G-11: Galaxy Image Classification: A comparison between ML and DL techniques

Ishan Bhatt (ivbhatt), Meghana Kota (mkota), Pragna Bollam (pbollam), Shilpa Kancharla (skancha)
Department of Computer Science, North Carolina State University,
Raleigh, NC, USA
e-mail: [ivbhatt, mkota, pbollam, skancha] @ncsu.edu

Code: https://github.ncsu.edu/ivbhatt/ALDA-G11-GalaxyClassification

1. Problem Statement

SDSS telescopes have captured over 40TB [7] worth of galaxy images and classification of these images is the first step towards obtaining a deeper understanding of physical processes within them, star formation, and the nature of the universe. Since we could not find an easily-accessible dataset for Galaxy-classification, we **compiled a dataset for Galaxy classification and provided benchmarks with some of the common learning-algorithms that would help in automating the Galaxy-classification which until recently had to be performed by-hand by expert astronomers** [4][5]. We are classifying the images of galaxies into four classes- spiral, elliptical, irregular, and invalid.

We **built a dataset for this domain**. We benchmarked the dataset using some of the most common learning (ML/DL) algorithms. To apply ML algorithms, we applied **PCA** on the generated dataset to reduce the dimensionality and extract relevant features. Once we engineered the features, we applied some of the major **ML techniques (SVM and Random Forest) and DL techniques (MLP and CNN)** directly using the images (without PCA).

2. Related works

[1] Nour Eldeen M. Khalifa, Mohamed Hamed N. Taha, Aboul Ella Hassanien , I. M. Selim; Deep Galaxy: Classification of Galaxies based on Deep Convolutional Neural Networks

For galaxy classification, a deep convolutional neural network architecture with 8 layers, and one main convolutional layer for features extraction with 96 filters, and two principal fully connected layers is applied. Based on the features, classification is done into three categories - spiral, elliptical, irregular.

[2] Siddhartha Kasivajhula, Naren Raghavan, Hemal Shah; Morphological Galaxy Classification Using Machine Learning

Set of Morphic features generated from Image analysis and direct image pixel data compressed through PCA are used to apply ML algorithms like Support Vector Machines (SVM), Random Forests (RF), and Naïve Bayes (NB) to classify the images. Finally, the performance of these algorithms on the data is compared on both morphic and PCA features.

[3] Jorge De La Calleja, Olac Fuentes; Machine learning and image analysis for morphological galaxy classification

For morphological galaxy classification, a neural network, a locally weighted regression method, and homogeneous ensembles of classifiers are used. Data is augmented and PCA is applied to reduce dimensionality and get relevant features. A 10 fold Cross Validation is applied on homogeneous ensemble regression methods to classify images into three categories - spiral, elliptical and irregular.

[4] Joseph H. Murrugarra LL., Nina S. T. Hirata; Galaxy image classification

For images from Sloan Digital Sky Survey, image cropping and preprocessing is done to get into labelled format. Then, a convolutional neural net is applied to extract features and given to a classifier model Support Vector Machine (SVM) to classify the images as Spiral or Elliptical. This model is given an accuracy of about 90-91%.

[5] Jorge de la Calleja, Olac Fuentes; Automated Classification of Galaxy Images

To conduct galaxy image classification, machine learning methods of Naive Bayes, random forests, and the rule-induction algorithm C4.5 are applied. After performing some preliminary image processing, PCA is also used to reduce dimensionality before the classification algorithms are applied. The results of this paper demonstrate that random forest achieves about a 91% accuracy when classifying images into three categories: elliptical, spiral, and irregular.

3. Approach

We applied **Principal Components Analysis (PCA) to our images in order to reduce the dimensionality of the data for ML approaches.** Since image data requires a lot of storage, we compress the images in order to preserve the most important features of the image data. **We then apply ML classification techniques to the compressed data**.

The first classification technique applied is **Support Vector Machines (SVM)**. This classifies the images into four classes by creating boundary hyperplanes between classes. It is more robust i.e. due to optimal margin gap between separating hyper-planes, it could do predictions better with test data. **SVMs are simple and efficient, and less likely to overfit.**

Moreover, we apply the **Random Forests (RF)** algorithm. We understand that random forests are bagged decision trees that are split on a random subset of features. Due to the fact that this technique **splits on a random subset of features, it reduces the variance and is robust to outliers.** Therefore, we have the potential to produce a robust galaxy classification model.

Furthermore, a **Multi Layer Perceptron (MLP)** is applied on the dataset generated. These MLPs use one perceptron for each input (e.g., pixel in an image) and classify an unknown pattern with other known patterns that share the same distinguishing features. Because of this, noisy or incomplete inputs are also classified as they are similar with pure and complete inputs.

For image classification, **Convolutional Neural Networks (CNNs) have been known to achieve better performance than standard MLPs**. Therefore, it makes sense for us to also provide a bench-mark using CNNs. Due to the sliding-window nature of the convolutional windows, spatial patterns are detected more efficiently by CNNs. Hence, these are better suited for image-related data.

4. Rationale

We use **PCA for feature selection**, as our data set is big, extracting more relevant features is helpful in efficient use of computational resources. **ML approaches we are using cannot handle datasets with very high dimensionality, therefore we need a way to reduce the dimensionality** by selecting the best features. PCA skips less significant components.

Another approach we could consider for dimensionality reduction was **t-distributed stochastic neighbor embedding (t-SNE)**, which requires hyperparameter tuning and relies on a probabilistic distribution. We opted to use **PCA because it is a more straightforward and reliable approach.**

The **SVM approach can be used for relatively high-dimensional data, such as image data**. SVM clearly classifies the data based on kernel equations. An alternative approach to SVM we considered was **logistic regression**. However, **logistic regression would not provide us with clear margins that separate data,** but rather just probabilistic boundaries.

Random Forest algorithm considers random selection of features at each node to determine the split. We decided to use random forests instead of **decision trees** because a **random forest is more robust than a single decision tree.** Compared to a single decision tree, random forests reduce bias by aggregating multiple decision trees and can produce more accurate results.

An **MLP** has the potential to detect patterns that may be too complex to be picked up by simpler ML techniques like Random Forests, etc. We believed a **simple neural network could detect less obvious patterns and features to improve classification accuracy**. We also note that MLPs do not consider spatial data, so we also apply the deep learning technique such as CNNs to see if we can get better results.

CNNs are not fully connected feed-forward networks. They replace the full-connections found in an MLP with a number of convolutional filters at each layer. This enables **CNNs to reduce the number of parameters without compromising the quality of the model.** Therefore, we believe that CNNs would be ideal for our classification problem as image data has high dimensionality. We need not perform PCA on the images first, but can directly process and feed image data into the CNN after Z-score normalization.

5. Dataset

We choose to use the full catalog of the Galaxy Zoo 1 dataset as our input. This dataset provides us with an object ID, coordinates to where a celestial object is, and a one-hot encoding of the category of the galaxy (*Elliptical, Spiral*) as well as an attribute that reflects the quality of the classification. The Sloan Digital Sky Survey provides an API from which we can fetch images of galaxies given their coordinates. Out of the 600K+ images in the original dataset, we picked only about 10K+ high-quality data points. The API

provides functionality to specify size of the image. **We used this API to get a 1000 images each of *Elliptical* and *Spiral* galaxies.** Based on initial analysis, we believed that 512x512 is a reasonable input size. However, at the time of applying PCA to this dataset, **we had to reduce the dimensions further to 128x128** as the RAM was not sufficient while running PCA on the dataset.

Additionally, we **web-scraped about 200 images of *Irregular* galaxies;** removed duplicate (faulty) images and spiral or elliptical images from them manually. Later, **data augmentation is applied on them to get about 1000 images of *Irregular* category. We also scraped 828 non-celestial object images and added them to our dataset and labelled them as *Invalid*.** Finally our generated dataset consists of **991 elliptical, 1001 spiral, 1000 irregular galaxies and 828 invalid images adding up to a total of 3820 images**. Each image is to be classified in one of the following four categories: *Elliptical, Spiral, Irregular,* and *Invalid.*

6. Hypotheses

Hypothesis 0.1: Even after implementing Z-score normalization, we expect the total-variance in the dataset to be equal to 1, in reality, **there should always be a small difference between the variance expected and the variance obtained.**

Hypothesis 0.2: Even though it is recommended, it is not required by ML algorithms that all classes are represented exactly equally in the dataset. **We believe the learning techniques should still be able to achieve respectable classification accuracy with slightly imbalanced datasets.**

Hypothesis 1: If about 95% variance of the original dataset is preserved in the Principal Components obtained after PCA, then **we should be able to reconstruct the original images using PCs.**

Hypothesis 2: For this particular use case, we predict Random Forest will outperform SVM. Simply because we think the patterns here are relatively simple but the ensemble nature of Random Forests will help it to make robust predictions.

Hypothesis 3: The MLP model should result in higher accuracies compared to our SVM and random forest models.

Hypothesis 4: The CNN should result in the highest accuracy compared to all machine learning models and the MLP model.

7. Experimental design

Using the API provided by the Sloan Digital Sky Survey, we scraped 2000 images (100 each of elliptical and spiral galaxies).

We scraped about 800 images of non-celestial objects (class:invalid) to help our models learn more robust features & learn to differentiate between galaxies and other objects.

Finding images of irregular galaxies was a bit of a challenge. We managed to scrape about 200 images of unique irregular galaxies and then used data-augmentation to generate 1000 images from them.

For data augmentation (irregular galaxies ONLY), we took web-scraped images and removed duplicate images manually. Then, we applied the following transformations:

- ChannelShuffle (RGB to any other order) with a probability of 0.35
- Zoom-in; Zoom-out upto 10% on all images.
- Horizontal Flip or Vertical Flip with a probability of 0.5
- Rotation by 90, 180, 270 or 360 degrees with a probability of 0.5

We generated exactly 1000 images of irregular galaxies using the above transformations. Figure 1.1 illustrates how we generate eight seemingly unique images from just one image!

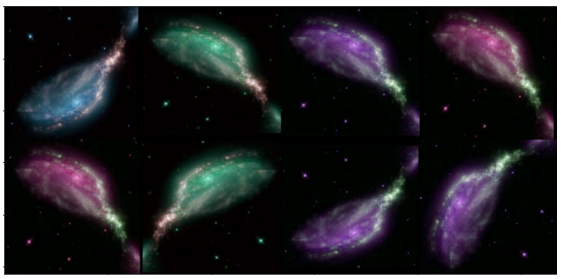

Figure 1.1: Data-augmentation (Eight images out of one!)

After augmentation, we resize images to 128x128 and combine them together. Our final dataset consists of:

- 1001 *Spiral* images
- 1000 *Irregular* images
- 991 *Elliptical* images
- 828 *Invalid* images

On these images, PCA was applied to reduce dimensionality. For PCA on images, the following transformations are applied:

- Flatten Images
- Z-Score Normalization

After the above transformations, each image is a vector of size 128*128*3 = 49152 and so after Z-score normalization we expect the total variance to be equal to 49152. However, we find that observed total variance is about 49116 which is slightly less than the theoretical expectation. Hence, **hypothesis 0.1** is confirmed.

After PCA, we would like to preserve at least about 95% of the variance present in the images. We would like to use the least number of PCs to do so. We start with 1 and go on doubling the number of PCs until we reach a point where 95% of the variance is preserved.
It is observed that for number_PCs = 256, a variance of 46784.8 is preserved which is about 95% of variance of the total dataset. And so, we will use 256 PCs to reduce the dimension. The below plot Figure 1.2 gives the variance explained by principal components on the dataset.

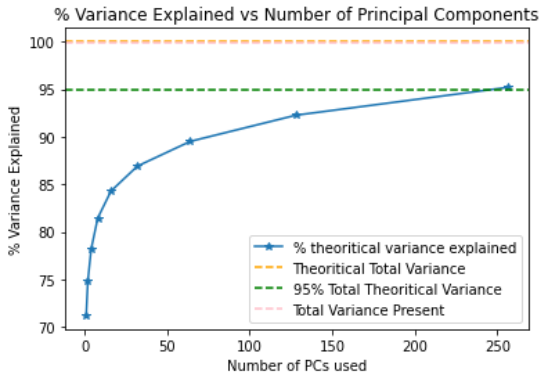

Figure 1.2: Percent of variances explained by number of Principal Components

Now, we reconstructed the images on the dataset using the PCs (as a sanity check) and it is observed that even after reducing the dimensions, the images preserved important features. This provides evidence for **Hypothesis 1**, thus showing that we are able to reconstruct the image to some degree using the PCs.
After applying PCA on the dataset, we plan to split our data into training and validation sets in order to create a baseline model. We use 70% of the data for training and 30% for validation.

Figure 1.3: Sample Images before applying PCA (high dimension)

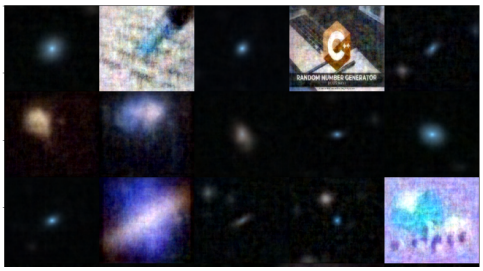

Figure 1.4: Same images after applying PCA (reduced dimension)

For training on dataset, both SVM and Random Forest Classifier are used. We compared the performances of different kernels (like rbf, poly, sigmoid) in SVM and we observed that a **linear kernel** gave higher accuracy for the images classification. For the Random Forest classifier, the **Gini** index is taken as a split criterion and the number of random trees to be generated are taken as **100**. For testing, the predictions of both SVM classifier and Random Forest Classifier are given. We are generating the classification reports with **precision, recall, F1-Score, and accuracy** for both SVM and Random Forest predictions. Also, an ROC curve is plotted to compare the **True Positive rate** and the **False Positive rate**.

We trained a Multi-Layer Perceptron on using the images dataset after Z-score normalization. We built our model using Keras (TensorFlow backend). Following hyperparameter choices were made after comparing runs over multiple iterations of grid-search.

- Adaptive momentum optimized the model much faster than RMSProp and Stochastic Gradient Descent with default learning_rate = 0.001
- Dropout rate of 0.25 was found to give the best test-performance out of 0.3 and 0.2
- Best-performing model had about 25K trainable parameters and was 2 layers deep with Rectified Linear activation.

Training is done with a batch size of 64 and a plot for training and validation learning curves is given in below Figure 1.5.

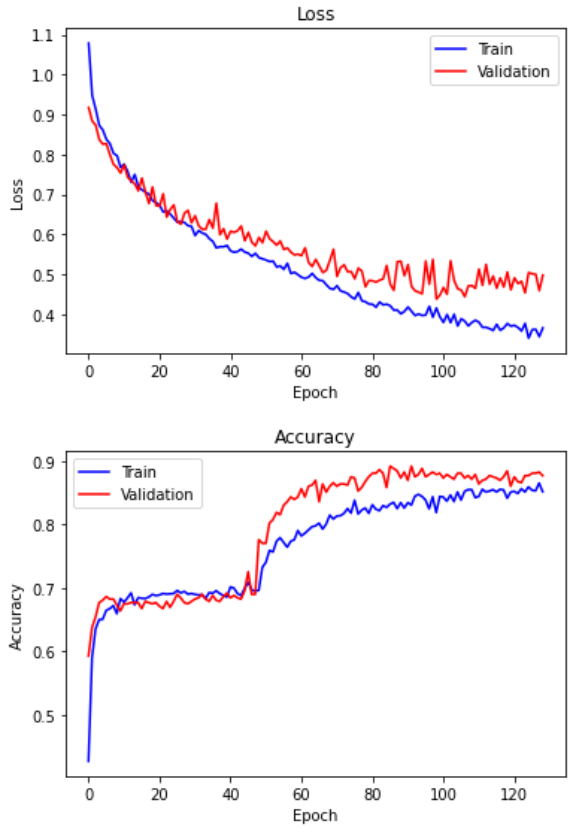

Figure 1.5: Loss and Accuracy plots generated for training and validation in MLP

We also train a Convolutional Neural Network (CNN) on the image data. In order to optimize the model performance we used hyper-parameter grid-search methods on the generated dataset. We found:

- Adaptive momentum optimized the model much faster than RMSProp and Stochastic Gradient Descent with default learning_rate = 0.001
- Dropout rate of 0.2 was found to give the best test-performance out of 0.25 and 0.3
- Best-performing model had about 17K trainable parameters and was 4 layers deep with Rectified Linear activation.

Figure 1.6 shows the training and validation learning curves of accuracy and loss for our CNN.

8. Results

For our experimental run with a linear SVM, we achieved 86.13% accuracy, with the precision, recall, and F1-scores for this model displayed in Table 1.1. Moreover, using the **random forest method, we achieved 92.06% accuracy** with the aforementioned metrics for it displayed in Table 1.1 as well. We compare this to the result found in the experiment done by Calleja and Fuentes who also performed PCA and then random forest, which received an accuracy of 91.64% [5]. The difference between our experiment and the experiment done by Calleja and Fuentes is that we include a fourth class of classification, the invalid category. Calleja and Fuentes use only three classes: elliptical, spiral, and irregular. Regardless of this difference, we found that our random forest classifier was able to distinguish invalid images as well while achieving a slightly higher accuracy. We found that the accuracy of our random forest run was higher than that of our linear SVM, thus corroborating our **Hypothesis 2**.

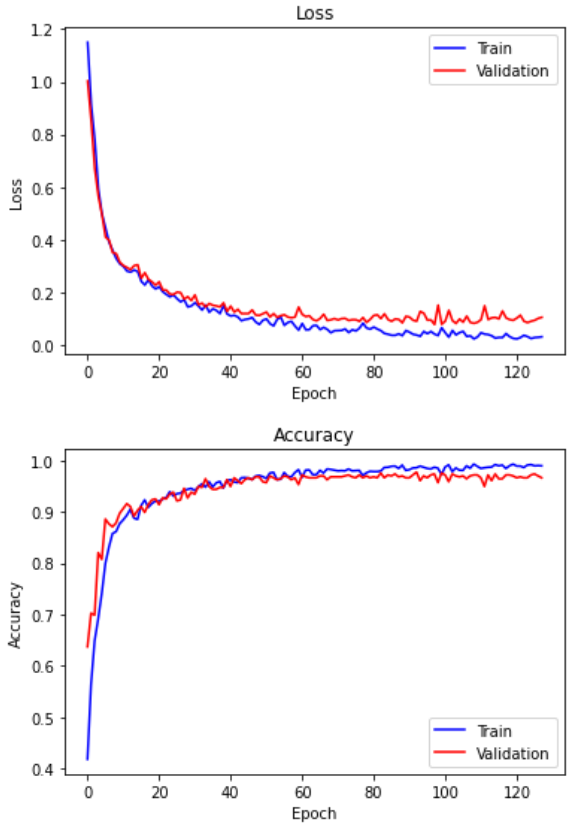

Figure 1.6: Loss and Accuracy plots generated for training and validation in Convolutional Neural Nets

The below Figure 1.9 shows the predictions of each image along with the actual class label. Table 1.1 displays the classification reports with precision, recall and F1-score for SVM classifier and Random Forest Classifier.

In Figure 1.7 and Figure 1.8, we see the ROC curve for the linear SVM and random forest classifiers, respectively. The ROC curves further evince that the random forest classifier achieves better performance

than the linear SVM. We can see that in Figure 1.8, the true positive rate is very close to 1.0 while the false positive rate is very close to 0.0 for classes 2 and 3 compared to the curves in Figure 1.7 for classes 2 and 3. We can also see that the curves for classes 0 and 1 are closer to the (0, 1) point of the curve in Figure 1.8 than in Figure 1.7.

Table 1.1: Classification Report of Linear SVM and Random Forest
P - Precision, R- Recall, F1 - F1 Score, WA - Weighted Average

	Linear SVM			**Random Forest**		
Class	P	R	F1	P	R	F1
0	0.79	0.88	0.83	0.89	0.90	0.90
1	0.86	0.81	0.84	0.92	0.85	0.89
2	0.86	0.94	0.90	0.91	0.97	0.94
3	0.97	0.80	0.88	0.96	0.97	0.96
WA	0.85	0.85	**0.86**	0.92	0.92	**0.92**

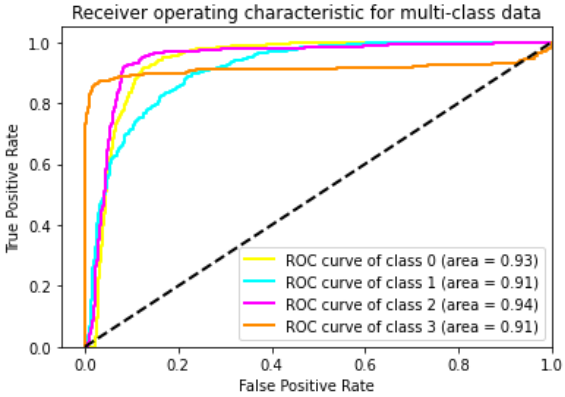

Figure 1.7: ROC curve for the linear SVM

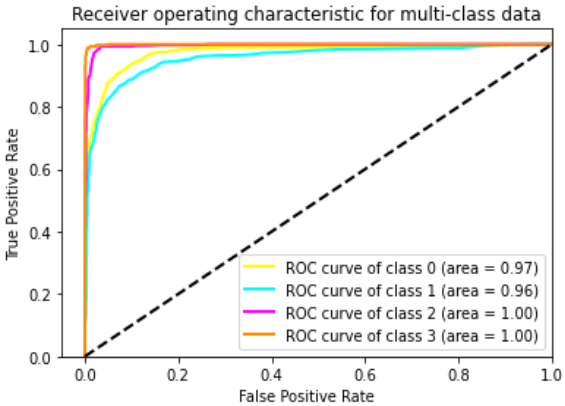

Figure 1.8: ROC curve for the Random Forest Classifier

Our MLP model achieved classification accuracy of about 87%. We have shown the loss and accuracy plots for MLP in Figure 1.5. We also provide the classification report for our MLP model in Table 1.2. An interesting feature of our accuracy plot in Figure 1.5 is that the validation accuracy ends up being greater than the training accuracy as model training continues. However, it is still within the margin of error and is a weak indication of generalizable performance. In Figure 1.9, we show a sample of images with their actual and predicted class that resulted from the MLP model.

Table 1.2: Classification Report of MLP and CNN
P - Precision, R- Recall, F1 - F1 Score, WA - Weighted Average

	MLP			**CNN**		
Class	P	R	F1	P	R	F1
0	0.80	0.85	0.82	0.95	0.98	0.96
1	0.86	0.88	0.82	0.97	0.93	0.95
2	0.93	0.89	0.91	0.98	0.97	0.97
3	0.88	0.96	0.92	0.97	0.98	0.98
WA	0.87	0.90	**0.87**	0.97	0.97	**0.97**

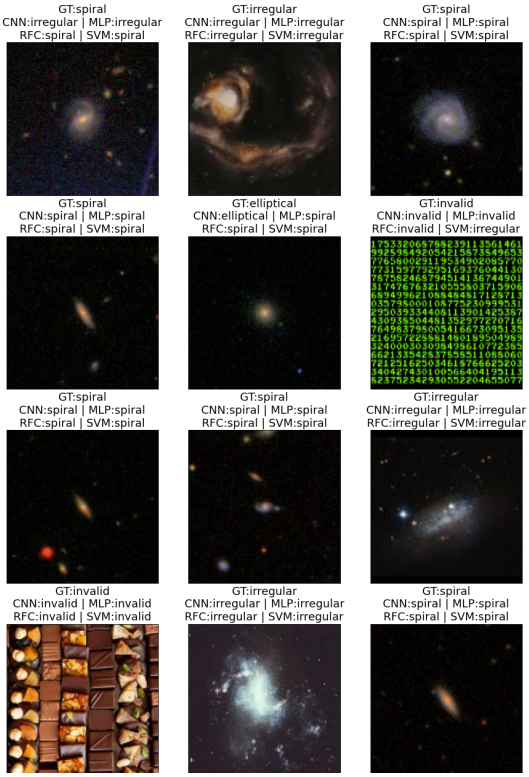

Figure 1.9 Sample images with their actual and predicted class for RFC, SVM,MLP and CNN model

Our CNN model performed at an accuracy of 97%. We have shown the loss and accuracy plots above in Figure 1.6. We also provide the classification report for our CNN model in Table 1.2. It is observed that as the hidden layers in the model are increased, the accuracy of the model increases. In Figure 1.9, we show a sample of images with their actual and predicted class that resulted from the CNN model.

9. Discussions

In relation to **Hypothesis 0.2**, we found that even though the distribution of our data was not equal, we were still able to achieve accuracy above 80% for each model.

The first algorithm we have used is **PCA**, which reduces the dimensionality of the dataset while preserving the most important features of the image data. We were able to reconstruct images from PCs (that retained about 95% variance from the original data). **This confirms Hypothesis 1**, while also **validating Hypothesis 0.1** in that we retained 95% of the variance, rather than 100% of it.

As shown in Table 1.1, the classification metrics for the random forest results are notably higher than their SVM counterpart values. **This is consistent with our Hypothesis 2**, as we predict that random forests would outperform the SVM. Perhaps this is due to random forests not being as sensitive to outliers because of the pooling nature of the algorithm.

The MLP can detect patterns that are complex for other algorithms to detect. If we examine the accuracies[1] of the linear SVM (86.13%), random forest (92.06%), and MLP (87%), we see that **Hypothesis 3 is not entirely supported.** The accuracy of random forest is higher than that of MLP. Moreover, while comparing Table 1.1 and 1.2, it is seen that most of the classification metrics for random forest are larger than those of the MLP. Since this use case requires robust detection of rather simple features, perhaps the **MLP's ability to detect complex features has gone under utilized**.

As shown from Table 1.2, the classification metrics for CNN are consistently higher than or equal to those of all the other classification algorithms implemented. Therefore, Hypothesis 4 is supported. **Hypothesis 4 is valid because the CNN's accuracy is best among the algorithms we compared.**

In Figure 1.9, some interesting results are presented:

- *Col:1; row:1 - CNN and MLP misclassify a spiral galaxy as an irregular galaxy: We notice how there are multiple smaller galaxies scattered around the main spiral galaxy which gives an appearance similar to a cluster-galaxy.*
- *col:3; row:2 - SVM misclassifies an invalid image as an irregular galaxy: We note that this is a particularly hard image since a lot of bright-green patches are present in a black-background giving an appearance similar to a cluster-galaxy.*
- *In all other cases, we notice how all the models are mostly in agreement with each other and the ground truth labels*

10. Conclusions

Through our experiments, we present various machine learning models as well as deep learning models that can be used for automating the identification of galaxies based on image data. From our results, we see that the CNN model performed the best on our dataset of elliptical, spiral, irregular, and invalid images as it achieved classification accuracy of 97%. While our CNN model may have performed the best, it is also possible to consider using ensemble methods from the machine learning classifiers to further achieve better accuracy. We believe this type of image detection could also be extended to other astronomical data, such as nebulae, star clusters, or images of astronomical data that contain multiple objects; therefore, the models we have created have other practical algorithms in astronomy.

Finally, we conclude by presenting a new benchmark for Galaxy-Image classification as well as a best-performing model (our CNN) that achieves about 97% classification accuracy which beats previous attempts [4][5] by at least 5%

We learnt how to perform the following ML tasks:

- Web-scraping images using SDSS APIs [12] and astropy [13]
- Image data-augmentation with imgaug [11]
- Applying PCA on images using sklearn [8]
- Random Forests, SVMs using sklearn [8]
- MLP and CNNs using keras [9]
- Generated classification report and ROC curves using matplotlib and sklearn [8]
- Plotting results with matplotlib [10]

[1] Since our dataset is almost perfectly balanced; we have used the terms Accuracy and F1-score somewhat interchangeably

11. Online Meetings

- Meeting 1 on 03/15/2021 for 45 minutes - Discussed about project ideas to implement.
- Meeting 2 on 03/20/2021 for one hour - Focused on this topic and how to implement this project, which algorithms to implement.
- Meeting 3 on 03/22/2021 for one hour - To complete the project proposal document.
- Meeting 4 on 03/28/2021 for one hour - Assigned tasks for each member and briefly talked about the procedures to follow.
- Meeting 5 on 04/05/2021 for 2 hours - Discussed problems concerned with deployment and resolved those issues.
- Meeting 6 on 04/07/2021 for 2 hours - To complete Midway report.
- Meeting 7 on 04/09/2021 for 2 hours - Discussing MLP implementation
- Meeting 8 on 04/16/2021 for 2 hours - Discussing CNN model implementation.
- Meeting 9 on 04/23/2021 for 2 hours - Discussing optimization of CNN model.
- Meeting 10 on 04/29/2021 for 3 hours - Finalizing all submission software and the report.

All the meetings are attended by everyone in our team.

References

[1] Nour Eldeen M. Khalifa, Mohamed Hamed N. Taha, Aboul Ella Hassanien , I. M. Selim; Deep Galaxy: Classification of Galaxies based on Deep Convolutional Neural Networks

[2] Siddhartha Kasivajhula, Naren Raghavan, Hemal Shah; Morphological Galaxy Classification Using Machine Learning

[3] Jorge De La Calleja, Olac Fuentes; Machine learning and image analysis for morphological galaxy classification

[4] Joseph H. Murrugarra LL., Nina S. T. Hirata; Galaxy image classification

[5] Jorge de la Calleja, Olac Fuentes; Automated Classification of Galaxy Images

[6] Chris J. Lintott, Kevin Schawinski, Anze Slosar, Kate Land, Steven Bamford, Daniel Thomas, M. Jordan Raddick, Robert C. Nichol, Alex Szalay, Dan Andreescu, Phil Murray, Jan van den Berg; Galaxy Zoo: Morphologies derived from visual inspection of galaxies from the Sloan Digital Sky Survey

[7] Yanxia Zhang , Yongheng Zhao; Astronomy in the Big Data Era

[8] Pedregosa, F., Varoquaux, Gael, Gramfort, A., Michel, V., Thirion, B., Grisel, O., … others. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12(Oct), 2825–2830.

[9] Chollet, F., & others. (2015). Keras. GitHub. Retrieved from https://github.com/fchollet/keras

[10] Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. Computing in Science & Engineering, 9(3), 90–95.

[11] Alexander B. Jung. (2018). imgaug. https://github.com/aleju/imgaug.

[12] Hector Robles, Hugo Soto. (2019). Identifying Galaxies Using the Deep Learning Reference Stack. https://01.org/blogs/2019/identify-galaxies-using-deep-learning-reference-stack

[13] Price-Whelan A., et al. The Astropy Project: Building an open-science project and status of the v2. 0 core package. The Astronomical Journal. 2018;156(3):123.