City Identification using Stereographic Projections of Celestial Observations

Fouad Ayoub
University of Maryland, College Park
fayoub1@terpmail.umd.edu

Sebastian Ashcallay
University of Maryland, College Park
sashcall@terpmail.umd.edu

Abstract

Astronomical observations have long served as a powerful tool for position determination (latitude and longitude) through constellations and stellar patterns. Under a controlled, simulated environment, this project explores whether modern convolutional neural networks (CNNs), such as ResNet, can classify geographic locations based on star field imagery. Two synthetic datasets of night sky projections were generated for known cities, dates, and times, then augmented to simulate realistic conditions including cloud cover, blur, and light pollution. Two models, corresponding to the two datasets, were trained to predict the city of origin from these images and evaluated using classification accuracy, top-3 performance, and confusion matrices. Results from both suggest that CNNs can effectively learn spatial patterns in celestial data, opening the door to machine learning for visual navigation and a future real-world implementation with pictures of the sky.

1. Introduction

Convolutional Neural Networks (CNNs) prove to be widely applicable in various classification tasks. including image and pattern recognition. Models like AlexNet, VGGNet, and ResNet have demonstrated the effective use of deep networks with a large learning capacity in a variety of real-world applications [4, 5, 7].

On the one hand, AlexNet and VGGNet have been widely used as pre-trained models for **transfer learning**, where the network's learned features are fine-tuned for a specific task with newly-labeled training data. AlexNet has been successfully applied to medical image analysis, such as the detection of diabetic retinopathy in retinal fundus images [3], while VGGNet has been used for fine-grained image classification, object detection, and semantic segmentation [1]. On the other hand, ResNet, which seeks to solve the problem of disappearing gradients for deep neural networks, has been shown to outperform previous CNN architectures [1]. In particular, ResNet has been successfully applied to medical imaging problems, such as the detection of breast cancer metastases in lymph node histology images [9].

Given the strong performance of CNNs in image classification, this project explores whether a CNN can identify the geographic location of an observer, chosen from a set of cities, based on a stereographic projection of a star-filled sky.

2. Motivation

> "Fix your course on a star, and you'll navigate any storm."
> — Leonardo Da Vinci

For centuries, humans have used the stars as compasses, primarily in maritime endeavors. The development of computers in the 20th century allowed us to create algorithms that used these techniques in a far more precise way, including GPS navigation. While other navigational techniques are also used in combination, celestial navigation remains one of the most reliable and robust ways of determining position in navigation [6]. Today, with the advent of pre-trained CNN models, a question that comes to mind is whether classification can help extract geographic information directly from star-field imagery.

Normally, the night sky varies subtly with an observer's latitude, longitude, and time — affecting the position of celestial poles, constellation orientation, and star density. While these variations can be imperceptible to the human eye, they are consistent and structured, making them suitable for spatial pattern recognition by neural networks.

Through standardized, stereographic projections to preserve angular relationships in the sky, our project examines if a CNN can accurately predict the geographic location in which an observer resides.

3. Related Works

3.1. 3D Object Classification with Stereographic Projection

Stereographic representations of 3D objects paired with CNNs have shown to provide comparable results to state-of-the-art classification methods while requiring less GPU

memory and network parameters [10]. The usage of a stereographic projection allows the information encoded into three spatial dimensions to be translated into a 2D image that preserves angular information. As a result, a simple, shallow CNN is capable of providing high accuracy in classification tasks. Yavartanoo, et al. (2019) show that stereographic projections provide traditional 2D CNNs the capacity to successfully conduct classification tasks in 3D spaces. This motivates our project in utilizing stereographic projections of star-field observations, since the 3D information captured can then be well-represented in only two dimensions.

3.2. Photo Localization with Deep Neural Network (ResNet)

Using the ResNet model, previous work has shown that CNNs can accurately classify geographic locations using images of the location itself [8]. The work examines this model's ability to conduct location classifications, and it concludes that a model like ResNet can be used to capture geographic relationships from images. Our work extends this finding and motivates our usage of the ResNet architecture specifically for its potential capability in geographic classification. With its success in classifying location by state (in the US), we have reason to believe it will also perform well in classifying location by US city. Additionally, since this classification is done solely using images unique to the location itself, this capability of the model may also lend itself towards the usage of star-field observations projected onto a 2D image.

4. Methodology

Our project approach to our location classification problem assumes that there is no additional information, such as GPS coordinates, at inference time. Instead, the solution relies on **transfer learning**, where a CNN pre-trained on a large-scale dataset (e.g., `ImageNet`) is fine-tuned to recognize spatial star patterns that are characteristic of Earth-based locations, specifically cities in the US. By leveraging knowledge learned from general visual features and adapting to sky maps, the model would be able to infer a likely city of origin from a single projected image of the night sky.

To begin, we selected a simplified dataset with minimal temporal variation to better isolate the geographic differences between cities. Once the model demonstrated competence under these controlled conditions, we introduced larger temporal gaps—extending the time interval between images from 10 minutes to 12 hours—to evaluate its ability to generalize across broader time scales. Additionally, we chose cities that were evenly distributed across the continental United States to minimize the potential confounding effects of neighboring city similarities (see **Figure 1**). Essentially, we ended with two different models: an initial **model A** and a more robust **model B**. How we arrived at these two models will be discussed in greater detail in the next sub-section.

Figure 1. Evenly-distributed cities across the US were used in Dataset B in order to generalize the model geographically.

4.1. Data Generation and Augmentation

First, to create our training datasets, a custom rendering pipeline was developed, using code authored by Viyaleta Apgar [2], who used astronomical libraries such as the `hipparcos` dataset (containing star location data) to simulate realistic sky maps. Under a function wrapper created specifically for this project, we built stereographic projections of the sky as circular star maps centered within a square image, taking city names, dates, and times as input.

We trained our first model on a dataset containing a moderate sample size of cities, but without a significant time variation (Dataset A). We used 20 major cities and generated observations for 05-15-2024, between the times of 10:10 PM and 10:20 PM. This resulted in 200 base images (20 cities $\times$ 10 minutes , each resembling **Figure 2**).

Then, to support geographic and temporal diversity, we created Dataset B, which included 32 major cities from the 50 US states. For each city, 12 hours throughout a single day (05-15-2024) were chosen to span temporal changes relating to Earth's rotation and star visibility at night (7 PM - 6 AM). This setup resulted in 384 base images (32 cities $\times$ 12 hours).

Furthermore, to improve the model's ability to generalize and simulate real-world observational conditions, clean-sky images in each dataset were augmented a number of times using a custom augmentation pipeline. This pipeline introduced realistic variations, including

1. **Cloud occlusion:** Random, semi-transparent patches on the sky map to mimic cloud cover.
2. **Light pollution:** Radial brightness gradients from the horizon inward.
3. **Blur:** Gaussian blurring to simulate atmospheric distortion or out-of-focus images.

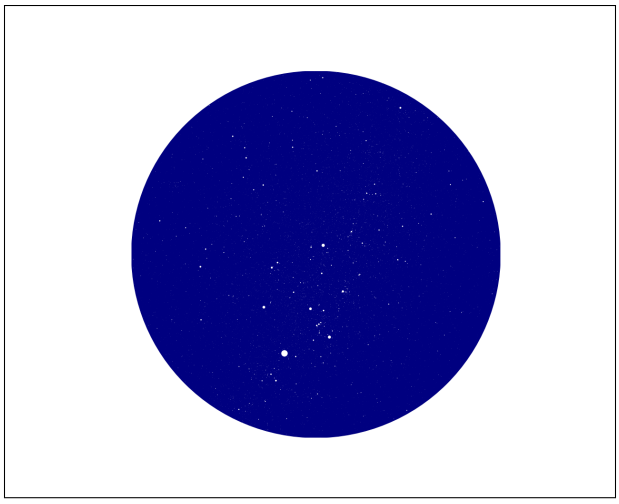

Figure 2. Base image. This is a stereographic projection of the sky and the stars from New York City, NY (01-15-2024 22:00:00).

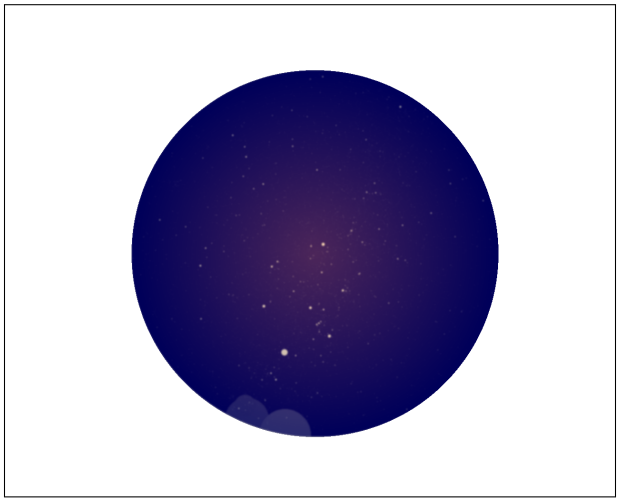

Figure 3. Augmented image. This is a stereographic projection of the sky and the stars, including blur, cloud occlusion, and light pollution conditions from New York City, NY (01-15-2024 22:00:00).

Augmentations were applied to the circular sky area, as seen in **Figure 3**. With the introduction of these realistic conditions, our image count was multiplied by the number of augmented images used:

- For Dataset A, 4 augmented images were used, each with an arbitrary combination of the augmentations described above, resulting in $200 \times 4 = 800$ total images.
- For Dataset B, 5 augmentations were used in the same manner, resulting in $384 \times 5 = 1920$ total images evenly distributed across all US states.

All initial, synthetic images were saved to a `dataset_clean` directory, while all augmented images were saved to a `dataset_augmented` directory. A subfolder was created for each location, and each image file was named after the city, date, time, and augmented index (if augmented).

4.2. Preprocessing

All augmented images were resized to 224×224 pixels to be compatible with standard CNN input dimensions. The images were then normalized using the mean and standard deviation values from the `ImageNet` dataset to align with pre-trained model expectations. Each image's label was automatically derived from its directory name, which corresponded to its geographic origin.

4.3. Model Architecture

The classification model was based on `ResNet18`, a pre-trained residual neural network architecture in `ImageNet`. As intended with transfer learning, the final fully connected (FC) layer of ResNet was replaced with a new layer that has 20 output units for Dataset A and 32 output units for Dataset B, one for each city. The rest of the network retained its pre-trained weights and was fine-tuned on the augmented night sky dataset. Additionally, training was performed using the Adam optimizer with a learning rate of 0.5% and cross-entropy loss over 10 epochs. Finally, the data set was split into training and validation sets using a $80/20$ split.

4.4. Evaluation

Our model's performance was evaluated using a variety of classification metrics.

- A **confusion matrix** of predicted versus actual locations was generated to visualize which cities were frequently misclassified and to identify confusion clusters.
- Per-class **precision, recall, and F1-score** were computed using scikit-learn's `classification_report`, revealing strong-performing classes and locations where the model struggled the most.
- **Top-3 accuracy** was calculated to determine how often the correct city appeared among the top predictions.

5. Experimental Results and Findings

Each model was trained on its respective dataset. The training loss and validation accuracy were recorded after each epoch in training and plotted in **Figures 4** and **5**.

5.1. Confusion Matrix

Using the `scikit-learn` package, we generated a confusion matrix (**Figure 6** and **7**) from our model's validation set, where each row represents the `Actual` class (true cities), and each column represents the `Predicted` class (predicted cities).

- Diagonal dominance in the matrix indicates correct predictions.

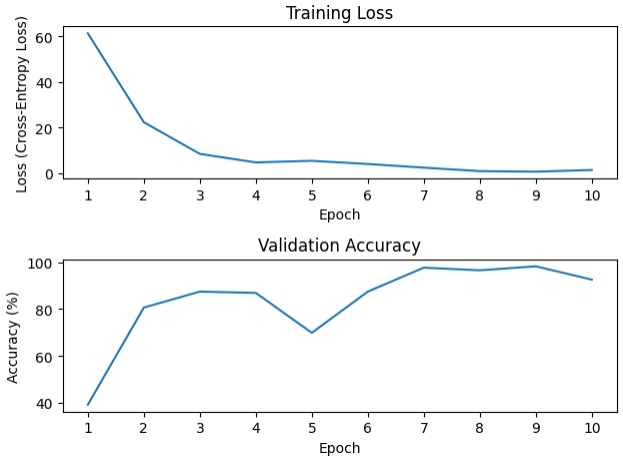

Figure 4. Validation loss and accuracy variation over 10 epochs when training on Dataset A.

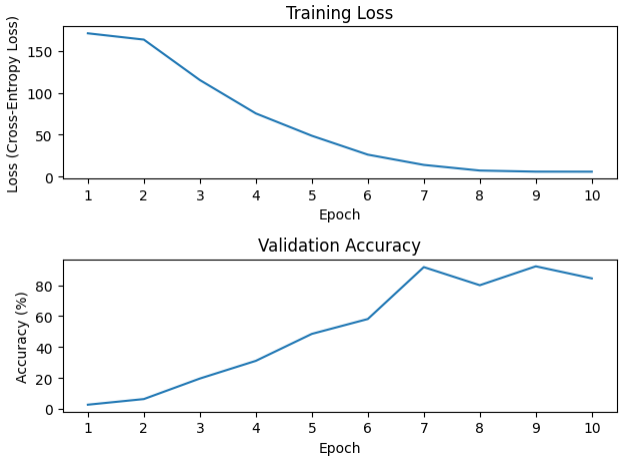

Figure 5. Validation loss and accuracy variation over 10 epochs when training on Dataset B.

- Off-diagonal errors show systematic confusions, often between geographically or visually similar regions.

For the confusion matrix of model A, we see a clear diagonal dominance, showing the success of the model overall. However, we also see some isolated off-diagonal errors; specifically, confusions between cities like Baltimore, MD, with Washington, DC, or New York, NY, with Chicago, IL. These confusions show that **inaccuracies arise when the model attempts to distinguish between cities that are geographically close or within similar latitudes**. For example, in the case of New York and Chicago, their latitude only varies by 1.14 degrees. Since the time interval for the dataset was determined by local time zones, the images for New York and Chicago may have been very similar and caused the model to confuse the two cities.

Similar results were seen in model B's confusion matrix,

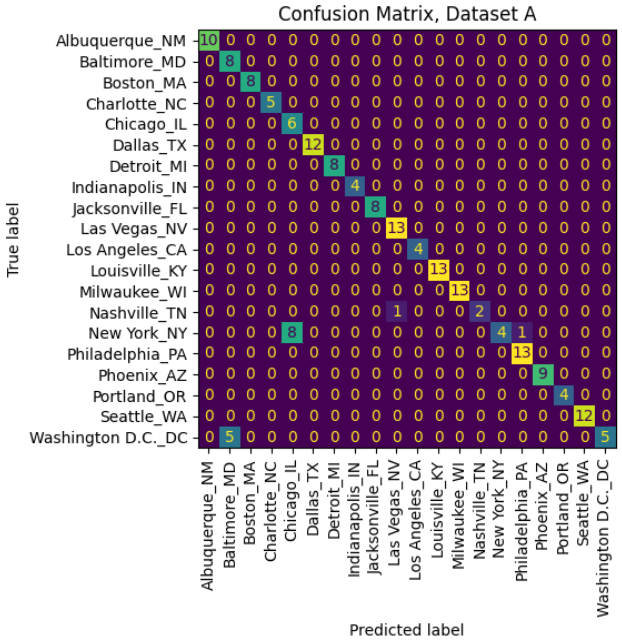

Figure 6. Confusion matrix when training on Dataset A.

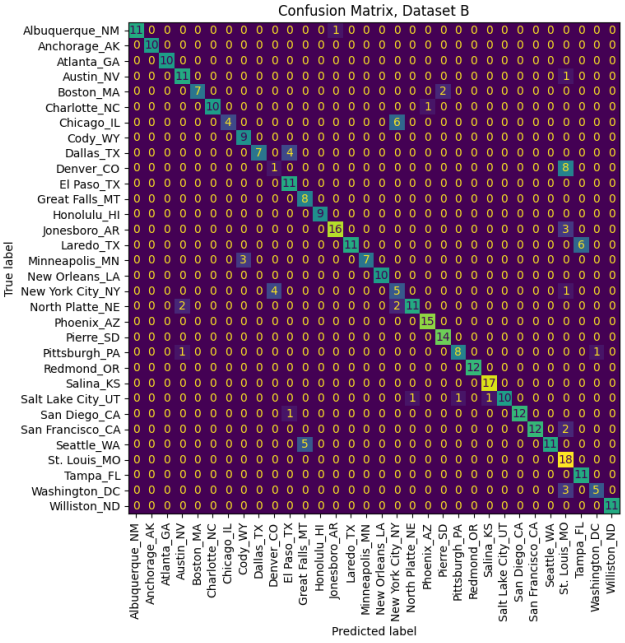

Figure 7. Confusion matrix when training on Dataset B.

with a clear diagonal dominance and off-diagonal errors between cities of similar latitudes. For instance, the model often confused Washington, DC, with St. Louis, MO, where the latitude difference is as low as 0.26 degrees.

Model	Precision	Recall	F1	Top-3
A	95.25%	91.48%	90.86%	100.00%
B	88.02%	84.38%	84.37%	99.74%

Table 1. Classification report and top-3 accuracies for models A and B, showing weighted averages for precision, recall, and F1-score, as well as top-3 accuracy scores.

5.2. Classification Report

Using the `scikit-learn` package, we generated a classification report for models A and B, which included precision, recall, and F1-score for each city. The reports are summarized in **Table 1**.

- **Precision:** How many predicted cities were correct.
- **Recall:** How many actual cities were correctly predicted.
- **F1-score:** Harmonic mean of precision and recall.

5.3. Top-3 Accuracy

In practical applications, it's valuable to check whether the correct city is among the top few predictions. Along with out classification report, we calculated top-3 accuracy scores from the validation sets of models A and B (see **Table 1**).

5.4. Prediction on Unseen Images

Finally, to simulate a real-world application, we rendered and augmented new images for combinations not seen during training: new locations (not previously classified) and new times. The trained models were asked to predict locations from these unseen images, and the top predicted cities were recorded along with their confidence scores. Ideally, the classifier closest to our tested city would yield the highest confidence score. When tested on College Park, MD,

1. Model A threw "Washington, D.C." with a confidence score of 61.90% and "Baltimore, MD" with a confidence of 33.69%, showing its ability to generalize its spatial understanding of the celestial observations to out-of-sample predictions.
2. Model B threw "Washington, D.C." as the predicted label 100% of the time, out of 100 iterations.

6. Conclusion

Convolutional Neural Networks have proven themselves to be useful in a wide variety of classification tasks when using pre-trained models for transfer learning. While the usage of CNNs to extract geographic information has been proven in previous works, these models have not yet been used to extract such information from celestial observations. By combining the proven predictive power of ResNet with stereographic projections, we trained models on multiple datasets to examine how it performs in classifying the geographic location of the observer. Datasets were created from synthetic images augmented to mimic real-world observational set-backs, such as cloud occlusions, light pollution, and blur.Models trained on datasets with varying temporal intervals proved to have great predictive capacity in the classification task, with **95.25%** precision for a dataset with small temporal intervals and **88.02%** precision for a dataset with large temporal intervals.

6.1. Next Steps

While this project demonstrates that CNNs are powerful inference tools, providing geographic location from star-field images with promising accuracy, it is important to acknowledge the limitations of this approach. Most notably, the model was trained exclusively on synthetic images generated under idealized and controlled conditions. These images do not account for the full range of distortions, obstructions, and noise present in real-world observations, such as atmospheric scattering, camera imperfections, sensor noise, or obstructed views.

As a result, the current system may not yet be applicable for direct use in outdoor navigation or mobile device contexts. Real-world deployment would require domain adaptation techniques, transfer learning from real sky image datasets, and possibly fine-tuning with field-collected data.

Nonetheless, the results offer a compelling proof of concept. They suggest that spatial patterns in the sky — even when subtly encoded — can be learned and used for geographic inference. With further development and real-world validation, this line of research could contribute to new forms of astronomical navigation and visually grounded spatial reasoning in machine learning.

6.2. Acknowledgments

The LLM ChatGPT was used to help polish and revise this report in order to ensure our ideas were well-represented. All images were created using Google Colab (https://colab.research.google.com/drive/1xN-SNFRwnfZRnlENAS7GklshPdF0-YJJ#scrollTo=thtuY1_BT3E6).

References

[1] Laith Alzubaidi, Jinglan Zhang, Amjad J. Humaidi, Ayad Al-Dujaili, Ye Duan, Omran Al-Shamma, J. Santamaría, Mohammed A. Fadhel, Muthana Al-Amidie, and Laith Farhan. Review of deep learning: concepts, cnn architectures, challenges, applications, future directions. *Journal of Big Data*, 8(1), 2021. 1

[2] Viyaleta Apgar. I made a sky map in python. here's how., 2022. 2

[3] I Arias-Serrano, PA Velasquez-Lopez, LN Avila-Briones, FC Laurido-Mora, F Villalba-Meneses, A Tirado-Espin, J

Cruz-Varela, and D Almeida-Galarraga. Artificial intelligence based glaucoma and diabetic retinopathy detection using matlab ? retrained alexnet convolutional neural network [version 2; peer review: 2 approved]. *F1000Research*, 12 (14), 2024. 1

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 1

[5] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2012. 1

[6] Zvonimir Lušić, Tony Pinčetić, and Paško Krančević. Role of celestial navigation in modern day and future navigation. In *Maritime Transport Conference*, 2024. 1

[7] K Simonyan and A Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, pages 1–14. Computational and Biological Learning Society, 2015. 1

[8] Sudharshan Suresh, Nathaniel Chodosh, and Montiel Abello. Deepgeo: Photo localization with deep neural network, 2018. 2

[9] Jun Wang, Qianying Liu, Haotian Xie, Zhaogang Yang, and Hefeng Zhou. Boosted efficientnet: Detection of lymph node metastases in breast cancer using convolutional neural networks. *Cancers*, 13(4), 2021. 1

[10] Mohsen Yavartanoo, Eu Young Kim, and Kyoung Mu Lee. Spnet: Deep 3d object classification and retrieval using stereographic projection. In *Computer Vision – ACCV 2018*, pages 691–706, Cham, 2019. Springer International Publishing. 2