Simplified GeoGuessr: Visual Place Recognition from Google Street View Images

Mia C. Onodera mco7@illinois.edu
Neil Ghose neilg4@illinois.edu
Himavanth Mahesh hmahesh2@illinois.edu

January 20, 2026

1 Introduction

Inspired by the GeoGuessr game, this project investigates city-level visual place recognition using only image information. Specifically, we study the problem of recognizing the city of origin from a single Google Street View image across nine geographically and culturally diverse cities. We evaluate and compare three approaches: a fine tuned GSV based deep visual place recognition model, a vision-based transformer model, and a hybrid model, GeoSceneNet. GeoSceneNet combines learned convolutional image features with interpretable scene descriptors, including texture, edge statistics, horizon position, and sky ratio, with the goal of capturing both high-level semantic information and lower-level scene structure relevant to geographic identification.

2 Details of the Approach

This project centers on a comparative study of three different approaches to visual place recognition: a GSV-based model adapted from an existing open-source implementation, a vision-based transformer model, and our hybrid model, GeoSceneNet, which fuses computer-vision based scene descriptors with CNN image features. All three models were trained and evaluated on a dataset of Google Street View images that we collected and organized across nine cities to ensure consistent and comparable inputs for each approach. In this section, we describe the design, preprocessing steps, training procedures, and motivations behind each model in order to highlight their differences and provide context for the performance comparison presented later in the Results section.

2.1 Dataset

The Images were collected using a custom Python script that queries the Google Street View Static API [1]. For each city, we defined a latitude and longitude bounding box covering the central urban area and uniformly sampled geographic coordinates within this region. At each sampled location, the script requested a forward facing street level image using a fixed field of view and resolution. All requests were authenticated using a Google Cloud API key, and images were downloaded and stored locally for preprocessing and model training.

Region	City	Country	Images
East Asia	Tokyo	Japan	784
Southeast Asia	Hanoi	Vietnam	786
North America	Chicago	USA	791
South America	Bogota	Colombia	792
Southern Europe	Rome	Italy	789
South Asia	Delhi	India	653
Middle East	Dubai	UAE	753
Sub-Saharan Africa	Lagos	Nigeria	647
Western Europe	London	UK	794
Total			**6,789**

Table 1: Overview of the Google Street View dataset used in this project. Images were collected using the Google Street View Static API with approximately equal samples per city.

2.2 GSV-Based Model

The GSV model [2] is based on ResNet-50 with additional fully convolutional aggregation layer known as Conv-AP. This allows the model to create embedding vectors used for visual place recognition. We found that using the default pretrained build on this model was not at all accurate for our dataset, predicting in an approximately uniform distribution of cities for all images in out dataset, when using nearest neighbors for prediction. This is not very surprising, as many of the cities in our dataset were not in the original GSV training set, so inaccuracies are expected.

In order to increase GSV's accuracy for our dataset, we fine-tuned the original model with our own data. First, we started with the same ResNet-50 Backbone and Conv-AP aggregation layers in the original model. We then hooked up the output embeddings into a single-layer linear classifier network. This was then trained using our data and the corresponding city ID using cross-entropy loss and the Adam optimizer. In order to improve accuracy, we trained over multiple epochs with random augmentations each iterations, applying a random crop, color jitter, and gaussian blur to each image.

Table 2: Validation accuracy across training epochs for GSV-Based Model.

Epoch	Train Accuracy	Val Accuracy
1	0.3580	0.5574
2	0.6132	0.6907
3	0.7192	0.7607
4	0.7883	0.7982
5	0.8284	0.8159
6	0.8713	0.8270
7	0.8990	0.8343
8	0.9192	**0.8387**
9	0.9322	0.8351
10	0.9497	0.8336

2.3 Vision-Based Transformer Model

The vision based transformer model serves as a fully learned baseline that relies exclusively on image content, without incorporating handcrafted scene descriptors or explicit geometric priors. The primary motivation behind this model is to evaluate how effectively a modern pretrained vision network can learn city specific visual cues directly from raw Google Street View images, and to establish a strong reference point against which the other approaches can be compared.

Model architecture: We adopt a pretrained CLIP vision model as the backbone and use its image encoder for feature extraction. We attach a task-specific linear classifier that maps the learned image embeddings to the nine city classes in our dataset. By fine-tuning the entire model on our data, the network adapts its representations to capture city-level visual patterns such as architectural style, road structure, vegetation, signage, and skyline characteristics. This design allows the model to learn discriminative features end-to-end without imposing constraints on which cues are considered important.

Training procedure: Input images are resized and normalized according to the pretrained model's requirements. During training, the model is optimized using cross entropy loss, and standard data augmentation techniques including random resized crops, color jitter, and gaussian blur are applied to improve robustness. The dataset is randomly shuffled and split into training and test sets (80/20 split). The model is trained for a fixed number of epochs, and the final model state is evaluated on the held-out test set.

Design rationale: Unlike GeoSceneNet, which explicitly encodes interpretable scene level features such as horizon lines and edge density, our VLM approach makes minimal assumptions about the visual structure of the problem. By fine tuning a pretrained CLIP encoder end to end, the model freely learns relevant visual cues directly from the data. This allows it to capture complex, fine grained distinctions between cities such as specific architectural styles, vegetation patterns, and road markings that are difficult to define manually. This flexibility enables the model to learn richer representations, leading to substantially higher classification accuracy than methods limited by fixed, hand crafted feature definitions

Table 3: Validation accuracy across training epochs for the vision based transformer model.

Epoch	Train Accuracy	Val Accuracy
1	0.7130	0.7974
2	0.8975	0.8592
3	0.9560	0.8316
4	0.9656	0.8776
5	0.9785	0.8632
6	0.9735	0.8395
7	0.9745	**0.8974**
8	0.9778	0.8684
9	0.9739	0.8816
10	0.9831	0.8579

2.4 Hybrid Model: GeoSceneNet

GeoSceneNet is our hybrid model that combines learned deep features with scene descriptors. The goal is to capture both high-level semantic information from the image and the lower-level geometric and appearance cues that correlate with certain cities.

Scene descriptors. For each Google Street View image, we first resize the image to 256×256 and convert a copy to grayscale. We then compute a 10-dimensional scene feature vector $s \in \mathbb{R}^{10}$:

- **Texture:** variance of the Laplacian response. High texture may denote bricks, walls, trees, crowded streets.Low texture may denote skies, water, concrete floors.
- **Edge orientation ratio:** ratio between horizontal and vertical Sobel responses. Cities that have boxy skyscrapers would have more vertical edges compared to small quaint houses.
- **Color and sky statistics:** mean and standard deviation of the hue (color), saturation (color intensity), value (brightness), and sky ratio. Dubai and Lagos may have more open skies, whereas dense downtown Chicago likely has almost no visible sky.
- **Horizon position:** identify a rough horizon line using the vertical Sobel. Scenes with tall buildings push the horizon down whereas open landscapes push it up.
- **Keypoint density:** number of interesting keypoints (corners, edges, and textures) per pixel, detected using the ORB algorithm on OpenCV. This feature captures the structural complexity of the scene.
- **Edge density:** fraction of edge pixels returned by Canny. A high edge density indicates that the scene contains many sharp boundaries, textures, and structural elements.

These values are concatenated into a single vector

$$s = [\text{texture}, \text{edge\_ratio}, h_{\text{mean}}, s_{\text{mean}}, v_{\text{mean}}, v_{\text{std}}, \text{horizon}, \text{kp\_density}, \text{edge\_density}, \text{sky\_ratio}]^\top.$$

texture – image sharpness
edge_ratio – horizontal vs. vertical edge ratio
h_mean – average hue
s_mean – average saturation
v_mean – average brightness
v_std – brightness variation
horizon – estimated horizon position
kp_density – keypoint density
edge_density – amount of edges
sky_ratio – proportion of sky

CNN image features. In parallel, we extract deep image features from the RGB image using a ResNet-18 backbone pretrained on ImageNet [4]. We remove the final classification layer because it is trained to predict the 1000 ImageNet classes, which are unrelated to our task [5]. By discarding this layer, we use the 512-dimensional output of the preceding layer as a general feature vector that captures useful visual information for city classification.

Feature fusion and classification head. GeoSceneNet concatenates the CNN features and the scene descriptors into a single vector and feeds the result into a two-layer classifier:

$$z = [f_{\text{cnn}}(I);\, s] \in \mathbb{R}^{d+10},$$
$$h = \sigma(W_1 z + b_1),$$
$$\hat{y} = \text{softmax}(W_2 h + b_2),$$

where W_1 and W_2 are learned weight matrices, σ is a ReLU activation, and $\hat{y}$ is the predicted probability distribution over city classes. In code, this corresponds to a two-layer MLP with a 256-dimensional hidden layer and dropout (0.5) between layers.

Training procedure. We train GeoSceneNet using cross-entropy loss and the Adam optimizer with a learning rate of 10^{-4}, weight decay of 10^{-4}, a batch size of 16, and 10 training epochs. To improve robustness, training images are augmented using random resized crops, color jitter, and occasional Gaussian blur. Validation and test images use a fixed preprocessing pipeline consisting of resizing and center cropping. During training, we track validation accuracy and save the model checkpoint with the highest performance.

Table 4: Validation accuracy across training epochs for GeoSceneNet.

Epoch	Train Accuracy	Val Accuracy
1	0.1326	0.4794
2	0.2219	0.5508
3	0.3833	0.6576
4	0.5566	0.7371
5	0.7078	0.7769
6	0.7935	0.8012
7	0.8529	0.7997
8	0.8797	0.8004
9	0.8957	**0.8093**
10	0.9143	0.8085

Intermediate outputs. GeoSceneNet shows clear and consistent learning behavior across epochs. The model is trained on 4,073 images, with 1,358 images each for validation and testing. During training, we observe a rapid decrease in loss and a steady increase in accuracy across the first 8–10 epochs. Validation accuracy rises from 0.48 in the first epoch to a peak of 0.8093 by epoch 9, after which the model begins to stabilize. This indicates that the model converges around epoch 10. We monitored these curves to ensure that training was stable and to detect any signs of overfitting; while the training accuracy continues to increase, the validation accuracy plateaus, suggesting the model has reached its optimal capacity under our chosen hyperparameters.

3 Results

3.1 GSV-Based Model

We found the GSV-based model to achieve very strong performance over our dataset, with a test accuracy of **83.36%**. The model converges around epoch 8 and remains stable afterward. Table 5 summarizes the per-class precision, recall, and F1-scores.

Table 5: Per-class precision, recall, and F1-score for GSV-Based Model on the test set.

City	**Precision**	**Recall**	**F1**
Bogota	0.83	0.83	0.83
Chicago	0.80	0.73	0.77
Delhi	0.93	0.88	0.90
Dubai	0.85	0.91	0.88
Hanoi	0.89	0.89	0.89
Lagos	0.92	0.93	0.92
London	0.74	0.76	0.75
Rome	0.80	0.80	0.80
Tokyo	0.78	0.80	0.79
Overall Acc.	**0.8336**		

The model achieves strongest performance on the cities Delhi, Lagos, and Hanoi, likely because of their distinct characteristics, especially compared to other cities in our model. The worst performing cities are London, Tokyo, and Chicago, all very urban areas that tend to be confused with one another, as shown by the confusion matrix below.

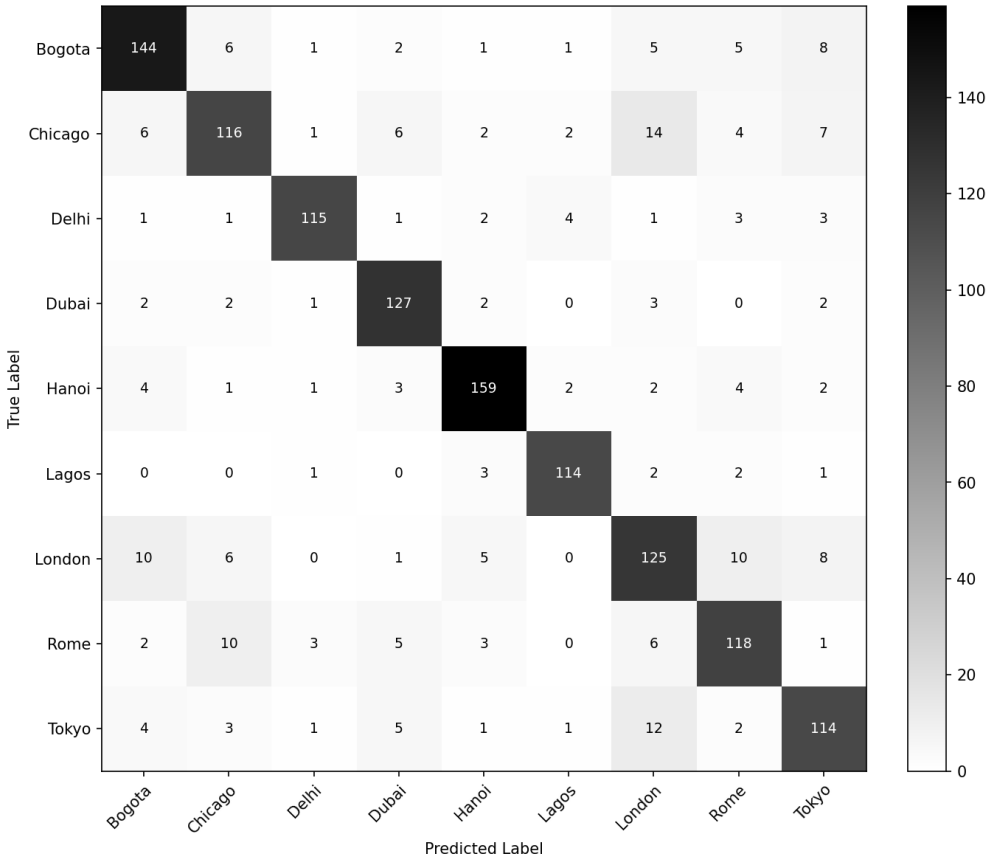

Figure 1: Confusion matrix for GSV-Based Model.

The performance of this model can likely be improved by increasing the dimensionality of the embedding vector prior to classification, as it would allow for greater detail in city representation, capturing more of what makes each city unique. This would likely also require more images for training, and more epochs to achieve greater accuracy.

3.2 Vision Based Transformer Model

We evaluate a Vision Based Transformer Model for the 9 class city classification task using a supervised training setup. While the full dataset contains over 6.9k images, the VLM is trained on a subset of approximately 3.9k images, with the remaining samples reserved for evaluation. Despite this reduced training set, the VLM achieves strong performance, obtaining an overall test accuracy of **97.13%** on 3,799 test images. Table 6 summarizes the per class precision, recall, and F1 scores.

Table 6: Per-class precision, recall, and F1-score for the VLM-based model on the test set.

City	Precision	Recall	F1
Bogota	0.9769	0.9592	0.9680
Chicago	0.9944	0.9249	0.9584
Delhi	0.9946	0.9840	0.9893
Dubai	0.9167	0.9954	0.9544
Hanoi	0.9978	0.9868	0.9922
Lagos	0.9896	0.9948	0.9922
London	0.9378	0.9569	0.9473
Rome	0.9822	0.9866	0.9844
Tokyo	0.9656	0.9525	0.9590
Overall Acc.		**0.9713**	

Across all classes, the VLM demonstrates consistently high precision and recall. Performance is strongest for Hanoi, Lagos, and Delhi, each achieving F1 scores above 0.98, indicating that the model effectively captures distinctive visual cues such as vegetation patterns, architectural styles, and street-level scene structure. Rome also exhibits strong performance (F1 = 0.98), suggesting robust recognition of historical and architectural features.

Slightly lower recall is observed for Chicago (0.92) and Tokyo (0.95), indicating that a small fraction of true images from these cities are misclassified as other urban centers. In contrast, Dubai achieves near perfect recall (0.995) but comparatively lower precision (0.92), suggesting occasional over prediction of Dubai when visually similar scenes from other cities are present. Overall, the precision recall balance remains favorable across all classes.

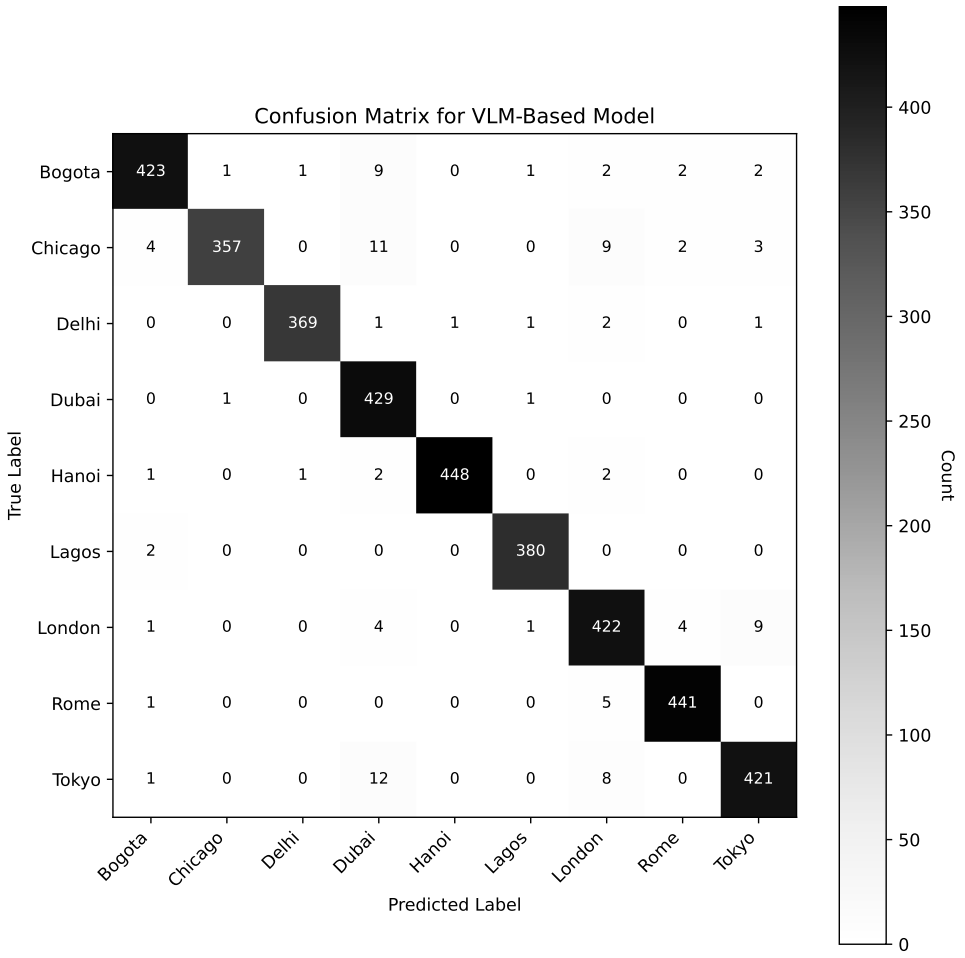

Figure 2: Confusion matrix for VLM Based Model.

To further examine error patterns, Figure 2 presents the confusion matrix for the VLM based model. The matrix is strongly diagonal dominant, confirming that most predictions are correct. The most common confusions occur among London, Chicago, and Tokyo, which share dense urban layouts, similar road geometry, and comparable color and texture statistics. These ambiguities are consistent with prior observations in scene based city recognition, where visually similar metropolitan environments are challenging to distinguish at the street level.

Importantly, misclassification rates remain low even for these challenging city pairs, aligning with the high per class F1 scores reported in Table 6.

Compared to the GSV based and hybrid models, the VLM benefits from its ability to leverage high-level semantic representations learned from large-scale vision language pretraining. Even when trained on a reduced subset of the available data, the model maintains strong generalization performance, suggesting that pretrained multimodal features provide a powerful prior for geographic scene understanding. These results highlight the effectiveness of VLMs as a strong baseline and potential upper bound for city level image classification tasks.

3.3 Hybrid Model: GeoSceneNet

GeoSceneNet achieves strong performance on the 9-class city classification task, obtaining a test accuracy of **76.29%**. The model converges around epochs 9–10 and remains stable afterward. Table 7 summarizes the per-class precision, recall, and F1-scores. Performance is highest for Delhi, Lagos, and Hanoi, which exhibit more distinctive visual cues (e.g., skyline, vegetation patterns). Lower scores appear in London and Tokyo, which the model often confuses, consistent with their architectural and street-layout similarities.

Table 7: Per-class precision, recall, and F1-score for GeoSceneNet on the test set.

City	**Precision**	**Recall**	**F1**
Bogota	0.86	0.71	0.78
Chicago	0.72	0.72	0.72
Delhi	0.89	0.90	0.89
Dubai	0.70	0.87	0.78
Hanoi	0.86	0.80	0.83
Lagos	0.82	0.89	0.85
London	0.67	0.64	0.65
Rome	0.73	0.70	0.72
Tokyo	0.68	0.69	0.69
Overall Acc.	**0.7629**		

To better understand error patterns, we analyze the confusion matrix in Fig. 3. The most common confusions occur between:

- **London** $\leftrightarrow$ **Tokyo**: dense urban scenes, similar road geometry, and comparable color/texture statistics.
- **Bogota** $\rightarrow$ **Rome**: similar building tones and vegetation ratios in certain neighborhoods.
- **Chicago** $\rightarrow$ **Dubai**: misclassifications mainly in images dominated by roads rather than skyline.

These failure cases suggest that while deep CNN features capture high-level semantics, the scene descriptors (sky ratio, edge density, horizon height) provide complementary cues. When used alone, the handcrafted scene-descriptor classifier reached only moderate accuracy (approximately in the 60% range), and the pure CNN model struggled with several ambiguous city pairs. The hybrid fusion model outperformed both, showing that combining global scene cues with deep visual features improves robustness.

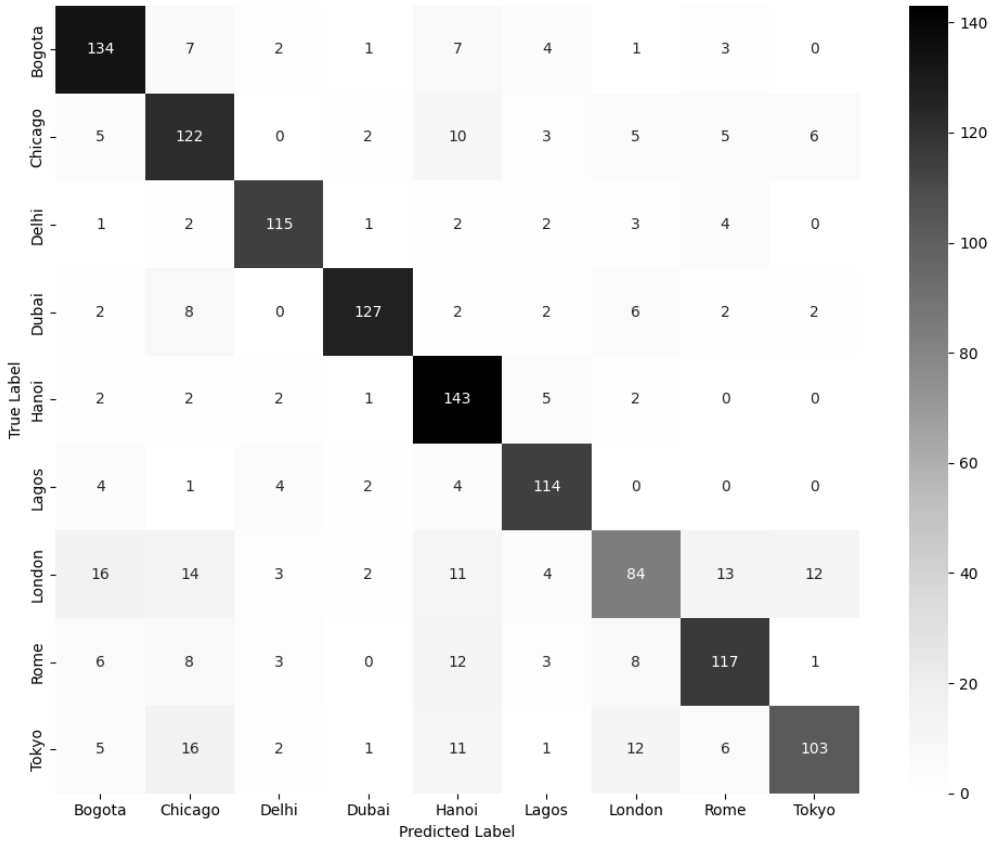

Figure 3: Confusion matrix for GeoSceneNet.

3.4 Geoguessr Interface

In order to evaluate the effectiveness of our models in practice, we built a small web application to allow users to test these models with any image. Here is a small demonstration of this tool used on a custom GeoGuessr map that contains the nine cities used in the dataset.

Please visit our GitHub Repository for details on how to run our web application and use our models more broadly.

4 Discussion and Conclusions

Across the three models we evaluated—the GSV-based baseline, the vision-based transformer model (VLM), and our hybrid GeoSceneNet—each exhibited different strengths in the task of city-level visual place recognition.

Table 8: Overall accuracy across all models.

Model	**Accuracy (%)**
Vision-based Model (VLM)	99.19
GSV-based Model	81.66
Hybrid GeoSceneNet	78.0

Among the three models, the VLM was the strongest by a large margin, reaching around 99% accuracy.

Table 9: Per-city accuracy (%) for all models.

City	VLM	GSV	GeoSceneNet
Bogota	98.00	85.52	77.00
Chicago	99.40	67.61	77.00
Delhi	100.0	88.15	88.00
Dubai	100.0	93.23	84.00
Hanoi	99.39	89.81	91.00
Lagos	100.0	84.73	88.00
London	98.79	67.08	53.00
Rome	100.0	79.05	74.00
Tokyo	97.40	84.88	66.00

It was able to learn very detailed visual patterns across cities. It showed how well a modern transformer architecture can perform when trained directly on our dataset.

This result highlights the effectiveness of large scale pretrained visual representations for city level recognition. Unlike the GSV based and hybrid models, which rely on hand-designed or mid-level scene descriptors, the VLM benefits from transformer features that capture a wide range of semantic and contextual cues, including architectural styles, skyline composition, road structure, and vegetation patterns. Even when trained on a subset of the available data, the VLM demonstrates strong generalization, suggesting that pretrained visual features provide a powerful prior for geographic scene understanding.

At the same time, the unusually high accuracy achieved by the VLM warrants careful consideration of potential data leakage or prior exposure. Because the images were collected from publicly available online sources, it is possible that visually similar images or closely related views from the same locations were included in the large scale datasets used during the original pretraining of the VLM. While we ensure that training and test splits are disjoint within our dataset, we cannot fully rule out the possibility that the pretrained model has encountered related imagery during pretraining, particularly given the release dates of modern vision language models and the long standing availability of street level imagery. This highlights an important limitation when evaluating powerful pretrained models on web sourced data: exceptionally high performance may reflect not only strong generalization but also implicit memorization or familiarity with the visual domain.

The GSV-based model performed in the middle, reaching 81.66%. It worked well for cities with very distinctive visual signatures—such as Dubai, which has a recognizable skyline and strong architectural cues—but it struggled on cities like Chicago, where the visual appearance varies more widely. GeoSceneNet showed a similar pattern. Although it incorporated additional scene descriptors, it ultimately performed the worst overall (macro F1 = 0.78). Both models tended to fail on cities where appearance changes significantly across neighborhoods or where distinguishing features are not consistently present.

While pretrained or manually derived features can capture broad scene-level signals, they are not expressive enough to handle the fine-grained distinctions required for this task. In contrast, the transformers in the VLM model can learn much richer and more adaptable representations directly from the data. Even though GeoSceneNet had lower accuracy, building it helped us identify which visual cues matter most and how these preprocessed features interact with CNN features during classification.

Dataset limitations. Although our dataset spans nine geographically and culturally distinct cities, it remains limited in diversity. Each city is represented by a constrained set of street-view images, and many viewpoints share similar compositions (e.g., street-level forward-facing scenes). This reduces exposure to visual variability such as different seasons, times of day, traffic density, or rare landmarks. Since the dataset is restricted to city-level labels, the models may also learn spurious correlations—such as vegetation type, sky color, or road markings— that do not generalize beyond the sampled regions. These limitations mean the models may perform well within our dataset but may not generalize to unseen cities or atypical viewpoints.

Future directions. Several extensions could strengthen the robustness and generality of our system. First, expanding the dataset to include more cities, more varied viewpoints, multiple times of day, and seasonal changes would reduce overfitting to specific visual cues. Second, incorporating multi-view or temporal information (e.g., sequences of GSV frames) could help the model build a more stable spatial representation. Third, transformer-based architectures or multimodal approaches that integrate textual cues (street signs, detected language, OCR metadata) may improve fine-grained recognition. Finally, experimenting with contrastive or self-supervised pretraining could yield models that generalize more effectively to new city environments.

In addition, future work could further disentangle the effects of true generalization versus prior exposure by evaluating models on imagery collected after the release dates of pretrained VLMs or on geographically distinct cities that are unlikely to be present in large-scale web datasets. Alternative validation strategies, such as temporally constrained datasets, synthetic viewpoints, or cross city generalization experiments, could provide a more rigorous assessment of robustness. These directions would help clarify the extent to which high performance reflects transferable visual understanding rather than implicit familiarity with web-sourced imagery.

5 Statement of Individual Contributions

- Mia Celena Onodera: Mia led the development of the computer-vision based deep-learning hybrid model, GeoSceneNet. She designed the scene descriptor features used in the model, handled preprocessing, and built the baseline classifier from scratch. She then trained and evaluated GeoSceneNet across all splits and refined its performance through failure-case analysis. In addition, she collected and helped develop the code used to collect and scrape Google Street View images.
- Neil Ghose: Neil developed the interface used to load, organize, and evaluate the models, as well as the web application to provide a user-facing method of evaluation. He led the development of the GSV-based model, where he adapted the open-source GSV architecture to our dataset, trained and tested the model, and refined its performance. In addition, he collected and helped develop the code used to scrape Google Street View images.
- Himavanth Mahesh: Himavanth led the development of the vision-based transformer model. He tuned hyper parameters, analyzed model performance, and contributed to improvements on the GeoSceneNet model. In addition, he collected Google Street View images for the training dataset.

Acknowledgments

We would like to thank Professor Forsyth and the course staff for their guidance and feedback throughout this project. In addition, we acknowledge the use of open-source libraries such as PyTorch, OpenCV, and scikit-learn, as well as AI-assisted coding tools such as ChatGPT for help with debugging, code generation, and explanation of concepts during development. All modeling decisions, experiments, and final implementations were designed and verified by the authors.

References

[1] Google. Google Street View API At:`https://developers.google.com/maps/documentation/streetviewoverview`, accessed December 2025.

[2] A. Ali-bey, B. Chaib-draa, P. Giguère. *GSV-Cities: Toward appropriate supervised visual place recognition.* Neurocomputing, 2022.

[3] 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.* CoRR, 2021.

[4] He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. *Deep Residual Learning for Image Recognition*. CVPR, 2016.

[5] Deng, Jia, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. *ImageNet: A Large-Scale Hierarchical Image Database*. CVPR, 2009.