

Resource-aware and Robust Visual Place Recognition for Real-World Environments

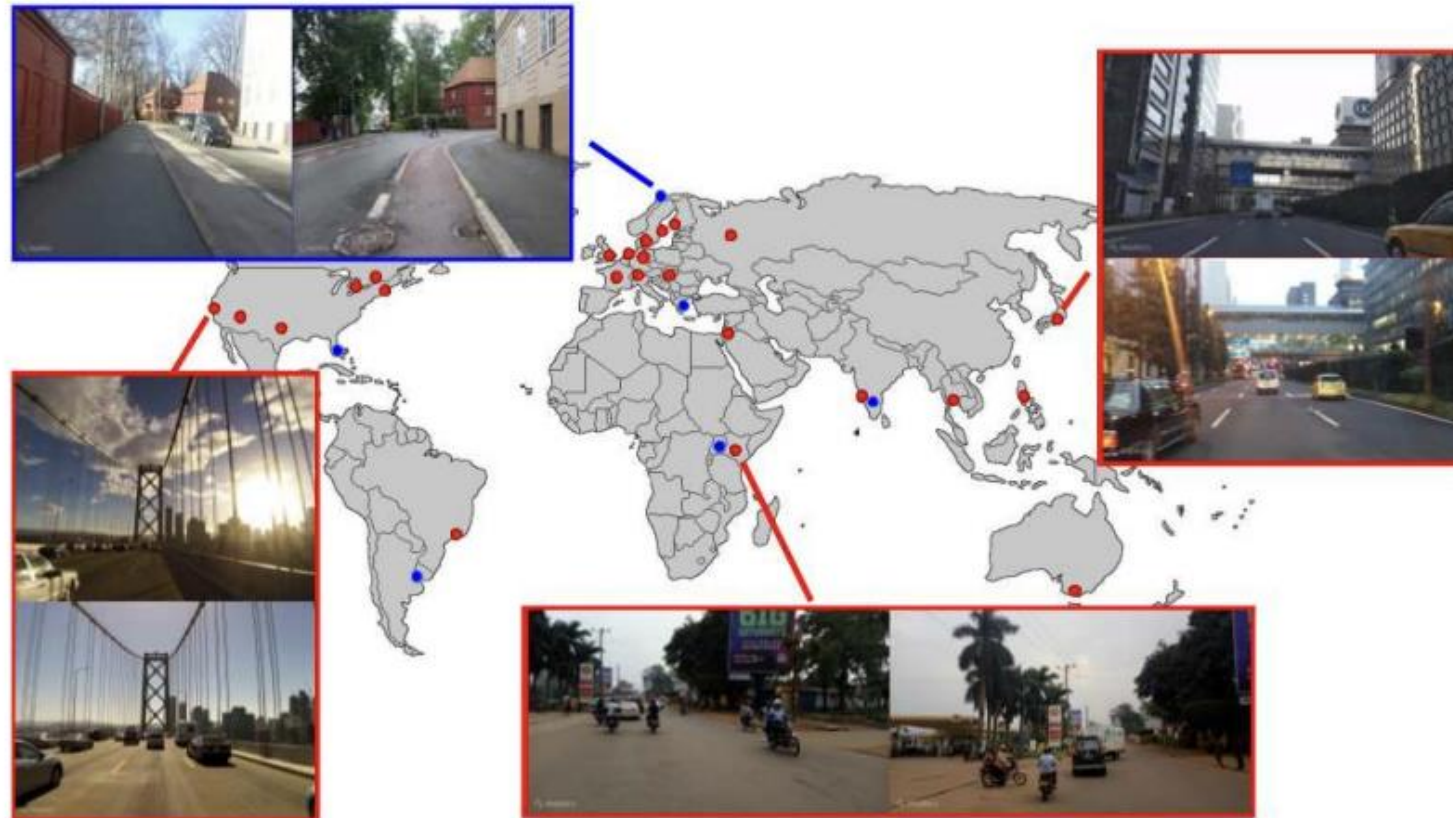
Ph.D dissertation defense talk

Jaeyoon Kim

(Advisor: Prof. Sung-Eui Yoon)

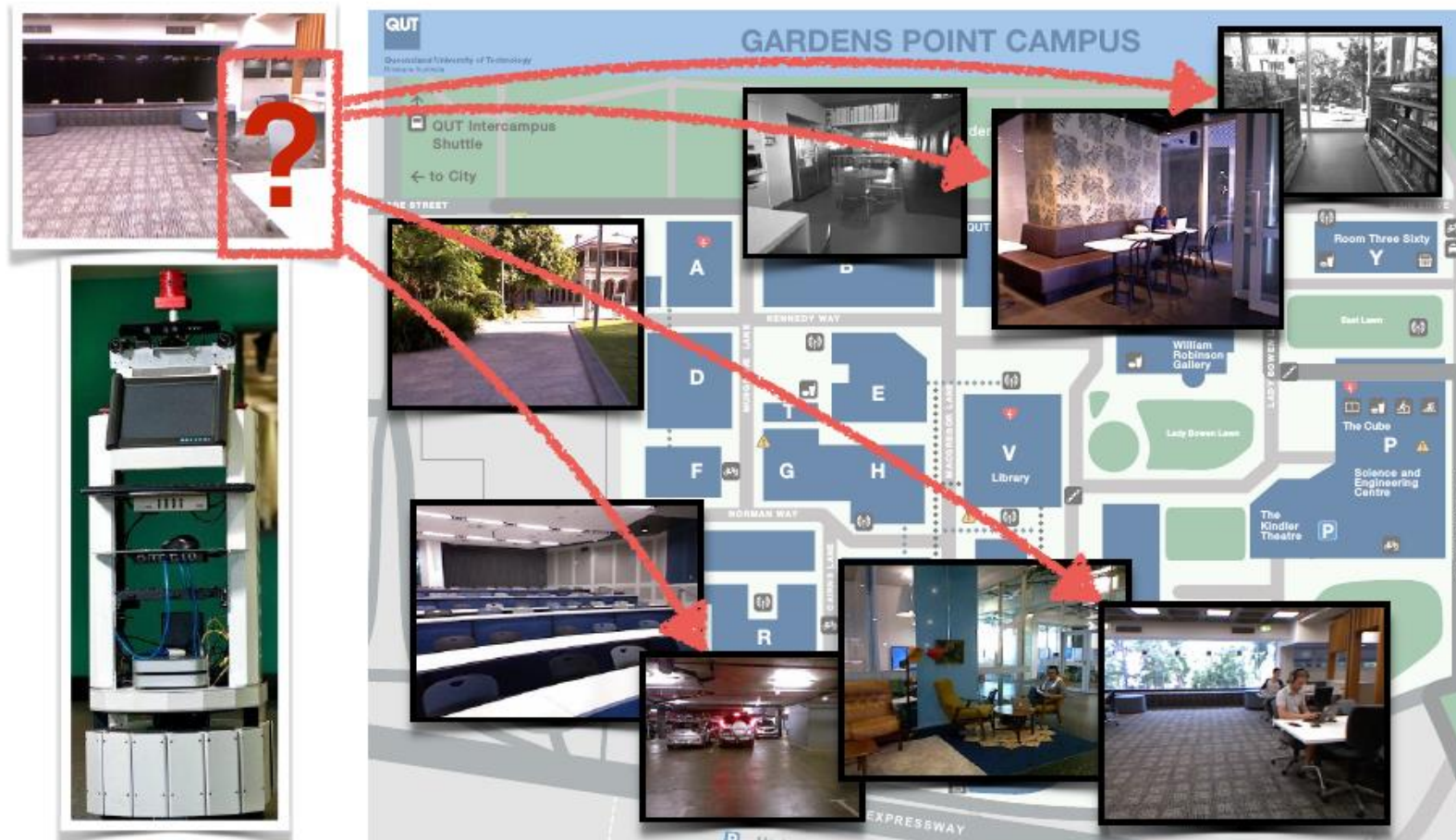
Introduction | Visual Place Recognition (VPR)

- The primary goal of VPR is to estimate the location of a query image based on visual data.



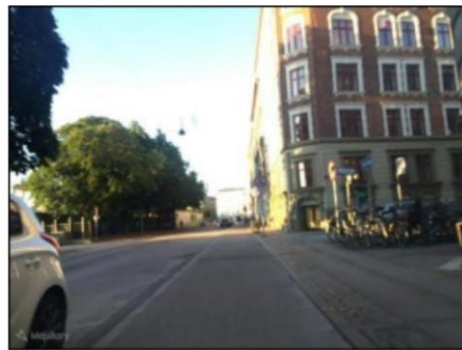
Introduction | Where is VPR used?

- VPR is **helpful for robots** to accurately **determine their location in the real world**, enabling reliable autonomous navigation.



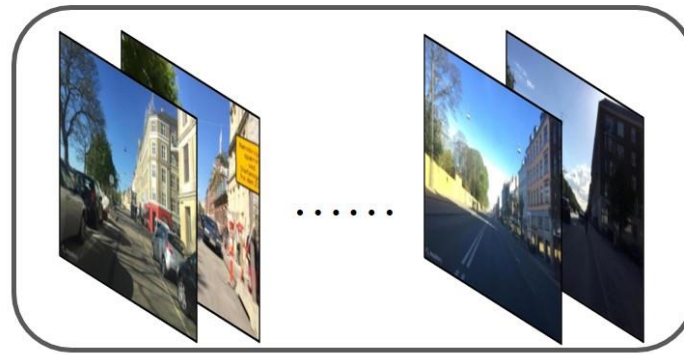
Introduction | Approaches to VPR - Retrieval

- The system searches a database of geo-tagged images by performing image retrieval.
 - It finds the **most visually similar images in the database** to the current query image.
 - The location of the most similar retrieved image then tells the system where it is.



Query Image from
Unknown Location

Search



Reference Images from
Known Locations

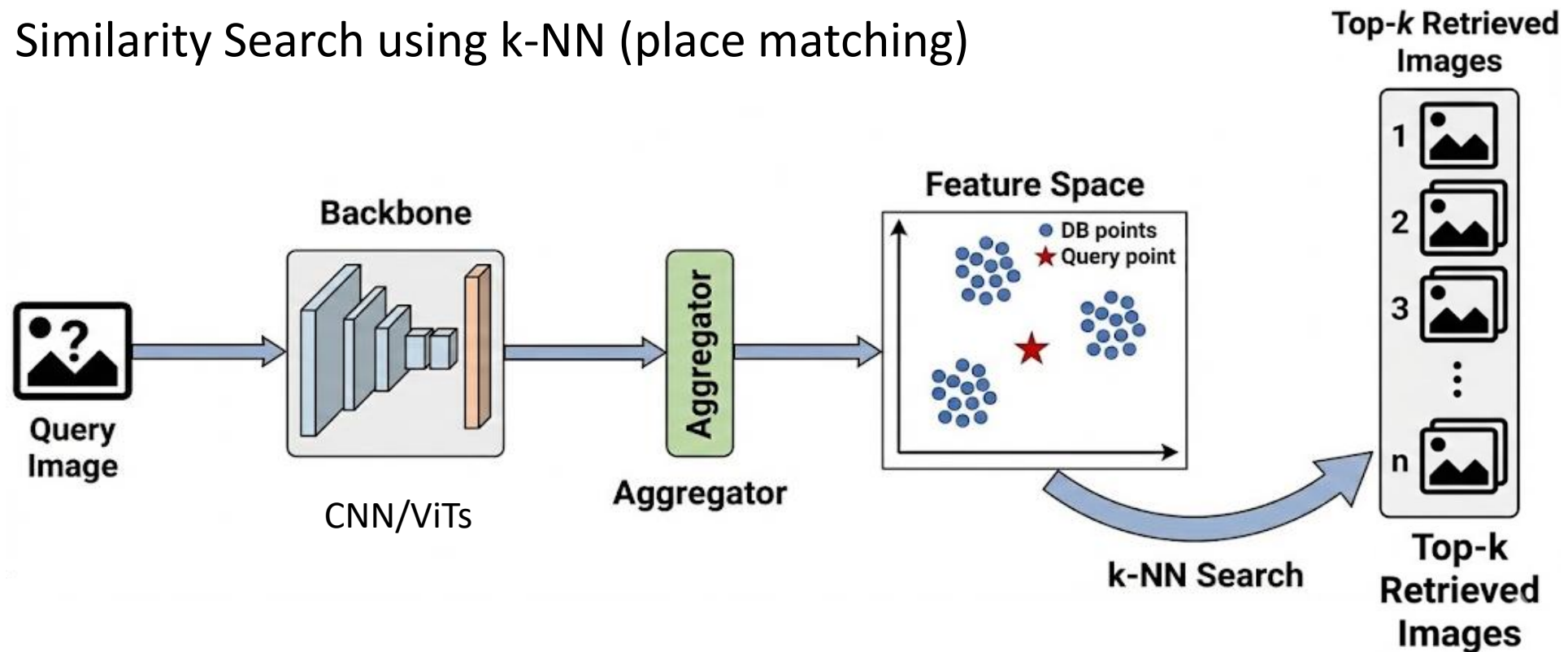


Retrieved Image

Introduction | Approaches to VPR - Retrieval

- **Common image retrieval pipeline:**

1. Feature extraction & aggregation
2. Metric space mapping
3. Similarity Search using k-NN (place matching)



Talk Roadmap

Our Goal: Towards Real-World Deployment of VPR.

Robustness

Research 1: Generalization to lifelong variation.

01

Static efficiency

Research 2: Asymmetric VPR for lightweight query

02

Dynamic efficiency

Research 3: Anytime VPR for resource-dynamics

03

Research 1:

**Class-Relational Approach for Visual Place
Recognition under Extreme Appearance Changes**

Motivation | Conventional Challenges

- **The real world is dynamic, and VPR systems face many hurdles:**
 - Illumination Changes: **Day vs. Night**, sunny vs. cloudy.
 - Viewpoint Changes: Looking at a building from **different angles or distances**.
 - Seasonal Variations: **Summer vs. Winter**, with trees in full bloom or covered in snow. These variations can drastically alter the appearance of a place, making recognition difficult.



(a)
Seasonal
Variation



(b)
Viewpoint
Variation



(c)
Dynamic
Objects



(d)
Illumination
Variation

Motivation | Proposed challenge

- Beyond the usual changes like weather or time of day, VPR faces an even **more extreme challenge: Lifelong Variation (LV)**.
 - This refers to permanent and irreversible physical **changes to a place over extended periods**.
 - Buildings can undergo significant modifications, like **renovations or architectural changes**.
 - Our research addresses how to make VPR robust to these long-term transformations.



Motivation | Why not collect more LV data?

- **The "Obvious" Solution: More Data.**

- A naive approach would be to simply collect a massive dataset that captures numerous examples of lifelong variation.
- **Longer temporal gaps** can give **richer LV signals**.
- **Collecting such data at a large scale is incredibly difficult.**



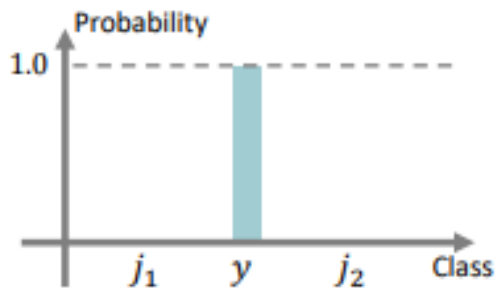
- Therefore, we aim to solve this problem by **improving the model's generalization capability**.

Our approach | Class-relational label smoothing (CRLS)

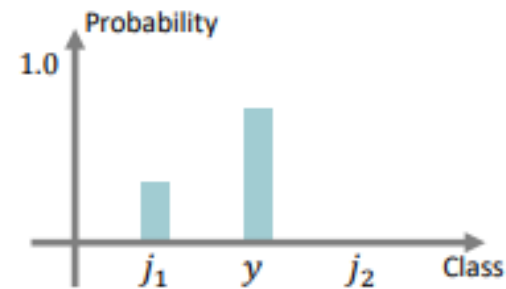
- **Different-but-similar places** act as **bridge samples (pseudo-positives)** for the target place.
- We **reallocate a small portion of the label mass** from the target to visually **similar classes** → encourages a **more generalizable** feature space.



(a) Three different classes located in different places



Hard label



CRLS (Ours)

Our approach | How do we get the relations? (self-bootstrapping)

- We **bootstrap** relations from the **current classifier weights**.
- Use the **classifier head weights** to compute **inter-class similarity** (cosine).
 - Inter-class similarity is defined as:

$$A_{y,j} = \underbrace{\sigma(W_y)}_{\text{L2-normalized weight of GT class } y} \overset{\text{class-weight cosine (relation score)}}{\cdot} \underbrace{\sigma(W_j)}_{\text{L2-normalized weight of class } j}$$

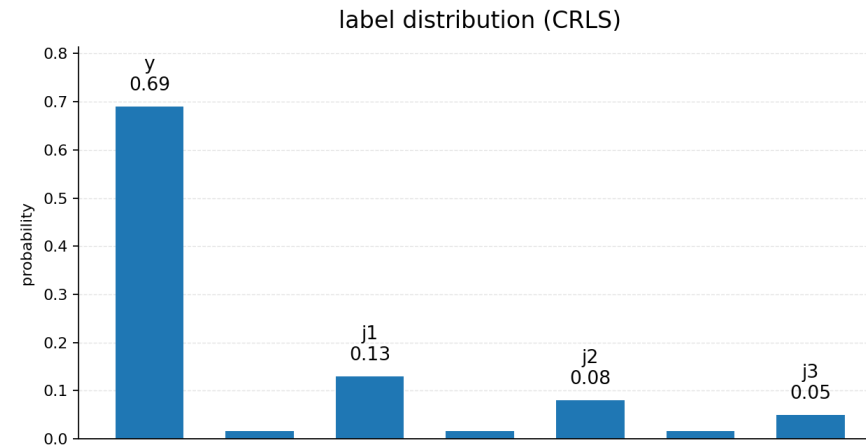
$A_{y,j} \uparrow$: Visually more similar to GT class y

$A_{y,j} \downarrow$: Visually less similar to GT class y

Our approach | Class-relational label smoothing (CRLS)

- **Make the label distribution aware of inter-class relations.**
 - **Our CRLS label distribution:**

$$f_{\text{CRLS}}(j) = \begin{cases} (1 - \alpha), & j = y, \\ \alpha \hat{A}_{y,j}, & j \neq y, \end{cases}$$

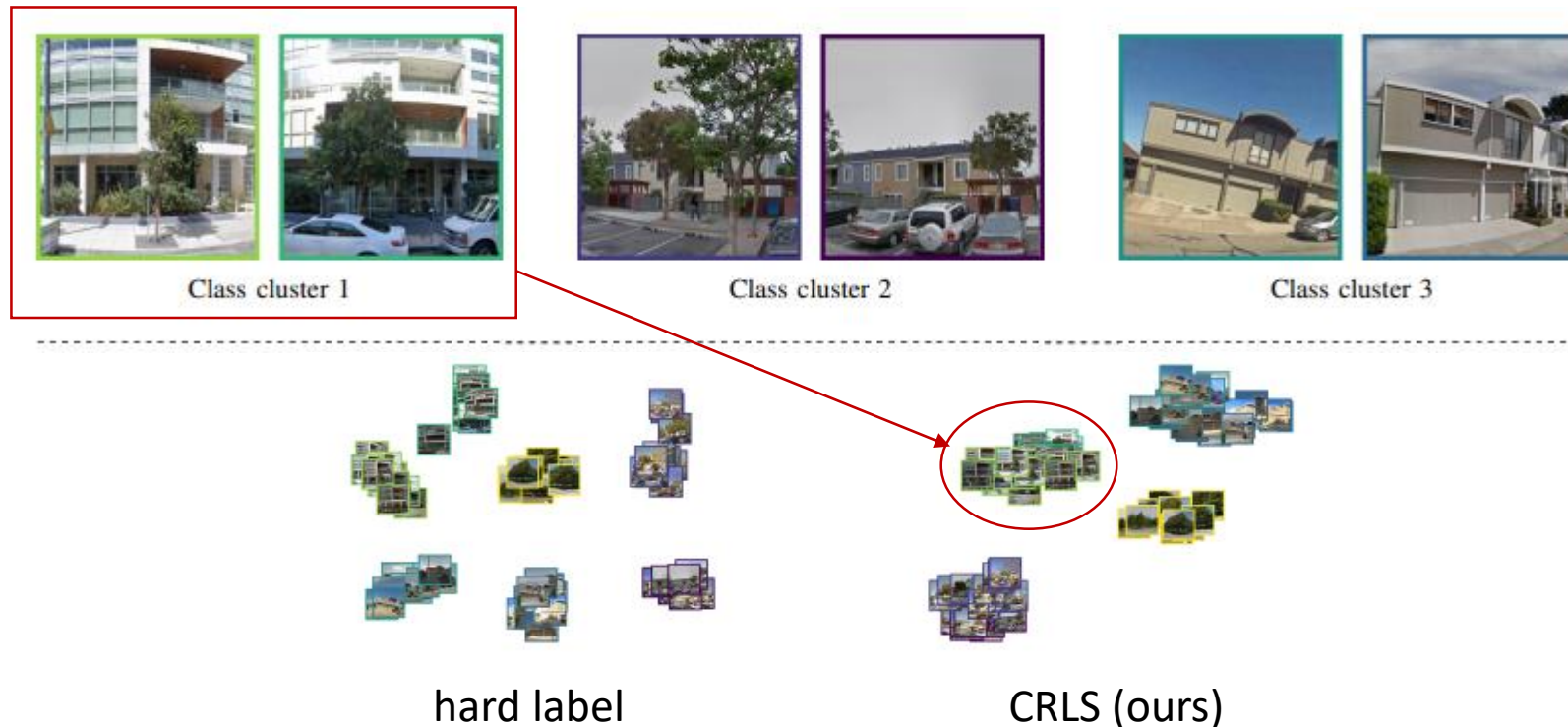


- **Relation distribution can be computed by a normalization of softmax:**

$$\hat{A}_{y,j} = \frac{\exp(A_{y,j}/\tau)}{\sum_{k \neq y} \exp(A_{y,k}/\tau)}, \quad \forall j \in \{1, \dots, K\}.$$

Analysis of the embedding space (t-SNE)

- **Hard labels:** even visually similar classes form **isolated islands** with gaps → vulnerable to long-term changes.
- **CRLS (ours):** pulls **look-alike classes** closer, yielding a **smoother, more continuous manifold**.



Class-relation analysis (via classifier weights)

- Nearest neighbors from the **class-weight cosine** are **visually consistent** with the anchor classes for our model.
- Baselines show **leakage to semantically different scenes**; CRLS ranks **look-alikes** higher and suppresses unrelated classes.



(a) An anchor class of street scene

(b) An anchor class of forest scene

Lifelong test datasets

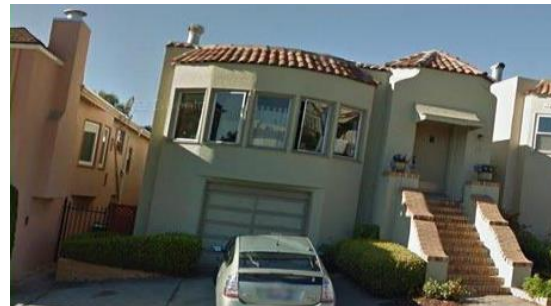
- We select **lifelong VPR benchmarks** containing temporal span **over seven-year**.
- **Same places, different years** - stress-testing robustness to building modification.

SF-XL test v1

2021



2013



AmsterTime

2017



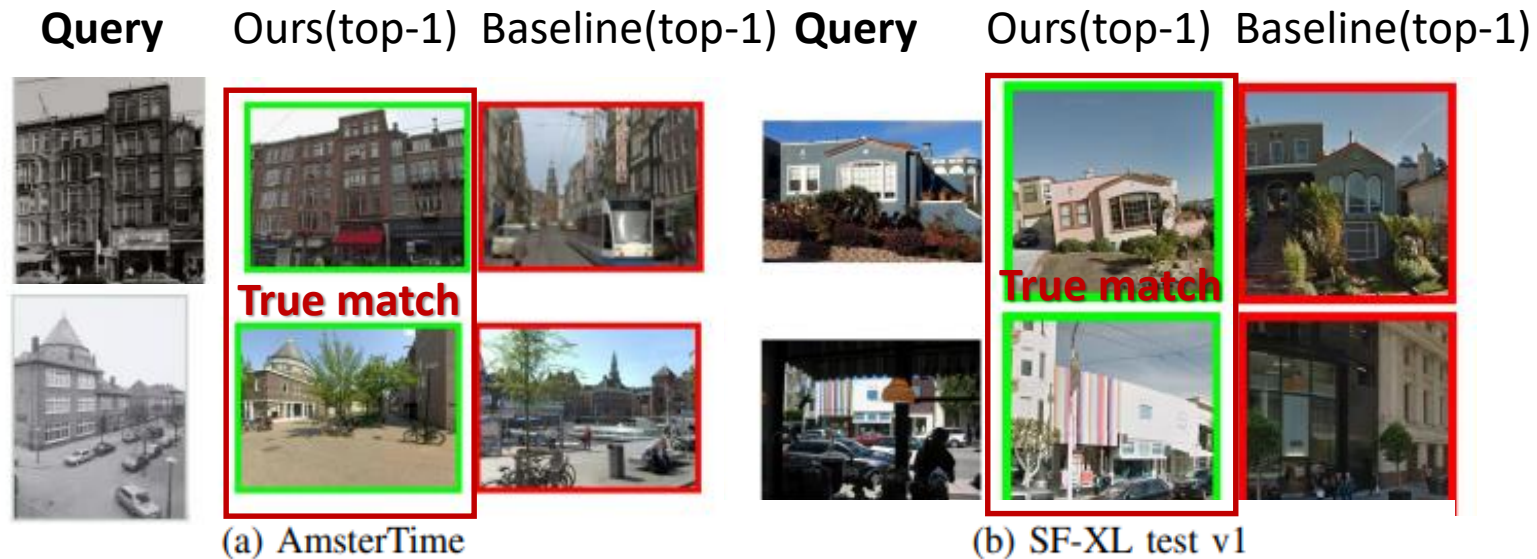
c. 1920s

100 years gap!



Qualitative results

- Across **AmsterTime** and **SF-XL v1**, our model **ranks the true match higher** under remodeling, seasonal/lighting changes, and occlusions.
- **Different-but-similar places can act as bridge samples** for lifelong variation.



Quantitative results

- Comparison with state-of-the-arts.
- Recall@1 as evaluation metric.

Method	Backbone	Dim.	SF-XL test v1	Amster.
GCL	ResNet-50	2048	11.4	14.6
R^2 Former	ResNet-50	256	19.0	16.7
MixVPR	ResNet-50	4096	72.5	40.8
CosPlace	ResNet-50	2048	76.4	47.7
EigenPlaces	ResNet-50	2048	84.1	48.9
Ours	ResNet-50	2048	86.0	51.1

Talk Roadmap

Our Goal: Towards Real-World Deployment of VPR.

Robustness

Research 1: Generalization to lifelong variation.

01

Static efficiency

Research 2: Asymmetric VPR for lightweight query

02

Dynamic efficiency

Research 3: Anytime VPR for resource-dynamics

03

AAAI (Accepted)

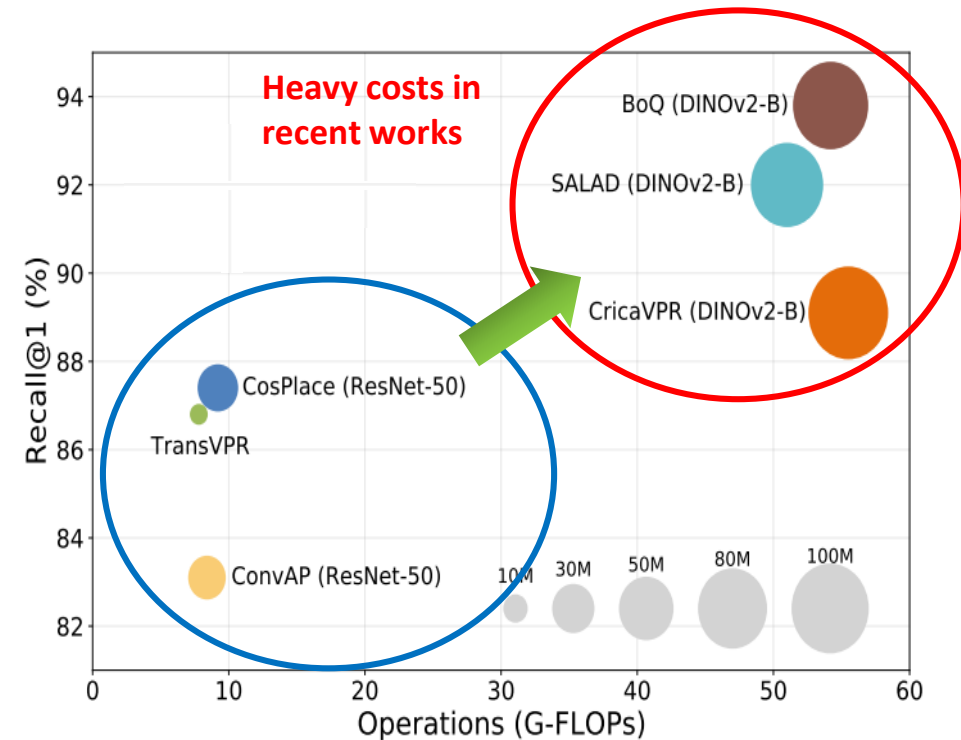
Research 2:

**Towards Test-time Efficient Visual Place Recognition
via Asymmetric Query Processing**

The Real-World Deployment Challenge (1)

(High Accuracy vs. Heavy Computation)

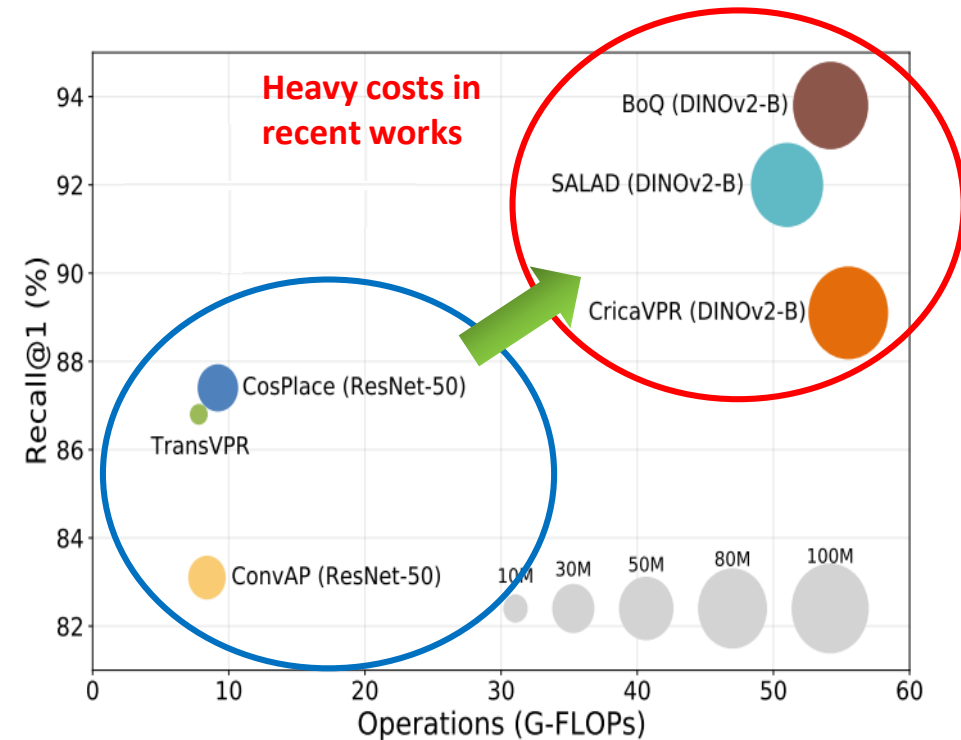
- Trend in Recent VPR Literature:
 - **Recent SOTA** methods (e.g., BoQ, SALAD) achieve **remarkable performance** by leveraging large-scale Foundation Models (e.g., DINOv2).
 - However, this comes at the cost of **massive computational overhead** (High G-FLOPs & Parameters).



The Real-World Deployment Challenge (2)

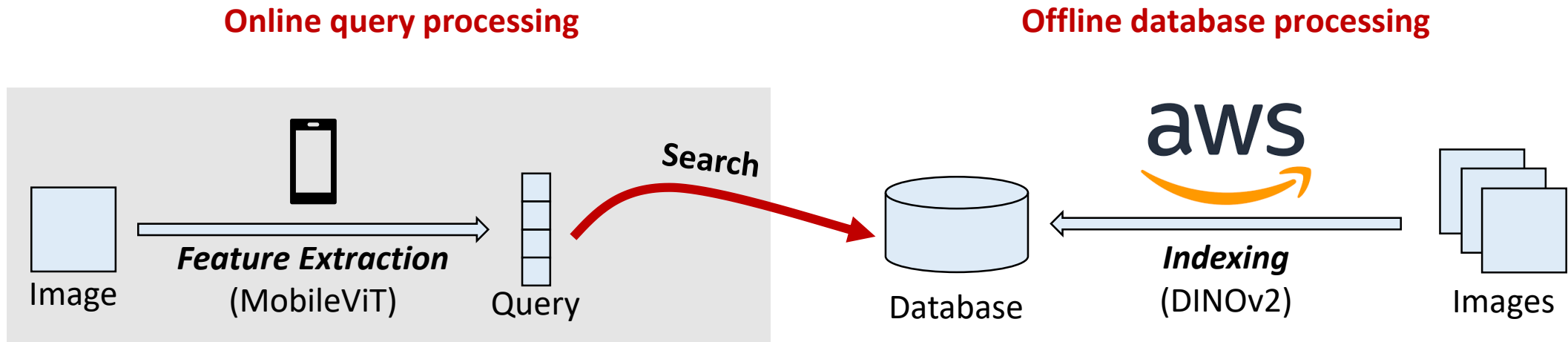
(High Accuracy vs. Heavy Computation)

- Real-World Deployment Gap:
 - Deploying such heavy models on **resource-constrained platforms** (e.g., mobile robots, drones) is often **impractical**.
 - Strict constraints on latency and hardware resources **hinder real-time application**.



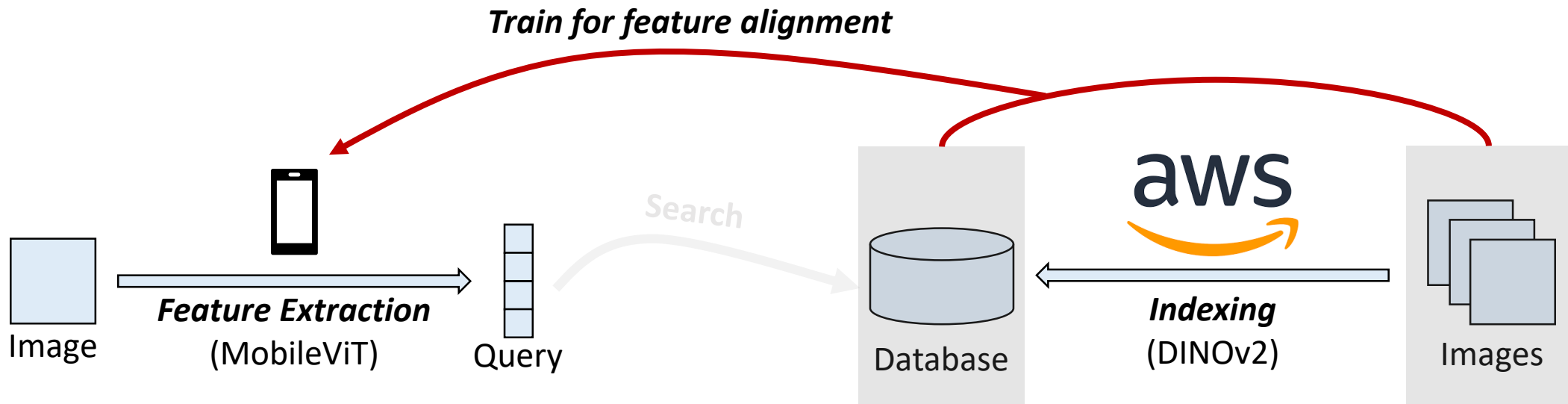
Background (1) – Asymmetric retrieval

- Use separate models with different capacities to process query and gallery images.
 - **High-capacity gallery model** for offline gallery (i.e., database) processing
 - **Lightweight query model** for online query processing
- Can use rich database features under resource constraints.



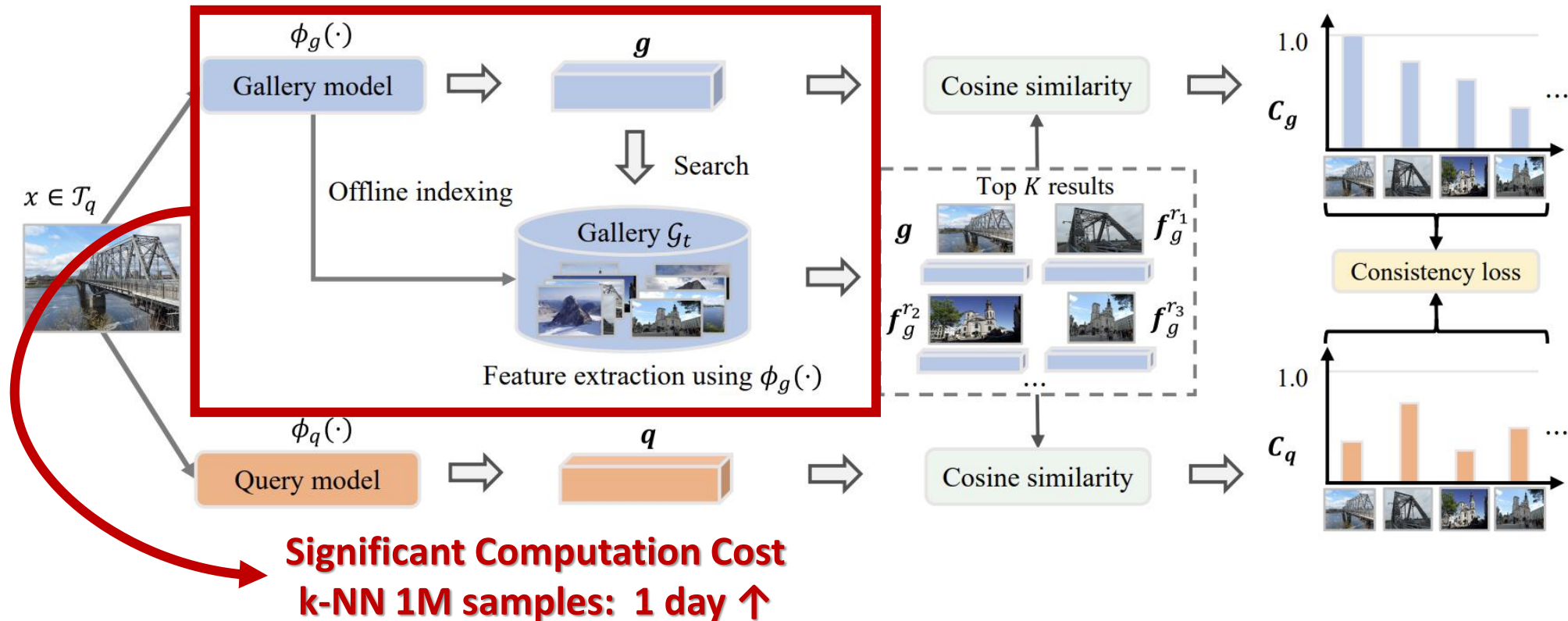
Background (2) – Compatible representation

- Key challenge: Train a query network that produces **compatible representations** with the gallery network.
 - **Use only database images and their embeddings** for compatible training.
 - Training data = Set of (image, embedding) pairs
 - Without labels



Background (3) – Use of k-NN for compatible training

- Usually leverage **k-nearest neighbor (k-NN) information** for compatible training.
 - **Effectively capture the manifold** of representation space (contextual information).



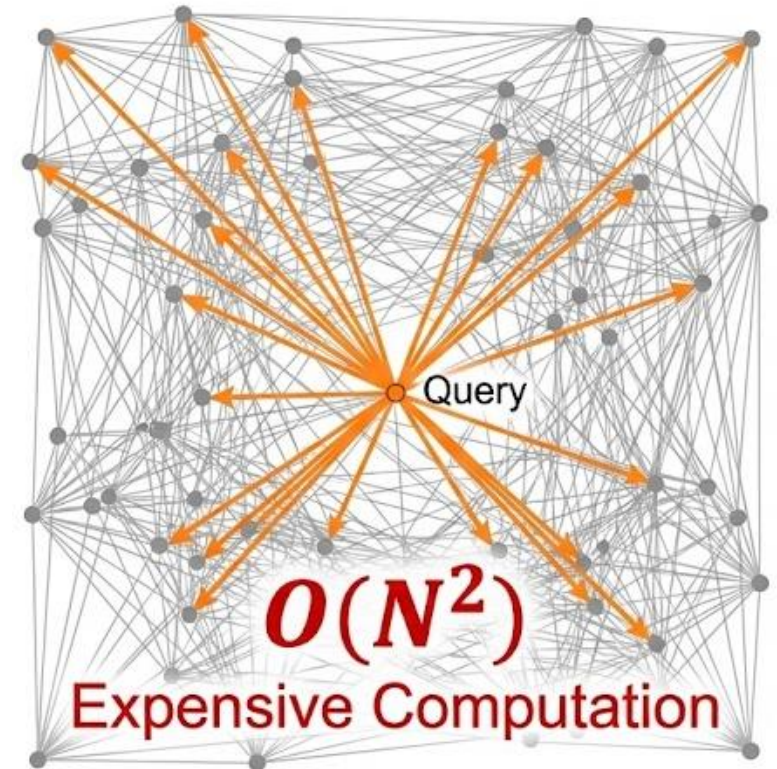
Background (4) – Limitation of k-NN for VPR

- k-NN captures only instance-level relationships:
 - Image A is similar to images B, C, D...
 - Local neighborhood information.

But misses the bigger picture:

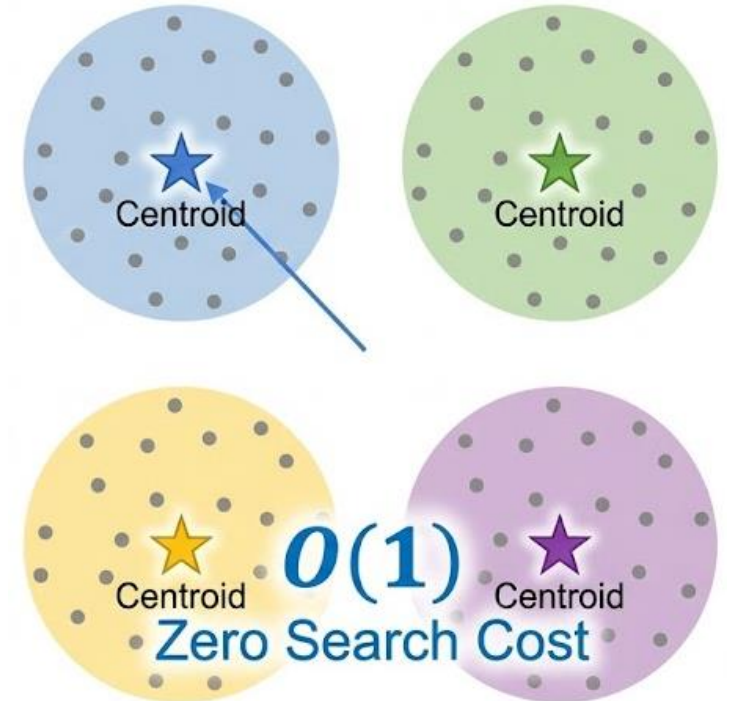
- ❌ Cannot capture place-level distribution
- ❌ Fails to model intra-place variations
(viewpoint, illumination, weather changes)

✅ We need **place-level manifold** understanding!



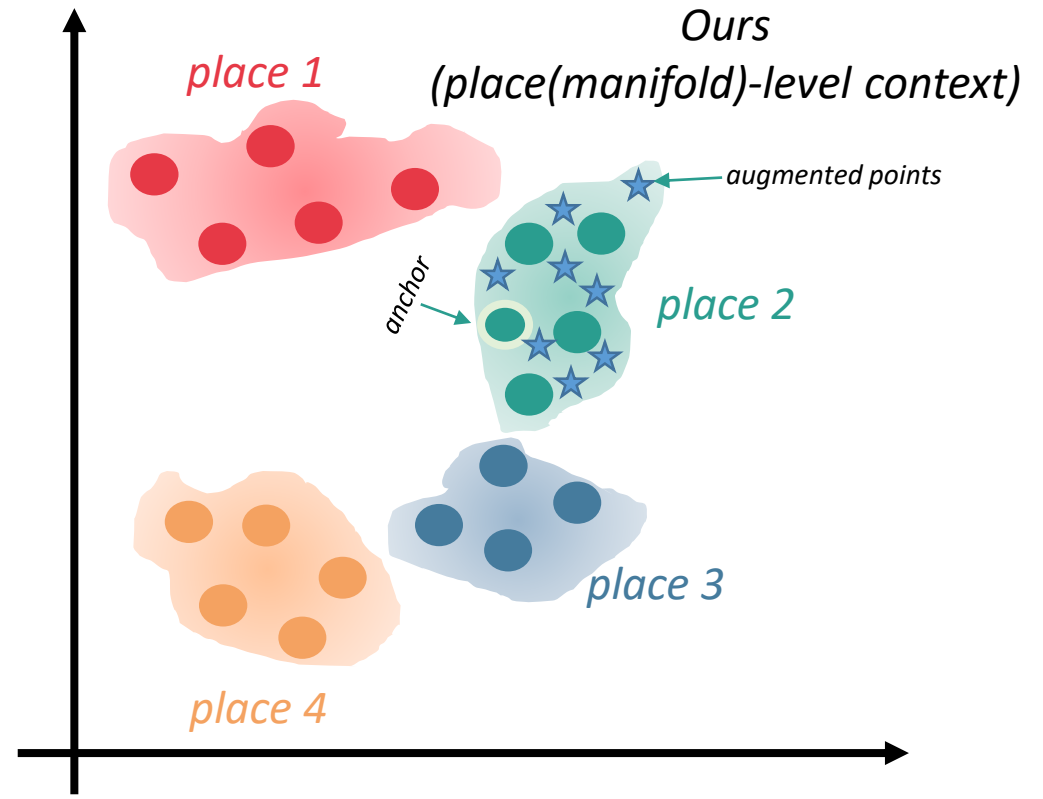
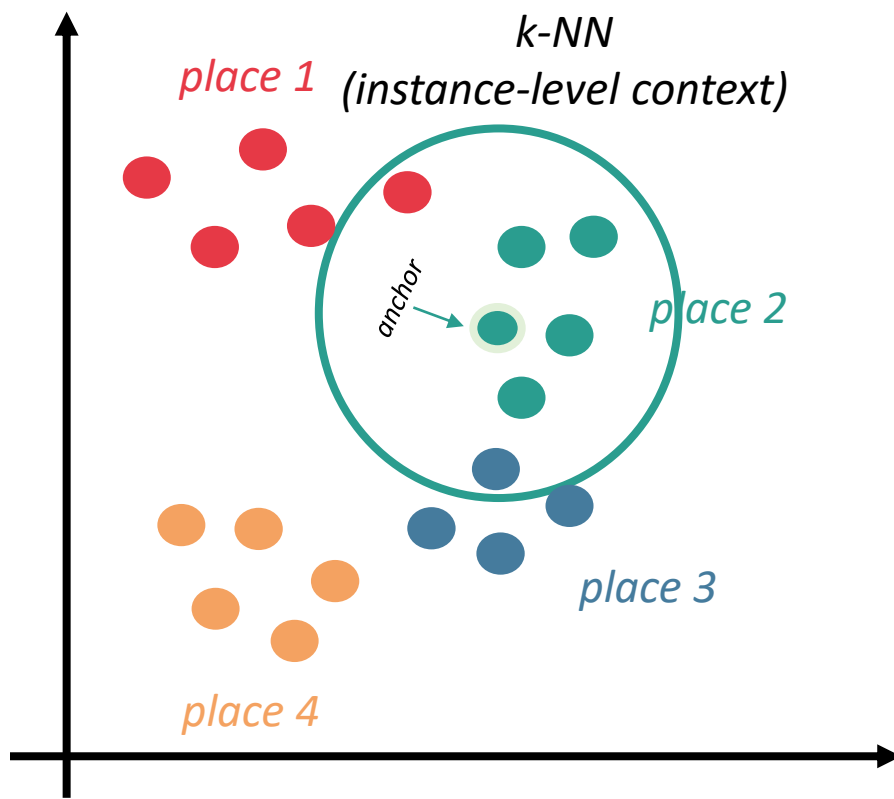
Our approach - Place-level manifold via GPS

- k-NN approach:
 - **Instance-level:** Find similar individual images
 - Expensive: $O(N^2)$ comparisons
 - Misses structure: No place-level understanding
- Our approach:
 - **Place-level:** Group images by GPS coordinates
 - **Efficient:** $O(1)$ with geolocation metadata
 - **Better structure:** Captures place distribution directly



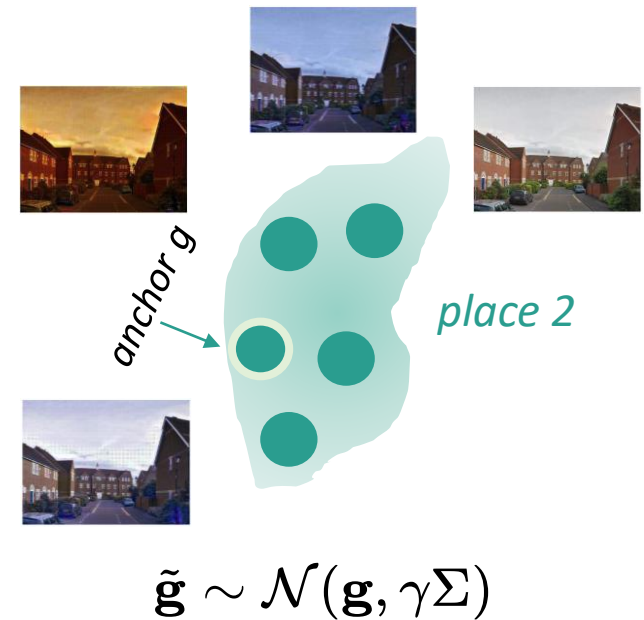
Our approach – Conceptual explanation

- Limitation of k-NN: Relies on unstable instance-level connectivity, often ***capturing unrelated manifolds***.
- Our Approach: Models ***Place-level Manifolds*** using geolocation priors. Embedding augmentation effectively covers the feature space to cover diverse variations.



Our approach – Place-level distribution modeling

- Place-level statistics:
 - Mean (μ): Centroid of gallery features at location
 - Covariance (Σ): Feature distribution spread within the same place
- Distribution modeling:
 - Augmented feature : $\tilde{\mathbf{g}} \sim \mathcal{N}(\mathbf{g}, \gamma\Sigma)$ γ : scaling parameter
 - Σ captures intra-place variations (viewpoint, lighting, weather)
 - These variations are non-discriminative for place recognition
 - Query model should learn to be robust to these changes

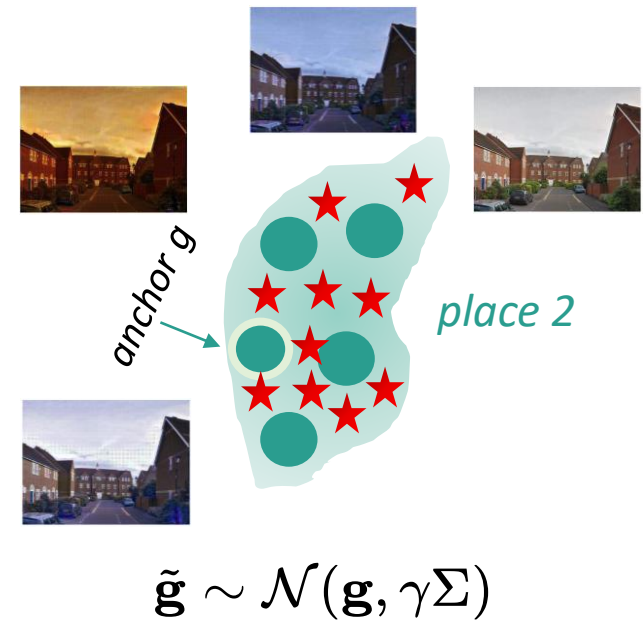


Our approach – Explicit embedding augmentation

- Sample generation from distribution:
 - Generate K augmented features from $\mathcal{N}(\mathbf{g}, \gamma\Sigma) : \{\tilde{\mathbf{g}}^1, \tilde{\mathbf{g}}^2, \dots, \tilde{\mathbf{g}}^K\}$
 - Each sample represents a plausible feature variation at that place

- Training with augmented features:
 - Compatibility loss computed for each augmented feature
 - Repeat this process K times per training iteration

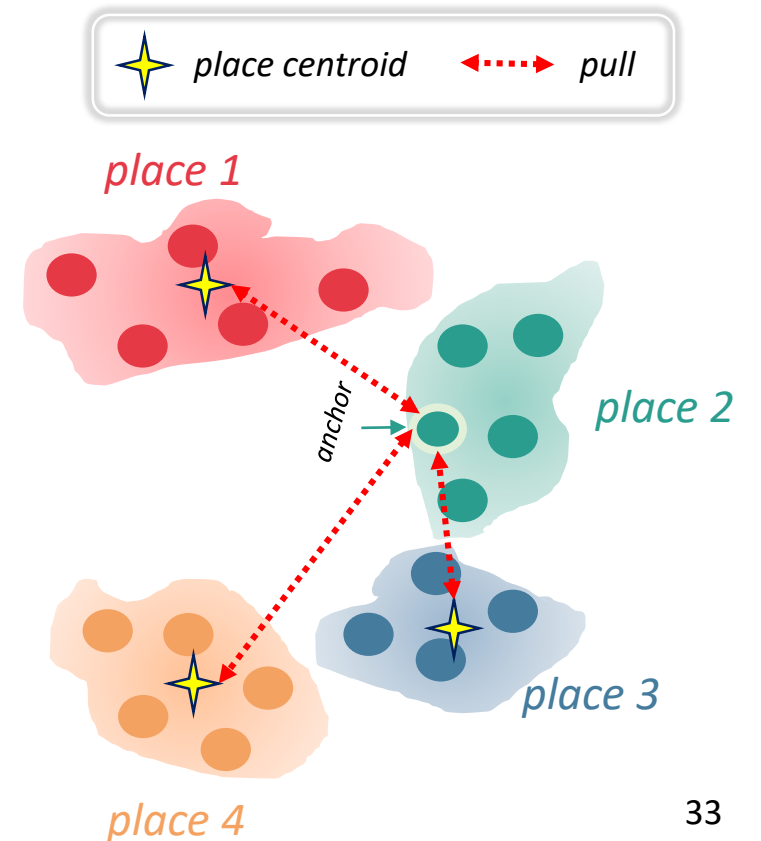
$$\mathcal{L} = \frac{1}{K} \sum_{k=1}^K \mathcal{L}_{\text{compatibility}}(\mathbf{q}, \tilde{\mathbf{g}}^k)$$



Our approach – Centroids as negative samples

- Contrastive learning with place-level negatives:
 - To improve discriminativeness, we use negative samples during training
 - Query learns to: align with its place, separate from other places

- Use place centroids as negatives:
 - Memory bank $M = \{c_1, c_2, \dots, c_m\}$: one centroid per place
 - For training image from place 2:
 - Positive: Align with place 2's gallery features
 - Negatives: Centroids from place 1, 3, 4, ... (different locations)

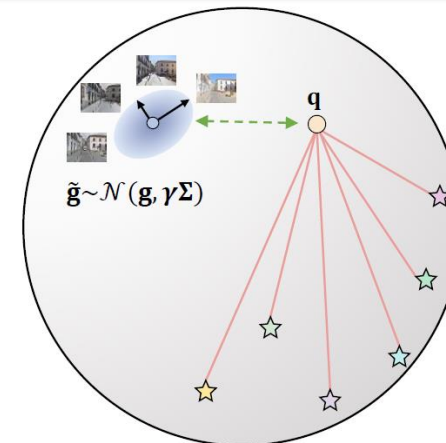
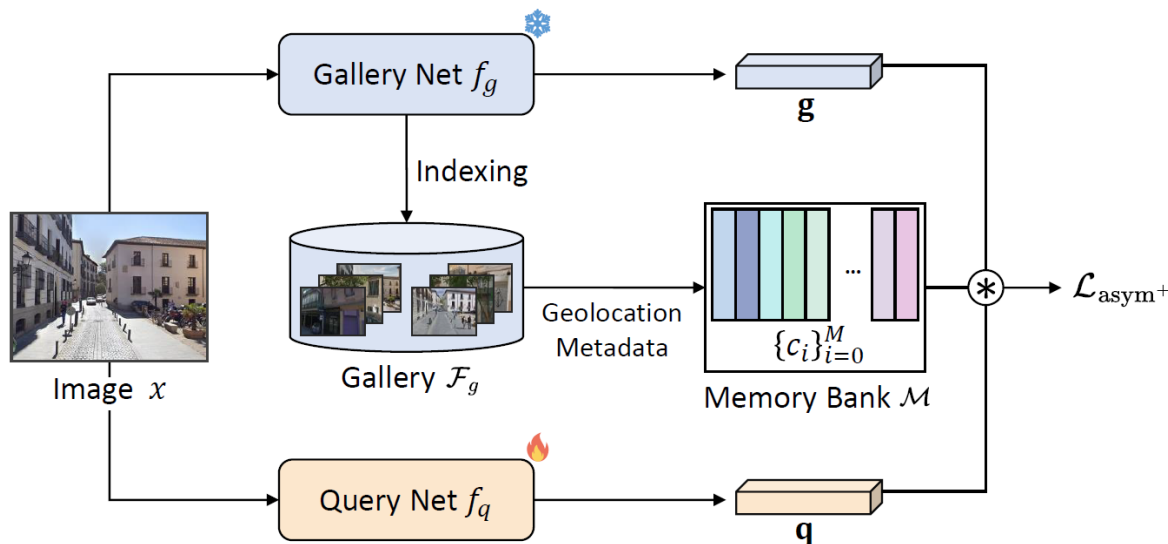


Our approach – Asymmetric Contrastive Learning

- Asymmetric contrastive learning:
 - q : Query embedding from lightweight model.
 - $\mathcal{N}(x)$: negative centroids not to belong to the location of input image x .

$$\mathcal{L}_{\text{asym}^+}^{\text{exp}} = -\frac{1}{K} \sum_{k=1}^K \log \frac{e^{q \cdot \tilde{g}^k / \tau}}{e^{q \cdot \tilde{g}^k / \tau} + \sum_{j \in \mathcal{N}(x)} e^{q \cdot c_j / \tau}}$$

Positive: augmented feature
Negative: place centroid



Asymmetric Contrastive Learning

Our approach - From explicit to implicit augmentation

- **Training inefficiency of explicit augmentation** needs to generate K samples per iteration: $\{\tilde{\mathbf{g}}_1, \tilde{\mathbf{g}}_2, \dots, \tilde{\mathbf{g}}_k\}$ and computes loss K times.
- **Mathematical derivation for efficiency:**
 - Apply limit as $K \rightarrow \infty$ to explicit loss
 - Derive closed-form solution without iteration

$$\mathcal{L}_{\text{asym}^+}^{\text{exp}} = -\frac{1}{K} \sum_{k=1}^K \log \frac{e^{\mathbf{q} \cdot \tilde{\mathbf{g}}^k / \tau}}{e^{\mathbf{q} \cdot \tilde{\mathbf{g}}^k / \tau} + \sum_{j \in \mathcal{N}(x)} e^{\mathbf{q} \cdot \mathbf{c}_j / \tau}}$$



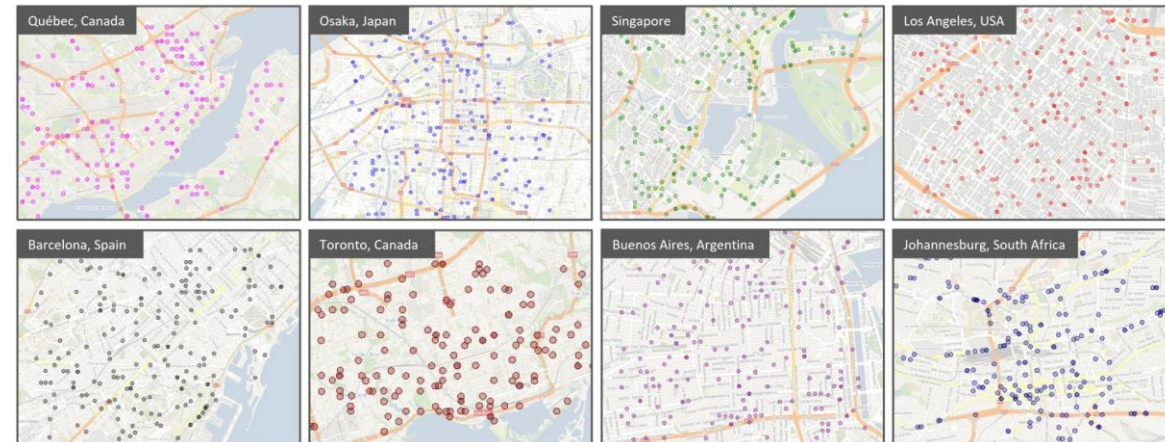
$$\mathcal{L}_{\text{asym}^+} = -\log \frac{e^{\mathbf{q} \cdot \mathbf{g} / \tau}}{e^{\mathbf{q} \cdot \mathbf{g} / \tau} + \sum_{j \in \mathcal{N}(x)} e^{\mathbf{q} \cdot \mathbf{c}_j / \tau + (\gamma / 2\tau^2) \mathbf{q}^T \Sigma \mathbf{q}}}$$

Covariance regularization replaces K iterations.

Experimental Setting

- Model
 - Gallery model: DINOv2-based SALAD (CVPR 24) & BoQ (CVPR 24)
 - Query model: **Lightweight transformer MobileViTv2** (TMLR 23) & **EfficientViT-B2** (ICCV 23)

- Training Dataset: GSV-Cities
 - 530k images covering 62k places from 40 cities



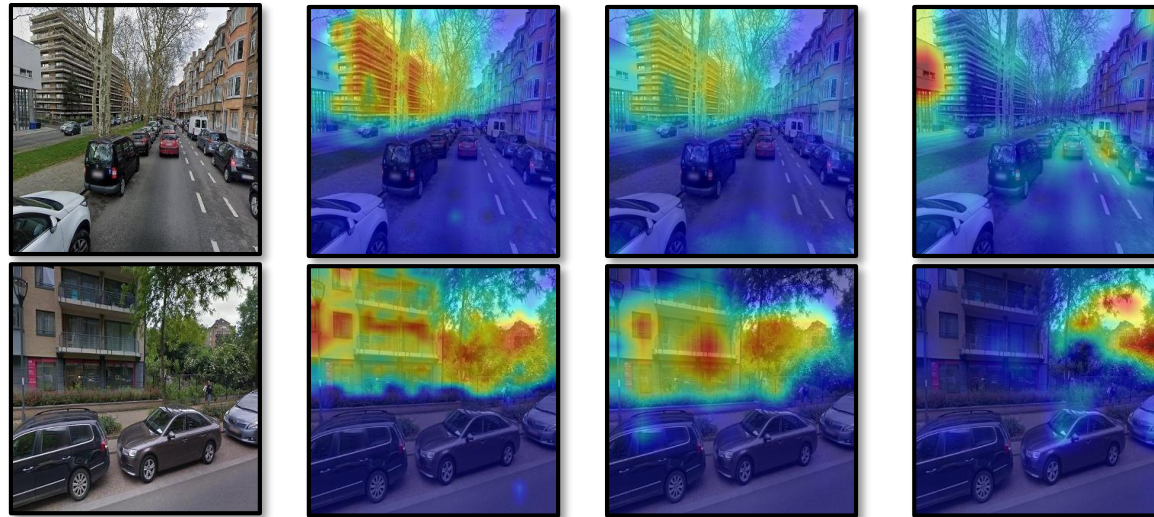
- Evaluation metric (retrieval accuracy): Recall@1 & Recall@5

Visual Evidence: Better Manifold Learning

Key message: Our place-level approach captures the manifold better than instance-level k-NN.

- Grad-CAM Visualization using Geographical Memory Bank as classifier.
- Our method show more similar activations with the gallery model.

(a) Original Image, (b) Gallery (BoQ), (c) Ours, (d) CSD (other method)



(a)

(b)

(c)

(d)

Much More Efficient Training than k-NN methods

Key message: GPS-based grouping is $5361\times$ faster than k-NN preprocessing, making large-scale training practical.

- Training Cost Comparison.
 - k-NN-based approach require significant computational costs.
 - Precomputation: Compute k-NN for all database images.
 - Training iteration & GPU memory: Load k-NN feature for each image of mini-batch.

Method	Precomputation (min)	Training Iteration (sec)	GPU Memory (GB)
CSD [51]	1392.76	1.34	23.9
Ours	0.26	0.19	17.5

Performance vs. k-NN Methods

Key message: Our place-level manifold learning outperforms ALL instance-level k-NN methods, especially on challenging datasets.

- Comparison with SOTAs when training with **BoQ (DINOv2)** as the gallery model.

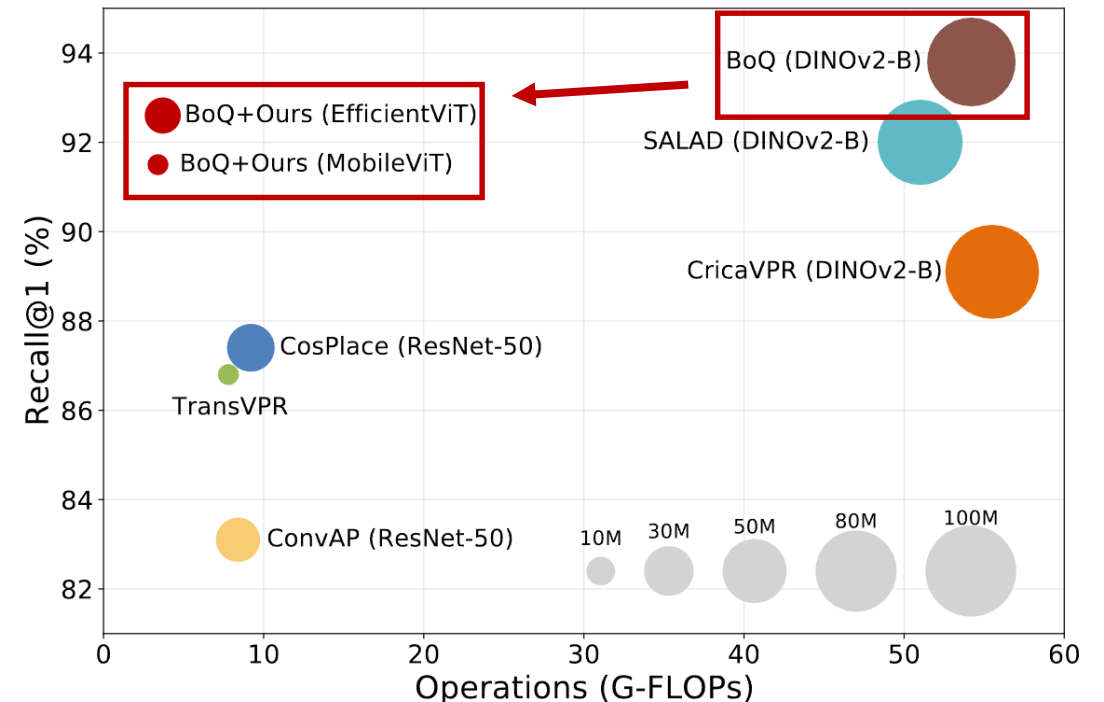
Method	Query Network	Pitts250k		MSLS Val		Tokyo24/7	
		R@1	R@5	R@1	R@5	R@1	R@5
<i>Training with BoQ of DINOv2 as gallery model</i>							
CSD (Wu et al. 2022)	k-NN methods	94.6	98.4	91.4	96.5	90.5	94.9
ROP (Wu et al. 2023)		91.6	97.1	90.5	96.0	84.1	91.4
MSP (Wu et al. 2023)	EfficientViT-B2	94.3	98.1	91.4	95.8	90.5	94.3
D3still (Xie et al. 2024)		95.0	98.4	91.9	95.9	91.7	95.9
Ours		95.4	98.5	92.3	95.9	92.7	96.5

Test-time efficiency Results

Key message: Achieve efficient but comparable performance with 4× faster inference and 15× less computation.

- Efficiency comparison of query backbones (FLOPs, Parameters, Latency)
- Accuracy-Efficiency trade-off on MSLS Val (Recall@1 vs. G-FLOPs, parameters)

	Backbone	FLOPs		Params		Latency	
		(G)	rel.	(M)	rel.	(ms)	rel.
SALAD	DINOv2-B	51.0	100%	88.0	100%	61.5	1.0×
	EfficientViT-B2	3.7	7.3%	15.8	18.0%	17.8	3.5×
	MobileViTv2	3.4	6.7%	5.4	6.1%	15.2	4.0×
BoQ	DINOv2-B	54.2	100%	95.2	100%	62.4	1.0×
	EfficientViT-B2	4.4	8.1%	22.3	23.4%	21.1	3.0×
	MobileViTv2	4.2	7.7%	12.1	12.7%	17.3	3.6×



Ablation for the proposed components

Key message: Both geographic memory bank and implicit augmentation are crucial for complete place-level manifold learning.

- Implicit embedding augmentation is more effective than explicit augmentations.
- Using geolocation metadata for memory bank construction is important.

Method	Pitts250k		MSLS Val		Tokyo24/7	
	R@1	R@5	R@1	R@5	R@1	R@5
Ours	95.4	98.5	92.3	95.9	92.7	96.5
<i>without</i> implicit embedding augmentation	94.3	98.1	91.2	95.7	90.2	95.6
<i>with</i> explicit embedding augmentation	94.7	98.4	91.5	96.2	91.4	95.2
<i>with</i> queue-based memory bank	94.2	97.9	91.4	96.1	90.2	95.9

Our approach 2 | Further efficiency analysis

- More comprehensive efficiency analysis – Retrieval time.
 - ANN (approximate nearest neighbor) can be utilized to accelerate conventional retrieval time.
 - We observe that **existing ANN** can further **help under asymmetric VPR tasks**.

Method	Search Type	Inference (ms)	Search (ms)	Total (ms)	R@1 (%)
Symmetric	<i>k</i> -NN (Faiss FlatL2)	62.4	6.6	69.0	93.8
	ANN (Faiss IVFPQ)	62.4	0.1	62.5	89.5
Asymmetric (Ours)	<i>k</i> -NN (Faiss FlatL2)	21.1	6.6	27.7	92.3
	ANN (Faiss IVFPQ)	21.1	0.1	21.2	87.6

Talk Roadmap

Our Goal: Towards Real-World Deployment of VPR.

Robustness

Research 1: Generalization to lifelong variation.

01

Static efficiency

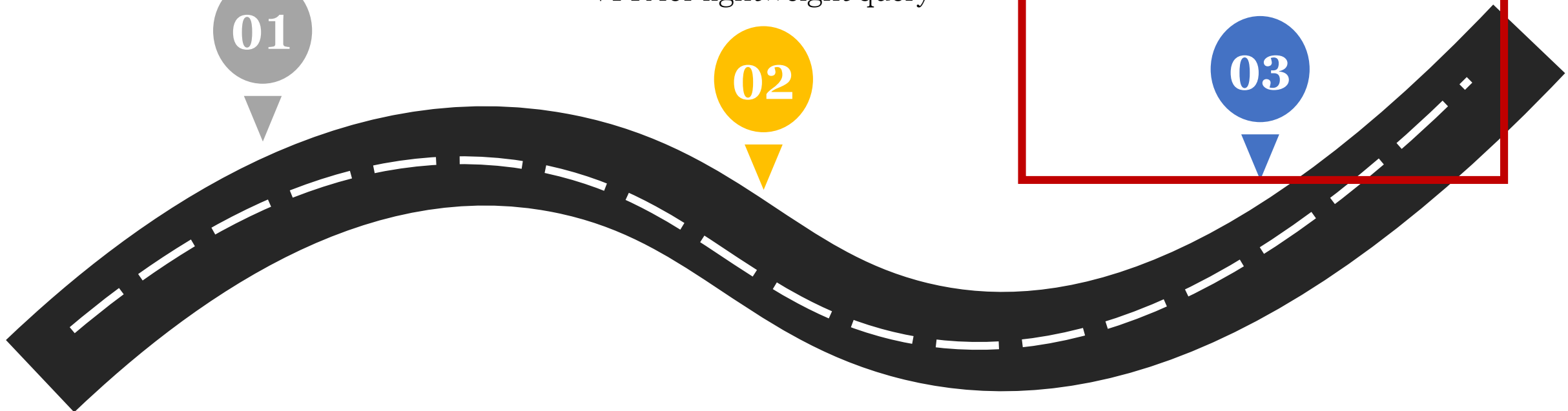
Research 2: Asymmetric VPR for lightweight query

02

Dynamic efficiency

Research 3: Anytime VPR for resource-dynamics

03

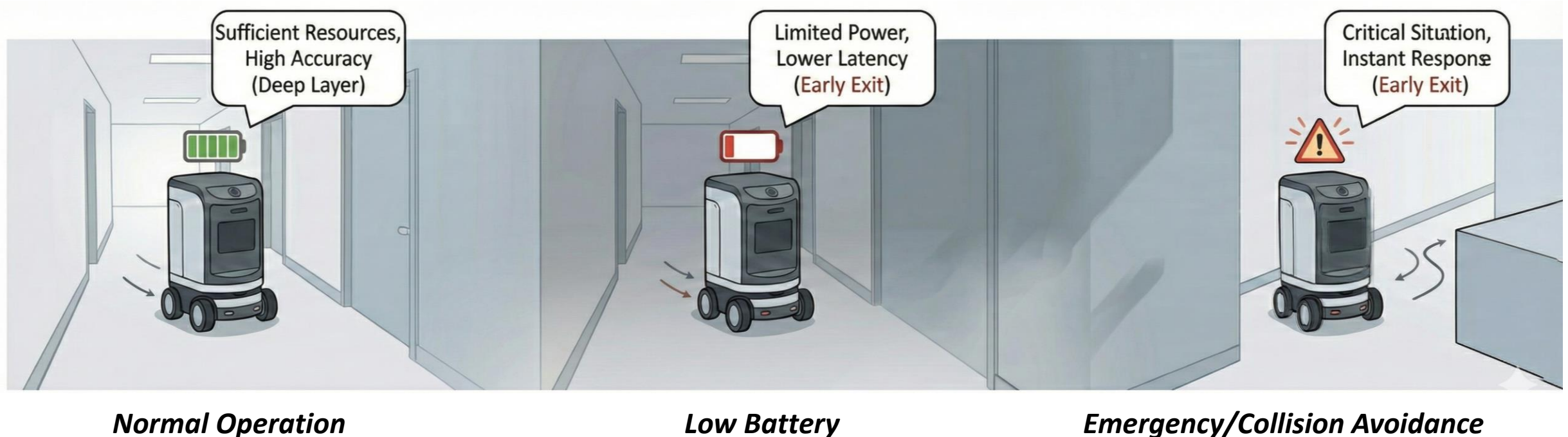


Research 3:

**Towards Anytime Visual Place Recognition via
Manifold-aware Codebook Composition**

The Problem: Static VPR systems (previous researches)

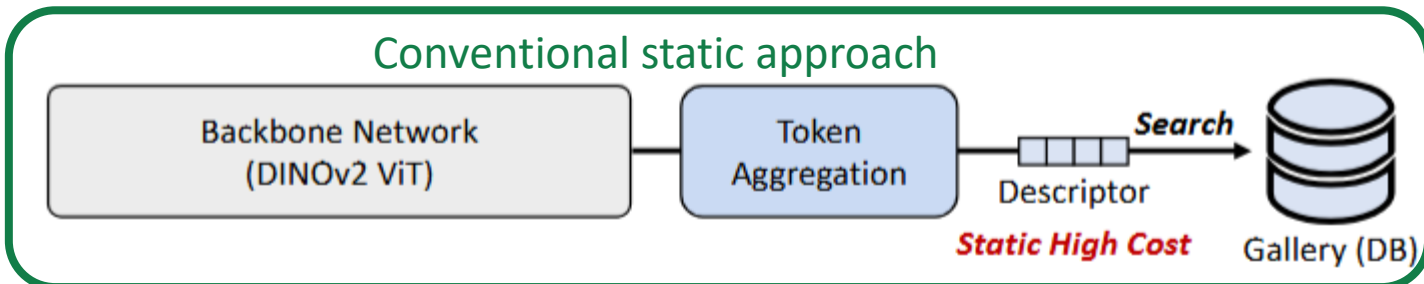
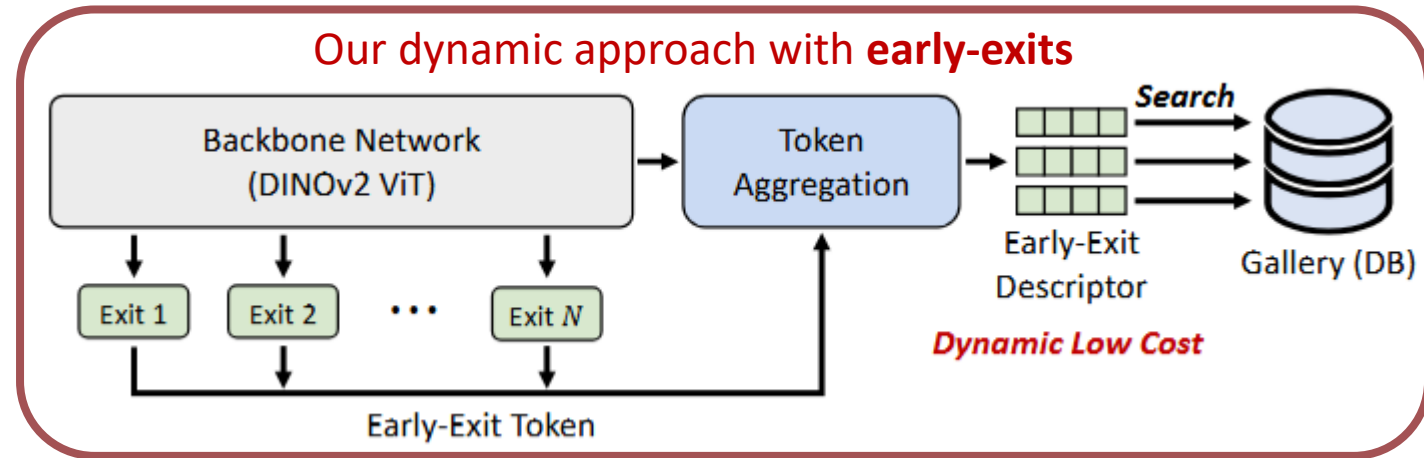
- Current VPR methods: **Fixed (static) computational cost**
- Challenge: Real-world robots face **dynamic resource** constraints
 - Battery limitations
 - Varying computational budgets
 - Need for adaptive performance resource adaptation via Early-exits.



Our Solution | Anytime VPR framework with early-exits

- Propose first **early-exit framework** for VPR Enabling dynamic inference.
- Perform retrieval at intermediate layers without executing the full network

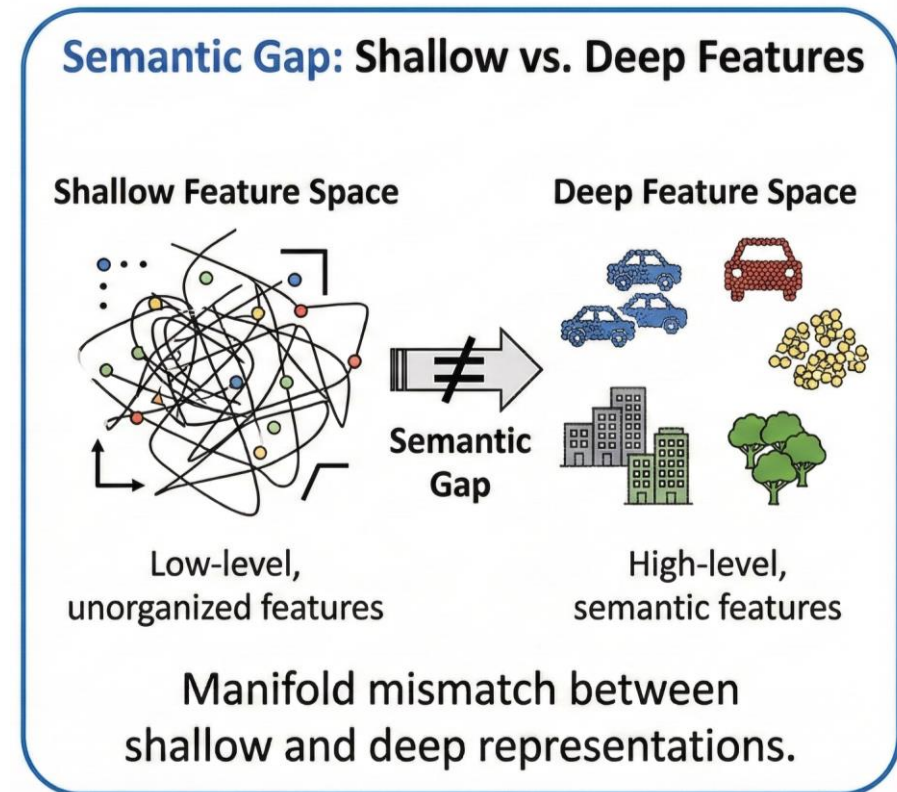
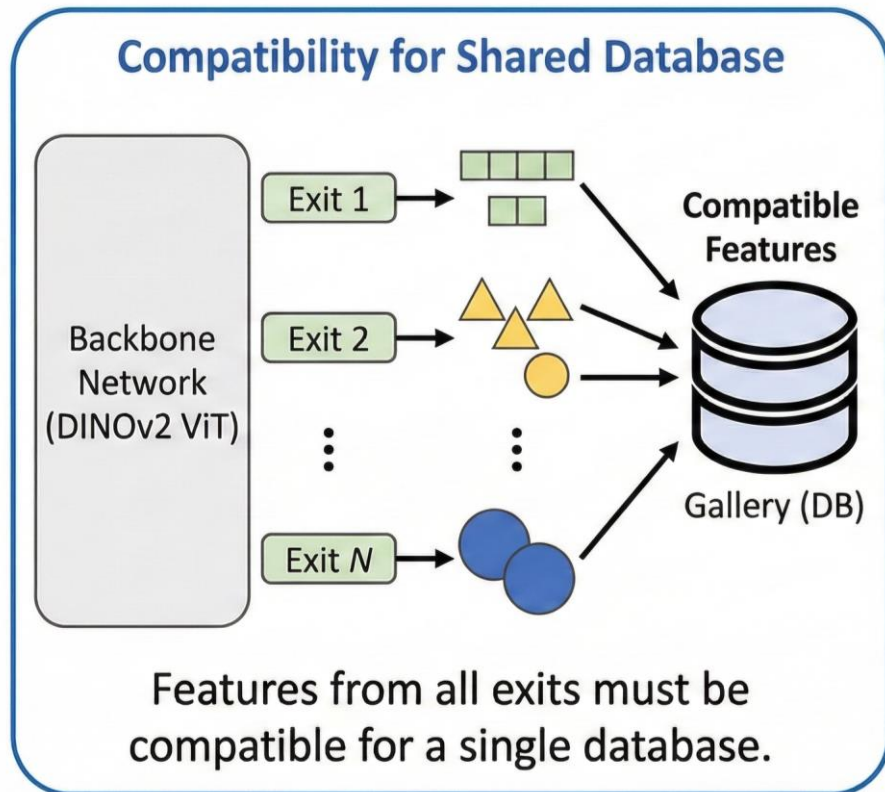
- **Flexible Computation:** Adapts to available resources.
- **Latency-Accuracy Trade-off:** Optimized for different constraints.



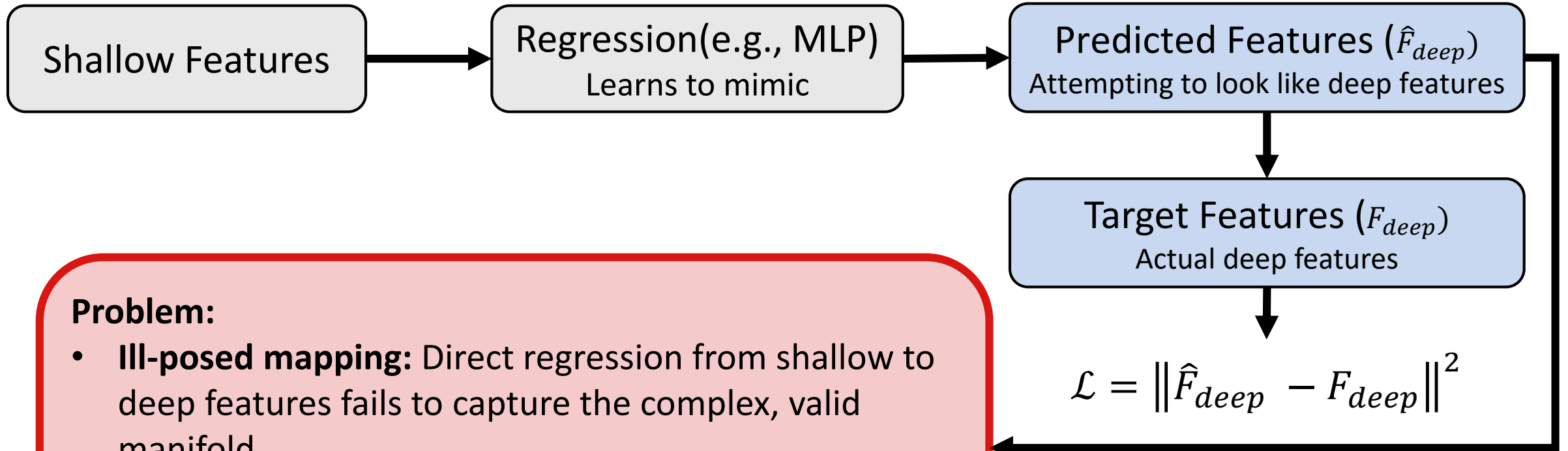
- **Fixed Computation Cost:** Always executes full network.
- **Inflexible:** Cannot adapt to varying constraints.

Motivation | Challenges when applying early-exits to VPR

- Three two challenges:
 - **Shared database requirement:** Early-exit features must be compatible
 - **Manifold mismatch:** Shallow features \neq Deep semantic features



Motivation | Naive solution for feature alignment

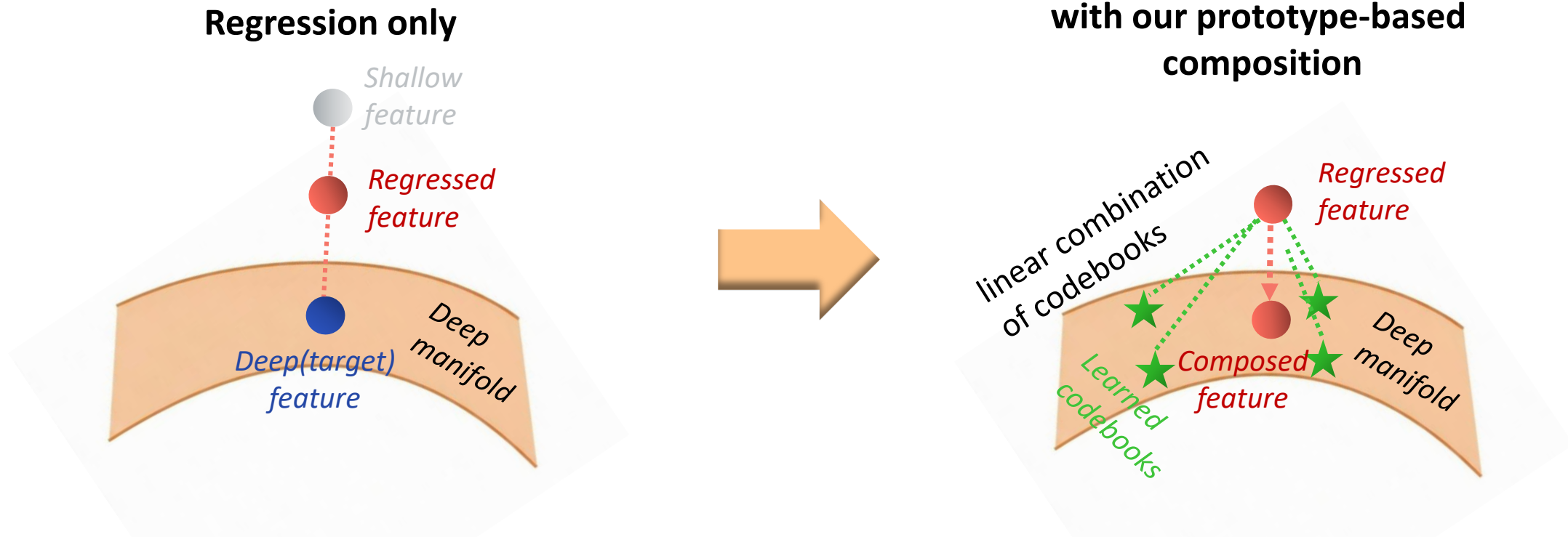


Problem:

- **Ill-posed mapping:** Direct regression from shallow to deep features fails to capture the complex, valid manifold.
- **Blurred features:** Consequently, the model produces "blurred" or averaged representations instead of distinct, sharp features.

Our Approach: Regression + Codebook Composition

- Instead of directly regressing feature values (which leads to blurring), we compose features using a pre-learned Reference Codebook.
- By limiting the output to the linear combination of codebook, the result is forced to stay on the valid semantic manifold.



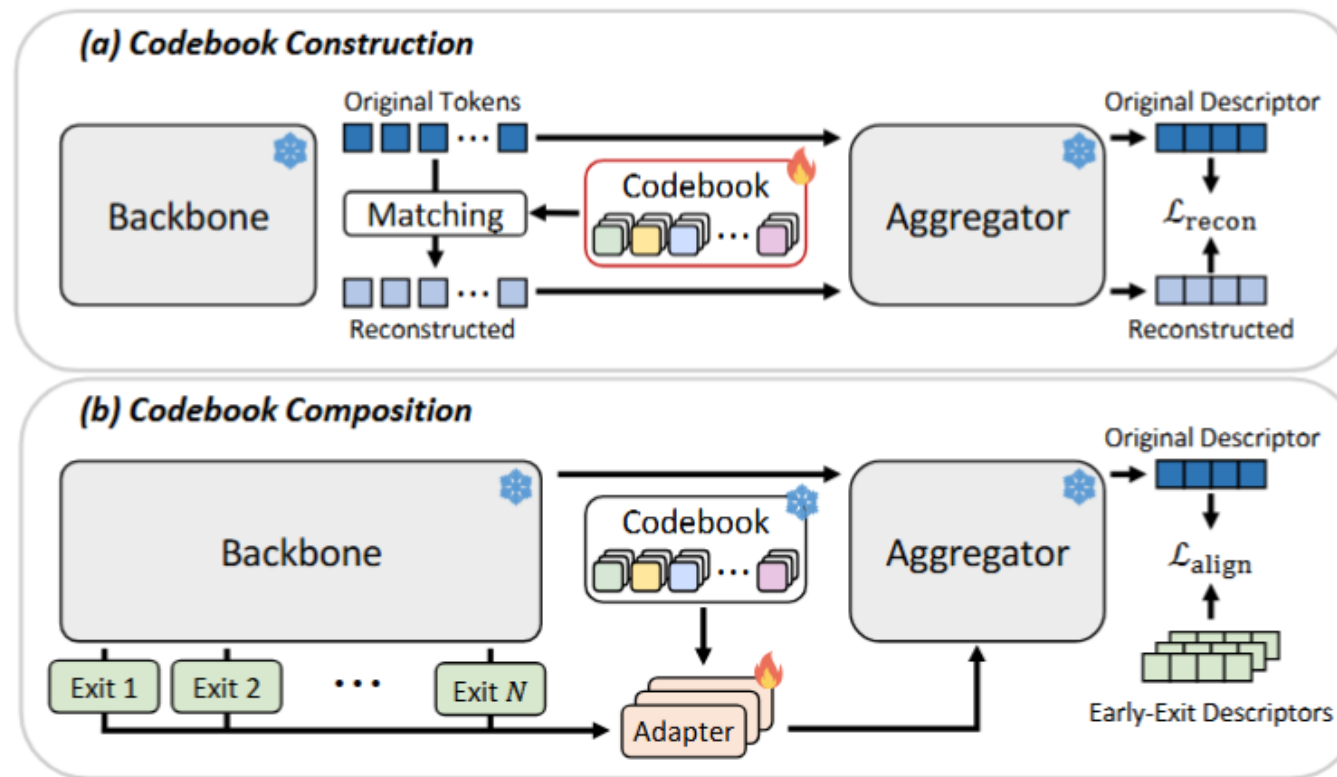
Our Approach: Two-Stage Pipeline for Codebook

- **Stage 1: Codebook Construction**

Learn reference codebook from deep features to span the semantic manifold.

- **Stage 2: Codebook Composition**

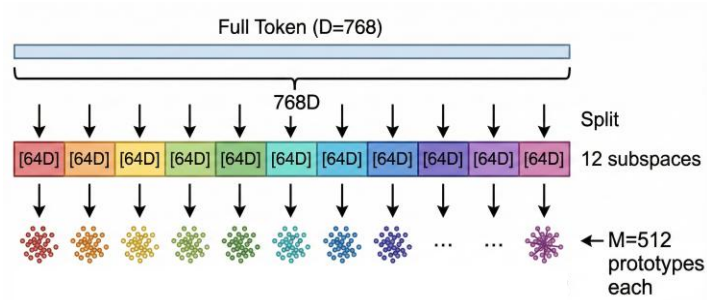
Train lightweight adapters at early exits to compose features from learned prototypes.



Our Approach: Stage 1 - Reference Codebook Learning

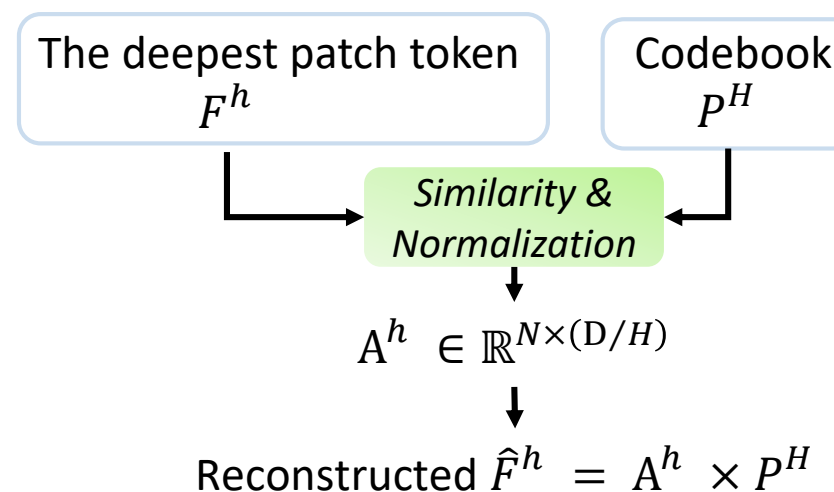
Goal of this stage: Learning the *Semantic Dictionary (codebook)* from *deepest layer*.

1. Product Quantization Decomposition



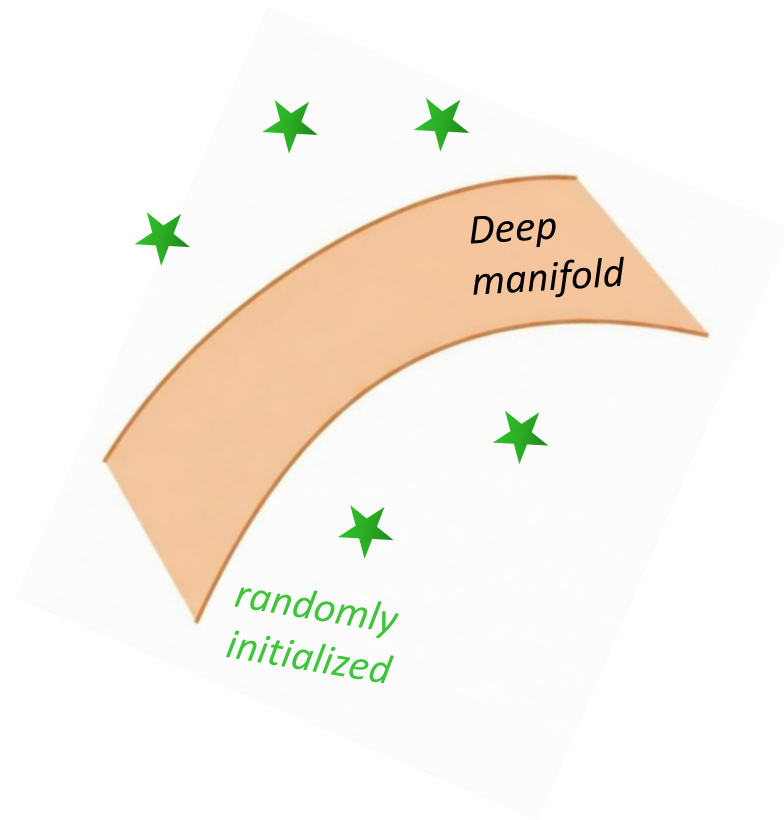
$$\text{Patch token } F = [F^{(1)}, F^{(2)}, \dots, F^{(H)}], \text{ where } F^h \in \mathbb{R}^{N \times (D/H)}$$
$$\text{Codebook } P = [P^{(1)}, P^{(2)}, \dots, P^{(H)}], \text{ where } P^h \in \mathbb{R}^{M \times (D/H)}$$

2. Reconstruction via Soft-Assignment:



3. Reconstruction loss: $\mathcal{L} = \|\text{aggregator}(\hat{F}^h) - \text{aggregator}(F^h)\|^2$

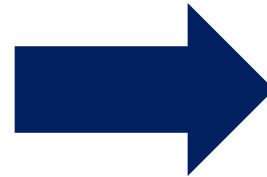
Our Approach: Stage 1 - Reference Codebook Learning



tic Diction

2. Recons

The de

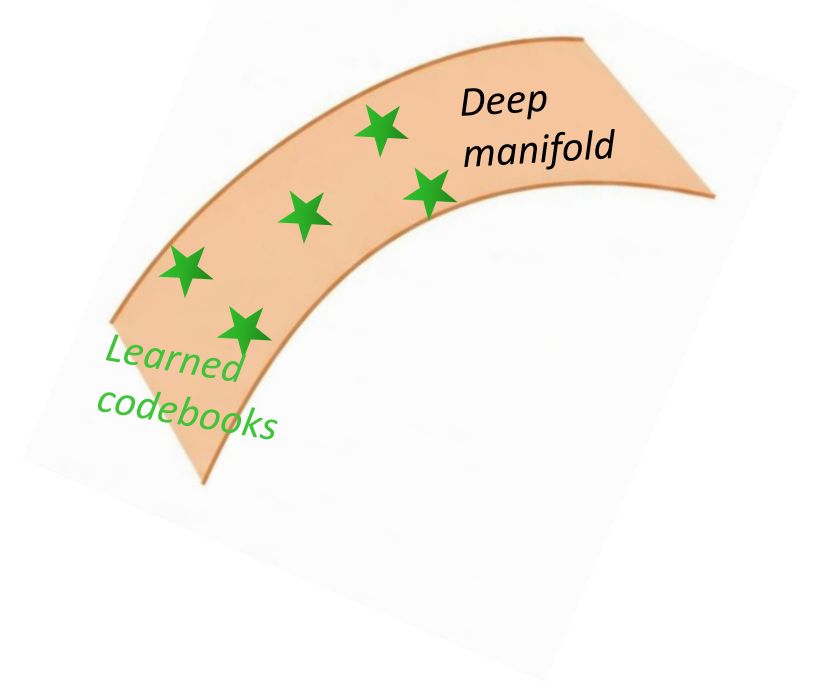


H)

(H)

regator(\hat{F}^h)

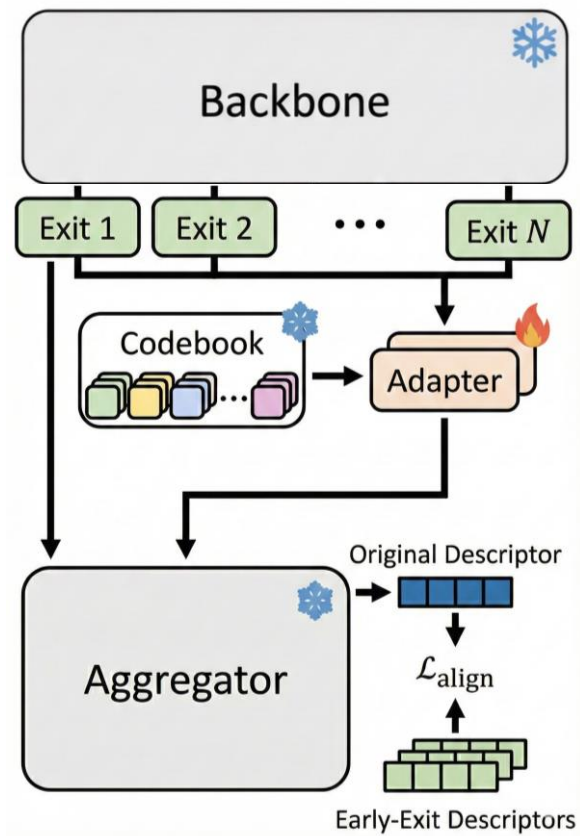
***After codebook learning
Projected on the manifold surface.***



Our Approach | Stage 2 – Codebook Composition

Goal of this stage: Training regressor(adapters) to align shallow features with deep semantics via prototype composition.

(b) Codebook Composition



1. (layer k) patch token $F_k \rightarrow$ adapter $\varphi_k \rightarrow$ regressed Q_k

 Learnable

2. Similarity & Normalization (Same as Stage 1) $\rightarrow A_k^h$

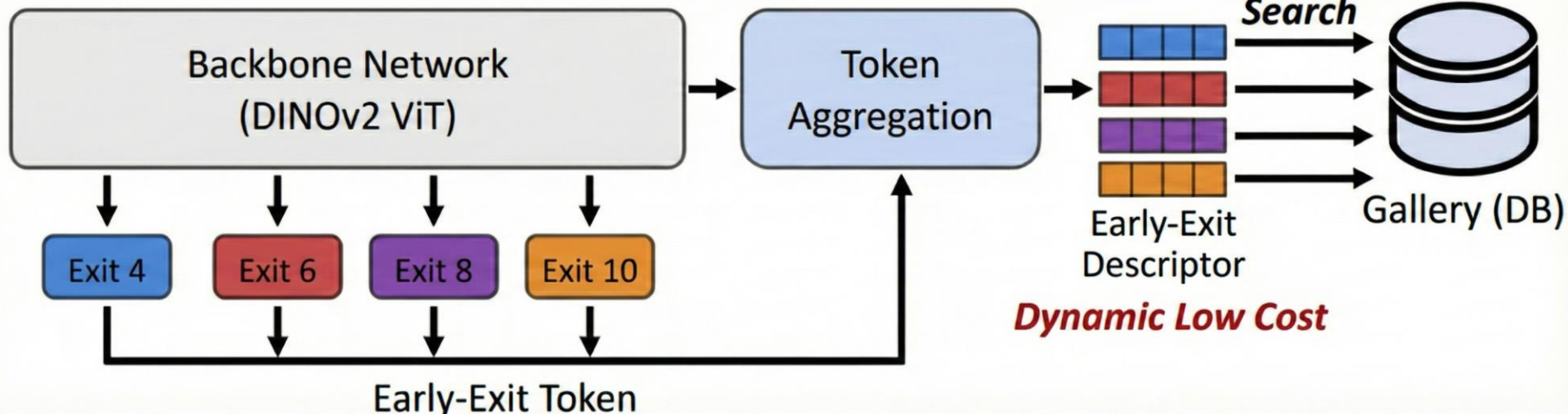
 FROZEN

3. Composed $\hat{F}_k^h = A_k^h \times P^H$

4. Concatenate over all heads.

Experimental Setup | Backbone and early exit layers

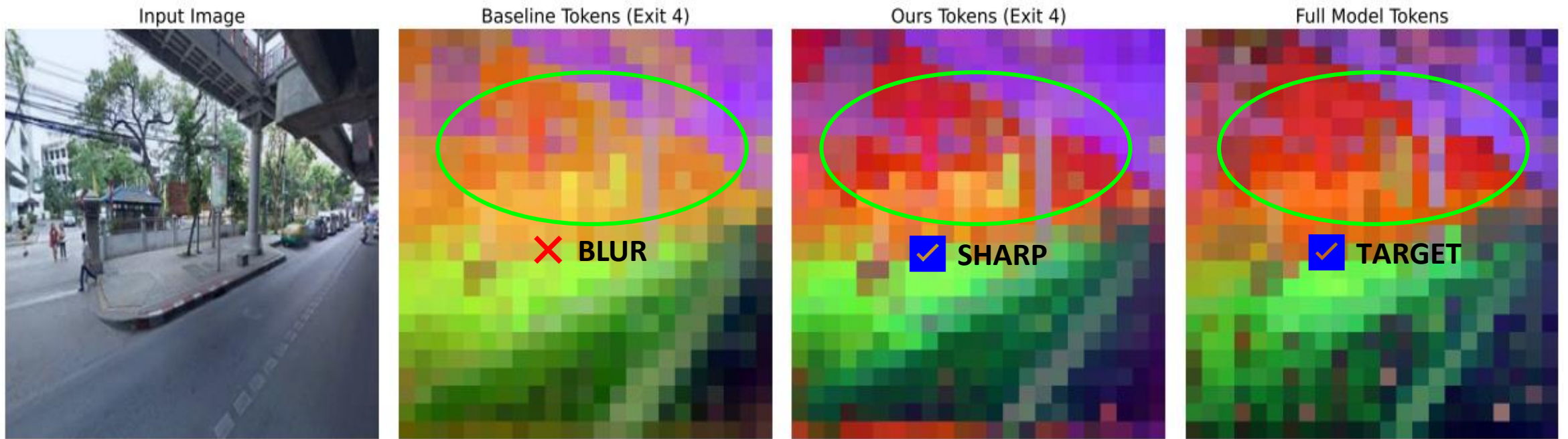
- Backbone: **Foundation model** of DINOv2-B (frozen)
 - 86M parameters
 - 12 transformer layers
 - Provides strong feature representations
- Early exit setup
 - Exit layers: **L4, L6, L8, L10** (out of 12 layers)
 - Progressive depth → Progressive performance



In-depth Analysis | Visualization of token features

Key message: Our codebook composition produces **sharper features** that better preserve semantic structure compared to blurred baseline regression.

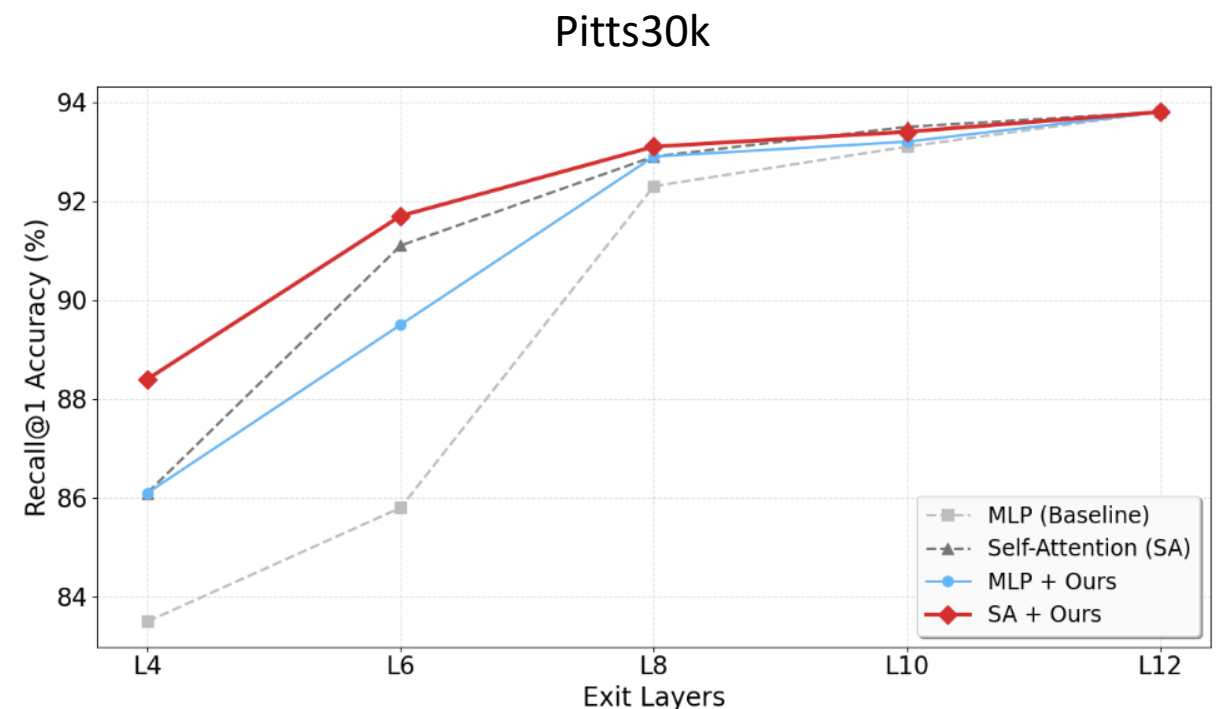
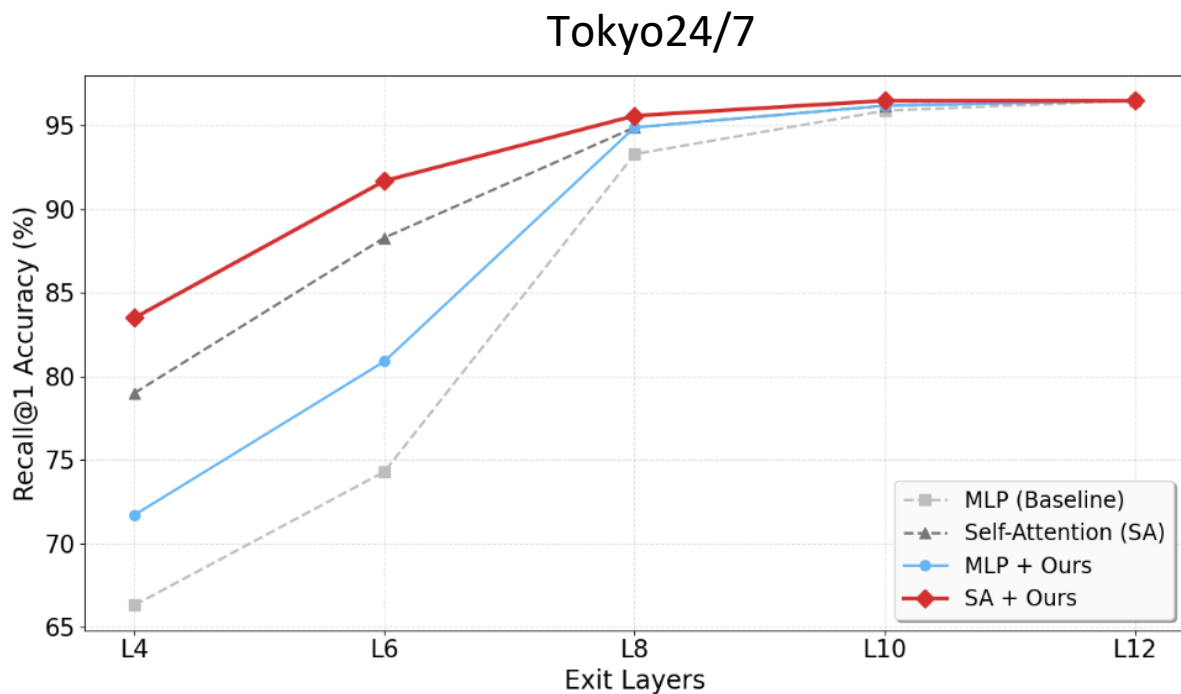
- Visualize token features in a RGB-like manner using 3-dim. PCA.



In-depth Analysis | Composition vs. Regression

Key message: Codebook composition dramatically outperforms pure regression, especially at early exits where semantic gap is largest.

- Evaluate **two regression baselines** (MLP and Self-Attention) and show that building our **codebook composition** on top of each.



Main Result | Universal Compatibility Across VPR methods

- **Universal Applicability:** Works with **any aggregator**, robust across diverse benchmarks with **NO architecture changes**.

Performance on multi-exits.

Method	Cost	R@1	R@5
<i>BoQ Aggregation (Pitts250k)</i>			
Full Model	100%	96.6	99.1
Ours _{L4}	41%	90.4	96.6
Ours _{L6}	57%	94.4	98.1
Ours _{L8}	72%	96.1	98.9
Ours _{L10}	87%	96.3	99.0

Multiple aggregators

Aggregator	Full	L4
SALAD	95.1	84.9
SuperVLAD	95.2	80.0
BoQ	96.6	90.4

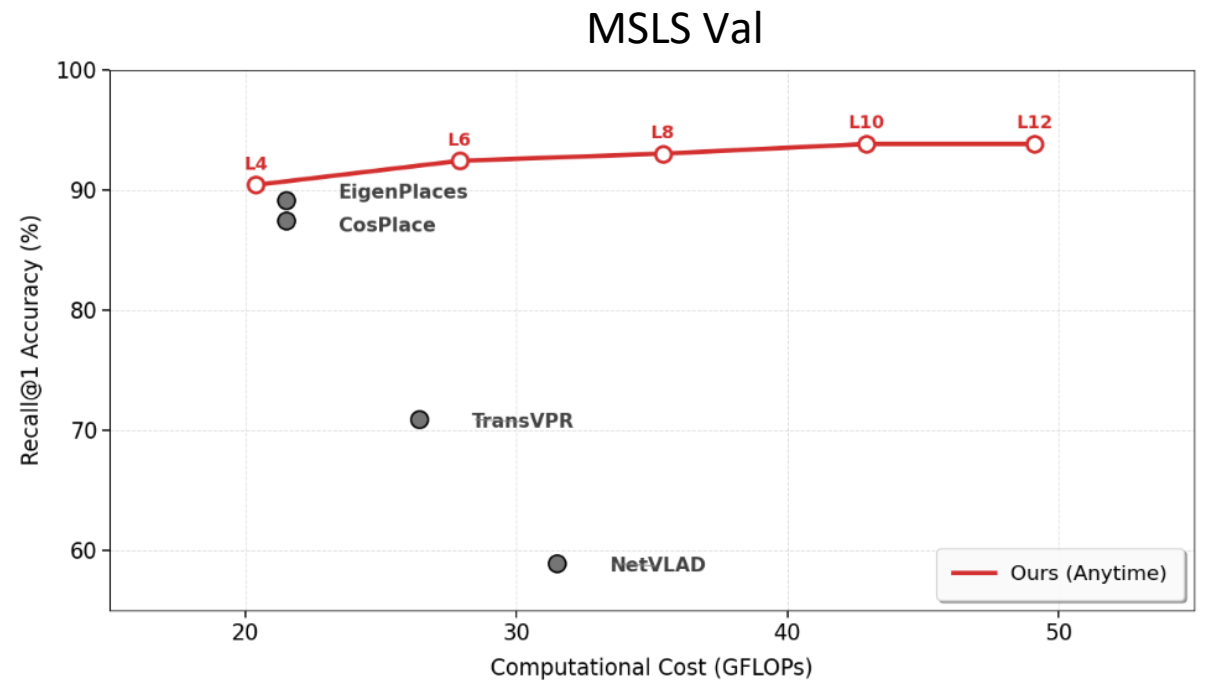
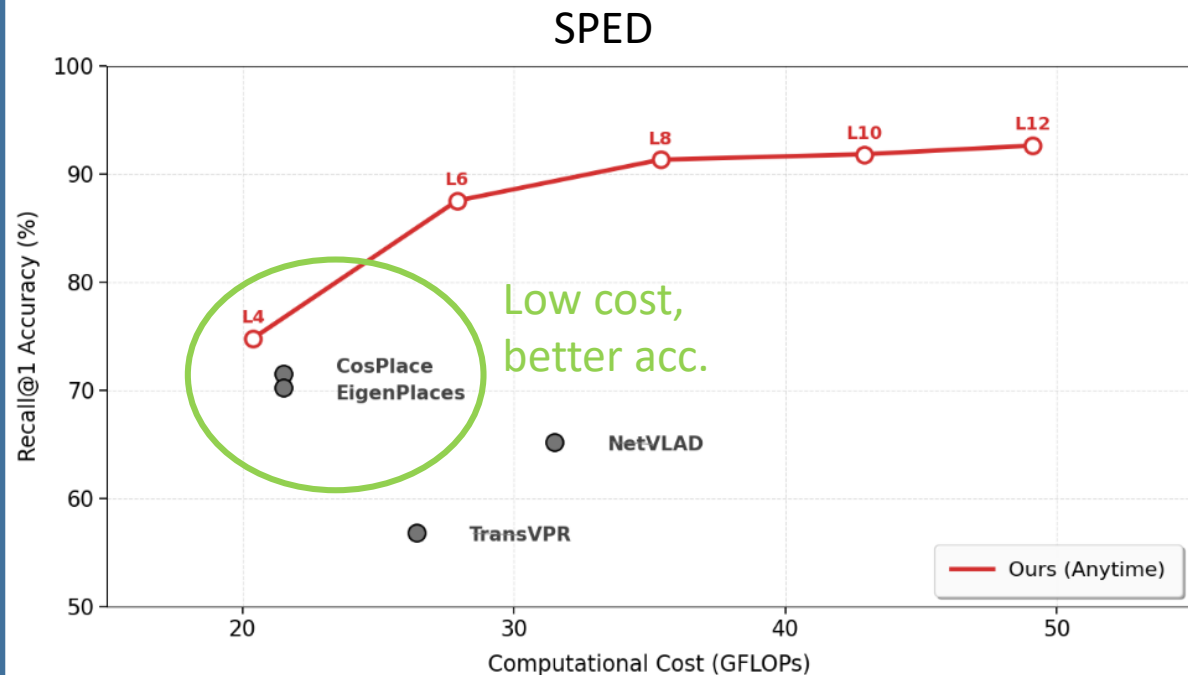
Across benchmarks (L4)

Dataset	R@1
Pitts250k	90.4
MSLS Val	90.4
Tokyo24/7	83.5
Eynsham	86.7
SPED	74.8

Main Result | Efficiency-Accuracy Trade-off

Key message: Our anytime framework **dominates conventional static methods** across the entire efficiency-accuracy spectrum.

- Comparison: Our **anytime** BoQ vs. **static** baselines (NetVLAD, TransVPR, CosPlace, EigenPlaces) on SPED & MSLS Val.



Ablation study | Design choices for codebook and composition

Key message: Every design choice in our codebook matters: PQ decomposition + Learned prototypes are both essential.

- Ablation study at L4 exit (most challenging scenario):
 - Test three codebook design variants
 - Evaluate on Pitts250k, MSLS Val, Tokyo24/7
 - Use BoQ aggregator for fair comparison

Method	Pitts250k		MSLS Val		Tokyo24/7	
	R@1	R@5	R@1	R@5	R@1	R@5
Ours	90.4	96.6	90.4	95.4	83.5	90.2
Monolithic codebook (no decomposition)	88.7	95.7	89.9	94.9	81.6	88.3
<i>k</i> -means clustered codebook	89.5	96.1	90.2	95.0	79.8	88.2
Randomly initialized codebook	88.6	95.7	89.9	94.5	75.9	86.3

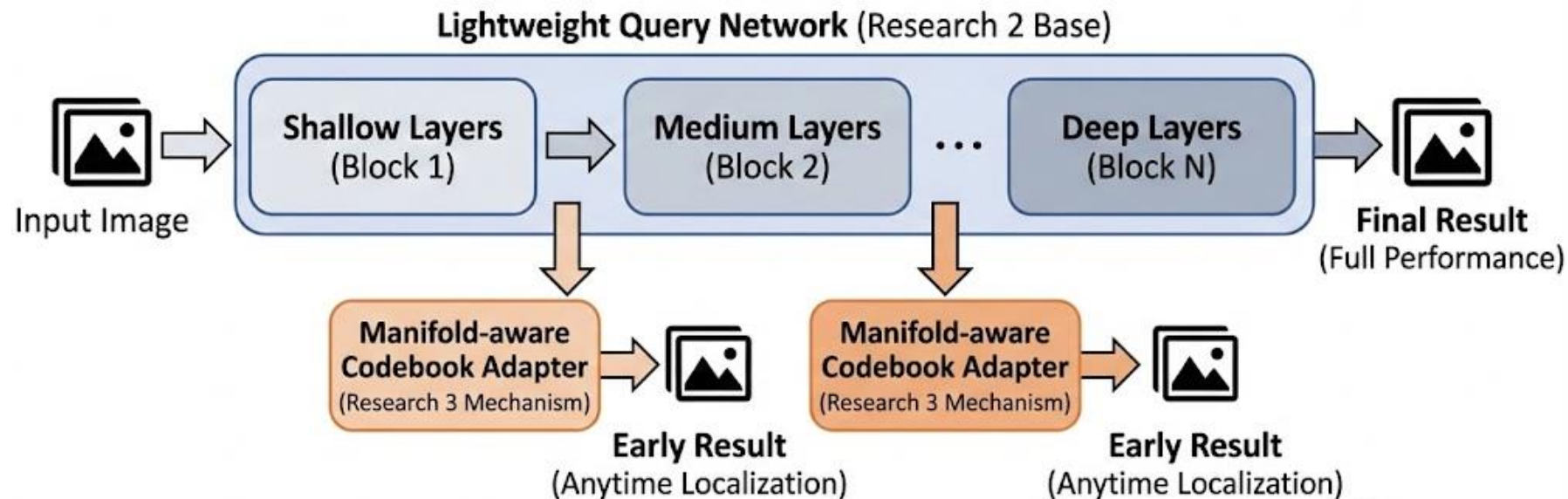
Future work - Limitations of heavy backbone in anytime VPR

- Current Anytime VPR validation context
 - Research 3 **validated dynamic inference** through early-exit strategy
 - Enables **flexible computation** based on resource availability

- Limitation of Current Anytime VPR:
 - Reliant on **heavy backbones** (DINOv2-B).
 - Even the earliest exit (L₄) incurs **higher latency than dedicated lightweight** models (e.g., MobileViT).
 - Not suitable for extreme edge devices (e.g., Micro-drones, IoT sensors).

Unified Framework - Dynamic lightweight VPR

- Concept: Bringing *Anytime* capability to *Lightweight* query networks.
- Proposed Direction:
 - Combine Research 2 (lightweight asymmetric query) + Research 3 (anytime early-exit)
 - Replace heavy backbone with lightweight architecture for query processing
 - Add early-exit capability to lightweight models
 - Enable true edge deployment with dynamic adaptation



Thank you for your attention!

Any question or feedback will be welcome