International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 07 Issue: 08 | Aug 2020 www.irjet.net p-ISSN: 2395-0072

Galaxy Morphology Classification using Deep Learning

Srujan G[1], Rohan R[1], Vinay B S[1], Manasa KB[2]

[1]UG Student, Dept. of Information Science and Engineering, NIE Institute of Technology, Mysuru, Karnataka, India.
[2]Assistant Professor, Dept. of Information Science and Engineering, NIE Institute of Technology, Mysuru, Karnataka, India.

---***---

Abstract - *With numerous digital sky surveys across a wide range of instruments and wavelengths becoming available over the past decade, the field of astronomy has become incredibly data rich. It has also increased the manpower and time required to analyze and process large scale data processing tasks like galaxy classification. We present a deep learning approach towards solving this computation problem, utilizing a convolutional neural network trained with the help of GPU accelerated platforms such as CUDA and made robust using techniques such as data augmentation and hyperparameter tuning.)*

Key Words*: *Deep Learning, Convolutional Neural Network, Machine Learning, CUDA, Astronomy, Image Processing.

1. INTRODUCTION

Human beings have been observing and recording celestial objects for millennia, from the era of the Stonehenge to the invention of the telescope, and more recently, the Hubble Space Telescope, broadening our understanding of the observable universe. In recent years, the number of telescopes, observatories, and instruments that operate on almost the entire electromagnetic spectrum has rapidly increased. Projects such as the Sloan Digital Sky Survey (SDSS) aim to map vast regions of the sky, while other such as the Event Horizon Telescope (EHT) attempt to image little-known objects such as black holes in the far reaches of the universe.

Some of the largest objects in the known universe are galaxies, enormous collections of stars, remnants of dead stars, interstellar gas and dust, and mysterious dark matter, all bound together gravitationally. The classification system for galaxies was put forth by renowned astronomer Edwin Hubble and it was called the Hubble Sequence. It classified galaxies into groups based on their visual appearance. They were generally divided into groups such as spiral and elliptical.

As the number of astronomical observations have increased exponentially, the manpower required to accurately process them has similarly increased at a commensurate rate. Data is being collected at a rate much faster than that can be analyzed manually. Not only do these tasks require a lot of skill and patience, they are also time consuming. The SDSS alone is expected to churn out 50 million images of galaxies in the near future. The Hubble Space Telescope (HST) collects an estimate 150 GB of data each week, the EHT is estimated to collect 350 TB of data per day. The image of black hole M87* alone was as large as 5 petabytes.

As a result, the field of astronomy requires solutions to help offload such tasks onto large computing systems. Such challenges have found solutions in the field of deep learning. In particular, the task of analyzing images of galaxies and classifying them into their respective groups based on their visual morphology can be done with the help of convolutional neural networks (CNN).

1.1 Convolutional Neural Networks

Artificial neural networks are deep learning computing systems that can extract and learn features or patterns from sensory data. Convolutional neural networks are specialized artificial neural networks that use convolution kernel filters in order to extract features and analyze visual imagery.

Convolution filters work by making use of convolution operations, which involve adding each element of an image to its local neighbours, weighted by the kernel. They're used extensively for common image processing tasks like blurring, sharpening, edge detections, etc. It is their ability to efficiently extract feature maps that make them adept at image recognition tasks. The stacking of convolution layers allows for a hierarchical decomposition of the input.

2. RELATED WORKS

Various research studies have been conducted on the viability of automating galaxy classification. Calleja et. al. presented an experimental study for using machine learning and image analysis in order to classify galaxies. A neural network with a locally weighted regression method, and homogenous ensembles of classifiers were used. They used the bagging ensemble method for the neural networks and they manipulated input features to create the ensemble of locally weighted regression. The galaxies were transformed by rotating, and center-cropping, in a fully automatic manner. Furthermore, Principal Component Analysis (PCA) was used for dimensionality reduction, and to extract relevant information from the image data. The preliminary experimental results were evaluated with a ten-fold cross-validation technique, and it showed that the homogenous ensemble of locally weighted regression produces the best results, with 91 percent accuracy when considering three types of galaxies (Elliptical, Spiral, and Irregular), and 95 percent when considering two types (Elliptical and Spiral).

Gauci et. al. made use of decision tree learning algorithms and fuzzy inferencing systems for this classification task. In order to distinguish between spiral, elliptical, or star/unknown bodies, the CART, C4.5, random forest, and fuzzy inferencing systems were developed. The morphology information was downloaded from the Galaxy Zoo project for the training and testing datasets, while the corresponding photometric and spectral parameters were sourced from Data Release 7 of SDSS. Accuracies for different galaxy morphologies were developed through various machine learning approaches and were analyzed. After comparing the results from the CART, C4.5, Random Forest, and Fuzzy Inferencing systems, it was observed that Random Forest gave the highest accuracies in all cases when 50 trees were used.

Chou et. al. set out to build an algorithm that can extract indicators for galaxy morphologies. A pipeline was developed which combined multiple computer vision feature detectors and ML regression. The performance was experimented using the cross-validation technique. There are 3 sections in the pipeline: feature extraction, machine learning regression, and probability normalization. Multiple techniques were used for image analysis, like PCA, SIFT, Hog, Fourier transforms, etc. Their neural network is pre-trained, called Overfeat, which was trained on the ImageNet dataset. An ensemble of classification and regression classifiers such as least-square linear regression, ridge regression, and random forests is also used. They tested if introducing non-linearity in the regression classifiers by expanding the feature vector with quadratic features showed any improvement in the performance. A combination of all the feature extraction methods along with the quadratic features in the ridge classifier turned out to be their best algorithm, with root mean squared error of 0.113.

Gonzalez et. al. presented a method to automatically detect and classify galaxies which include a novel augmentation procedure to make trained models more robust against the data taken from different instruments and contrast-stretching functions. Training of the deep learning models was done using the public data such as the SDSS and Galaxy Zoo dataset, and private ones such as the Next Generation Virgo (NGVS) and Fornax (NGFS) surveys. Training were strongly bound to the conversion method from raw FITS data into a 3 channel RGB image. Therefore, a proposal for using 5 conversion methods in data augmentation. This resulted in great improvement in the overall detection of galaxies from different instruments, data reduction procedure, and bands. The deep learning framework DARKNET and YOLO real-time object detection system were used to train the detection and classification methods. They were implemented in C and the CUDA platform, making extensive use of graphical processing units (GPU), which could process an SDSS image in 50 ms, or a DECam image in around 3 seconds.

3. SYSTEM DESIGN

Machine learning models are only as good as the dataset they are trained on. Model training accuracy increases with higher quality datasets. The dataset used for the purposes of this project is the Galaxy Zoo dataset, which contains 61578 images of galaxies sourced from the SDSS. These images are to be split into training and validation datasets for the purposed of training the model. While the number of images in the training dataset is already quite high, the distribution of images between the types of galaxies could possibly be unequal. Therefore, there is a need to artificially increase the size of the dataset in order to have more instances of the same galaxy type to increase their chances of classification by the model.

Data augmentation is the process of artificially increasing the size of a dataset without collecting any new data. One of the reasons that this works is that CNNs are rotation invariant, which means that they can robustly classify images even if they are placed in different orientations. Similarly, images are also rotation invariant, since there is no up or down in space. They are also scale invariant and translation invariant to a small degree. All these invariances can be exploited to perform data augmentation. As a result, the increased quality and size of the training dataset helps us combat overfitting in the machine learning model. Overfitting is a type of modelling error wherein the machine learning model learns to classify data that it has already seen before but fails to accurately classify unseen data. That is, it is said to have memorized the training data and cannot replicate that performance with new test data. In other words, the model fails to generalize well with new data.

Data augmentation can be done in the following ways:

- Random zoom
- Random rotation
- Vertical flip
- Horizontal flip
- Random contrast shift
- Random brightness shift
- Rescale

The Galaxy Zoo dataset comes with a CSV file that maps the 61578 images to 37 labels from the Galaxy Zoo decision tree. Of these 37 labels, 6 are of prominent galaxy shapes; elliptical, cigar-shaped, in-between, barred-spiral, spiral, and on-edge. Therefore, these are the 6 different classes our model will classify the images of the galaxies into. Rather than using the CSV file to look up the label of the training images, it is prudent to use a more efficient solution for segregating the training images. TensorFlow provides an API that can read images from separate directories that indicate which label they belong to. Hence, while augmenting the dataset, we also segregate the labelled training images into their own directories which are named according to which class they belong to.

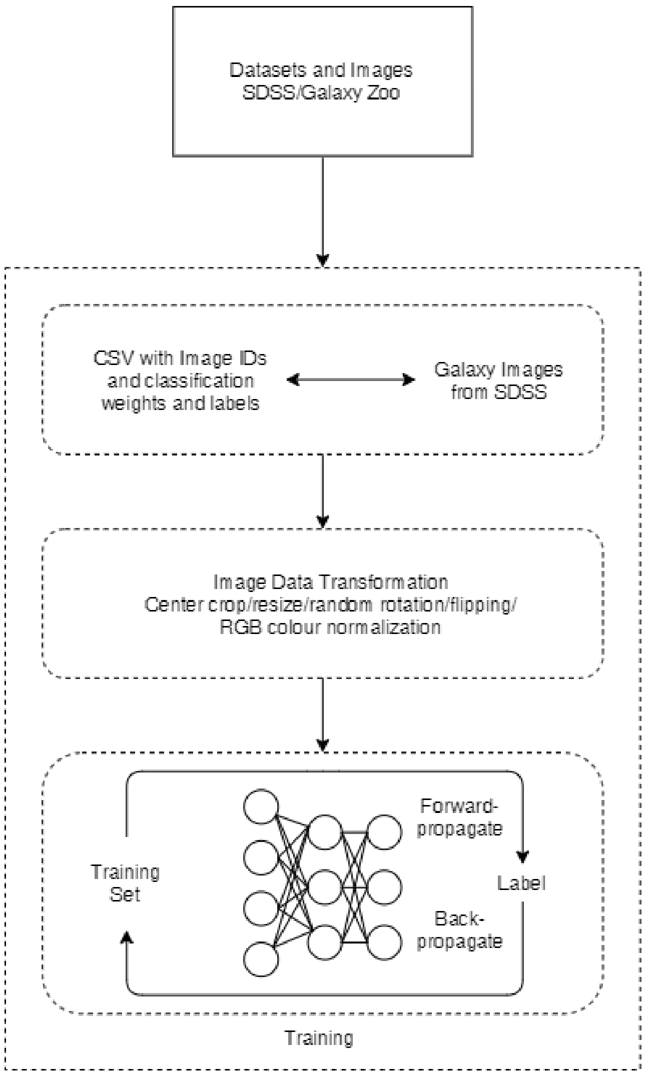

Fig -1: Data Pipeline Architecture

4. METHODOLOGY AND IMPLEMENTATION

Initially, the we took up the real-time approach to data augmentation using TensorFlow's built-in APIs, but the image transformation tasks proved too resource intensive and took up a lot of training time. Therefore, we switched to offline augmentation with the help of a Python library called Augmentor. This also allowed us to segregate the images into their respective directories based on the class they belong to. From the entire Galaxy Zoo dataset, the correctly labelled images corresponding to the 6 types of galaxies came up to 31468 images.

Some transformations were to be performed randomly on some images, and others were performed on all of them, such as resize and zoom. Galaxy Zoo had 424 x 424 pixel colour JPEG images, which we cropped with the zoom operation in to 265 x 265 pixel images, since the void of space around the image of the galaxy wasn't going to influence the performance of the model. Then we also down sampled the images 3x to keep the input size of the network manageable. We performed the following image transformation operations using Augmentor in order to artificially increase the size of the dataset:

- Zoom (center-crop) by a factor of 1.6
- Resize to 70 x 70 pixel images
- Random rotation by 90 degrees with a probability of 0.2
- Vertical (top to bottom) flip, with a probability of 0.5
- Horizontal (side to side) flip, with a probability of 0.5
- Random contrast shift with a probability of 0.5, min. factor of 0.7, and max. factor of 1.5
- Random brightness shift with a probability of 0.5, min. factor of 0.7, and max. factor of 1.8

The above data augmentation operations brought the dataset to 46486 images. We then split them into two datasets: one for training with 23245 images, and the other for validation with 23241 images. This helps us use the flow_from_directory API from TensorFlow's ImageDataGenerator class, which lazy loads images into memory.

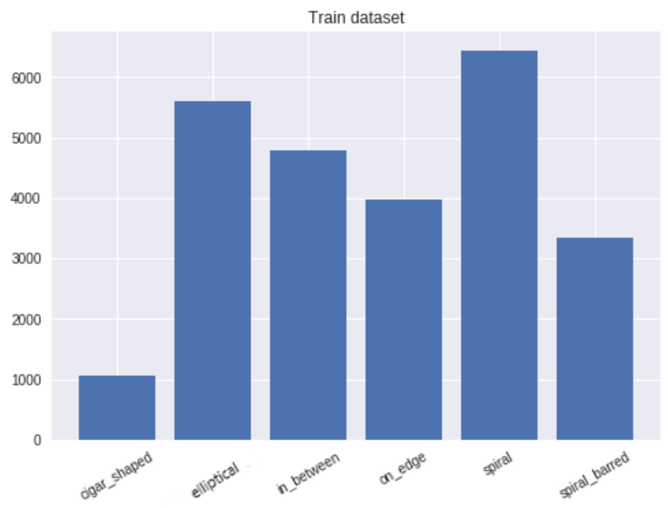

Fig -2: Distribution of Training Images

4.1 Avoiding Overfitting

Data augmentation on its is not sufficient to reduce the effects of overfitting. We also need to tune the hyperparameters of the neural network in order to optimize it for our learning task. Hyperparameters are those which define the architecture of the neural network. Hyperparameter tuning is the process whereby we choose a set of optimal hyperparameters for a learning algorithm.

Dropout is a regularization method that approximates a large number of neural networks with different architectures parallelly. During training, some number of layer outputs are ignored or "dropped out" randomly. This forces the neural network to treat it like a new layer with a different number of nodes and connectivity to the prior layer and makes it learn more robust features. We set dropout layers in our TensorFlow model at probabilities of 0.6 and 0.4.

Learning Rate Scheduler is a callback function in TensorFlow that reduces the learning rate as the number of epochs increases during training. After a certain number of epochs, the callback function starts exponentially decreasing the learning rate value to avoid overfitting.

Early Stopping is another callback function for TensorFlow that monitors the training process on a validation set and stops the process if the performance on that dataset starts to degrade. We set the 'patience' field to 5, which means that it stops training if the performance does not improve for 5 consecutive epochs.

Reduce LR on Plateau is another function that monitors a quantity and if no improvement is seen for a 'patience' number of epochs, it reduces the learning rate by a certain factor. Here, it monitored the validation loss for 2 consecutive epochs, waits for 1 epoch as a cooldown, and reduces the learning rate by a factor of 0.2.

Neural networks are trained using stochastic gradient descent and require a proper loss function when designing and configuring the model. The loss function is used to compute how well a model is performing, and this is computed as a loss value. The value is used to update the weights of the neural network during backpropagation. For our particular learning task, which involves classification across 6 categories of galaxies, we require the use of categorical cross-entropy as a loss function. The use of this function also necessitates the use of a SoftMax activation function in the final layer of the network, since it measures the performance of a classification model whose output is going to be a probability value between 0 and 1. The entropy value increases as the predicted probability diverges from the actual label.

$$-\sum_{c=1}^{M} y_{o,c} \log(p_{o,c})$$

Fig -3: Categorical Cross-Entropy Loss

5. EXPERIMENTAL RESULTS

The neural network was trained on the development workstation equipped with a quad core Intel Core i5 4460 clocked at 3.2 GHz, with 8 GB of DDR3 RAM, and an NVIDIA GTX 970 graphics card with 1664 CUDA cores and 4 GB of VRAM. The model was set to train for 60 epochs with a batch size of 64, and the image dimensions set to 45 x 45 pixels to manage the network complexity. After going through 23 variants of the network, the best performing model ran for 48 epochs until the callback function stopped the training process once the performance started to degrade.

The model trained for around 30 minutes, and achieved net accuracy of 90%.

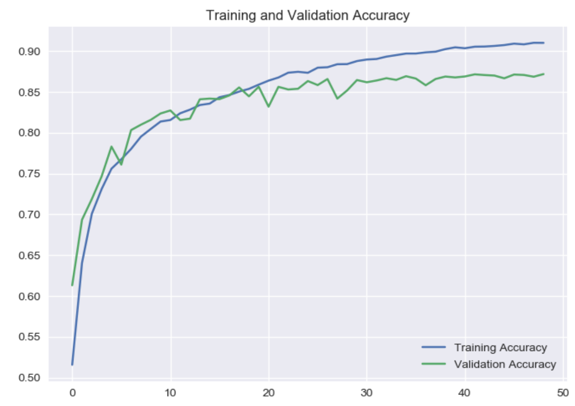

Fig -4: Training and Validation Accuracy

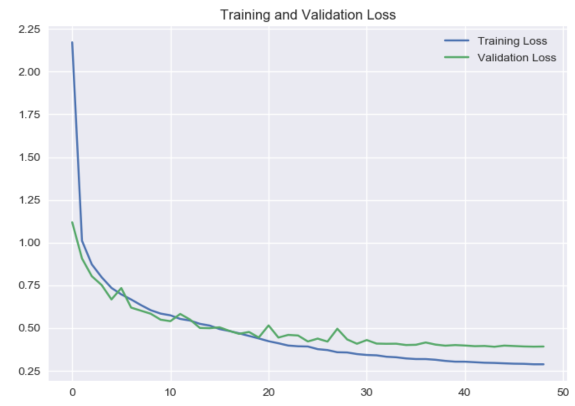

Fig -5: Training and Validation Loss

6. CONCLUSION

In this study, we have developed a CNN to solve the problem of classifying galaxies based on their morphological features from raw images sourced from the SDSS catalogue. The network was able to predict various aspects of the galaxy morphology directly from raw pixel data. The network is robust and is able to avoid overfitting despite a complex architecture by careful tuning of hyperparameters and data augmentation.

Our results indicate that network performance is highly dependent on the quality of data used for training. A large, well-distributed dataset goes a long way in improving the accuracy and generalization ability of the model.

REFERENCES

[1] Jorge De La Calleja, Olac Fuentes, Machine learning and image analysis for morphological galaxy classification, Monthly Notices of the Royal Astronomical Society, Volume 349, Issue 1, March 2004, Pages 87–93, https://doi.org/10.1111/j.1365-2966.2004.07442.x

[2] Gauci, Adam & Adami, Kristian & Abela, John, Machine Learning for Galaxy Morphology Classification, 2010.

[3] Freed, Matthew & Lee, Jeonghwa. (2013). Application of Support Vector Machines to the Classification of Galaxy Morphologies. 322-325. 10.1109/ICCIS.2013.92.

[4] Fang-Chieh Chou, "Galaxy Zoo Challenge: Classify Galaxy Morphologies from Images", 2014.

[5] Wang, Lei & Ma, Zhixian & Xu, Haiguang & Zhu, Jie. (2016). Classification of X-Ray Galaxy Clusters with Morphological Feature and Tree SVM. 701-704. 10.1109/ICMLA.2016.0124.

[6] Ralph, Nicholas O. et al. "Radio Galaxy Zoo: Unsupervised Clustering of Convolutionally Auto-Encoded Radio-Astronomical Images." Publications of the Astronomical Society of the Pacific 131.1004 (2019): 108011. Crossref. Web.