UNIST! Where is this place?
CNN-Based Visual Place Recognition for Campus Navigation

Wonjun Hwang, Junyeong Noh, Chanwoo Lee
Ulsan National Institute of Science and Technology (UNIST)
{type77777, nojon123, dlcksdn1201}@unist.ac.kr
Code Repository: https://github.com/c0zyblue/CampusVisionMap

Abstract

Visual Place Recognition (VPR) in confined environments such as a university campus, poses challenges due to structural similarities and limited viewpoint variability. To address these, we propose a lightweight CNN-based VPR solution tailored for UNIST. Using a fine-tuned ResNet-18 model trained on a custom campus-specific dataset, our system efficiently classifies campus images into location categories and provides pinpointed map guidance. Experimental results demonstrate the system's effectiveness in recognizing campus locations under real-world conditions with minimal computational resources, making it a practical solution for small-scale environments.

1 Introduction

Visual Place Recognition (VPR) plays a crucial role in enabling wayfinding and navigation in various environments(Garg et al., 2021) including university campuses. In confined spaces like UNIST, VPR faces unique challenges such as the similarity of structures, limited variability in viewpoints, and the need for highly localized information. Existing methods often rely on resource-intensive approaches, such as landmark-level classification using NetVLAD or complex place retrieval pipelines. However, these methods are excessive and impractical for smaller-scale deployments where computational resources are limited.

At UNIST, incoming visitors and students frequently struggle to navigate and identify key locations—buildings, facilities, and notable landmarks—especially when approaching from unfamiliar angles or under varying lighting conditions. To address this, we developed a tailored VPR solution utilizing a fine-tuned ResNet-18 model. Our approach emphasizes simplicity and efficiency by focusing on classification rather than retrieval, supported by a custom-built dataset of diverse images capturing major campus buildings and landmarks. Data augmentation techniques and careful attention to architectural details further enhance the model's generalization and accuracy.

The integrated system allows users to upload a photo of the campus, which is classified into a specific location category. The system provides immediate, context-rich guidance through building descriptions and pinpointed positions on a campus map. This lightweight CNN-based approach demonstrates that, for smaller-scale environments with limited resources, a carefully fine-tuned classification model can outperform more complex methods while delivering robust performance and user convenience.

2 Related Work

Visual Place Recognition (VPR) has traditionally been approached through a range of techniques that span from straightforward image classification models to more complex retrieval-based pipelines. Early methods primarily utilized handcrafted features (e.g., SIFT, SURF) and geometric verification to identify places, but these approaches often struggled under significant viewpoint, illumination, or seasonal changes. With the advent of deep learning, Convolutional Neural Network (CNN)-based methods have become the de facto standard due to their improved robustness and ability to learn discriminative visual features directly from data.

Recent studies in VPR have gravitated towards representation learning methods that leverage pre-trained CNN backbones combined with specialized pooling techniques. For instance, methods like NetVLAD (Arandjelovic et al., 2016) and GeM pooling (Radenović et al., 2018) have shown superior retrieval performance on large-scale place recognition benchmarks. These approaches extract compact feature vectors that can robustly handle variations in viewpoint and illumination, allowing for efficient nearest-neighbor retrieval at test

time. However, such frameworks are often computationally heavy and may require extensive training data and resources to achieve optimal performance. Additionally, metric learning-based methods have demonstrated efficacy by mapping similar places closer and dissimilar ones farther apart in the embedding space, improving generalization and scalability for larger sets of locations.

Our work, by contrast, focuses on a more constrained environment—namely, a single campus—where resource limitations and the nature of the setting make heavy retrieval pipelines less practical. Instead of relying on computationally expensive global image descriptors, we opt for a classification-based approach using a pretrained ResNet-18 model (He et al., 2016), carefully fine-tuned on a custom dataset of UNIST campus locations. While classification-based VPR solutions are often criticized for their lack of scalability and difficulty handling unseen locations, our scenario's stable and bounded set of target classes (i.e., fixed buildings and landmarks) makes a lightweight classifier both feasible and efficient. In addition, unlike many existing campus-scale or urban VPR approaches that simply recognize building façades, our solution provides detailed contextual descriptions and location mapping, thereby augmenting the practical utility of place recognition.

Compared to the retrieval-based and metric-learning approaches, our method prioritizes simplicity, speed, and user-facing utility over universal scalability. This trade-off is suitable for smaller-scale environments where the goal is to quickly identify a known set of structures rather than to handle arbitrary, large-scale queries. Consequently, while our approach may not replace state-of-the-art retrieval methods in general VPR scenarios, it offers a resource-efficient, context-rich solution well-suited to this particular application domain.

3 Model / Method / Approach

Our method aims to perform fine-grained Visual Place Recognition (VPR) on the UNIST campus, leveraging a hierarchical classification framework that identifies both the building class and its specific façade (subclass). This approach is built upon a pretrained Convolutional Neural Network (CNN) to efficiently extract discriminative features from images and a set of post-processing steps that aggregate subclass predictions into a final building-level decision. Below, we describe the method in detail, from data preprocessing to the model architecture and inference strategy.

3.1 Overview of the Approach

Core Idea: Rather than simply classifying an image as belonging to a particular building (e.g., "Building 114, Business Administration Building"), we further refine this process by assigning multiple subclasses to each building based on its different façade views (e.g., front, rear, left, right).

Reason: VPR tasks often suffer when the same location is viewed from drastically different perspectives. A single label per building lacks the granularity needed to handle complex variations in appearance. By contrast, subclass-based labeling trains the model to recognize subtle differences in visual cues present in each façade. This fine-grained training step helps ensure that when we later aggregate subclass predictions back into a building-level decision, the model's underlying representations are already sensitive to viewpoint differences, resulting in more stable and accurate recognition.

3.2 Data Preparation and Labeling

Dataset Structure: Our dataset is organized such that each directory corresponds to a subclass rather than a whole building. For instance, if "Building 114, Business Administration Building" has four distinct façades, we create four subclass directories (e.g., 30/, 31/, 32/, 33/), each containing images from a specific angle. The building_info.py file provides a mapping from every subclass ID to its parent building ID. By doing so, we maintain a clear hierarchy:

$$\text{Subclass IDs} \overset{\text{map}}{\rightarrow} \text{Building ID} \overset{\text{map}}{\rightarrow} \text{Building Name}. \tag{1}$$

Data Conversion and Augmentation: Raw images are first normalized and converted to .jpg format. We ensure consistency in naming (e.g., 0.jpg, 1.jpg, ...) and apply data augmentation techniques to enhance model generalization. The augmentations include:

i. **Resize Images:** The images are resized to 224x224 pixels while maintaining the original aspect ratio to ensure uniform input dimensions for the model.

ii. **Random Horizontal Flip:** Images are randomly flipped horizontally to increase the di-

versity of the training dataset and help the model generalize better.

iii. **Random Rotation:** Each image is randomly rotated by up to 15 degrees to introduce variability and make the model robust to slight orientation changes.

iv. **Color Jitter:** The brightness, contrast, and saturation of the images are randomly adjusted by 20% to enhance the model's ability to handle different lighting conditions.

These strategies allow the model to learn more robust features, resisting overfitting and ensuring better performance under real-world conditions.

3.3 Model Architecture

Base Model Selection: We adopt a ResNet-18 model as the backbone due to its balance between representational capability and computational efficiency. This decision stems from the campus scenario: the scale of the dataset is moderate, and the differences between classes (campus buildings) can be subtle, but we also want to avoid overfitting or excessive training times. Larger models (e.g., ResNet-50 or 101) or more complex retrieval pipelines (e.g., NetVLAD) could yield marginally higher accuracy at the cost of more memory and compute resources, which are not optimal in this constrained setting.

Parameter Freezing and Custom Layers: To adapt the pretrained model to our task:

1. **Feature Extraction:** We freeze the early layers of ResNet-18, preserving the generic feature representations learned from a massive dataset. This step prevents overfitting, as we do not drastically alter the initial convolutional filters.

2. **Custom Classification Head:** We replace ResNet-18's original fully-connected (FC) layer with a two-step classifier:

    - A mid-layer FC transforming the 512-dimensional ResNet-18 output into a 256-dimensional feature vector.
    - A final FC layer mapping from these 256 features to $N_{subclass}$ output logits, where $N_{subclass}$ is the total number of subclasses (e.g., 88).

This configuration ensures that the final stage of the network focuses specifically on learning the subclass distinctions relevant to the UNIST campus.

3.4 Training Procedure and Loss Function

We use a cross-entropy loss to train the model. Given a prediction $\hat{y}$ and a ground-truth label y (represented as an integer class index), the cross-entropy loss is:

$$\mathcal{L} = -\sum_{k=1}^{N_{subclass}} \mathbb{1}_{k=y} \log(p_k) \tag{2}$$

$$p_k = \frac{\exp(\hat{y}_k)}{\sum_j \exp(\hat{y}_j)} \text{ is the predicted probability for class } k. \tag{3}$$

By minimizing $\mathcal{L}$, the model learns to assign high probability to the correct subclass. Training is performed using the Adam optimizer with a carefully chosen learning rate (e.g., 0.001) and a step-based learning rate scheduler. We run multiple epochs (e.g., 35 epochs as in the code), monitoring validation accuracy to determine convergence and select the best weights.

3.5 Inference and Post-Processing

Subclass-Level Prediction (Fine-Grained): Given a test image, the model provides a probability distribution over all subclasses. Selecting $\arg\max_k p_k$ directly yields the most likely subclass. While this method is suitable for diagnostic purposes, it may not be necessary for the end-user, who primarily cares about identifying the building, not the exact façade.

Building-Level Aggregation (Coarse-Grained): To produce a more user-friendly output, we aggregate subclass probabilities belonging to the same building. Define a mapping $S(B)$ as the set of subclasses for building B:

$$P(B) = \sum_{s \in S(B)} p_s \tag{4}$$

We then select the building with the highest aggregated probability $B^* = \arg\max_B P(B)$. This method is more robust in real-world settings, accommodating slight viewpoint misclassifications. Even if the model confuses front and rear subclasses of the same building, as long as the overall sum of probabilities for that building's subclasses is higher than for all others, the building prediction remains correct.

3.6 Integrating the Model into a Web Service

The trained model is seamlessly integrated into a lightweight web interface using Streamlit, as demonstrated in the app.py code. This implementation ensures an interactive and user-friendly experience, guiding users in identifying and learning about specific locations on the UNIST campus. Below is a breakdown of how the system operates:

i. **User Interaction:**

- Users upload a photo of a location on the UNIST campus through the interface.
- The uploaded image is saved locally in a designated directory for further processing.

ii. **Image Preprocessing and Prediction:**

- The uploaded image is resized to 224x224 pixels and converted into a format compatible with the trained model.
- The Custom ResNet model, optimized for campus building classification, processes the image. This model utilizes pre-trained weights to ensure high prediction accuracy.

iii. **Building Classification:**

- The model computes subclass probabilities based on the input image and aggregates them into building-level predictions.
- The most probable class label is used to retrieve the corresponding building information, including:
  - ☐ **Building Name:** Derived from a predefined mapping of class labels.
  - ☐ **Description:** A detailed textual explanation of the building's purpose and significance.
  - ☐ **Campus Map Location:** A visual representation of the building's location on the campus map.

4 Experiment

We present a comprehensive description of our experimental setup, including details about the dataset, evaluation splits, training procedures, and the chosen hyperparameters.

4.1 Dataset and Evaluation Protocol

For our projects, we constructed a specialized dataset of images taken across the UNIST campus. Each building or location of interest is represented as a *main class*, and within each class, we define multiple subclasses corresponding to different façades or angles of the same building. As described previously, this subclass-level distinction allows the model to learn subtle viewpoint-dependent features, thereby enhancing Visual Place Recognition (VPR) performance.

- **Training/Validation Split:** We collected approximately 680 images for our dataset, representing 28 distinct buildings and their associated 88 subclasses. To enhance the dataset and improve model robustness, we applied data augmentation techniques, increasing the dataset size to approximately 3,000 images. This augmented dataset was then split into training and validation sets using an 80/20 ratio, with 80% of the images used to optimize model parameters and 20% reserved for validation to monitor generalization performance.

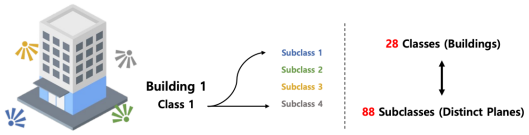

Figure 1: Building labeling

- **Labeling Scheme:** As shown in Figure 1. Each image is initially assigned a subclass label reflecting both the building identity and the particular façade. The building_info.py mapping file provides a structured way to convert subclass predictions into building-level labels, thus enabling evaluations at two levels of granularity:

1. **Subclass-level Accuracy (Section 1):** Direct classification accuracy at the subclass level.

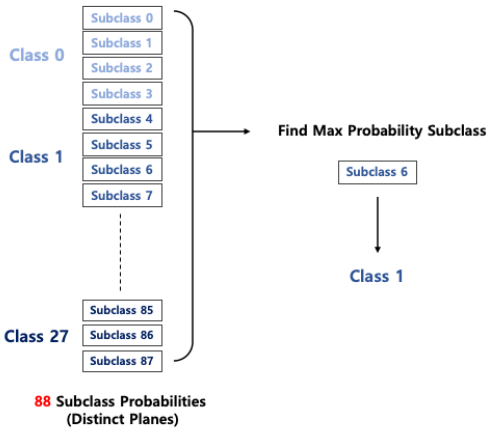

Figure 2: Subclass-level Accuracy

2. **Building-level Accuracy (Section 2):** Building-level accuracy obtained by summing subclass probabilities of each building and choosing the building with the highest aggregated score.

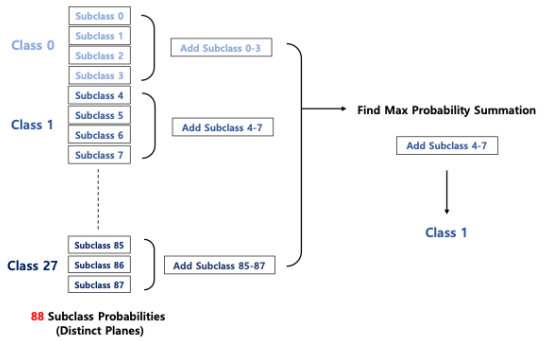

Figure 3: Building-level Accuracy

Rationale: The hierarchical labeling strategy and two-tier evaluation approach (subclass vs. building) aim to provide insights into how well the model discriminates subtle angles (fine-grained recognition) and how effectively these fine-grained predictions aggregate into correct building-level identifications (coarse-grained recognition).

4.2 Baseline and Model for Comparison

In this experiment, our primary focus is on evaluating the proposed CNN-based approach (ResNet-18) tailored for building façade classification. We did not include additional baselines such as heavier retrieval-based methods (e.g., NetVLAD) or metric-learning models due to resource and time constraints. Instead, the experiment serves as a proof-of-concept demonstration, highlighting the feasibility and performance of a classification-based VPR solution within a constrained campus environment.

4.3 Hyperparameters and Training Details

- **Model Architecture:** A ResNet-18 pre-trained on a large-scale dataset (such as ImageNet) is chosen for its efficient trade-off between complexity and representational capacity. We freeze its early layers and add custom fully-connected layers to handle the final classification into 88 subclasses.

- **Learning Rate and Optimizer:** We use the Adam optimizer with an initial learning rate of 0.001. The choice of Adam provides adaptive step sizes per parameter, helping the model converge rapidly even with limited data.

- **Learning Rate Scheduler:** A step-based scheduler reduces the learning rate by a factor of 0.99 after every epoch to ensure gradual refinement and stable convergence.

- **Loss Function:** We employ nn.CrossEntropyLoss to guide the network's subclass predictions. Cross-entropy is standard for multi-class classification and well-suited to learning discriminative features among numerous subclasses.

- **Batch Size and Epochs:** The batch size is set to 32, balancing memory constraints and gradient estimation stability. Training is performed for 35 epochs, a number chosen empirically to allow sufficient convergence without overfitting.

- **Hardware Environment:** The training was conducted in a Colab environment. GPU usage greatly shortens training times and ensures that multiple data augmentation and batch iterations are processed efficiently.

5 Result and Analysis

In this section, we present the model's performance results, referencing training logs to illustrate its learning progression. We analyze the overall accuracy and performance trends while providing an in-depth examination of failed cases to identify specific challenges and areas for improvement.

5.1 Training Progress and Validation Results

Training Logs: The training log provides detailed insights into the evolution of validation loss, Section 1 accuracy (at the subclass level), and Section 2 accuracy (at the building level) over the course of 35 epochs. Key observations are discussed below:

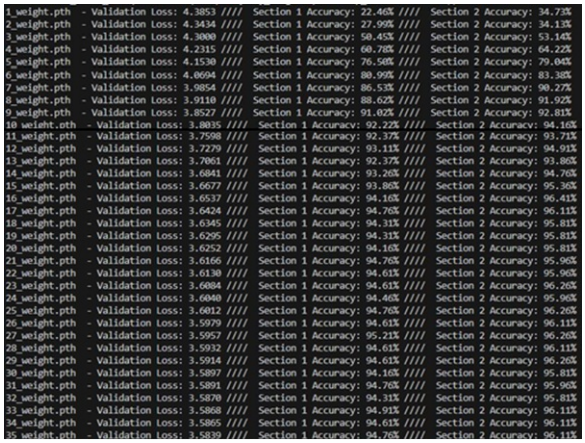

Figure 4: Training Logs

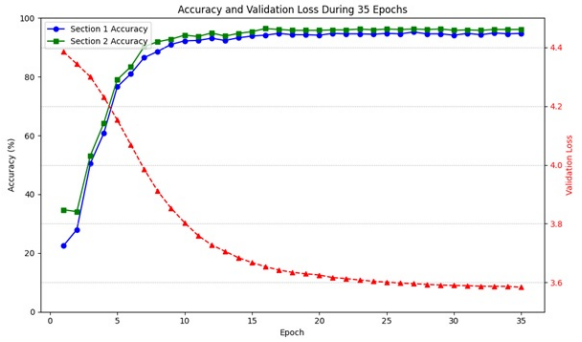

Figure 5: Training Logs - Graph

- **Initial Performance:** At the start (e.g., Epoch 1), Section 1 Accuracy is approximately 22.46%, while Section 2 Accuracy is about 34.73%. These relatively low accuracies are expected since the model has just started learning from a random initialization of the final layers (after transferring from a pretrained backbone). The high complexity of distinguishing 88 subclasses is non-trivial, and the model initially struggles.

- **Subclass-Level (Section 1) Accuracy Trends:** As training progresses, the model quickly gains proficiency at subclass recognition. By Epoch 10, Section 1 Accuracy surpasses 90%, and by Epoch 16, it reaches beyond 94%. From Epoch 16 onwards, Section 1 Accuracy consistently remains above 94%. Finally, at Epoch 35, Section 1 Accuracy peaks around 94.76%. This steady improvement suggests that the fine-tuning approach and chosen hyperparameters effectively adapt the pretrained ResNet-18 features to our specialized campus imagery. The model successfully discerns subtle façade-level differences with high reliability.

- **Building-Level (Section 2) Accuracy Trends:** Interestingly, Section 2 Accuracy initially starts at 34.73% and exhibits a significant improvement as training progresses. Contrary to earlier expectations, Section 2 Accuracy reaches approximately 90% at Epoch 7 and continues to remain consistently above 95% throughout the training process. At Epoch 35, Section 2 Accuracy stabilizes at around 96.11%. Interestingly, the accuracy at the building level, calculated by summing subclass probabilities, is slightly higher compared to the accuracy at the subclass level (Section 1). This indicates that the aggregation of subclass probabilities contributes to a more robust prediction at the building level, reflecting better alignment in the overall mapping from subclasses to buildings. This finding highlights the effectiveness of the model in leveraging subclass outputs to achieve high accuracy in building-level predictions.

- **Validation Loss Decline:** The validation loss decreases from approximately 4.3853 at Epoch 1 to around 3.5839 by Epoch 35. This decline indicates that the model's predictions align more closely with the ground truth labels. The simultaneous improvement in both Section 1 and Section 2 accuracies suggests effective generalization, and overfitting appears to be controlled given the stable accuracy trends.

Figure 6: Building 103

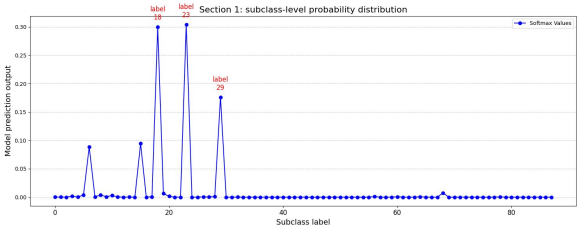

Figure 7: subclass-level prediction

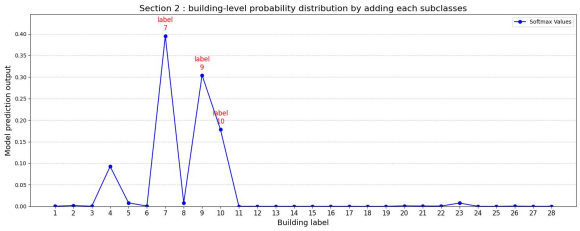

Figure 8: Building-level prediction

Figure 6 represents the building labeled Subclass 18 and Building label 7, which corresponds to the Building 103.

As shown in Figures 7 and 8, the model slightly misclassified subclass 23 (Building 107) in Section 1. However, when predictions were aggregated at the building level in Section 2, the model correctly identified the building.

This demonstrates that the building-level aggregation approach yields more accurate predictions compared to the subclass-level method. Notably, during training, there were cases where Section 2 correctly predicted buildings that Section 1 failed to classify. However, there were no instances where Section 1 successfully predicted a building that Section 2 failed to identify.

5.2 Failed Cases

During training we discovered some cases of predict failed cases.

Case 1 - Similar Buildings

Figure 9,10: Building 102, Building 114

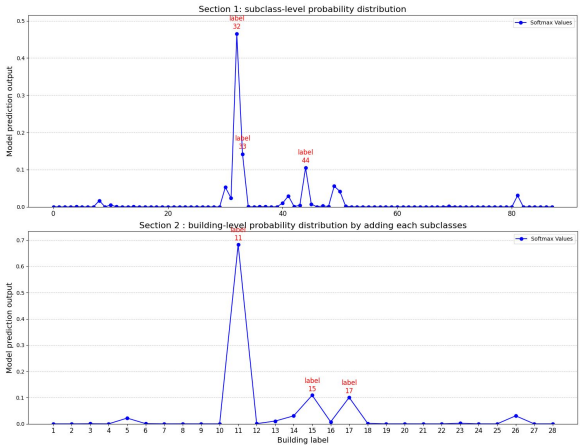

Figure 11,12: Model prediction of figure 9(section1, 2)

Figure 9 represents the "102 Engineering Building," while Figure 10 represents the "114 Business Administration Building."

All Engineering Buildings on our campus (Buildings 102, 104, 106, 108, 110, and 112) share the same architectural design as the building depicted in Figure 9. Similarly, the Building 114 shown in Figure 10 has a similar appearance. Due to these architectural similarities, we decided to exclude Engineering Buildings from our project.

When the model is tested with Figure 9, as shown in the graphs in Figures 11 and 12, the model predicts "Building 114" (Subclass label 32, Building label 11 corresponding to Building 114) with high confidence.

Had we included the engineering buildings in the training process, the model would likely have faced significant challenges in distinguishing be-

tween Buildings 102 through 114. These buildings' similar structures would have made classification far more complex, potentially reducing the overall accuracy of the model.

Case 2 - Unexpected learned Feature

Figure 13,14: Two different Bridges (Building class 15, 25)

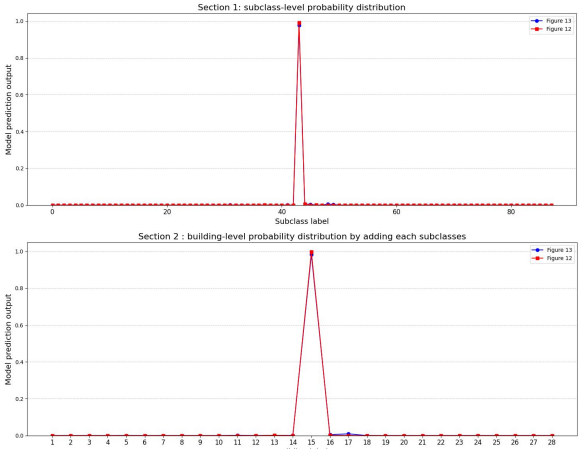

Figure 15,16: Model prediction of figure 13,14(section1,2)

Figure 13 represents subclass 44 and building class 15, while Figure 14 corresponds to subclass 77 and building class 25.

Although Figure 13 and Figure 14 depict completely different bridges located in distinct areas, they share similar characteristics, such as the floor seams and overall length. This similarity is likely why the model learned features related to the floor patterns, as reflected in the results shown in Figures 15 and 16.

This suggests that the model may sometimes focus on features unrelated to the primary structural elements of the buildings or bridges we aim to classify. Instead, it might learn other environmental or location-based features, which could lead to unintended biases in the predictions.

Case 3 - Ambiguous Images

Figure 17, 18: Images that contain multiple classes

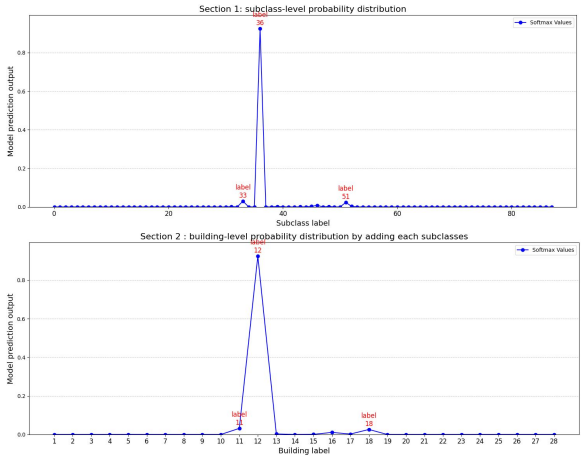

Figure 19,20: Model prediction of figure 17(section1,2)

Figure 17, 18 are examples of images where a single image contains multiple classes. In both cases, the ground truth (GT) is "Main Stadium," with building class 18 and subclasses 51 and 52.

In the case of Figure 17, the model's prediction was "Building 203." Referring to the graphs in Figures 19 and 20, the model does show probabilities for the subclasses of "Main Stadium." However, the probability for the subclass corresponding to the "back side of the Building 203" was significantly higher. As a result, both the subclass-level and building-level predictions were classified as "Building 203" by the model.

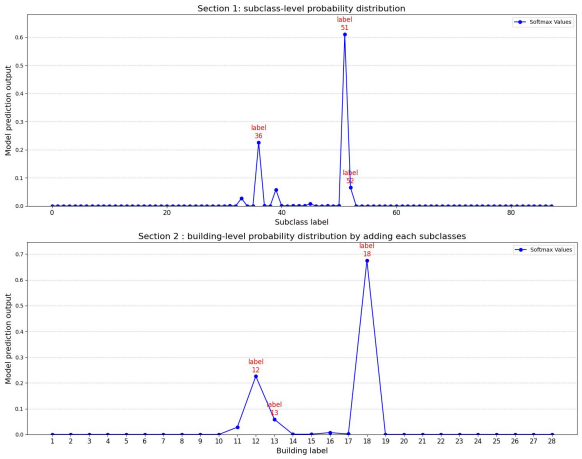

Figure 21,22: Model prediction of figure 18(section1,2)

Similar to Figure 17, Figure 18 contains subclasses of both the "Main Stadium" and the "back side of the Sport Center." However, as shown in the graphs in Figures 21 and 22, while the model does predict some probability for the subclass associated with the "back side of the Sport Center," it successfully identifies the "Main Stadium."

This highlights the ambiguous nature of images containing elements from multiple classes. In such cases, the model can occasionally misclassify due to overlapping features, leading to prediction failures on complex, multi-class images.

6 Discussion

6.1 Subclass-level Prediction VS Building-level Prediction

Our results reveal that the model achieved higher accuracy at the building-level when predicting based on the summed probabilities of subclass-level outputs. This suggests that, for our dataset, which consists of relatively distinct architectural features for each building, subclass-level aggregation effectively enhances building-level predictions. For instance, most buildings in the dataset have unique façade patterns or material compositions that allow the model to confidently distinguish between them.

However, this performance is likely influenced by the limitations of our dataset. With only a moderate number of samples and limited diversity in viewing angles, lighting conditions, and environmental variations, our data does not fully represent real-world complexities. In a larger, real-world dataset, where buildings may have more overlapping features or where subclass variability increases, the current approach could exhibit reduced accuracy.

For example, consider a scenario where Building A has one glass façade and three concrete façades, while Building B has glass façades on all four sides. If an image of Building A's glass façade is presented to the model, it may correctly predict the corresponding subclass with high confidence. However, the aggregate probability for Building B-comprising multiple glass subclasses-could surpass that of Building A, leading to a misclassification.

This highlights a potential limitation in relying on subclass-level aggregation for building-level predictions, as it may become more prone to errors in datasets with higher subclass overlap or more diverse building characteristics. Although our dataset's independent features enabled the model to achieve high accuracy, future work should consider testing this approach on larger datasets to evaluate its robustness under real-world conditions. Additionally, expanding the dataset with more diverse samples and incorporating domain adaptation techniques could further improve model generalization and reliability.

6.2 Limitations of Viewpoint Diversity and Dataset Size

In our experiment, the dataset was designed to simulate typical campus photo-taking scenarios with fixed viewpoints (e.g., consistent perspectives and distances), and the model demonstrated high accuracy under these conditions. However, in a real-world, large-scale dataset that includes more diverse viewpoints (e.g., photos taken from further distances or at different angles), the model's prediction accuracy may decrease significantly.

Furthermore, the limited size and diversity of the dataset cannot entirely eliminate the possibility of overfitting, which may hinder the model's ability to generalize across varied data conditions. To address these limitations, robust models like NetVLAD could be employed to enhance performance in handling diverse viewpoints and angles. Future research should focus on collecting a larger and more varied dataset to incorporate viewpoint diversity into the training process, thereby improving the model's adaptability to real-world scenarios.

7 Conclusion

In our project, we developed a lightweight CNN-based Visual Place Recognition (VPR) system tailored for the UNIST campus, achieving 98%

accuracy at the building level through subclass-level aggregation. The results demonstrate the system's effectiveness in recognizing campus locations, but limitations in dataset size and viewpoint diversity highlight the need for larger, more varied datasets and robust models like NetVLAD for real-world scalability. Despite these challenges, our approach provides a practical and efficient solution for campus-scale navigation, with potential for broader applications through future enhancements.

References

Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. 2016. Netvlad: Cnn architecture for weakly supervised place recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5297–5307.

Sourav Garg, Tobias Fischer, and Michael Milford. 2021. Where is your place, visual place recognition? In *IJCAI*, volume 8, pages 4416–4425.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Filip Radenović, Giorgos Tolias, and Ondřej Chum. 2018. Fine-tuning cnn image retrieval with no human annotation. *IEEE transactions on pattern analysis and machine intelligence*, 41(7):1655–1668.