CIS 4190/5190: Applied Machine Learning | Fall 2025

Image2GPS Localization via Regression and Contrastive Retrieval

Team Members: Wanting Yao, Chunyu Jiang, Ruizhe Gao | *Project: [Img2GPS]*

1 Introduction

Predicting GPS coordinates from images has wide applications in GPS-denied scenarios. In this project, we propose two approaches: (1) an EfficientNet-B0-based regression model with a linear head to directly predict GPS coordinates, and (2) a more interpretable retrieval-based method that performs k-NN image retrieval on top of learned feature embeddings. These two approaches achieve comparable performance, with the latter employing a two-stage training strategy that incorporates contrastive loss for representation learning, thereby making the method more interpretable by explicitly aligning the learned feature space with geometric distance. We train and evaluate our methods on a self-collected dataset containing over 3k images captured within a designated campus area. Notably, our approaches outperform both a ResNet-based baseline and a naive mean-coordinate predictor by up to 45%. We release our dataset on Hugging Face at rwxyzgao/IMG2GPS_ALL , and our code is available at IMG2GPS.

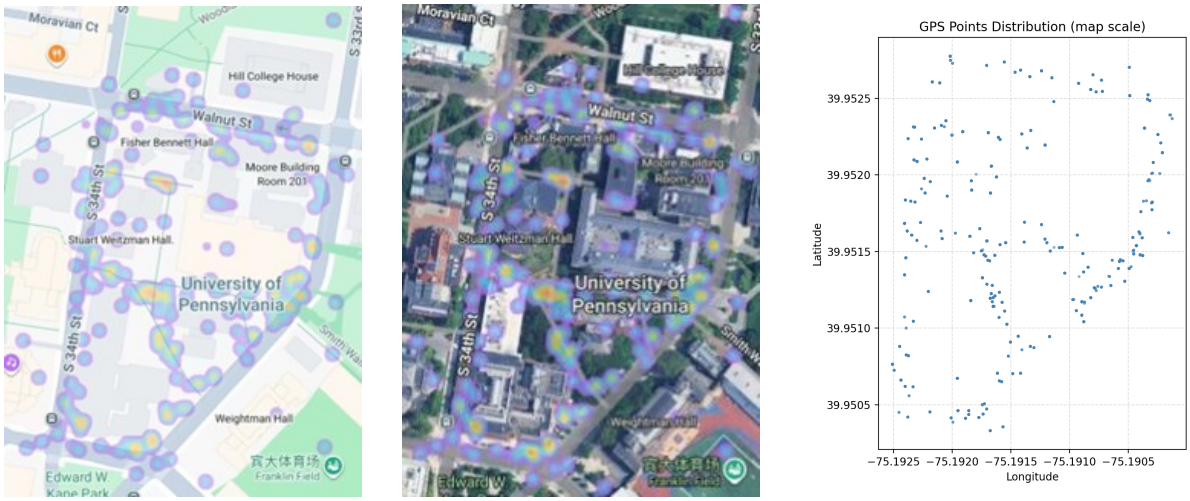
2 Core Components

2.1 Data Collection

2.1.1 Data collection protocol

To build the dataset, we collected images using personal smartphones across a rectangular region of the University of Pennsylvania campus bounded by 33rd Street and Walnut Street, and 34th Street and Spruce Street. The dataset contains approximately 3,200 images with GPS tags covering this area.

At this spatial resolution, GPS coordinates typically differ at the fourth or fifth decimal place. To provide meaningful location distinctions, images were collected at roughly 5–10 meter intervals to maintain sufficient separation between neighboring locations. At each location, eight photos were taken to capture a full 360-degree view. Data collection was conducted across multiple seasons and conditions, including varying weather and ongoing campus construction which increases visual diversity.



(a) Map View (b) Satellite View (c) Data after Cleaning

Figure 1: **Heatmap and Plot of Dataset**

All location labels were obtained directly from the EXIF GPS metadata embedded in the images at the time of capture. The corresponding heatmap generated by Google Photo Gallery is shown in Fig. 1.

2.1.2 Data Curation and Preparation

At the image level, all raw images were converted to JPEG format and resized to a fixed resolution of 1200×900. We then filtered samples using a latitude–longitude bounding box corresponding to the collection area. Finally, we cleaned the metadata by removing GPS points that appeared only once in the dataset, as such isolated points are more likely due to transient GPS noise rather than true location variation. The spatial distribution of the remaining valid GPS points after this cleaning process is shown in Fig. 1c.

For internal evaluation, the dataset was randomly shuffled and split into training and validation sets using an 80/20 ratio. Final model evaluation was conducted on a separate test set that was not used during training or validation, in order to prevent data leakage.

2.1.3 Dataset Summary

After data collection and curation, the dataset contains a total of 3,219 images, which are split into 2,589 training samples and 630 validation samples. Each sample consists of an image paired with its corresponding geographic coordinates. The dataset is organized using a CSV file with three fields: image filename, latitude, and longitude.

The dataset was collected incrementally over a period of approximately one and a half months, with the total number of samples increasing over time as new data was added. The dataset sizes at different stages are summarized in Tab. 5.

Table 1: **Training and Validation Splits for Different Dataset Sizes**

Dataset	#Training Samples	#Validation Samples
Image2GPS-tiny	1323	310
Image2GPS-small	1754	439
Image2GPS-base	2589	630

2.2 Model Design

2.2.1 EfficientNet Regression

EfficinetNet [1] is a family of state-of-the-art convolutional neural networks (CNNs) from Google AI that achieves high accuracy with fewer parameters and faster speeds by using a novel scaling method called compound scaling, which systematically balances network depth, width, and image resolution for optimal performance.

In this Image2GPS project, we adopt EfficientNet as backbone to extract spatial features from input images. Different versions of EfficientNet (B0, B1, B2, and B3) were evaluated on the same dataset in order to identify the optimal backbone architecture. Considering both the empirical performance and the recommended correspondence between model capacity and dataset size, EfficientNet-B0 was selected as the final backbone. This choice is particularly appropriate for our setting, as the dataset contains approximately 3,000 training images with a resolution of 224×224, for which larger EfficientNet variants tend to introduce unnecessary model complexity and a higher risk of overfitting.

On top of the EfficientNet backbone, a lightweight regression head consisting of two fully connected layers is employed to map the extracted visual features to geographic coordinates. Specifically, the high-dimensional feature representation produced by the backbone is first projected to a 256-dimensional embedding, followed by a final linear layer that outputs a 2-dimensional vector corresponding to latitude and longitude. A non-linear activation function ReLU and dropout are applied between the two linear layers to enhance model expressiveness and improve generalization.

To further improve localization accuracy, we incorporate a spatial attention pooling layer with 4 heads after the convolutional feature extractor. Unlike standard global average pooling, which treats all spatial locations equally, the spatial attention mechanism learns to assign different importance weights to different regions of the feature map. This allows the model to selectively focus on spatially informative areas of the image, such as distinctive buildings, road patterns, or landmarks, while suppressing less relevant background regions.

2.2.2 k-NN Retrieval with Learned Feature Embeddings

Inspired by how humans identify a place from visual cues, we propose a retrieval-based localization approach[2, 3]. Given an input image, we first extract its feature embedding using a backbone network, then compare it with embeddings from the training set. Using cosine similarity based k-NN regression, we retrieve the top-k most similar images and predict the GPS coordinates as the weighted mean of their locations.

In practice, we experiment with two types of feature extractors: Vision Transformers (ViT) and convolutional neural networks (CNNs). For ViT, we adopt the architecture and pretrained weights from DINO[2] and perform zero-shot inference. For CNNs, we use the fine-tuned EfficientNet-B0 model introduced in Section 2.2.1 and fit a k-NN regressor on the extracted features for GPS prediction.

Besides, we explore a representation learning strategy that explicitly aligns the feature space with geometric distance. The motivation is to enforce that images closer in Haversine distance should also be closer in the learned embedding space. To this end, we optimize a contrastive loss objective[4] that minimizes the mean squared error between pairwise cosine similarities of feature embeddings and distance-based target similarities derived from GPS coordinates. This encourages geographically nearby samples to cluster together while pushing distant locations apart in feature space. The contrastive loss objective is

$$\mathcal{L} = \frac{1}{B(B-1)} \sum_{i \neq j} \left(\mathbf{f}_i^\top \mathbf{f}_j - \exp\left(- \frac{\mathrm{Hav}\left((u_i, v_i), (u_j, v_j)\right)}{\tau} \right) \right)^2,$$

where B is the batch size, $\mathbf{f}_i \in \mathbb{R}^d$ denotes the feature embedding of the i-th image, (u_i, v_i) are its GPS coordinates, $\mathrm{Hav}(\cdot)$ denotes the Haversine distance, and τ is the scaling factor.

2.3 Evaluation

2.3.1 Evaluation Protocols

- **Testing Dataset:** We randomly collected 153 images in the designated areas for testing purposes. However, due to the fast-moving collection process, the GPS labels in this testing dataset suffer from latency, which introduces excessive noise into the ground-truth labels and renders the testing set unusable. Therefore, we instead use the released dataset which contains 100 images as our testing set to evaluate our model locally.
- **Metrics:** We consider two metrics to evaluate the performance of our model. One is the *mean squared error* between the predicted GPS coordinates and the ground-truth GPS labels (both normalized). This metric is the same as the training loss and provides a straightforward comparison with the training and validation losses. The other metric is the *average Haversine distance(AHD)*, which gives a clear indication of the model's error in terms of geographical distance.
- **Best Model:** To address the overfitting problem, we apply an early stopping strategy and save the model with the lowest validation loss at each run. We then evaluate the best-validation model on the training set. To ensure reproducibility, we set a fixed random seed of 42. We evaluate models with different backbone architectures as well as different hyperparameters. For models that use the k-NN algorithm to predict GPS coordinates, we fit the k-NN model during the evaluation process. The model with the smallest *average Haversine distance* is selected for submission to the leaderboard.

2.3.2 Results

With EfficientNet-based regression, we experiment with different backbones and evaluate their performance on the test set. Following the scaling law introduced in the original EfficientNet paper [1], we test multiple EfficientNet variants from B0 to B3, where the model size progressively increases. The corresponding results are reported in Tab. 2. Contrary to the common assumption that larger models yield better performance, EfficientNet-B0 achieves the best overall leaderboard performance and is also our strongest model. This can be explained by the scaling law, which suggests that model capacity should be aligned with dataset size to avoid overfitting. Since our dataset contains only 3k images, smaller models are better suited to this data scale.

Table 2: **Model Scaling Performance of EfficientNet Backbones**

Backbone	#Params	Train-Loss	Val-Loss	Val-AHD(m)	Test-AHD(m)	LB(m)
EfficientNet-b3	11.09M	0.1494	0.1450	32.41	55.22	-
EfficientNet-b2	8.06M	0.1263	0.1516	33.23	46.07	-
EfficientNet-b1	6.84M	0.1146	0.1376	31.48	56.26	-
EfficientNet-b0	4.34M	0.1118	0.1321	31.06	48.54	47.43

Each model is trained for 100 epochs, and the training loss curves for the four models are shown on the left-hand side of Fig. 2. As shown on the right-hand side, the validation loss begins to exceed the training loss after approximately 30 epochs, indicating the onset of overfitting. To mitigate this, we select the model checkpoint that achieves the lowest validation loss on the validation set.

Figure 2: **Training and Validation Loss Curves of EfficientNet Models**

The performance of EfficientNet is improved by introducing attention mechanism, as shown in Fig. 3. Compared to EfficientNet with only linear head, the EfficientNet with both attention pooling and linear layer has a smoother loss curve and a smaller validation AHD. It is also obvious that by fine tuning the hyperparameters, the model can yield a better performance (The only difference between Attention_SOTA and att_3kimg is hyperparameters).

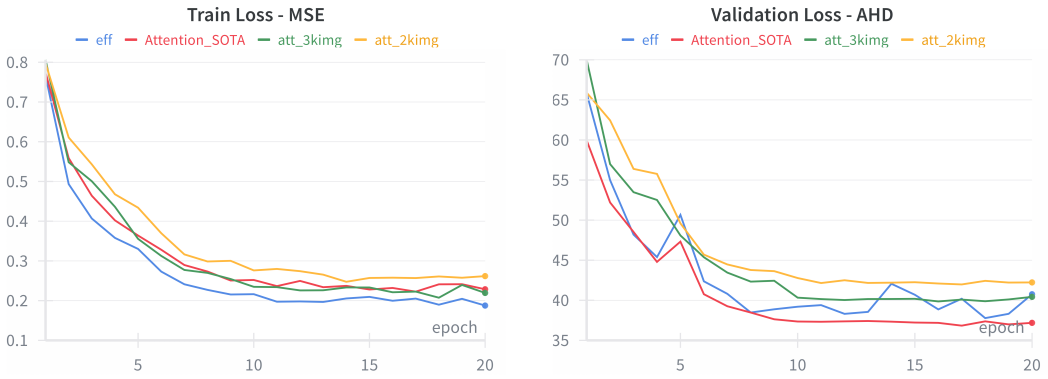


Figure 3: **Loss of EfficientNet and EfficientNet with Attention**

For feature extraction with k-NN regression, we evaluate three models: a ViT backbone, an EfficientNet-B0 trained with MSE loss, and an EfficientNet-B0 trained with a contrastive learning objective. k-NN regression is then performed on the extracted feature embeddings. The best performance is achieved by EfficientNet-B0 with contrastive learning, using $k = 3$. This is likely because our dataset typically contains around three visually similar images per location. The results are shown in Tab. 3.

Overall, the best EfficientNet regression model achieves an average Haversine distance of 48.54 m, while our best feature-extraction-based k-NN model achieves 49.48 m on the local test set. On the leaderboard, our method further improves to 47.43 m. In comparison, a naive mean-coordinate predictor yields an error of 88.76 m, indicating that our approaches reduce localization error by approximately 45%.

Table 3: **k-NN Regression Performance with Different Backbones and k**

Backbone	Training Recipe	k-NN	Val-AHD (m)	Leaderboard (m)
ViT-S/8	zero-shot	1	59.56	-
		3	54.23	51.26
		5	55.30	-
EfficientNet-b0	regression mse loss	1	61.77	-
		3	61.75	54.36
		5	61.82	-
EfficientNet-b0	representation learning + contrastive loss	1	54.68	-
		3	49.48	53.52
		5	50.09	-

3 Exploratory Components

- **Datasets:** After the data collection, we noticed GPS errors in a small number of photo groups. In certain areas or on specific dates, the recorded locations showed large deviations when visualized on the map. These errors are related to weak GPS signals, which may be affected by battery saving modes on mobile phones or signal obstruction near buildings. To solve this problem, we applied a data cleaning procedure. First, we removed samples from specific dates that have large GPS drift. Second, we filtered out isolated location points that appeared only once in the dataset, as these points are more likely caused by GPS noise instead of the true location. As a result, the dataset size was reduced from 3842 images to 3219 higher-quality samples, improving overall label reliability.

- **Technique:** We explore a representation learning objective that explicitly aligns the feature space with geometric distance by matching pairwise cosine similarities between feature embeddings to their corresponding Haversine distances. This alignment improves the performance of k-NN regression and enhances the interpretability of the retrieval-based approach.

- **Analysis:** One important insight is the significance of data. As shown in Tab. 4, increasing the number of training samples reduces the test AHD. This indicates that larger dataset improves the performance of model.

Table 4: **Model Performance under Different Dataset Scales**

Dataset	Train-Loss	Val-Loss	Val-AHD(m)	Test-AHD(m)
Image2GPS-tiny	0.1115	0.1560	34.66	68.86
Image2GPS-small	0.1096	0.1461	34.33	58.15
Image2GPS-base	0.1118	0.1321	31.06	48.54

- **Application:** Practical applications of our Image2GPS system include campus exploration, tourism, and photo check-in localization. On social media platforms, users frequently share appealing photos taken at scenic or distinctive locations on campus. However, viewers who wish to visit these locations often do not know where the photos were taken, as precise geographic information is usually unavailable or omitted. Our model addresses this problem by estimating the GPS coordinates directly from a single image. Given a photo shared online, the Image2GPS system can predict its approximate location within the campus area, enabling the users to navigate to the corresponding spot.
- **Teaching:** We will release our code and provide documentation for deploying our Image-to-GPS algorithm.

4 Team Contributions

- **Wanting Yao:** Data collection, model choice, model design, technique, report.
- **Chunyu Jiang:** Data collection, model design, hyperparameter tuning, analysis, report.
- **Ruizhe Gao:** Data collection, dataset processing, HP tuning, model choice, datasets, report.

5 Extra Credit

The presentation video.

6 Acknowledgments

Thanks to Team 10 for sharing their exploration on merging CNNs with ViTs. Inspired by their reference paper, we experimented with using DINO as a feature extraction backbone. Additionally, ChatGPT was used for debugging.

7 Suggestions for future iterations of this project

It would be better if the backend test set used a more up-to-date dataset, since campus infrastructure changes over time. Outdated images may introduce out-of-distribution data caused by construction or renovations, which can negatively affect the fairness and reliability of model evaluation.

References

[1] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019.

[2] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021.

[3] Filip Radenović, Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondřej Chum. Revisiting oxford and paris: Large-scale image retrieval benchmarking. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5706–5715, 2018.

[4] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

Appendix

A Additional Data Curation and Implementation Details

A.1 An Alternative Method for GPS Error Handling

We also considered reassigning isolated GPS points to nearby locations based on temporal proximity. However, this approach could introduce incorrect labels by forcing samples to match inaccurate coordinates. Since new data collection was relatively easy in our setting, we chose to retake images instead of reusing potentially noisy labels.

A.2 Data Collection Conditions

The dataset was collected across multiple dates and conditions. Images were taken during November and December, mainly in the afternoon and early evening. During data collection, variations in campus construction activities and parking vehicles were naturally captured, introducing additional visual diversity. We also report an auxiliary observation on dataset size and model performance for reference.

A.3 Data Augmentation Inspection

Following the reference implementation, we applied standard data augmentation techniques, including random cropping, resizing, and color jittering. During inspection of augmented samples, we observed that random cropping could produce images dominated by sky or vegetation, which contain limited location-specific visual cues. We briefly tested removing random cropping to preserve more global scene context. However, this change led to worse validation performance. Therefore, we retained the original augmentation pipeline.

B Other attempts to enhance to model

We explored other loss functions. HaversineLoss directly calculates the distance on the earth surface between the predicted GPS coordinates and the real GPS coordinates. However, during the training process the training loss sometimes became NaN after a few epochs. NaN appears probably because the network's latitude and longitude predictions are not constrained, so during training, especially in later epochs, they can temporarily go out of the valid range, which makes the Haversine formula numerically unstable (for example, taking the square root of a negative number due to floating-point errors). Once this happens, the loss becomes NaN and propagates through the model. Furthermore, there is no significant improvement after employing HaversineLoss, therefore this method is eventually disgarded.

C Hyperparameters

Table 5: **Hyperparameters**

Learning Rate	1e-3
Batch Size	128
Dropout	0.3
K	3
Heads	4

D Evaluation Results

Fig.4 visualizes the predicted and ground-truth GPS coordinates for different EfficientNet models, with error lines indicating the localization error. From top-left to bottom-right, the subplots correspond to EfficientNet-B0 through EfficientNet-B3, respectively.

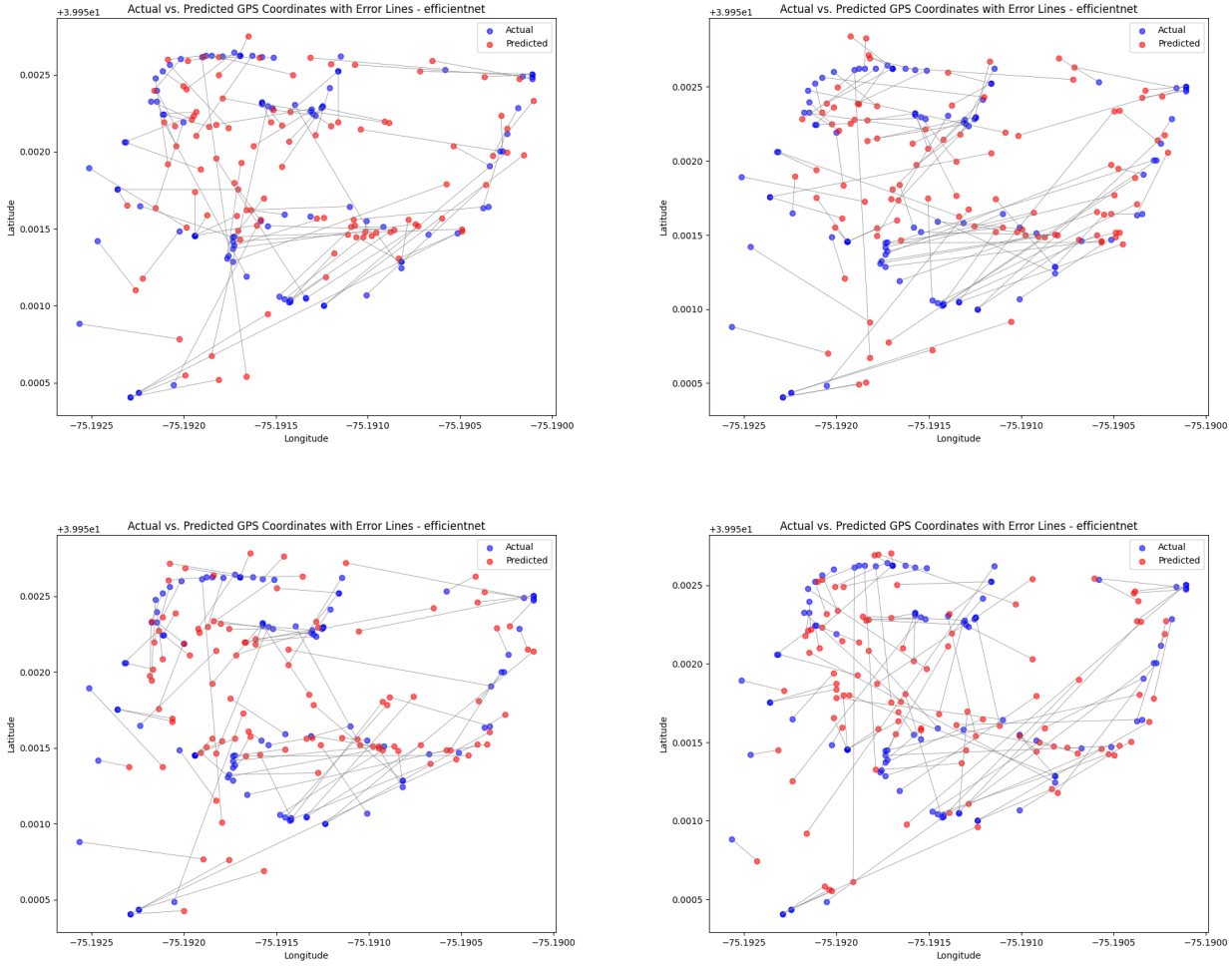


Figure 4: Evaluation Results of Different EfficientNet Models

E Features Shaping Result

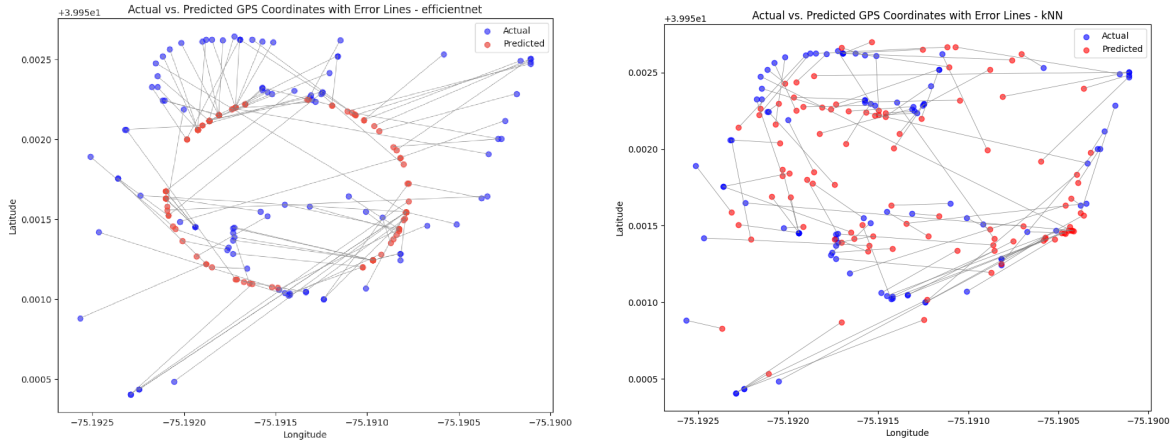


Figure 5: **Comparison of k-NN Results Before and After Contrastive Loss Shaping**

Fig. 5 compares k-NN localization results before and after applying contrastive loss–based feature shaping. After contrastive training, the predicted locations align more closely with the ground-truth GPS coordinates, and the error lines are generally shorter and more consistent. This suggests that contrastive loss helps structure the feature space to better reflect geometric proximity, resulting in improved k-NN retrieval performance.

Before feature shaping, the k-NN results with $k = 1$ (see Fig. 5, left) show a circular pattern because the learned feature space is dominated by visual similarity rather than spatial relationships. Many visually similar locations on campus are mapped close together in feature space, causing multiple test images to be matched to the same nearest neighbor and collapse onto a small set of predicted points. Contrastive loss mitigates this issue by encouraging features to align with geographic distance, leading to more spatially consistent predictions.