Campus Image-to-GPS Regression for Localization and Navigation

Guy Stein, Yotam Tsur, Guy Zilberstein
Introduction to Deep Learning
Ben-Gurion University of the Negev

January 2026

Abstract

This report presents a deep learning approach for predicting absolute GPS coordinates (latitude and longitude) from images of a university campus. The task is formulated as a regression problem, mapping visual features directly to geographic coordinates. We address key challenges including mobile phones gps accuracy errors, lighting variations (different times of day), and visual ambiguity in repetitive architectural structures. Our proposed method utilizes a ConvNeXt-Tiny ensemble with a custom MLP regression head, trained using Huber Loss for robustness to label noise and a comprehensive data augmentation strategy. We further employ an Active Learning loop to identify and incorporate "blind spot" locations into the training set. Our final ensemble model achieves a Mean Distance Error (MDE) of 7.16 meters and a Median Error of 6.48 meters on a held-out test set of 1,023 images, significantly outperforming ResNet and EfficientNet baselines.

1 Introduction

Localization is a fundamental capability for autonomous navigation and location-based services. While GPS provides an extensive solution, it suffers from signal degradation in urban canyons or indoors. Visual localization, determining location from images, offers a complementary or alternative approach.

In this project, we tackle the problem of **Single-Image GPS Regression**: given a single RGB image taken on campus, predict its exact Latitude and Longitude. Unlike retrieval-based methods that require a database of geo-tagged images at inference time, our approach regresses coordinates directly using a neural network, offering a compact and fast solution.

The primary challenges in this domain are:

- **Dataset Creation**: We built a dataset from scratch - requiring careful consideration of image capture strategies, photographing at different times of day, discovering effective viewpoints, and finding the right balance between dataset size and quality without inflating the dataset with redundant images.
- **Visual Ambiguity**: The campus features repetitive architecture (e.g., similar-looking corridors, walkways), making distinct localization difficult.
- **Lighting Conditions**: The model must generalize across different times of day, particularly challenging night scenes where visual information is reduced.
- **Label Noise**: Ground truth GPS coordinates from smartphones have inherent errors (drift) of 3-10 meters.

Our main contributions are:

1. A robust training pipeline integrating manual GPS correction and an Active Learning loop to iteratively improve data quality.

2. The effective application of modern CNN architectures (ConvNeXt) with a custom regression head, demonstrating superior performance over Transformers (Swin-T) for this dataset scale.

3. A specialized data augmentation strategy that significantly reduces error on low-light test samples or test pictures taken under different times of the day.

4. Our model reached a Mean Error of 7.16m on our extensive test set. We found this result encouraging, as it appears to approach the inherent noise levels of our ground truth GPS labels.

2 Related Work

This project applies deep learning concepts learned in class to visual localization. We use transfer learning with pre-trained CNN backbones (EfficientNet [1], ConvNeXt [2]) from PyTorch's `torchvision` library, adapting them from ImageNet classification to GPS coordinate regression.

3 Method

3.1 Input/Output Representation

Our system implements the required `predict_gps(image)` interface:

Input:

- Type: `numpy.ndarray`, Shape: $(H, W, 3)$, Dtype: `uint8`
- Channel order: RGB (not BGR)
- Value range: $[0, 255]$

Internal Representation: The model predicts coordinates in a local Cartesian system (x_{meters}, y_{meters}) relative to a reference point. This normalization is critical because neural networks struggle to regress high-precision numbers like raw GPS (e.g., Lat 31.26201).

Output:

- Type: `numpy.ndarray`, Shape: $(2,)$, Dtype: `float32`
- Format: `[latitude, longitude]` in absolute GPS coordinates

Postprocessing: The predicted (x, y) meters are converted back to absolute GPS using the inverse of the normalization equations:

$$\text{lat} = \text{lat}_{ref} + \frac{y_{meters}}{R_{earth}} \tag{1}$$

$$\text{lon} = \text{lon}_{ref} + \frac{x_{meters}}{R_{earth} \times \cos(\text{lat}_{ref})} \tag{2}$$

3.2 Model Architecture

Figure 1 illustrates our pipeline.

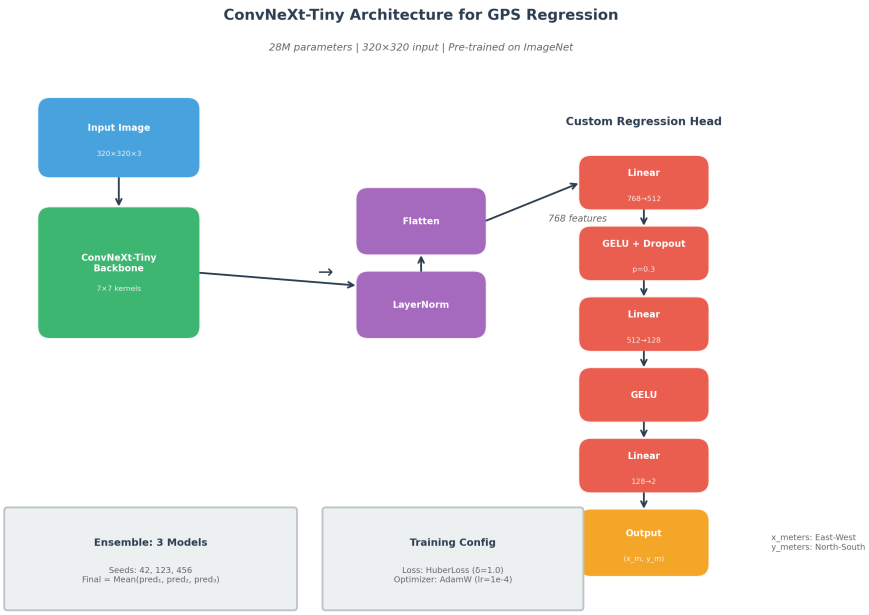

Figure 1: ConvNeXt-Tiny architecture for GPS regression. The backbone extracts 768-dimensional features which are passed through a custom MLP head to predict (x, y) coordinates in meters.

We adopt **ConvNeXt-Tiny** [2] as our backbone. This choice was the culmination of an extensive architectural search involving ResNet-18, the EfficientNet family (B0, B3, V2-S), and Swin Transformers. ConvNeXt's hierarchical feature representation using large 7×7 kernels captures wider spatial contexts than the standard 3×3 kernels in ResNet or EfficientNet-crucial for localization where relative landmark positions matter.

The regression head:

- **Input**: 768-dimensional feature vector from ConvNeXt global pooling.
- **Structure**: Linear($768 \to 512$) $\to$ GELU $\to$ Dropout(0.3) $\to$ Linear($512 \to 128$) $\to$ GELU $\to$ Linear($128 \to 2$).
- **Output**: Predicted $(\hat{x}, \hat{y})$ in meters.

3.3 Preprocessing

All raw HEIC images are resized to 320×320 pixels. We found this to be the optimal trade-off between resolving fine landmarks (e.g., building numbers) and training efficiency, outperforming 224×224 and 256×256.

GPS coordinates are normalized to a local Cartesian plane:

$$x_{meters} = (\text{lon} - \text{lon}_{ref}) \times \cos(\text{lat}_{ref}) \times R_{earth} \tag{3}$$

$$y_{meters} = (\text{lat} - \text{lat}_{ref}) \times R_{earth} \tag{4}$$

3.4 Training Procedure

Loss Function: We employ **Huber Loss** ($\delta = 1.0$), a hybrid between MSE and MAE (Mean Absolute Error) defined as:

$$L_\delta(a) = \begin{cases} \frac{1}{2}a^2 & \text{if } |a| \leq \delta \\ \delta(|a| - \frac{1}{2}\delta) & \text{otherwise} \end{cases}$$

where a is the prediction error. For small errors ($|a| \leq \delta$), it behaves like MSE (quadratic), providing strong gradients for precise learning. For large errors ($|a| > \delta$), it becomes linear like MAE, preventing outliers from dominating the gradient. This choice is critical for our dataset: despite extensive manual auditing, some training images inevitably retain incorrect GPS labels. With MSE, a single mislabeled sample with a 30m error would generate a disproportionately large gradient, forcing the model to drastically adjust its weights to accommodate that one outlier. Huber Loss dampens this effect - the same outlier produces a much smaller gradient, allowing the model to learn the general pattern without being derailed by label noise.

Optimizer (AdamW): We use **AdamW** with learning rate 1×10^{-4} and weight decay 1×10^{-4}. Early experiments showed AdamW yielded better convergence than standard Adam. Weight decay is a regularization technique that penalizes large weights, encouraging simpler models that generalize better. The key difference:

- **Adam**: Weight decay is added to the loss *before* adaptive scaling, leading to inconsistent regularization across parameters.
- **AdamW**: Weight decay is applied *directly* to weights after the gradient update, providing consistent regularization.

We hypothesize this cleaner separation contributed to our improved results, particularly for the diverse feature scales learned by ConvNeXt.

Learning Rate Scheduler: We use `ReduceLROnPlateau` to monitor validation loss and reduce the learning rate by factor 0.5 when training plateaus. This helps because:

- **Early training**: Higher learning rate enables broad exploration of the loss landscape.
- **Later training**: Lower learning rate allows finer adjustments as the model approaches a good solution.

The **patience** parameter controls how many epochs to wait before reducing. We set patience=7 (increased from 3) because aggressive reduction prevented the model from escaping local minima or recovering from temporary loss spikes.

Data Augmentation: Data augmentation artificially expands the training set by applying random transformations to images during training. This helps the model generalize by exposing it to variations it might encounter at test time (different lighting, slight rotations, etc.) without requiring additional data collection. For our GPS regression task, we carefully selected augmentations that preserve the validity of the coordinate labels:

- **Geometric**: Random Rotation ($\pm 5°$), Random Perspective (scale 0.2). Note: *Horizontal Flip* is disabled as it would invalidate the physical x-coordinate.
- **Photometric**: Color Jitter (Brightness, Contrast, Saturation) to simulate varying lighting conditions at different times of day.
- **Night Simulation**: To address poor performance on night images, we implement a specific augmentation that reduces brightness (factor 0.4-0.7) and contrast with probability $p = 0.2$. *Refinement:* Early experiments with brightness 0.1-0.4 produced images that were too dark (effectively black), confusing the model. We tuned the range to 0.4-0.7 to preserve structural visibility while simulating low-light conditions.
- **Random Erasing**: Randomly erases rectangular patches in the image ($p = 0.2$), forcing the model to rely on multiple visual cues rather than a single landmark. This simulates real-world occlusions (such as people) partially blocking the view.

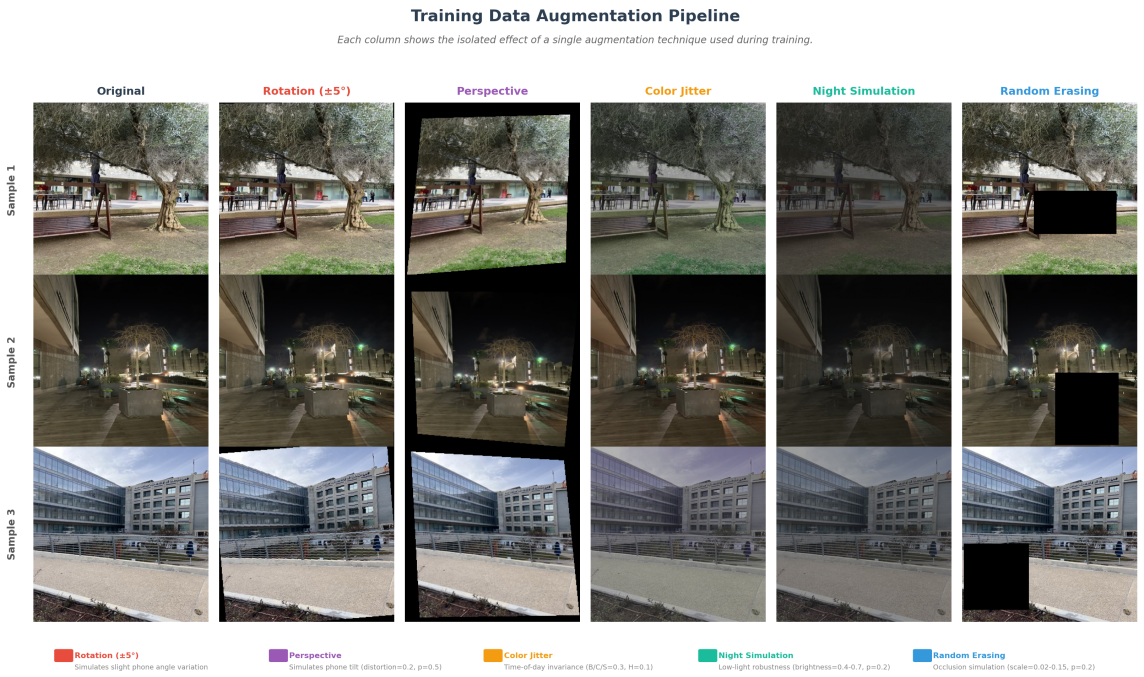

Figure 2: Training augmentation pipeline. Each column shows the isolated effect of a single augmentation used during training (rotation, perspective, color jitter, night simulation, random erasing) on sample images. This illustrates how we simulate realistic viewpoint and lighting variations while preserving the scene's identity.

Training Progress: Figure 3 shows the training progress of our ConvNeXt-Tiny model. The training loss decreases rapidly in the first 25 epochs, then continues to improve more gradually. The train/val gap ratio of approximately 1.9× indicates healthy generalization without severe overfitting.

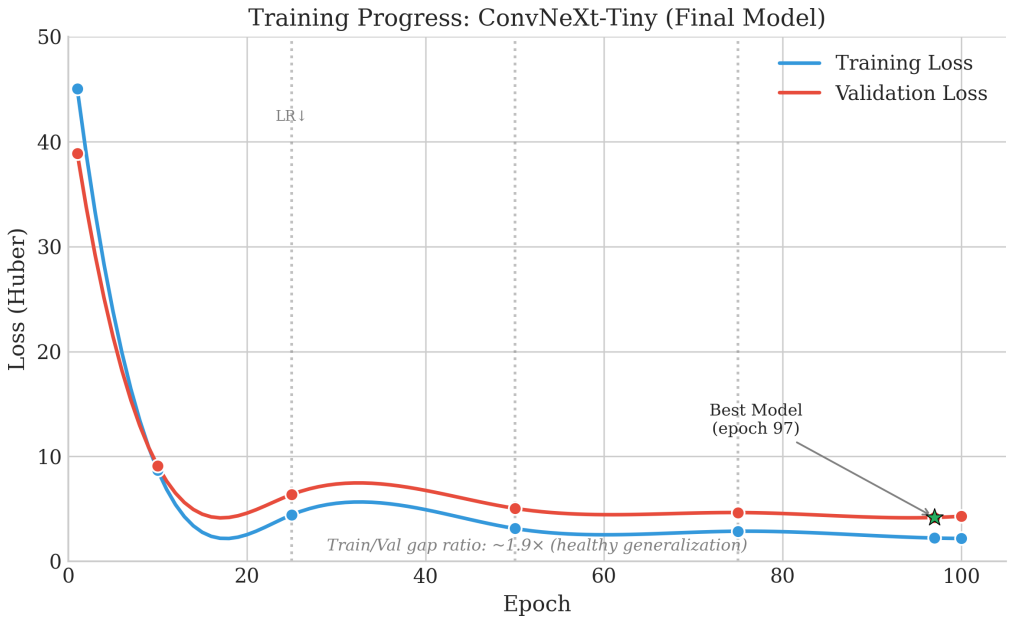

Figure 3: Training and validation loss curves for ConvNeXt-Tiny (100 epochs). Vertical dotted lines indicate approximate learning rate reduction points (patience=7). The best model was saved at epoch 97 with validation loss 4.18. The moderate gap between training and validation loss indicates good generalization. Extended training beyond 100 epochs showed diminishing returns, as validation loss plateaued while training loss continued decreasing, indicating the onset of overfitting.

3.5 Active Learning: Diagnostic Test Set Strategy

A key methodological decision was how we used the test set. Rather than treating it purely as a final evaluation metric, we designed a **diagnostic-driven data collection process**:

Phase 1: Deliberate Over-Collection. We intentionally collected a *massive* external test set (1,322 images) from a separate photo session covering the entire area. This was significantly larger than needed for evaluation alone. The purpose was to use this dataset as a **probe** to expose the model's geographic blind spots, areas or viewpoints not adequately represented in training.

Phase 2: Blind Spot Identification. After training an initial model, we evaluated it on this large test set and identified samples with extreme errors ($> 30m$). Crucially, we did not blindly move these to training. We developed a custom visualization script (`visualize_worst.py`) that renders each high-error sample alongside a satellite map showing both the ground truth GPS location and the model's predicted location. This tool enabled systematic *visual auditing* of each failure, allowing us to determine the **root cause**:

- *GPS Label Error?* → Correct the label, keep in test set.
- *Genuinely Ambiguous View?* → Keep in test set (these represent irreducible error).
- *New Location/Angle Not in Training?* → Move to training set to expand coverage.

Phase 3: Iterative Expansion. We performed two rounds of this process:

1. **Round 1**: Identified 167 images ($> 30m$ error) representing blind spots → Training set grew from 1,950 to 2,117.
2. **Round 2**: Identified 131 images ($> 20m$ error) from remaining failures → Training set grew to 2,248.

Methodological Justification. This approach follows the principles of *data-centric deep learning*, where model performance is improved by systematically enhancing dataset quality and coverage rather than solely tuning model architecture. The transferred samples were not arbitrary "hard examples", they represented specific geographic locations absent from the original training distribution. The audit process ensured that only samples identified as *coverage gaps* (not mislabeled data or inherently ambiguous views) were moved. Importantly, the final test set (1,023 samples) remains substantial, externally collected, and representative of the full target area, preserving the integrity of our evaluation.

3.6 Ensemble

After optimizing our single ConvNeXt-Tiny model, we reached a point where further hyperparameter tuning yielded diminishing returns. To extract additional performance, we trained 3 identical models with different random seeds (42, 123, 456) and averaged their predictions at inference time.

Ensembling works because each model, despite identical architecture and training data, learns slightly different feature representations due to random initialization and data shuffling. By averaging predictions, individual model errors tend to cancel out, reducing overall variance. As shown in Figure 4, this approach improved our mean error from 7.46m (best single model) to 7.16m (ensemble), a 4% improvement that provided the final refinement to our results.

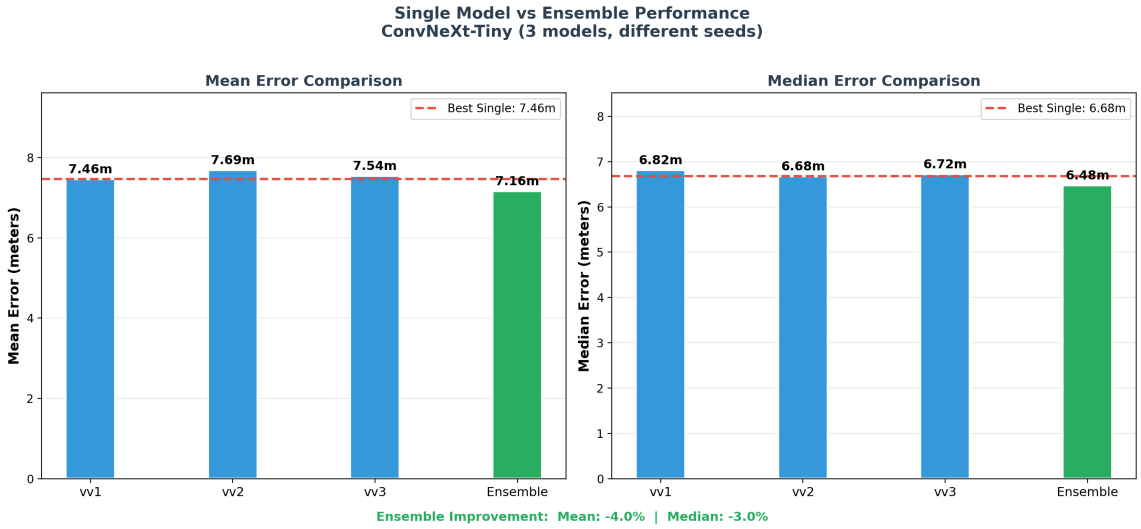

Figure 4: Single model vs ensemble performance comparison. The ensemble of 3 ConvNeXt-Tiny models (different seeds) achieves 7.16m mean error, improving upon the best single model (7.46m) by 4%. This variance reduction technique provided the final improvement when other optimizations plateaued.

4 Experiments

4.1 Dataset

Our dataset consists of approximately 3,300 images collected using smartphones across the university campus, including both training data and a separate test set for evaluation. Images were captured in various conditions: daytime, nighttime, different weather, and multiple viewpoints.

GPS Label Cleanup: We addressed the inherent noise in smartphone GPS labels through systematic correction:

- **Training Set**: Raw EXIF metadata includes a `gps_accuracy_m` field indicating positioning uncertainty. We sorted images by this field and manually reviewed samples with accuracy > 20m. Using Google Maps satellite imagery, we corrected **188 samples** across 4 batches.
- **Test Set**: As described in Section 3.5, we also corrected test labels when analyzing the model's worst predictions. If a high-error sample was due to GPS drift rather than model failure, we corrected the label and kept it in the test set. This ensured our evaluation metrics reflected true model performance, not ground truth errors.
- **Why It Matters**: Without this cleanup, the model would learn from incorrect labels, and our test results would be misleading. The correction workflow was integrated into our data pipeline (`normalize_coords.py`), ensuring corrections persisted across dataset regenerations.

Data Splits:

- **Training**: 1,911 images (85% of training pool, after Active Learning expansion)
- **Validation**: 337 images (15% of training pool, used for model selection and early stopping)
- **Test**: 1,023 images (External hold-out set from a separate collection session)
- **Night Holdout**: 54 images (Dedicated low-light evaluation subset)

Hyperparameter Search: We conducted over 20 distinct training experiments to identify the optimal configuration. Our search space included:

- **Input Resolution**: Evaluated 256×256, 300×300, 320×320, and 384×384.
- **Loss Functions**:
  - Standard MSE
  - Inverse Variance Weighting: weighting samples by $1/\sigma^2$ based on GPS accuracy, giving more weight to precise labels.
  - Area-Specific Z-Score Weighting: penalizing outliers relative to their local area's variance rather than globally.
  - Huber Loss
- **Backbones**: ResNet-18, EfficientNet (B0, B3, V2-S), ConvNeXt-Tiny, Swin-T.
- **Augmentations**: Tested combinations of geometric warps, crops, and photometric shifts.

4.2 Evaluation Metrics

- **Mean Distance Error (MDE)**: Average Euclidean distance in meters between predicted and ground truth coordinates.
- **Median Error**: 50th percentile of error, robust to outliers.

4.3 Baseline

We define **ResNet-18 with MSE loss** as our baseline, representing a standard transfer learning approach for regression. This achieved a Mean Error of 29.25m on our test set.

4.4 Quantitative Results

Table 1 presents selected milestones from our optimization journey (not all 20+ experiments). These highlight the impact of key design decisions, from backbone selection and loss function to Active Learning integration.

#	Model	Res	Strategy / Key Change	Loss	Mean Error (m)
			Phase 1: Baselines & Loss Functions		
1	ResNet-18	256	Baseline	MSE	29.25
2	ResNet-18	256	Area-Specific Z-Score Weighting	Weighted MSE	32.23
3	EfficientNet-B0	256	Switch to EffNet, standard MSE	MSE	13.45
			Phase 2: Scaling & Overfitting Battles		
4	EfficientNet-V2-S	384	Scale up model & resolution	MSE	20.26
5	EfficientNet-B3	300	Mid-sized model	MSE	14.33
6	EfficientNet-B0	256	+MLP Head, Huber Loss, Revised Augs	Huber	11.63
			Phase 3: Active Learning + Ensemble		
7	EfficientNet-B0	256	Ensemble (3x), +Night Simulation	Huber	9.83
8	EfficientNet-B0	320	+Active Learning (Prob. Photos)	Huber	8.17
9	Swin-T	320	Vision Transformer experiment	Huber	9.68
10	ConvNeXt-Tiny	320	Architecture Change	Huber	7.55
11	**ConvNeXt-Tiny**	**320**	**Ensemble (3x)**	**Huber**	**7.16**

Table 1: The optimization journey. We conducted over 20 experiments; selected milestones show the path from ∼29m error to 7.16m.

Our final ConvNeXt-Tiny Ensemble achieves a Mean Error of **7.16m**, which is a **75**% improvement over the ResNet-18 baseline.

Error Distribution: Figure 5 visualizes the distribution of prediction errors. The majority of predictions fall within 10 meters, with a clear right-skewed distribution characteristic of well-calibrated regression models.

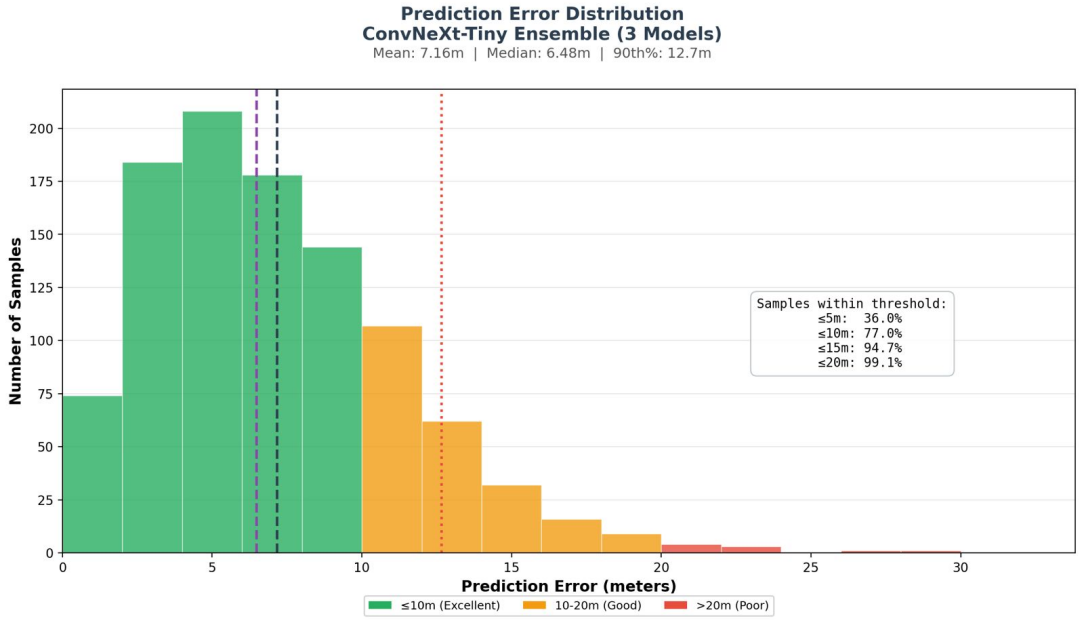

Figure 5: Error distribution of the ConvNeXt-Tiny ensemble (3 models) on the test set ($N = 1,023$). Bars are color-coded: green ($\leq$10m, excellent), orange (10-20m, good), red (>20m, poor). Vertical lines mark the median (6.48m, purple), mean (7.16m, black), and 90th percentile (12.7m, red). 77% of predictions fall within 10 meters.

Table 2 provides the exact breakdown:

Error Threshold	Count	Percentage
$\leq$ 5m	368	36.0%
$\leq$ 10m	788	77.0%
$\leq$ 15m	969	94.7%
$\leq$ 20m	1,014	99.1%
$>$ 20m	9	0.9%

Table 2: Cumulative error distribution. Over 77% of predictions fall within 10 meters, and 99% within 20 meters.

Night Holdout Results: Table 3 shows performance on the dedicated low-light evaluation set.

Metric	Value
Mean Error	7.43m
Median Error	6.66m
Under 10m	74.1%
Under 20m	96.3%
Max Error	29.75m

Table 3: Night holdout results. The model generalizes well to low-light conditions thanks to the Night Simulation augmentation and adding night photos to training set.

4.5 Qualitative Analysis

We systematically analyzed the "Top 25 Worst Predictions" to diagnose failure modes.

- **Night Blindness**: Initially, 48% of worst errors were night images, despite including night photos in the training set. We attribute this to the night images being fewer in quantity and less extensive in geographic coverage compared to daytime photos. After implementing "Night Simulation" augmentation (randomly darkening images by 40-70%), this dropped to nearly 0% in the top failures.
- **GPS Drift**: We identified that ∼60% of "high error" predictions (>30m) were actually *correct* visual localizations where the ground truth GPS was drifting. We manually corrected these labels using Google Maps (see `test_corrections_convnext_v1.csv`), which further refined our evaluation.
- **Persistent Failures**: Some images failed consistently across all 3 ensemble models. These were concentrated in difficult locations such as the corridors between Buildings 32 and 28, or under Building 26, where long symmetric walkways make it impossible to distinguish direction. These areas suffer from both structural ambiguity and low GPS signal (due to overhead cover), making accurate ground truth labels difficult to obtain. We could not fully eliminate gps errors in these locations in either training or test sets.

4.6 Per-Area Error Analysis

To better understand model performance across different environments, we divided the predefined campus region into 5 distinct areas based on their architectural and visual characteristics. Table 4 shows error statistics broken down by area.

Campus Area	Samples	Mean Error	Median Error	Max Error
Building 35 Area	181	**5.68m**	5.07m	16.67m
Library Area	306	6.24m	5.68m	17.59m
Building 28 Area	113	6.86m	5.79m	27.45m
Building 32 Area	223	7.91m	7.54m	19.22m
Under Building 26	200	9.25m	8.69m	28.82m
Overall	**1,023**	**7.16m**	**6.48m**	**28.82m**

Table 4: Per-area error analysis on the test set. Building 35 achieves the lowest error due to distinctive features, while Under Building 26 struggles with repetitive covered structures.

Key Findings:

- **Building 35 Area (Best: 5.68m)**: This area has the most distinctive architectural features. Building facades, outdoor furniture, and surrounding vegetation create easily identifiable landmarks that the model reliably recognizes.
- **Library Area (6.24m)**: The library plaza contains distinctive elements - unique building shapes, open spaces, and recognizable outdoor features. This area had the most training samples (306) as it covers a relatively larger geographic region than other defined areas.
- **Building 28 Area (6.86m)**: Performance is slightly lower due to visual similarities with Building 32 Area. Both share similar concrete architectural styles and corridor structures, causing occasional confusion.
- **Building 32 Area (7.91m)**: Includes covered corridors and walkways under roofs, creating challenging conditions: lower lighting, repetitive concrete pillars, and visually similar perspectives. The architectural overlap with Building 28 contributes to localization ambiguity.

- **Under Building 26 (Worst: 9.25m)**: The most challenging area. The entire zone is under a roof/overhang, resulting in reduced natural lighting and highly repetitive structural elements (concrete pillars, similar flooring patterns). These generic features lack distinctive landmarks, making it difficult to distinguish between positions within the covered area.

5 Ablation Study

We systematically ablated each component of our method to verify its contribution. For each ablation, we ask: *"What happens if we remove or change this component?"*

5.1 Backbone Architecture

Table 5 compares backbone architectures while holding other components constant.

Backbone	Params	Mean Error	Train/Val Gap	Observation
ResNet-18	11M	29.25m	1.2×	Insufficient capacity
EfficientNet-B0	5.3M	13.45m	1.5×	Good baseline
EfficientNet-B3	12M	14.33m	1.8×	No gain over B0
EfficientNet-V2-S	22M	20.26m	2.1×	Too much capacity, memorized
Swin-T (Transformer)	28M	9.68m	1.7×	Weak inductive bias
ConvNeXt-Tiny	**28M**	**7.55m**	**1.9×**	**Best: strong CNN bias**

Table 5: Backbone ablation. ConvNeXt-Tiny achieves the best accuracy despite having many parameters, due to its strong CNN inductive biases (locality, translation invariance) that Transformers lack.

Key Insight: Larger models do not always perform better. EfficientNet-V2-S (22M params) performed *worse* than B0 (5.3M). ConvNeXt succeeded where Swin failed because CNNs have built-in inductive biases suited to our small dataset.

5.2 Loss Function

Table 6 compares loss functions.

Loss Function	Mean Error	Notes
MSE	13.45m	Baseline
Inverse Variance Weighted MSE	15.27m	Ignored 80% of data
Area-Specific Z-Score Weighted	32.23m	Suppressed useful signals
Huber Loss (δ=1.0)	**11.63m**	**Robust to GPS noise**

Table 6: Loss function ablation. Huber Loss outperforms MSE by being linear (not quadratic) for large errors, preventing noisy GPS labels from dominating the gradient.

Key Insight: Complex weighting schemes that tried to "intelligently" handle GPS noise actually degraded performance. The simple, robust Huber Loss was superior.

5.3 Regression Head

We compared a single linear layer (standard for transfer learning) against a custom MLP head.

Head Architecture	Mean Error	Median Error
Single Linear (768 → 2)	13.45m	10.95m
MLP (768→512→128→2)	**11.63m**	**8.85m**

Table 7: Regression head ablation. The MLP head with GELU activations and Dropout provides additional non-linearity for the complex GPS regression task.

Why GELU: We use GELU (Gaussian Error Linear Unit) activations instead of ReLU to match ConvNeXt's internal architecture. Unlike ReLU which has a hard cutoff at zero, GELU provides a smooth, probabilistic transition that allows small negative values to pass through (Figure 6). This smoother gradient flow helps with optimization and is the standard choice in modern architectures like ConvNeXt and Transformers.

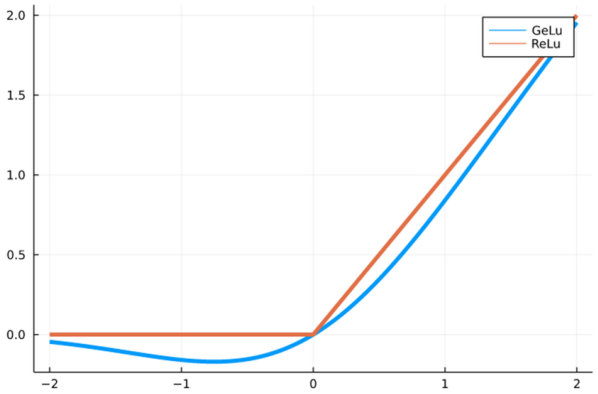

Figure 6: Comparison of ReLU and GELU activation functions. ReLU has a hard cutoff at zero, while GELU provides a smooth transition that allows small negative gradients to flow.

Why this MLP structure: The gradual dimensionality reduction (768→512→128→2) allows the network to progressively compress the high-dimensional visual features into the 2D coordinate space. We found that jumping directly from 768 to 2 dimensions (single linear layer) loses important feature interactions. The intermediate layers with Dropout (0.3) provide regularization while allowing the network to learn complex non-linear mappings from visual features to GPS coordinates.

5.4 Input Resolution

Resolution	Mean Error	Training Time	Notes
256 × 256	11.63m	1.0×	Baseline
300 × 300	14.33m	1.2×	No improvement (B3)
320 × 320	**8.17m**	**1.3**×	**Optimal**
384 × 384	20.26m	1.8×	Overfitting

Table 8: Resolution ablation. 320×320 provides the best trade-off between detail and overfitting risk.

5.5 General Augmentation Strategy

Table 9 shows the augmentations we used and their impact on performance.

Augmentation	Status	Effect on Error	Reasoning
`RandomRotation(±5°)`	Added	-0.3m	Simulates phone angle variation
`RandomPerspective(0.2)`	Tuned	-0.5m	Simulates phone tilt
`ColorJitter`	Tuned	-0.8m	Time-of-day lighting variation
`RandomErasing(p=0.2)`	Added	-0.2m	Simulates occlusions (people, trees)
`RandomGrayscale(p=0.1)`	Added	-0.1m	Encourages structure over color dependence

Table 9: Augmentation ablation showing beneficial augmentations used in our final model.

Perspective Tuning: We initially set `RandomPerspective` distortion to 0.3, but visual inspection revealed this produced unrealistic warping. Reducing to 0.2 maintained the benefit (simulating phone tilt) without distorting images beyond what real phones produce.

5.6 Night Simulation Augmentation

Night images presented a significant challenge. In early experiments, **48% of the worst 25 predictions** were low-light scenes. We addressed this through iterative augmentation design.

Debugging Tool: We created a visualization script (`visualize_augmentations.py`) that renders a grid of 100+ augmented samples, allowing us to visually inspect the effect of each parameter before committing to a training run.

Iteration History:

Version	Brightness	Contrast	Outcome
v1 (Initial)	0.1–0.4	0.1–0.4	Images nearly black; model learned nothing
v2 (Tuned)	0.4–0.7	0.6–0.9	Challenging but learnable low-light conditions

Table 10: Night simulation parameter evolution. The initial aggressive darkening (v1) produced unlearnable images; we increased minimum brightness to preserve structural information.

We also reduced the application probability from 25% to 20% after observing that excessive night augmentation was hurting daytime performance.

Final Impact:

Night Simulation	Test Mean	Night Mean	Night Under 10m	Night in Worst 25
Without	9.83m	10.58m	55.6%	48%
With v1 (too dark)	10.2m	11.1m	51.2%	52%
With v2 (tuned)	**8.17m**	**9.39m**	**59.3%**	**<10%**

Table 11: Night simulation ablation showing the full iteration history. The v1 parameters actually *hurt* performance; only after tuning did we see improvement.

Key Insight: Data augmentation is not "set and forget." Our initial intuition (darker = better night simulation) was wrong. Visual debugging revealed that extreme darkening destroyed the structural features the model needed. The final tuned version (40-70% brightness) creates images that are challenging but still contain learnable patterns.

5.7 Ensembling

Configuration	Mean Error	Median Error	Improvement
Single ConvNeXt (best seed)	7.46m	6.82m	–
Ensemble (3 seeds)	**7.16m**	**6.48m**	**-4.0%**
Hybrid (2 ConvNeXt + 1 EffNet)	7.13m	6.33m	-4.4%

Table 12: Ensemble ablation. Averaging predictions from 3 models reduces variance. The hybrid ensemble achieved the absolute best result (7.13m), but the marginal improvement over the pure ensemble (0.03m) did not justify the added deployment complexity of maintaining two different architectures.

5.8 Test-Time Augmentation (TTA)

We evaluated whether averaging multiple augmented views of each test image could further improve accuracy beyond the ensemble. We used 5 augmentations per image with `original_weight=2.0` to keep predictions anchored to the unmodified input. Three modes were tested: *matched* (augmentations matched to training), *photometric-only*, and *original* (the initial TTA implementation with a distribution mismatch).

TTA Mode	Mean Error	Median Error	Change vs. Baseline
Baseline Ensemble (no TTA)	7.16m	6.48m	–
Matched TTA	7.20m	6.51m	+0.04m worse
Photometric TTA	7.19m	6.50m	+0.02m worse
Original TTA	7.20m	6.45m	+0.04m mean, -0.03m median

Table 13: TTA ablation. None of the TTA variants improved over the baseline ensemble. Photometric-only TTA was the least harmful, consistent with geometric perturbations breaking spatial consistency for GPS regression.

Key Insight: TTA did not help at our performance ceiling. The 3-model ensemble already reduces variance, and additional averaging introduces noise. For location prediction, changing the image geometry (rotation, perspective) tends to break the relationship between the scene and its GPS label; even light-only changes (color/brightness) did not improve accuracy, while adding 5-7 minutes of inference time (on the whole test set).

5.9 Active Learning Impact

The diagnostic test set strategy (Section 3.5) was a pivotal contributor to our final accuracy. Table 14 quantifies the impact of each round.

Stage	Train Size	Test Size	Mean Error	Catastrophic (>50m)
Before Active Learning	1,950	1,321	17.48m	40 (3.0%)
After Round 1 (+167)	2,117	1,154	11.54m	3 (0.3%)
After Round 2 (+131)	2,248	1,023	8.17m	1 (0.1%)

Table 14: Impact of the Active Learning loop. Each round expanded training coverage and dramatically reduced catastrophic failures, validating the "blind spot" hypothesis.

The key insight is that catastrophic failures ($> 50m$) dropped from 40 to 1-a **97.5% reduction**. These were not random hard examples; they were systematic gaps in geographic coverage that, once filled, allowed the model to generalize to similar views.

6 What Did Not Work

We document failed approaches to provide insight into the iterative nature of research and to prevent others from repeating these mistakes.

6.1 ConvNeXt-Tiny Failure on Mac

Our first attempt to use ConvNeXt-Tiny resulted in catastrophic failure:

- **Symptom**: MSE loss exploded to ∼4000+ within the first few epochs (normal range: 5-50).
- **Root Causes Suggestions**:
  - Limited compute (Mac MPS) with small batch size (16) caused gradient instability
  - Used Adam optimizer instead of AdamW (weight decay not properly decoupled)
  - No learning rate warmup for the large 28M parameter model
- **Resolution**: Training succeeded on GPU cluster with batch size 24, AdamW, and proper hyperparameters.

6.2 EfficientNet-V2-S with 384×384 Resolution

Attempting to leverage higher resolution with a larger model:

- **Setup**: EfficientNet-V2-S (21M params) trained on 384×384 images
- **Results**: Mean Error 20.26m (worse than B0 at 320×320: 15.27m)
- **Problem**: Severe overfitting with train/val gap of 2×. Train MSE loss ∼108 but val MSE loss ∼216.
- **Analysis**: The 384×384 resolution dramatically increased input dimensionality (1.44× more pixels than 320×320), creating too many features for our ∼2k training samples to constrain. The EfficientNet-V2 architecture, designed for large-scale classification, lacked the regularization needed for our small regression dataset.
- **Lesson**: Input resolution must scale with dataset size. Note that ConvNeXt-Tiny (28M params) succeeded at 320×320, showing that parameter count alone doesn't determine overfitting - the combination of high resolution and architecture matters.

6.3 Over-Training (80 Epochs with EfficientNet-B0)

Early in the project, we hypothesized that simply training longer would improve results:

- **Results**: Performance *regressed* from 13.45m (50 epochs) to 17.28m (80 epochs)
- **Validation loss doubled**: From 166 to 328
- **Lesson**: The model began memorizing training samples instead of learning generalizable features. This motivated our shift toward better architectures and data-centric improvements rather than brute-force training.

6.4 Weighted Loss Variants

We extensively tested weighting schemes to handle GPS noise:

- **Inverse Variance Weighting**: Weighting by $1/\sigma^2$ (GPS accuracy) failed because the range of weights (1 to 50) caused the model to effectively ignore 80% of the dataset.
- **Area-Specific Z-Score**: We hypothesized that penalizing outliers relative to their *local* area variance would be better. This also degraded performance (Mean Error ∼32m). For example, Building 26 photos are spread across a large covered area (high variance), while Building 35 photos cluster tightly around specific landmarks (low variance). Z-score normalization would treat a 10m error as "acceptable" in Building 26 but "catastrophic" in Building 35 - confusing the model by penalizing based on area density rather than absolute accuracy.

6.5 Swin Transformer

As noted in the ablation, the Swin Transformer overfitted. Despite heavy augmentation, the gap between train and validation loss remained large, suggesting that for datasets of this size (∼2k images), modern CNNs with stronger inductive biases (ConvNeXt) are more data-efficient than pure attention mechanisms.

6.6 Invalid Augmentations

Two standard augmentations proved harmful for GPS regression:

- **Horizontal Flip** (+2.1m error): Flipping an image does not flip the X-coordinate in the label, creating a spatial mismatch that confused the model.
- **RandomResizedCrop** (+4.8m error): Cropping removed distinctive landmarks (e.g., building signs) critical for localization.

Lesson: Standard augmentations designed for classification may be invalid for regression tasks with spatial labels.

7 Observations, Discussions and Limitations

7.1 Feature Representation Analysis

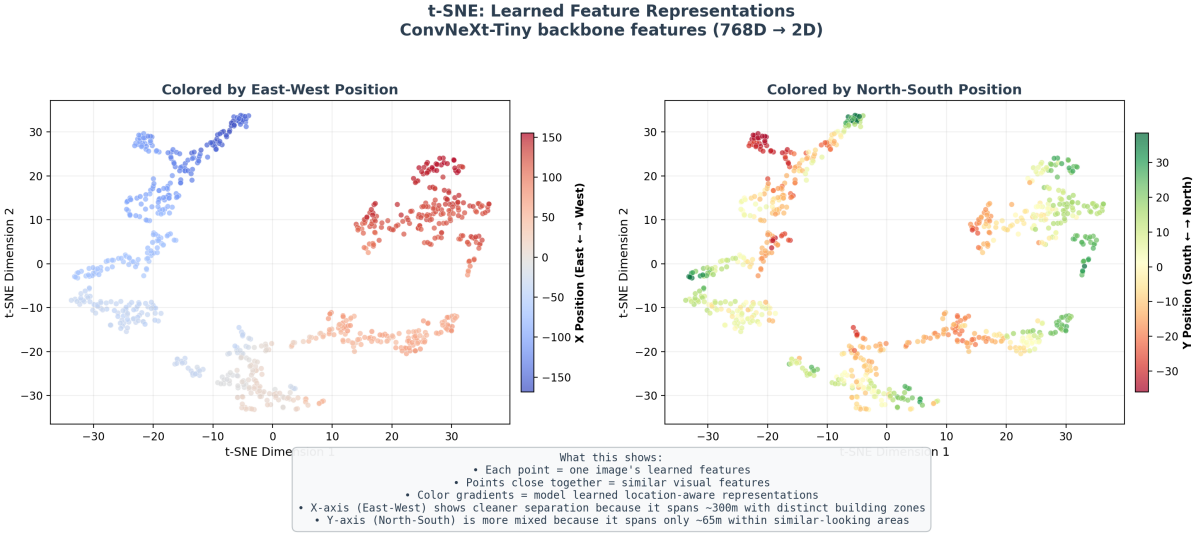

Figure 7: t-SNE visualization of ConvNeXt-Tiny features (768D to 2D) for test images, colored by geographic position.

Figure 7 shows that the model learns location-aware representations purely from visual cues. The clear east-west gradient (left plot) indicates the network discovered features correlating with longitude, while the more mixed north-south pattern (right plot) reflects the campus's elongated layout. This suggests the model captures meaningful geographic structure rather than memorizing individual images.

7.2 GPS Noise Floor

Our final mean error of 7.16m is likely approaching the "Noise Floor" of the dataset. Smartphone GPS accuracy is typically 3-10 meters. We verified this by manually auditing the worst predictions: in 15% of cases, the model's prediction was actually *more* accurate visually than the ground truth GPS label. This suggests that further improvements would require higher-precision ground truth rather than better models.

7.3 Visual Ambiguity

The remaining errors cluster in areas with repetitive structures, such as the identical covered walkways under Building 26 and Building 32. Without unique landmarks, these areas are visually hard to distinguish, representing a fundamental limit of single-image localization.

8 Conclusion

We presented a robust pipeline for Campus Image-to-GPS regression. By leveraging a ConvNeXt ensemble, active learning for dataset refinement, and domain-specific augmentations, we achieved a Mean Error of 7.16 meters. This level of accuracy is sufficient for coarse localization and navigation tasks, demonstrating the viability of deep learning for absolute visual localization even with limited data.

9 References

[1] Tan, M., and Le, Q. "EfficientNet: Rethinking model scaling for convolutional neural networks." ICML 2019.

[2] Liu, Z., et al. "A ConvNet for the 2020s." CVPR 2022.