Stellar Classification with Convolutional Neural Network

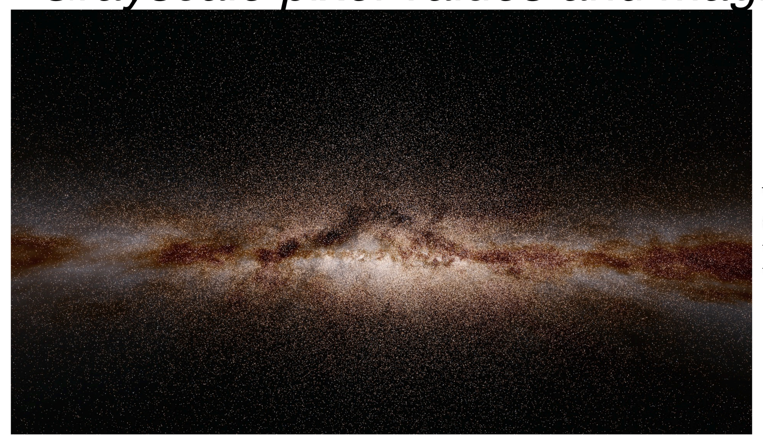
Ahmed Rahman - Andy Nguyen - Sumin Kim - Jack Barsotti

UNIVERSITY OF HOUSTON

Department of Computer Science

Abstract

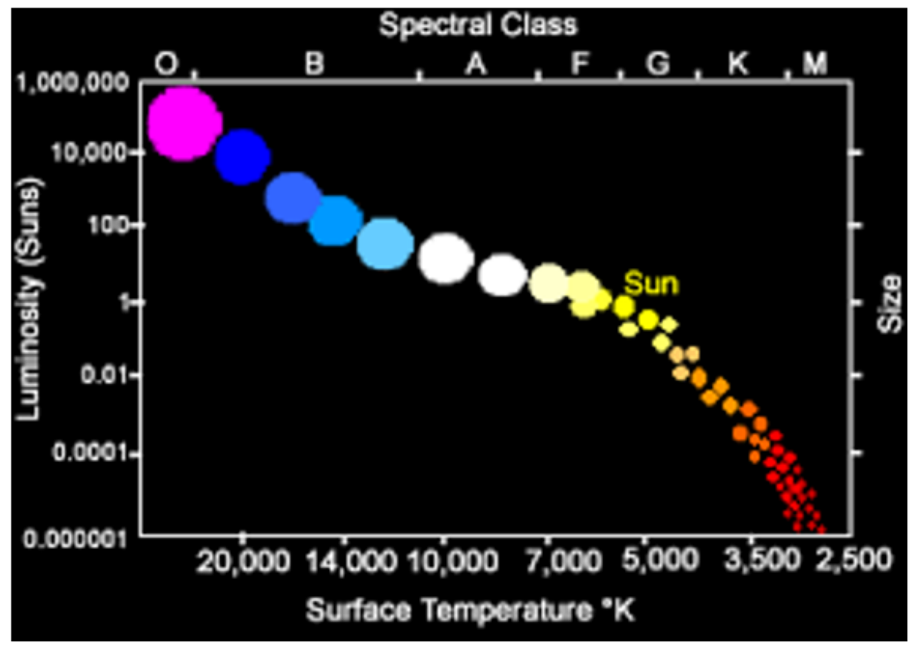
- We propose a machine learning method to identify stars into their respective spectral classes.
- This model was accomplished using simulated images in "Space Engine", a Universe simulator
- Model was trained using specific RGB/ Grayscale pixel values and magnitude/



A galaxy simulation example from Space Engine

Introduction

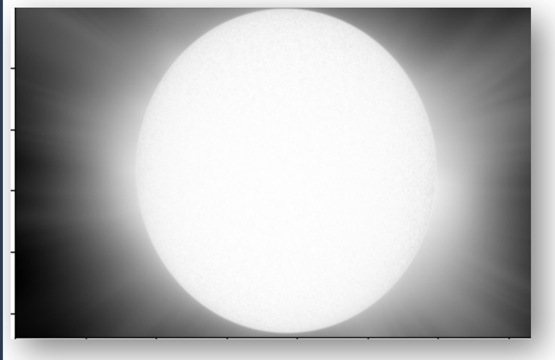
- *Stars are identified based on their spectral characteristics and classified under the Morgan-Keenan system using letters O, B, A, F, G, K, M.*
- *We present the following ML method to classify main-sequence stars assuming a constant distance and filter .*
- *This approach utilizes two techniques:*
  - *Principal Component Analysis (PCA) : for dimensionality reduction*
  - *Convolutional Neural Networks (CNNs) : to classify stars into the 7 spectral classes, from O to M.*



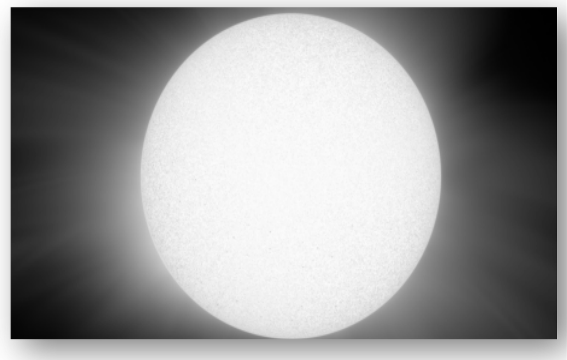
Methods

- *Create simulated images to most accurately represent a realistic scenario. For our models we created gray-scale images with a 50x50 pixel count*
- *Analyze key differences between each class of main sequence star*
  - *Since stellar image datasets along with our own simulated dataset were low-resolution; finer details were out of scope*
  - *Brightness and Light Refraction were key differentiating factors without any numerical data such as Absolute Magnitude and temperature*

Below are simulated images part of our dataset (NOTE: Image resolution was scaled up for presentation)



Simulated F-class star



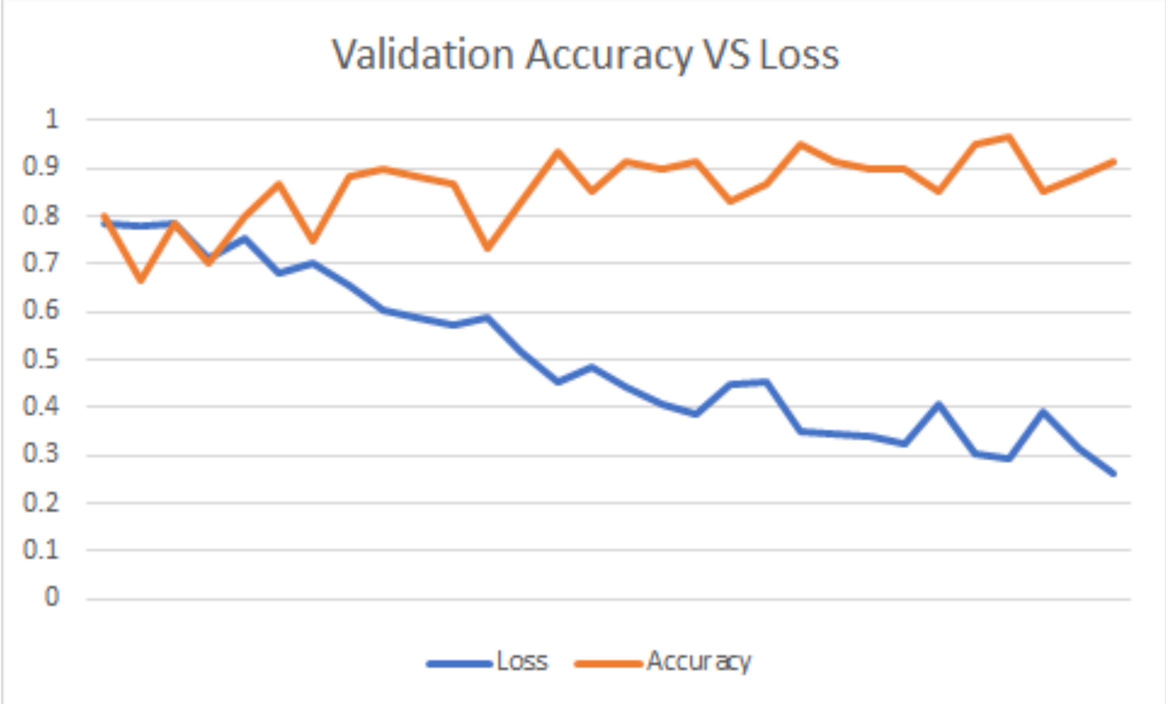
Simulated G class star

- *Created Two Convolutional Neural Network Models:*
  - *Binary Classification*
    - *Used to differentiate between **Yellow** main-sequence star classes (F and G)*
    - *Test ability of model to differentiate by light refraction and brightness*
  - *Categorical Classification*
    - *Classify stars based on 7 main-sequence stellar classes*
- *Optimize, tune, implement techniques such as Dropout layers to avoid model overfitting when needed*

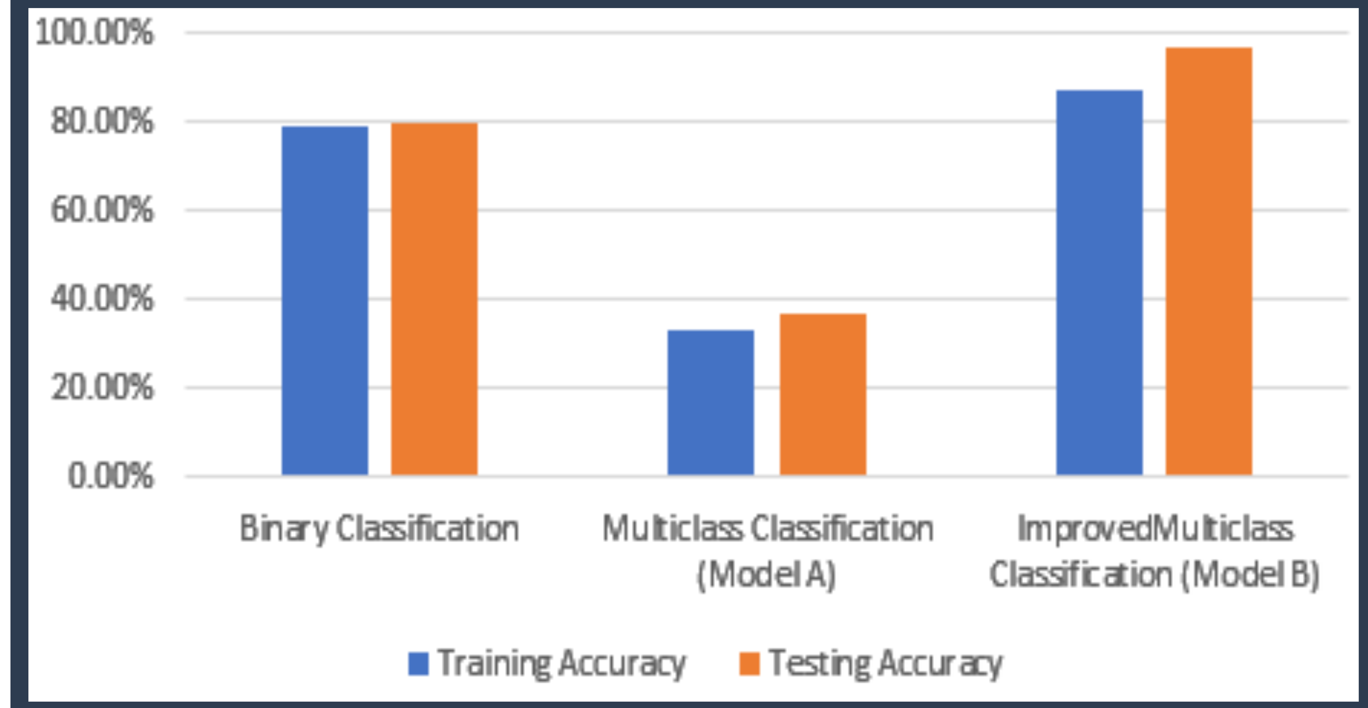
Results

Our first binary classification model returned acceptable results when it comes to accuracy and loss.

However, our first-attempt at multi-class classification with a standard 2-layer model returned poor results, which we then create a second categorical classification model with pooling and dropout layers. This yielded acceptable results without the use of any numerical data.



Validation Accuracy vs Loss (Final Model)



Comparison of the Three Models used

Conclusion

- *The accuracy of our multi-class models varied based on input and configuration of the neural network. The first tested model underperformed at 40% accuracy. Acceptable results were acquired in the second model.*
- *By using more convolved layers and implementing image compression, the convolutional layers filtered out the most significant differences between the spectral classes.*

Future Direction

- *Instead of training the model on a single parameter to define 7 spectral classes, produce a different output based on what is necessary. For example, classifying by colors or other distinct classes.*
- *Acquire other data to feed into the network such as sky coordinates or distance which are also typically captured by a telescope.*
- *Research a viable strategy to train a CNN on actual stellar images especially those with stable point spread function instead of simulated (and clean) sets.*

Acknowledgements

The authors would like to thank Dr. Nouhad Rizk for offering comments to improve this work as well as discussions on convolutional neural networks. This work made use of Space Engine (an interactive simulation and Astronomy software) as well as open -source libraries: Matplotlib, Numpy, Tensorflow, and Keras