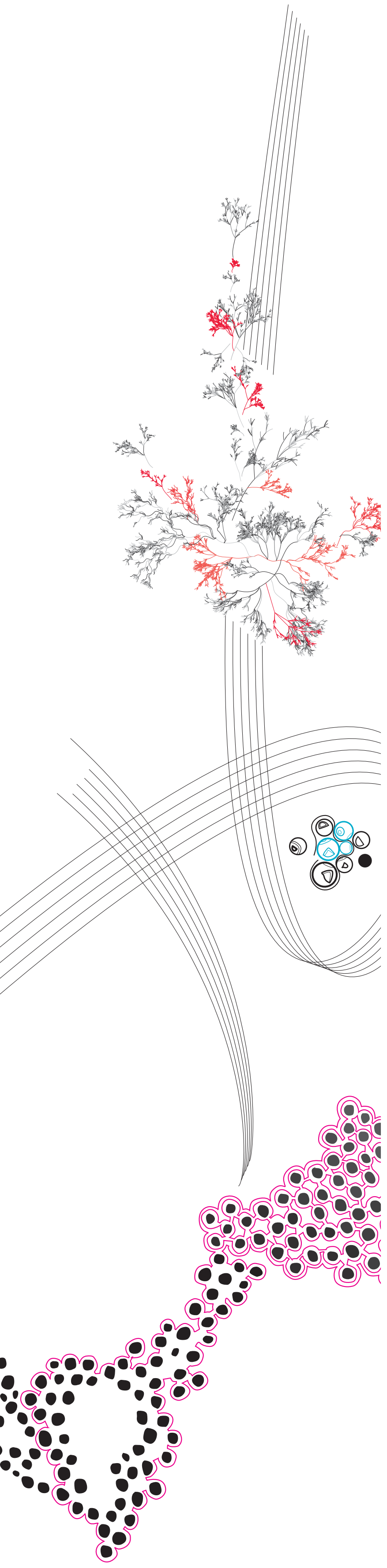
MSc Computer Science
Final Project



Enhancing Sequential Visual Place Recognition With Foundational Vision Model and Spatio-Temporal Feature mixing

Lucian Chirca

Supervisor: dr. N. Strisciuglio, dr. E. Talavera Martínez

External supervisor: dr.ing. Y. Huang

August, 2024

Department of Computer Science
Faculty of Electrical Engineering,
Mathematics and Computer Science,
University of Twente

UNIVERSITY OF TWENTE.

Contents

1 Introduction **4**

2 Related work **5**
- 2.1 Image to image visual place recognition . . . 5
- 2.2 Sequence to sequence methods . . . 6

3 Contributions **7**

4 Research questions **7**

5 Methodology **7**
- 5.1 Architecture . . . 7
  - 5.1.1 Step 1: Feature extraction . . . 7
  - 5.1.2 Step 2: Feature aggregation . . . 8
- 5.2 Experimental setup . . . 10
- 5.3 Dataset . . . 10
  - 5.3.1 Configuration 1: Same dataset + cross dataset testing . . . 10
  - 5.3.2 Configuration 2: Same dataset testing . . . 11
  - 5.3.3 Mapillary Street Level Sequences . . . 11
  - 5.3.4 MSLS cities . . . 11
  - 5.3.5 Oxford RobotCar . . . 11
  - 5.3.6 Nordland . . . 12
- 5.4 Evaluation criteria . . . 12
- 5.5 Implementation details . . . 13
  - 5.5.1 Training . . . 13
  - 5.5.2 Hyperparameters . . . 14

6 Results **15**
- 6.1 Effect of DINOv2 feature extractor fine-tuning depth . . . 15
- 6.2 Effect of MLP mixer aggregator depth . . . 15
- 6.3 Effect of sequence length . . . 15
- 6.4 Comparison with state of the art . . . 16
- 6.5 Choice of aggregation layer . . . 16

7 Discussion **18**
- 7.1 Research questions . . . 18
- 7.2 Limitations and future work . . . 18
  - 7.2.1 Model convergence . . . 18
  - 7.2.2 Patch-level descriptors . . . 18
  - 7.2.3 Other foundation vision models . . . 18
- 7.3 Addressing main challenges in VPR . . . 19

8 Conclusion **19**

A Appendix A: Code **20**

Abstract

Research into Visual Place Recognition (VPR) is useful in fields such as robotics, where a robot needs to navigate an environment and assess if it has seen a place before. To do this, we can use a sequence of image frames to be able to better assess the location where they were taken. However, research in this direction suffers from a lack of training data, poor performance when trained on small datasets, poor cross-dataset performance and limited research into the use of temporal information. Recently, research into image to image visual place recognition has shown convincingly the ability of foundation vision models to have high same-dataset and cross-dataset performance, even when fine-tuned with limited data. This work focuses on bringing the benefits of foundation vision models from the field of image to image VPR to the field of sequence to sequence VPR and expands on the existing methods of generating spatio-temporal image sequence descriptors and shows that our methods outperform all previous state of the art methods in sequence to sequence VPR on 3 major datasets: MSLS, Nordland and Oxford RobotCar. We expand the MLP Mixer model architecture to generate spatio-temporal descriptors and we perform novel research into the use of the DINOv2 foundation model as a backbone in visual place recognition for the goal of matching image sequences. Consequently, we show that our model performs well with limited training data, we show the highest same-dataset and cross-dataset performance in all our experiments, compared to previous state of the art in sequence to sequence VPR, thus solving the problems of the current state of the art. As a result of this research, simultaneous localization and mapping (SLAM) systems [10], such as robot navigation, will be able to better navigate their environment since our models output the correct result more often than previous methods.

1 Introduction

Visual place recognition (VPR) is the task of matching an image or a sequence of images of a place with a different image or sequence of images of the same place, taken at a different time, based on a set of discriminative visual cues. Research in this field plays a pivotal role in robot navigation, augmented reality, indoor localization, and any localization task where geo-location via satellites (e.g. GPS) is not possible.

Research in this area generally tries to generate a vector of features (descriptor) from images to be used for image retrieval. The goal is to have a descriptor that is as compact as possible and captures as much discriminative visual information from the images as possible. After that, we try to find in our database of image descriptors the vector that is closest to our query image descriptor. This is typically done with a nearest neighbor search.

The main challenges in VPR are

- **Variability in environmental conditions.** Pictures taken in outdoor environments can vary significantly due to changes in lighting conditions (day vs. night, shadows, weather conditions), seasonal changes (foliage, snow cover), and occlusions (obstacles, moving objects). See Figure 4.
- **Viewpoint and Scale Changes.** Images of the same place can look very different when captured from different viewpoints or scales. Recognizing a place despite these variations requires the algorithm to be invariant or robust to such transformations. See Figure 1.
- **Appearance Changes Over Time.** Places can undergo significant changes over time due to factors such as construction, renovation, or natural alterations. Maintaining recognition accuracy over long periods requires the ability to handle temporal variations.

To overcome these challenges, a VPR system should be able to only pay attention to visual elements which are relevant for scene recognition and unlikely to change over time, such as landmarks, structural patterns (in architecture, urban or natural layouts) and texture and material (like the use of bricks, concrete, wood, etc.). This is the reason why visual transformers (ViTs) [19, 20, 43, 22, 29, 46] have shown significant performance recently in visual place recognition, due to their weaker inductive bias, which allows them to understand more complex relationships in the data than traditional CNN architectures [9]. One particular example is the use of pre-trained vision foundation models, such as the DINOv2 [33] model [22, 19, 20], with or without finetuning, which has several notable benefits.

The DINOv2 model was pre-trained on a large dataset (142M images), which means that it learned to identify general features like lines, edges, contours and shapes, as well as the attention maps to identify objects. Using it as a feature extractor helps us leverage the information it has gathered without having to train our own system on such a large dataset. Consequently, it is likely to perform better at feature extraction than most models trained only on VPR datasets, due to the amount of data and computational resources that went into the training phase and it is also likely that it generalizes better to unseen VPR datasets.

Furthermore, the DINOv2 model has been trained using knowledge distillation to transfer knowledge from a more complex teacher model, which means that it is also efficient in terms of model size and thus computational cost.

One could also consider other foundation vision models, such as CLIP [36], however, this work focuses on the DINOv2 model, since it is the most promising model holding the current state of the art results in image to image VPR. Therefore, we can provide a more detailed view of its performance compared to if multiple foundation models are investigated.

Traditionally, approaches to visual place recognition have often relied on static image descriptors [5, 6, 40, 43] which may fall short in capturing the dynamic nature of real world environments, such as changing lighting conditions and viewpoint variation. Modeling the temporal relationship between image features in a visual sequence may help extract more information about how a scene evolves and lead to a more robust system and lessen the impact of lighting variation, occlusion and temporary visual components such as pedestrians, cars, etc.

To address this, research has been done into sequence to sequence (seq2seq) VPR [7, 30, 16, 15, 31], where the goal is to match a given query image sequence to another image sequence of the same place, somewhere in the database. Therefore, in this work, we primarily focus on sequence to sequence VPR.

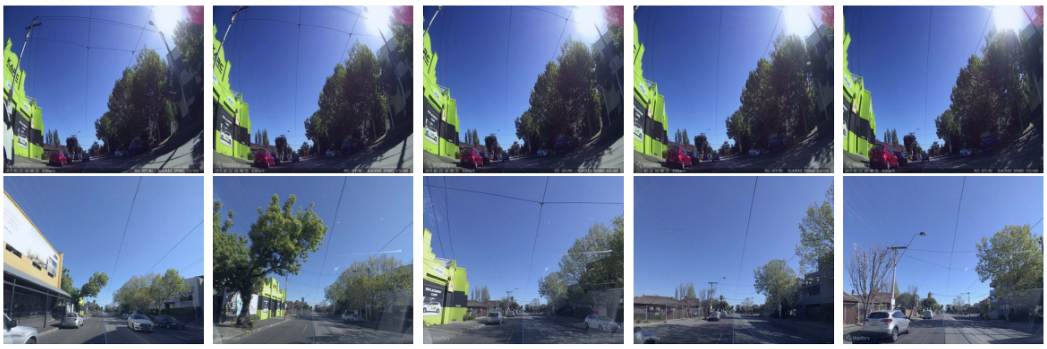


FIGURE 1: A query and positive pair showcasing viewpoint variation between sequences taken in the same place. A VPR system should be robust to these types of changes.

2 Related work

2.1 Image to image visual place recognition

The task of visual place recognition has traditionally been approached as an image to image (im2im) retrieval problem, where the goal is to find the nearest database image given a query image. This is done by extracting visual information from an image and encoding that data into a vector that represents that image (image descriptor) and which can be used for retrieval of other visually similar images. In the retrieval step, given a query image descriptor, nearest-neighbor search is performed on the database of image descriptors and the top closest matches are found.

Popular examples of image to image visual place recognition include NetVLAD [2], which uses a convolutional neural network (CNN) to extract image features and a trainable generalized VLAD [21] layer to generate the final image descriptor, CosPlace [5] which treats the training phase as a classification problem, thus doing away with expensive pair mining, AnyLoc which showed the ability of foundation vision models to have high performance across VPR datasets with varied environments (aquatic, aerial, urban, etc.), even without fine-tuning [22], the paper [24], which introduced Generalized Contrastive Loss (GCL) to leverage the amount of visual overlap between images during training, and finally, [25], which casts place recognition as a regression problem etc.

Recent improvements in image to image place recognition have demonstrated significant results on datasets such as MSLS, for example, EffoVPR [42] managing to achieve a recall at 1 of 92.8 (on the MSLS val test) and establishing a new state of the art by leveraging the use of foundation vision models. Similar works include DinoSALAD [20], which developed a novel way to aggregate visual information, using concepts from the mathematical field of optimal transport, using the DINOv2 foundation model [33] as a backbone, SelaVPR [26], which introduced a new method of fine-tuning foundation models based on sequential and parallel adapters added to each transformer block, DinoMix [19], which first used the MLP mixer model to generate frame-level descriptors from the output of the DINOv2 foundation model and ProGeo [28], which demonstrated how to effectively leverage the CLIP [36] multi-modal foundation model, which not only achieves competitive results on major benchmarks such as Pittsburgh-30k-test [41], but also exploits textual descriptions of places, which provides added explainability to the model's performance, since it makes it easier to ascertain if the model correctly learned to describe the visual information in the image.

Despite these impressive results, little research has been done into the use of pre-trained foundation vision models specifically for sequence to sequence VPR, with most methods in this subfield relying on the use of Convolutional neural networks (CNNs), VGG [38], ResNet-50 [18], etc. These backbone networks have shown inferior performance to pre-trained foundational models in image to image VPR. Besides performance reasons, previous sequence to sequence VPR methods suffer from limited training data. The paper JIST [7], tries to solve this problem by training their feature extractor also on image to image VPR datasets, however training then takes longer than just training on sequence to sequence datasets, since in this case the training has to be done on multiple datasets (image to image and sequence to sequence) and requires more data and resources than normal. Foundation vision models solve this problem, since they have been pre-trained on a large dataset of images. Therefore, the aim of this work is to solve these problems and take the first steps towards the use of pre-trained foundation vision models, for the task of sequence to sequence VPR. We chose to focus on DINOv2 specifically, which is single-modal and thus easier to train, requiring only image data.

Additionally, another focus will be to address one major drawback of traditional image to image VPR, which is that it doesn't make use of temporal information present in general VPR tasks, such as when a car is driving down a road, with a camera than can record image sequences (videos), which could provide useful information for retrieval. Additionally, matching single images is sensitive to noise, because perturbations in the image, such as occlusion, lens flare, etc. can lead to the wrong prediction being made if too much of the visual information in the query image is occluded. Consequently, this work focuses on sequence to sequence (seq2seq) VPR to address these limitaions.

2.2 Sequence to sequence methods

Sequence to sequence VPR generally splits into two approaches: **sequence matching** and using **sequence descriptors** [7]. The first approach takes two image sequences, and compares each image in the first sequence with the corresponding image in the second sequence, to create a similarity matrix. If there is a high degree of similarity, the sequences are considered to be from the same place. This is usually done under assumptions such as constant velocity [29]. One of the first examples of this is SeqSLAM [31]. In recent literature, SeqMatchNet [16] has addressed the issue of previous works relying on trained image descriptors without considering the later score aggregation. However, the sequence matching approach still has intrinsic drawbacks, including the issue of potential false positives due to erroneous single-image descriptor matches and the fact that the matching time increases linearly with the sequence length [29].

For these reasons, the aim of this work is focus on the second type of seq2seq VPR, which is that of aggregating frame-level descriptors into a sequence-level descriptors, with an aggregation layer, and using that sequence descriptor for matching. The main benefits of this method are i) allowing us to integrate temporal information (i.e. learning how spatial features evolve over time) and ii) using an aggregation layer which could compress the information from the individual images into a compact sequence descriptor (implementation described in the methodology).

In this direction, notable examples include [12], which first introduced this idea, using three methods: fusion of the frame-level features with a fully connected layer, concatenation of single image descriptors and integration over time of the single image features via an LSTM network. Further research has explored applying non-learned discrete convolutions on single frame descriptors in the work of Garg et al. [14]. Recently, SeqNet [15] explored learned descriptors using a 1-D temporal convolution as a learned pooling of frame level features into a sequence descriptor, however it uses a two-step approach, commonly seen in image to image VPR [43, 46], which can be computationally expensive.

Improving on the mentioned methods, the work of [45] leverages advancements in video action recognition based on spatio-temporal fusion to provide a novel approach in VPR that combines spatial and temporal information using transformers. However, it is computationally expensive since it relies on one backbone network (a CNN) and two separate vision transformers: one for the temporal and one for the spatial dimension. Additionally, the spatial and temporal dimension are computed separately and simply added together at the end. This may fail to capture dependencies between these two dimensions.

To address these issues, this work introduces modifications to the traditional MLP mixer to aggregate image-descriptors from a sequence of images to perform feature mixing on both the spatial and the temporal dimension. Research exists into the use of MLP mixers for image to image VPR, one notable example being DinoMix [19], which aggregates image features using the mixer model, achieving impressive results on the Pittsburgh-30k-test dataset [41]. However, that method only works for image to image VPR and no research has yet been done into the use of the MLP mixer model for sequence to sequence VPR. Novel research has, however, been done into the field of time-series forecasting, demonstrating the use of an MLP mixer [8] for mixing operations on both the time and feature dimensions to extract information. This indicates an opportunity for the use of the MLP mixer model for aggregating single frame descriptors to generate a sequence descriptor, since we are dealing with time series data and we want to mix the temporal and spatial information.

3 Contributions

This work focuses on the two main limitations of the current state of the art.

Firstly, the problems of lack of research into the use of foundation vision models for sequence to sequence visual place recognition are addressed by using the DINOv2 model as a feature extractor, and measuring its performance with different aggregation methods and comparing it to previous systems. This also solves the problems of lack of training data, poor same dataset and cross-dataset performance and long training times of having to teach a model from scratch

Secondly, the limited research into spatio-temporal descriptors is addressed by leveraging the MLP-mixer model methodology of feature mixing to extract both spatial and temporal information from image sequences.

Consequently, to address these two limitations in field of sequence to sequence visual place recognition, we put forward the following contributions:

- Researching the use of a visual foundation model (DINOv2) as a backbone network in the field of sequence to sequence visual place recognition, to extract visual information from each image in a sequence.
- Adapting the MLP mixer model [40] as seen in [40, 19, 8], to work with image-level descriptors across the time domain to produce spatio-temporal descriptors, which makes it a novel aggregation layer in the specific field of sequence to sequence visual place recognition.

4 Research questions

To find the best configuration and hyperparameters of the Mixer aggregator and DINOv2 foundation vision model, and to compare our contributions to other methods and, two main research questions are put forward.

The main metric for performance we will use to answer the research questions is explained in the evaluation criteria section.

- **RQ1:** How does using the MLP mixer aggregation layer compare with other aggregation methods namely, NetVLAD, SeqGeM, and a Fully Connected (FC) layer in terms of performance, when used with DINOv2?
  - **RQ1.1** How does the performance of the MLP mixer aggregator change when using different number of mixer blocks?
  - **RQ1.2** How does the performance of the MLP mixer aggregator change by using different sequence lengths?
- **RQ2:** How do other backbone networks such as Convolutional Neural Networks (CNNs), as seen in previous SOTA sequence to sequence VPR, compare to using the pre-trained foundation model, DINOv2, when used with the same aggregation layers?
  - **RQ2.1** What is the effect of fine-tuning the DINOv2 feature extractor on the performance of the proposed system?

5 Methodology

5.1 Architecture

Following the choice of using sequence-level descriptors, the goal of the models used in this work is to receive an image sequence of length L containing images of size $H \times W \times 3$ and to produce a sequence descriptor of length K, where K depends on the method used. This sequence descriptor is then used in k-nearest neighbor search to find the top candidates for image retrieval. The process of generating the sequence descriptor is done in two steps: **Feature extraction** and **Feature aggregation**. The overall two-step process is the same as in all the papers we compare to which generate sequence descriptors (e.g. [7, 29, 45], etc.).

5.1.1 Step 1: Feature extraction

The goal of this step is to extract visual information from each image in the sequence, obtaining one frame-level tensor per image. Internally, every image of size $H \times W \times 3$ is passed into DINOv2, which splits the image into patches of size $P \times P$, where $P = 14$. Then these patches are passed through a number of ViT blocks (outlined

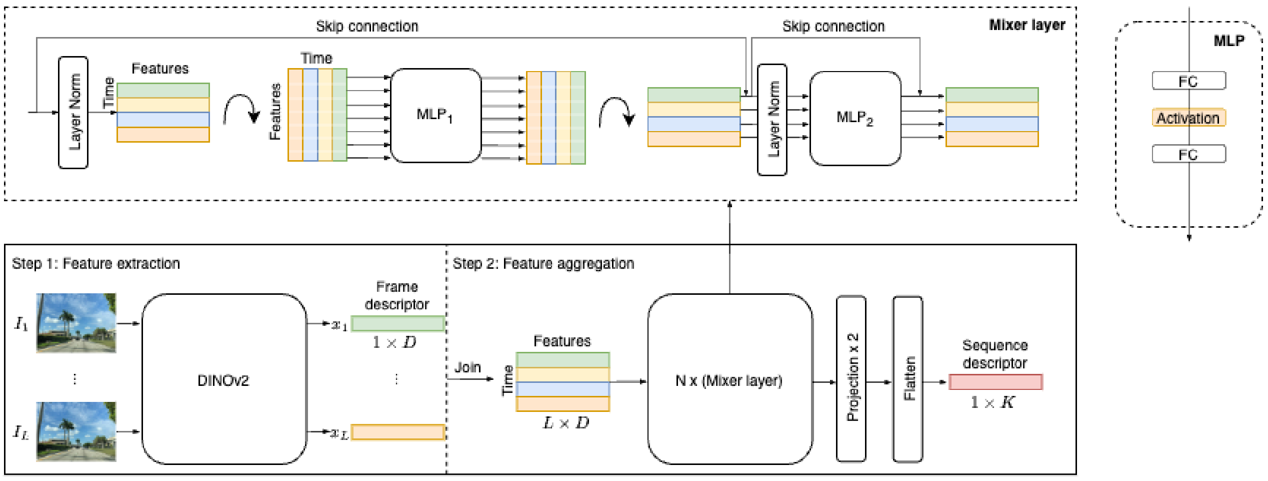


FIGURE 2: The system architecture diagram of the DINOv2 + Mixer method showing the two-step process of generating the sequence descriptor. In Step 1, each image in the input sequence is passed through the feature extractor to produce a frame descriptor. In Step 2, the frame descriptors are joined to produce a 2D array of features. The columns are the temporal features and the rows are the spatial features. The array passes through N Mixer layers then gets projected down to a compact spatio-temporal (sequence) descriptor of size K.

in Table 1) to produce a single tensor x, per image. Where $x \in R^D$ when we extract only the CLS token, when using the FC, Mixer or SeqGeM aggregator, and $x \in R^{Y \times D}$ when the aggregation layer is SeqVLAD, where Y is the number of patches in the input image and D depends on the size of the DINOv2 model according to Table 1. The consequence of this is that, in the second case, when $x \in R^{Y \times D}$, we extract more local visual information (one descriptor per patch), however, in the other case, we extract the global image-level descriptor.

Fine-tuning
Since the DINOv2 model has been trained on non-VPR-specific data, fine-tuning allows us to preserve the representational power of the DINOv2 model while improving task-specific performance. To do this, only the last encoder blocks are retrained, while the blocks up to the last ones are frozen. This is because we are interested in preserving the model weights used in recognizing low level features (e.g. lines, curves, etc.) but teach the model the high level features relevant to our task (e.g. background information, texture etc.).

5.1.2 Step 2: Feature aggregation

The aim of this step is to take the output from step one, and to generate a single descriptor that represents the visual information in the image sequence, and which can be used for retrieval.

The encoder in this work is fixed (i.e. we use DINOv2), however, we experiment with 4 aggregation layers, namely, a fully connected (FC) layer, our implementation of the MLP-mixer, adapted for seq2seq VPR, SeqVLAD [30] and SeqGeM [7].

MLP-mixer
The aggregation step when using the Mixer aggregator starts with joining the image descriptors into one matrix of size $L \times D$. This matrix is fed sequentially into N identical Mixer layers (as shown in Figure 2), since the MLP Mixer model is isotropic. The Mixer layers are composed of 2 MLP blocks: one responsible for spatial features and one for temporal features, same as in [8]. The consequence of this arrangement is that the model learns how to mix the features of the temporal and spatial dimensions, instead of features from these dimensions simply being added together as in [45]. Inside these MLP blocks, there are two fully connected layers and an activation layer (see Figure 2), as in [19]. These MLP blocks use weight sharing so they are efficient in terms of model size. The activation layer, which in this cases uses the ReLU function, provides non-linearity, which allows the model to learn non-linear functions. We chose ReLU [13] for its simplicity, computational efficiency and since in deep networks it deals with the vanishing gradient problem [17].

After that, the $L \times D$ matrix needs to be projected to a fixed dimension, using 2 projections layers (one for each dimension). The reason for choosing two projections instead of one is because it significantly reduces the number of parameters needed, thus reducing the risk of overfitting.
Lastly, the output is flattened to obtain a K dimensional vector, where $K = M \times N$, M is the dimension of the projection along the vertical dimension (formerly the temporal dimension) and N is the dimension of the projection along the horizontal dimension (formerly the spatial dimension). For this work, we set $M = 8$ (for sequence length 5) and $N = 512$ which results in $K = 4096$. It's worth noting that the value of M is adjusted dynamically based on sequence length, to result in a fixed value for K, same as in [19]. With a backbone output dimension of 768 and higher (for DINOv2 size B and higher, see Table 1), the projection happens from a higher dimension to a lower dimension (N). The result of this is that information gets compressed, so the model can generalize better. The value of N is also not significantly different from D, so we don't lose too much information in the process. If we increase the backbone output dimension, we would also need to increase the value of N, so we don't lose significant resolution on the encoder descriptors.

The value of K has been set to 4096, since it is a popular output dimension [19, 7, 29], and it makes results easier to compare. The consequence of having a fixed output dimension is that it makes it possible to easily compare sequence descriptors extracted with different sequence lengths.

To aggregate image-level descriptors into a single sequence descriptor, we chose to look into the MLP Mixer model since it is able to model both the relationships between the features inside the frame, as well as the evolution of a single feature over time, possibly extracting dependencies and long-range interactions within the image and encoding discriminative information. Furthermore, it takes as input a 2d array and it fits naturally with the output of the feature extractor. Lastly, the mixer model is efficient in the number of parameters, since it uses weight sharing.

It's worth noting, however, that it requires image sequences to be of fixed length, since the fully connected blocks inside of it that handle the temporal information need to have the same number of neurons as the sequence length. This can be seen in Figure 2, under MLP$_1$ and is a consequence of the way the MLP mixer handles input values. Therefore, the number of parameters grows linearly with the image sequence length, requiring the model to be retrained on for each different sequence length.

Compared with traditional methods used in sequence to sequence VPR, such as NetVLAD [2] or SeqGeM [7], it has the advantages of being able to model more complex relationships of the image features as well as having an inherent ability to use the temporal information present in image sequences.

Other aggregators

In this work, other aggregators are also being used. The simplest one of those is a fully connected (FC) layer that takes the L image descriptors of length D from Step 1 and flattens them to a descriptor of dimension $L \times D$ and is fed through a multilayer perceptron with output dimension K, where $K = 4096$. No activation function is used here, which makes it a simple linear transformation. Its purpose is to serve as a baseline to compare other aggregators against, which is the reason we chose to use it.

The next aggregator is SeqVLAD which has shown state of the art results recently [29]. It's input is an array of patch-level descriptors of dimension D. Its output is a sequence descriptor of dimension $D \times C$, where C is the number of output clusters, which in this case is 64. It is based on VLAD [21], which aggregates local features into a global image descriptor, and has shown remarkable performance over the years in VPR and still remains performant, hence why we also use it in this work, and because it is the only aggregator, from the ones we experiment with, that aggregates patch-level descriptors.

Lastly, this work also uses the SeqGeM aggregator, proposed in [7], which is a pooling layer that takes as input a matrix of size $L \times D$ (same as in the Mixer case) and outputs a descriptor of dimension D. It has shown state-of-the art results recently [7], hence why it is used here.

Model	Descriptor size	Num. blocks	Num. parameters
S	384	12	21M
B	768	12	86M
L	1024	24	300M
g	1536	40	1100M

TABLE 1: The different sizes of DINOv2 models.

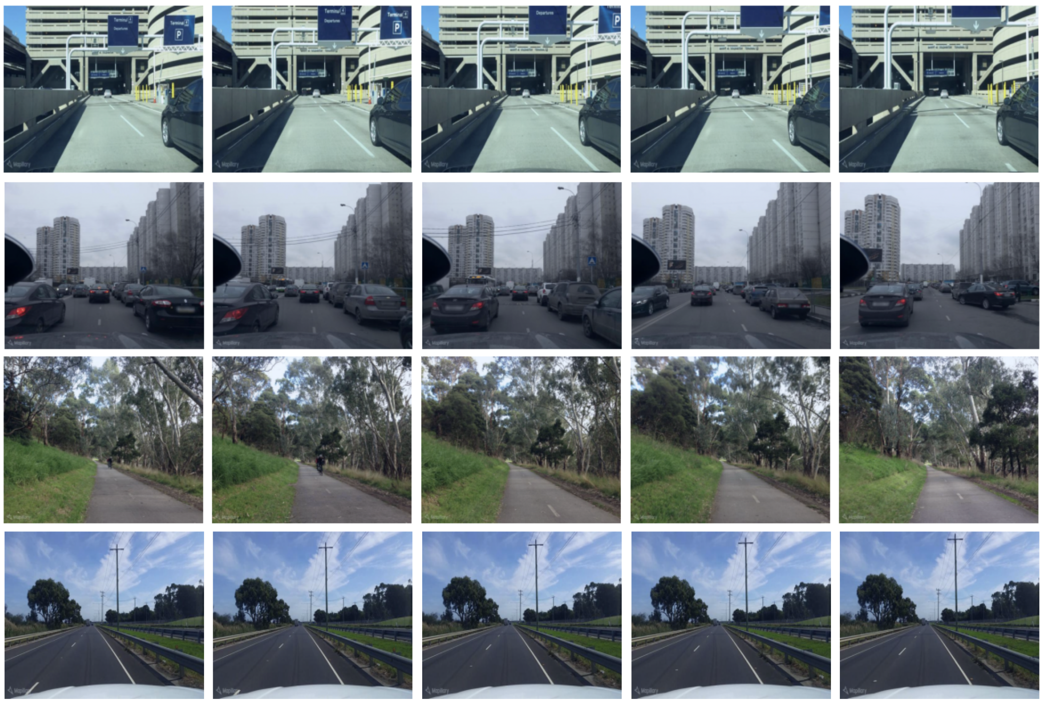


FIGURE 3: MSLS dataset sequence examples. This collection highlights the variation in locations and environments present in the dataset.

5.2 Experimental setup

5.3 Dataset

For this work, 3 datasets containing image sequences are used: MSLS, Oxford RobotCar and Nordland. Additionally, a subset of MSLS, which we will call MSLS cities in this work, uses image sequences from the city Melbourne for training and sequences from the cities Amman, Boston, San Francisco, Copenhagen for testing. We created this subset to comply with the training and testing conditions of [45] in order to be able to compare the performance of our methods to the methods tested in that paper.

There are a few configurations that we will use in our experiments, namely

5.3.1 Configuration 1: Same dataset + cross dataset testing

This is the configuration shown in Tables 6, 4 and 3. The goal of this testing configuration is to measure the performance of a given method trained on the MSLS dataset, since it contains the largest variety of environments, and tested on MSLS, Nordland and Oxford RobotCar, to measure the same-dataset and cross-dataset performances. This likely shows how good a model is at learning general features of image sequences from a variety of environments and weather conditions and how well it performs in environments it has not been trained with. This configuration is similar to how the paper [29] does it, however, we additionally test on Nordland.

In the ablation studies, comparing different configurations of the DINOv2 + Mixer method (Tables 4, 3), we resize the images to 224x224 to speed up training, however, for the other experiments, we resize to 392x392, so we can effectively compare our methods to previously researched methods.

Additionally, for the ablation on different input sequence lengths (Figure 6), we resize input images to 224x224, to speed up training and to avoid out-of-memory errors during training.

- Training: MSLS train set
- Validation: MSLS validation set
- Testing: MSLS test set, Nordland test set, Oxford RobotCar test set

5.3.2 Configuration 2: Same dataset testing

In this configuration, we train and test on the same dataset (Nordland, Oxford RobotCar and MSLS cities), using the validation set of the respective datasets. The Table 5 shows the results of this configuration. This way of testing allows us to directly compare our with the paper [45].

We resize input images to 392x392 in this configuration, so the results are comparable with methods that use images of dimensions 384x384.

5.3.3 Mapillary Street Level Sequences

MSLS [44] is a large dataset containing street-level dashcam footage from 30 major cities across six continents. The 1.6 million images in this dataset contain metadata related to sequence information, GPS coordinates and compass angles. Furthermore, the images capture times spanning all seasons over a nine year period and contain different weather, cameras, daylight conditions and structural settings.

In Figure 3, we can see that the MSLS datasets captures a breadth of environments, weather and lighting conditions from 6 continents. This should help the model generalize better and learn features which are shared across environments.

The dataset is split into three subsets, containing the following cities (same as in [29]):

- Testing: Copenaghen and San Francisco (former validation set)
- Validation: Amsterdam, Manila
- Train: The remaining cities

Since the MSLS dataset doesn't contain ground truth data for the images in their test set (82 299 images), the original test set was excluded from the resulting dataset, and instead the former validation set was used for testing, following the procedure of [29].

Detailed information about the dataset including weather, lighting variation, occlusion, seasons, coverage, etc. can be found in the original paper for the dataset [44].

5.3.4 MSLS cities

In order to be able to compare our methods under the same conditions as those in [45, 15], we also perform experiments while training on Melbourne, a subset of MSLS, and test on the cities Amman, Boston, San Francisco and Copenhagen. The validation set for these experiments is the same as the one we use with the full MSLS set, since it's from the same dataset. We also truncate the training set to contain around 5000 images, to comply with the methods we are comparing with. This has the consequence of showing which models perform well even with few training samples.

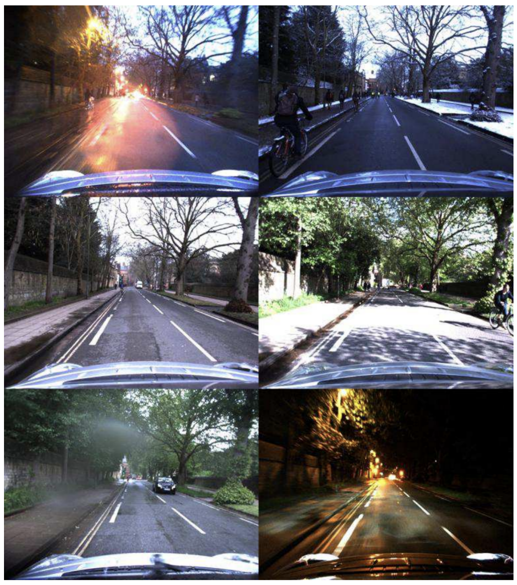


FIGURE 4: Oxford Robotcar dataset images. Here we can see that the same place is shown over a longer period of time, featuring large variations in lighting and weather.

5.3.5 Oxford RobotCar

This dataset [27] consists of images taken by an autonomous car around central Oxford. It contains

Table 2: Number of sequences contained in the datasets.

Dataset Name	Image Resolution	Seq. Length	# db/queries # train	# db/queries # validation	# db/queries # test
MSLS	574×480	5	733k / 393k	8.1k / 5.8k	13.6k / 8k
		10	568k / 309k	5.4k / 4.2k	8.5k / 4.6k
		15	478k / 259k	4k / 3.3k	6.1k / 3.3k
Melbourne	574×480	5	5.1k / 4.1k	- / -	- / -
Amman	574×480	5	- / -	- / -	0.7k / 0.6k
Boston	574×480	5	- / -	- / -	11.1k / 5.4k
San Francisco	574×480	5	- / -	- / -	4.7k / 3.6k
Copenhagen	574×480	5	- / -	- / -	8.9k / 4.5k
Nordland	640×360	5	15k / 15k	3k / 3k	3k / 3k
Oxford RobotCar	1280×960	5	3.6k / 3.3k	3.9k / 3.7k	3.6k / 3.9k

around 20 million images collected over the period of 1 year and a total distance traversed of 1000km by 6 cameras mounted to the vehicle, along with GPS ground truth. The images were recorded over the period of one year and capture different combinations of traffic, weather and road conditions.

The dataset contains images from May 2014 until December 2015 therefore it contains images from all four seasons of a 10km rides of the same route in central Oxford, resulting in more that 100 traversals. In the original paper [27], we can find the breakdown of weather and other conditions that are present in the entire dataset.

In Figure 4, we observe that the Oxford Robotcar dataset focuses on capturing the same route over an extended period of time, allowing us to see the same place in many different weather and lighting conditions. This can help our model better learn to be invariant to weather and lighting conditions.

To be consistent with [29, 45], we split the dataset in the following 3 subsets

- Train set:
  queries: lap 2014-12-17-18-18-43 (winter night, rain);
  database: lap 2014-12-16-09-14-09 (winter day, sun);
- Validation set:
  queries: lap 2015-02-03-08-45-10 (winter day, snow);
  database: lap 2015-11-13-10-28-08 (fall day, overcast).
- Test set: queries: lap 2014-12-16-18-44-24 (winter night);
  database: lap 2014-11-18-13-20-12 (fall day).

The dataset is pre-processed to select only frames that are at least 2m apart, same as in [45]. The number of resulting sequences is shown in Table 2.

5.3.6 Nordland

A dataset containing 40 hours of full-HD video of a rail journey over all four seasons [39], with various combinations of weather and lighting conditions. It is a challenging dataset, because it contains significant seasonal variation, since its composed of images of the same place in all 4 seasons (see Figure 5), as well as generally unstructured environments. The advantage of this dataset is that it is able to show that the model is able to perform well in conditions of seasonal variation in unstructured environments, since the previous datasets capture mostly structured environments (urban areas)

In the dataset we are using, the image frames are approximately 20 meters apart [16].

Following [45, 16], we use Summer/Winter for training and Spring/Fall for testing.

Although weather variation exists in the dataset, the original dataset paper [39] does not provide this information.

5.4 Evaluation criteria

A sequence is said to be correctly classified if any of its images are within 25 meters of any other image of the predicted sequence. During training, this distance is set to 10 meters. These values were

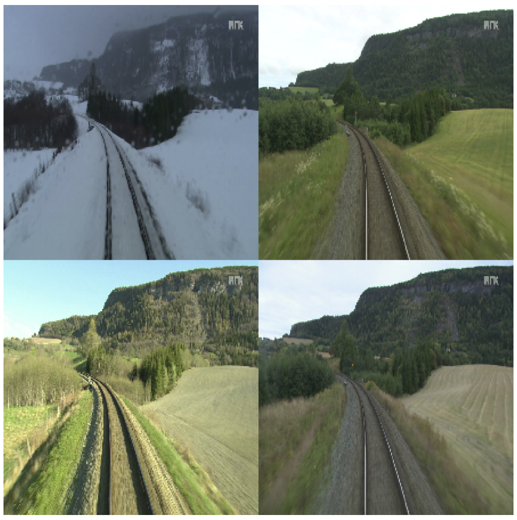


FIGURE 5: Nordland dataset images. The same location is shown during each of the four seasons.

TABLE 3: Ablation on DINOv2 fine-tuning for sequence length 5, mixer depth 4, descriptor dimension 4096, image size 224x224 trained on MSLS with patience 5. Bold are the highest recalls for that column, underlined are the second best.

#Finetuned DINOv2 blocks	R@1 (MSLS)	R@1 (Oxford)	R@1 (Nordland)
Model size S			
0	61.7	36.0	58.0
2	$\underline{86.5}$	77.0	85.3
4	86.0	77.2	80.6
6	80.8	59.4	68.1
8	77.4	45.9	72.7
10	74.1	14.8	73.5
12 (all)	72.2	23.0	61.1
Model size B			
0	64.9	54.4	63.6
2	**86.7**	$\underline{85.4}$	**88.2**
4	86.4	**86.5**	$\underline{85.6}$
6	82.5	58.2	60.3
8	78.8	61.2	76.5
10	75.2	41.4	76.8
12 (all)	75.1	41.5	64.4

chosen to be in line with other papers are using [29]. The consequence of these values is that, during training, we force as much distance as possible between positive and negative samples. We consider the GPS information from our datasets as ground truth and don't account for GPS errors, although there might be some since expect these errors to be in the minority of training and testing cases.

To measure the performance of the system, we use the recall at k, since it is a popular evaluation metric and allows us to more easily compare with other methods. It measures the proportion of query images for which the correct database image was found in the top k candidates for retrieval. A high recall at 1, for example, means that the image sequence that the model retrieves as the closest match is correct most of the time. This measure is useful, since we are dealing with an image (sequence) retrieval problem, and thus it shows how often we retrieve the right image sequence.

We chose to display the recall values in all our figures and tables as the number of percent recall (e.g. 80 means 80% recall). This way of showing recall was chosen, since it can also be seen in other papers such as [29, 7] and it is easier to read. For example a recall of 60 means that, on average, for 60 query images out of 100, the correct match was found in the top k results.

5.5 Implementation details

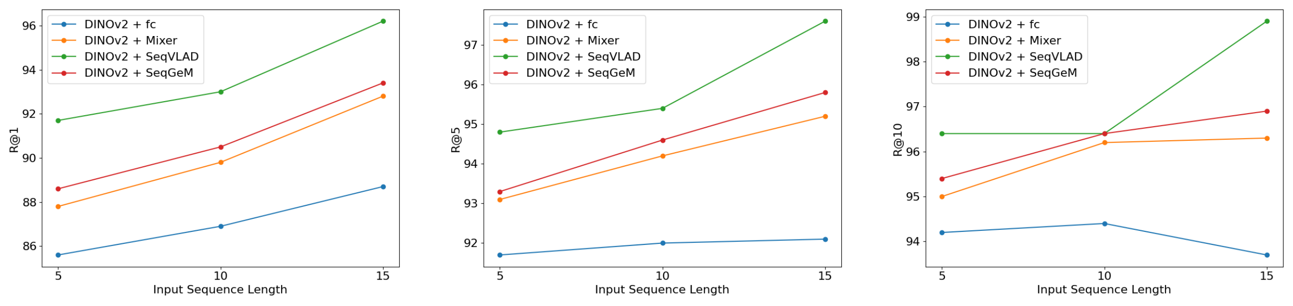
5.5.1 Training

For training, triplet margin loss (first introduced in FaceNet [37]) is used (see Equation 5.5.1). It is a popular loss function in the field of visual place recognition, which makes it easier to compare with other works. The fundamental idea behind triplet loss is to learn a feature space where similar instances are closer together and dissimilar instances are farther apart, based on the margin α, which separates the negative and positive pairs. For the margin α, we chose a value of 0.1, like in [29, 45], in order to comply with related research methodology and train under the same conditions. Setting a higher margin enforces a larger distance between anchor-positive pair and anchor-negative pair in descriptor space, which is usually a good thing, however if the value of the margin is too high, it might make the model struggle to satisfy the loss condition for all triplets and risking underfitting.

$$\mathrm{L(A, N, P)} = \max(\|f(A) - f(P)\|_2 - \|f(A) - f(N)\|_2 + \alpha, 0)$$

where:
A, N, P = Anchor, Negative and Positive inputs
f = model
α = margin

Regarding optimizer, ADAM [23], an extension to traditional stochastic gradient descent, was used, since it computes adaptive learning rates, which is



(A) Recall at 1. (B) Recall at 5. (C) Recall at 10.

FIGURE 6: Ablation study on the impact on recall of using different sequence lengths as input on our methods. This figure shows how well DINOv2 performs for different sequence lengths with the 4 aggregators used in this work.

well suited for the problem we are trying to solve, since we are dealing with large models and relatively large quantities of data. The learning rate was set to 0.00001 (same as [7]) and the weight decay to 0. Consequently, the chosen learning rate makes sure the training is stable, since having a learning rate too high can prevent the model from converging, since it could forget previously learned features, in this case. The weight decay set to 0 has the consequence of potentially preserving useful features that are captured by higher weights, although in some cases a weight decay of 0 runs the risk of overfitting.

To find hard negative training sample we use a hard mining strategy, where we cache 1000 randomly chosen negative samples to avoid caching the entire database. This cache is recomputed every 1000 triples. During one epoch, we go through 5000 queries. [29]. These triplets could be split into multiple training batches, however, to reduce GPU resource requirements, we had the batch size set to 1.

Lastly, the training will be done under a patience of 5, meaning that models will stop training if their recall at 5 did not improve in the last 5 epochs. This is done so the training conditions are the same as in [29]. The consequence of this is that we prevent overfitting if our model starts to fit the training data too closely. This ensures the model is general enough to capture relevant features in the test and validation dataset, and since we are also testing on different datasets, we make sure this way that the performance is also good cross-dataset. The upper limit for the number of maximum allowed epochs is 100, however even the longest training models only reached around 20 epochs. Longer than this, the model will likely overfit.

TABLE 4: Ablation on Mixer aggregator depth for sequence length 5, DINOv2 size B, with the last 2 blocks fine-tuned, descriptor dimension 4096, image size 224x224 trained on MSLS. Bold are the highest recalls for that column, underlined are the second best.

#Mixer layers	R@1 (MSLS)	R@1 (Oxford)	R@1 (Nordland)
1	86.3	$\underline{84.0}$	**90.6**
2	**87.8**	78.0	$\underline{89.5}$
4	$\underline{86.7}$	**85.4**	88.2
6	84.4	69.3	83.0
8	86.1	72.0	79.6
10	85.6	74.6	81.1

5.5.2 Hyperparameters

The shape of the input images for determining the best DINOv2 + Mixer configuration and for the ablation on input sequence length is 224x224, to speed up training time. However, during experiments comparing with other methods, the image size is 392x392 so as to be comparable with other methods, which use use an image size of 384x384 and to be divisible by 14 for DINOv2 to work. Each image is normalized with a mean of 0.485, 0.456, 0.406 and standard deviation of 0.229, 0.224, 0.225, for each channel, respectively, consistent with the implementation of [29]. This is to ensure consistency within the training and evaluation data.

Regarding the model architecture, the final (sequence) descriptor size is fixed to 4096. This is kept small to encourage compact sequence representations, speed up matching time and reduce number of parameters to reduce the risk of overfitting. The feature extractor uses a patch size of 14x14, with a stride of 14, which are the default parameters of the DINOv2 model.

6 Results

In this section we investigate the results of our ablation studies and experiments to measure the impact of our contributions and to be able to answer the research questions. Firstly, we measure the impact of various hyper-parameters, such as model size, input sequence length, etc. After that, we compare the methods we investigate in this work with previous state of the art. While we do that, we also try to investigate any patterns that emerge out of the results and possible reasons for them.

6.1 Effect of DINOv2 feature extractor fine-tuning depth

Table 3 shows the performance of increasing the number of fine-tuned DINOv2 blocks under the same conditions, for two different sizes of models. The sizes S and B (see Table 1) were chosen because the latter sizes (L and g) were too large and took too much time to train. From the results in the table, we can observe that size B of the model tends to have better performance across all used datasets. Additionally, it has the best overall results from the table, namely, when having only the last 2 layers fine-tuned, achieving a R@1 of 86.7. This is explained by the fact that having no fine-tuned layers doesn't train the model to be sufficiently task-specific, while fine-tuning all the blocks means that the model loses the valuable data it has been pre-trained with. Hence, after this ablation study, the model size is to B and the fine-tuning depth is set to 2. Similar papers that used the DINOv2 foundation model also note that fine-tuning brings significant performance improvement over not fine-tuning [19, 20, 42].

6.2 Effect of MLP mixer aggregator depth

To establish the optimal number of mixer blocks for descriptor aggregation, we look at Table 4, which shows a mixer depth of 2 to be optimal, achieving the highest performance on the the dataset it's been trained with, and the second highest performance when used with the Nordland dataset. It's worth noting that increasing the mixer depth after 2 shows lower performance: on the same dataset (MSLS) and cross-dataset (Oxford and Nordland). This is likely explained by over-fitting when the aggregator has too many parameters. Looking at [19], we observe that the same mixer depth was empirically chosen there as well, likely indicating that it is the correct configuration for the mixer model.

6.3 Effect of sequence length

Figure 6 shows how the recall values of our methods change with increasing input sequence length. We can observe that in all three sub-figures, except Figure 6c, for the DINOv2 + FC method, as the input sequence increases, the recall also increases. This is because more visual information can be used when generating the sequence descriptor. The model, in this case, has to learn to ignore the irrelevant information and only encode the data that is unique to the input image sequence and that can be used in the retrieval step. The backbone network, DINOv2, is also forced to learn features that are more relevant across time (static features) and ignore features that are only seen in a couple of frames (lens flare, cars, pedestrians, etc.). Since the backbone network is a visual transformer, it has to tune its attention mechanism to focus on the patches with static, discriminative visual features. The aggregator then takes these informative image-level and patch-level descriptors (depending on the aggregator) and is able to compress this information into a compact representation of the image sequence.

Looking at Figure 6a, we can observe that the recalls for all our methods increase roughly at the same rate, however, for the other two sub-figures, the method DINOv2 + SeqVLAD, has the highest positive change in recall from sequence length 5 to sequence length 15, namely an increase in recall at 5 of 2.95%, compared with 0.43%, 2.25% and 2.68% and an increase in recall at 10 of 2.59% compared with -0.53%, 1.36% and 1.57% over the methods that use the aggregators FC, Mixer and SeqGeM, respectively. This is likely explained by the fact that the SeqVLAD aggregator uses patch level-descriptors, which offer more fine-grained information of the image sequence, leading to more input data for the aggregation layer and thus, better retrieval performance.

All the aggregators we use with DINOv2 also have the advantage that their output dimension is fixed, regardless of input sequence length, meaning

TABLE 5: Quantitative results on MSLS cities, Nordland, and Oxford RobotCar. Following testing configuration 2, described in experimental setup, the MSLS subsets Amman, Boston, SF and Cph are used to test the models trained on Melbourne, while results for Nordland and Oxford are obtained by training and testing on their respective train and test datasets.

Method	Amman			Boston			SF			Cph			NordLand			Oxford		
	R@1	R@5	R@10	R@1	R@5	R@10	R@1	R@5	R@10	R@1	R@5	R@10	R@1	R@5	R@10	R@1	R@5	R@10
Our methods																		
DINOv2 + SeqVLAD	**70.6**	79.9	81.9	**76.1**	**82.9**	**86.1**	**88.0**	**91.4**	**94.8**	**91.5**	**94.3**	**95.2**	99.9	**100.0**	**100.0**	98.6	99.1	99.3
DINOv2 + Mixer	69.8	**80.8**	**84.2**	62.1	68.0	71.6	78.7	85.8	89.6	78.9	85.2	87.7	**100.0**	**100.0**	**100.0**	**98.8**	**99.5**	**99.8**
DINOv2 + SeqGeM	62.8	71.5	75.9	70.5	77.5	81.9	85.4	90.3	93.5	82.0	87.8	90.5	99.5	**100.0**	**100.0**	98.3	99.2	99.6
DINOv2 + fc	61.8	68.4	72.9	61.3	67.0	71.2	75.8	83.6	87.8	79.0	84.9	87.9	98.8	99.6	99.8	96.4	98.3	99.0
Previous methods*																		
NetVLAD [2]	18.9	25.1	27.7	17.9	23.8	26.7	28.9	39.8	45.5	40.5	53.4	59.4	37.7	54.3	61.5	-	-	-
NetVLAD+SeqMatch [15]	24.6	30.2	33.0	20.4	23.9	25.7	36.3	43.0	46.0	50.4	61.2	65.7	61.0	70.5	74.6	-	-	-
SeqNet [15]	26.9	37.6	40.8	27.4	35.4	39.0	55.6	67.1	72.8	46.2	58.1	63.7	79.7	90.5	93.0	-	-	-
SeqVLAD [29]	30.0	44.8	51.9	46.6	62.8	67.8	66.1	82.6	86.3	56.4	72.2	77.7	96.4	99.2	99.3	84.4	92.9	95.8
Spatio-Temporal-SeqVPR [45]	30.3	42.3	51.1	50.4	64.5	68.8	68.0	84.1	86.4	60.8	76.5	80.1	97.1	99.5	99.5	86.8	94.4	96.8
SeqVLAD w/ PCA	29.4	44.2	52.6	46.5	62.3	67.5	65.6	82.2	85.9	56.0	72.0	77.4	96.3	99.1	99.4	84.7	93.2	96.1
Spatio-Temporal-SeqVPR [45] w/ PCA	30.6	41.1	51.0	50.2	64.5	69.1	67.1	83.9	86.0	60.4	76.0	80.1	97.1	99.5	99.5	86.6	94.5	96.9

* Results taken from [45]

that the matching step takes the same amount of time regardless of input length. Therefore, we can conclude that it is advantageous to use the methods in this work with as high input sequence length as possible, to maximize perfromance, provided the computational restraints allow it.

6.4 Comparison with state of the art

If we look at Tables 6 and 5, we observe that all methods that use the DINOv2 vision foundation model as backbone have better performance than all previous methods on all proposed datasets.

When trained on MSLS (Table 6, we notice that our methods are able to achieve not only better results on the dataset they have been trained on, but also have significantly higher cross-dataset performance. This is likely because the backbone model is able to generalize better than previous backbone models used.

Training on Melborune (Table 5), we can observe an even higher performance improvement as compared to previous SOTA. For example, the R@1 for Amman is more than twice that of the previous methods. This is likely to do with the fact that the training dataset is much smaller than in other experiments and it allows us to leverage the pre-trained knowledge that the DINOv2 backbone comes with. From these experiments, we can see that using SeqVLAD as aggregation offers the best performance overall from our proposed methods, which is also seen when trained on MSLS training set. This is likely because it is able to use more visual information, since SeqVLAD aggregates patch-level descriptors which is more data, since the other methods only use the image-level descriptor (CLS token). However, the downside is the fact that the descriptors of SeqVLAD are much larger (e.g. 49152 for DINOv2 + SeqVLAD), compared to using SeqGeM, for example, which is the same length as the length of the CLS token of DINOv2 (768 for size B). This makes matching time slower and requires more computational resources.

Lastly, when training and testing on Nordland and Oxford (Table 5), our methods achieve near-perfect (for Nordland DINOv2 + Mixer, perfect) recall values. Although the previous SOTA also have high recalls on these datasets, they are still considerably lower than our methods, especially on the Oxford dataset. For these datasets, we notice that the Mixer model works better as an aggregator than our other methods, although not by much. It could be that the temporal information can be better leveraged in these cases since the images from these datasets are more homogeneous.

6.5 Choice of aggregation layer

Each of the 4 aggregators used in this work have their distinct advantages and disadvantages. Just using a fully connected layer (DINOv2 + FC) as an aggregator is used in this work to serve as a base-line for comparison. Its output dimension is the same as the Mixer model, and it has the same input dimension, however, although it has more parameters than the Mixer model (15.7M for FC vs 2.76M for Mixer), its performance is consistently lower. This is because the mixer model is efficient in terms of parameters and likely indicates that it

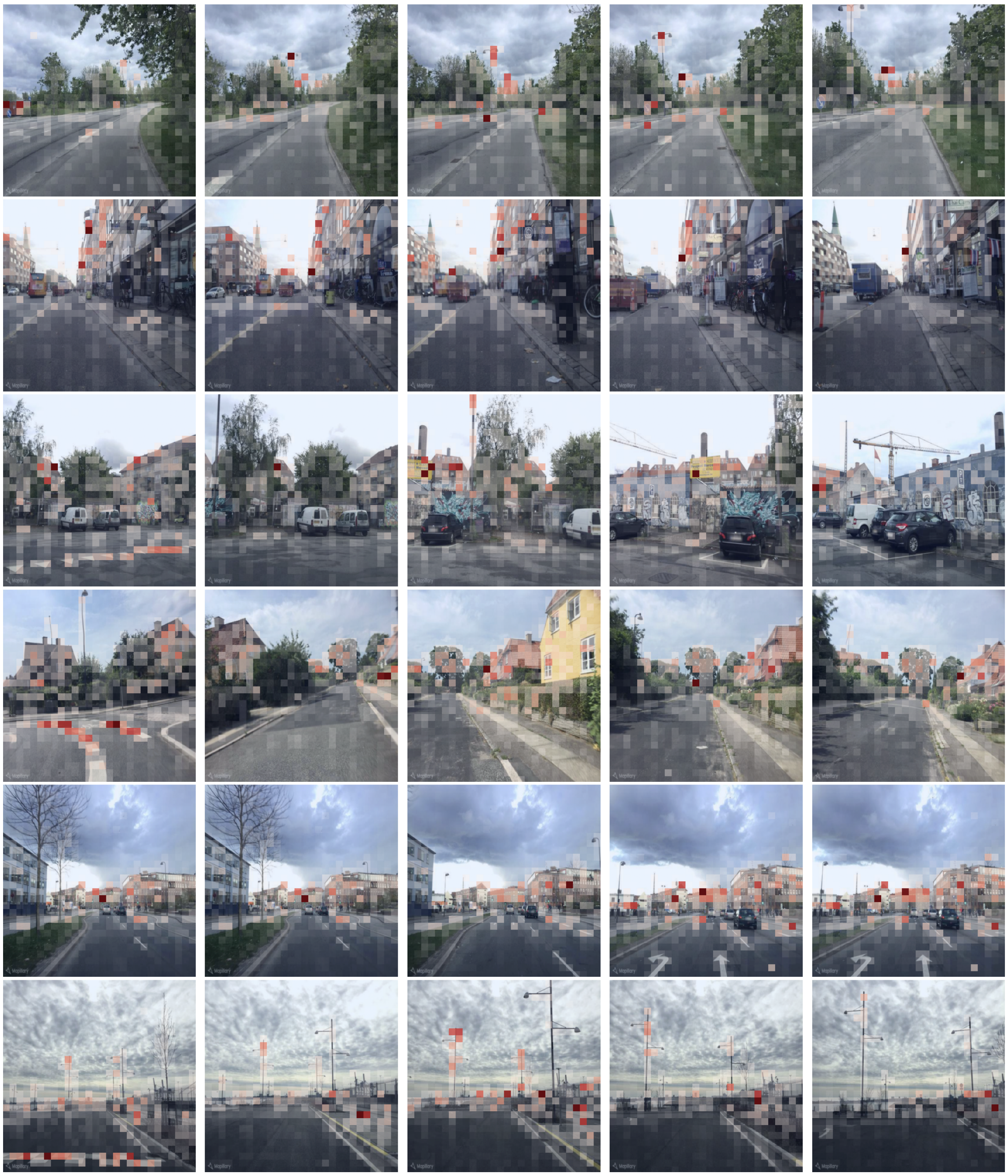


FIGURE 7: Visualization of the attention map for the last layer of the encoder using the method DINOv2 + SeqVLAD, for image sizes 392x392. The attentions for the 12 attention heads of the CLS token are averaged and the attention map is overlayed on the original image using a red hue. This figure shows that the model focuses on static visual features such as road markings and buildings and ignores movable objects like cars.

extracts more information from the image descriptors than a simple FC layer. However, when looking at performance, SeqVLAD seems to be the best aggregator in this work. Its performance is highest in Tables 6 and 5. It also seems to have the best cross-dataset performance (Table 6. However, its disadvantage is large descriptor size. When memory and matching speed are important, using DINOv2 with SeqGeM seems to be the best option. It uses the smallest descriptors and offers decent performance (second best in Table 5.

7 Discussion

7.1 Research questions

In this section we answer the proposed research questions, starting with **RQ1**.

> **RQ1** How does using the MLP mixer aggregation layer compare with other aggregation methods namely, NetVLAD, SeqGeM, and a Fully Connected (FC) layer in terms of performance, when used with DINOv2?

The MLP mixer aggregator offers better performance than the baseline aggregator (FC), with fewer parameters. However, compared to SeqGeM and SeqVLAD, its performance is lower when trained on Melbourne, but better on Nordland and Oxford datasets.

> **RQ1.1** How does the performance of the MLP mixer aggregator change when using different number of mixer blocks?

The depth of the mixer model affects the performance slightly on the same dataset it's trained on and significantly across datasets, which is likely explained by overfitting.

> **RQ1.2** How does the performance of the MLP mixer aggregator change by using different sequence lengths?

In Figure 6, we can see that increasing the input sequence length always positively impacts the performance of the MLP mixer aggregator. However, for higher values of k in R@k, we notice that the rate of positive change in recall is lower than when we measure R@1. This is likely because if the correct prediction is not in the top 1 or top 5 of results, it's also unlikely to be in the top 10 of results, since it's likely caused by the model not being sufficiently able to produce the correct match.

> **RQ2** How do other backbone networks such as Convolutional Neural Networks (CNNs), as seen in previous SOTA sequence to sequence VPR, compare to using the pre-trained foundation model, DINOv2, when used with the same aggregation layers?

We can observe that the DINOv2 foundation model outperforms all other backbone models that we compare with, using the same aggregator. For example, when using the SeqVLAD aggregator, it has better performance than the CCT384 backbone in Table 6. With the SeqGeM aggregator it's the same conclusion in the table.

> **RQ2.1** What is the effect of fine-tuning the DINOv2 feature extractor on the performance of the proposed system?

We notice in Table 3, that fine-tuning greatly affects performance, but the number of fine-tuned blocks should be kept small.

7.2 Limitations and future work

7.2.1 Model convergence

In this work we used a patience parameter of 5, so it's the same as in the papers we are comparing with, such as [29], however, some experiments have shown that our models stop training before convergence. With a patience of 5, models train usually for 6-15 epochs, however, with a patience of 10, they train up to 30 epochs. This indicates that the model can still train, but it is stopped prematurely. It might be worth exploring what is the peak performance that can be obtained with the models in this work, which is likely to push the state of the art performance even further.

7.2.2 Patch-level descriptors

This work considers only one case when patch-level descriptors are used: together with SeqVLAD aggregator. This allows it to use much more visual information than only the CLS token. In the future, it might be worth exploring other aggregators that use patch descriptors such as the SALAD aggregator in [20], since in that paper it serves as a replacement for SeqVLAD.

7.2.3 Other foundation vision models

Due to the success of the DINOv2 vision foundation model in im2im VPR ([19, 20, 26, 42, 22] and

Table 6: Quantitative results obtained by training our models on MSLS and testing on MSLS, Nordland, and Oxford RobotCar (as per configuration 1 described in the experimental setup). We compare with the current state of the art in sequence to sequence VPR that train on MSLS.

Method	Backbone	Aggregator	Image Size	Descriptor dimension	R@1 (MSLS) Train on MSLS	R@1 (Oxford)Train on MSLS	R@1 (Nordland) Train on MSLS
Our methods							
DINOv2 + SeqVLAD	DINOv2	SeqVLAD	392x392	49152	**93.0**	93.0	95.9
DINOv2 + Mixer	DINOv2	Mixer	392x392	4096	90.6	90.4	95.3
DINOv2 + SeqGeM	DINOv2	SeqGeM	392x392	768	89.9	**93.8**	**96.2**
DINO + FC	DINOv2	fc	392x392	4096	88.5	84.1	95.0
Previous methods*							
JIST* [7]	ResNet-18	SeqGeM	384x384	512	90.6	79.0	-
SeqVLAD [29]	CCT384	SeqVLAD	384x384	24576	88	75.4	-
SeqVLAD [29]	CCT224	SeqVLAD	224x224	24576	86.7	68.4	-

* Results taken from referenced paper

seq2seq VPR (in this work), it might be worth exploring other foundation models. For im2im VPR, work has already been done into using the CLIP [36] foundation model [22, 28], however it might also be worth exploring using it for seq2seq VPR, as has been done for DINOv2 in this work.

7.3 Addressing main challenges in VPR

By using the DINOv2 foundation vision model and training it in the way we mention, we achieve a system that solve the challenges mentioned in the introductory parts of this work, namely, variability in environmental conditions is handled by the fact that our models are robust to environmental conditions since they have been trained on datasets with high environmental variability (seasonal variation, different lighting conditions, movable objects, etc.).

Viewpoint and Scale Changes is handled by the fact that the DINOv2 backbone was initially trained in a self-supervised manner, since in self-supervised learning, the model is trained on tasks where the input data is transformed in various ways, and the model is tasked with predicting certain properties of these transformations. This is how it is able to be robust to viewpoint and scale changes. This reasoning also applies to the problem of appearance changes over time, since the model is able to become invariant to those changes.

The problem of movable objects is solved by training the attention map to ignore uninformative visual features, as shown in Figure 7.

The problem of occlusion/lens flare is solved by the fact that we are dealing with image sequences, so even if an image in the sequence is occluded, the model can still make the right prediction using the other images.

Lastly, we are also solving the problems of limited number of available sequence datasets, training with limited data, slow convergeance and limited cross-dataset performance by using a foundation vision model.

8 Conclusion

In conclusion, this work has addressed the current gaps in research, namely the fact that foundation vision models, although highly performant in image to image visual place recognition, have made no appearance in sequence to sequence place recognition and the fact that limited research has been done into generating spatio-temporal sequence descriptors.

The results in this work establish a new state-of-the art and show convincingly that the DINOv2 foundation model used with state of the art aggregators has significantly better performance, compared with previous methods, when tested on the same dataset it's been trained on, on datasets it's not been trained on and when trained with little data.

We also showed that the Mixer model is a parameter efficient way to obtain good performance combining spatial and temporal information, and we made the case for exploring foundation visual models further in this area of research. These results should hopefully encourage research into ways to better leverage foundation vision models and ways to better combine visual and temporal information present in image sequences which can prove useful not only in visual place recognition but in general video-retrieval tasks.

In conclusion, this work shows how our methods tackle the major problems of visual place recognition and achieves superior results, establishing a new state of the art in sequence to sequence visual place recognition due to the use of the DI-

NOv2 foundation model. This work also proposed a method to generate spatio-temporal descriptors by adapting the MLP mixer model to work with sequence data, and in some cases achieved better results than the current state of the art aggregators (see Table 5).

A Appendix A: Code

To speed up the implementation of the system, code is reused from similar work on sequence to sequence visual place recognition. The starting point is forking the public repository of [29], located at `https://github.com/vandal-vpr/vg-transformers`. It has been chosen for the following reasons:

- Contains all the code necessary for pre-processing and loading the MSLS and Oxford Robotcar datasets, as well as the logic to generate the train/test/val splits
- Contains logic for performing hard mining of anchor, positive and negative sequence triplets used for learning
- Already has all the logic for training the model, caching, evaluation, validation, saving model checkpoints and finding the best performing model
- Contains the implementation of VPR methods such as pooling layers (GeM [35], SPoC [3], NetVLAD [2], etc.) and feature extractors (Google's ViT [11], VGG16 [38], ResNet [18], TimeSFormer [4] etc.). These can be used to compare SeqDinoMix against

The models in this work are written, like the other models in the repository, using PyTorch's API [34], in python. For dependencies and virtual environment management, Anaconda [1] together with python's version manager, pip, was used. As for training the model, NVIDIA's CUDA toolkit [32] was used to take advantage of GPUs to reduce training time. The GPUs used are from UTwente's HPC cluster, where the experiments run. Regarding version management and to easily transfer the code to the cluster, Github is used.

References

[1] Anaconda software distribution, 2020. URL: `https://docs.anaconda.com/`.

[2] Relja Arandjelović, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. Netvlad: Cnn architecture for weakly supervised place recognition, 2016. `arXiv:1511.07247`.

[3] Artem Babenko and Victor Lempitsky. Aggregating deep convolutional features for image retrieval, 2015. `arXiv:1510.07493`.

[4] Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding?, 2021. `arXiv:2102.05095`.

[5] Gabriele Berton, Carlo Masone, and Barbara Caputo. Rethinking visual geo-localization for large-scale applications, 2022. `arXiv:2204.02287`.

[6] Gabriele Berton, Gabriele Trivigno, Barbara Caputo, and Carlo Masone. Eigenplaces: Training viewpoint robust models for visual place recognition, 2023. `arXiv:2308.10832`.

[7] Gabriele Berton, Gabriele Trivigno, Barbara Caputo, and Carlo Masone. Jist: Joint image and sequence training for sequential visual place recognition. *IEEE Robotics and Automation Letters*, PP:1–8, 01 2023. `doi:10.1109/LRA.2023.3339058`.

[8] Si-An Chen, Chun-Liang Li, Nate Yoder, Sercan O. Arik, and Tomas Pfister. Tsmixer: An all-mlp architecture for time series forecasting, 2023. `arXiv:2303.06053`.

[9] Luca Deininger, Bernhard Stimpel, Anil Yuce, Samaneh Abbasi-Sureshjani, Simon Schönenberger, Paolo Ocampo, Konstanty Korski, and Fabien Gaire. A comparative study between vision transformers and cnns in digital pathology, 2022. `arXiv:2206.00389`.

[10] M.W.M.G. Dissanayake, P. Newman, S. Clark, H.F. Durrant-Whyte, and M. Csorba. A solution to the simultaneous localization and map building (slam) problem. *IEEE Transactions on Robotics and Automation*, 17(3):229–241, 2001. `doi:10.1109/70.938381`.

[11] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil

Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021. arXiv:2010.11929.

[12] Jose M. Facil, Daniel Olid, Luis Montesano, and Javier Civera. Condition-invariant multi-view place recognition, 2019. arXiv:1902.09516.

[13] Kunihiko Fukushima. Visual feature extraction by a multilayered network of analog threshold elements. *IEEE Transactions on Systems Science and Cybernetics*, 5(4):322–333, 1969. doi:10.1109/TSSC.1969.300225.

[14] Sourav Garg, Ben Harwood, Gaurangi Anand, and Michael Milford. Delta descriptors: Change-based place representation for robust visual localization. *IEEE Robotics and Automation Letters*, 5(4):5120–5127, October 2020. URL: http://dx.doi.org/10.1109/LRA.2020.3005627, doi:10.1109/lra.2020.3005627.

[15] Sourav Garg and Michael Milford. Seqnet: Learning descriptors for sequence-based hierarchical place recognition, 2021. arXiv:2102.11603.

[16] Sourav Garg, Madhu Vankadari, and Michael Milford. Seqmatchnet: Contrastive learning with sequence matching for place recognition amp; relocalization. In Aleksandra Faust, David Hsu, and Gerhard Neumann, editors, *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pages 429–443. PMLR, 08–11 Nov 2022. URL: https://proceedings.mlr.press/v164/garg22a.html.

[17] Xavier Glorot, Antoine Bordes, and Y. Bengio. Deep sparse rectifier neural networks. volume 15, 01 2010.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. arXiv:1512.03385.

[19] Gaoshuang Huang, Yang Zhou, Xiaofei Hu, Chenglong Zhang, Luying Zhao, Wenjian Gan, and Mingbo Hou. DINO-Mix: Enhancing Visual Place Recognition with Foundational Vision Model and Feature Mixing. *arXiv e-prints*, page arXiv:2311.00230, October 2023. arXiv:2311.00230, doi:10.48550/arXiv.2311.00230.

[20] Sergio Izquierdo and Javier Civera. Optimal transport aggregation for visual place recognition, 2023. arXiv:2311.15937.

[21] Hervé Jégou, Matthijs Douze, Cordelia Schmid, and Patrick Pérez. Aggregating local descriptors into a compact image representation. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3304–3311, 2010. doi:10.1109/CVPR.2010.5540039.

[22] Nikhil Keetha, Avneesh Mishra, Jay Karhade, Krishna Murthy Jatavallabhula, Sebastian Scherer, Madhava Krishna, and Sourav Garg. Anyloc: Towards universal visual place recognition, 2023. arXiv:2308.00688.

[23] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. arXiv:1412.6980.

[24] Maria Leyva-Vallina, Nicola Strisciuglio, and Nicolai Petkov. Data-efficient large scale place recognition with graded similarity supervision, 2023. arXiv:2303.11739.

[25] María Leyva-Vallina, Nicola Strisciuglio, and Nicolai Petkov. Regressing transformers for data-efficient visual place recognition, 2024. arXiv:2401.16304.

[26] Feng Lu, Lijun Zhang, Xiangyuan Lan, Shuting Dong, Yaowei Wang, and Chun Yuan. Towards seamless adaptation of pre-trained models for visual place recognition. *arXiv preprint arXiv:2402.14505*, 2024.

[27] Will Maddern, Geoff Pascoe, Chris Linegar, and Paul Newman. 1 Year, 1000km: The Oxford RobotCar Dataset. *The International Journal of Robotics Research (IJRR)*, 36(1):3–15, 2017. URL: http://dx.doi.org/10.1177/0278364916679498, arXiv:http://ijr.sagepub.com/content/early/2016/11/28/0278364916679498.full.pdf+html, doi:10.1177/0278364916679498.

[28] Chen Mao and Jingqi Hu. Progeo: Generating prompts through image-text contrastive learning for visual geo-localization, 2024. arXiv:2406.01906.

[29] Riccardo Mereu, Gabriele Trivigno, Gabriele Berton, Carlo Masone, and Barbara Caputo. Learning sequential descriptors for sequence-based visual place recognition, 2022. arXiv:2207.03868.

[30] Riccardo Mereu, Gabriele Trivigno, Gabriele Berton, Carlo Masone, and Barbara Caputo. Learning sequential descriptors for sequence-based visual place recognition, 2022. arXiv:2207.03868.

[31] Michael J. Milford and Gordon. F. Wyeth. Seqslam: Visual route-based navigation for sunny summer days and stormy winter nights. In *2012 IEEE International Conference on Robotics and Automation*, pages 1643–1649, 2012. doi:10.1109/ICRA.2012.6224623.

[32] NVIDIA, Péter Vingelmann, and Frank H.P. Fitzek. Cuda, release: 10.2.89, 2020. URL: https://developer.nvidia.com/cuda-toolkit.

[33] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision, 2024. arXiv:2304.07193.

[34] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[35] Filip Radenović, Giorgos Tolias, and Ondřej Chum. Fine-tuning cnn image retrieval with no human annotation, 2018. arXiv:1711.02512.

[36] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021. arXiv:2103.00020.

[37] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2015. URL: http://dx.doi.org/10.1109/CVPR.2015.7298682, doi:10.1109/cvpr.2015.7298682.

[38] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015. arXiv:1409.1556.

[39] Niko Sünderhauf, Peer Neubert, and Peter Protzel. Are we there yet? challenging seqslam on a 3000 km journey across all four seasons. *Proc. of Workshop on Long-Term Autonomy, IEEE International Conference on Robotics and Automation (ICRA)*, page 2013, 01 2013.

[40] Ilya Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, Mario Lucic, and Alexey Dosovitskiy. Mlp-mixer: An all-mlp architecture for vision, 2021. arXiv:2105.01601.

[41] Akihiko Torii, Josef Sivic, Masatoshi Okutomi, and Tomas Pajdla. Visual place recognition with repetitive structures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(11):2346–2359, 2015. doi:10.1109/TPAMI.2015.2409868.

[42] Issar Tzachor, Boaz Lerner, Matan Levy, Michael Green, Tal Berkovitz Shalev, Gavriel Habib, Dvir Samuel, Noam Korngut Zailer, Or Shimshi, Nir Darshan, and Rami Ben-Ari. Effovpr: Effective foundation model utilization for visual place recognition, 2024. arXiv:2405.18065.

[43] Ruotong Wang, Yanqing Shen, Weiliang Zuo, Sanping Zhou, and Nanning Zheng. Transvpr: Transformer-based place recognition with multi-level attention aggregation, 2022. arXiv:2201.02001.

[44] Frederik Warburg, Søren Hauberg, Manuel López-Antequera, Pau Gargallo, Yubin Kuang, and Javier Civera. Mapillary street-level sequences: A dataset for lifelong place recognition. In *Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[45] Junqiao Zhao, Fenglin Zhang, Yingfeng Cai, Gengxuan Tian, Wenjie Mu, Chen Ye, and Tiantian Feng. Learning sequence descriptor based on spatio-temporal attention for visual place recognition. *IEEE Robotics and Automation Letters*, 9(3):2351–2358, March 2024. URL: http://dx.doi.org/10.1109/LRA.2024.3354627, doi:10.1109/lra.2024.3354627.

[46] Sijie Zhu, Linjie Yang, Chen Chen, Mubarak Shah, Xiaohui Shen, and Heng Wang. r^2former: Unified $r$etrieval and $r$eranking transformer for place recognition, 2023. arXiv:2304.03410.