Deciphering the Cosmos: Advanced Classification Techniques for Galaxy, Quasar, and Star Identification

Applied Data Mining STAT7240
Kennesaw State University, Georgia

Kelly Lavertu
klavertu@students.kennesaw.edu

***Abstract*—** The classification of astronomical objects plays a crucial role in advancing our quest for extraterrestrial life and deepening our understanding of the universe's origins. The task is challenging due to the typically grainy nature of images of galaxies, stars, and quasars, which makes distinguishing their features with the naked eye difficult. This necessitates the training of sophisticated algorithms capable of accurately discerning these celestial bodies. Given the subtle differences in the appearance of stars, quasars, and galaxies to human observers, it is essential for these algorithms to classify and predict with high precision. Accurate classification is pivotal, as it could be the key to identifying Earth-like exoplanets. This project aims to develop and evaluate three classification algorithms – Random Forest, XGBoost, and a Convolutional Neural Network (CNN) – for their effectiveness in differentiating between these types of astronomical objects. The focus is on assessing the performance and practical utility of these widely used machine learning algorithms in the context of astronomical object classification.

Keywords — Galaxies, stars, quasars, image classification, machine learning algorithms

1. Introduction

Classification of celestial objects is fundamental to the understanding of the universe and how it was formed. Galaxy typing can lead to a better understanding of how a galaxy has emerged and changed over time. Since quasars are some of the brightest objects in the universe, they can offer valuable insight into galaxy formation. We can also determine the potential habitability of planets in the orbit of successfully categorized stars. Up until recently, the characterization of galaxies, quasars, and stars was a cumbersome process. However, years of research have led to new developments in machine learning algorithms that have made it possible for us to classify these objects faster by utilizing computers to learn and classify these objects themselves.

As part of this project, there will be an implementation of three different classifiers for the purposes of characterizing different types of galaxies, quasars, and stars based on certain defining features. The goal is to discover the best-performing algorithms by optimizing hyperparameter tuning for each model, thereby improving upon methods from similar studies. One such study includes Plewa's random forest classifier, which achieved an overall accuracy of 84% [1]. In another model, Tarsitano et al. used XGBoost with PCA decomposition and achieved 86% accuracy and 93% accuracy for early- and late-type galaxies, respectively [2].

There were a few articles dedicated to the use of neural networks for astronomical image classification. Accuracies of 98% (2 classes), 94% (3 classes), and 82% (10 classes) were achieved in Shaiakhmetov et al.'s SpinalNet classifer to categorize galaxies into several subclasses [3]. The last model referenced was from Hui et al.'s work on a Galaxy Morphology Classification with DenseNet. They compared several pre-trained CNN models and were able to obtain accuracies between 83% and 89% [4].

There are, however, some key differences between this project and the aforementioned studies. Specifically, this research is dedicated to the classification of galaxies, quasars, and stars. One of the other studies is concentrated on classifying subclasses of stars, while another is based on categorizing subclasses of galaxies. Additionally, the output of the algorithm will give insights into how accurate the classifiers were at classifying different galaxies, quasars, and stars. The final project will describe:

- Project's feature set and classifier design—what type of classifier was chosen and why
- Project's training and test data
- Implementation of classifier including the design and code development
- Classifier performance and accuracy compared to previous methods
- Strengths and weaknesses of the classification approach

2. Methodology

2.1 Dataset

In this project, a custom dataset was used by fetching images from the Sloan Digital Sky Survey (SDSS) SciServer database. The images are 175x175 and are in color, which can be seen in Figure 1. There are approximately 30,625 features for images 175x175 in size. 3,000 images were used for the Random Forest and XGBoost models and were split as follows:

- 1,000 images per category
- 80% training for a total of 2,400 images
- 20% testing for a total of 600 images

Since a validation set was introduced for the CNN, more images were gathered for a total of 4,499 images—they were split as follows:

- 1,500 images per category
- ~80% training for a total of 3,605 images
- ~10% validation for a total of 444 images
- ~10% testing for a total of 450 images

Due to how the database is set up, preprocessing the images was mostly built into the SQL queries required to obtain the images. For example, we were able to select a number of galaxies based on their type, i.e., spiral, elliptical, or unknown and then build the label names from these chosen characteristics. Additionally, the CNN required rescaling at a factor of 1/255 because the images are in color.

Galaxy *Quasar* *Star*

Figure 1: Examples of a galaxy, quasar, and star taken from the dataset

2.2 Methods

In the field of machine learning, there is the ubiquitous challenge of astronomical image classification. There have been many attempts to build a successful system to classify celestial objects. One of the most challenging aspects of creating such a system is that the features of a particular celestial object can be muddled since images can be grainy and identifiable features can potentially overlap. In an effort to improve upon previous methods, this project compares the results for 3 algorithms and then compares them to established research. There are 3 different algorithms used for this project:

I. Random Forest
II. XGBoost
III. Convolutional Neural Network

I. RANDOM FOREST

The Random Forest algorithm, an ensemble method, operates by constructing multiple decision trees and then integrating their outcomes. This approach enhances the prediction accuracy and robustness of the model. For the purposes in this project, each image from the testing set is compared pixel by pixel to each image in the training set based on the most significant features of the images. This process is repeated for each subset until the forest is fully grown. Once the model is trained, it can be used to classify new images based on the rules of the forest and then assign the image to the appropriate class.

This type of algorithm was chosen because we wanted to compare the results of a supervised algorithm to those of an unsupervised algorithm, like the CNN. Often times, the random forest model doesn't overfit with the introduction of more features. It can also handle both numerical and categorical data, as well as multi-output problems.

Finally, the accuracy is usually high for these types of algorithms.

II. XGBoost

XGBoost, an advanced ensemble technique, utilizes gradient boosting on decision trees. It combines multiple weak predictive models to form a more accurate and powerful aggregate model. This method is known for its efficiency and effectiveness in various machine learning challenges.

The XGBoost algorithm was chosen because like the Random Forest model, it usually has a higher accuracy. Additionally, the algorithm is designed to be fast even though it can be computationally intense because of the memory usage. Regularization can also be used to correct overfitting and generalize more to the test data.

III. CNN

The third and final algorithm used in this project is the Convolutional Neural Network. This type of algorithm is selected due to its exceptional ability to process large datasets, especially those involving complex image data. Unlike traditional algorithms, CNNs excel in identifying intricate patterns in images through layered processing. This deep learning approach allows the network to improve its feature detection and classification accuracy iteratively.

The inputs are the set of pixels from the n by n grid the images originate from, in this case 175x175. The architecture of the algorithm, beginning with a starting node for each input pixel, also contains two hidden layers and three activation functions. The first hidden layer of the neural network is a linear layer. The input, the previously mentioned nodes, goes into 128 nodes. The next step is the first ReLU function. Following the ReLU function is the second linear hidden layer that starts with 128 nodes as input and scales down to 64 nodes. From here, there is another ReLU function to help train the model. Following the second ReLU activation function there is another linear hidden layer with an input size of 175 and an output size of 3. Finally, there is one more activation function, called Softmax.

ReLU (Rectified Linear Unit) is a pivotal activation function in neural networks, designed to introduce non-linearity into the model. It activates a node only if the input is above a certain threshold, thus facilitating faster and more effective learning. On the other hand, Softmax serves as a probabilistic activation function, typically used in the final layer of a neural network to represent probabilities of multiple classes, making it essential for multi-class classification tasks.

A significant enhancement to this CNN model is the incorporation of the Adam optimizer. This optimizer represents an advanced form of Stochastic Gradient Descent, efficiently adjusting the network weights iteratively based on training data. It is known for its effectiveness in converging rapidly to high-quality solutions, even in complex and high-dimensional spaces. The purpose of the Adam optimizer is to properly tune the parameters by minimizing the loss function associated with the function that is the algorithm.

3. Results

3.1 Random Forest Results

The accuracy of the Random Forest testing data was around 92% for the best model, which was an 8% improvement from Plewa's model [1]. The galaxies were predicted at 99% accuracy, quasars at 91%, and stars at 92%. The precision, recall, and F1-score remained relatively high for galaxy classification, while the quasar and star categories were consistently around 90%. More details can be seen in the classification report in Table 1.

Random Forest Algorithm			
	Precision	**Recall**	**F1-score**
Galaxy	97%	100%	99%
Quasar	90%	88%	89%
Star	90%	90%	90%
Accuracy			92%

Table 1: Classification report for galaxy, quasar, and star accuracy using the Random Forest Algorithm

Interestingly, Run 6, which took an hour, was seven times longer than Run 7 despite training approximately 38% less data. We suspect that the reduction in mtry, or the number of features considered for the best split at each node, is the cause of the dramatic decrease in training time. As a result, training only took around 8 minutes in Run 7, which is impressive given the amount of training data. This was achieved with 1,000 trees and a minimum number of leaf nodes of 1. Ultimately, these were the hyperparameters chosen for testing— see Table 2 for more details on hyperparameter tuning.

Random Forest									
Run #	**Accuracy**	**Kappa**	**mtry**	**min.node.size**	**splitrule**	**num.trees**	**cv folds**	**Training Size**	**Runtime**
1	82%	0.74	346	1, 3, 5, 7, 9	gini	4	10	10%	5 minutes
2	81%	0.72	2, 4, 6, 8, 346	1, 3, 5, 7, 9	gini	4	10	10%	25 minutes
3	73%	0.6	2, 4, 2006	1, 3, 2005	gini	4	10	10%	9 minutes
4	80%	0.71	346	1	gini	8	10	10%	1 minute
5	85%	0.78	346	1	gini	100	10	10%	1 minute
6	90%	0.85	3065	1	extratrees	1000	10	50%	1 hour
7	92%	0.87	489	1	extratrees	1000	5	80%	8 minutes

Table 2: Set of runs that compare the different accuracies and Kappa values achieved with hyperparameter tuning in Random Forest Model

According to the confusion matrix in Figure 2, 200 galaxies were predicted perfectly. 178 quasars were predicted and 25 of them were misclassified (20 stars, 5 galaxies). Finally, 179 stars were predicted and only 21 of them were misclassified (20 as quasars and 1 as a galaxy).

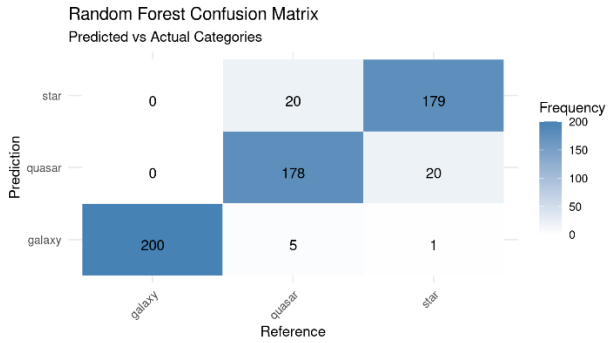

Figure 2: Confusion Matrix for Predicted vs. Actual Celestial Objects for the Random Forest model

3.2 XGBoost Results

The accuracy of the XGBoost results was comparable to that of the Random Forest model, achieving an overall accuracy of 91.54%, a 2% average improvement from Tarsitano's model [2]. Galaxies were predicted with a 99% accuracy, quasars 91%, and stars 91%, which was also on par with the Random Forest. In terms of precision, recall, and F1-score, the XGBoost model performed as well as the Random Forest or worse in some categories, as can be seen in Table 3.

XGBoost Algorithm			
	Precision	**Recall**	**F1-score**
Galaxy	97%	100%	98%
Quasar	87%	89%	88%
Star	91%	86%	88%
Accuracy			91.54%

Table 3: Classification report for Galaxy, Quasar, and Star accuracy using the XGBoost Algorithm

The major difference between the best performing Random Forest model and the best performing XGBoost model is that it took much longer to run at 1.4 hours. The longest run took 2 hours even with a 50% reduction in data— see Table 4 for more details on each run.

XGBoost								
Run #	**Accuracy**	**Kappa**	**ETA**	**max_depth**	**nrounds**	**cv folds**	**Training Size**	**Runtime**
1	83%	0.75	0.2	6	100	5	10%	2 hours
2	84%	0.77	0.3	6	100	5	10%	10 minutes
3	84%	0.76	0.4	6	100	5	10%	15 minutes
4	86%	0.79	0.3	3	100	5	10%	16 minutes
5	86%	0.79	0.1	3	150	5	20%	58 minutes
6	90%	0.85	0.1	3	100	10	40%	2 hours
7	91%	0.87	0.1	3	100	10	80%	1.5 hours

Table 4: Set of runs that compare the different accuracies and Kappa values achieved with hyperparameter tuning in XGBoost Algorithm

The number of galaxies that were predicted matched that of the Random Forest model, as seen in Figure 3. Quasars were classified slightly better, with only 17 being misclassified as stars. However, only 171 stars were categorized correctly, this time 27 of them were classified incorrectly as quasars and 2 as galaxies.

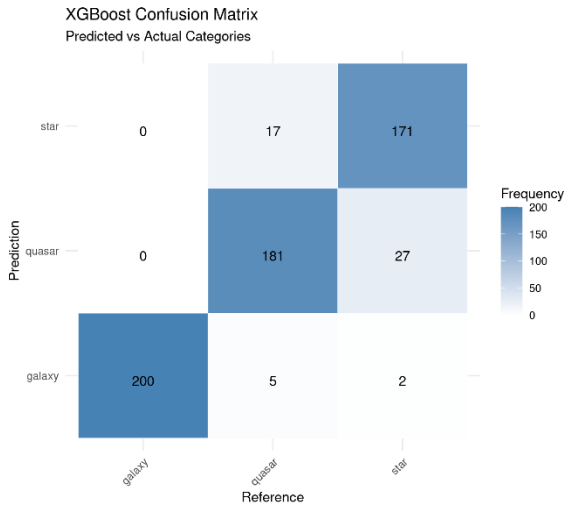

Figure 3: Confusion Matrix for Predicted vs. Actual Celestial Objects for the XGBoost model

3.3 CNN Results

The last algorithm created and tested was a Convolutional Neural Network, which was the most promising because of its ability to handle large amounts of data. Despite this, it performed the worst out of all three algorithms. There were several models tested, each with different hyperparameters and layers. In an effort to find the best accuracy beyond custom models, we used Bayesian Hyperparameter Optimization, but the highest testing accuracy was 79%. Additionally, we tested some pre-trained models like ResNet, but that one performed even worse with a 71.53% accuracy and 3-hour runtime.

The best run from our custom model took 3.5 minutes and achieved 85% accuracy using a training size of 80% for all runs. This was on par with Hui's pre-trained models that obtained accuracies ranging from 83% to 89% [4]. We utilized the Adam optimizer, 20 epochs, a learning rate of 0.001 and a batch size of 64. Interestingly, increasing the number of epochs did not necessarily improve accuracy, and in fact led to overfitting of the data. See Table 5 for more details on hyperparameter optimization.

CNN

Run #	Accuracy	Loss	layers	lr	epochs	batch_size	kernel_size	Runtime
1	81%	0.91	5	0.1	30	32	(3,3)	3 minutes
2	82%	0.96	5	0.1	30	32	(4, 4)	3 minutes
3	81.25%	0.96	5	0.001	30	32	(3,3)	5.5 minutes
4	82%	1.28	5	0.0001	30	32	(3,3)	9.5 minutes
5	80.38%	0.74	12	0.0001	30	32	(7,7)	1.5 hours
6	71.53%	2.8	ResNet	0.001	30	32		3 hours
7	85%	0.35	5	0.001	20	64	(3,3)	3.5 minutes

Table 5: Comparison of Different Runs for the CNN Algorithm

In Figure 4, we can see how well the best CNN model performed during the training and validation stages by evaluating Loss vs. Accuracy. Training accuracy reached an accuracy as high as 87%, while validation accuracy never rose above 84%.

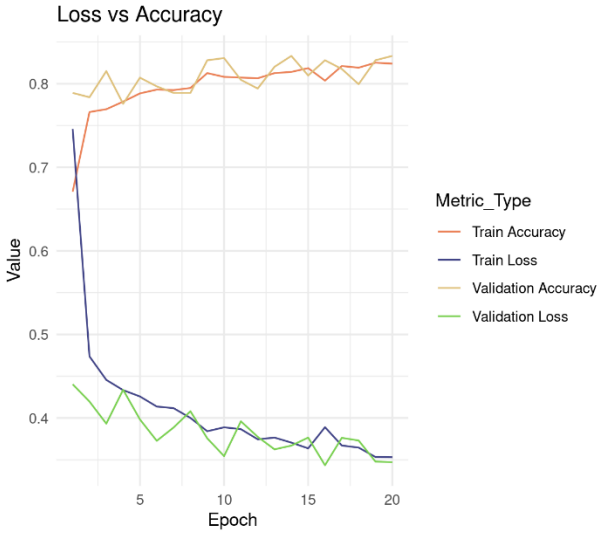

Figure 4: Loss vs. Accuracy for the training and validation stages of the CNN algorithm

In future experiments, additional testing will be done to further improve upon the accuracy of the CNN. While adding more layers to the model did not improve accuracy or runtime, we would like to explore other automatic processes for tuning hyperparameters and layers such as the Keras Tuner and H2O.

3.4 Overall Results

In comparing all three algorithms strictly based on accuracy, the Random Forest model performed the best by reaching a 92% testing accuracy in 8 minutes. The XGBoost model was not far behind, achieving a 91% testing accuracy, but it took 1.5 hours to obtain. Finally, the CNN algorithm scored the lowest accuracy at 85% but was able to achieve this in 3.5 minutes. The testing results for each model can be seen in Table 6. As with all algorithms, there are benefits and drawbacks to each.

Testing Results

Algorithm	Lowest Accuracy	Highest Accuracy	Runtime
Random Forest	80%	92%	8 minutes
XGBoost	70%	91%	1.5 hours
CNN	32%	85%	3.5 minutes

Table 6: Comparison of different testing results achieved by the three algorithms

4. Conclusion

As part of this research, various methods for Galaxy-Quasar-Star classification are compared. Over the years various scientists have proposed new methods for the classification of celestial objects which have aided in our search for extraterrestrial life, as well as the understanding of how the universe was formed.

Overall, the Random Forest algorithm performed the best with a 92% accuracy and relatively short runtime. This was surprising given the known computational intensity of Random Forest algorithms. It was the most promising improvement, as there was an 8% accuracy increase from previous methods. The XGBoost model took much longer but achieved similar results with 91% accuracy. Although the CNN required the least amount of time, it scored the lowest in accuracy at 85%.

While the three models have differing accuracy rates, the introduction of more images will likely increase the accuracy for all of them. However, since this is an image classification problem, the CNN model may end up being the best of the three since it is built to handle problems like this.

Acknowledgement

We would like to express our gratitude to our professor, Dr. Paul Johnson, who is guiding the process of this research project with respect to STAT 7240 Applied Data Mining.

References

[1] P. M. Plewa, "Random Forest Classification of Stars in the galactic centre," Monthly Notices of the Royal Astronomical Society, vol. 476, no. 3, pp. 3974–3980, 2018. doi:10.1093/mnras/sty511

[2] F. Tarsitano, C. Bruderer, K. Schawinski, and W. G. Hartley, "Image feature extraction and Galaxy Classification: A novel and efficient approach with automated machine learning," Monthly Notices of the Royal Astronomical Society, vol. 511, no. 3, pp. 3330–3338, 2022. doi:10.1093/mnras/stac233

[3] D. Shaiakhmetov, R. R. Mekuria, R. Isaev, and F. Unsal, "Morphological classification of galaxies using SpinalNet," 2021 16th International Conference on Electronics Computer and Computation (ICECCO), 2021. doi:10.1109/icecco53203.2021.9663784

[4] W. Hui, Z. Robert Jia, H. Li, and Z. Wang, "Galaxy morphology classification with DenseNet," Journal of Physics: Conference Series, vol. 2402, no. 1, p. 012009, 2022. doi:10.1088/1742-6596/2402/1/012009