Stata Navigation

Krishanu Datta krishanu@mit.edu
Sara Pasquino pasquino@mit.edu
Hayden Ratliff hratliff@mit.edu

May 14, 2024

Abstract

Navigating complex indoor environments like the Stata Center can often be challenging, even for those familiar with the layout. In this paper, we present a novel system designed to automate the process of guiding individuals through such spaces using their smartphones. Leveraging deep learning techniques, particularly Convolutional Neural Networks (CNNs), we classify user-provided images to determine their location within the Stata Center's first floor. Our methodology includes custom-designed CNNs as well as fine-tuning of existing foundation models. To build our classification dataset, we captured videos and extracted the video frames from 37 locations on the first floor of the Stata Center. The classification model achieved a 46% top-1 and 63% top-5 accuracy out of sample. Furthermore, we introduce a route optimization algorithm that, given a destination, provides directional guidance to users, considering real-world distances and orientation. Our approach demonstrates promise in addressing the navigational challenges of complex indoor spaces and lays the groundwork for further advancements in this domain.

1 Introduction

Indoor navigation poses unique challenges compared to outdoor scenarios, with complex structures often leading to confusion and disorientation. MIT's Stata Center exemplifies such complexity, notorious for its labyrinthine layout. Traditional mapping applications excel outdoors but often fall short indoors, where GPS signals are unreliable, and architectural features obscure line-of-sight. This paper addresses this issue by proposing a system that leverages deep learning and route optimization techniques to guide individuals through the Stata Center's intricate spaces.

Our approach utilizes Convolutional Neural Networks (CNNs) for location classification, enabling users to pinpoint their whereabouts within the Stata Center's first floor by taking a photograph with their smartphones. We employ custom CNN architectures and pre-trained foundation models to achieve accurate location identification. Complementing this is a route optimization algorithm that directs users to their desired destination, considering spatial constraints and user orientation.

We present our data collection and pre-processing methodologies, and delve into the classification and optimization algorithms, highlighting their efficacy in providing seamless guidance through the Stata Center's maze-like terrain. We hope we are able to replicate and pave the way for increased accessibility from CV indoor navigation tools in the future [10] [8].

2 Related work

Several approaches to image classification and image-based navigation in the literature are relevant to our project.
The ImageNet leaderboard provides a survey of state-of-the-art models for image classification tasks [2] [5].

Morar et al. (2020) explore sensor-based navigation in complex buildings, highlighting the effectiveness of combining multiple data sources for improved

accuracy and reliability [6].
Approaches combining natural language processing and computer vision include the Room-to-Room (R2R) visual navigation challenge [1], which benchmarks systems interpreting and executing complex navigational instructions, analogous to our classification and routing tasks.
The top-performing model is Meta-Explore, introduced by Hwang et al. (2023) [4]. Meta-Explore is a hierarchical vision-and-language navigation model using scene object spectrum grounding, enabling dynamic exploration and understanding of indoor environments, particularly relevant for structurally complex buildings like the Stata Center.

3 Methods

3.1 Data Collection and Preprocessing

We restricted data collection to the first floor of the Stata Center to promptly start experimenting.

We identified 37 locations as labels for our classification model, including classrooms, restrooms, exits/entrances, and other locations. For each point, we took a 10-15 second video at 30 frames per second, resulting in 300-450 observation points per label. See Appendix Figure 1 for space discretization and label locations. We then extracted all frames from each video and resized them, resulting in 10,189 points split 80%-20% between train and test sets. However, the test set was nearly identical to the train set since consecutive video frames differed minimally. Indeed, our initial approach led to data leakage from the train set to the test set.

To address this, and make sure that our model's performance was accurately assessed, we collected a True Test Set of 5 images per label from different angles and locations within each label area. Our experimental results use this new test set.

3.2 Location Classification

We experimented with several approaches to location classification, including Custom-Designed CNNs using Keras' TensorFlow, and Embedding Similarity using OpenAI's CLIP [7]. We also experimented with transfer learning, by building custom classifications on top of pre-trained foundation models like ResNet18 using PyTorch.

3.2.1 Custom-Designed CNN

We carried out several different experiments within this section, such as varying the network depth and width, augmenting the train data through random rotation and flip of the images, and adding a dropout layer after the last pooling layer in the network for regularization. We landed on the architecture outlined in Figure 2

3.2.2 Embedding Similarity with CLIP

Since the simple CNN showed low accuracy, we adopted Embedding Similarity, a state-of-the-art approach. We extracted image embeddings using OpenAI's CLIP model, a foundation model trained on internet images labeled with natural language[7]. This model captures image features like edges, colors, shapes, and semantic meaning.

With extracted embeddings for our data (Train, Original Test, and True Test), we classified True Test Set images by maximizing cosine similarity between the given image and images in the assigned dataset, now considered a "vocabulary" or "corpus". We conducted different experiments within this approach:

- Max Score per Label, Train Data Vocabulary: For each label, we select the training image with the highest cosine similarity to the test image. Classify based on the label of the most similar image overall.
- Max Score per Label, All Data Vocabulary: Same as above, but using all available images (train + test) as potential matches.
- Average Score per Label, Train Data Vocabulary: We compute the average similarity score between the test image and all training images within each label, then classify based on the label with the highest average similarity.

- Average Score per Label, All Data Vocabulary: Same as above, but using all available images.

3.2.3 Foundation Models

In addition to building custom CNN models, we utilized pre-trained foundation models for the image classification step. We experimented with AlexNet, ResNet18, and VGG16 for location classification [9] [3].

These models were chosen because they perform well on the ImageNet dataset, because we had previous experience using them in this class, and because they were small enough to be practically utilized in Google Colab [2] [5]. .

To train our classifiers, we extracted image embeddings from the aforementioned foundation models, and then trained custom classification heads on top of these embeddings. We adopted this strategy to save computational resources and training time, allowing us to only have to pass our images through each foundation model once.

We trained two different classification heads for each foundation model, a simple classification head and a deeper one. The simpler classification head utilized two hidden layers and two dropout layers, while the deeper classification head utilized three fully connected, three batch normalization, and three dropout layers.

3.2.4 Classification Evaluation

We chose several KPIs to evaluate both our models: top-1 accuracy, also known as out-of-sample accuracy; top-5 accuracy, the percentage of samples identified as one of the 5 closest labels by distance to the true label; median error, the median distance between true and predicted label; mean error; and maximum error.

These KPIs reflect the ultimate use case of our model. Because the locations we identified on Stata's first floor are quite close together, it is reasonable that there will be some misclassification error between labels that are 20 feet away from eachother. So, we examine top-5 accuracy and classification error distance to determine whether our classifications are sufficiently within the vicinity of the true label.

3.3 Route Optimization

In addition to having a working image classification model, we built an algorithm which helps the user get where they need to go (provided the correct results of image classification).

3.3.1 Naive Routing

This naive algorithm works by converting pixel measurements from our first-floor map into real-world distances and providing directional assistance. Initially, it establishes a conversion scale from pixels to feet using known measurements. Navigation from the starting point to the destination involves determining the angle and distance between the points, translating this into feet, and computing the necessary turns based on the user's current orientation. The algorithm can provide guidance using either absolute compass directions or relative turning instructions, depending on the user's input. (See function output in Listing 1)

4 Results

4.1 Custom-Designed CNN

The best performing CNN (Figure 2) that we ended up using for the rest of the analysis had:

- 3 Convolutional Layers
- 1 Dropout Layer for Regularization
- Weight Initialization
- Train Data Augmentation for Regularization
- ADAM Optimizer for Adaptive Learning Rate

In our initial naive approach, which caused data leakage from train to test set, these Custom-Designed CNNs showed impressive results, such as out of sample accuracy above 90%. Nonetheless, when applied to the True Test Set, even the best performing CNN had an Evaluation Accuracy of 14.05% (Table 1). This accuracy is very low, yet it is worth to mention that it represents a significant improvement over baseline, assuming our baseline to be a random classifier with $\frac{1}{37} = 3\%$ accuracy.

4.2 Embedding Similarity with CLIP

Both in terms of Evaluation Accuracy and Average Distance Error, the best performing Embedding Similarity model is the "Max Score per Label" Embedding Similarity. This model's accuracy (30.3%) represents a 1000% improvement over baseline, although in absolute terms it is not sufficiently good for deployment.

Max Score per Label likely performed better than Average Score per Label because images within a label represent diverse viewpoints and finding the single most similar image is more effective than averaging similarity across the label. Using All Data Vocabulary did not improve performance compared to Train Data Vocabulary, possibly because the additional test set data was too small to significantly change the "best candidate" images for each label.

The results of the experiments listed below are shown in Table 1. Confusion Matrices for each of these models can be found in the Appendix (Figure 3, Figure 4, Figure 5, Figure 5, Figure 7)

4.3 Foundation Models

Across all of our modeling approaches, the best performing model was the ResNet Deeper model, which utilized a ResNet18 for image embeddings and then our deeper classification head. This model had a top-1 accuracy of 46.49%, a top-5 accuracy of 63.78%, and a median error of only 29 feet. Additionally, this model's top-1 accuracy represented a 1450% improvement relative to the baseline [3].

The results of all of our foundation model experiments can be found in Table 2 in the Appendix.

5 Discussion

5.1 Key Findings

The paper explores the performance of various models in classifying locations within the Stata Center, presenting key insights into their strengths and weaknesses.

The best performing model for location classification was found to be the ResNet Deeper model, achieving a top-1 accuracy of 46.49%. This model outperformed other architectures such as AlexNet and VGG in terms of both accuracy and error metrics. While the custom-designed CNNs initially showed impressive results, they ultimately fell short in the true test set evaluation, highlighting challenges in generalization. On the other hand, the embedding similarity approach with CLIP showcased promising results, with the "Max Score per Label" method achieving an accuracy of 30.3%. Despite its lower accuracy compared to the ResNet model, the embedding similarity approach offers a unique perspective by leveraging semantic embeddings for classification [9] [3].

Overall, the ResNet Deeper model emerged as the most robust and reliable option for location classification within the Stata Center, demonstrating the efficacy of transfer learning and deep neural network architectures in addressing complex navigational challenges.

5.2 Interpretations

The findings suggest that deep learning models, particularly ResNet architectures, hold promise in accurately classifying locations within complex indoor environments like the Stata Center. The success of these models highlights the importance of leveraging advanced neural network architectures and transfer learning techniques to tackle real-world navigational challenges. By providing tailored directional guidance based on user orientation, the system enhances the navigational experience and improves overall user satisfaction.

5.3 Limitations

While the paper addresses key indoor navigation challenges within the Stata Center, notable limitations remain:

- The models utilize single-frame images for location classification, which may not fully capture the dynamic nature of indoor environments. Factors like lighting conditions, occlusions, and environmental changes could impact accuracy and reliability.
- The proposed system's applicability to indoor environments beyond the Stata Center is unclear. The unique architectural features and layout of the Stata Center may not be representative of other indoor spaces.
- Training and deploying deep learning models for indoor navigation could pose practical challenges in resource-constrained environments or mobile applications.

5.4 Future Work

To enhance the indoor navigation system's practicality, several key areas warrant further research. Firstly, developing a user-friendly interface integrating all models would streamline access for users, enhancing the system's usability. Secondly, refining the route optimization algorithm to provide comprehensive navigation routes using classical algorithms like Dijkstra's or A* would offer detailed guidance from start to endpoint. Accurate graphical representations of the Stata Center are crucial for this refinement. Additionally, expanding the dataset to include other floors would broaden the system's scope. Extending the route optimization model to handle between-floor directions could facilitate seamless navigation across different levels of the building, enhancing overall usability and versatility.

6 Conclusion

This paper introduces a novel system to address indoor navigation challenges, using MIT's Stata Center as a case study. Leveraging deep learning techniques like custom CNNs, embedding similarity with CLIP, and foundation models like ResNet, the system aims to classify user-provided images and provide directional guidance. Despite initial promising results with custom CNNs, challenges emerged during evaluation, including data leakage and lower accuracy on the true test set.

The ResNet Deeper model proved most robust, achieving 46.49% top-1 accuracy and a 63.78% top-5 accuracy, indicating potential for real-world deployment. The embedding similarity approach with CLIP also showed promise, albeit with lower accuracy. Challenges remain, including scalability, generalizability, and computational constraints.

Further research is needed to refine the interface, improve route optimization algorithms, and expand the dataset to multiple floors. Despite challenges, this work lays the groundwork for advancements in indoor navigation systems, offering insights into deep learning's efficacy in addressing complex navigational challenges.

7 Individual Contributions

We split work equally on this project. All three project members designed the methodology and took part in both data collection sessions at the Stata Center. Additionally, all members contributed equally to the creation of the final report and the final presentation. All members were involved in finding relevant literature for this project as well. We made sure to even divide the time we spoke during our presentation as evenly as possible. There were three large tasks/experiments in the methodology of our project: CNN/Embedding Similarity, Foundation models and Route Optimization. All three were key pieces in completing a final product we were proud of, and we believe met the expectations relating to the breadth and depth of our scope. Sara and Krish worked together on both the CNN and Embedding Similarity Models, as well the Route Optimization algorithm, while Hayden specialized on the constructing, training and testing the foundation models.

For the task of building out a CNN framework, as well as one for Embedding Similarity models, Sara and Krish worked side by side, employing a pair programming approach. They decided to employ this approach in which team members who are driving and who are navigating (no pun intended) are often associated with many advantages including enhanced code quality as knowledge is shared and faster problem solving when errors (which are detected at an increased rate) are detected. They began with data preprocessing to extract the image frames from all the videos they had collected as data points and split the dataset into training and testing sets. They then trained three different CNN models (low resolution images, minimal augmentation, and one convolutional layer with dropout; higher resolution images, more convolutional layers, and dropout, higher resolution images, more augmentation, more convolutional layers, dropout, and weight initialization) and saved the best-performing model. For evaluation, they computed distance errors to measure how well the model predicts spatial relationships between images and plotted confusion matrices to visualize classification performance. They also explored multiple approaches of using image embeddings (generated using CLIP from OpenAI) to classify images. As a final step they determined the accuracy and average distance error for each approach and compared it to the CNN model's performance, summarizing the results in a table.

When looking at the use of foundation models to yield better results, Hayden played the primary role through his work. He wrote code to extract embeddings from images using pre-trained deep learning models (ResNet-18, AlexNet, VGG-16) and then save them as CSV files. He defined functions that load pre-trained models without their classification layers, freezing their parameters, and also defined a function to extract embeddings from images using these models. After gathering these extracted embeddings for each model and each dataset split (train, validation, test), Hayden wrote code to build custom Torch models on top of embeddings extracted from foundation models. Using functions which the group defined together for calculating distances between predicted and actual labels, he trains and evaluates the model on the validation set. He recorded metrics such as loss, accuracy, and average distance and plotted histograms to visualize the error of the model.

The final task of route optimization was once again carried out by Sara and Krish working as a pair programming team. They wrote code to calculate the direction and distance from one point to another within a mapped area. More specifically, they initially defined functions to convert from pixels to feet and calculate the direction towards due north. They then created a function to extract coordinates from specific locations within the Stata Center and determine the direction a person should face and the distance to travel to reach a destination from a starting point. Finally, they conclude by providing examples of how to use these functions with specific locations and orientations.

References

[1] Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton van den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 2

[2] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. 1, 3

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. 3, 4

[4] Minyoung Hwang, Jaeyeon Jeong, Minsoo Kim, Yoonseon Oh, and Songhwai Oh. Meta-explore: Exploratory hierarchical vision-and-language navigation using scene object spectrum grounding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6683–6693, June 2023. 2

[5] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *Advances in Neural Information Processing Systems*, 25, 2012. 1, 3

[6] Anca Morar, Alin Moldoveanu, Irina Mocanu, Florica Moldoveanu, Ion Emilian Radoi, Victor Asavei,

Alexandru Gradinaru, and Alex Butean. A comprehensive survey of indoor localization methods based on computer vision. *Sensors*, 20(9), 2020. 2

[7] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. *arXiv preprint arXiv:2103.00020*, 2021. 2

[8] L. Ruotsalainen, A. Morrison, M. Mäkelä, J. Rantanen, and N. Sokolova. Improving computer vision-based perception for collaborative indoor navigation. *IEEE Sensors Journal*, 22(6):4816–4826, Mar 2022. 1

[9] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015. 3, 4

[10] R. Tapu, B. Mocanu, and T. Zaharia. A computer vision system that ensures the autonomous navigation of blind people. In *2013 E-Health and Bioengineering Conference (EHB)*, pages 1–4, 2013. 1

Appendix

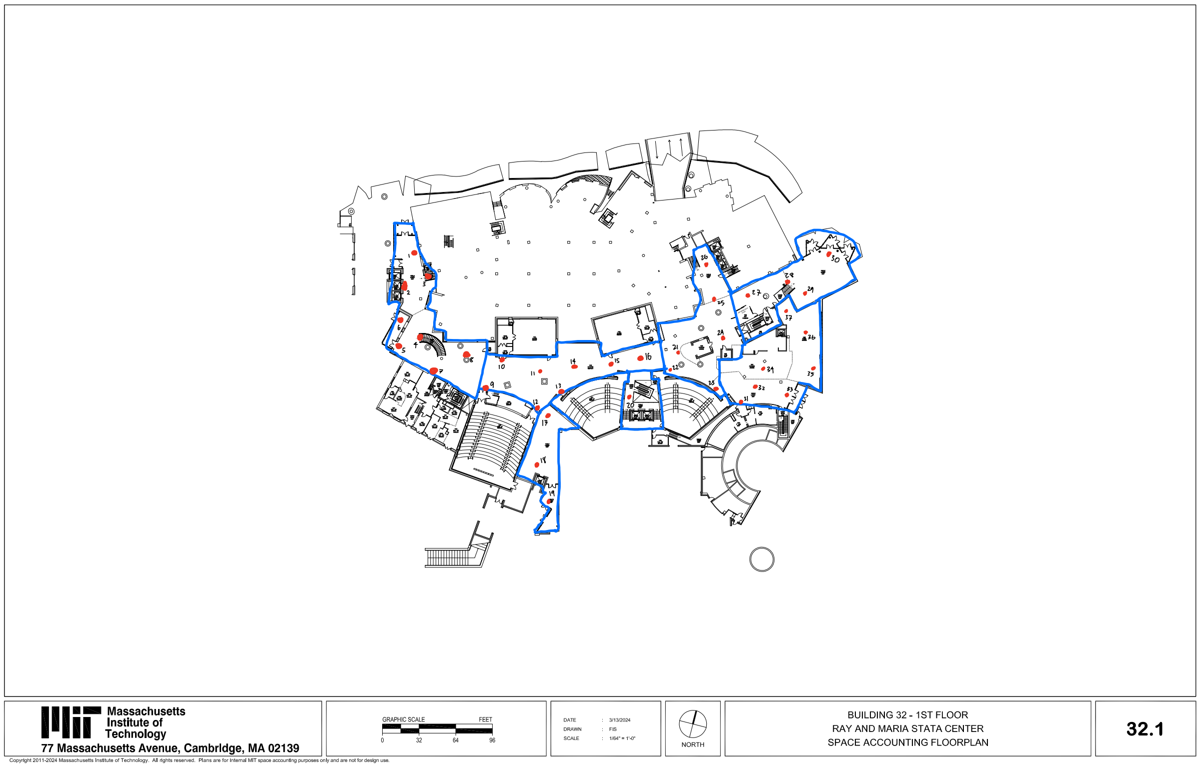

Massachusetts Institute of Technology
77 Massachusetts Avenue, Cambridge, MA 02139
GRAPHIC SCALE
FEET
0
32
64
96
DATE
DRAWN
SCALE
NORTH
BUILDING 32 - 1ST FLOOR
RAY AND MARIA STATA CENTER
SPACE ACCOUNTING FLOORPLAN
32.1

Figure 1: Floor 1 of Stata Center with Data Collection Points

Listing 1: Function to provide navigation instructions

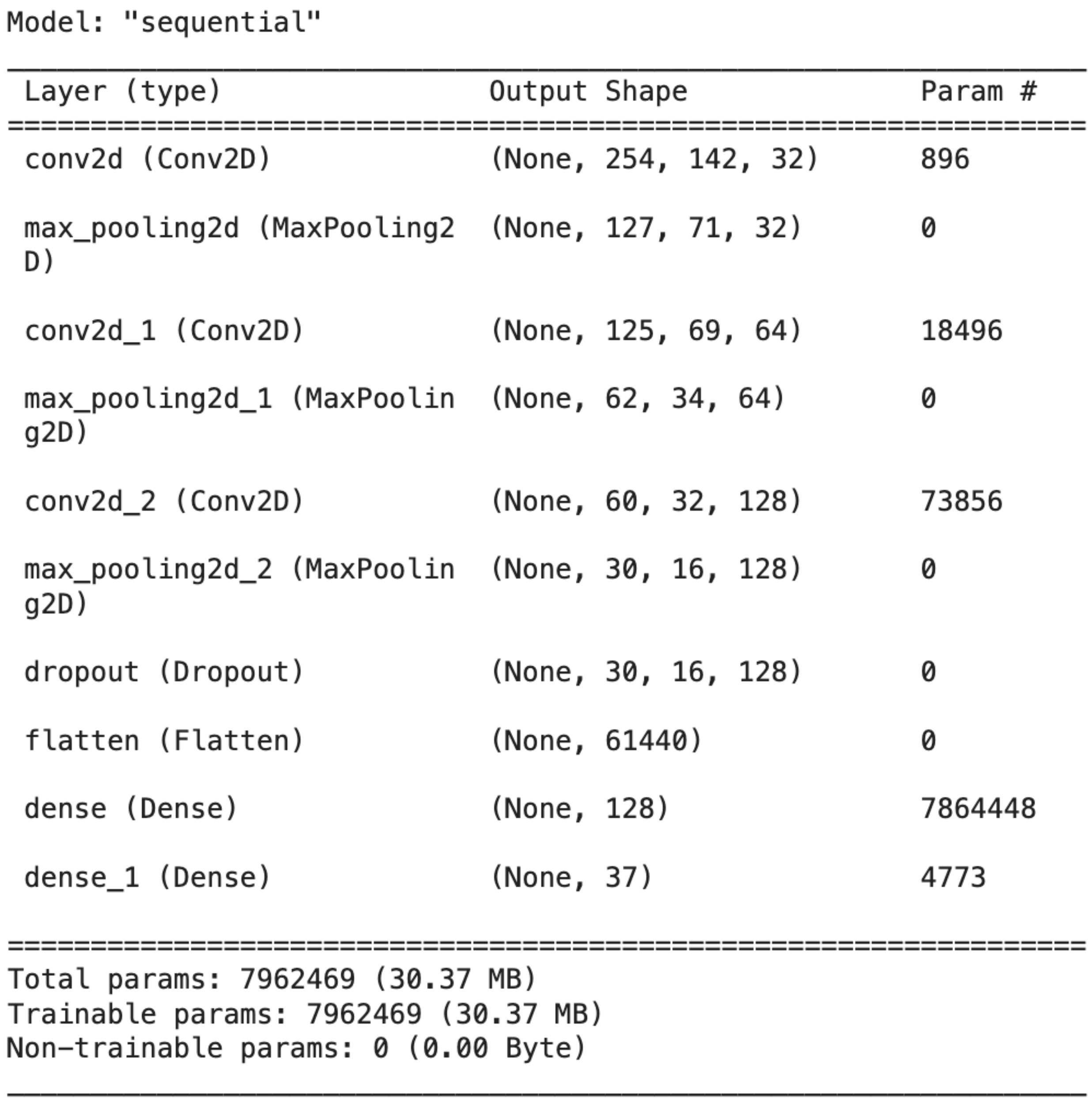
```
def which_way(start, end, orientation=-1):
    ...
    if orientation != -1:
        print(f"You must turn {turn} degrees to the {left_right} to face your destination")
        print(f"Alternatively, you can face {round_head} degrees {direction} using a compass")
        print(f"Your destination is {dist_in_feet} feet away from you")
```

Table 1: Performance Table of CNN and CLIP

Model	Top-1 Accuracy	Top-5 Accuracy	Median Error	Mean Error	Max Error
Best Performing CNN	14.05%	44.32%	204.98 ft	238.43 ft	744.16 ft
CLIP (Max Score and Train Data)	30.27%	56.22%	106.7 ft	194.88 ft	820.24 ft
CLIP (Max Score and All Data)	30.27%	56.22%	106.7 ft	194.88 ft	820.24 ft
CLIP (Average Score and Train Data)	12.97%	39.46%	281.97 ft	326.46 ft	820.24 ft
CLIP (Average Score and All Data)	12.97%	39.46%	281.97 ft	326.46 ft	820.24 ft

Table 2: Performance Table of Foundation Models

Model	Top-1 Accuracy	Top-5 Accuracy	Median Error	Mean Error	Max Error
AlexNet Simple	34.05%	49.73%	53 ft	88.64 ft	360 ft
AlexNet Deeper	35.68%	52.43%	46 ft	89.72 ft	360 ft
VGG Simple	31.89%	51.89%	46 ft	83.71 ft	344 ft
VGG Deeper	32.97%	52.43%	46 ft	85.60 ft	360 ft
ResNet Simple	40.54%	57.30%	35 ft	78.88 ft	379 ft
ResNet Deeper	46.49%	63.78%	29 ft	68.49 ft	387 ft



```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 254, 142, 32)      896

 max_pooling2d (MaxPooling2  (None, 127, 71, 32)       0
 D)

 conv2d_1 (Conv2D)           (None, 125, 69, 64)       18496

 max_pooling2d_1 (MaxPoolin  (None, 62, 34, 64)        0
 g2D)

 conv2d_2 (Conv2D)           (None, 60, 32, 128)       73856

 max_pooling2d_2 (MaxPoolin  (None, 30, 16, 128)       0
 g2D)

 dropout (Dropout)           (None, 30, 16, 128)       0

 flatten (Flatten)           (None, 61440)             0

 dense (Dense)               (None, 128)               7864448

 dense_1 (Dense)             (None, 37)                4773

=================================================================
Total params: 7962469 (30.37 MB)
Trainable params: 7962469 (30.37 MB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

Figure 2: Best Performing CNN Architecture

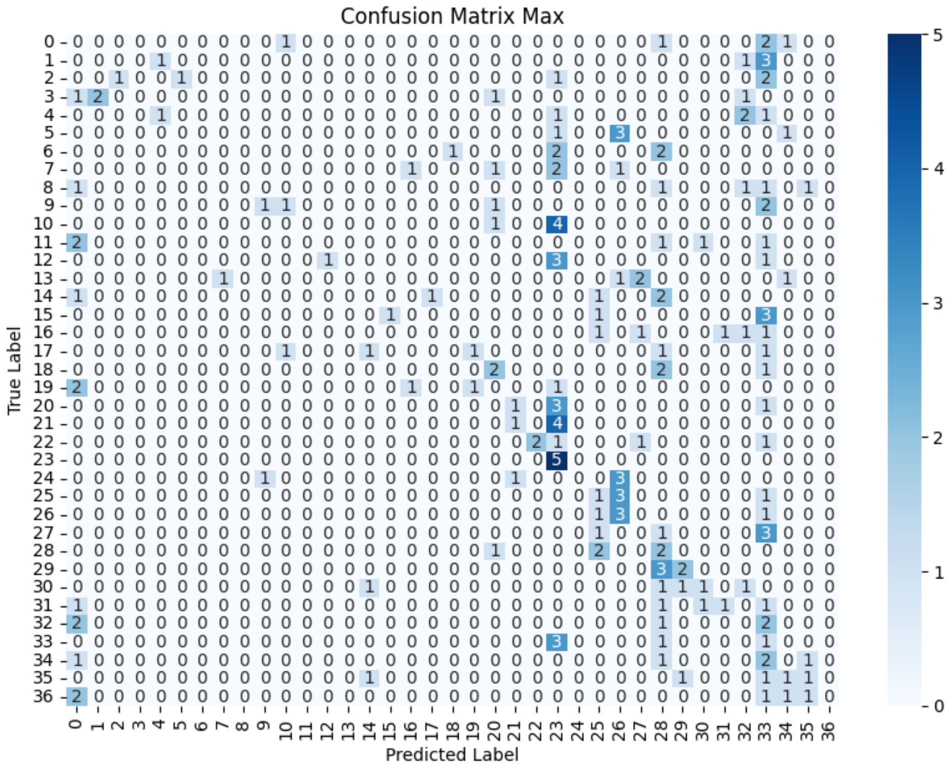


Figure 3: Confusion Matrix CNN

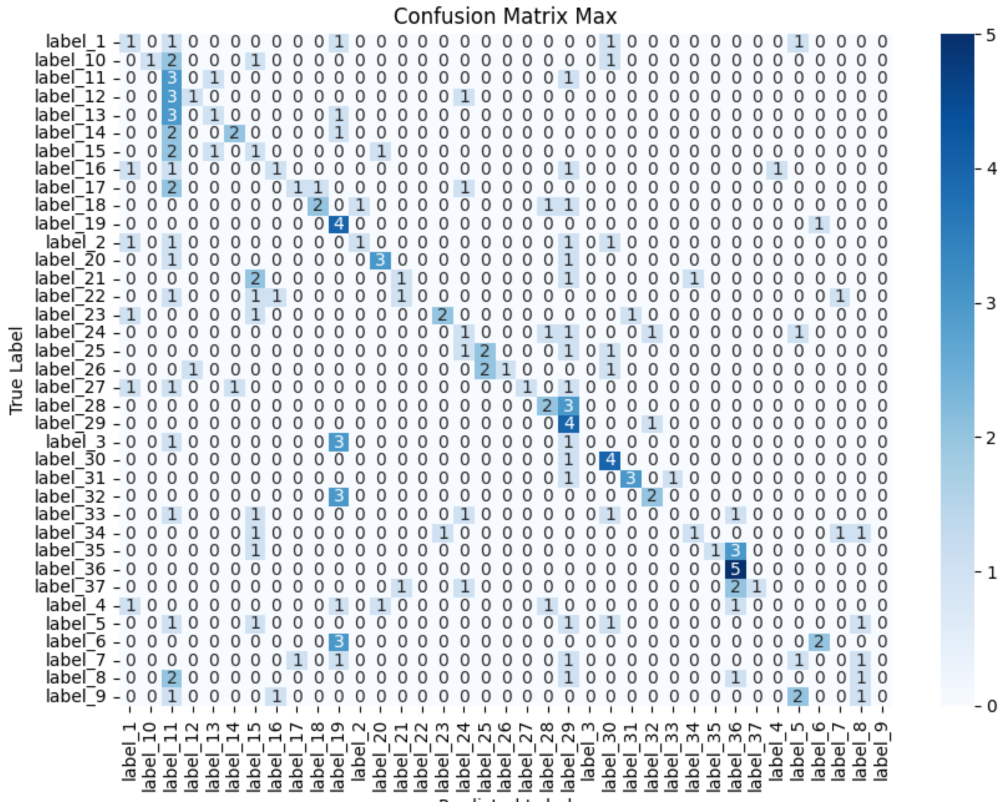


Figure 4: Confusion Matrix Max Score per Label, Train Data as Vocabulary

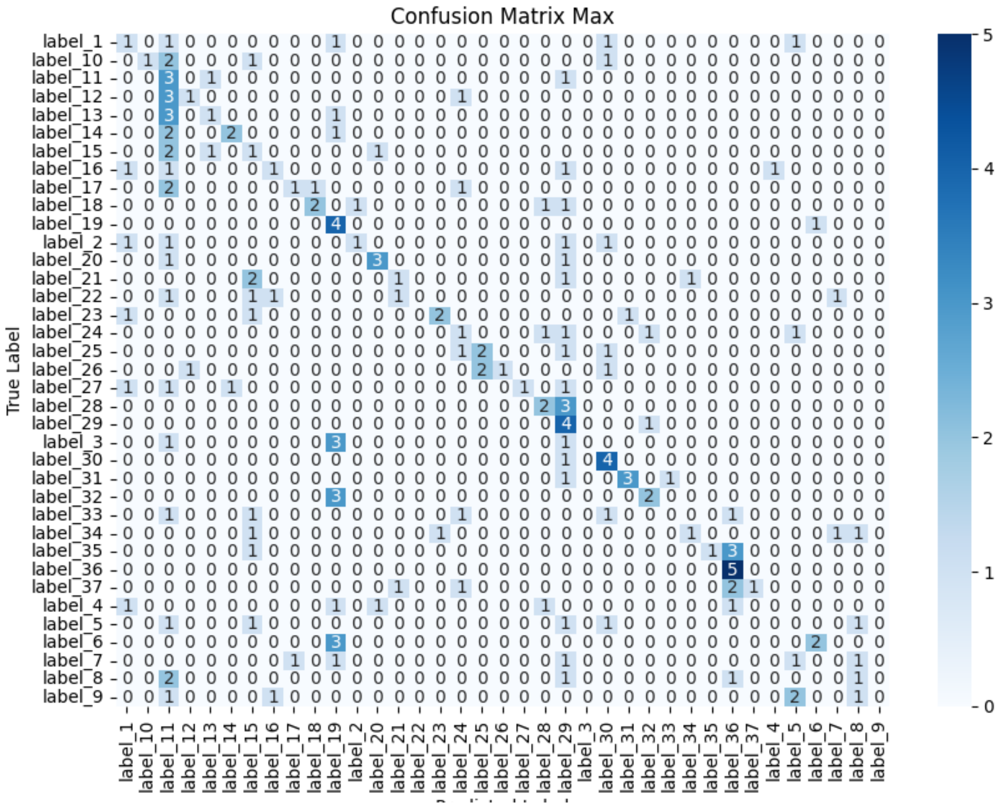


Figure 5: Confusion Matrix Max Score per Label, All Data as Vocabulary

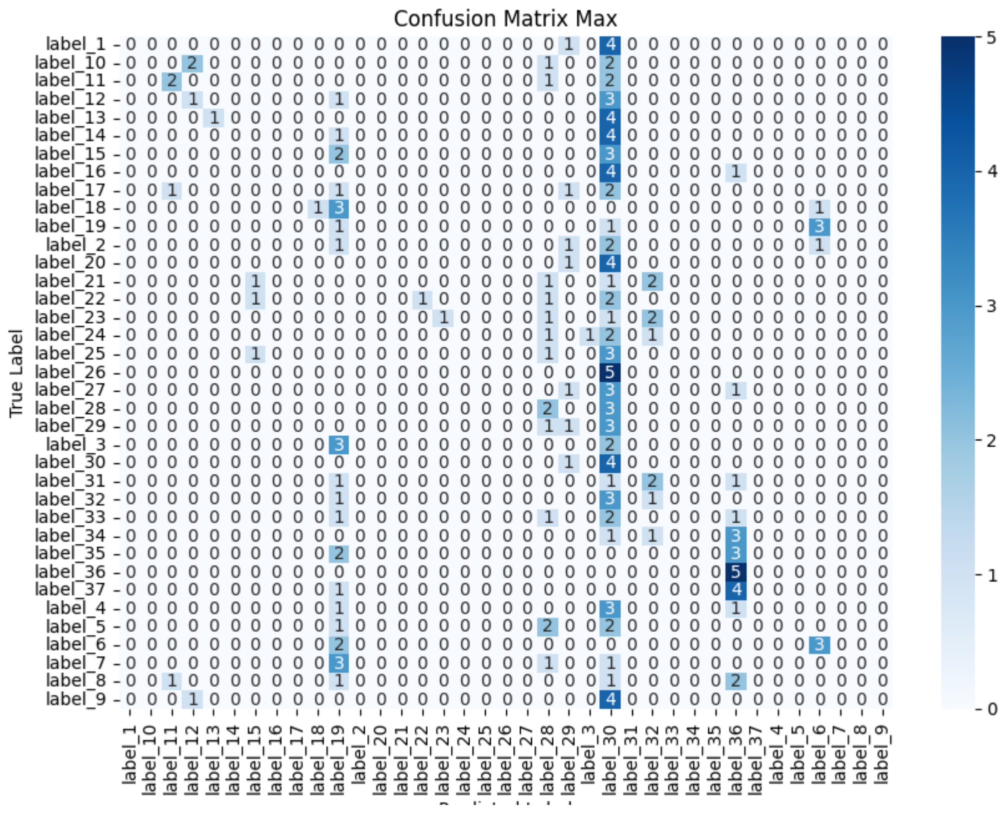


Figure 6: Confusion Matrix Average Score per Label, Train Data as Vocabulary

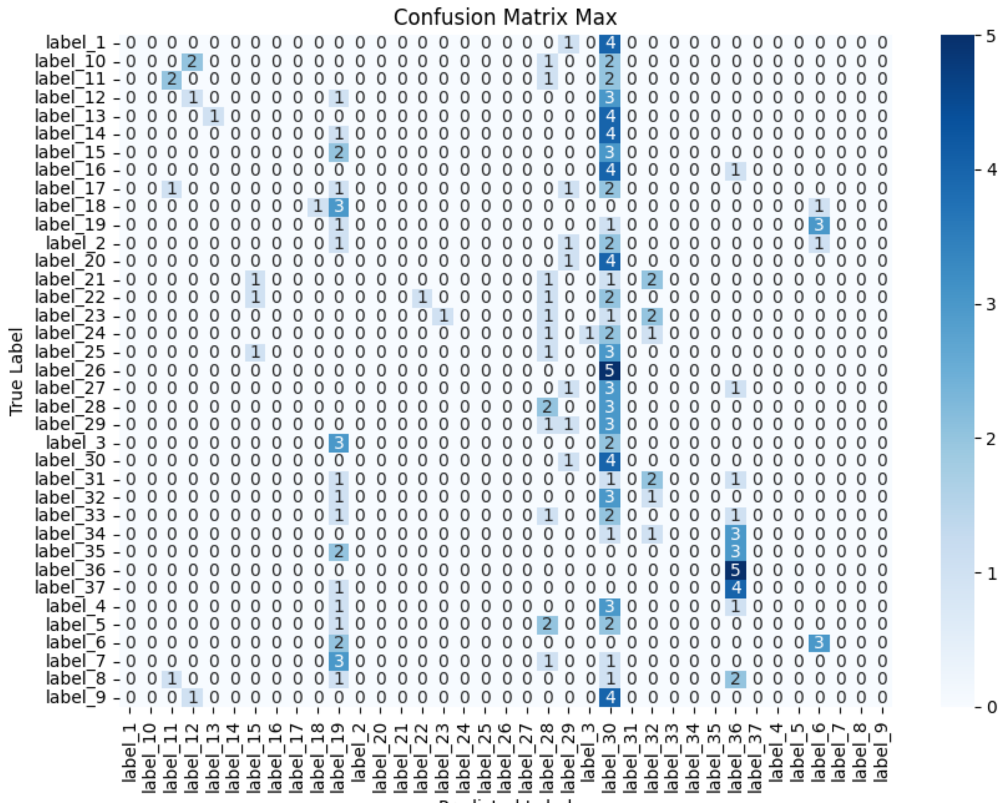


Figure 7: Confusion Matrix Average Score per Label, All Data as Vocabulary