Learning from the unseen: Reducing train-test domain gaps by fine-tuning on reference images at test time

by

Olaf Verburg

In partial fulfilment of the requirements for the degree of
Master of Science
at Delft University of Technology,
to be defended publicly on Friday January 24th, 2025 at 14:00

Faculty: Mechanical Engineering
Programme: MSc Robotics

Student number:	5099722	
Graduation committee:	Dr. J.F.P. Kooij,	supervisor
	MSc M. Zaffar,	daily supervisor
	Dr. J. Kober,	graduation committee member

TU Delft

Learning from the unseen: Reducing train-test domain gaps by fine-tuning on reference images at test time.

O. Verburg (5099722)
Supervisor: J.P.F Kooij, Daily supervisor: M. Zaffar

***Abstract*—Visual place recognition (VPR) is a form of visual localization. Current approaches are designed to handle common VPR challenges, such as appearance and viewpoint variations. With the introduction of DINOv2, vision foundation models have been used as feature extractors to improve performance for VPR techniques, as they show great generalizing capabilities for image representations. By fine-tuning these large models on VPR-specific datasets, performance increases even more. A problem with these big VPR datasets is the bias towards urban environments. To solve this problem, we propose to use a simple pipeline to fine-tune existing techniques on the reference databases of test datasets. Our experiments show that performance improves by reference database fine-tuning for multiple techniques on different datasets. To handle appearance and viewpoint variations as well, image augmentations can be used during training. With this complete pipeline, techniques improve performance. The experiments show improvement even if a large query-reference domain gap exists for that dataset given that a part of the test queries are know during fine-tuning.**

I. Introduction

Visual Place Recognition (VPR) is an image retrieval problem that has been researched both from the robotics and computer vision fields [1]. VPR is often used to support different localization techniques, when these are not available [1]. The goal of VPR is to find the location of an input image (query) by comparing it to a reference set of images with known locations [2]. The problem of VPR originates from the field of Simultaneous Localization And Mapping (SLAM), where it is used for loop closure [3]. Traditional VPR techniques rely on small hand-crafted features and are limited to small environments [2]. Modern techniques consist of two steps: a deep learning feature extractor, which creates feature maps, and an aggregator to turn these maps into single descriptors [2]. Common deep-learning approaches used for VPR are Convolutional Neural Networks (CNNs) [4]–[9] and Vision Transformers (ViT) [10]–[16]. These large deep-learning based techniques are first pre-trained on large task-agnostic datasets and later fine-tuned on large VPR specific dataset [1], [17].

The main challenges seen in VPR datasets are due to appearance changes over time [18] (the day-night cycle, weather changes, or seasonal differences) and due to variation in viewpoint (looking from the same location to the other side of the road) [19]–[21]. To counter these challenges multiple big VPR-specific datasets have been designed to be as diverse as possible [6], [18], [22]. A trend seen in these large datasets is that they mostly consist of data taken from cars in urban environments [11], [18].

To improve the performance of existing techniques on challenging datasets, we propose to use the data available in the reference database to fine-tune techniques further. By using the images from the known test database to create a new training dataset, models can fine-tune to the specific domain of the test dataset, which could results in better retrieval of the unknown test queries (figure 2).

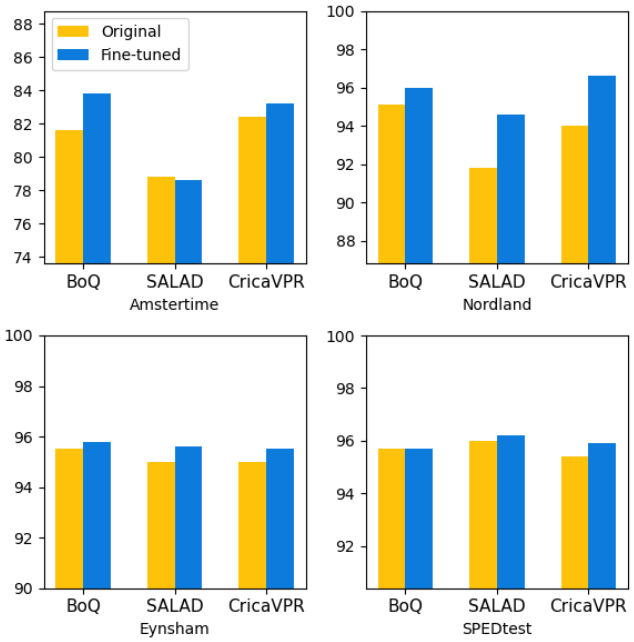

Fig. 1: The Recall@5 performance of almost all evaluated techniques improves or stays the same when the techniques (BoQ [15], SALAD [12], CricaVPR [14]) were fine-tuned on the reference database images using the proposed fine-tuning pipeline (see Figure 3). Each plot shows the results of the original models and the fine-tuned version of three techniques on one of the four evaluated datasets (Amstertime [23], Nordland [21], Eynsham [24], SPEDtest [25]).

This paper is structured as follows: chapter II presents the relevant works within VPR. Chapter III explains the complete pipeline and used components. The evaluated datasets and techniques for the experiments are presented and discussed in chapter IV. Chapter V show the results of the experiment along side discussion and ablations studies. The conclusion and further recommendations are presented last (chapter VI).

II. Related work

The original deep-learning methods used for VPR rely on CNN-based backbones, which were pre-trained on ImageNet-1k [26] and used off-the-shelf for VPR [27]. Currently, CNN-based techniques fine-tune the backbone using VPR-specific data to improve performance. These CNN backbone extract features, that are combined using specifically designed aggregation models [8], [28], [29]. With the introduction of

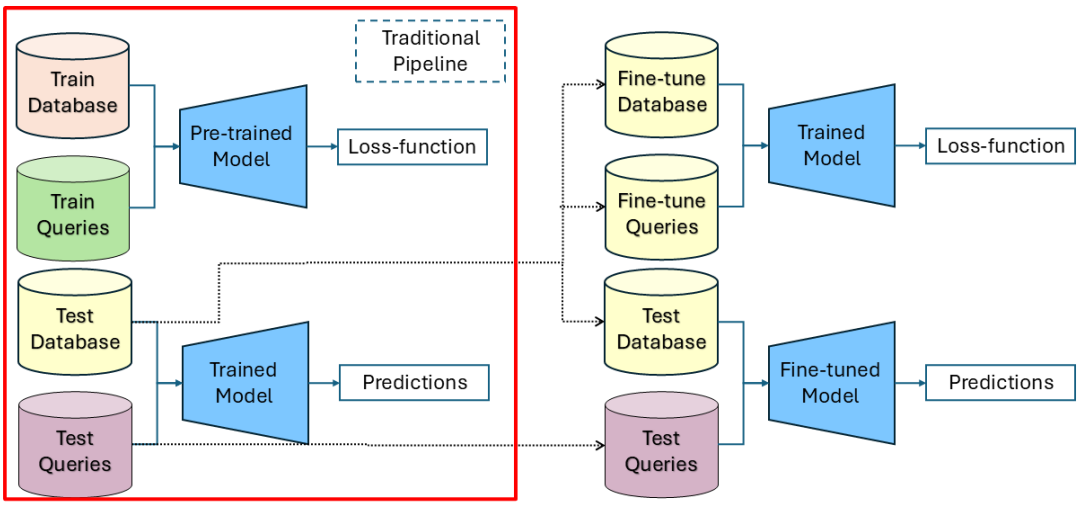

Fig. 2: Traditional VPR pipelines use different datasets to train and test on, during testing the database images are assumed known while the queries

vision foundation models (VFM) such as CLIP [30], DINO [31] and DINOv2 [32], the performance of VPR techniques improved substantially [11]–[13].

VFMs are big pre-trained models that can capture information from visual input to be used on downstream tasks with minimal or no fine-tuning at all [11], [30], [32]. The authors of [11] show that using an off-the-shelf DINOv2 backbone in combination with already existing aggregation techniques improves performance for challenging datasets. The next step in the development of VPR techniques is the application of fine-tuning on VPR-specific datasets for techniques that use VFM backbones [12], [14]. These large fine-tuned models can outperform recent CNN-based approaches [6], [7], despite their training methods not being as advanced [16].

The first fine-tuning methods do not consider specific datasets, but are designed to improve the performance of pre-trained CNN methods, which were first pre-trained on the commonly known ImageNet-1k dataset (14M images) [1], [26]. To improve performance with fine-tuning, large VPR-specific datasets were constructed to be used as downstream fine-tuning [6], [18], [22]. The use of these big datasets, in combination with new fine-tuning techniques[7], [18], [33], results in high performance for newer methods in VPR [7], [15], [16].

Training on large VPR-specific datasets improves performance for all test datasets but comes with a problem [11]. These big datasets consist mostly of data from urban locations, making them less suitable for datasets in other environments [34]–[36]. To counter this problem the images from the reference set can be used, as some recent works already have done [20], [37]–[39]. The authors of [39] and [37] leverage the information in the reference set to improve performance by enhancing the features of methods at test time. In [38], a technique is presented that utilizes the reference set images to train methods. A direction yet unexplored is to combine the reference set data with traditional fine-tuning methods.

Methods that use reference set images to train a model apply image augmentations to improve performance for unseen queries [20], [38], [40]. These methods all create image augmentations in different ways. The authors of [40] use image style augmentations to improve performance for different recording agents. The pipeline presented by [20] consists of two parts that work together to improve domain adaptation. First, a generative model is trained to learn the domain using a few target domain queries in combination with training data. In the second phase, they create new data using their generator model to train their domain adaptation model. The authors of [38] present a self-supervised learning pipeline to train a model from scratch. To improve performance for different domains, they include two augmentations; a set of image augmentations created using the Kornia library [41] and a 90-degree rotation geometry class. They state that they do not use random perspective augmentations because of the limited viewpoint variation in their evaluated datasets. The augmentations presented by [38] are most promising for our use case. The experiments done for the style augmentations were designed for small indoor environments [40], which is not the case for most VPR test datasets. The pipeline of [20] is designed to adapt a model to a new domain using a data generator and a specific model architecture, making it hard to apply to fine-tune any model on the go.

As stated, quite some research has been done to improve results for datasets with a large domain gap to the training data. We build upon the current research and contribute to the field of VPR in the following ways:

- A pipeline is introduced to fine-tune models based on the reference database of test datasets, without the need for additional validation or training data. By using this simple pipeline R@1 values improved with up to 2 percent for difficult datasets.
- It is observed that even for datasets with large query-reference domain gaps fine-tuning can improve performance, given that the challenges seen are represented by the used validation set.
- Multiple ablation studies are conducted, offering clearer insights into how our method works.

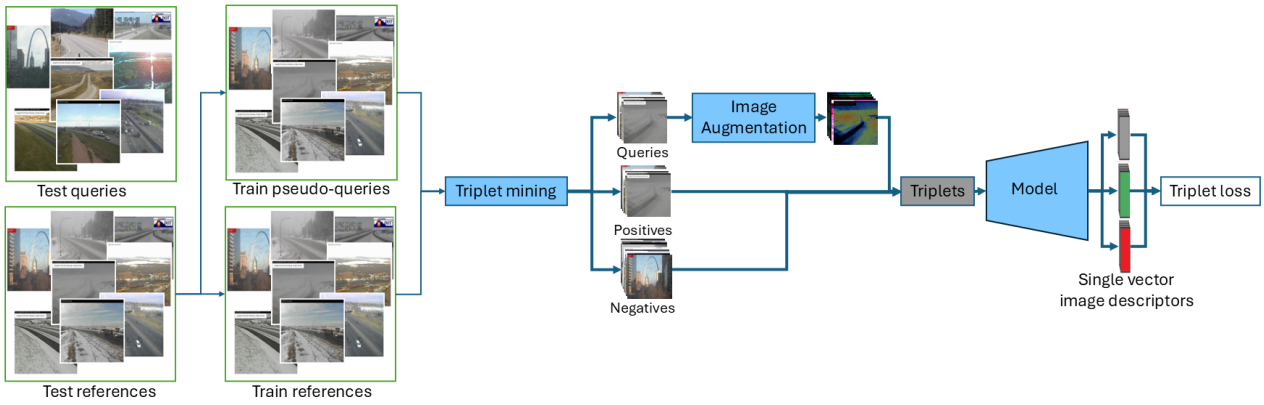

Fig. 3: Overview of our reference database fine-tune pipeline. Before training, triplets are mined once from the non-augmented reference images, every epoch new random image augmentations are applied to the query based on the probabilities used in [38].

III. Methodology

A. Problem definition

The problem of VPR is defined in the following way. Given an input image (query), the goal is to find its location by comparing it to a reference database of known location-image pairs. An image representation is created for the query (f_q) and reference images (f_r), by calculating the distance d between the query descriptor and the reference descriptors in the feature space the matching scores are created. The reference with the smallest distance to the query is the best match. The reference set is known before test time, while the queries are revealed at test time.

B. Reference database fine-tuning

Traditional VPR training datasets include training query and reference images [19], [22], [42], [43]. Newer techniques have presented methods for using datasets that are not structured this way. Still, they require the dataset to be either dense [6], or to be structured into multiple non-overlapping locations [18]. To use existing methods to fine-tune on the reference set only, pseudo-queries need to be selected from the reference set. To do this the reference set is copied, as we use reference images to find triplets to use for training.

$$L = \sum_{j} l(||\mathbf{f_q} - \mathbf{f_p}|| - ||\mathbf{f_q} - \mathbf{f_{n_j}}|| + m) \quad (1)$$

Where L is the triplet loss, l is the hinge loss ($l(x) = max(x, 0)$), m is the soft margin and $\mathbf{f_q}, \mathbf{f_p}$ and $\mathbf{f_{n_j}}$ represent the descriptors of the query, positive and negatives respectively.

C. Triplet mining

To mine the triplets, the positives are defined by selecting reference images within ten meters of the query and using the positive farthest from the query in the feature space. For queries which have a soft positive other than the exact match, the exact is not used as a positive in the triplets. To pick the hardest negatives, the reference images further than 25 meters are selected, from which the closest 2 images in the feature space are used as negatives. The triplets are created before any augmentations are applied. This method has been used by multiple works already [8], [9], [13]. The evaluated techniques (table II) did not train with the pipeline just presented. The training techniques presented in their works could not be used, as they rely on a big and neatly ordered dataset [12], [14], [15], [18],
The triplets are used to calculate the loss during training time (equation 1) [13].

D. Image augmentations

To improve the robustness of the fine-tuned models, the query images are augmented during training. The image augmentations will be created using the Kornia library [41]. To fine-tune, the same set of augmentations is used as done by [38], with the addition of the random perspective augmentations. The method of [38] uses a separate geometry module to include different perspectives, in our pipeline the random perspective is used because of the difference in evaluated datasets. Figure 4 presents some examples of augmented images.

IV. Experiments

A. Datasets

To evaluate the effect of reference set fine-tuning, we use multiple challenging datasets that show different domains compared to the training dataset used by the authors of the evaluated techniques (section IV-B). The training dataset is also presented below. The datasets are formatted using the code of [9], which has been used by recent papers [13], [14].
a) Amstertime: The Amstertime dataset was introduced in 2022 and contains 1231 query-reference pairs [23]. The dataset consists of reference images from the past and modern-day query images taken in Amsterdam, resulting in a unique and difficult VPR dataset.
b) SPEDtest: The SPEDtest dataset is a dataset collected using images from security cameras, the entire dataset consists of 2.5M images taken at 2543 different cameras. The test split is a subset of this dataset, it consists of 668 image pairs. The reference images are from the winter and the queries are images from the same cameras during summer time[25].
c) Nordland: The Nordland dataset is a set of recordings of a train journey through Nordland, the original dataset contains four sequences of 10 hours [21]. To evaluate, a scaled-down version is used, which consists of 27k queries and 27k reference images [44]. The reference images are

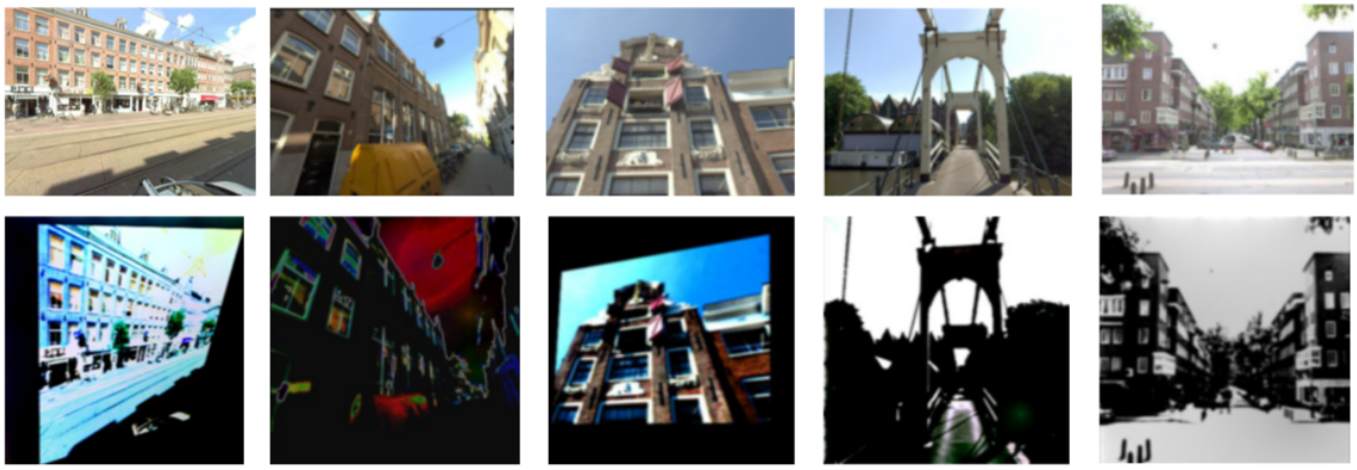

Fig. 4: Reference images from the Amstertime [23] dataset (upper row) and the variants with augmentations applied with the Kornia library [41] (lower row).

TABLE I: Overview of the evaluated test datasets. The main challenges that are present in each dataset are noted down **S**: change of seasons. **LT**: long-term change. **GS**: gray-scale, **W**: weather

Name	Type	N_r	N_q	Challenges
AmsterTime[23]	Long-term	1231	1231	**LT&GS**
Eynsham[24]	Country&Urban	23935	23935	**GS**
SPEDtest[25]	Security cams	668	668	**S&W**
Nordland[21]	Urban	27592	27592	**S**
GSV-Cities[18]	Training	529683		**LT&S&W**

taken in summer, and the query images are taken in the winter resulting in a big appearance change due to the large amounts of snowfall in winter [21]. Two main splits are used for the Nordland datasets, we use the test split with a threshold of 10 frames around the query. From this dataset, the final 10% of the queries are used as validation queries, and the rest are used for testing. By removing part of the dataset from the test we keep our test and validation sets separate, but this also results in a difference between our Nordland results and reported performance by the authors of the evaluated techniques.

d) Eynsham: The Eynsham dataset is a collection of images taken during a trip through the countryside of Oxford, the same route was driven twice to get the query and reference sets [24]. The images in this set are all gray-scale resulting in a unique challenge.

e) GSV-Cities: Google StreetView Cities (GSV-Cities) is a large-scale VPR dataset that is designed to fine-tune a pre-trained model for VPR [18]. It consists of 560k that span 67k places across the world, this dataset has been used by multiple works to train models [6], [7], [12], [14], [15], [18], [29]. While the dataset consists of images representing different challenges, a domain between the training data and the test datasets still exists (figure 5).

B. Evaluated techniques

The evaluated techniques are described in this section to show the effect of fine-tuning on different architectures. The main reason for choosing the techniques presented, is their training framework, as all methods are single-stage trained meaning that the fine-tuning is easier to use to fine-tune other VPR techniques.

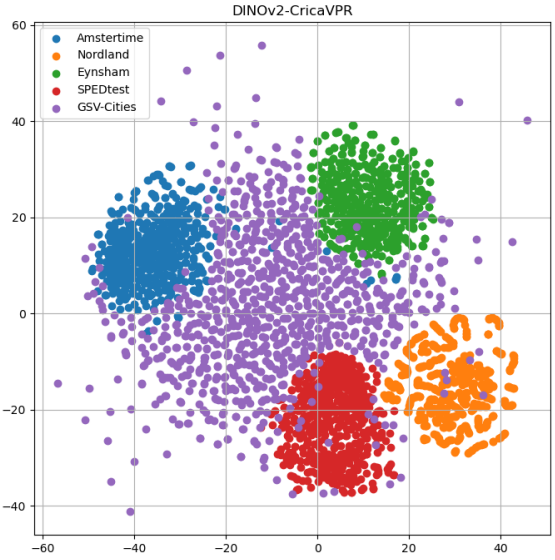

Fig. 5: T-SNE plot of the training dataset and the selected test datasets. The training data is a very diverse set, but still the test sets form their own separate cluster with little overlap, meaning that some but not many images similar to the test dataset are found.

a) Cross-image correlation-aware visual place recognition (CricaVPR) [14]: The architecture of CricaVPR is not different from the standard VPR pipeline, as it consists of a backbone and a simple aggregator. They add special adapter layers to the backbone to properly fine-tune the DINOv2 backbone for the VPR task. During training only their adapter layers are trained, instead of the complete DINOv2 model [14]. The aggregator of CricaVPR is a combination of the already existing generalized mean aggregator [28] and their novel technique, which compares feature map patches across different query and reference images [14]. By cross-referencing multiple queries in each batch using a multi-head attention layer [10], information is shared between images,

resulting in more discriminative image representations [14].

b) Sinkhorn algorithm for locally aggregated descriptors (SALAD) [12]: The second method tries to leverage the full potential of the DINOv2 backbone, by combining the clusters assigned to each local feature and the global feature into a single descriptor [12]. This method is inspired by NetVLAD, which assigns local features to clusters by clustering them in the feature space [8]. To prevent a bias towards the training data, the architecture of [12] learns how to assign the clusters from the local features directly with a fully connected layer and to optimise the assigned clusters using the Sinkhorn algorithm for optimal assignment [12], [45].

To train the pre-trained backbone, without losing the generalization capabilities [11], [32], the final four blocks of the model are trained [12].

c) Bag of learnable queries (BoQ) [15]: BoQ is the final method evaluated, it uses the strong attention mechanisms of the transformer models. The output of the backbone is first scaled down using linear projection the scaled-down feature is then passed through multiple BoQ blocks. Each block consists of two steps, first the input is passed through an encoder. The output of the encoder is used as input for the next BoQ blocks and as input for the attention mechanism. The second step is an attention mechanism, in which learnable queries are used to assess the importance of the input features by using a multi-head attention layer [10], [15]. The outputs of the attention layers of each BoQ block are concatenated, dimensionally reduced and normalized to gain the global image descriptor [15].

TABLE II: Overview of VPR techniques used for fine-tuning.

Name	Backbone	Backbone fine-tuning	Image size	# params	#trainable params	Dim.
CricaVPR[14]	DINOv2[32]	All blocks*	224x224	107M	20.2M	10752
SALAD[12]	DINOv2[32]	Last 4 ViT blocks	322x322	88.0M	31.3M	8448
BoQ[15]	DINOv2[32]	Last 2 ViT blocks	322x322	95.2M	22.8M	12288

* the original ViT blocks are frozen, but an additional adaptation layer is added to each block, which is trained.

C. Implementation details

We implemented the fine-tuning method based on other works. The commonly used triplet margin loss function with a margin of 0.1 is used to fine-tune [9], [13]. Before training triplets are mined once, each triplet consists of a hard positive ($< 10m$) and two hard negatives ($> 25m$). During training, mining is not performed because the triplets change in appearance due to the image augmentations applied to the query image in each triplet. The learning rate is empirically set to 1e-7 for BoQ and 1e-6 for CricaVPR and SALAD. The models train till the Recall@5 on the validation set has not improved for 15 epochs and 5 epochs for the larger datasets, which all show a clear decrease in recall on the validation set after this window. An epoch is the passing of the entire training dataset. The performance is evaluated every 1008 iterations to obtain the best model. For smaller datasets, such as SPEDtest and Amstertime, the performance is evaluated every complete epoch, as these datasets contain less than 1008 triplets. A batch size of 16 is applied for both training and evaluation. The input image size on which the techniques were originally evaluated is used both during fine-tuning and evaluation (table II).

The validation sets are created before evaluating the models. The reference images are copied and randomly augmented, these images are later used as queries during the validation step. Before training 30% of the reference images are selected to be used as validation queries, for these images the augmented version is loaded during validation. For Nordland, we split the test queries and use the first 90% of the test queries as test dataset and the final 10% as validation set (section V-C).

For evaluation, Recall@N (R@N) is used, which is defined by the percentage of queries for which at least one positive match exists in the top N retrieved images [7], [9], [12], [14], [15]. The positive threshold is defined differently for the used datasets so we follow the dataset configurations of [9]. For Amstertime and SPEDtest only the unique pair is positive, for Nordland all images within 10 frames are positive and for Eynsham, a positive is any reference image within $25m$ of the query image[7], [9], [12], [14], [15].

V. Results and Analysis

A. Quantitative results

The results of our experiments are shown in table III, we benchmarked the fine-tuned models against the original models. For the large datasets (Nordland and Eynsham), the fine-tuned models perform better than the original ones. On the smaller datasets, the results vary between the techniques. For SPEDtest no big improvements are seen. The experiments on Amstertime show inconsistent results for the different techniques, which is discussed in section V-C.

B. Qualitative results

In figure 6, the qualitative results are presented. The images displayed, are from cases when the original models all failed and all the fine-tuned models predicted correctly. As our model is not perfect, cases exist for which it is the other way around, the original models predicted it correctly but forgot after fine-tuning. In the images from Nordland, some obvious improvements are visible, in row one the fine-tuned models are able to recognize the second track while the original models did not. No images existed for the smaller dataset on which all methods improved. For Nordland 36 images existed for which the fine-tuned models perform better, while only one image exists for which all fine-tuned models perform worse. In the Eynsham dataset, the fine-tuned model all improved on four images and perform worse on non.

C. Discussion

The overall results of the experiments are positive, as we can see improvement across multiple datasets. Some of the datasets do however show no big change at all (SPEDtest), one possible reason for this is the size of the reference sets of these datasets. The smaller datasets contain less than 1000 images to fine-tune, which results in noisy behaviour during training time. For the larger datasets, we see a consistent improvement across the techniques. This noisy behaviour is also the reason for the larger patience values used on these datasets, this be described in-depth in the ablation studies.

An important thing that needs to be noted is the use of part of the queries as a validation set for Nordland. The effect of the use of test queries will be shown in the ablation studies. By using additional data during training, we did not

TABLE III: Results of the original models compared to the fine-tuned models. The first four columns show the R@N for the original models and the last 8 columns show the R@N for the fine-tuned models with the difference between the original models and the fine-tuned versions. Results in bold note the best performance between the original techniques and their fine-tuned equivalent.

Model	**Finetune/test dataset**	**Results original model**				**Results fine-tuned model**							
		R@1	**R@5**	**R@10**	**R@20**	**R@1**	Δ	**R@5**	Δ	**R@10**	Δ	**R@20**	Δ
BoQ	Amstertime	62.6	81.6	85.5	88.5	**64.9**	2.3	**83.8**	2.2	**87.6**	2.1	**90.9**	2.4
SALAD	Amstertime	58.2	78.8	83.7	**87.8**	**58.5**	0.3	**79.3**	0.5	**84.1**	0.4	**87.7**	-0.1
CricaVPR	Amstertime	64.3	82.4	**87.2**	91.3	**65.1**	0.8	**83.3**	0.9	**87.2**	0.0	**91.7**	0.4
Avg	Amstertime	61.7	80.9	85.5	89.2	**62.9**	1.2	**82.1**	1.2	**86.3**	0.8	**90.1**	0.9
BoQ	SPEDtest	**92.9**	**95.7**	**97.0**	98.0	92.8	-0.1	**95.7**	0.0	**97.0**	0.0	**98.2**	0.2
SALAD	SPEDtest	**92.3**	**96.0**	96.9	97.4	91.8	-0.5	**96.0**	0.0	96.7	-0.2	**97.7**	0.3
CricaVPR	SPEDtest	91.9	95.4	96.7	97.0	**92.3**	0.4	**96.4**	1.0	**97.2**	0.5	**97.7**	0.7
Avg	SPEDtest	**92.4**	95.7	96.9	97.5	92.3	-0.1	**96.0**	0.3	**97.0**	0.1	**97.9**	0.4
BoQ	Nordland	88.7	95.1	96.9	98.2	**91.0**	2.3	**96.0**	0.9	**97.5**	0.6	**98.4**	0.2
SALAD	Nordland	83.2	91.8	94.5	96.4	**85.8**	2.6	**93.5**	1.7	**95.6**	1.1	**97.3**	0.9
CricaVPR	Nordland	88.6	94.9	96.5	97.7	**92.3**	3.7	**96.6**	1.7	**97.8**	1.3	**98.6**	0.9
Avg	Nordland	86.8	93.9	96.0	97.4	**89.7**	2.9	**95.4**	1.4	**97.0**	1.0	**98.1**	0.7
BoQ	Eynsham	92.1	95.5	96.4	97.0	**92.2**	0.1	**95.8**	0.3	**96.6**	0.2	**97.2**	0.2
SALAD	Eynsham	91.4	95.0	95.9	96.6	**92.1**	0.7	**95.6**	0.6	**96.5**	0.6	**97.1**	0.5
CricaVPR	Eynsham	91.6	95.0	95.8	96.4	**91.9**	0.3	**95.5**	0.5	**96.5**	0.7	**97.0**	0.6
Avg	Eynsham	91.7	95.2	96.0	96.7	**92.1**	0.4	**95.6**	0.5	**96.5**	0.5	**97.1**	0.4

restrict the models to using the references, this approach did however allow us to show that even for datasets where an obvious query-reference domain gap exists (winter-summer), an improvement can be made by fine-tuning on the reference set. training

D. Ablations

a) Specialization: To evaluate the different fine-tuned techniques against each other, we evaluate all four fine-tuned BoQ models on the different datasets (figure 7). In the figure it can be seen that the fine-tuned model mostly improved performance for all datasets, except for the model fine-tuned on the Eynsham dataset. Multiple experiments were performed to analyze the difference between Eynsham and the other datasets. The first experiment evaluated the effect of using gray-scale images. Table IV presents the results of this experiment, it can be seen that this did not influence the results as much as fine-tuning on Eynsham. Another difference between Eynsham and the other datasets is the availability of multiple viewing directions. To test the effect of the viewing directions, we fine-tuned on Eynsham but only selected the closest match as positive. This experiment resulted in a decrease in performance on the Eynsham dataset. The main reason that this experiment probably failed, is because all but the positives were selected as negatives, meaning that some true positive reference images could be used as a negative for the triplets, which would result in the model learning the wrong things.

TABLE IV: Results on the Nordland and Eynsham datasets, BoQ fine-tuned on Nordland and on the completely gray-scale version of Nordland. Results in bold note the best results on the test dataset.

Model	**Test set**	**Fine-tune set**	**R@1**	**R@5**	**R@10**	**R@20**
BoQ	Eynsham	Original model	92.1	95.5	**96.4**	**97.0**
BoQ	Eynsham	Nordland	**92.2**	**95.6**	**96.4**	**97.0**
BoQ	Eynsham	Nordland gray	**92.2**	**95.6**	**96.4**	**97.0**
BoQ	Nordland	Original model	90.6	95.9	97.4	98.5
BoQ	Nordland	Nordland	92.5	**96.9**	**98.0**	**98.7**
BoQ	Nordland	Nordland gray	**92.6**	**96.9**	97.9	**98.7**

b) Learning rate: For each method, the learning rate at which the model originally was trained was not sufficient for fine-tuning. To find the right learning rate, the models were fine-tuned multiple times on the Amstertime dataset [23]. In table V, the results are presented. As can be seen, the original learning rates result in a reduction in performance after fine-tuning. The results are different for each of the models. For BoQ both 1e-7 and 1e-8 show good improvements in performance, we chose to use a final learning rate of 1e-7 for BoQ, as the model required too long to converge when using 1e-8. For CricaVPR, the results also show good performance when the model fine-tune with a learning rate of 1e-8, with the same problem as for BoQ as the models would not converge within a normal time limit, the second best learning rate for CricaVPR was 1e-6, which is used for the experiments. For SALAD a learning of 1e-6 is used, as that showed the best improvements.

TABLE V: Recalls of the model fine-tuned at different learning rates, the row with learning rate "-" shows the results of the original model. Results in bold note the best results between the different learning rates for each technique.

Model	**Dataset**	**LR**	**R@1**	**R@5**	**R@10**	**R@20**
BoQ	Amstertime	-	62.6	81.6	85.5	88.5
BoQ	Amstertime	1e-04	20.0	35.3	42.4	50.1
BoQ	Amstertime	1e-05	59.6	79.9	85.2	89.2
BoQ	Amstertime	1e-06	64.0	83.2	86.9	90.4
BoQ	Amstertime	1e-07	**64.9**	83.8	**87.6**	**90.9**
BoQ	Amstertime	1e-08	**64.9**	**83.9**	87.2	90.7
CricaVPR	Amstertime	-	64.3	82.4	87.2	91.3
CricaVPR	Amstertime	1e-04	56.9	75.2	82.3	85.1
CricaVPR	Amstertime	1e-05	62.1	81.6	86.8	90.0
CricaVPR	Amstertime	1e-06	**65.1**	83.3	**87.2**	**91.7**
CricaVPR	Amstertime	1e-07	64.9	83.2	**87.2**	91.6
CricaVPR	Amstertime	1e-08	**65.1**	**83.4**	**87.2**	**91.7**
SALAD	Amstertime	-	58.2	78.8	83.7	**87.8**
SALAD	Amstertime	1e-04	1.4	4.6	6.5	9.2
SALAD	Amstertime	1e-05	44.1	66.1	72.2	78.9
SALAD	Amstertime	1e-06	**58.7**	**79.3**	**84.1**	87.7
SALAD	Amstertime	1e-07	58.1	78.6	83.8	87.7
SALAD	Amstertime	1e-08	58.3	78.6	83.8	87.6

c) Use of test queries: To show the effect of reference set fine-tuning on datasets with a large query-reference domain gap, a part of the test queries of the Nordland dataset is used

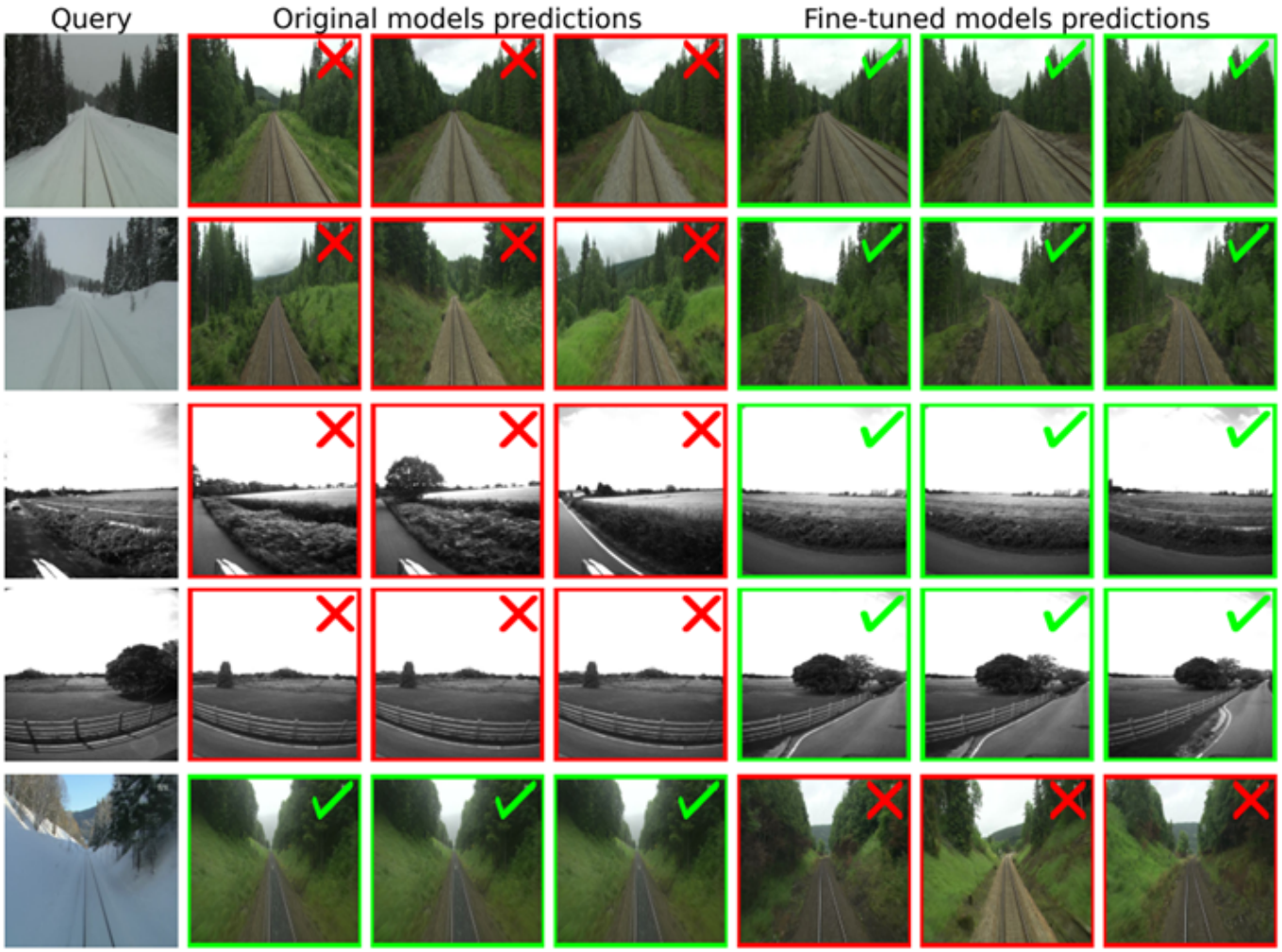

Fig. 6: Qualitative results from fine-tuning model on the reference set of test datasets. The fine-tuned models

to validate the model performance during training, the amount of queries to use for this is tested with the BoQ model. The results of our pipeline without any test queries is also presented for all three techniques.
The performance of fine-tuning with different amounts of test queries is similar from a certain amount of validation images. The main reason for smaller/no improvement for the 0.5% of test queries validation experiment is the size of the validation set, making it hard to really validate a change. The larger validation sets all show similar results.

TABLE VI: Results of the BoQ technique fine-tuned on Nordland, using different amounts of test queries as a validation set the percentage states to part of the test queries used for validation of the total amount of test queries (27k). The first row shows the performance of the BoQ technique without fine-tuning. Results in bold note the best results.

Model	Dataset	Amount of queries	R@1	R@5	R@10	R@20
BoQ	Nordland	Original model	90.4	95.9	97.4	98.5
BoQ	Nordland	0.50%	90.6	95.9	97.4	98.5
BoQ	Nordland	1%	92.5	**96.9**	**98.0**	**98.7**
BoQ	Nordland	5%	**92.7**	**96.9**	**98.0**	**98.7**
BoQ	Nordland	10%	92.5	**96.9**	**98.0**	**98.7**

The models performed differently when using the original pipeline, for BoQ the results were worse than the models fine-tuned with a validation set created from the test queries, but it still improved (table VII). For CricaVPR and SALAD, the performance decreased drastically when fine-tuning without the use of test queries, which is why we opted to use test queries for our main experiments on the Nordland datasets. The odd result from table VII is the Recall@1 for the BoQ technique, as this still shows quite some improvement compared to the original model. The main reason for this was that the R@1 value for BoQ started suffering after the best validation R@5 had been reached. The main trend seen in the test performance curves was the same for all methods.

TABLE VII: Performance of the different techniques fine-tuned on the complete Nordland dataset with our pipeline, results in bold note the best results between the original models and their fine-tuned equivalent.

Model	Dataset	Method	R@1	R@5	R@10	R@20
BoQ	Nordland	Original model	90.4	**95.9**	**97.4**	**98.5**
BoQ	Nordland	Our pipeline	**92.1**	**95.9**	96.9	97.7
CricaVPR	Nordland	Original	**91.2**	**96.2**	**97.6**	**98.5**
CricaVPR	Nordland	Our pipeline	89.3	93.9	95.4	96.6
SALAD	Nordland	Original	**85.9**	**93.5**	**95.6**	**97.2**
SALAD	Nordland	Our pipeline	84.4	92	94.3	96

d) Validation split selection: To create the validation split 30% of the database was randomly selected to not be included in the triplets, but to be used as validation queries. Normally VPR pipelines use a distinct validation and training set, to prevent overfitting on the training data. As the goal for our fine-tuning pipeline is to learn the domain of the training data

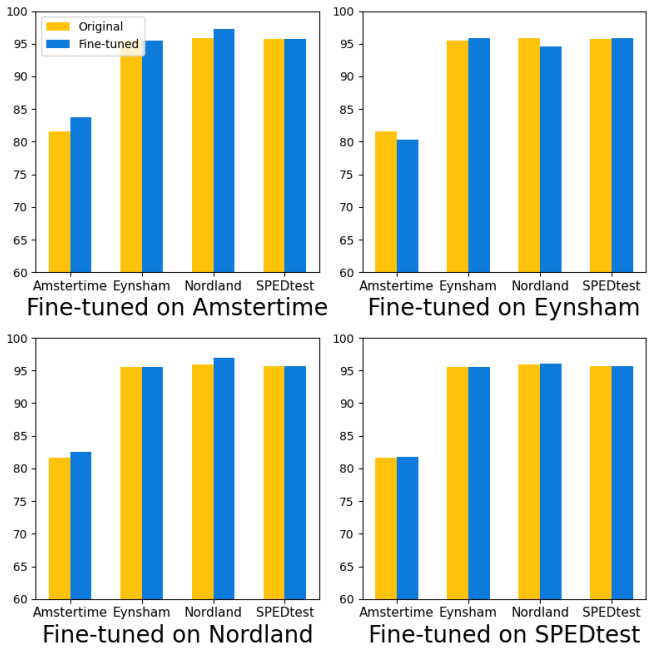

Fig. 7: The Recall@5 results of the fine-tuned BoQ models on the different test datasets. Each plot shows the results of one of the fine-tuned models, each bar shows the results of the original model and the fine-tuned model on the dataset beneath the bar.

better both randomly selected and a distinct split (first 70% training data, last 30% validation queries). This was tried on both Nordland and Eynsham, as these were the only datasets that covered a sequence of images, the other two datasets were a collection of unique image pairs.

In table VIII, the results are presented. For both datasets the performance improved more with the separation between the validation and training images within the reference set. The main reason for this was the similarity between the training data and the test data. For the Nordland experiments, part of the test queries were used as to validate and the 30% of the reference images, which were meant to be used as a validation set, were discarded. With the denser training data, better triplets were created, which resulted in better improvement.

TABLE VIII: Results of BoQ fine-tuned with randomly selected and separated validation sets. Results of a reduced version of Nordland (not the 10% used for validation) and on Eynsham. Results in bold note the best results for each dataset.

Model	Dataset	Validation split	R@1	R@5	R@10	R@20
BoQ	Nordland	Original model	88.7	95.1	96.9	98.2
BoQ	Nordland	Random	91.0	96.0	97.5	98.4
BoQ	Nordland	Seperate	**91.8**	**96.6**	**97.8**	**98.7**
BoQ	Eynsham	Original model	92.1	95.5	96.4	97.0
BoQ	Eynsham	Random	92.2	95.8	96.6	97.2
BoQ	Eynsham	Seperate	**92.5**	**95.9**	**96.7**	**97.3**

e) Training positives threshold: Most of the current VPR research papers use a positive threshold of 25m for testing and validation and 10m for training to make the models more discriminative [8], [9], [13]. As the testing data is used during fine-tuning, we can maybe leverage this information better if we use a positive threshold of 25m during training as well as during validation and testing. As this variable only affect datasets that are location based, these experiments are only performed for Eynsham and Nordland.

The results in table IX show an improvement on both Eynsham and Nordland for training positives threshold of 25m. The difference is bigger for Eynsham than Nordland, this is because the Nordland train dataset changes less over distance compared to the Eynsham dataset, which is captured on the road from a car. The improvement is an interesting result, as the 10m hard positive limit has been used for normal training a lot in previous works. This could also mean that a different training positives threshold could benefit other training pipelines as well.

TABLE IX: Results of BoQ fine-tuned with a positive threshold of 10m and of 25m. Results of a reduced version of Nordland (not the 10% used for validation) and on Eynsham. Results in bold note the best results for each dataset.

Model	Dataset	Train positive threshold(m)	R@1	R@5	R@10	R@20
BoQ	Nordland	Original model	88.7	95.1	96.9	98.2
BoQ	Nordland	10	91.0	96.0	**97.5**	**98.4**
BoQ	Nordland	25	**91.2**	**96.1**	**97.5**	98.3
BoQ	Eynsham	Original model	92.1	95.5	96.4	97.0
BoQ	Eynsham	10	92.2	95.8	96.6	97.2
BoQ	Eynsham	25	**92.9**	**96.1**	**96.9**	**97.4**

f) Patience: To stop fine-tuning after the best R@5 has been reached on the validation set, a patience is defined. To evaluate the effect of the patience parameter, the results of a run without early stopping is analyzed. For the bigger datasets (Nordland and Eynsham) it could be seen that the best results are already achieved when a patience of 2 is used instead of the 5 on which the original experiments were performed. For the smaller experiments (figure 8), a patience of 15 is selected, while this was not the best/final R@5 on the validation set, it showed decent results while maintaining normal fine-tuning times.

g) Amstertime: Another important thing noticed during the experiments was the inconsistency for the Amstertime dataset. While looking through this dataset, multiple image pairs appeared that showed the same query or reference image, but excluded other pairs with the same image from being marked as true positive. We tried to clean up the dataset but did not succeed in doing so.

VI. Conclusions and Recommendations

In this paper, a fine-tuning pipeline is presented that uses images from the reference database of test datasets to fine-tune techniques. The performed experiments show that the method works with multiple existing techniques across some of the evaluated datasets with an average increase of 0.9%point recall@5 across the evaluated datasets and techniques. With the results of the experiments we conclude that useful information exists in the reference databases of test datasets, our pipeline is not able to evaluate the effectiveness of training for datasets with large query-reference domain gaps. The pipeline is able to do so if part of the test queries are available during fine-tuning. The ablation studies show that multiple improvements can be made on our pipeline to improve fine-tuning even more. Next to the increase in performance, the experiments also show that performance can even improve for datasets with large query reference domain gaps as long as test queries can be used to evaluate the model during fine-tuning.

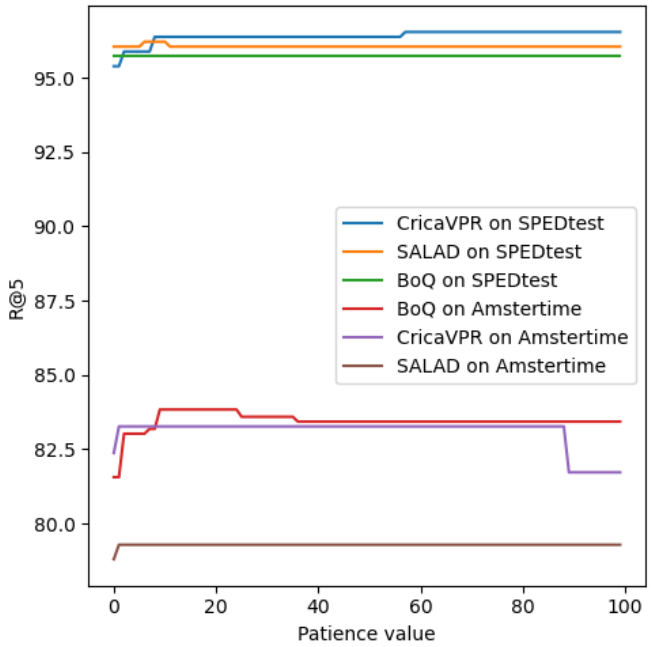

Fig. 8: Recall@5 performance of all models on Amstertime and SPEDtest. The recall values change because a smaller patience results in training being stopped when a local minimum has been reached. The legend is ordered highest to lowest recall@5 with patience 15.

In future work, the use of the information can be done even better. An adaptation of the highly used multi-similarity loss with online mining could be used to create better triplet and improve performance even more.

While we performed multiple ablation studies, some blanks could be filled in, in the experiment an image augmentation set has been used from previous works. On the augmentations, no ablations were performed, in future work different augmentations could be used to recreate specific domain changes between the reference images and the queries (day-night or summer-winter) by using generative AI models, which have become popular in recent years. We also show in our ablations that some of the settings we chose, were suboptimal. By combining all the shown improved ablation settings, better results might be possible. For the Amstertime dataset, a new refined version should be made, as this is currently an important VPR benchmark dataset, due to the challenge it poses. By refining the dataset to have a correct ground truth, a clearer insight would be gained into how difficult this dataset is. Another direction that could be explored would be to create an ensemble of different fine-tuned models, as they are each specialized in certain environmental situations, it could create a good all-around technique.

References

[1] X. Zhang, L. Wang, and Y. Su, "Visual place recognition: A survey from deep learning perspective," *Pattern Recognition*, vol. 113, p. 107 760, 2021.

[2] S. Schubert, P. Neubert, S. Garg, M. Milford, and T. Fischer, "Visual place recognition: A tutorial," *IEEE Robotics & Automation Magazine*, pp. 2–16, 2023.

[3] H. DurrantWhyte and T. Bailey, "Simultaneous localization and mapping: Part i," *IEEE Robotics & Automation Magazine*, vol. 13, no. 2, pp. 99–110, 2006.

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, pp. 1097–1105, 2012.

[5] Z. Chen, A. Jacobson, N. Snderhauf, *et al.*, "Deep learning features at scale for visual place recognition," in *Proceedings of the IEEE international conference on robotics and automation (ICRA)*, 2017, pp. 3223–3230.

[6] G. Berton, C. Masone, and B. Caputo, "Rethinking visual geo-localization for large-scale applications," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[7] G. Berton, G. Trivigno, B. Caputo, and C. Masone, "EigenPlaces: Training viewpoint robust models for visual place recognition," *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023.

[8] R. Arandjelovi, P. Gront, A. Torii, T. Pajdla, and J. ivic, "NetVLAD: CNN architecture for weakly supervised place recognition," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[9] G. Berton, R. Mereu, G. Trivigno, *et al.*, "Deep Visual Geo-Localization Benchmark," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[10] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, "Attention is all you need," *Advances in Neural Information Processing Systems*, 2017.

[11] N. Keetha, A. Mishra, J. Karhade, *et al.*, "AnyLOC: Towards universal visual place recognition," *IEEE robotics and automation letters*, vol. 9, no. 2, pp. 1286–1293, 2024.

[12] S. Izquierdo and J. Civera, "Optimal transport aggregation for visual place recognition," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[13] F. Lu, L. Zhang, X. Lan, S. Dong, Y. Wang, and C. Yuan, "Towards seamless adaptation of pre-trained models for visual place recognition," *Proceedings of the IEEE International Conference on Learning Representations (ICLR)*, 2024.

[14] F. Lu, X. Lan, L. Zhang, D. Jiang, Y. Wang, and C. Yuan, "CricaVPR: Cross-image correlation-aware representation learning for visual place recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024, pp. 16 772–16 782.

[15] A. Ali-bey, B. Chaib-draa, and P. Gigure, "BoQ: A place is worth a bag of learnable queries," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024, pp. 17 794–17 803.

[16] I. Tzachor, B. Lerner, M. Levy, *et al.*, "EffoVPR: Effective foundation model utilization for visual place recognition," *arXiv preprint arXiv:2405.18065*, 2024.

[17] C. Masone and B. Caputo, "A survey on deep visual place recognition," *IEEE Access*, vol. 9, pp. 19 516–19 547, 2021.

[18] A. Ali-Bey, B. Chaib-Draa, and P. Gigure, "GSV-Cities: Toward appropriate supervised visual place recognition," *Neurocomputing*, vol. 513, pp. 194–203, 2022.

[19] A. Torii, R. Arandjelovi, J. ivic, M. Okutomi, and T. Pajdla, "24/7 place recognition by view synthesis," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[20] V. Paolicelli, G. Berton, F. Montagna, C. Masone, and B. Caputo, "Adaptive-Attentive Geolocalization from few queries: A hybrid approach," *Frontiers in computer science*, vol. 4, 2022.

[21] N. Suenderhauf, P. Neubert, and P. Protzel, "Are we there yet? challenging SeqSLAM on a 3000 km journey across all four seasons," in *Proceedings of the ICRA 2013 Workshop on Long-Term Autonomy*, 2013, pp. 1–3.

[22] F. Warburg, S. Hauberg, M. Lpez-Antequera, P. Gargallo, Y. Kuang, and J. Civera, "Mapillary street-level sequences: A dataset for lifelong place recognition," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[23] B. Yildiz, S. Khademi, R. Siebes, and J. C. Van Gemert, "AmsterTime: A visual place recognition benchmark dataset for severe domain shift," *26th International Conference on Pattern Recognition (ICPR)*, 2022.

[24] M. Cummins and P. Newman, "Appearance-only SLAM at large scale with FAB-MAP 2.0," *The International Journal of Robotics Research*, vol. 30, no. 9, pp. 1100–1123, 2010.

[25] Z. Chen, L. Liu, I. Sa, Z. Ge, and M. Chli, "Learning context flexible attention model for long-term visual place recognition," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 4015–4022, 2018.

[26] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and F. Li, "ImageNet: A large-scale hierarchical image database," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

[27] Z. Chen, O. Lam, A. Jacobson, and M. Milford, "Convolutional neural network-based place recognition," *Australasian Conference on Robotics and Automation (ACRA)*, 2014.

[28] F. Radenovi, G. Tolias, and O. Chum, "Fine-tuning CNN image retrieval with no human annotation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 7, pp. 1655–1668, 2019.

[29] A. Ali-Bey, B. Chaib-Draa, and P. Gigure, "MixVPR: Feature mixing for visual place recognition," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[30] A. Radford, J. W. Kim, C. Hallacy, *et al.*, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*, PMLR, 2021, pp. 8748–8763.

[31] M. Caron, H. Touvron, I. Misra, *et al.*, "Emerging properties in self-supervised vision transformers," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 9650–9660.

[32] M. Oquab, T. Darcet, T. Moutakanni, *et al.*, "DINOv2: Learning robust visual features without supervision," *Transactions on Machine Learning Research*, 2023.

[33] M. Leyva-Vallina, N. Strisciuglio, and N. Petkov, "Data-efficient large scale place recognition with Graded Similarity Supervision," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[34] C. Boittiaux, C. Dune, M. Ferrera, *et al.*, "Eiffel tower: A deep-sea underwater dataset for long-term visual localization," *The International Journal of Robotics Research*, vol. 42, no. 9, pp. 689–699, 2023.

[35] M. Schleiss, F. Rouatbi, and D. Cremers, "VPAIR – aerial visual place recognition and localization in large-scale outdoor environments," *ICRA 2022 Aerial Robotics Workshop*, 2022.

[36] R. Sahdev and J. K. Tsotsos, "Indoor place recognition system for localization of mobile robots," in *13th Conference on Computer and Robot Vision (CRV)*, 2016, pp. 53–60.

[37] P. Neubert and S. Schubert, "SEER: Unsupervised and sample-efficient environment specialization of image descriptors," *Robotics: Science and Systems (R: SS)*, 2022.

[38] M. A. Musallam, V. Gaudillire, and D. Aouada, "Self-supervised learning for place representation generalization across appearance changes," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2024, pp. 7448–7458.

[39] M. Zaffar, L. Nan, and J. F. Kooij, "On the estimation of image-matching uncertainty in visual place recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024, pp. 17 743–17 753.

[40] P. Wozniak and D. Ozog, "Cross-domain indoor visual place recognition for mobile robot via generalization using style augmentation," *Sensors*, vol. 23, no. 13, p. 6134, 2023.

[41] E. Riba, D. Mishkin, D. Ponsa, E. Rublee, and G. Bradski, "Kornia: An open source differentiable computer vision library for pytorch," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2020, pp. 3674–3683.

[42] A. Torii, J. ivic, T. Pajdla, and M. Okutomi, "Visual Place Recognition with repetitive structures," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2013.

[43] W. Maddern, G. Pascoe, C. Linegar, and P. Newman, "1 year, 1000 km: The Oxford RobotCar dataset," *The International Journal of Robotics Research*, vol. 36, no. 1, pp. 3–15, 2016.

[44] S. Hausler, S. Garg, M. Xu, M. Milford, and T. Fischer, "Patch-NetVLAD: Multi-scale fusion of locally-global descriptors for place recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 14 141–14 152.

[45] M. Cuturi, "Sinkhorn distances: Lightspeed computation of optimal transport," *Advances in neural information processing systems*, vol. 26, 2013.