Detecting Active Black Holes Using Radial Light Profile Analysis

Alex Fay
Computer Science, Pomona College

April 2020

1 Purpose

Black Holes are believed to be at the center of many elliptical and spiral galaxies. However, detecting black holes has been historically difficult, as previous techniques involved watching star movements and analyzing gravitational forces and their respective movements. As telescope time is incredibly valuable and expensive, this takes time away from analyzing other celestial objects and is incredibly time consuming. Through analyzing radial light profiles, I aim to build a small archive of potential active black holes for further study.

Through this, I hope to prove the following:
1. Radial Light Profiles can be used to detect active black holes by detecting abnormalities in high energy profiles
2. Machine learning can be used to increase efficiency and accuracy of galaxy sorting and profile fitting functions
3. Explore current techniques of machine learning algorithms to increase my understanding and experience of real world applications

2 Collecting Data

The following sections will cover a step by step process of what I did to get to where I am now as well as an explanation of why.

The Hubble Space Catalog is a massive archive of all Hubble images. As part of Hubble's Mission, a group of researchers have decided to create a 3D virtual map - piecing together each Hubble image into a coherent map. This is called the Sloan Data Sky Survey. I decided to retrieve data from here as it was one of the few places in which retrieving data was well documented and provided flexibility. Other data sets within Hubble had not labelled any objects, meaning many pictures would return thousands of faint, almost invisible objects which would be useless for this project.

For retrieving data, go to skyserver.sdss.org and click on Data and then SQL. Run the SQL query below:

SELECT objID, ra, dec
FROM Galaxy
WHERE
(r - $extinction_r$) between 13 and 24

Luminosity Intensity is sorted on a log scale, where 1 is extremely bright and 30 is not visible. For spiral galaxies, the average light brightness is 20.2. I took a normal distribution sample in order to max out the number of spiral galaxies, which is believed to have more black holes than other types of galaxies. Changing the brightness numbers above, I got 34 percent of my data points from 19-20 and 20-21, 13.5 percent from 18-19, 21-22, etc. However, after analyzing these images, many in the 20 and above range were way to faint to work with. Elliptical galaxies have an average intensity of about 15. I decided to take two samples, a normal distribution of spiral and elliptical galaxies. The total images I pulled was 650,000. This returns a csv file with the galaxy names. I have uploaded the csv files into the Data folder on Github.

SDSS has a search image function but it can only handle 100 images at a time. This was far too slow for my purposes so I built my own image search script. Please see imageRetrieval.py in the "Getting Data" Folder. This retrieves each image as a jpg, 120 by 120 pixels (as zoomed in as I can with the galaxy) and labels the galaxy by Brightness and GalaxyID.

Short Review of Code:
This program works by reading in the csv file (you will need to change it to each file according to radial brightness or compile a master file), then for each element in csv file, retrieve image and relabel. The search works by taking the search url and appending the galaxy name and location in ra and dec. This only works because the search system retrieves images individually using this process. I found this out by inspecting the HTML code of the website and examining the network requests.

3 Building Training Model

Different galaxies have very different shapes. Depending on the rotation and shape, the radial light profile will be very different. I decided to sort the galaxy by type as irregular, elliptical, elliptical cigar, and spiral with and without bars, since they each have different light profiles. This would mean the the prediction ML program I build for the center brightness would be different for each. I originally had included different categories (including on edge), but changed this shortly after my presentation. I plan to use a rotation detection algorithm for this.

The Galaxy Zoo (a project sponsored by SDSS) achieve has classified 140,000 images as stars, galaxies, and nebulas. Volunteers have hand sorted through the data and labelled galaxies as merging, round, having spiral arms, etc. I retrieved the csv file of the data poll results (i.e. the percent of people who responded the galaxy was smooth and similar features).

Using the hubble classification system, I coded up a decision tree following the same guideline and using majority rules for Galaxy Zoo Polling Data. Please see DecisionTreeHubbleClassification.jpg in the classification folder for an overview. It is not at all straight forward.

Code Explanation for galaxyZooTrainingRules.py:

This code takes in the Galaxy Zoo poll csv and classifies each galaxy as spiral, elliptical, elliptical cigar (longer and skinny), lenticular, spiral without bars, spiral with bars, Star, and irregular. This changed since my presentation as I will be using a rotation algorithm and I did more research on each galaxies radial light profiles.

I made three main functions: elliptical, spiral, and lenticular tests. Within elliptical is a test for cigar shape and within spiral is a test for bars. The main function has a test for stars (checks if majority said it is a star) before running tests. If all tests return false, the galaxy is grouped as irregular. Please, please see the galaxyZoo picture mentioned above, it will clarify the process.

For testing elliptical, the program checks if the galaxy is smooth and not a disk. Elliptical galaxies are the only galaxies that are smooth besides irregular. The it tests it is irregular, elliptical, or elliptical cigar. If it is elliptical cigar, LongAndSkinny will be the largest number (pulled from csv). If it is irregular, such that more people said it was odd rather than NotOdd, then I return false and let it run the spiral test. Otherwise it is elliptical.

The spiral test is by far the most complicated as spiral galaxies have a lot of variation. Some of them have rings, bars, rectangular light jets, bulges, and more. To test this, I first check if it a disk, then if it is on edge or not (the galaxy is not facing face on). Depending on this I run two different tests.

For galaxies that are edge on, I do the following:
1. Test if not boxy. If boxy do skip to next inner test
2. Check if irregular through notOdd function, but exclude rings and lensArcs.
3. Label Spiral

For boxy galaxies not on edge, run a bar test
1. check for bar present and check for spiral arm
2. if not odd, then label barred spiral
3. if odd, count as barred spiral if it has a ring or lensArc.

If the above fail (not on edge):
1. Check for a spiral arm present (although many pictures are too faint to truly see this)
2. check for if odd or not and include rings and lens 3. label or continue

I will not bore you to death with similar explanations. The rest of the code follows a similar pattern to test for spiral bars, lenticular galaxies (those in between spiral and elliptical), and more. This outputs a csv with the label of each galaxy and the galaxyID/objID which I use for my model.

4 CNN Model

Yay! Finally!!!!
Training the CNN model using the galaxyZoo data above, I will sort the images of the galaxies we retrieved earlier.

Navigate to galaxyZooClassification.py. I have been trying to make this more accurate and after changing the labels due to the rotation, I keep getting a 'str' has no feature ndim even when I put the data from the csv back into a numpy array. I am still trying to redebug this. :(Not what I wanted.

Code Explanation:

Import A LOT of stuff. I would recommend running this in google colab so that way importing is a lot easier.
Next, unzip the training and testing images. The zip file, even when cut in 10 smaller pieces, is too big for GitHub, I can email it to you.
Then crop each image to a 212 by 212 pixel size. The data set which I pulled in the first section is 120 by 120 pixels. By cropping the training data to as close as possible due to galaxy size, this gets rid of a lot of background noise and makes my training data as similar as possible. You can run this function separately as it will save the images.

Why use a Convolutional Neural Net:
CNN are fantastic for picking out features in each image, as each node in the network can be tailored to maximize the reward. With the wide variety of data,

CNN's can easily classify multiple classes of data. They typically require massive amounts of data, but this is not a problem as the SDSS has over 6 million images. I originally tried small samples with Decision Trees, a random forest, and more, but even on a small data set of 3k images, CNN had the highest accuracy. However CNNs are slow. This takes about 3 hours to run.

After converting the label column (made from the previous section) and making it into a list with each values I use this for training. This is where the current error is. Keras will not except the new labels, but I don't know why yet.

Next, I created a convBlock with 5 layers. I used the relu activation which takes the returns 0 for negative numbers and x for any positive input x. I experimented with several different activation and this one was both fast and accurate. I added a Dense layer at the end of my VGG$_1$6()$modelusingasigmoidactivation$.

In the main function, I first crop the images, then assign the RMSprop as an optimizer (chose this by reading through Tensorflow tutorial), and compiling my model. For computing loss, I used mean square error. I then run a standard fitting function with 40 epochs (number chosen through testing) and then plot the results and evaluate the model. The model has a 72 percent accuracy using the old labels (edge, irregular, star, elliptical, spiralwithBar, spiralWithoutBar].

5 Radial Light Profiles

Please radialLight.py
This function takes in an image and returns the radial light profile. The solution for this is actually extremely elegant. I was able to find some code on stack over flow and modify it for my use.

Code Explanation:
1. Assign the center of the 120 by 120 image
2. Shape image and read in image array
3. Use the first two values of each tuple to calculate the radius. We ignore the third as it is the color values.
4. Use binvalue to take the integral of the brightness. This adds each pixel value within the radius. Using r.ravel() is equivelant to dr or slowly incramenting the radius. 5. The radial light profile is the the total light profile (tbin) divided by the increase in radius (nr)
6. return radius
7. Plot using matplotLib

Please see SpiralGalaxyRadialLightProfile.png for an example of a galaxy. You can run this on any image retrieved from sdss, or any image in general. The peak represents the center of the galaxy while the tail represents background noise of empty space. The small dip near the end is from a star in the picture used.

6 Next Steps

There is a lot of work to be done and can be done with this project. I an determined to continue with my original goal of finding black holes. In order to do this, I need to make a radial light prediction program. Galaxies that are well above the predicted value (150 or 200 percent ish) would be classified as potential black hole candidates. Using the label from the CNN, each type of galaxy will have a different predicted value. For example, elliptical galaxies would be far less bright and less steep on the graph then spiral galaxies I am thinking about doing this using an expectation maximization algorithm and cutting off the graph where the slope equal 1. Then predict the next 40 values or something. If the value is 2 times greater than expected then sort it as abnormal. I am also planning use a rotation algorithm to figure out how much of the galaxy is face on. Edge galaxies will be likely thrown away, but galaxies that are slightly tilted can have thier tilt calculated and factor this into the EM algorithm. This idea is still in the works.

In the short term, I am trying to increase the galaxy label accuracy to 85 percent, get it back to working, and go back through and comment my code better and clean up some code if there is time. I plan to train the EM algorithm by rewarding it for getting closer to the actual value depending on the galaxy type. I also plan to test it by putting a star or a bright object (photoshop) in the center and see if it detects it. This would create a spike in the light profile and prove that it works if classified. It would also allow me to see the exact point at which the galaxy classifies as catalog worthy by dimming the object.

7 Links and References

Retrieving Data using SQL Query Documentation:
http://skyserver.sdss.org/dr16/en/help/docs/realquery.aspxgalview

Image Retrieval Tool:
http://skyserver.sdss.org/dr16/en/tools/chart/listinfo.aspx

Galaxy Zoo/ Hubble Galaxy Classification Tree:
https://data.galaxyzoo.org/gz$_trees/gz_trees.html$

Getting Data From Galaxy Zoo:
https://data.galaxyzoo.org/

Why finding black holes is hard:
https://www.space.com/3457-tricky-task-detecting-black-holes.html

8 Thankyou

Thankyou to Professor Quintin at Pomona College for giving me this idea and helping me figure out next steps through several meetings. I would not have gotten this far without you.

Thankyou for reading this paper and getting this far. To Professor Gu, thankyou for teaching this class. I think it is incredibly interesting and I'm now considering going to graduate school to do research in this field.