Alan Tran, Caleb Wolf
Professor Huth
CS 342
28 April 2025

CNN + ViT for Geolocation Task

Background

The primary goal of this project is to build a neural network capable of learning a geolocation classification task where the model can predict which geographic grid cell a street-level image was taken from. This project was inspired by Geoguessr, an online game where users attempt to guess the location of a Google Street View image by placing a pin on a virtual map. Our implementation of the grid system is based on the paper "PlaNet - Photo Geolocation with Convolutional Neural Networks," where the authors used the Google S2 Geometry library to discretize the Earth's surface into grid cells (Weyand et al., 2016). Our model of choice was a hybrid architecture that combines the strengths of a convolutional neural network (CNN) for local feature extraction and a vision transformer (ViT) for capturing global spatial relationships.

Dataset and Data Preprocessing

Building on the approach of the PlaNet article (Weyand et al.), we treat geolocation as a classification problem rather than a regression one. Specifically, we split the globe into non-overlapping geographical cells that make up the target classes. This approach means the model output is a probability distribution over the surface of the globe (Weyand et al.). By contrast, directly regressing latitude and longitude cannot easily express model confidence across the surface of the Earth.

To define our cells, we leverage Google's S2 geometry library, which projects the sphere onto an enclosing cube and then recursively subdivides each face via a quadtree (Weyand et al.). S2 cells at a fixed level are roughly equal in area and avoid the severe distortions that latitude-longitude grids suffer near the poles. However, our training data exhibit strong density imbalances (urban street scenes far outnumber rural ones), so equal-area tiling would result in massive class imbalances. Instead, we adopt an adaptive partitioning: any S2 cell containing more than $t_1 = 100$ images is split into its four children, and this continues until every cell has fewer than 100 samples. Finally, we prune any leaf cell with fewer than $t_2 = 20$ images. By choosing a relatively low t_1 and a relatively high t_2, we aimed for a balance between spatial granularity in dense regions and a manageable number of classes overall.

Because of compute and memory constraints, we restrict our experiments to the OSV-Mini-129k dataset, a 129,000-image subset of OpenStreetView-5M focused on 10 Western U.S. states (josht000, 2025). OSV-Mini-129k offers street-level imagery whose geographic distribution closely tracks population density, while remaining small enough to fit our hardware.

The high t_2 cutoff combined with sparse road coverage in rural areas means some low-density regions are omitted. After adaptive partitioning, we ended with 2,199 S2 cells as our target classes, with large cells spanning rural areas and much smaller cells in urban neighborhoods.

Model

We chose to use a hybrid CNN + ViT architecture to leverage the strengths of both models for the geographic localization, specifically predicting where a photo was taken. The first component of the model is a pretrained ResNet50 as the CNN backbone. The goal of this layer is to learn to extract local features such as local textures, edges, and low-level patterns. For example, CNNs are great at capturing small details in different road textures, building materials like red brick, and tree leaf patterns. We set features_only=True so that the CNN outputs intermediate feature maps instead of classification logits to allow passing these spatial features to the next layers for further processing.

Next, we apply adaptive pooling to standardize the CNN feature maps to a fixed size of 14×14 patches, which matches the input expected by the ViT. The feature maps are then flattened and reshaped into 196 patches (14 × 14 = 196), and each patch is passed through a projection layer to map the CNN's output channels to the ViT's embedding size (384 dimensions for the small ViT model). The fourth component is a pretrained vision transformer (vit_small_patch16_224) with its classification head removed to not use the ImageNet classes. The vision transformer applies self-attention across all image patches to model global spatial relationships in the scene. Rather than just identifying specific objects, the vision transformer will also understand how different patches relate to each other. For example, the ViT can infer cityscape arrangements, recognize the presence of flat land in the foreground with mountains in the background, or detect a river on the right side of an image. After the vision transformer, we apply global average pooling to produce a fixed-size feature vector and then apply a dropout of 0.2 to reduce the risk of overfitting. Finally, a linear classifier outputs the logits for the number of classes (grid cells). There is no activation function applied to the logits because we chose cross-entropy loss as our loss function during training, which is used for classification tasks, like in this specific use case of mapping images to different grid cells.

Using both a CNN and ViT captures both fine-grained local textures and broad spatial structures, which are both essential for fine-grain geospatial classification tasks. For example, the CNN might detect the texture of dry grass and the presence of an asphalt road, while the ViT could recognize the straightness of the road typical of central U.S. highways, and mountain ranges in the background suggestive of the Rocky Mountains. By leveraging the strengths of a CNN and a ViT, the model can learn both local features and global relationships, which should improve the geolocation accuracy.

Training

We decided on a three-stage training approach to maximize the use of pretrained CNN and ViT features while carefully adapting them to our geolocation task. For epochs 0-2, we froze the ResNet50 and ViT encoder layers and trained only the freshly initialized projection and classifier layers to allow these new layers to stabilize and to prevent catastrophic forgetting of the pretrained features. From epochs 3-19, we unfroze all the layers to allow for full fine-tuning on this geospatial task with a learning rate of 1e-4. In the final fine-tuning phase, which contained epochs 20-22, we froze the CNN backbone again and allowed the ViT to continue training. The reasoning is that the CNN already has strong local feature extractors, whereas vision transformers benefit from longer fine-tuning because their attention patterns are more flexible and adapt to new tasks more slowly. Throughout training, we used an adaptive learning rate scheduler to ensure stable convergence and applied weight decay regularization to mitigate overfitting.

Results

The model achieved an accuracy of 16.55% on the held-out test set, meaning it correctly predicted the grid cell in approximately one out of every eight images. Given that there were 2,199 possible classes, random guessing would give an expected accuracy of only 0.045%. This indicates that our model performs approximately 3,677 times better than randomly guessing. When evaluating the top 3 predictions, the model's accuracy improved to 26.55%. When evaluating the top 5 predictions, it increased to 31.76%. When looking at loss, the validation loss was 2.998, whereas the testing loss was 5.95, which indicates moderate overfitting to the validation set. The model may have learned dataset-specific patterns that do not generalize well to unseen data, which presents stronger regularization, usage of larger or more diverse training data, or data augmentation as methods to improve generalization. In order to see how well this model works, we ran experiments using ablations and a probability distribution.

Experiment 1

The goal of this experiment is to perform an occlusion sensitivity analysis using heatmaps to better understand which parts of an image the CNN-ViT hybrid model relies on when predicting geographic locations. We hypothesize that the model will focus on meaningful, distinctive features, like vegetation, mountains, and buildings, rather than the sky or roads, which are common across many images and thus less informative for geolocation. We used a 32x32 pixel occlusion patch with a stride of 8 pixels. At each location, we measured the drop in model confidence for the correct class when a region was occluded. We visualized the original image and overlaid the heatmap to interpret model focus. Below are two bad and two good examples based on how appropriately the model's attention was distributed.

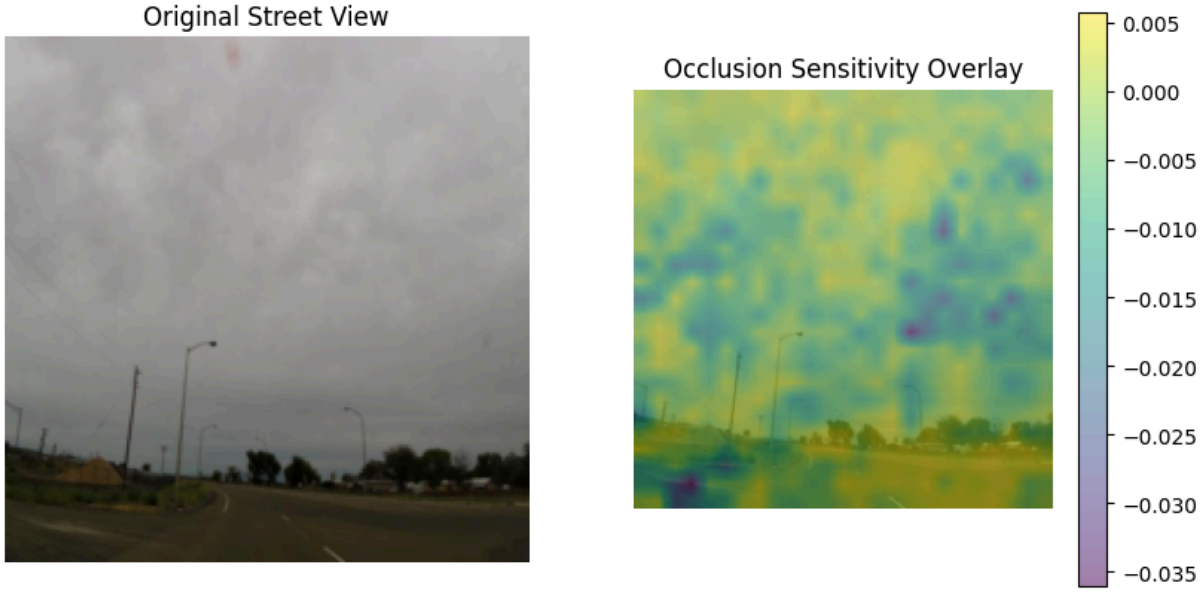
Good Examples:

1. *In this example, the model is correctly sensitive to the sidewalk, houses, and vegetation while largely ignoring the road or sky. Since skies and roads are prevalent in many images, the model's focus on more specific scene elements indicates that it is correctly prioritizing meaningful features for geolocation prediction.*

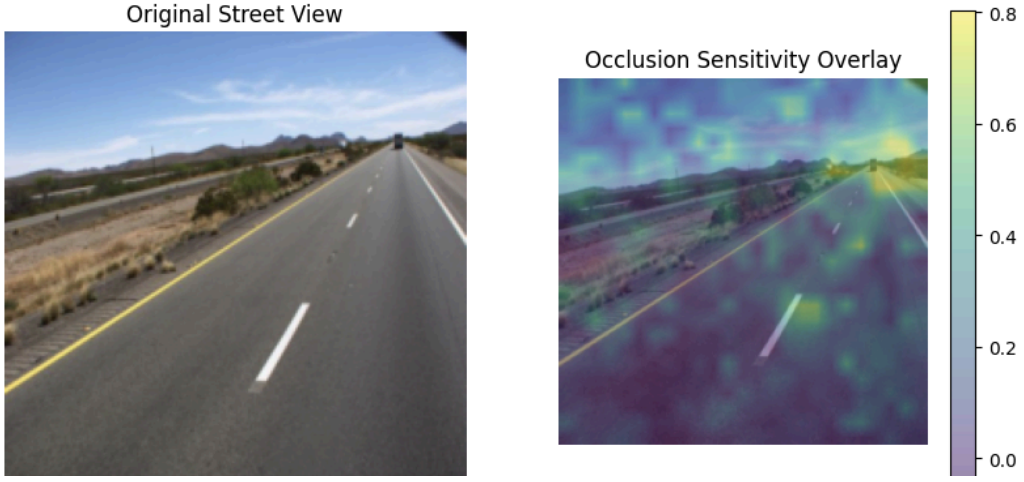
2. *In this example, the model largely ignores the sky and parts of the road and focuses its attention on the vegetation on the right side of the road. While the vegetation on the left side is mostly disregarded, the model correctly emphasizes the vegetation on the right side of the road, which is arguably more discriminative due to the autumn colors of the*

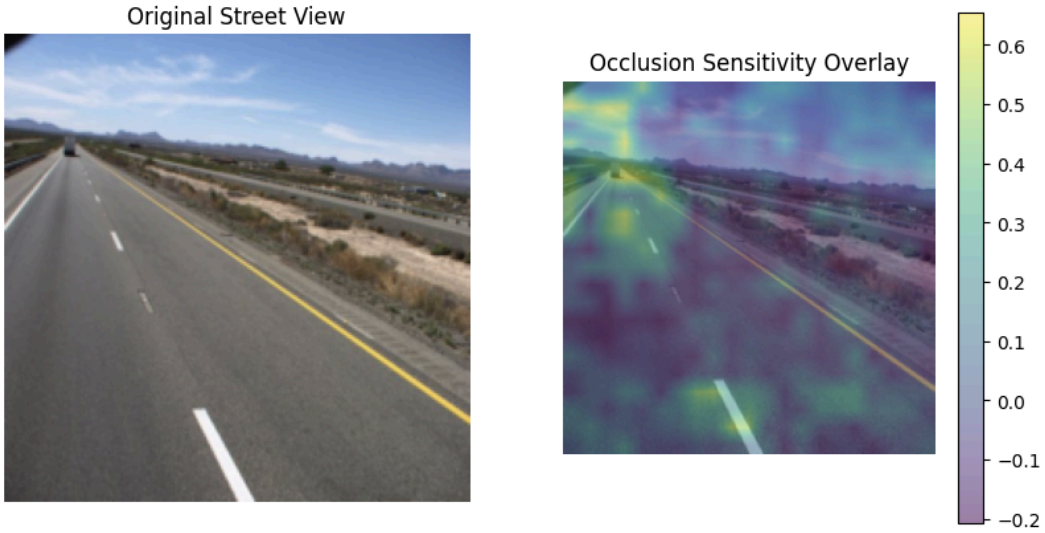
tree leaves. Additionally, the model is also focused on the mountain range in the background, a highly distinctive feature that should help the model better predict the geolocation.

Bad Examples:



1. *In this example, the model focuses some attention on the building on the left side, but a significant portion of its attention is incorrectly placed on the sky, which occupies approximately 80% of the image. While the model ignores part of the sky and road, the model underperforms in this example because too much attention is allocated to the sky, which is visually repetitive and offers little discriminative information for geolocation.*





2. *This example consists of two images showing a rural road with vegetation on both sides of the road and mountains in the background. Instead of focusing on the broader environmental features, the model overfits by placing significant attention on a truck present on the road in both images. As a result, the model cheats and appears to associate the presence of the truck with a specific grid cell, which is not helpful for generalizing in this geolocation task.*

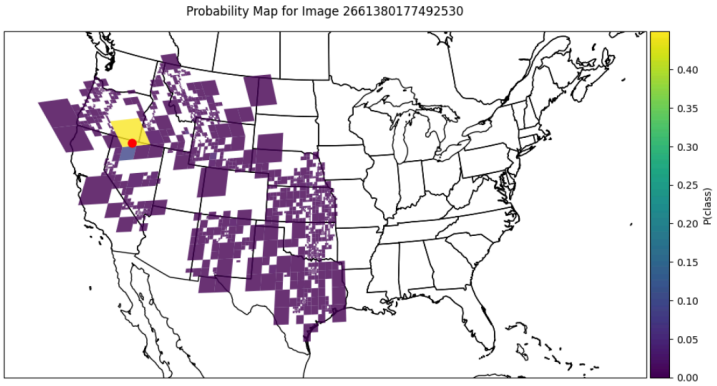
When the model performed well, it focused on meaningful, discriminative features such as vegetation, building structures, and mountain ranges. These features are highly specific to certain geographic regions and thus are valuable for accurate geolocation. In the good examples, the model correctly ignored irrelevant features like the sky and road surfaces, which are common across many locations and thus do not aid in discerning specific regions. However, in several failed examples, the model misallocates attention. The model would sometimes focus excessively on the sky because the sky took up the majority of the image. This implies that the model did not train on enough diverse data with varied proportions of scene elements. Additionally, the model occasionally overfits to an object like a truck, relying on their presence as a shortcut for location prediction rather than focusing on broader environmental features that are more geographically generalizable. These results suggest that while the model has learned to prioritize scene-level features in many cases, it remains vulnerable to learning shortcuts, which could potentially be mitigated with more image augmentation and a larger diverse dataset.

Experiment 2

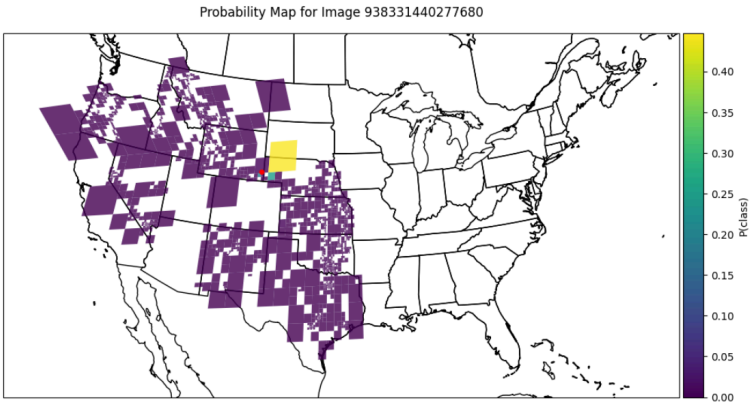
By taking the network's raw logits and applying the softmax function, we obtain a full probability distribution over our learned S2 cells. Visualizing this distribution on a geographic plot gives insight into the model's internal reasoning: a tightly clustered mass of high probability around one cell indicates strong confidence, whereas a spread-out or multi-modal distribution signals uncertainty. In this experiment, we overlay each cell's probability mass on a map of the

United States using Cartopy, drawing brighter fills for higher predicted likelihoods. We hypothesized that the model would cluster most of its probability mass around one mode if it was confident in a prediction or around a couple of modes if it was uncertain. By comparing several good (correct, concentrated) and bad (incorrect, diffuse) examples, we aim to qualitatively assess how well the output distributions align with the true locations, and to learn what these patterns tell us about where the model is strong and where it still struggles. Note that for these maps, the red dot indicates the ground-truth location of the image.

Good Examples:

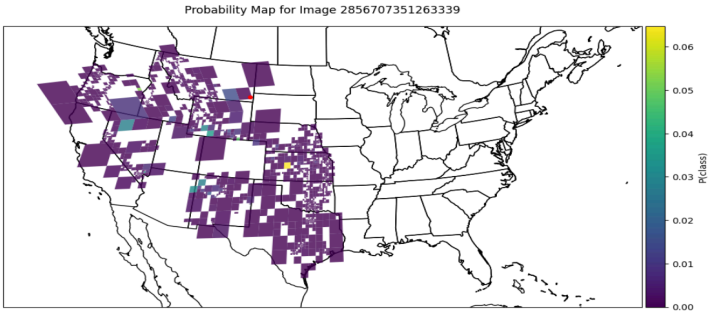


Good Ex. 1: *The model was very confident in this image. It allocated some mass to the cell below the chosen one, but not much anywhere else. Since this cell is large, the majority of this road is likely in the chosen cell, leading to this confidence. Additionally, the dirt on the camera lens could be a clue to the model that the image was taken in this specific location*.

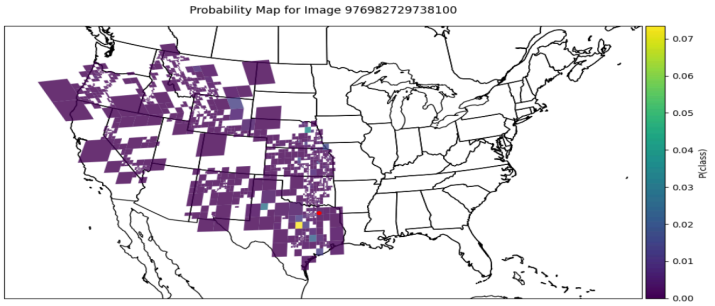


Good Ex. 2: *In this image, although the model's top prediction was incorrect, most of the probability mass was clustered around the guess. It indicates that the model learned something about the region itself, or possibly that the road spans all of those cells. Even if the road did span all of those cells, the model would likely allocate some mass to the correct cell, which it did not. However, the tight cluster indicates a close prediction.*

Bad Examples:



Bad Ex. 1: *Probability distribution is spread and there is no clustering.*



Bad Ex. 2: *Probability distribution is also spread and there is no clustering.*

> *While these images don't seem especially difficult to localize compared to the "good" examples, the model has no idea where to place them. Even the highest-weighted classes only have a probability of ~0.06 and 0.07, respectively. There is no clustering around any modes, and although the model places some extra mass in a few classes, these classes are spread out across the training area with little discernible pattern. This suggests that the model is underconfident with these images: when it gets an input that it isn't sure where to place, rather than clustering mass around a few of its best guesses, it reverts to randomly guessing*

Ultimately, while the model exhibited tight clustering for images it was confident on, it was less than we had expected, and there was no clustering at all for images it wasn't confident on. This implies that the model is miscalibrated: overconfident when it thinks it knows where an image is, and not confident when it doesn't know where an image is. While not directly relevant to the model's accuracy, the model's overall guessing strategy could be better. This could likely be accomplished with more training data, leading to better generalizations across regions.

References

Weyand, T., Kostrikov, I., & Philbin, J. (2016, February 17). *PlaNet - Photo Geolocation with Convolutional Neural Networks*. arXiv. Retrieved April 25, 2025, from https://arxiv.org/abs/1602.05314

Josht000. (2025, March 2). *OSV-Mini-129k: OpenStreetView 5M for small budgets.* [Data set]. Kaggle. https://www.kaggle.com/datasets/josht000/osv-mini-129k

Model Card

Click here to go to the GitHub ReadMe to see the model card

Datasheet

📄 Datasheet for OSV Mini (129K) — A Subset of OSV5M Dataset

---

1. Motivation

For what purpose was the dataset created?
This subset was created to enable **geographic image localization research** at a manageable scale for researchers with **limited computational resources**. The original OSV5M dataset contains over 5 million images globally; this smaller subset of 129,000 images focuses exclusively on **10 selected U.S. states**, allowing for experimentation with hybrid CNN-Transformer models and grid-based geolocation.

Who created the dataset (e.g., which team, research group) and on behalf of which entity?
The dataset was created by selecting a filtered portion of the publicly available **OSV5M dataset** (from OpenStreetView Project), curated and repackaged by Joshua T. for Kaggle publication.

---

2. Composition

What do the instances that comprise the dataset represent (e.g., documents, photos, people)?
The instances are **Street View-style images** captured from various road locations.

How many instances are there?
There are **129,083 images** in this subset.

What data does each instance consist of?
Each instance consists of:

- An RGB image (224×224 recommended preprocessing size).
- A geolocation label mapped to a 9×9 km grid cell index.

Is any information missing from individual instances?
No; each instance includes both the image and its corresponding grid label.

Is the dataset self-contained, or does it link to other datasets?
Self-contained. Images are provided as individual JPEG files.

Does the dataset contain all possible instances, or is it a sample?
It is a **filtered sample** from the OSV5M global dataset, restricted to **10 U.S. states**.

What states are included?

- Texas
- Kansas
- Oregon
- Nevada
- Oklahoma
- Montana
- Idaho
- Nebraska
- New Mexico
- Wyoming

---

3. Collection Process

How was the data collected?
The original OSV5M data was collected through scraping publicly available Street View imagery under fair use policies.
This subset was **filtered geographically** using bounding boxes for selected states, matching latitude and longitude metadata to U.S. boundaries.

Who collected the data?
Original collection by the OSV5M team; curation for this subset was done by Joshua T. (Kaggle author).

When was the data collected?
The original data is from 2021–2022; the subset was curated in early 2024.

What mechanisms or procedures were used for collection?
Automated filtering scripts based on geolocation metadata.

Was any ethical review conducted (e.g., IRB review)?
No IRB review was conducted, as the data is collected from publicly available imagery.

---

4. Preprocessing / Cleaning / Labeling

Was any preprocessing done?

- Images were resized and cropped to standard dimensions for efficient model training.

- Labels were assigned by mapping latitude and longitude to a **Google S2 Geometry Grid System**, and unused grid cells were dropped. Each grid cell has between 20-100 images making these grid cells dynamic in size.

Was the data manually verified?
No extensive manual verification was performed beyond geographic filtering.

---

5. Uses

What are the intended uses of the dataset?

- Geographic image localization.
- Training and evaluating deep learning models for geolocation.
- Experiments in vision-transformer hybrids, CNN baselines, or multi-modal learning.

Who is the target audience?
Researchers, students, and practitioners in computer vision, geospatial machine learning, and localization applications.

Out-of-scope uses?

- Not suitable for commercial deployment without proper legal review.
- Not for personal location prediction or identification of individuals.

---

6. Distribution

Will the dataset be distributed? How?
Yes. It is publicly available on Kaggle under the terms provided by the original OSV5M dataset (open and research-friendly license).

Under what license is the dataset distributed?
Creative Commons license (non-commercial, research use) — reflecting OSV5M licensing conditions.

---

7. Maintenance

Who is supporting/hosting/maintaining the dataset?
Currently hosted on Kaggle by Joshua T.

Will the dataset be updated (e.g., to correct labeling errors, add new instances)?
No active update plan. Static version.

How should users cite the dataset?
Users are encouraged to cite both the Kaggle page and the original OSV5M source:

- OSV Mini Dataset (Kaggle): https://www.kaggle.com/datasets/josht000/osv-mini-129k
- OSV5M Dataset (Original): https://huggingface.co/datasets/osv5m/osv5m