Constellation Recognition

Shuangyu Zhao
Michigan State University
zhaoshua@msu.edu

Lijiang Xu
Michigan State University
xulijian@msu.edu

Yan Lyu
Michigan State University
lyuyan@msu.edu

ABSTRACT

Constellations, defined by clusters of bright stars within specific regions of the sky, are easily recognizable with spatial and temporal information. However, without such data, identifying these patterns against the vast expanse of the night sky or within a digital image is challenging. Our research focuses on identifying constellations from varied image sources without spatial or temporal data, and addresses the significant challenge of distinguishing these celestial patterns amidst noise factors like missing crucial stars and brightness and background variations. This work plays a pivotal role in both advanced space technology and basic survival skills. It is crucial for celestial navigation in spacecraft autonomous attitude determination, where constellation recognition is foundational. In such scenarios, identifying star patterns is vital for associating observed star directions with cataloged inertial directions, a process critical in both 'Lost-in-space' and recursive cases of attitude estimation. Simultaneously, in the absence of digital devices, this knowledge becomes a vital survival tool, transforming the night sky into a natural compass. This dual application highlights constellation recognition as both a cornerstone in space exploration and an essential, timeless skill for terrestrial navigation in remote areas.

In our constellation recognition project, we employed several advanced deep learning models to effectively process astronomical images. These included the MLP (Multilayer Perceptron) for its versatility in classification tasks, the pioneering convolutional neural networks AlexNet and the VGG series (VGG11, VGG16) known for their effectiveness in image recognition, and the 34-layer ResNet. By collecting and processing images of six constellations and performing constellation identification. AlexNet proved to be the most effective, achieving an f1 score of 0.82, surpassing other models including complex CNNs (Convolutional Neural Network). The study revealed that a more intricate model architecture does not necessarily lead to better performance in constellation classification, with a key finding that the six brightest stars are ideal for effective classification in the MLP model.

KEYWORDS

Constellation recognition, CNN, MLP, Classification

GitHub Link

https://github.com/shaunzhao666/CSE881_project

1. INTRODUCTION

Constellations, defined by clusters of bright stars within specific regions of the sky, are easily recognizable with spatial and temporal information. The sizes, positions, and types of constellations can help to determine the coordinates of the observation point. Therefore, constellation recognition is significant for locating and navigating not only for people on the ground but also aircraft in the space. However, identifying star patterns against the vast expanse of the night sky or within a digital image is challenging unless you are a master of constellations. While modern technology, particularly smartphone applications, offers convenience in identifying constellations, there is still a significant demand for understanding and detecting constellations without digital assistance for survival and celestial navigation[1], astronomy and research, cultural and historical significance, and reducing light pollution. Our research addresses this challenge, focusing on the identification of constellations from images without relying on any specific spatial or temporal information. This work holds significant value for places where digital devices fail, knowledge of the stars can be a literal lifesaver. The night sky can serve as a reliable compass.

In this study, images derived from different sources were identified as a significant contributor to noise and were carefully treated during the preprocessing phase. Besides, binary threshold, median filter, and dilatation kernel were applied to filter out the noises, including star labels, clouds, brightness, and so on. Most importantly, image rotation was applied to expand the dataset. The coordinates and areas of stars in every image were extracted by image moment. We employed a range of deep learning models, including MLP, AlexNet, VGG11, VGG16, and 34-layer Resnet models, to classify 6 constellations: Ursa Major, Cassiopeia, Gemini, Perseus, Orion, and Summer Triangle. All pre-trained models underwent fine-tuning to adapt to the narrowed-down target number. f1 was chosen to evaluate the models.

The primary contributions of this paper are summarized as follows:

(1) Enhanced Training Versatility: A wide variety of input images enables our model to effectively process constellation recognition for images with missing crucial stars and brightness and background variations.
(2) Efficacy of Fine-Tuning Pre-trained CNNs: We demonstrate the effectiveness of fine-tuning pre-trained CNN models in improving their performance for constellation classification tasks.
(3) Identification of an Optimal Model: Through fine-tuning, the pre-trained AlexNet has emerged as the superior model constellation recognition, outperforming others in accuracy and reliability.

(4) Insights into CNN Model Design: Our research suggests that simpler CNN architectures yield better results in constellation recognition, particularly with sparse datasets.
(5) Impact of Weight Initialization: We investigate the influence of weight initialization on model performance, employing multiple runs to mitigate its impact and ensure robust findings.
(6) Optimization of the MLP model: By determining the optimal number of the brightest stars for model input and incorporating the cosine similarity between the brightest star and others as a novel feature, we enhance the MLP model's capability, providing valuable insights for future model architecture refinements.

2. RELATED WORK

Ji and colleagues[2] proposed a constellation detection algorithm, approaching constellation detection as a one-to-many template matching problem. This method recommended converting images to grayscale and binarized using a threshold-based approach. Based on this algorithm, Molnar and collaborators[3] proposed an optimization-based constellation matching approach to develop the matching step. Nadamoto and others[4] proposed an innovative data preprocessing method for constellation identification, utilizing the coordinates and areas of the top k brightest stars in each image to generate new images. These new images were then used to train VGG19, with extracted coordinates and areas serving as input for PointNet. However, while PointNet is suitable for 3-D object classification and segmentation[5], constellations, despite being theoretically 3-D constructions, can effectively be treated as 2-D patterns due to their considerable distance from us. Additionally, this paper inspired us to explore more pre-trained CNN models for constellation recognition. Alex and others introduced the AlexNet Model[6], VGGNet(as experimented on by Nadamoto with VGG19) was designed by Simonyan[7], and He introduced ResNet in 2016[8]. These pre-trained CNN models demonstrated exceptional performances in the Imagenet Large Scale Visual Recognition Challenge.

3. PROBLEM STATEMENT

This paper worked on classifying 6 constellations using sky images, including Ursa Major, Cassiopeia, Gemini, Perseus, Orion, and Summer Triangle. The images came from screenshotting on the Stellarium website and the photos were taken at different times.

4. METHODOLOGY

This project employed a robust methodology for preprocessing constellation images, incorporating binary threshold, median filter, and dilatation kernels with tailored parameters based on image sources to effectively filter out noises on images. Image moments were then utilized for extracting star coordinates and areas, and preparing data for MLP models. Following data preprocessing, the performance evaluation of MLP, VGGNet, AlexNet and Resnet provided comprehensive insights into their efficacy in classifying constellations.

5. EXPERIMENTAL EVALUATION

5.1 Experimental Setup

(1) Image collecting

The original constellation images were gathered from two different sources: the Stellarium website and photographs captured by our group member, Lijiang, using an iPhone 14 Pro. To maintain consistency, all images were screenshots at a fixed resolution of 770x500 pixels. We selected six constellations for our classification project, namely Ursa Major, Cassiopeia, Gemini, Perseus, Orion, and the Summer Triangle, as depicted in Figure 1.

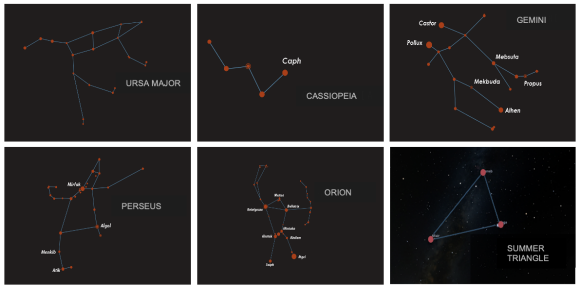


Figure 1. Selected six constellations

To ensure our model is adaptable to a wide range of input data, we deliberately incorporated images with blockage of stars and photos under different weather and background conditions. For each constellation, our dataset comprises 4 Stellarium images with atmosphere effects (AP), 4 without atmosphere effects (NAP), and 4 photographs taken under different weather and background conditions. Within the 8 Stellarium images for each constellation, we intentionally omitted a key star from half of them. It was designed to introduce missing values in the dataset for simulating the real-world obstructions by any structures, thereby enhancing the model's training versatility. The photographs were also taken in diverse conditions, including a blue sky with clouds in the afternoon, a clear starry night, and a cloudy night with long exposure for clearer constellation visibility. These photos also featured various backgrounds, such as urban structures, wires, and cables, adding further complexity to the dataset. This approach enables our model to effectively process a wide variety of input images for accurate constellation recognition.

Additionally, to further enrich our dataset, each screenshot of size 770x500 underwent initial filtering (specific threshold, median filter, and dilation kernel), followed by rotation in 15° increments, ranging from 0° to 345°, and scaled down to a fixed output size of 256x256 pixels, as demonstrated in Figure 2. In total, the dataset encompasses 1728 images across the 6 constellations, under various conditions, generating 24 images for each screenshot.

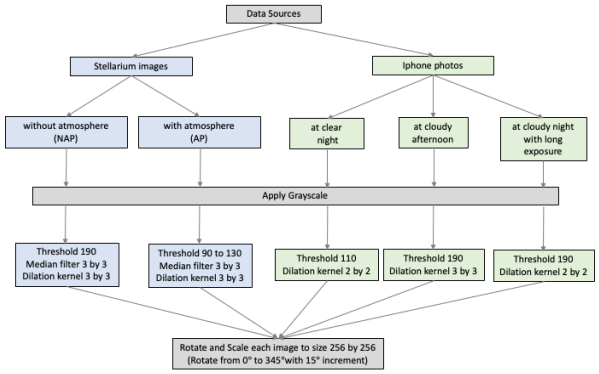


Figure 2. Schematic of the Dataset

(2) Data preprocessing

The preprocessing of the data entailed three key stages, detailed in Figure 2. Initially, we converted the images to grayscale to streamline the process. Subsequently, we employed a variety of hyperparameters for the binary threshold, median filter, and dilation kernel, adapting our approach to suit the different origins of the images to ensure that the star contours were distinctly outlined. In the final step, we transformed all the stars through rotation and scaling before meticulously placing them on a 256x256 pixel black canvas. This process, aimed at enriching our dataset, is mapped out in the workflow presented here:

a. Apply grayscale.

The grayscale was applied to the screenshots with a size of 770x500 pixels since a 1-channel gray image was always easier to deal with than a 3-channel colored one. The grayscale image of Ursa Major is shown in Figure 3.

Figure 3. NAP Ursa Major image before (left) and after (right) applying grayscale

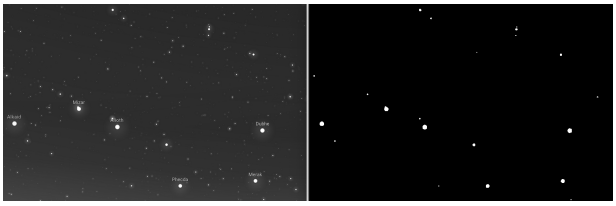
b. Experiment on binary thresholds, median filters, and dilatation kernels for images from different sources.

To remove all the non-star objects and reduce the possible noise, the grayscale images were subjected to a three-step filtering. The first step was to carefully calibrate the binary threshold to remove background scenery. Next, we employed a median filter to efface the star labels from Stellarium screenshots. Finally, a dilation kernel was applied to amplify the stars, ensuring they retained their true size and shape. The hyperparameters selected for each type of image based on their source are shown in Table 1.

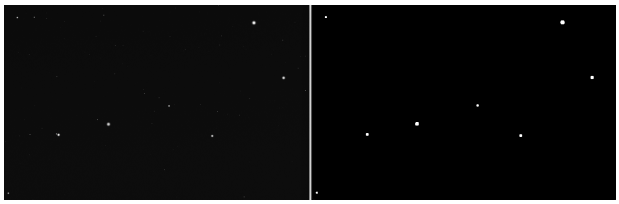
Table 1. Parameters of filtering for different sources' image

	NAP Images	AP Images	Photos (at clear night)	Photos (at cloudy afternoon)	Photos (at cloudy night with long exposure)
Threshold	190	90 - 130	110	190	190
Median filter	3x3	3x3	N/A	N/A	N/A
Dilation kernel	2x2	3x3	2x2	3x3	2x2

We selected the thresholding values to 190 for images with inherently brighter backgrounds, such as non-atmospheric Stellarium images (NAP), and photographs taken during a cloudy afternoon or a cloudy night with long exposure. For photographs captured on clear nights, which typically have lower brightness levels, the thresholding values were reduced to 110. Variations in atmospheric Stellarium images (AP) were accounted for by the presence of labels on the brighter stars within certain constellations: for Gemini, Perseus, and Orion, the threshold was set at 130, while it was set at 90 for the remaining constellations. The median filter was specifically employed for Stellarium images to remove labels. The dilation kernel was selected to not only compensate for any diminishment of the stars after thresholding and median filtering but also to emphasize the brightness of the stars. The grayscale images, processed under these varying conditions, are displayed in Figure 4.

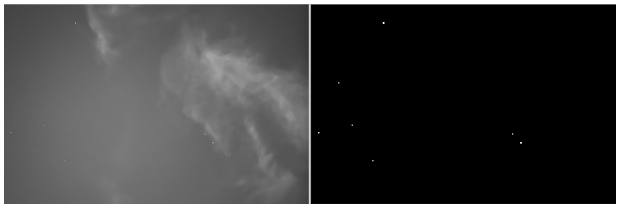


(a) NAP Ursa Major image



(b) AP Ursa Major image

(c) Ursa Major photo at clear night with roof



(d) Summer Triangle photo at cloudy afternoon

(e) Orion photos at cloudy night with wires and cables

Figure 4. Grayscale (left) and filtered (right) for images from different gathering sources.

c. Generate multiple images from the filtered image by rotation and scaling

We enhanced our dataset by applying rotation and scaling to the contours of stars in a filtered image, rather than manipulating the image directly. This method helps maintain the stars' circular shapes, which can become distorted when the entire image is scaled. Initially, we extracted the contours from the original image. By applying image moments to each contour, we were able to determine the coordinates and areas of each star. Following this, we modified the original contours' coordinates and areas through rotation and scaling, and then redrawn them onto a black background of a fixed output size, which CNN models require.

The rotation was applied around the center of the input image, with increments of 15° ranging from 0° to 345°. For each angle of rotation, contours were scaled using a distinct scale factor. This factor was determined by adjusting the bounding box of the rotated star contours to fit a fixed output image size of 256 by 256. To guarantee that all stars remained within the output image, we reduced the calculated scale factor by a safety margin of 0.1. Since area is proportional to the square of length, it was reduced by the square of the scale factor. Consequently, 24 images were generated from each filtered image, resulting in a total of 288 images (12 x 24) for each constellation. These fixed-size images were then used as a dataset for CNN models. For the pre-trained MLP model, we extracted two critical features from all 288 images for all 6 constellations: the coordinates and areas of the stars. The stars were then ordered by their area, from largest to smallest. In cases where two stars had identical areas, we added a minuscule value of 0.0001 to one of them to ensure differentiation. The Numpy files were then generated for both CNN and MLP models as described.

For detecting centroids and areas in a binary image, image moments are commonly used. An image moment, denoted as $M_{i,j}$ for an image $I(x,\ y)$, is defined as follows:

$$M_{i,j} = \sum_x \sum_y x^i y^j I(x,y)$$

Here, $I(x,\ y)$ represents the pixel intensity at location $(x,\ y)$, which is either 0 (black) or 1 (white) in the binary image. For the contours. Consequently, $M_{0,0}$ corresponds to the area of the contour. The centroid coordinates $(C_x,\ C_y)$ of the contour are derived using equations:

$$C_x = \frac{\sum_x \sum_y xI(x,y)}{\sum_x \sum_y I(x,y)} = \frac{M_{1,0}}{M_{0,0}} \qquad C_y = \frac{\sum_x \sum_y yI(x,y)}{\sum_x \sum_y I(x,y)} = \frac{M_{0,1}}{M_{0,0}}$$

These formulas essentially calculate the average x and y position of the contour, allowing us to accurately recreate the star contours with the desired transformation.

(3) Models evaluation

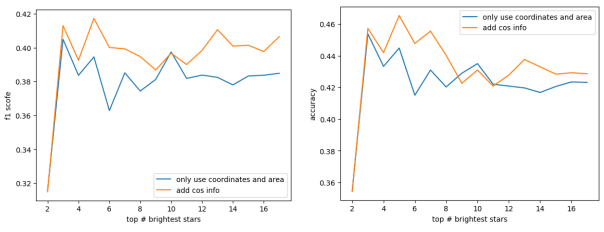
There were 80% of the original dataset was used for training and the rest 20% was used to estimate them. MLP, AlexNet, VGGNet, and ResNet are applied to recognize constellations. The dataset comprises 6 well-balanced classes. While accuracy serves as a straightforward measure for evaluating classifiers, the f1 score is typically calculated for each class individually and then averaged in multi-label classification. This approach provides insights into the model's performance for each class. When focusing on the performance of a particular class, the f1 score is more reasonable, especially because one constellation has significantly fewer images compared to the combined total of images from the other five constellations. Therefore, f1 scores perform better in multi-class classification.

(4) Computing platform and software

- Google colab
- Anaconda
- NVIDIA GeForce RTX3080 Ti

5.2 Experimental Results

For the MLP model, we train the model by not only the stars' coordinates and areas, but also by adding the cosine values of angles between the brightest one and the others. The results are shown in Figure 4. The model trained by both stars' coordinates, area and cosine info has higher f1 score and accuracy than the one only trained by stars' coordinates and area. The optimal number of stars is five.



(a) f1 score **(b) accuracy**

Figure 4. f1 scores and accuracies when choosing different numbers of top brightest stars.

All the pre-trained CNN models were designed to classify a vast array of targets and work on colorful images. However, in the scope of this project, the focus is narrowed down to only six specific constellations, and the images are simply black the with white dots. Consequently, adjustments have been made to fine-tune these models, ensuring optimal performance for the targeted celestial objects. Compared with the original code, in all three fully connected layers, only the output layer is kept. Table 2 attached below can support the fine-tuned decision.

Table 2. Performance differences with and without fine-tuning

F1 score	AlexNet		VGG11		VGG16	
	original	remove FCLs	original	remove FCLs	original	remove FCLs
mean	0.047	0.82	0.047	0.19	0.047	0.048
Standard deviation	0.00035	0.046	0.00031	0.29	0.00033	0.00035

Table 3 shows the performances of all models in classifying 6 constellations. AlexNet has the best performance, and MLP is the next. For VGG11, based on different weight initialization, the f1 scores reach 0.7 twice in ten iterations, while the rest f1 scores are all around 0.047. As for VGG16, the f1 scores are all around 0.48 in 10 iterations.

Table 3.. f1 scores of different models applied in constellation classification without FCLs

F1 score	AlexNet	VGG11	VGG16	34-layers ResNet	MLP
mean	0.82	0.19	0.048	0.148	0.42
Standard deviation	0.046	0.29	0.00035	0.0756	0.017

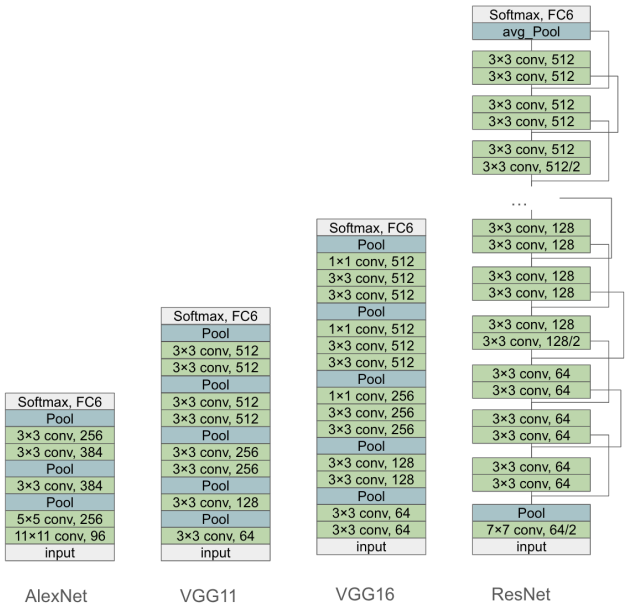


Figure 5. Models' structure.

5.3 Discussion

The experimental results reveal the superior performance of the Fine-tuning AlexNet in classifying constellation images. With an impressive f1 score of 0.82, the Fine-tuning AlexNet with data preprocessing mentioned above surpasses the outcomes reported in comparable papers. Interestingly, the results from VGG11 and VGG16 also suggest that a complex CNN model with numerous layers may not be well-suited for the constellation classification task, indicating that a more intricate architecture cannot necessarily be translated to improved performance. Furthermore, the experiments on VGG11 and ResNet reveal a sensitivity to weight initialization, with its performance being greatly influenced by this factor compared with other models. This underscores the importance of careful weight initialization when employing VGG11 for classifying constellations. For MLP, adding attributes will improve models' performance slightly.

The significant improvement in accuracy of the modified AlexNet model for star recognition, after removing the last two fully connected layers(FCLs), can be attributed to a few reasons:

1) Overfitting Reduction: AlexNet is originally designed for ImageNet, which has over a million images and 1000 classes. Our star recognition task has significantly less data and fewer classes. The original fully connected layers(FCLs) in AlexNet are quite large, which could easily lead to overfitting when used on smaller datasets. By removing these layers, the model's capacity can be reduced, making it more suitable for our dataset size.
2) Feature Relevance: The convolutional layers of AlexNet are designed to extract spatial hierarchies of features from an image (such as edges, textures, and more complex patterns). These features might be more relevant for star recognition than the high-level features the FCLs tend to learn. By focusing on the convolutional layers, your model might be leveraging more relevant features for this task.
3) Simplicity and Efficiency: Simplifying the model by removing complex layers can lead to a more efficient learning process. Fewer parameters mean faster training and often better generalization, especially when the amount of data is limited.
4) Data Specificity: The constellation image is fundamentally different from the data most CNN model was originally designed for. For instance, star images might not require the same level of abstraction and complexity that natural images do. Therefore, a simpler model might be more effective.

6. CONCLUSIONS

This paper indicates that AlexNet without FCLs emerges as the most proficient classifier for constellation images. It can be attributed to the overfitting reduction and dataset sparsity. Furthermore, contrary to the empirical assumptions, heightened model complexity does not translate to improved performance in the realm of constellation classification, as evidenced by the results. For the MLP models, this paper establishes the optimal number of top brightest stars is 6 for effective classification in MLP.

7. FUTURE RESEARCH

Future research should focus on developing a simplified CNN architecture tailored specifically for constellation recognition using shallower layers and larger convolutional window. This entails experimenting with various layer structures, activation functions, and connection patterns to create a model that can effectively capture the unique spatial relationships and patterns inherent in constellation images.

Besides, the diversity of the dataset will be expanding. This involves collecting a larger set of constellation images, potentially from different sources and with varying image qualities, to create a more comprehensive and challenging dataset. The inclusion of images with different noise levels, scales, and orientations will help in developing a more robust model. Moreover, for the celestial navigation, there is only brightness variation in the images. To cope with this application, we need to design a system that can intelligently choose the correct binary thresholding values to distinguish the stars from the background when the sun is close to the aircraft.

8. REFERENCES

[1] Galkin, V. A., and A. V. Makarenko. "Neural network approach to recognition of visible constellations by sky photo image." Journal of Physics: Conference Series. Vol. 1864. No. 1. IOP Publishing, 2021.

[2] Ji, S., Wang, J. and Liu X. *Constellation detection*, Stanford University Tech. Rep., 2016, [online] Available: https://web.stanford.edu/class/ee368/Project_Spring_1415/Reports/Ji_Liu_Wang.pdf.

[3] Molnár, Z. and Kiss, D. *Constellation Recognition on Digital Images*. 2023 IEEE 17th International Symposium on Applied Computational Intelligence and Informatics (SACI), Timișoara, Romania, 2023, pp. 000443-000448.

[4] Nadamoto, S., Mori, N. & Okada, M. *Constellation identification method using point set data*. Artif Life Robotics 28, 361–366 (2023).

[5] Qi, C.R., Su, H., Mo, K. and Guibas, L.J., 2017. *Pointnet: Deep learning on point sets for 3d classification and segmentation*. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 652-660).

[6] Krizhevsky, A., Sutskever, I. and Hinton, G.E., 2012. *Imagenet classification with deep convolutional neural networks*. Advances in neural information processing systems, 25.

[7] Simonyan, K. and Zisserman, A., 2014. *Very deep convolutional networks for large-scale image recognition*. arXiv preprint arXiv:1409.1556.

[8] He, K., Zhang, X., Ren, S. and Sun, J., 2016. *Deep residual learning for image recognition*. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).