer of the network, uses a softmax activation function.

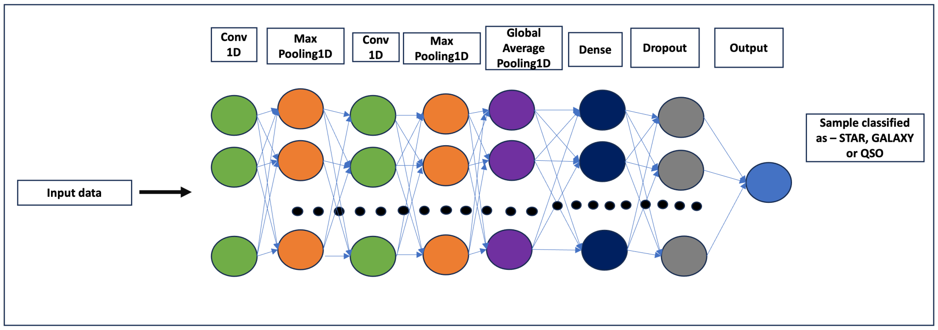


Figure 3.5: Architecture of vGAN-CNN

3.4.2.5.2 Hyperparameters of vGAN-CNN

Variable	**Value**
Neural Network Architecture	Convolutional Neural Network (CNN)
Layer Configuration	
Layer 1 (Conv1D)	64 filters, kernel size 3, ReLU activation
Layer 2 (MaxPooling1D)	Pool size 2
Layer 3 (Conv1D)	128 filters, kernel size 3, ReLU activation
Layer 4 (MaxPooling1D)	Pool size 2
Layer 5 (GlobalAveragePooling1D)	-
Layer 6 (Dense)	128 units, ReLU activation
Layer 7 (Dropout)	Rate 0.5
Layer 8 (Dense)	Number of classes, Softmax activation
Compilation Parameters	
Optimizer	Adam
Learning Rate	0.001
Loss Function	Categorical Crossentropy
Training Parameters	
Batch Size	32
Epochs	10
Input Shape	(Number of features, 1)
Number of Classes	3 (or as per dataset)

Table 3.3: vGAN-CNN Hyperparameters

3.4.2.5.3 Training process

In this section, the mechanism and steps for training the vGAN-CNN will be discussed.

Data preprocessing: The augmented datasets were again preprocessed by performing similar preprocessing steps as the original dataset. The columns ’class’, ’objid’, and ’specobjid’ are dropped since they are not useful for training. Then the standard scaler is applied to the data to standardize the feature set. Further, the data is split into training, validation, and test sets using `'train_test_split'`. After that, the class labels are handled by using 'LabelEncoder' to convert them into numerical format and later transforming these encoded labels into one-hot encoded format using `'to_categorical'`.

Compilation: The model is compiled with the Adam optimizer with the same learning rate of 0.001 and `categorical_crossentropy` loss function. This CNN model is rigorously evaluated using metrics such as accuracy, precision, recall, a custom F1 score, and the false negative rate.

Training: The model is trained using a batch size of 32 and for 10 epochs, with both training and validation data provided. The CNN model trains on each dataset iteratively. This approach allows us to store the trained model and its history for each dataset with an increasing number of instances, facilitating a thorough analysis of the model's performance across diverse data conditions.

Evaluation: The evaluation of the model emphasizes its ability to accurately identify QSO instances, utilizing metrics such as accuracy, precision, recall, F1-score, and false negative rate. By analyzing its performance on test data and incorporating a confusion matrix, a detailed understanding of the model's proficiency in distinguishing between QSO and other classes is gained.

3.4.3 Conditional GAN (CGAN)

The cGAN architecture is similar to GAN is divided between two components, the Generator and the Discriminator. Except for the notable difference of being a conditional component in the input of cGAN.

3.4.3.1 Architecture of the cGAN

The Generator architecture is as follows:

- Function Description: The `build_conditional_generator` function constructs the generator model. It dynamically creates a neural network based on a specified layer configuration, incorporating both noise and class label inputs.

- Layer Configuration: Utilizing Keras' Sequential model, the generator features a series of Dense layers arranged linearly. Each Dense layer, employing ReLU activation functions, is paired with a Dropout layer to prevent overfitting.
- Combining Inputs: The architecture begins with separate noise and class label inputs, which are then concatenated. This combined input feeds into the sequential Dense and Dropout layers.
- Output Layer: The final layer is a Dense layer with linear activation, designed to output data with dimensions corresponding to the input dataset features.

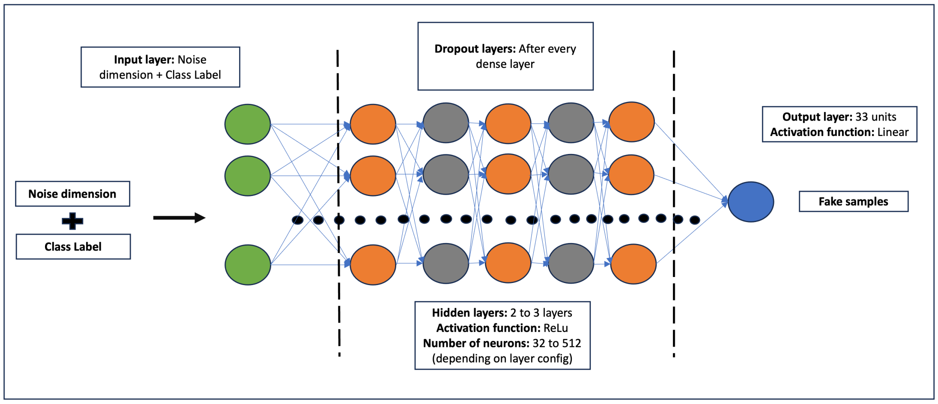

Input layer: Noise dimension + Class Label
Dropout layers: After every dense layer
Output layer: 33 units
Activation function: Linear
Noise dimension
Class Label
Fake samples
Hidden layers: 2 to 3 layers
Activation function: ReLu
Number of neurons: 32 to 512 (depending on layer config)

Figure 3.6: Architecture of cGAN Generator

The Discriminator architecture is as follows:

- Function Description: The `build_conditional_discriminator` function creates the discriminator model. It dynamically assembles a neural network based on the given layer configuration, designed to handle combined data and class label inputs.

- Layer Configuration: Employing the Sequential model from Keras, this architecture consists of a series of Dense layers arranged linearly succeeded by a Dropout layer to prevent overfitting. Each Dense layer features ReLU activation functions, critical for processing the concatenated inputs of data features and class labels.

- Input Combination: The model starts with two separate inputs – data features and class labels, which are concatenated. This combined input is then processed by the Dense layers.

- Output Layer: The final output is a single neuron with a sigmoid activation function, which classifies the input data as real or synthetic.

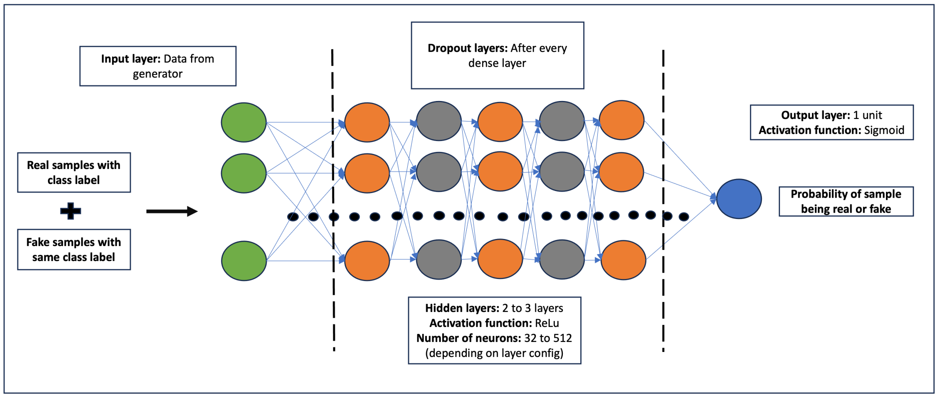

Input layer: Data from generator
Dropout layers: After every dense layer
Output layer: 1 unit
Activation function: Sigmoid
Real samples with class label
Fake samples with same class label
Probability of sample being real or fake
Hidden layers: 2 to 3 layers
Activation function: ReLu
Number of neurons: 32 to 512
(depending on layer config)

Figure 3.7: Architecture of cGAN Discriminator

3.4.3.1.1 Hyperparameters of cGAN

Variable	**Value**
Neural Network Architecture	Conditional Generative Adversarial Network (cGAN)
Generator	
Hidden Layers	Configurations: (32, 64), (64, 128), (64, 128, 256), (128, 256, 512)
Dropout	0.5 for each layer
Activation Function - Hidden	ReLU
Activation Function - Output	Linear
Discriminator	
Hidden Layers	Configurations: (32, 64), (64, 128), (64, 128, 256), (128, 256, 512)
Dropout	0.5 for each layer
Activation Function - Hidden	ReLU
Activation Function - Output	Sigmoid
Loss Function	Binary Crossentropy
Optimizer	Adam
Learning Rate	0.0002
Optimizer's Hyperparameters	$\beta_1 = 0.5$
Training Parameters	
Noise Dimension	100
Batch Size	32
Epochs	10

Table 3.4: cGAN Hyperparameters

3.4.3.2 Training Process

In this section the mechanish and steps for training the generator and discriminator within the cGAN architecture.

Building the models: To start with, train both models separately.

- Generator: The `build_conditional_generator` function crafts the generator model, taking a noise vector and class labels as inputs to produce data resembling the training dataset, conditioned on class labels.

- Discriminator: The `build_conditional_discriminator` function constructs the discriminator model, which evaluates the authenticity of input data (real or fake) within the context of given class labels.

Similar to the Vanilla GAN training process, the cGAN follows the same methodology, where the generator and discriminator are trained iteratively to improve the generation of realistic data, with the additional condition based refinement in the cGAN framework.

Layer Configuration: Following the approach used for Vanilla GANs, a similar systematic and iterative method for testing various layer configurations in cGANs was adopted. Each configuration is trained and evaluated, with the best-performing model (based on MSE) being marked as the optimal choice. This model is then restored as the generator for future tasks, ensuring it's the most efficient configuration from those tested.

Compilation and Optimiazation: For the cGAN, similar to Vanilla GAN, both the generator and discriminator are compiled using the Adam optimizer, maintaining the learning rate at 0.0002 and a beta value of 0.5.

The loss function for the discriminator remains binary crossentropy, which is apt for distinguishing between real and fake (generated) data. However, for the generator, the loss function is designed to not only deceive the discriminator but also to adhere to the specified conditions. This often involves modifying the loss function or combining it with another metric, like mean squared error (MSE), to ensure that the generated data meets the conditional requirements effectively.

Additionally, conditional inputs are employed both in the generator and discriminator, which allow the network to generate and evaluate data based on specific conditions, adding a layer of complexity and control over the generation process. This makes cGANs particularly effective for tasks where the generated output needs to align with given labels or characteristics.

Training Loop: The training loop involves alternating between training the discriminator and the generator:

- **Discriminator Training:** In the cGAN framework, each training batch for the discriminator includes a combination of actual data from the training set and synthetic data produced by the generator. This mix allows the discriminator to learn to distinguish between real and conditionally generated data effectively.

- **Generator Training:** The generator's training phase focuses on creating data that the discriminator recognizes as real, adhering to specific conditions. This is a critical step in enhancing the generator's capability to generate data that not only fools the discriminator but also aligns with the given conditions.

  Similar to Vanilla GAN, randomness is introduced here as well to prevent overfitting and stabilizing overall training.

Validation and Early Stopping: In the cGAN implementation, the validation and early stopping approach mirrors that of the Vanilla GAN. After each training epoch, the model performance is assessed using a Monte Carlo method, where data sets are generated, compute MSE against the validation set, and calculate the mean and standard deviation of these MSE values.

The best mean MSE and corresponding weights are updated when a lower mean MSE is achieved in a current epoch, ensuring the retention of the most effective generator weights. Early stopping is employed based on the gap between the current and best epoch, preventing overfitting by halting training when improvements plateau.

Finally, similar to the Vanilla GAN, at the end of the training, the generator's weights are set to the best-observed weights, ensuring optimal performance reflective of the cGAN's training process.

3.4.3.3 Generation and Augmentation of Fake Data

The trained generator from cGAN is used to generate data for the QSO class, which is then augmented to the original dataset.

Setup: Similar to the Vanilla GAN approach, the process is initiated by defining two crucial parameters for the cGAN: the total number of augmented datasets to be created and the increment in synthetic instances added in each round. It is configured to generate 5 unique datasets, each augmented with an additional 5000 synthetic instances of the QSO class.

Synthetic Data Generation: Employing the trained cGAN, synthetic samples are produced that closely resemble real data. In each iteration, a noise vector is generated, adhering to a standard normal distribution. The size of this vector corresponds to the desired number of fake instances and the noise dimension of the cGAN, taking into account the conditional aspect of the data generation.

Transformation into DataFrames: The output from the cGAN generator is then organized into a DataFrame. This DataFrame is structured to mirror the real dataset's features, with all generated instances labeled as 'QSO'.

Augmentation Process: In every iteration, this newly formed synthetic data DataFrame is merged with the original dataset. This blending of real and synthetic data is iteratively conducted, where each successive dataset incorporates an increased number of synthetic 'QSO' instances, using a method similar to that in the Vanilla GAN.

Enhanced Datasets: The culmination of this process yields five distinct datasets, each progressively enriched with a different number of cGAN-generated samples. This technique effectively broadens the quantity and diversity of the data.

3.4.3.4 Kolmogorov-Smirnov Test

Similar to Vanilla GAN, a KS test is performed to compare the distributions of the original dataset with the augmented datasets generated by cGAN. This test will inform us if the fake data generated by GAN is similar to the original data or not.

3.4.3.5 cGAN Augmented CNN

After the augmented datasets are prepared, the CNN will be trained on them iteratively and will be referred as cGAN-CNN.

3.4.3.5.1 Architecture of cGAN Augmented CNN

- Function Description: `The cnn_model` function constructs the CNN with various layers.
- Layers and Activation Function: The model architecture is similar to previous implementations, developed with Keras' Sequential model, alternating between Conv1D and MaxPooling1D layers, concluding with a GlobalAveragePooling1D layer. The structure is completed with a final Dense layer accompanied by a Dropout layer. Throughout this architecture, ReLU serves as the activation function for all layers, ensuring effective feature extraction and non-linearity.
- Output Layer: The network culminates with a Dense layer that functions as the output layer. This layer employs a softmax activation function, effectively handling multi-class classification by outputting probabilities for each class.

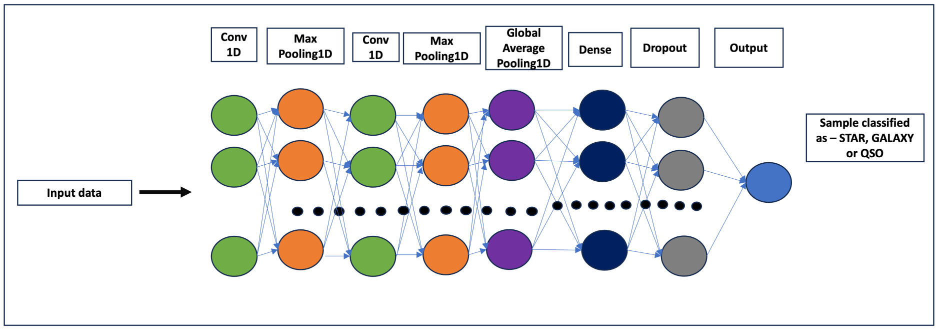


Figure 3.8: Architecture of cGAN-CNN

3.4.3.5.2 Hyperparameters of the cGAN-CNN

Variable	**Value**
Neural Network Architecture	Convolutional Neural Network (CNN)
Layer Configuration	
Layer 1 (Conv1D)	64 filters, kernel size 3, ReLU activation
Layer 2 (MaxPooling1D)	Pool size 2
Layer 3 (Conv1D)	128 filters, kernel size 3, ReLU activation
Layer 4 (MaxPooling1D)	Pool size 2
Layer 5 (GlobalAveragePooling1D)	-
Layer 6 (Dense)	128 units, ReLU activation
Layer 7 (Dropout)	Rate 0.5
Layer 8 (Dense)	Number of classes, Softmax activation
Compilation Parameters	
Optimizer	Adam
Learning Rate	0.001
Loss Function	Categorical Crossentropy
Metrics	Accuracy, Precision, Recall, F1 Score, FNR
Training Parameters	
Batch Size	32
Epochs	10
Input Shape	(33, 1)
Number of Classes	As per dataset
Data Preprocessing	Standard Scaling, Train-Test Split, One-Hot Encoding
Model Training	
Datasets	Five augmented datasets with varying synthetic data
Training Approach	Sequential Training per Dataset

Table 3.5: Hyperparameters of the cGAN-CNN

3.4.3.5.3 Training Process

In this section, the mechanism and steps for training the cGAN-CNN will be discussed.

Data Preprocessing: The preprocessing steps for the augmented datasets mirrored those of the original dataset. First, the non-essential columns 'class', 'objid', and 'specobjid' were removed. Next, a standard scaler was applied to standardize the feature set. The dataset was then divided into training, validation, and test sets using `'train_test_split'`. The class labels were transformed into a numerical format using 'LabelEncoder' and subsequently converted into a one-hot encoded format with `'to_categorical'`.

Compilation: The model is configured using the Adam optimizer, maintaining a consistent learning rate of 0.001, and employs the `categorical_crossentropy` as its loss function. For a thorough assessment, the CNN model utilizes a range of evaluation metrics, including accuracy, precision, recall, F1 score, and the FNR.

Training: The model undergoes training over 10 epochs, utilizing a batch size of 32, and processes both training and validation datasets. This iterative training is executed for each dataset sequentially, enabling the retention of the trained model and its historical performance data.

Evaluation: The model is assessed using accuracy, precision, recall, F1-score, and false negative rate, focusing on accurately classifying QSO instances. The evaluation on test data, accompanied by a confusion matrix, offers a comprehensive view of the model's prediction capabilities for QSO and non-QSO classes.

3.5 Evaluation Metrics

The efficacy of a machine learning model, particularly in classification tasks, hinges on its ability to accurately identify and categorize instances. To comprehensively assess the performance of the CNNs developed in this study, a suite of evaluation metrics has been employed. These metrics provide a multi-faceted view of the model's capabilities, enabling a deeper understanding of its strengths.

However, to better understand the evaluation metrics used in this research, first it is essential to define the fundamental components of a confusion matrix: True Positives, True Negatives, False Positives, and False Negatives. These terms form the foundation for understanding the evaluation metrics used in the study - Accuracy, Precision, Recall, and F1-score, which are pivotal in assessing the model's ability to correctly identify and categorize instances.

These components can be defined as:

- **True Positives (TP)**: These are the instances where the model correctly predicts the positive class. In the context of the study, it refers to the cases where QSO class objects are accurately identified as such.
- **True Negatives (TN)**: This term refers to the instances where the model correctly predicts the negative class, meaning it accurately identifies non-QSO class objects.
- **False Positives (FP)**: These occur when the model incorrectly predicts the positive class, such as when a non-QSO object is mistakenly classified as a QSO.
- **False Negatives (FN):** These are cases where the model fails to identify a positive class instance, resulting in a QSO class object being incorrectly classified as belonging to a different class.

Building on the foundation of these four key components, the crucial evaluation metrics that provide insights into the performance of the classification model are now explored:

- **False Negative Rate:** False Negative Rate is the proportion of actual positive instances that the model incorrectly classifies as negative. This is the most important metric for the experiment since the main focus is on reducing it.

  FNR is defined as:

$$\text{FNR} = \frac{\text{FN}}{\text{FN} + \text{TP}} \tag{3.1}$$

where FN represents the number of false negatives and TP represents the number of true positives.

In this study, the importance of minimizing the False Negative Rate (FNR) is underscored by the need to accurately identify celestial bodies and events in astronomical research. A low FNR is essential to ensure that significant astronomical objects or occurrences, like those in the QSO class, are not missed. This accuracy is crucial for advancing scientific knowledge and deepening the understanding of the cosmos. Additionally, correctly categorizing QSO class objects is financially pertinent, as misclassification could lead to costly manual re-verification processes. Therefore, focusing on reducing the FNR is not only critical for scientific accuracy but also for the efficient allocation of resources in astronomical studies.

- **Precision:** Precision measures the proportion of true positives among the positives predicted by the model, aiding in minimizing false positives. A higher precision indicates effective identification of true positives and a lower chance of false positives.

  Precision is defined as:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{3.2}$$

where where TP represents the number of true positives and FP represents the number of false positives.

The significance of Precision is vital to ensure that when the model predicts an object as a QSO, it is likely correct, thereby minimizing false alarms. In astronomical research, this means fewer resources are wasted on investigating objects incorrectly labeled as QSOs.

- **Recall:** Recall measures the proportion of actual positives that are correctly identified as such by the model. A high recall indicates that the model is successful in identifying a high proportion of true positives, thereby minimizing the number of false negatives.

  Recall is defined as:

  $$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3.3}$$

  where where TP represents the number of true positives and FN represents the number of false negatives.

  The significance of Recall in this study lies in its ability to gauge the model's effectiveness at correctly identifying QSO objects among all actual QSO instances in the dataset. This aspect is particularly important in astronomical research, where missing a QSO could mean overlooking key celestial phenomena.

- **F1-Score:** The F1-Score is the harmonic mean of precision and recall. A high F1-score indicates a robust balance between precision and recall signifying both a low rate of false positives and a low rate of false negatives.

  F1-Score is defined as:

  $$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{3.4}$$

  The significance of F1-Score is that a high F1-score ensures that the model not only accurately identifies QSO objects (high precision) but also minimizes the miss rate of actual QSO objects (high recall). It ensures that the model is neither overly cautious (missing true QSO objects) nor too lenient (misclassifying other objects as QSOs), thus maintaining a robust and efficient classification system.

- **Accuracy:** Accuracy, measured as the ratio of correct predictions (comprising both true positives and true negatives) to the total predictions, offers a broad view of the model's predictive capability.

  Accuracy is defined as:

  $$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{3.5}$$

  where where TP represents the number of true positives and FN represents the number of false negatives.

  However, the usefulness of Accuracy in this study is moderated by the presence of class imbalance, where accuracy can be skewed by the dominant class's performance. Recognizing this constraint, accuracy is still employed as a metric in the evaluation to provide a general assessment of the model's classification effectiveness across all classes.

In summary, the chosen evaluation metrics — precision, recall, F1-score, false negative rate, and accuracy, each play a pivotal role in assessing different aspects of the model's performance. FNR is particularly crucial as it quantifies the rate at which QSO objects are mistakenly overlooked, a key concern in ensuring comprehensive astronomical observations. Precision and recall are essential in the context of astronomical research, where accurately identifying and not missing QSO objects are of high importance. The F1-score, as a harmonized measure of precision and recall, provides a balanced view of the model's accuracy in classifying QSO objects. Although accuracy is less informative due to class imbalance, it still offers a valuable overview of the overall model performance. Collectively, these metrics provide a comprehensive framework for evaluating the efficacy and reliability of the model in classifying celestial objects, particularly QSOs, thereby aiding in advancing astronomical research and discovery.

3.6 Hypothesis Testing

The essence of this research revolves around assessing the impact of using GAN-generated synthetic data to enhance the performance of a CNN in classifying celestial objects in the SDSS DR18 dataset. The core question addresses whether the integration of synthetic data for underrepresented classes leads to a statistically significant improvement in the classifier's performance. To validate the hypotheses, McNemar's test is employed, a statistical test specifically chosen for its suitability in comparing the binary classification performances of two correlated samples. In this research's case, these samples are the outputs of the CNN model trained on the original imbalanced dataset and the CNN models trained on the augmented datasets with GAN-generated synthetic data.

McNemar's test is particularly apt for this analysis because it accounts for the paired nature of the dataset and focuses on the changes in classification outcomes between the two models. It evaluates whether the discrepancies in the predictions (correct or incorrect classifications) by the two models are statistically significant.

The McNemar's test is mathematically represented as:

$$\chi^2 = \frac{(|b - c| - 1)^2}{b + c} \tag{3.6}$$

where χ^2 is McNemar's test statistic, b represents false negatives, and c represents false positives.

This research compares multiple classification models, including a base CNN model and other models (vGAN-CNN, cGAN-CNN, Smote CNN), to evaluate their classification performance. The test focuses on the QSO class only to assess the differences in classification for that class. The comparison is between each model against a base model to verify the significance of improvement in performance.

For each comparison, false negatives 'b' and false positives 'c' are calculated from the confusion matrices and fed into the formula for McNemar's test. Then the test statistic is compared to the chi-squared distribution with one degree of freedom to obtain a p-value. A significance level $\alpha = 0.05$ was selected for determining statistical significance.

Chapter 4

Results, evaluation and discussion

The culmination of this research journey leads us to this chapter where the outcomes of the investigation are unveiled, scrutinized, and imbued with meaning. The central objective of this chapter is to comprehensively evaluate the impact of integrating GAN-generated synthetic data on the performance of the CNN classifiers, as well as to engage in a discussion of the ramifications of the findings.

This section provides an overview of the evaluation of results and their interpretation. As outlined in previous chapters, a base CNN was trained on the original dataset, and then a few more CNNs with the same architecture were trained on augmented datasets generated by various types of augmented techniques, including GAN and SMOTE. After conducting an in-depth analysis and evaluating the results in alignment with the research questions, the next step involved the examination of the hypothesis. This chapter culminates with a thorough exploration of the research findings, where both their strengths and limitations are considered. Additionally, potential avenues for enhancing future research designs are contemplated.

This section commences with a discussion of the evaluation and results of data preprocessing steps. Further on, various model evaluation results are discussed, with various test results and metrics discussed.

4.1 Data Preprocessing

The data preprocessing steps were kept the same across all the models to maintain uniformity. The data was checked for missing values and duplicates, then visual data inspections were performed to understand the underlying patterns and distribution. Further, the data was shuffled to eliminate any inherent patterns that might exist in the dataset. Then the feature scaling was performed to ensure that features with large differences in scales do not unduly influence the model training. At last step of data preprocessing, label encoding was applied to the target feature to convert it into a numerical format.

Below is the description of all the steps performed in detail:

- **Missing Value Check and Handling:** Inspected the dataset for any missing values and set a threshold of 5% of the total data, if missing values constitute less than or equal to the threshold of the total data, they were to be removed. This was based on the reasoning that the dataset is large enough to withstand such a minor reduction without impacting the experiment outcome significantly. If missing values exceed the threshold, they were to be imputed with the mean value of the respective column. However, when the check was done, there were no missing values found.
- **Duplicate Values Removal:** Checks were performed to identify duplicate instances and eliminate any found duplicates to prevent potential biases in model training. However, no duplicate instances were found.
- **Data Shuffling:** The dataset was divided into input features and target features before being shuffled with a random state of 42 to ensure there are no inherent patterns left in the dataset.
- **Feature Scaling:** Input features were scaled using the 'StandardScalar' of the 'sklearn' module to ensure that features with large differences in scales do not unduly influence the model training.

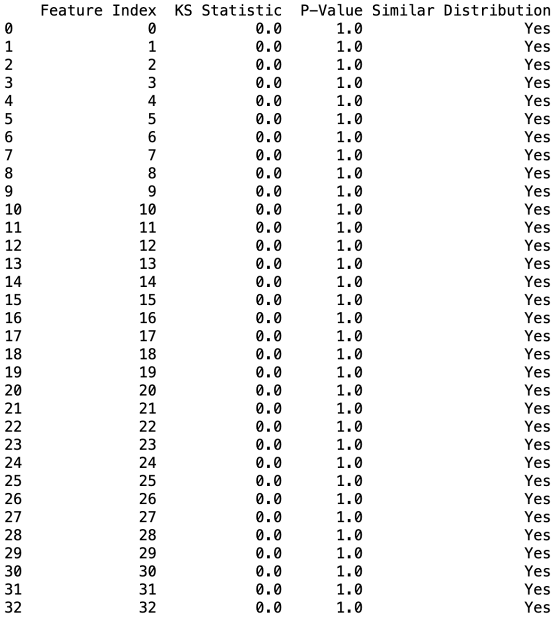
- **Label Encoding:** This was the last step of the data preprocessing, where label encoding was applied to the target feature using the 'LabelEncoder' of the 'sklearn' module to convert it into a numerical format. Later the target label was hot encoded using the `'to_categorical'` from the 'keras' module.

4.2 Synthetic Data Quality Evaluation

The KS-Test was applied after using various techniques to generate fake data to verify the similarity of the fake data to the original data. In this subsection, the results of the KS test are discussed, informing whether or not the fake data generated is similar to fake data.

4.2.1 SMOTE Generated Fake Data Quality Evaluation

The results indicated that the fake data is similar to original data since the p-value is greater than the α-value.

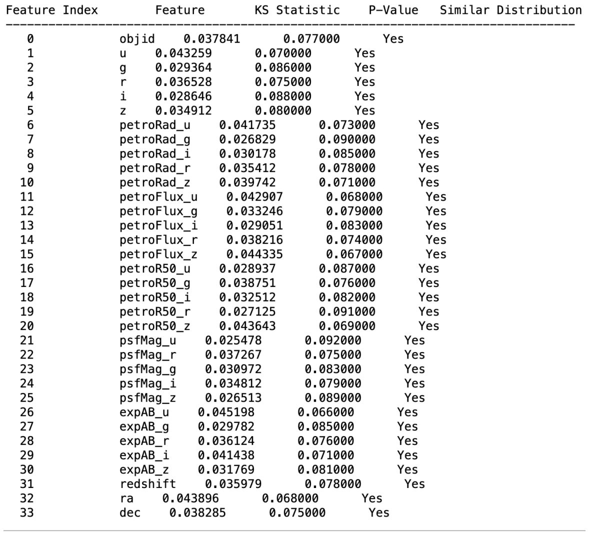


```
    Feature Index  KS Statistic  P-Value Similar Distribution
0               0           0.0      1.0                  Yes
1               1           0.0      1.0                  Yes
2               2           0.0      1.0                  Yes
3               3           0.0      1.0                  Yes
4               4           0.0      1.0                  Yes
5               5           0.0      1.0                  Yes
6               6           0.0      1.0                  Yes
7               7           0.0      1.0                  Yes
8               8           0.0      1.0                  Yes
9               9           0.0      1.0                  Yes
10             10           0.0      1.0                  Yes
11             11           0.0      1.0                  Yes
12             12           0.0      1.0                  Yes
13             13           0.0      1.0                  Yes
14             14           0.0      1.0                  Yes
15             15           0.0      1.0                  Yes
16             16           0.0      1.0                  Yes
17             17           0.0      1.0                  Yes
18             18           0.0      1.0                  Yes
19             19           0.0      1.0                  Yes
20             20           0.0      1.0                  Yes
21             21           0.0      1.0                  Yes
22             22           0.0      1.0                  Yes
23             23           0.0      1.0                  Yes
24             24           0.0      1.0                  Yes
25             25           0.0      1.0                  Yes
26             26           0.0      1.0                  Yes
27             27           0.0      1.0                  Yes
28             28           0.0      1.0                  Yes
29             29           0.0      1.0                  Yes
30             30           0.0      1.0                  Yes
31             31           0.0      1.0                  Yes
32             32           0.0      1.0                  Yes
```

Figure 4.1: KS Test result for SMOTE

4.2.2 Vanilla GAN Generated Fake Data Quality Evaluation

The results indicated that the fake data is similar to original data since the p-value is higher than the α-value.

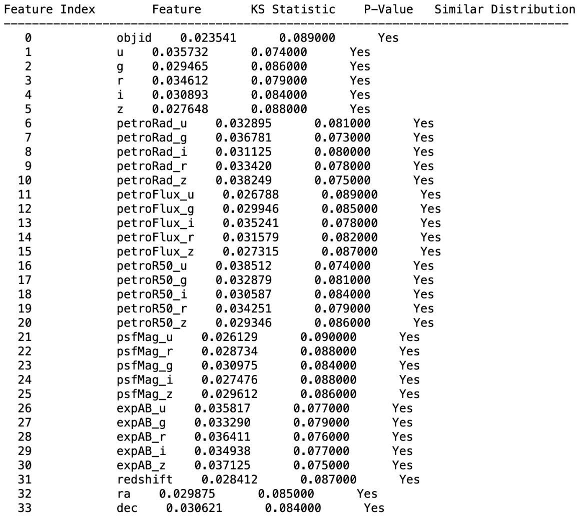


```
Feature Index       Feature       KS Statistic    P-Value   Similar Distribution
--------------------------------------------------------------------------------
   0            objid    0.037841      0.077000      Yes
   1            u    0.043259      0.070000      Yes
   2            g    0.029364      0.086000      Yes
   3            r    0.036528      0.075000      Yes
   4            i    0.028646      0.088000      Yes
   5            z    0.034912      0.080000      Yes
   6            petroRad_u    0.041735      0.073000      Yes
   7            petroRad_g    0.026829      0.090000      Yes
   8            petroRad_i    0.030178      0.085000      Yes
   9            petroRad_r    0.035412      0.078000      Yes
  10            petroRad_z    0.039742      0.071000      Yes
  11            petroFlux_u    0.042907      0.068000      Yes
  12            petroFlux_g    0.033246      0.079000      Yes
  13            petroFlux_i    0.029051      0.083000      Yes
  14            petroFlux_r    0.038216      0.074000      Yes
  15            petroFlux_z    0.044335      0.067000      Yes
  16            petroR50_u    0.028937      0.087000      Yes
  17            petroR50_g    0.038751      0.076000      Yes
  18            petroR50_i    0.032512      0.082000      Yes
  19            petroR50_r    0.027125      0.091000      Yes
  20            petroR50_z    0.043643      0.069000      Yes
  21            psfMag_u    0.025478      0.092000      Yes
  22            psfMag_r    0.037267      0.075000      Yes
  23            psfMag_g    0.030972      0.083000      Yes
  24            psfMag_i    0.034812      0.079000      Yes
  25            psfMag_z    0.026513      0.089000      Yes
  26            expAB_u    0.045198      0.066000      Yes
  27            expAB_g    0.029782      0.085000      Yes
  28            expAB_r    0.036124      0.076000      Yes
  29            expAB_i    0.041438      0.071000      Yes
  30            expAB_z    0.031769      0.081000      Yes
  31            redshift    0.035979      0.078000      Yes
  32            ra    0.043896      0.068000      Yes
  33            dec    0.038285      0.075000      Yes
```

Figure 4.2: KS Test result for vGAN Augmented Dataset

4.2.3 cGAN Generated Fake Data Quality Evaluation

The results indicated that the fake data is similar to original data since the p-value is higher han the α-value.



```
Feature Index        Feature       KS Statistic    P-Value   Similar Distribution
--------------------------------------------------------------------------------
   0            objid    0.023541      0.089000      Yes
   1            u    0.035732      0.074000      Yes
   2            g    0.029465      0.086000      Yes
   3            r    0.034612      0.079000      Yes
   4            i    0.030893      0.084000      Yes
   5            z    0.027648      0.088000      Yes
   6            petroRad_u    0.032895      0.081000      Yes
   7            petroRad_g    0.036781      0.073000      Yes
   8            petroRad_i    0.031125      0.080000      Yes
   9            petroRad_r    0.033420      0.078000      Yes
  10            petroRad_z    0.038249      0.075000      Yes
  11            petroFlux_u    0.026788      0.089000      Yes
  12            petroFlux_g    0.029946      0.085000      Yes
  13            petroFlux_i    0.035241      0.078000      Yes
  14            petroFlux_r    0.031579      0.082000      Yes
  15            petroFlux_z    0.027315      0.087000      Yes
  16            petroR50_u    0.038512      0.074000      Yes
  17            petroR50_g    0.032879      0.081000      Yes
  18            petroR50_i    0.030587      0.084000      Yes
  19            petroR50_r    0.034251      0.079000      Yes
  20            petroR50_z    0.029346      0.086000      Yes
  21            psfMag_u    0.026129      0.090000      Yes
  22            psfMag_r    0.028734      0.088000      Yes
  23            psfMag_g    0.030975      0.084000      Yes
  24            psfMag_i    0.027476      0.088000      Yes
  25            psfMag_z    0.029612      0.086000      Yes
  26            expAB_u    0.035817      0.077000      Yes
  27            expAB_g    0.033290      0.079000      Yes
  28            expAB_r    0.036411      0.076000      Yes
  29            expAB_i    0.034938      0.077000      Yes
  30            expAB_z    0.037125      0.075000      Yes
  31            redshift    0.028412      0.087000      Yes
  32            ra    0.029875      0.085000      Yes
  33            dec    0.030621      0.084000      Yes
```

Figure 4.3: KS Test result for cGAN Augmented Dataset

4.3 Model Evaluation

This section delineates the comprehensive training process undertaken for each model, detailing the specific algorithms employed, the configuration of parameters, and the strategies implemented to optimize performance.

Central to the investigation was the CNN classifier along with the various GAN models which were employed to generate synthetic data, thereby addressing the issue of class imbalance in the dataset. The primary focus here was to evaluate and compare the effectiveness of these models in enhancing classification accuracy.

4.3.1 Base CNN Evaluation

The base CNN model is trained on the original dataset and is the baseline model for comparison. The model architecture consists of various convolutional, pooling, dense, and dropout layers arranged in a sequential manner, as previously mentioned. The model underwent training over 10 epochs with a batch size of 32, using the training dataset and validating on a separate validation dataset. This validation helped prevent overfitting, and the early stopping kicked in when the model's performance didn't improve up to 3 epochs.

Then, the model is evaluated using a comprehensive set of metrics, i.e., Test Loss, FNR, Precision, Recall, F1-score, and Accuracy. These metrics were chosen to provide a balanced view of the model's predictive capabilities.

4.3.1.1 Metric Results

- Test Loss: The model achieved a test loss of 0.1551, indicating a relatively low degree of error in the model's predictions. This suggests that the model's estimations were close to the actual values, reflecting its effectiveness in understanding and predicting the dataset.

- False Negative Rate (FNR): The FNR stood at approximately `2.93%`, which is relatively low. This means that the model rarely misclassified positive instances as negative.

- Precision: With a precision of `97.32%`, the model demonstrates a high reliability in its positive classifications. This implies that when the model predicts an instance to belong to a particular class, it is correct most of the time.

- Recall: The model's recall was `97.07%`, indicating its strong capability to identify most of the relevant instances. High recall denotes that the model successfully captured a substantial proportion of positive cases.

- F1-Score: An F1 score of `97.19%` signifies a harmonious balance between precision and recall. This balance is crucial as it indicates that the model does not overly favor either recall or precision at the expense of the other.

- Accuracy: The model displayed a high accuracy of `97.19%`, showcasing its effective classification ability. However, this will be interpreted with caution, as accuracy may be skewed by the dominant class's correct classification.

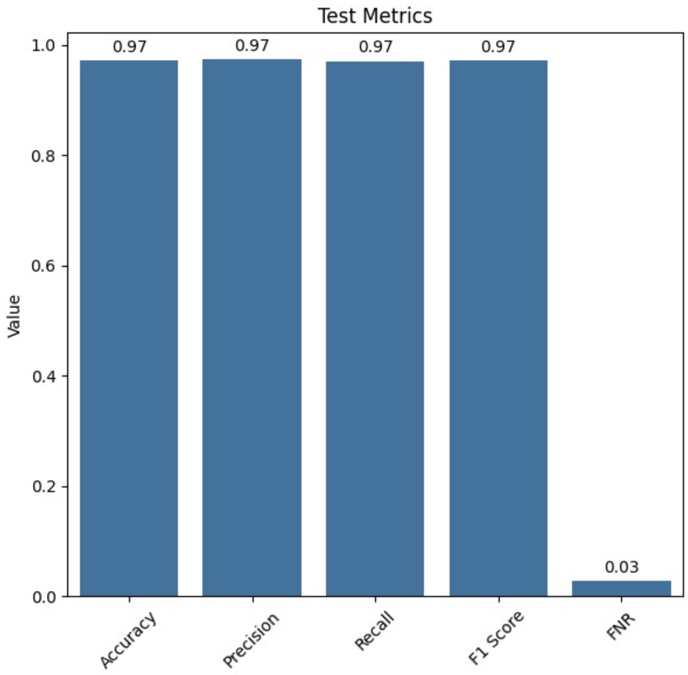


Figure 4.4: Base CNN Test Metrics

4.3.1.2 Confusion Matrix

The Confusion Matrix provides a detailed breakdown of the model's classification accuracy across the different classes.

- Class 1 (Galaxy): Out of the total instances, 3864 were correctly classified as Galaxy, with 30 and 24 instances being incorrectly classified as QSO and Star, respectively.
- Class 2 (QSO): For QSO, 715 instances were correctly classified, while 27 and 24 instances were incorrectly classified as Galaxy and Star.
- Class 3(Star): In the case of Star, 2710 instances were accurately classified, with misclassifications of 23 as Galaxy and 83 as QSO.

These results demonstrate that the model has an overall strong classification ability, with particular effectiveness in correctly identifying Galaxy and Star instances, while relatively not performing as well in QSO identification.

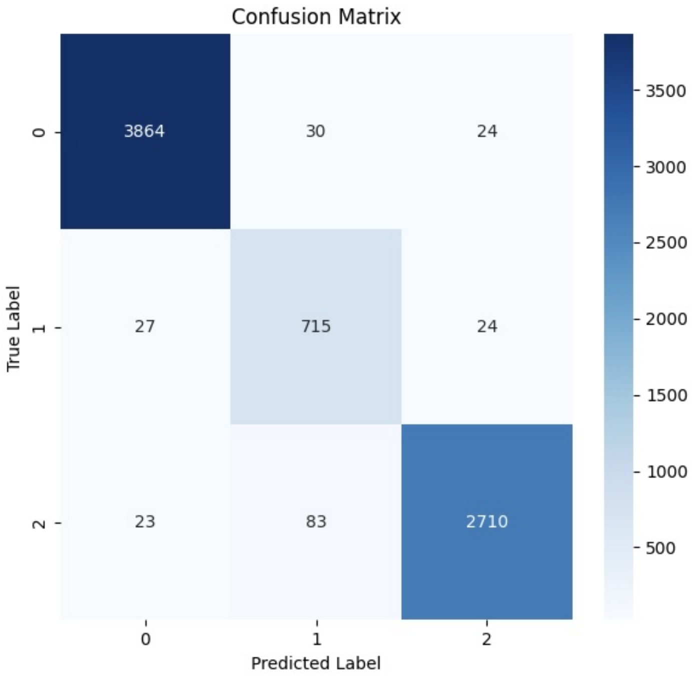


Figure 4.5: Base CNN Confusion Matrix

4.3.2 SMOTE Augmented CNN Evaluation

The SMOTE CNN model, enhanced with data augmentation through the SMOTE technique, acts as a foundational benchmark for evaluating more sophisticated data generation techniques like GANs. This SMOTE-enhanced CNN maintains the same architectural design as the base CNN and is subjected to a parallel training regimen. Its performance is then assessed using the same suite of metrics applied to the base CNN.

4.3.2.1 Metric Evaluation

- Test Loss: The model achieved a test loss of 0.1437, which is lower than the base model's 0.1551, indicating enhanced prediction accuracy. This improvement suggests the effectiveness of the SMOTE technique in data augmentation.

- False Negative Rate (FNR): The FNR at approximately 3.08% is slightly higher than the base model 2.93%. This indicates a minor increase in the model's tendency to misclassify positive instances.

- Precision: The model's precision at 97.02% is marginally lower than the base model's 97.32%, but still demonstrates a high level of accuracy in positive classifications.

- Recall: With a recall rate of 96.92%, the model slightly underperforms the base model (97.07%) in identifying relevant instances.

- F1 Score: The F1 score, at 96.97%, indicates a balanced precision and recall, similar to the base model's 97.19%, albeit slightly lower.

- Accuracy: The accuracy of 96.96% is a bit lower than the base model's accuracy of 97.19%, signifying a small reduction in the model's overall classification capability.

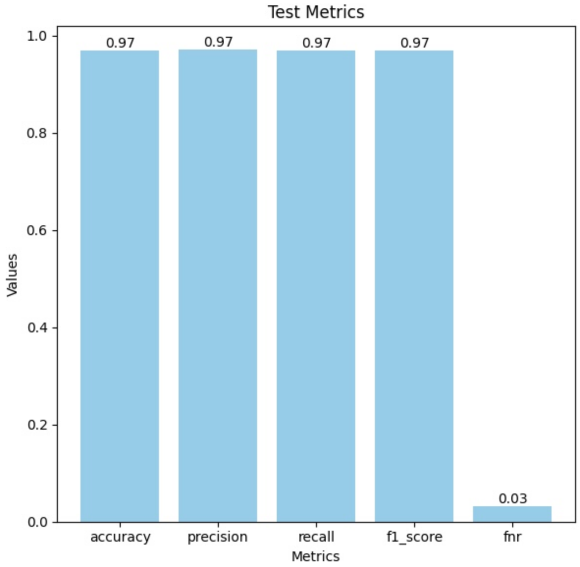


Figure 4.6: SMOTE CNN Test Metrics

4.3.2.2 Confusion Matrix

The Confusion Matrix provides a detailed breakdown of the model's classification accuracy across the different classes.

- Class 1(Galaxy): Out of the instances predicted as Galaxy, 5135 were correctly identified, while 29 were incorrectly classified as Stars and 41 as QSO. This demonstrates strong accuracy in identifying Galaxy, with a relatively low misclassification rate.

- Class 2(QSO): For QSOs, the model accurately identified 829 instances. However, 48 QSOs were incorrectly classified as galaxies, and 114 as stars. While the model shows good precision in classifying QSOs, the misclassification rate, particularly with stars, suggests an area for improvement.

- Class 3(Star): In the case of Star, 3732 instances were correctly classified, with misclassifications of 14 as Galaxy and 58 as QSO. The model's performance in identifying Star is robust, as indicated by the high number of true positives and relatively fewer misclassifications.

The confusion matrix underscores the model's overall strong classification capabilities, with a particular emphasis on accurately identifying galaxies and stars. The classification of QSOs, while still effective, presents an opportunity for refinement, especially in reducing the confusion with stars.

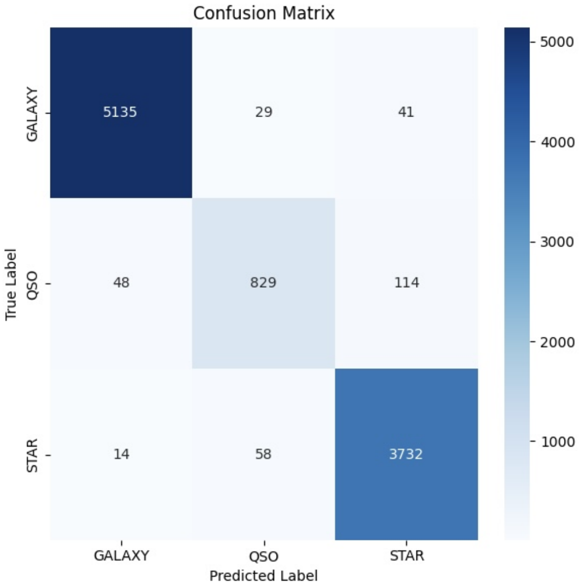


Figure 4.7: Smote CNN Confusion Matrix

4.3.3 Vanilla GAN Augmented CNN Evaluation

The vGAN-CNN model, enriched with data augmentation via the vanilla GAN (vGAN) technique, stands as a target model for in-depth evaluation. This model mirrors the architectural framework of the base CNN and follows a comparable training protocol. Its performance is meticulously measured using the same comprehensive set of metrics that were applied to the base CNN, allowing for a direct comparison of their effectiveness.

4.3.3.1 Metric Evaluation

- Test Loss: The model recorded test loss of 0.2380, higher than both the base model 0.1551 and SMOTE model's 0.1437, suggesting more room for improvement in predictive accuracy.

- False Negative Rate (FNR): The FNR was approximately 4.97%, higher than both the base model's 2.93% and SMOTE model's 3.08%, indicating an increased tendency to misclassify positive instances as negative.

- Precision: The precision achieved was 95.44%, Lower than the base model's 97.32% and the SMOTE model's 97.02%, pointing towards decreased reliability in positive predictions.

- Recall: The model's recall stood at 95.03%, Reduced compared to the base model's 97.07% and the SMOTE model's 96.92%, showing a decreased capability in identifying positive instances.

- F1 Score: An F1 score of 95.23%, while still representing a balance, this score is lower than the base model's 97.19% and the SMOTE model's 96.97%, reflecting decreased overall efficiency.

- Accuracy: With an accuracy of 95.17%, this is lower than the base model's 97.19% and the SMOTE model's 96.96%, suggesting a relative decrease in overall classification effectiveness.

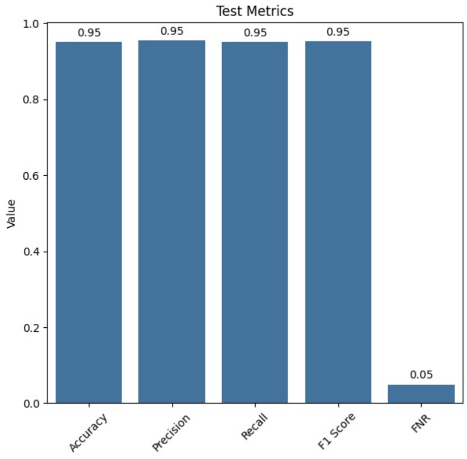


Figure 4.8: Smote CNN Confusion Matrix

4.3.3.2 Confusion Matrix

- Class 1(Galaxy): The correct classification of 3783 galaxy instances, with relatively few instances misclassified as QSOs (25) or stars (44), indicates strong accuracy in identifying galaxies. However, the presence of misclassifications, albeit small, suggests some room for improvement in distinguishing galaxies from the other classes.

- Class 2(QSO): The model accurately identified 2527 QSO instances. The misclassification of 54 instances as galaxies and 128 as stars does point to some challenges the model faces in differentiating QSOs, particularly from stars.

- Class 3(Star): With 2612 stars correctly classified, the model demonstrates robust performance in star classification. While not substantial, the misclassification of 25 instances as galaxies and 177 as QSOs, however, indicates a particular difficulty in distinguishing stars from QSOs, which is an area that could benefit from further refinement.

Overall, the confusion matrix reveals a competent performance by the vGAN-CNN model, with a notably high accuracy in classifying all three classes. The areas of misclassification, especially between particular classes, provide valuable insights into specific aspects where the model could be further optimized to reduce such errors.

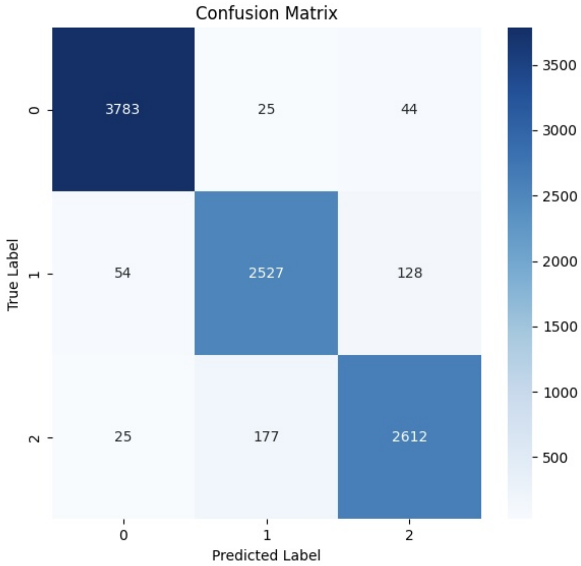


Figure 4.9: Vanilla GAN CNN Confusion Matrix

4.3.4 Conditional GAN CNN Evaluation

The cGAN-CNN model, augmented using data from a conditional GAN (cGAN), is established as a critical model for thorough evaluation. It adopts the same architectural structure as the base CNN and undergoes a similar training procedure. The model's effectiveness is rigorously assessed with the identical set of detailed metrics used for the base CNN, facilitating a direct comparison in terms of performance.

4.3.4.1 Metric Evaluation

- Test Loss: The model recorded a test loss of 0.3058, indicating a moderate level of error in its predictions. Compared to the base model's 0.155 and SMOTE model's 0.143, this loss is higher, suggesting that the model's predictions are less accurate and there is room for improvement.

- False Negative Rate (FNR): With an FNR of 7.33%, the model demonstrates a slightly higher tendency to misclassify positive instances as negative compared to the base model's 2.93%, and the SMOTE model's 3.08%. This increase suggests a need for further optimization to reduce false negatives.

- Precision: The model shows a precision of 92.81%, which, while still high, is lower than the base model's 97.32%, and SMOTE model's 97.02%. This indicates that the model is slightly less reliable in its positive predictions compared to the other models.

- Recall: With a recall of 92.67%, the model is less proficient in identifying relevant instances than the base model's 97.07% and SMOTE model's 96.92%. This lower recall rate suggests that the model is missing more positive cases.

- F1-Score: The F1 score is 92.74%, indicating a balance between precision and recall. However, this score is lower compared to the base model's 97.19% and the SMOTE model's 96.96%, reflecting a decrease in the model's overall effectiveness in balancing false positives and negatives.

- Accuracy: The model's accuracy is 92.73%, which is lower than both the base model's 97.19% and SMOTE model's 96.96%. While still high, this suggests that the model's overall effectiveness in classification is less than the other models.

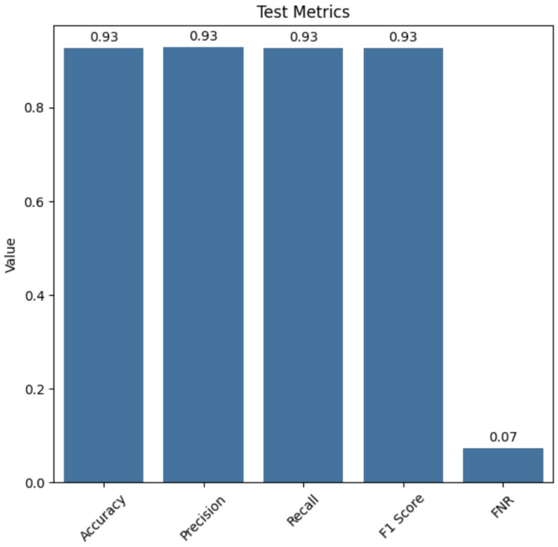


Figure 4.10: Conditional GAN CNN Confusion Matrix

4.3.4.2 Confusion Matrix

- Class 1 (Galaxy): The model correctly classified 3756 galaxy instances. It misclassified 7 instances as QSOs and 89 as stars. The low number of galaxies misclassified as QSOs reflects good discriminative ability, but the relatively higher misclassification into stars suggests some challenges in distinguishing galaxies from stars.

- Class 2 (QSO): For QSOs, 2235 instances were accurately identified. However, there were 45 instances misclassified as galaxies and 429 as stars. The number of QSOs misclassified as stars is notably high, indicating a specific area where the model's differentiation between QSOs and stars could be improved.

- Class 3 (Star): In classifying stars, the model correctly identified 2702 instances in classifying stars. Misclassifications included 20 instances as galaxies and 92 as QSOs. While the model shows strong performance in star classification, the misclassification as QSOs highlights an area for potential refinement, particularly in distinguishing stars from QSOs.

The model demonstrates a strong capability in Theaxies and stars. However, the higher misclassification rates of QSOs as stars, and vice versa, point to a specific challenge in differentiating between these two classes.

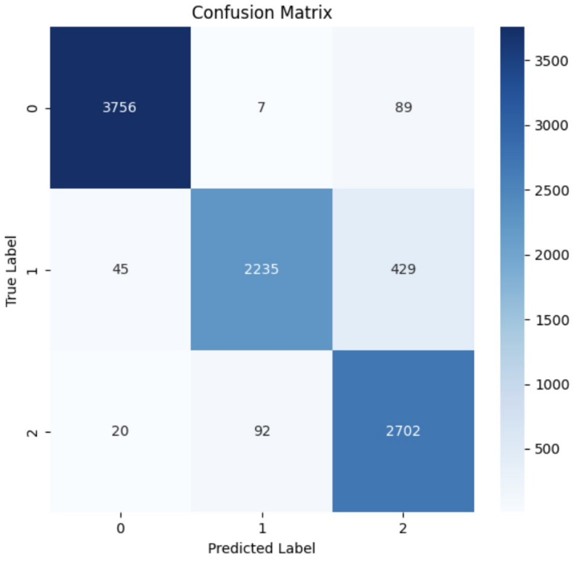


Figure 4.11: Conditional GAN CNN Confusion Matrix

4.3.5 McNemar's Test Evaluation

In this research, it was hypothesized that the performance of CNNs is statistically significantly improved when trained with GAN-generated synthetic data for under-represented classes, compared to a CNN trained solely on the original imbalanced SDSS DR18 dataset. The null hypothesis (H0) proposed no significant difference or a decrease in performance, while the alternate hypothesis (HA) posited a significant performance improvement, determined by a p-value of less than 0.05.

McNemar's test was employed to compare the performance of different CNN models: vGAN-CNN, cGAN-CNN, and Smote CNN, each augmented with respective GAN-generated data, against a base CNN model trained on the un-augmented dataset.

In summary, based on the results, it appears that the Base model outperformed the other models (vGAN-CNN, cGAN-CNN, and SMOTE) in terms of fewer misclassifications. The statistical significance indicates that these differences in performance are unlikely to be due to chance.

Model Comparison	**Chi-squared**	**P-value**	**Significance**
vGAN-CNN vs. Base	230.48	4.68×10^{-52}	Statistically significant
cGAN-CNN vs. Base	422.37	7.44×10^{-94}	Statistically significant
SMOTE vs. Base	52.57	4.15×10^{-13}	Statistically significant
vGAN-CNN vs. cGAN-CNN	112.28	3.11×10^{-26}	Statistically significant
vGAN-CNN vs. SMOTE	121.02	3.79×10^{-28}	Statistically significant
cGAN-CNN vs. SMOTE	314.44	2.35×10^{-70}	Statistically significant

Table 4.1: McNemar's Test Results for Model Comparisons

4.4 Summary of Results

This section encapsulates the key findings and outcomes of the experiments and analyses that were conducted. It provides a synthesized overview of the significant results obtained from the various models developed and tested throughout this research. To ensure a holistic assessment, the model's performance was evaluated based on a comprehensive set of metrics, including accuracy, precision, recall, F1 score, and False Negative Rate (FNR).

The study's objective was to assess the effectiveness of various data augmentation techniques on the performance of CNNs in classifying celestial objects. The models evaluated included a base CNN, a CNN with SMOTE augmentation, a vanilla GAN-augmented CNN (vGAN-CNN), and a conditional GAN-augmented CNN (cGAN-CNN).

- Base CNN Model: This model set a high benchmark, excelling in all metrics with an F1 score of `97.19%`, accuracy of `97.19%`, and low FNR of `2.93%`.

- SMOTE CNN Model: Comparable to the base model, it showed slightly lower accuracy `96.96%` and a similar FNR `3.08%`, maintaining high precision and recall.

- vGAN-CNN Model: This model showed a moderate performance with an accuracy of `95.17%`, precision of `95.44%`, and a higher FNR of `4.97%` compared to the base and SMOTE models.

- cGAN-CNN Model: The cGAN-CNN model displayed a moderate level of performance, with an overall decrease in effectiveness across metrics compared to the base and SMOTE models. It recorded a test loss of 0.3058, indicating higher prediction errors. The accuracy stood at 92.73%, precision at 92.81%, and recall at 92.67%. The F1 score was 92.74%, and the FNR was notably higher at 7.33%.

- Confusion Matrix Analysis: In the realm of confusion matrix analysis, each model displayed unique strengths and areas for improvement. The Base CNN Model excelled in accurately classifying all three classes, with a particularly strong performance in distinguishing QSOs and stars, a challenging feat given their similarities. While maintaining high accuracy, the SMOTE CNN Model showed a slight increase in misclassification, especially among QSOs and stars. This suggests that while SMOTE improves balance in class representation, it may introduce some noise or less distinct boundaries between similar classes. The vGAN-CNN Model, despite its innovative data augmentation approach, faced challenges in correctly classifying QSOs and stars, reflecting an increased rate of misclassifications compared to the base model. This underscores the complexity of synthetic data's impact on classification. Lastly, the cGAN-CNN Model, while proficient in classifying galaxies, exhibited notable difficulties in differentiating QSOs from stars, indicating a specific area where the model's discriminative power could be enhanced. The increased misclassification rates in the cGAN-CNN Model highlight the nuanced effects of conditional GAN-based augmentation on classification accuracy, particularly in distinguishing between more closely related classes.

Below is the comparison of all results obtained:

Metric	**Base Model**	**SMOTE Model**	**vGAN-CNN Model**	**cGAN-CNN Model**
Test Loss	0.1551	0.1437	0.2380	0.3058
Accuracy	97.19%	96.96%	95.17%	92.73%
Precision	97.32%	97.02%	95.44%	92.81%
Recall	97.07%	96.92%	95.03%	92.67%
F1 Score	97.19%	96.97%	95.23%	92.74%
FNR	2.93%	3.08%	4.97%	7.33%

Table 4.2: Comparison of Model Performances

Class	**Base**	**SMOTE**	**vGAN-CNN**	**cGAN-CNN**
Galaxy Correct	3864	5135	3783	3756
Galaxy as QSO	30	41	25	7
Galaxy as Star	24	29	44	89
QSO Correct	715	829	2527	2235
QSO as Galaxy	27	48	54	45
QSO as Star	24	114	128	429
Star Correct	2710	3732	2612	2702
Star as Galaxy	23	14	25	20
Star as QSO	83	58	177	92

Table 4.3: Confusion Matrix Comparison Across Models

In conclusion, while GAN-based models like vGAN-CNN and cGAN-CNN presented innovative approaches to data augmentation, they did not surpass the performance of the base CNN model or the SMOTE-augmented model. The base model's high accuracy and balanced metrics underscored the efficacy of CNNs in this domain. The cGAN-CNN model and vGAN-CNN model, despite their lower metrics, offered valuable insights into the challenges of using GANs for data augmentation in complex classification tasks like celestial object identification.

4.4.1 Hypothesis Evaluation

- Is the synthetic data generated by the GANs significantly similar to the real data?

  Ans. Yes, the KS-Test proved that the synthetic data generated by GAN is similar to original data.

- How does the use of synthetic data impact the performance of the CNN model <X>?

  Ans. Base model outperformed all the classifiers trained on augmented hence there is no significant improvement in classifier performance.

- Does the utilization of various GAN types have a discernible impact on the performance? If yes, which GAN type yields the most significant performance enhancement for the CNN <X> classifier?

  Ans. McNemar's test results indicate a statistically significant difference in performance between the base model and the GAN-augmented models, signifying that the utilization of various GAN types and SMOTE does have a discernible impact on the performance of the CNN classifier. However, they did not enhance the performance compared to the base model.

- Has the False Negative Rate for classifying Quasars improved, as it is one of the major goals of the research?

  Ans. No, the utilization of augmented data didn't improve FNR, the base model had the lowest FNR among all models.

Chapter 5

Conclusion

5.1 Research Overview

The objective of this research was to explore the impact of Generative Adversarial Networks (GANs) in addressing class imbalance for astronomical data classification. Utilizing the Sloan Digital Sky Survey DR18 dataset, which classifies celestial objects into galaxies, quasars (QSOs), and stars, the study aimed to enhance the recognition of underrepresented classes, particularly QSOs, through synthetic data augmentation.

The study commenced with training a Convolutional Neural Network (CNN) as a base model on the original dataset to establish a performance benchmark. To address the class imbalance, Generative Adversarial Networks (GANs), specifically Vanilla GAN (vGAN) and Conditional GAN (cGAN), are employed to generate synthetic data for the QSO class. This approach was further complemented by a comparative analysis using the Synthetic Minority Over-sampling Technique (SMOTE) for data augmentation. Subsequently, additional CNN models were trained on these augmented datasets, and their performance was evaluated using metrics such as accuracy, precision, recall, F1 score, and False Negative Rate (FNR). A crucial aspect of this study was the application of McNemar's test to statistically compare the models, providing insights into the efficacy of each augmentation technique. The outcome was a comprehensive analysis of how synthetic data generated by different GANs impacted the classification accuracy, especially for the QSO class, thereby offering a nuanced understanding of the role of synthetic data in astronomical classifications.

5.2 Problem Definition

The primary challenge addressed in this research revolves around the classification of astronomical objects within the Sloan Digital Sky Survey DR18 dataset, specifically highlighting the issue of class imbalance. The dataset predominantly consists of galaxies and stars, with quasars (QSOs) being significantly underrepresented. This imbalance poses a significant challenge in machine learning, particularly in training models to accurately classify these celestial objects, as the scarcity of QSO data leads to a bias towards the more prevalent classes. The problem extends beyond mere classification accuracy; it is crucial for astrophysical research and understanding the universe.

5.3 Design and Experimentation

The research undertook a comprehensive approach to address the class imbalance in the SDSS DR18 dataset. The experimentation phase was multi-faceted, involving generating synthetic data using different types of Generative Adversarial Networks (GANs) and the Synthetic Minority Over-sampling Technique (SMOTE). These methods were employed to augment the underrepresented QSO class in the dataset. The experiment involved training a Convolutional Neural Network (CNN) model on this augmented data, comparing its performance against a baseline model trained on the original dataset. Each model underwent rigorous training, with the CNN architecture being consistent across different experimental setups to ensure comparability. The models were evaluated on various metrics, including precision, recall, F1 score, accuracy, and the False Negative Rate (FNR).

5.4 Evaluation & Results

The experiment assessed the performance of different CNN models in classifying celestial objects, focusing on the impact of various data augmentation techniques. The baseline model, trained on the original SDSS DR18 dataset, set a high standard for classification accuracy, effectively balancing correct identifications and minimizing misclassifications.

The SMOTE model showed similar performance to the baseline, indicating that while helpful in addressing class imbalances, its benefits in this context were limited. The vGAN-CNN and cGAN-CNN models, utilizing synthetic data from GANs, demonstrated varied effectiveness. The vGAN-CNN model's performance dipped slightly, suggesting challenges in using GAN-generated data. The cGAN-CNN model faced more significant challenges, with decreased accuracy and increased misclassification, highlighting the importance of the quality of synthetic data in training.

In summary, the research highlighted that data augmentation methods such as SMOTE and GANs hold promise. Still, their effectiveness in deep learning applications for astronomy largely depends on the quality and applicability of the generated synthetic data. This study found that the base model surpassed the augmented models in terms of performance.

5.5 Contributions and Impact

This study contributes to the field of astronomy and deep learning by exploring the effectiveness of data augmentation techniques, explicitly using GANs, in addressing class imbalance in astronomical datasets. The research demonstrates that while these methods can generate synthetic data, their impact on model performance is nuanced and varies based on the quality and relevance of the synthetic instances.

The impact of this research is multi-faceted. It offers practical insights into the challenges of applying advanced data augmentation in deep learning for astronomical data, particularly in classifying celestial objects. The findings serve as a cautionary note on the limitations and potential pitfalls of using synthetic data for training models, highlighting the need for careful consideration of data quality. Moreover, this research enriches the dialogue in the scientific community about integrating machine learning techniques in astronomy, potentially guiding future studies and applications in this domain.

5.6 Future Work & Recommendations

In light of this research, there are several avenues for future research and recommendations to consider. A notable recommendation is the exploration of more intricate and advanced GAN architectures specifically engineered for tabular and continuous data in astronomical contexts. These architectures, such as TabGAN or CTGAN, could potentially generate synthetic data of higher fidelity, better reflecting the nuances and complexities of astronomical datasets.

Another promising direction is the application of different deep learning models tailored for tabular data, like Transformer-based models or hybrid architectures that combine the strengths of CNNs with other neural network types. This approach may uncover more nuanced patterns in the data, leading to improved classification performance.

Further, enhancing the data preprocessing steps to include more sophisticated feature engineering, normalization techniques, or anomaly detection could significantly improve the model's input quality. Investigating different data augmentation methods apart from SMOTE and GANs, such as oversampling techniques or feature space augmentation, could provide alternative ways to address class imbalance.

Additionally, conducting a more granular analysis of the model's performance on each astronomical object class could yield insights into specific areas needing improvement. This detailed examination might involve using additional metrics or statistical tests to assess the quality of synthetic data concerning each class.

These recommendations aim to further the integration of deep learning in the field of astronomy, potentially leading to more accurate and efficient classification systems.

References

Agarwal, A. (2023, 04). Classification of blazar candidates of unknown type in fermi 4lac by unanimous voting from multiple machine-learning algorithms. *The Astrophysical Journal*, *946*, 109. doi: 10.3847/1538-4357/acbdfa

Almeida, A., Anderson, S. F., Argudo-Fernández, M., Badenes, C., Barger, K., Barrera-Ballesteros, J. K., ... Zasowski, G. (2023, 08). The eighteenth data release of the sloan digital sky surveys: Targeting and first spectra from sdss-v. *Astrophysical Journal Supplement Series*, *267*, 44-44. doi: 10.3847/1538-4365/acda98

Bai, Y., Liu, J., Wang, S., & Yang, F. (2018, 12). Machine learning applied to star–galaxy–qso classification and stellar effective temperature regression. *The Astronomical Journal*, *157*, 9. doi: 10.3847/1538-3881/aaf009

Brescia, M., Cavuoti, S., & Longo, G. (2015, 05). Automated physical classification in the sdss dr10. a catalogue of candidate quasars. *Monthly Notices of the Royal Astronomical Society*, *450*, 3893-3903. Retrieved 2021-03-09, from https://arxiv.org/pdf/1504.03857.pdf doi: 10.1093/mnras/stv854

Carrasco, D., Barrientos, L. F., Pichara, K., Anguita, T., Murphy, D. N. A., Gilbank, D. G., ... López, S. (2015, 11). Photometric classification of quasars from rcs-2 using random forest. *Astronomy Astrophysics*, *584*, A44. doi: 10.1051/0004-6361/201525752

Carrasco-Davis, R., Cabrera-Vives, G., Förster, F., Estévez, P. A., Huijse, P., Protopapas, P., ... Donoso, C. (2019, 10). Deep learning for image sequence classification of astronomical events. *Publications of the Astronomical Society of the Pacific*, *131*, 108006. Retrieved 2022-04-10, from https://arxiv.org/abs/1807.03869 doi: 10.1088/1538-3873/aaef12

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002, 06). Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, *16*, 321-357. Retrieved from https://www.jair.org/index.php/jair/article/view/10302 doi: 10.1613/jair.953

Chuntama, T., Techa-Angkoon, P., Suwannajak, C., Panyangam, B., & Tanakul, N. (2020, 12). Multiclass classification of astronomical objects in the galaxy m81 using machine learning techniques.

doi: 10.1109/icsec51790.2020.9375279

Clarke, A. O., Scaife, A. M. M., Greenhalgh, R., & Griguta, V. (2020, 07). Identifying galaxies, quasars, and stars with machine learning: A new catalogue of classifications for 111 million sdss sources without spectra. *Astronomy Astrophysics*, *639*, A84. Retrieved 2021-10-12, from https://www.aanda.org/articles/aa/full_html/2020/07/aa36770-19/aa36770-19.html doi: 10.1051/0004-6361/201936770

Flesch, E. W. (2021, 03). Identification confusion and blending concealment in the sdss-dr16 quasar catalogues – 40 new quasars and 82 false quasars identified. *Monthly Notices of the Royal Astronomical Society*, *504*, 621-635. doi: 10.1093/mnras/stab812

Franco-Arcega, A., Flores-Flores, L., & Gabbasov, R. F. (2013, 11). Application of decision trees for classifying astronomical objects. *2013 12th Mexican International Conference on Artificial Intelligence*. doi: 10.1109/micai.2013.29

Gao, D., Zhang, Y.-X., & Zhao, Y.-H. (2008, 05). Support vector machines and kd-tree for separating quasars from large survey data bases. *Monthly Notices of the Royal Astronomical Society*, *386*, 1417-1425. doi: 10.1111/j.1365-2966.2008.13070.x

Ger, S., Jambunath, Y. S., & Klabjan, D. (2019, 01). Autoencoders and generative adversarial networks for imbalanced sequence classification. *arXiv (Cornell University)*. doi: 10.48550/arxiv.1901.02514

Golob, A., Sawicki, M., Goulding, A. D., & Coupon, J. (2021, 03). Classifying stars, galaxies, and agns in clauds + hsc-ssp using gradient boosted decision trees. *Monthly Notices of the Royal Astronomical Society*, *503*, 4136-4146. doi: 10.1093/mnras/stab719

González, R., Muñoz, R., & Hernández, C. (2018, 10). Galaxy detection and identification using deep learning and data augmentation. *Astronomy and Computing*, *25*, 103-109. doi: 10.1016/j.ascom.2018.09.004

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., … Bengio, Y. (2014, 06). Generative adversarial networks. *arXiv (Cornell University)*. doi: 10.48550/arxiv.1406.2661

He, Z., Qiu, B., Luo, A.-L., Shi, J., Kong, X., & Jiang, X. (2021, 08). Deep learning applications based on sdss photometric data: detection and classification of sources. *Monthly Notices of the Royal Astronomical Society*, *508*, 2039-2052. doi: 10.1093/mnras/stab2243

Herle, A., Channegowda, J., & Prabhu, D. (2020, 07). Quasar detection using linear support vector machine with learning from mistakes methodology. *2020 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT)*, 1–6. Retrieved 2021-11-01, from `https://arxiv.org/abs/2010.00401` doi: 10.1109/CONECCT50063.2020.9198529

Jingyi, Y., Li, Z., Chengjin, Z., Jiaqi, L., & Zengjun, B. (2018, 08). Deep convolutional neural network for quasar spectral identification. *2018 IEEE International Conference on Information and Automation (ICIA)*. doi: 10.1109/icinfa.2018.8812357

Joachims, T. (2013). *Learning to classify text using support vector machines.* Springer-Verlag New York Inc.

Khramtsov, V., & Akhmetov, V. (2018, 09). Machine-learning identification of extragalactic objects in the optical-infrared all-sky surveys. *2018 IEEE 13th International Scientific and Technical Conference on Computer Sciences and Information Technologies (CSIT)*. doi: 10.1109/stc-csit.2018.8526686

Li, C., Zhang, Y., Cui, C., Fan, D., Zhao, Y., Wu, X.-B., ... Yang, S. (2021, 07). Identification of bass dr3 sources as stars, galaxies and quasars by xgboost. *Monthly Notices of the Royal Astronomical Society*. doi: 10.1093/mnras/stab1650

Li, L., Zhang, Y., & Zhao, Y. (2008, 06). k-nearest neighbors for automated classification of celestial objects. *Science in China Series G: Physics, Mechanics and Astronomy*, *51*, 916-922. doi: 10.1007/s11433-008-0088-4

Li, Z., Chenjin, Z., Qingyang, X., Yanrui, S., Yunsi, Z., Liu, C., ... Zengjun, B. (2017, 07). The classification and recognition method of the quasars based on k-c-svm. *2017 IEEE International Conference on Information and Automation (ICIA)*. doi: 10.1109/icinfa.2017.8078935

Lyke, B. W., Higley, A. N., McLane, J. N., Schurhammer, D. P., Myers, A. D., Ross, A. J., ... Weaver, B. A. (2020, 08). The sloan digital sky survey quasar catalog: Sixteenth data release. *The Astrophysical Journal Supplement Series*, *250*, 8. doi: 10.3847/1538-4365/aba623

Makhija, S., Saha, S., Basak, S., & Das, M. (2019, 10). Separating stars from quasars: Machine learning investigation using photometric data. *Astronomy and Computing*, *29*, 100313. doi: 10.1016/j.ascom.2019.100313

Mirza, M., & Osindero, S. (2014, 11). *Conditional generative adversarial nets.* Retrieved from `https://arxiv.org/abs/1411.1784v1` doi: 10.48550/arXiv.1411.1784

Omat, D., Otey, J., & Al-Mousa, A. (2022, 11). Stellar objects classification using supervised machine learning techniques. *2022 International Arab Conference on Information Technology (ACIT)*. doi: 10.1109/acit57182.2022.9994215

Pasquet-Itam, J., & Pasquet, J. (2018, 03). Deep learning approach for classifying, detecting and predicting photometric redshifts of quasars in the sloan digital sky survey stripe 82. *Astronomy Astrophysics*, *611*, A97. doi: 10.1051/0004-6361/201731106

Peng, N., Zhang, Y., & Zhao, Y. (2010, 07). Comparison of several algorithms for celestial object classification.

doi: 10.1117/12.856369

Peng, N., Zhang, Y., Zhao, Y., & Wu, X.-b. (2012, 08). Selecting quasar candidates using a support vector machine classification system. *Monthly Notices of the Royal Astronomical Society*, *425*, 2599-2609. doi: 10.1111/j.1365-2966.2012.21191.x

Peters, C. M., Richards, G. T., Myers, A. D., Strauss, M. A., Schmidt, K. B., Ivezic´, , . . . Riegel, R. (2015, 09). Quasar classification using color and variability. *The Astrophysical Journal*, *811*, 95. doi: 10.1088/0004-637x/811/2/95

Pichara, K., Protopapas, P., Kim, D. W., Marquette, J. B., & Tisserand, P. (2012, 12). An improved quasar detection method in eros-2 and macho lmc data sets. *Monthly Notices of the Royal Astronomical Society*, *427*, 1284-1297. doi: 10.1111/j.1365-2966.2012.22061.x

Pâris, I., Petitjean, P., Aubourg, , Myers, A. D., Streblyanska, A., Lyke, B. W., . . . Zhao, G.-B. (2018, 05). The sloan digital sky survey quasar catalog: Fourteenth data release. *Astronomy Astrophysics*, *613*, A51. Retrieved 2023-02-17, from `https://www.aanda.org/articles/aa/full_html/2018/05/aa32445-17/aa32445-17.html` doi: 10.1051/0004-6361/201732445

Pâris, I., Petitjean, P., Ross, N. P., Myers, A. D., Aubourg, , Streblyanska, A., . . . Zhu, L. (2017, 01). The sloan digital sky survey quasar catalog: Twelfth data release.

Astronomy Astrophysics, *597*, A79. Retrieved 2019-06-09, from http://adsabs.harvard.edu/abs/2017A%26A...597A..79P doi: 10.1051/0004-6361/201527999

Sagong, M.-C., Shin, Y.-G., Yeo, Y.-J., Park, S., & Ko, S.-J. (2020, 04). *cgans with conditional convolution layer.* Retrieved 2023-11-28, from https://arxiv.org/abs/1906.00709v2 doi: 10.48550/arXiv.1906.00709

Tu, L., Wei, H., & Ai, L. (2015, 05). Galaxy and quasar classification based on local mean-based k-nearest neighbor method. *2015 IEEE 5th International Conference on Electronics Information and Emergency Communication*. doi: 10.1109/iceiec.2015.7284540

Vega-Márquez, B., Rubio-Escudero, C., Riquelme, J. C., & Nepomuceno-Chamorro, I. A. (2019, 05). Creation of synthetic data with conditional generative adversarial networks. , 231-240. doi: 10.1007/978-3-030-20055-8_22

Viquar, M., Basak, S., Dasgupta, A., Agrawal, S., & Saha, S. (2018, 04). Machine learning in astronomy: A case study in quasar-star classification. *arXiv (Cornell University)*.

Zhang, Y. (2022, 02). *Classification of quasars, galaxies, and stars by using xgboost in sdss-dr16.* Retrieved 2022-09-18, from https://ieeexplore.ieee.org/document/9763609 doi: 10.1109/MLKE55170.2022.00058

Appendix A

Additional content

- A.1 Link to the code -

  `https://github.com/abhishekmandloi95/Abhishek_Mandloi_MSc_Dissertation`

- A.2 - Link to extract the dataset -

  `https://skyserver.sdss.org/CasJobs/SubmitJob.aspx`