Visual Place Recognition for Improved Open and Uncertain Navigation

William Hugh Burrough Smith

Doctor of Philosophy
Heriot-Watt University and The University of Edinburgh
June 2023

Declaration

I declare that this thesis was composed by myself and that the work contained therein is my own, except where explicitly stated otherwise in the text.

(*William Hugh Burrough Smith*)

HERIOT WATT UNIVERSITY

Inclusion of Published Works

Please note you are only required to complete the Inclusion of Published Works Form if your thesis contains published works under Regulation 6 (9.1.2)

Declaration

This thesis contains one or more multi-author published works. In accordance with Regulation 6 (9.1.2) I hereby declare that the contributions of each author to these publications is as follows:

Citation details	**Image Pre-processing vs. Transfer Learning for Visual Route Navigation**. William H. B. Smith, Robert Fisher, Yvan R. Petillot. In UKRAS20 Conference: "Robots into the real world", 2020
William H. B. Smith	Developed theory, implemented theory, ran experiments, wrote paper
Yvan Petillot	Helped with design of theory
Robert B. Fisher	Helped with design of theory, experiments and writing of the paper
Signature:	William H. B. Smith
Date:	28/06/2022

Citation details	**OpenSceneVLAD: Appearance Invariant, Open Set Scene Classification**. William H. B. Smith, Michael Milford, Klaus McDonald-Maier, Shoaib Ehsan, Robert Fisher. In Proceedings of IEEE International Conference on Robotics and Automation (ICRA), 2022
William H. B. Smith	Developed theory, implemented theory, ran experiments, wrote paper
Michael Milford	Provided feedback on paper and theory
Klaus McDonald-Maier	Provided general supervision feedback
Shoaib Ehsan	Provided feedback on paper and theory
Robert Fisher	Helped with experiments and writing of the paper
Signature:	William H. B. Smith
Date:	28/06/2022

veni, vidi, vixi

Abstract

Visual place recognition localises a query place image by comparing it against a reference database of known place images, a fundamental element of robotic navigation. Recent work focuses on using deep learning to learn image descriptors for this task that are invariant to appearance changes from dynamic lighting, weather and seasonal conditions. However, these descriptors: require greater computational resources than are available on robotic hardware, have few SLAM frameworks designed to utilise them, return a relative comparison between image descriptors which is difficult to interpret, cannot be used for appearance invariance in other navigation tasks such as scene classification and are unable to identify query images from an open environment that have no true match in the reference database. This thesis addresses these challenges with three contributions. The first is a lightweight visual place recognition descriptor combined with a probabilistic filter to address a subset of the visual SLAM problem in real-time. The second contribution combines visual place recognition and scene classification for appearance invariant scene classification, which is extended to recognise unknown scene classes when navigating an open environment. The final contribution uses comparisons between query and reference image descriptors to classify whether they result in a true, or false positive localisation and whether a true match for the query image exists in the reference database.

Acknowledgements

I would like to thank my supervisors Prof. Robert Fisher and Prof. Yvan Petillot for their help during this project.

I am particularly grateful to Prof. Fisher who agreed to be my second supervisor a year into the project, despite being under no obligation to do so. His kindness, generosity and academic integrity has taught me a huge amount.

Thanks to Dr. Sven Shepstone, Dr. Pablo Martinez Nuevo and the research group at Bang and Olufsen in Struer, Denmark for welcoming me as an intern and your kind words about my work. In particular I'd like to thank Jeffrey Thomsen and Cornelius Müller for your friendship. Also, my heartfelt thanks to Bo Nyborg, my language teacher and my best Danish friend, I look forward to our next whisky mac.

Thank you to Dr. Shoiab Ehsan for the time I spent at Essex University and the opportunity to be part of the research group. I'd also like to thank Prof. Michael Milford whose detailed and thoughtful feedback was very useful to me.

I would like to express my gratitude to my friends outside work, namely Giula Malusà, John Hill, Christiane Sandbye, and David Ager for their support, hospitality and tolerance of me. I'd also like to thank the Wetterhamns for their hospitality and kindness. An honourable mention also goes to Wilhelmiina Toivo for telling me to leave the first chapter's contribution alone, without your advice I'd probably still be writing this thesis.

I would like to thank my family for their support. In particular my Mum Helen, and my Dad Matthew, who have given me a huge amount of emotional and financial

support. I really could not have done this without you. Thank you for the phone calls Mum, you turned up every day for me. I'd also like to thank both my maternal grandparents whose kindness and financial support have enabled me to get to this point. A massive thanks also to my brother, Ted. Your phone calls, invitations to game and endless funny stories have done more to brighten my days than you will ever know.

Finally, thank you to the Edinburgh Centre for Robotics and EPSRC for funding my research and giving me the opportunity to be part of such a great program.

Contents

List of Figures

List of Tables

Nomenclature

CNN Convolutional Neural Network

CPU Central Processing Unit

DCC Descriptor Comparison Classification

DNN Deep Neural Network

GPS Global Positioning System

GPU Graphical Processing Unit

IMU Inertial Measurement Unit

OC Outcome Classification

OPR Online Place Recognition

OSC Open Set Classification

OSR Open Set Recognition

OSS2 OpenSeqSLAM 2.0

PDF Probability Density Function

SIS Sequential Importance Sampling

SLAM Simultaneous Localisation and Mapping

VO Visual Odometry

VPR Visual Place Recognition

VT&R Visual Teach and Repeat

Part I

Introduction

Chapter 1

Introduction

Visual place recognition [1, 2] is a fundamental area of research in the field of robotics that overlaps heavily with computer vision and deep learning and has applications for other fields such as consumer photography and augmented reality [2, 3]. It refers to a critical requirement for enabling autonomous mobile robots:

A robot should be able to use onboard cameras to localise itself within a geographical area and extract relevant metadata to enable intelligent behaviour in real-time.

This thesis focuses on fulfilling this requirement by expanding visual place recognition to address uncertain and open world conditions inherent to robotics. Monocular RGB images are considered because robots are often fitted with RGB cameras which are of comparatively low cost compared to other sensors, high quality pre-recorded data is readily available and further data is easy to collect. A comparison of RGB data with alternatives is available in Section 2.1.2.

1.1 Problem Statement

Visual Simultaneous Localisation and Mapping (vSLAM) is a foundational research area for robotics that has developed significantly over time [4, 5, 6]. However, current approaches such as ORB-SLAM [5] are limited by their use of image descriptors [7] which are vulnerable to variations in the navigated environment's appearance, these image descriptors are used to transfer raw image data to information that can be used by the algorithm and therefore represent a performance bottleneck. Deep learning has introduced the possibility of learning a visual SLAM algorithm from scratch, but is yet to solve challenges associated with gathering training data, generalising to new

environments and interpretation of taught models [8, 9].

Visual place recognition is an area of research that frames localisation as image retrieval by comparing a query place image against a reference set of place images to find the best match, and therefore localise the query image. This naive search represents an extreme example of the localisation undertaken in vSLAM which uses a variety of other techniques to reduce the naivety of this search but fundamentally still requires a comparison between place images. Deep learning has recently been used to generate state-of-the-art descriptors for this image comparison, such as NetVLAD [10]. Other forms of data can be used as a reference to compare query images against, for example traditional maps [11], but images are easy to collect, use for comparison and are the dominant form of reference data [10, 12, 13, 14, 15].

Deep learning approaches for visual place recognition therefore represent a good compromise for using deep learning to improve vSLAM without the aforementioned challenges of learning a vSLAM approach from scratch because they are easier to interpret and can potentially be combined with tried and tested vSLAM approaches [16]. This thesis therefore focuses on visual place recognition and its application to vSLAM. Current approaches focus on invariance to appearance changes caused by dynamic viewpoint, lighting or weather conditions [7, 17, 13, 18]. However, these approaches often fail to consider tasks specific to robotics, such as leaving a pre-defined geographical area during navigation and getting lost, or estimating the uncertainty of predictions generated by a deep neural network, which prevents their application to real-world problems:

Computational Requirements of large deep neural networks (DNNs) taught to generate visual place recognition descriptors make them impractical for robotics, because current hardware is not powerful enough to run them in real-time [16] (*addressed in Chapter 3*).

Integration of deep learnt visual place recognition descriptors into visual SLAM pipelines is difficult because they were originally built for traditional image descriptors such as SURF [4] and ORB [6]. Deep learnt descriptors generate a single descriptor for each image [10, 13] rather than traditional local image descriptors which means that vSLAM pipeline elements that depend on local descriptors or information derived from them such as map point extraction and bundle adjustment (2.2) are not

relevant. Additionally, deep learnt image descriptors are capable of predicting further information such as their own uncertainty which is not possible with traditional image descriptors and is therefore not considered in current vSLAM pipelines (*addressed in Chapter 3*).

Ambiguous image retrieval results can be difficult to interpret correctly. Image retrieval returns a ranking of the reference database images in order of similarity to the query image where the true match is often not the first result, but may be in the top ten, for example (*addressed in Chapters 3 & 5*).

Non-Transferable but very useful characteristics of visual place recognition descriptors cannot be applied to other navigation related tasks, such as scene classification [19, 20]. Learning to classify different scenes and to generate visually invariant descriptors are two different tasks with no current approach to learn them simultaneously. Networks taught on each task currently exist but there are no ways currently to combine them (*addressed in Chapter 4*).

Open Set recognition aims to identify input data that was not defined at training time. For example, current classification scenarios aim to differentiate between a fixed number of classes but do not consider how to handle test data that does not belong to any of those classes, because there is no prior knowledge of this data this is clearly a very challenging task. This problem relates closely to robot navigation in an open world, but has only recently been considered for classification [21, 22] and remains largely unconsidered for visual place recognition (*addressed in Chapters 4 & 5*).

1.2 Original Contributions

This thesis focuses on using deep learning and probabilistic filtering to address the previously highlighted problems. These problems are not limited to robotics so this thesis's contributions are relevant to wider areas of computer vision (scene classification and image retrieval) and deep learning (filtering deep learning predictions and open set recognition), links to these areas are explored more in Section 2. This section highlights the contributions made in each chapter.

Chapter 3: Particle Filtering for Robust Real-Time Visual Teach and Repeat

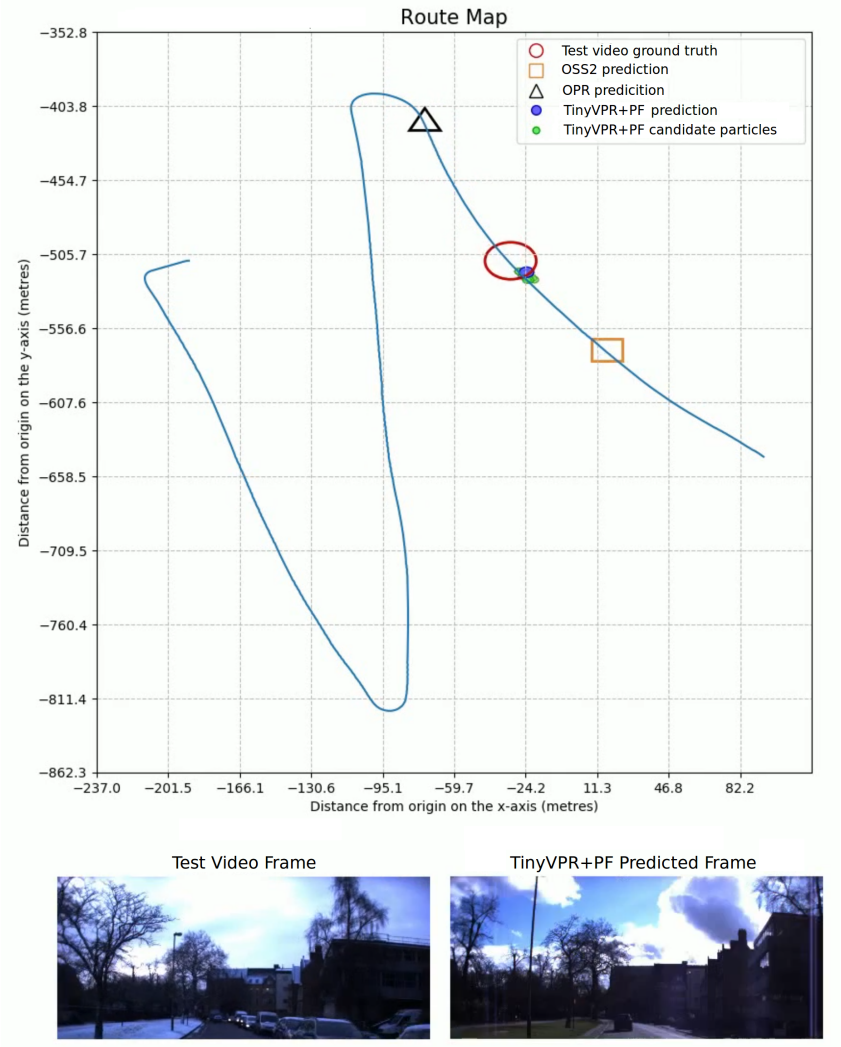


Figure 1.1: This figure shows a route around Oxford (blue line) being repeated by comparing test video frames of the route repetition with pre-recorded reference examples of the route. The proposed particle filter system (TinyVPR+PF) is shown to produce a more accurate localisation prediction than two state-of-the-art alternatives OpenSeqSLAM 2.0 (OSS2) and Online Place Recognition (OPR).

Theory and Algorithms

1. A compact VPR descriptor called TinyVPR for a subset of visual SLAM called visual teach and repeat that improves state-of-the-art localisation performance on a specific route while significantly reducing network parameters.

2. A particle filter to combine TinyVPR descriptors and visual odometry to enable real-time visual teach and repeat, as seen in Figure 1.1 that significantly improves robustness to large visual changes and variable traversal speed input.

TinyVPR is covered in: **Image Pre-processing vs. Transfer Learning for Visual Route Navigation.** William H. B. Smith, Robert Fisher, Yvan R. Petillot. In *UKRAS20 Conference: "Robots into the real world", 2020.*

Chapter 4: OpenSceneVLAD: Appearance Invariant, Open Set Scene Classification

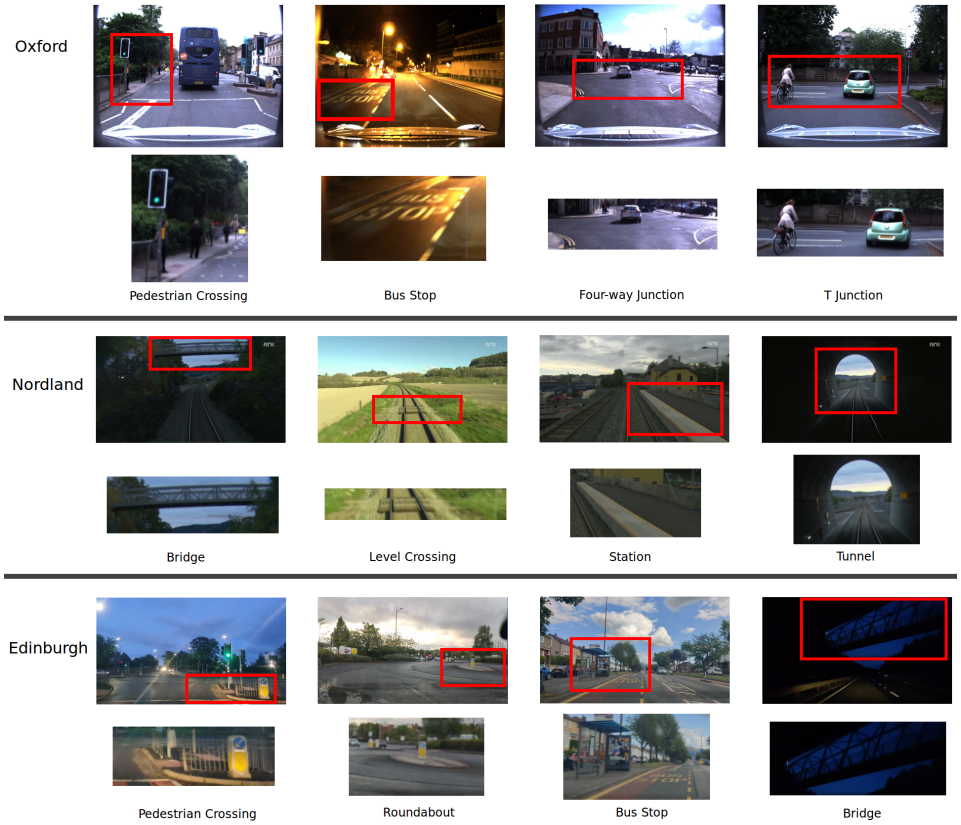


Figure 1.2: Samples of scene classification data from the three datasets used with appearance variations, including the Edinburgh dataset collected as part of this thesis' contributions. A red box is used to highlight the most class relevant areas of each image, which are also enlarged for the reader's convenience. Duplicate of Figure 4.1

Theory and Algorithms

1. For the first time the problem of visual invariance for scene classification is identified and improved with a fusion of visual place recognition and scene classification neural networks called SceneVLAD.

2. For the first time open set recognition is considered for scene classification with test data from previously undefined scene classes. Intra-class splitting is used to

create OpenSceneVLAD for improved classification of the open and closed set scene data that also includes visual changes.

Data

1. A visual place recognition dataset collected around Edinburgh in suburban, motorway and rural environments.
2. A scene classification dataset that includes changes in visual conditions (Figure 1.2).
3. An open set scene classification dataset derived from visual place recognition datasets.

OpenSceneVLAD: Appearance Invariant, Open Set Scene Classification. William H. B. Smith, Michael Milford, Klaus McDonald-Maier, Shoaib Ehsan, Robert Fisher. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA), 2022*

Chapter 5: Descriptor Comparison Classification for Open Set Recognition and Outcome Classification

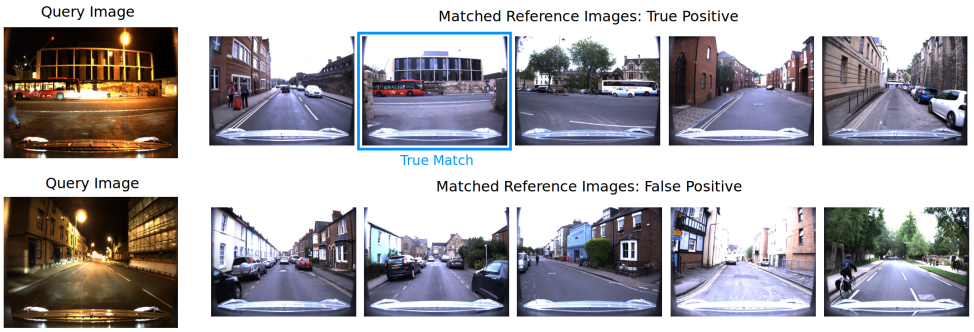


Figure 1.3: Illustrated example of outcome classification task using images from the Oxford RobotCar dataset. The top query image's 5 closest reference image matches include a true match (true positive) and the bottom query image's do not (false positive). Please note, the 5 closest matches are shown here for illustrative purposes only. Duplicate of Figure 5.2.

Theory and Algorithms

1. For the first time the problems of outcome classification (identifying true and false positive image retrieval results) and open set recognition for image retrieval are formulated and applied to visual place recognition.

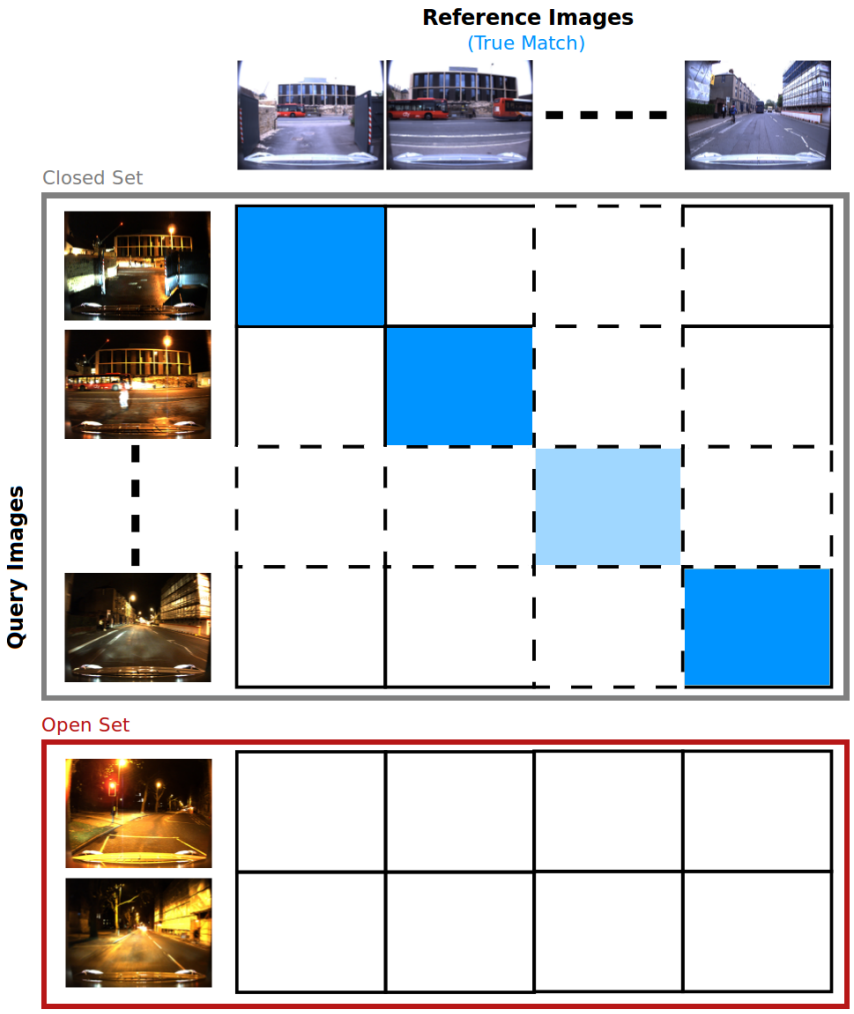


Figure 1.4: Illustrated example of open set recognition task using images from the Oxford RobotCar dataset showing examples of closed set query images that have a potential match in the reference image database and open set query images that do not. Duplicate of Figure 5.3.

2. For the first time descriptor comparison classification is proposed to reliably address both problems.

3. The addition of supervised contrastive learning is used to improve descriptor comparison classification for both problems.

Source code related to this thesis has, or will be, released publicly (https://github.com/WHBSmith). The remaining chapters are used to provide background on the research areas related to these contributions (**Chapter 2**) and the conclusion of this thesis (**Chapter 6**).

Chapter 2

Literature Review

Visual place recognition (VPR) has recently been defined by Garg et al. [2] as *'the ability to recognise one's location based on two observations perceived from overlapping fields-of-view'*. This section begins with an overview of visual SLAM and image retrieval, which provides the context for then introducing VPR for robotics. The other research areas relevant to this thesis's contribution are then explored. The contents of this literature review is summarised below:

- **Visual SLAM** is a fundamental area of robotics that typically uses VPR image descriptors to consistently represent a single place uniquely with respect to other places.
- **Image Retrieval** is the format of the VPR task when considered in isolation: a query image is localised by comparing it to a database of reference images to find the closest match.
- **Visual Place Recognition** is the main subject of this thesis and aims to create accurate and robust place image descriptors for use in an image retrieval framework.
- **Visual Teach & Repeat** is a subset of visual SLAM that relies on VPR to solve autonomous route repetition, a task that remains relevant in many robotics scenarios.
- **Particle Filtering** has traditionally been used in robotics for noisy SLAM predictions, but can also be applied to deep learning predictions.

- **Scene Classification** is a well-defined area of research, closely related to visual navigation, that aims to classify place images as members of scene classes.
- **Deep Learning** includes specific areas of research within itself that are relevant to robotics and VPR scenarios.

2.1 Visual SLAM (vSLAM)

vSLAM [23] uses visual sensors (this thesis focuses on RGB cameras) to calculate the position and orientation of the sensors relative to their surroundings, while simultaneously mapping the environment. A typical vSLAM pipeline extracts image descriptors to compare against previously surveyed descriptors (VPR). The current camera pose is then estimated using the descriptor matching results and local bundle adjustment to jointly optimise the camera pose and the map. Finally loop-closure detection is used to recognise previously visited places and therefore provides an opportunity to correct localisation drift.

This section gives a brief introduction to vSLAM, establishes its link with VPR and demonstrates the need for more robust and sophisticated image descriptors. Learning vSLAM end-to-end is also explored and found to be a promising, but not yet viable approach for robotics. Please note, in the interest of simplicity this work does not distinguish between image feature extraction and description, instead the term *image descriptor* is used to encapsulate the result of the entire process.

2.1.1 Local Descriptors

Early vSLAM approaches were restricted to small, static, indoor environments [24], but the introduction of scale invariant image descriptors such as SIFT [25] and SURF [26] which are algorithms designed to locate features in small parts of an image, commonly known as 'keypoints'. These keypoints are scale and rotation invariant so enabled larger and more complex indoor environments to be navigated. For example, Se et al. [27] use SIFT image descriptors to build and query a database of visual landmarks for navigation and later basic simultaneous mapping and loop closure [28].

One of the first monocular vSLAM approaches was MonoSLAM [4] which tracks visual landmarks between frames using SURF feature matching and an Extended Kalman Filter. Another vSLAM approach, PTAM [29], developed for VR and AR ap-

plications introduced bundle adjustment for better accuracy and multi-threading for more efficient computational performance. ORB-SLAM [30] extends PTAM further by introducing ORB descriptors [31] which are a fusion of a FAST keypoint detector [32] and BRIEF descriptors [33] for a degree of invariance to rotation and scale, resulting in a very fast multi purpose descriptor with robustness to viewpoint changes. ORB-SLAM also added parallelisation of tracking, mapping and loop closure detection, as well as pose graph optimisation making it one of the most complete open-source vSLAM approaches available [34].

2.1.2 Local Descriptors & 3D Data

3D data was combined with local descriptors to further improve vSLAM. Newcombe et al. combine depth maps with camera motion to estimate 3D structures [35]. ORB-SLAM2 also extends ORB-SLAM to stereo and RGB-D sensors [5]. Salas-Moreno et al. [36] propose registering 3D objects into a database in advance and recognising them at test time to refine map building and reduce storage requirements. A similar approach segments 3D objects to aid localisation [37]. RGB-D cameras are less ubiquitous than RGB cameras but allow scale to be directly calculated, however consumer versions only have a range of about 4 metres which makes them of limited use outdoors.

Light Detection And Ranging (LiDAR) is an alternative way to generate 3D data, particularly for invariance to appearance changes from ambient lighting conditions [38] and obstacle detection [39]. It has an improved range of up to around 500 metres which makes it well-suited for outdoor navigation, but LiDAR is more expensive, vulnerable to occlusion. Additionally approaches for processing 3D data are less advanced than RGB data because of the wide availability of RGB data and the fact that it is sufficient for many tasks.

State-of-the-art vSLAM approaches continue to use local geometric descriptors, which are significantly outperformed by deep learnt equivalents, particularly for appearance invariance [7]. The results of an early experiment carried out by the author when evaluating ORB-SLAM2 are shown in Figure 2.1 and demonstrate the negative effect of appearance changes from lighting conditions. RTABMap [40] is an open source vSLAM approach originally released in 2013 but updated to create dense 3D and 2D maps that can be used as drop-in alternatives to SLAM using 2D LiDAR, it uses BRIEF descriptors [33] for odometry. OpenVSLAM [6] is a recent and well-engineered,

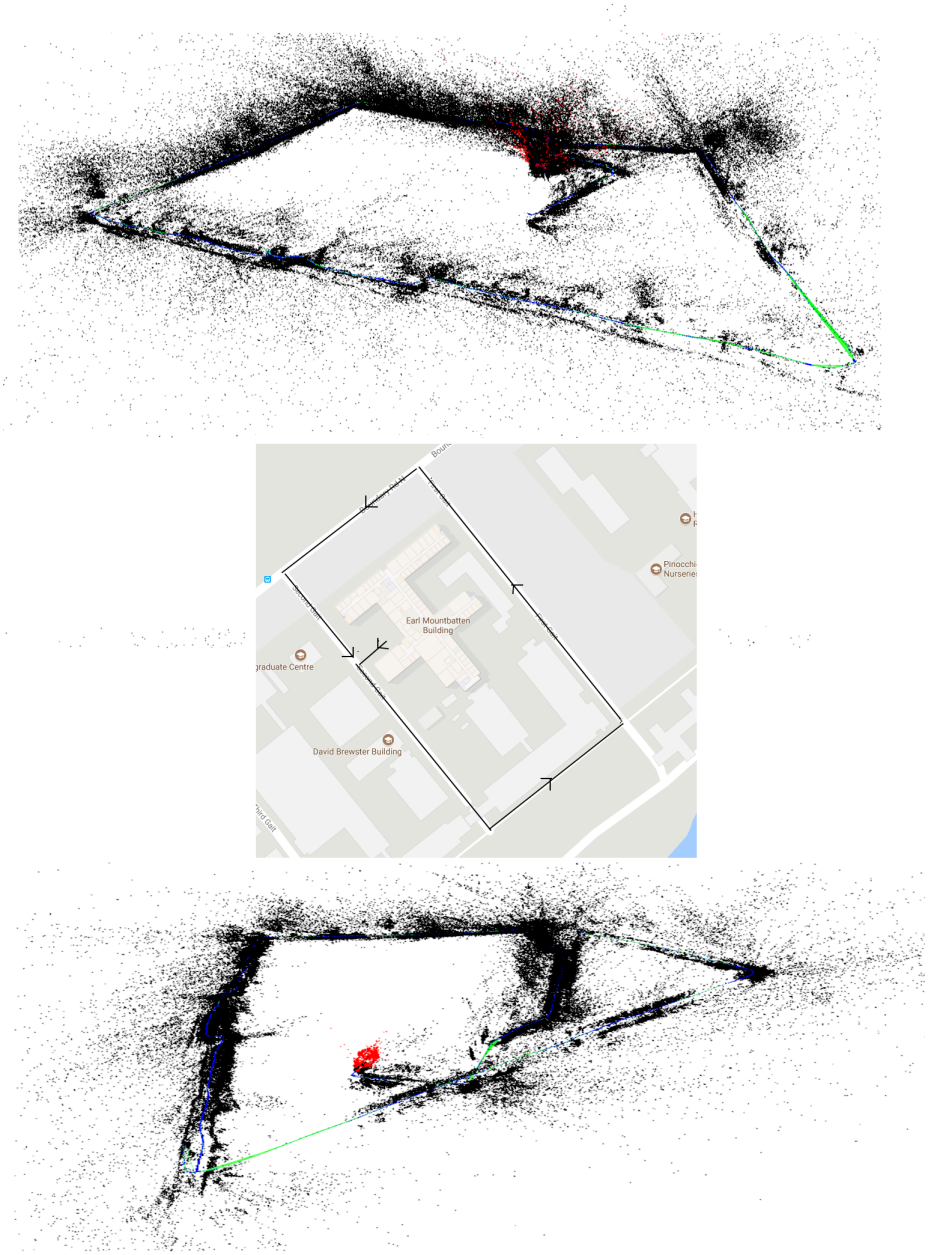


Figure 2.1: An evaluation of the ORB-SLAM2 [5] algorithm carried out at the beginning of the project by the author. A map was built using ORB-SLAM2 by navigating around Heriot-Watt University campus (centre map) at dusk (top) and the night (bottom) using a remote-controlled 'Husky' UGV [1]. In the figure black dots represent collected point cloud data, red dots represent the start and end point and the green/blue line represents the robot's path. The final map is viewed at an oblique angle. This figure shows the created map degrading as lighting conditions worsened, resulting in a wedge-shaped map rather than the correct square. This shows that, despite point cloud data, the appearance changes from the lighting affect the remaining ORB image descriptor matching and therefore result in decreased vSLAM performance. Additionally ORB-SLAM2 is given the opportunity for loop closure as the robot returns to its starting point, but this fails.

vSLAM approach that uses ORB descriptors [31] and a unique frame tracking module that enables fast and accurate localisation. ORB-SLAM3 [41] remains a state-of-the-art approach by updating ORB-SLAM2 with more comprehensive bundle adjustment, improved IMU tracking and active mapping which results in a pipeline with 31 components (Figure 2.2). Large pipelines are required to compensate for image descriptors that are unable to return accurate localisation estimates when compared against a large number of image descriptors across a wide geographical area, an assumption challenged by the latest advances in VPR. Please note, from now on image descriptors used specifically for comparing place images against a reference database of image descriptors are described as 'VPR descriptors'.

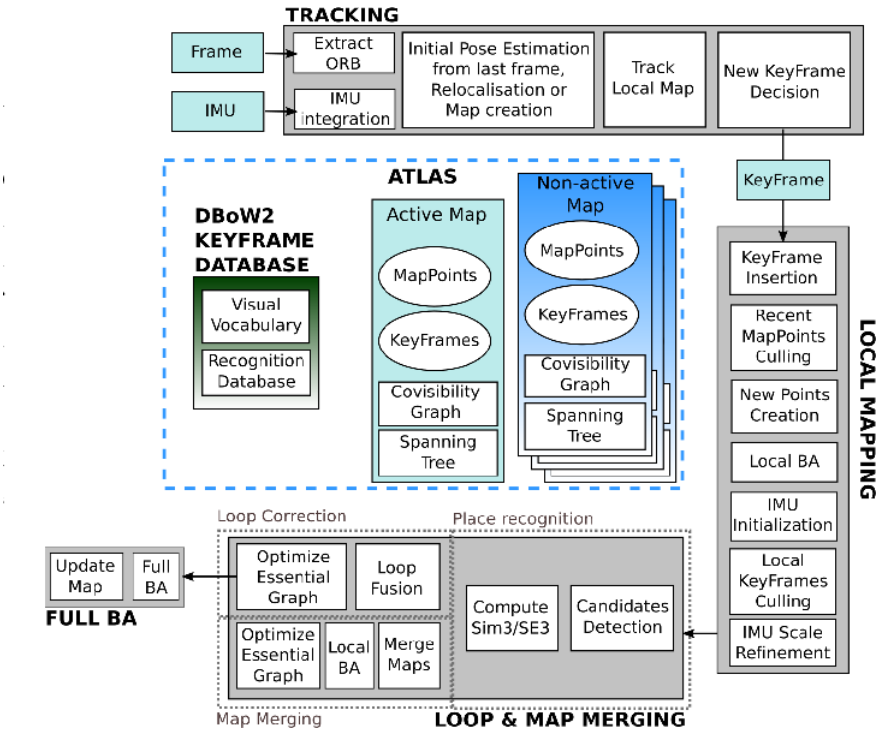


Figure 2.2: A diagram showing the large number of ORB-SLAM3 system components [41] including bundle adjustment (BA).

2.1.3 Deep Learning for vSLAM

Deep learning has enabled significant advancements in robotics, but learnt navigation approaches lack maturity compared to the well understood and explored traditional vSLAM algorithms. There are many different learnt approaches to visual navigation [42], but this section focuses on approaches for large scale outdoor navigation using

[1]https://clearpathrobotics.com/husky-unmanned-ground-vehicle-robot/

two main approaches.

2.1.3.1 Convolutional Neural Networks (CNNs)

CNNs have rarely been implemented on robot hardware because of the associated computational requirements, however this is starting to be addressed, as discussed later in Section 2.7.1. CNN-SLAM [16] proposes two parallel pipelines. The first uses a CNN to estimate the camera pose by minimising the photometric error between the current frame and the nearest keyframe. The second uses a CNN to performs semantic segmentation on each frame and depth information is predicted for each keyframe but this is not a feature based approach and is not tested outdoors. Another noteworthy approach is PoseNet [43], which trains a CNN directly on traversals of a geographical area to regress a pose, at test time traversals from different poses are used for testing - the disadvantage of this approach is that it is constrained to quite small areas and does not generalise to different areas without re-training.

2.1.3.2 Deep Reinforcement Learning (DRL)

DRL tries to avoid error accumulated in complex vSLAM pipelines by using CNN image descriptors to infer navigation policy (vNavigation) from navigation data using a number of approaches:

1. *Direct* approaches learn from place images collected by agents moving around an environment to find the goal object for sparse rewards [8].

2. *Hierarchical* approaches decompose vNavigation into subproblems and solve each of them to generate a global navigation policy [9].

3. *Multitask* DRL agents uses shared neural network parameters from related tasks to improve generalisation to different environments [44].

4. *External memory* for DRL agents can improve visual navigation performance in partially observable and large-scale environments [45].

5. *Vision-and-language* navigation models [46] fuse language instructions with vision state inputs to integrate control with navigation.

DRL methods for this task show promise, but the large amount of training data required, poor generalisation and lack of interpretability means that these approaches remain restricted to simulated environments for now.

2.2 Image Retrieval

The previous section establishes that VPR descriptors are a key component of enabling robust vSLAM in outdoor environments and that using DRL to learning the task of vSLAM is not yet practically possible. In isolation VPR is typically formulated as image retrieval: a query image is compared against a reference database of images to find the nearest match. Although Özaydin et al. suggest that traditional geometric descriptors, such as SURF [26], may be comparable with off-the-shelf CNN descriptors for image retrieval in some scenarios [47] most recent image retrieval descriptors have been based on CNNs since they were established as the state-of-the-art for recognition tasks [48]. CNN descriptors can either be used off-the-shelf or fine-tuned for a specific task.

2.2.1 Off-the-Shelf CNN Descriptors

Single [49] or multiple [50] feedforward passes can be used to extract image retrieval descriptors using off-the-shelf CNNs pre-trained for general image classification, such as VGG16 [12]. Descriptors can also be extracted from final fully-connected layers [51] to give a representation based on lower level features, or intermediate convolutional layers [52] for a representation based on higher level features.

Traditionally, pooling-based aggregation methods for global descriptors are directly applied to DNN output and then the whole model is used end-to-end [53]. Three embedding approaches can also be used to further improve the descriptors: Bag of Words (BoW), Vector of Locally Aggregated Descriptors (VLAD) and Fisher Vectors (FV) [54, 55, 10, 56]. Local descriptors can be used to describe small parts of an image, but require individual storage, are not well-suited for large-scale scenarios and incur additional cross-matching costs. To help mitigate this an initial search is done with global descriptors and then a re-ranking of the top results is done using local descriptors [57, 58]. An attention mechanism can also be used to compute an attention map using channel-wise or spatial-wise pooling [59], or an attention map can be learnt

with image patches or feature maps [60].

2.2.2 Fine-tuned CNN Descriptors

CNNs can be fine-tuned using supervised training to improve descriptor performance. The most straightforward way to do this is to retrain the network using cross entropy loss [61] and then use intermediate layers for descriptor extraction. However, a more common approach is to use similarity learning. This is an important technique with examples referenced throughout this thesis and is explained here in detail as it is foundational to image retrieval.

Similarity Learning also known as metric learning [62], is distinctly different from the traditional problem of classification. Rather than trying to classify test data as a member of a pre-defined class, it aims to represent test data such that the distance between 'similar' data samples is minimised and the distance between 'dissimilar' data samples is maximised according to measures such as the Euclidean distance. For VPR this means a data representation can be learnt that is invariant to appearance changes from dynamic lighting, weather and even seasonal conditions. Assume a dataset $\mathcal{X} = \{x_1, \ldots, x_N\}$ is available, from which certain similarity measures between different pairs or triplets of data are collected. These similarities are described by the sets shown in Equation 2.1. With these data and similarity constraints, the task is to find, after establishing a family of distances $\mathcal{D}$ (often Euclidean), those representations that best adapt to the criteria specified by the similarity constraints. To do this, a certain loss function ℓ is set, and the sought-after representations will be those that solve the optimization problem in Equation 2.2. Examples of loss functions used for image retrieval are siamese loss [63], triplet loss [64] as well as other approaches [65, 66].

$$\begin{aligned} S &= \{(x_i, x_j) \in \mathcal{X} \times \mathcal{X} : x_i \text{ and } x_j \text{ are similar. } \}, \\ D &= \{(x_i, x_j) \in \mathcal{X} \times \mathcal{X} : x_i \text{ and } x_j \text{ are not similar. } \}, \\ R &= \{(x_i, x_j, x_l) \in \mathcal{X} \times \mathcal{X} \times \mathcal{X} : x_i \text{ is more similar to } x_j \text{ than to } x_l\}. \end{aligned} \tag{2.1}$$

$$\min_{d \in \mathcal{D}} \ell(d, S, D, R) \tag{2.2}$$

Unsupervised similarity learning is less studied, but also possible. For exam-

ple, manifold learning finds initial similarities between extracted descriptors and uses them to construct an affinity matrix, which is re-evaluated over time and used to mine training data [67]. The affinity matrix is interpreted as a weighted graph and the pairwise affinities are re-evaluated in the context of all other elements by diffusing the similarity values through the graph [68]. Clustering approaches, such as k-means, are also used to cluster deep features for generating pseudo-labels for pairwise training data [69].

Two big problems that remain for image retrieval are that the results are ambiguous: a ranking of the reference images in order of similarity to the query image often does not include the best match as the most similar and there is no way to recognise a query image that does not have a true match in the reference database at all. A true match for visual place recognition is a reference frame within a user defined geographical distance from the query image.

2.3 Visual Place Recognition (VPR)

Visual place recognition overlaps with many research areas, as shown in Figure 2.3, but is contextualised in this thesis specifically for robotics by its relationship to vSLAM and image retrieval. There are several comprehensive literature surveys of VPR [1, 3, 70, 71], but for the purpose of this thesis it can be thought of as a variation of image retrieval relevant specifically to robotics because of its ability to create appearance invariant descriptors that are needed for vSLAM. VPR is a major application of image retrieval so advances in each are often intertwined.

2.3.1 Local Descriptors

Scale invariant descriptors, such as sped-up robust features (SURF) [26], are used in initial VPR approaches [72, 73], FAST [32] and BRIEF [74] descriptors are also used [75]. These local descriptor approaches were improved by using a visual bag-of-words model to quantise descriptors into a finite number of visual words. For each image, every descriptor is assigned to a particular word reducing images to binary strings with a length equal to the number of words in the vocabulary, which varies from 5000 up to 100,000 [76]. Bag-of-words representations do not describe the geometric structure of the image resulting in some degree of viewpoint invariance [77], but this can reduce

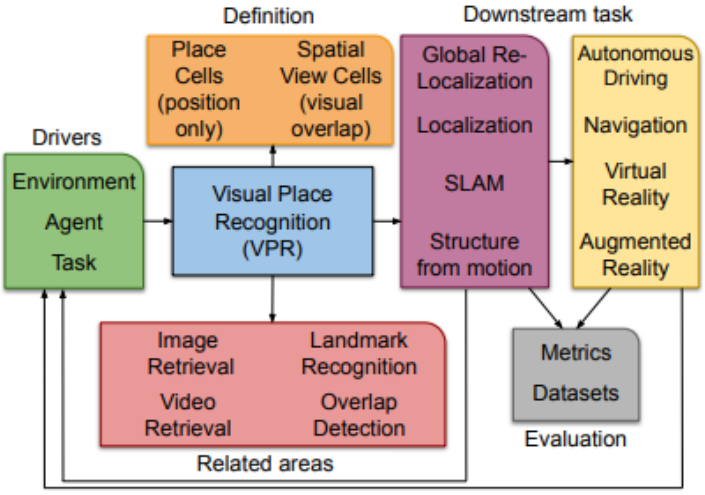


Figure 2.3: An illustrated overview of VPR from Gar et al. [2] showing its relationship to other research areas.

the overall accuracy of VPR, which can be mitigated by requiring 3D data [78]. Additionally, the bag-of-words approach is typically pre-trained on images which may limit this approach's ability to generalise to test data.

2.3.1.1 Appearance Invariance

Geometric descriptors such as SURF [26] have been shown to be non-repeatable in the face of variations in appearance from changes in lighting, weather, and seasonal conditions [79]. In response, a variation on SURF, Upright-SURF, was combined with a consistency check using the epipolar constraint for improved appearance invariance over small 40 image datasets [80]. Other approaches propose image pre-processing techniques, such as shadow removal [81] and illumination invariant colour spaces [82], but still struggle in many scenarios.

An alternative approach to generating visually invariant descriptors is to learn the relationship between different place appearances. Ranganathan et al. [83] learns a probability distribution over a bag-of-words vocabulary from multiple traversals of a single environment in different illumination conditions which could then be used for matching. Carlevaris-Bianco et al. [84] use SURF descriptor patches of place images in different lighting conditions to train a multi-layer perceptron siamese neural network [85] to cluster similar place image descriptors in different conditions together and away from other place image descriptors.

2.3.2 Global Descriptors

The aggregation of local descriptors collected from an image into a single representation can be done directly with whole-image global descriptors. HOG descriptors [86] use a distribution of intensity gradients or edge directions within an image to describe local object appearance and shape. Gist descriptors [87] summarise the gradient information (scales and orientations) for different parts of an image. Global descriptors have been used for VPR [88] and are shown to have advantages for viewpoint changes and occlusions which are key to robotic applications, but are less robust to illumination changes [89].

2.3.3 Off-the-Shelf CNN Descriptors

Similarly to image retrieval, CNNs pre-trained on adjacent tasks such as object recognition [90], scene classification [91] and semantic segmentation [92] have been used for extracting VPR descriptors. Chen et al. [7] use a holistic approach that feeds the whole image to a CNN and uses all activations from a layer as the descriptor. Further research suggests that mid-level network descriptors are more robust to appearance change, while higher level descriptors are more robust to changes in viewpoint [93]. Supporting evidence for this is presented by Yue et al. [90] who encode the output from OxfordNet [12] and GoogLeNet [94] into VLAD descriptors, compress them using PCA [95] and report similar results. Landmarks can also be detected using Edge Boxes [96] or BING [97] and descriptors extracted specifically from them, for example Sünderhauf et al. [98] use AlexNet's conv3 layer [99] for descriptor extraction. Regions of interest can also be derived from intermediate CNN layers. Chen et al. [17] propose using VGG16 [12] to extract regions of interest using a bag-of-words approach [100]. Region-VLAD [101] uses a lightweight scene classification CNN AlexNet365 [17] to extract regional descriptors which are then encoded using a VLAD approach specially adapted for computational efficiency.

2.3.4 Fine-tuned CNN Descriptors

2.3.4.1 Semi-Supervised Methods

Fine-tuning a network for VPR usually involves similarity learning [102]. For VPR 'similar' images are a single place in different appearance variations, while 'dissimi-

lar' images are of any other place and appearance variation. This effectively teaches the network *some differences between place images are due to appearance variation and other differences are because the underlying places are different*. There are several variations of this loss, including contrastive loss [103], triplet loss [104] and quadruplet loss [105], the differences between these losses and training examples for each is pictured in Figure 2.4.

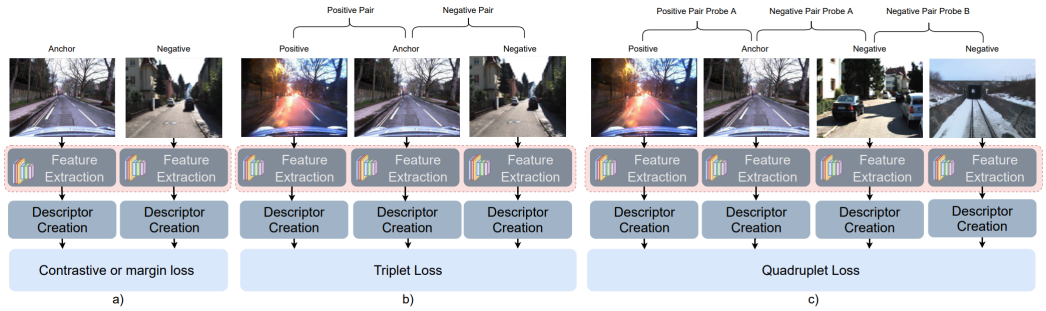


Figure 2.4: A diagram from Barros et al. [70] showing samples of training data used for different contrastive loss variations and the differences between the following losses: contrastive/margin (a), triplet (b) and quadruplet (c).

Arguably the single biggest contribution to VPR is NetVLAD [10] which uses a VGG-16 [12] network pre-trained on the ImageNet dataset [106] up to the conv5 layer with an appended trainable VLAD layer inspired by the "Vector of Locally Aggregated Descriptors" (Figure 2.5). This network is trained using a weakly supervised triplet loss with a large VPR dataset derived from Google Street View data collected from different locations with different appearance variations to generate appearance invariant VPR descriptors. NetVLAD established a new benchmark for performance on Oxford 5K [107], Paris 6K [108] and Holidays [109] datasets that consist of thousands of images spread over large geographical locations and remains the backbone of current state-of-the-art [13] approaches. Other approaches have also used VLAD descriptors, Yu et al. [110] propose a spatial pyramid-enhanced VLAD (SPE-VLAD) layer built around the use of VGG16 [12] and ResNet-18 [111]. A weighted triplet loss is then used for weakly supervised training using GPS tags and the Euclidean distance between the image representations. Qiu et al. [112] use a ResNET-based siamese network [85] (a neural network with shared parameters that is particularly useful when training data is scarce) and trained using an L2-based loss function [113] for VPR descriptor generation. PatchNetVLAD [13] is one of the latest approaches to VPR and the winner of the Facebook Mapillary Visual Place Recognition Challenge at ECCV2020

and therefore can reasonably claimed as the current state-of-the-art for VPR. It uses a re-trained version of NetVLAD [10] for rural or urban environments and then re-ranks the top results using local patch descriptor local matching.

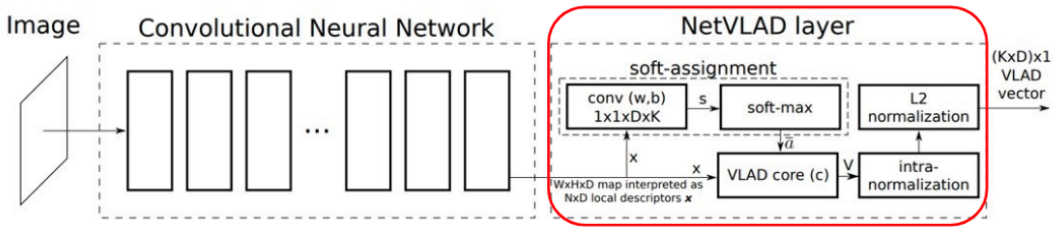


Figure 2.5: Network architecture diagram of the original NetVLAD [10] neural network referenced throughout this paper.

State-of-the-art algorithms [14, 15] introduce significantly larger scale datasets such as Nordland [114], Oxford RobotCar [115] and St. Lucia [116] which cover tens, or even hundreds, of kilometers of urban and rural environments in a wide variety of visual conditions which in turn increases the priority of improving VPR descriptor efficiency. Deep hashing can be used to reduce descriptors to binary codes which have low storage requirements [117], in this case MobileNet [118] is used as the base network. Network pruning can also improve efficiency by removing unnecessary neurons or setting the weights to zero [119]. Hausler et al. [120] propose removing feature maps at the beginning of a HybridNet [14] network while using descriptors from later layers of the network for efficient matching which are selected using a triplet loss calibration procedure.

2.3.4.2 Unsupervised Methods

Unsupervised learning does not require labeled data, which can be impractical for the large scale datasets associated with this task.

Latif et al. [121] use Generative Adversarial Networks (GANs) for domain translation for appearance invariance without labelled cross-domain place image correspondences. However, only the transition between summer and winter images is considered and as all possible translations would have to be learned the paper itself describes this approach as impractical.

An autoencoder-based approach [122] uses HOG [86] image descriptors for extracting salient features and a projective transformation (homography) [123] to describe differing viewpoints as input to a minimal autoencoder architecture to gener-

ate a VPR descriptor, but only compares itself against off-the-shelf CNN descriptors. Tang et al. [124] also use a modified autoencoder for adversarial learning that is used to disentangle VPR descriptors from domain-related descriptors, but does not tackle appearance invariance.

Wang et al. [18] propose the use of attention for VPR descriptors that are robust to very long-term visual changes. A combination of weakly supervised triplet learning and an unsupervised multi-kernel maximum mean discrepancy (MK-MMD) loss function is used for training. This work is useful for historical data, but these time periods are not currently considered during robotic navigation.

2.3.5 Limitations

Visual place recognition initially used traditional image descriptors for small-scale VPR in fairly constant lighting conditions, but this was found to be vulnerable to appearance variation from changes in lighting, weather and season. The same limitations are found in current state-of-the-art vSLAM systems which use the same descriptors. The introduction of fine-tuned CNNs using variations of contrastive learning has enabled VPR to expand to larger datasets and resulted in significantly improved visual robustness, but many challenges still remain.

Deep learnt descriptors generate a single descriptor for each image [10, 13] rather than traditional local image descriptors which means that vSLAM pipeline elements that depend on local descriptors or information derived from them such as map point extraction and bundle adjustment (2.2) are not relevant. Additionally, deep learnt image descriptors are capable of predicting further information such as their own uncertainty which is not possible with traditional image descriptors and is therefore not considered in current vSLAM pipelines

1. **Computational Requirements** - State-of-the-art CNN VPR descriptors are rarely used in mobile robotics because they are too computationally demanding for real-time operation on robotic hardware.

2. **Integration** - traditional vSLAM pipelines include modules such as map point extraction and bundle adjustment which rely on local descriptors that are not produced by deep neural networks.s

3. **Ambiguity** - VPR results are a ranked similarity of the reference image database which is ambiguous and difficult to interpret.

4. **Non-Transferable** - Appearance invariance of VPR descriptors cannot be transferred to other navigation-relevant tasks, such as scene classification.

5. **Open Set** - The recognition of query images with no potential true match in the reference image database for VPR has not yet been considered.

These highlighted challenges were motivation for this thesis's contributions, stated in Chapter 1. The following sections of this literature review are dedicated to the remaining research areas relevant to these contributions.

2.4 Visual Teach and Repeat (VT&R)

VT&R represents a subset of the visual SLAM problem [125] that avoids the need for a full VPR pipeline by relying on accurate VPR descriptors and visual odometry (VO) for navigation. VT&R was originally designed for GPS-denied environments [79, 126] and therefore only requires reference images collected along a pre-defined route to autonomously navigate it. A robot is initially piloted along a traversal of a route, during this teaching phase it records images of the route in different appearance variations which are used as reference for the robot to autonomously repeat the route. Reference images do not need an exact location associated with them for relative localisation predictions. Furgale et al. [79] were the first to implement this technique outdoors over kilometre distances using stereo cameras for calculating visual odomoetry between frames and SURF descriptors for localisation. The pipeline used for this approach is shown in Figure 2.6.

2.4.1 Visual Invariance

Furgale et al. [79] use SURF descriptors [26] for this approach which are found to struggle when presented with significant lighting variations which were partially addressed using image pre-processing [127]. VT&R is commonly used for long term repetitions and as such requires VPR descriptors to be updated which can be done by recording newly seen images as 'experiences' [75], or by using VPR descriptors that

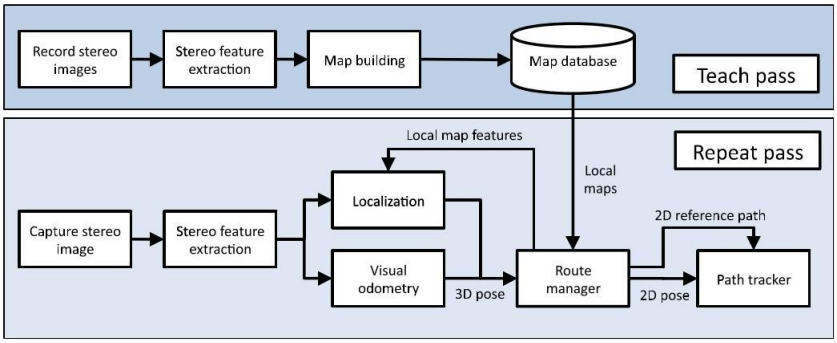


Figure 2.6: The foundational VT&R pipeline proposed by Barfoot et al. [79] showing the teach and repeat passes.

are robust to long term visual changes. Multi-view approaches are considered for improved robustness but require multiple cameras and still use SURF descriptors [128]. Krajník et al. [129] show that CNN and their proposed GRIEF descriptors perform better, but do not integrate them into a VT&R pipeline.

2.4.2 Image Sequence Matching

Full VT&R traditionally includes a control system seen in Figure 2.6 as the 'path tracker', but the problem can also be considered purely from a computer vision perspective and can be known as online place recognition or image sequence matching. An early approach, FAB-MAP 2.0 [76], uses SURF descriptors to create a bag of visual words representation from a set of training images. A Chow-Liu tree representation of feature likelihood is then used to determine the probability of loop closure, or of visiting a new location. Another approach [130] uses FAST corners [32] and BRIEF feature descriptors [74] but fails when presented with large visual changes.

OpenSeqSLAM 2.0 (OSS2) [132] outperforms FAB-MAP 2.0 by constructing a matrix of query and reference image descriptor comparisons. The matrix's contrast is enhanced and used to calculate the most likely trajectory. SeqSLAM is robust to seasonal changes [133], but struggles with mild viewpoint changes. Fast-SeqSLAM [134] extended this approach by introducing HOG image descriptors [86] and optimised the sequence matching. Recently SeqSLAM was modified to use CNN descriptors [135], which also increased performance. Online Place Recognition (OPR) [131] outperforms OSS2 by using lazy data association and CNN descriptors in a directed acyclic data association graph to track sequences online, instead of OSS2's offline approach. OPR is designed to work at variable speeds and can be used with HOG descriptors or CNN

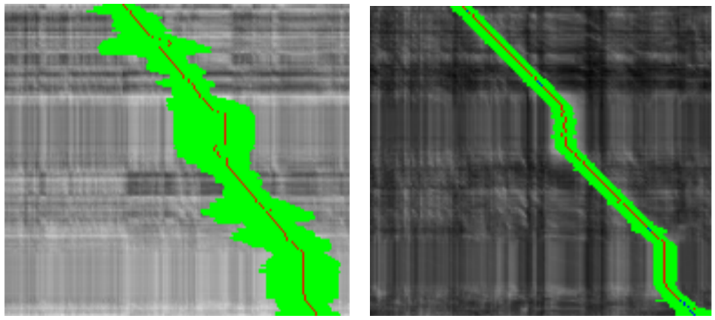


Figure 2.7: Descriptor comparison matrices showing HOG (left) vs. Overfeat CNN (right) descriptors being used for image sequence matching in online place recognition [131]. CNN descriptors are shown to be more accurate with a tighter spread of potential matches (shown in green). Variable speed inputs are also shown here, with vertical lines indicating a stationary camera. Duplicate of Figure 3.12

descriptors from the OverFeat CNN pre-trained on general image classification [136], an example of this can be seen in Figure 2.7.

2.5 Particle Filtering

One of the contributions of Chapter 3 is an implementation of a particle filter, this subsection provides a general introduction to the filter in the context of robotic SLAM and then provides a more in-depth mathematical derivation.

2.5.1 Overview

Sequential Monte Carlo algorithms use repeated random sampling to approximate the state of possible events that depend only on the state of previous events, also known as a Markov chain. Particle filters are a type of Sequential Monte Carlo algorithm that use Bayes' rule to estimate the posterior distribution of a system's states, given partial or noisy observations of previous states and was introduced for robotics by Sebastian Thrun [137, 138, 139], whose work provides the basis for the following mathematical explanation.

Suppose the state of the Markov chain at time, t is given by x_t. Furthermore, the state x_t depends on the previous state x_{t-1} according to the probabilistic law $p(x_t|u_t, x_{t-1})$, where u_t is the control asserted between $t-1$ and t. The Markov chain state is unknown, but can be estimated by measuring a stochastic projection z_t of the true state x_t via the probabilistic law $p(z_t|x_t)$. For example, a robot's position is un-

known but can be estimated given its sensor readings and recent movements.

In robotics, particularly SLAM, $p(x_t|u_t, x_{t-1})$, is usually referred to as an actuation or motion model, and $p(z_t|x_t)$ as a measurement model. Assume all available sensor measurements are given by $z^t = z_0, \ldots, z_t$ and controls $u^t = u_0, \ldots, u_t$. Given that the initial state is distributed according to some distribution $p(x_0)$, where $p(x_0) = p(x_0|z^0, u^0)$, the equation for the posterior distribution is given by Equation 2.3.

$$p\left(x_t \mid z^t, u^t\right) = \text{ const. } \cdot p\left(z_t \mid x_t\right) \int p\left(x_t \mid u_t, x_{t-1}\right) p\left(x_{t-1} \mid z^{t-1}, u^{t-1}\right) dx_{t-1} \quad (2.3)$$

To recover a posterior distribution over the state x_t at any time t, from all available sensor measurements $z^t = z_0, \ldots, z_t$ and controls $u^t = u_0, \ldots, u_t$ the posterior distribution can recursively estimated with incoming measurements and Bayesian statistics.

In robotics, particle filters are usually applied in continuous state spaces where closed form solutions for Equation 2.3 are only known for highly specialized cases. If the arguments are linear Equation 2.3 is equivalent to a Kalman filter [140, 141]. If the system actuation and measurements models aren't linear they can be approximately linearised, if this is done using a first order Taylor series expansion the result is an extended Kalman filter [141]. Particle filters address a more general Markov chain case by approximating the posterior of a set of M sample states $x_i^{[t]}$, or particles. Here each $x_i^{[t]}$ is a state sample associated with each particle's index from 1 to m. The most basic version of a particle filter is a two step process:

1. **Initialisation:** At time $t = 0$, draw a set X_0 of M particles according to $p(x_0)$.

2. **Recursion:** At time $t > 0$, generate a set $\bar{X}_t$ of particles $x_t^{[i]}$ for each particle $x_{t-1}^{[i]} \in X_{t-1}$ by drawing from the actuation model $p(x_t|u_t, x_{t-1}^{[i]})$. Draw M particles from $\bar{X}_t$, so that each $x_t^{[i]} \in \bar{X}_t$ is drawn (with replacement) with a probability proportional to $p(z_t|x_t^{[i]})$. Call the resulting set of particles X_t.

As $M \to \infty$, this recursive procedure leads to particle sets X_t that converge uniformly to the desired posterior $p(x_t|z^t, u^t)$. Particle filter computation time depends on the number of particles in the filter and are well suited to being implemented in parallel. Linearisation that would be needed for Kalman filters is also not required. However, particle filters do not work so well for large dimensional spaces as a large

number of particles are needed to populate that space. Chapter 3 applies a particle filter to VT&R that circumvents this problem by constraining the search space to a reference database of images, a related approach is taken by Xu et al. [142].

2.5.2 Mathematical Derivation

For completeness a mathematical derivation of the particle filter is presented here for Equation 2.3 and the weight calculation (Equation 3.5) described in Section 3.2.2. The particle filter is a Bayesian filter, which performs state estimation by combining a measurement model with a prior probability using Bayes' theorem. For a general Bayesian filter consider a non-linear stochastic system defined by a stochastic discrete-time state space dynamic equation (Equation 2.4) and the stochastic observation process (Equation 2.5), where at time t, the unknown state vector is $\mathbf{x}_t$, the dynamic noise vector is $\mathbf{w}_t$, the observation vector is $\mathbf{y}_t$ and the observation noise vector is $\mathbf{v}_t$. Functions $\mathbf{f}_t$ and $\mathbf{h}_t$ respectively relate the prior state to the current state and the current state to the observation vector.

$$\mathbf{x}_t = \mathbf{f}_t\left(\mathbf{x}_{t-1}, \mathbf{w}_{t-1}\right) \tag{2.4}$$

$$\mathbf{y}_t = \mathbf{h}_t\left(\mathbf{x}_t, \mathbf{v}_t\right)) \tag{2.5}$$

A Bayesian filter estimates, $p\left(\mathbf{x}_t \mid \mathbf{y}_{1:t}\right)$, the posterior probability density function (PDF) using the observations defined by $\mathbf{y}_{1:t} \triangleq \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_t\}$. The non-linear, non-Gaussian, state-space model in Equation 2.4 specifies the predictive conditional transition density, $p\left(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{y}_{1:t-1}\right)$, of the current state given all previous state observations. Also, the observation process in Equation 2.5 specifies the likelihood function of the current observation given the current state, $p\left(\mathbf{y}_t \mid \mathbf{x}_t\right)$. The prior probability, $p\left(\mathbf{x}_t \mid \mathbf{y}_{1:t-1}\right)$, is defined by Bayes' rule in Equation 2.6, where the previous posterior PDF is $p\left(\mathbf{x}_{t-1} \mid \mathbf{y}_{1:t-1}\right)$. The measurement model can be used to generate the PDF, $p\left(\mathbf{x}_t \mid \mathbf{y}_{1:t}\right)$, in Equation 2.7 which relates directly to Equation 2.3, where c is a normalisation constant.

$$p\left(\mathbf{x}_t \mid \mathbf{y}_{1:t-1}\right) = \int p\left(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{y}_{1:t-1}\right) p\left(\mathbf{x}_{t-1} \mid \mathbf{y}_{1:t-1}\right) d\mathbf{x}_{t-1} \tag{2.6}$$

$$p\left(\mathbf{x}_t \mid \mathbf{y}_{1:t}\right) = cp\left(\mathbf{y}_t \mid \mathbf{x}_t\right) p\left(\mathbf{x}_t \mid \mathbf{y}_{1:t-1}\right) \tag{2.7}$$

The filtering problem is to recursively estimate $\mathbf{x}_t$, given $\mathbf{y}_{1:t}$. For a general distribution, $p(\mathbf{x})$, this consists of the recursive estimation of the expected value of any function of $\mathbf{x}$, such as $\langle g(\mathbf{x})\rangle_{p(\mathbf{x})}$, using Equations 2.6 and 2.7. However, this requires calculation of integrals of the form shown in equation 2.8, but these cannot be evaluated in closed form so must be approximated.

$$\langle g(\mathbf{x})\rangle_{p(\mathbf{x})} = \int g(\mathbf{x}) p(\mathbf{x}) d\mathbf{x} \tag{2.8}$$

In many cases, $p\left(\mathbf{x}_t \mid \mathbf{y}_{1:t}\right)$ may be a multivariate, or multi-modal PDF and is therefore difficult to generate samples from to approximate the distribution. To overcome this difficulty Importance Sampling can be used. Suppose that $q\left(\mathbf{x}_t \mid \mathbf{y}_{1:t}\right)$ is another PDF from which samples can be easily drawn, this is known as the Importance Density and could be an analytical Gaussian PDF. Let $p\left(\mathbf{x}_t \mid \mathbf{y}_{1:t}\right) \propto q\left(\mathbf{x}_t \mid \mathbf{y}_{1:t}\right)$, where the symbol $\propto$ means that $p\left(\mathbf{x}_t \mid \mathbf{y}_{1:t}\right)$ is proportional to $q\left(\mathbf{x}_t \mid \mathbf{y}_{1:t}\right)$ at every $\mathbf{x}_t$.

Since $p\left(\mathbf{x}_t \mid \mathbf{y}_{1:t}\right)$ is a normalized PDF, then $q\left(\mathbf{x}_t \mid \mathbf{y}_{1:t}\right)$ must be a scaled and unnormalised version of $p\left(\mathbf{x}_t \mid \mathbf{y}_{1:t}\right)$ with a different scaling factor at each $\mathbf{x}_t$. Therefore the scaling factor, or weight, can be written as shown in Equation 2.9.

$$w\left(\mathbf{x}_t\right) = \frac{p\left(\mathbf{x}_t \mid \mathbf{y}_{1:t}\right)}{q\left(\mathbf{x}_t \mid \mathbf{y}_{1:t}\right)} \tag{2.9}$$

To estimate the weights recursively Equation 2.9 can be re-written using Equation 2.7 as Equation 2.10.

$$w\left(\mathbf{x}_t\right) = \frac{cp\left(\mathbf{y}_t \mid \mathbf{x}_t\right) p\left(\mathbf{x}_t \mid \mathbf{y}_{1:t-1}\right)}{q\left(\mathbf{x}_t \mid \mathbf{y}_{1:t}\right)} \tag{2.10}$$

Using the expansion of $p\left(\mathbf{x}_t \mid \mathbf{y}_{1:t-1}\right)$ from Equation 2.6 and a similar expansion for $q\left(\mathbf{x}_t \mid \mathbf{y}_{1:t}\right)$ allows Equation 2.10 to be expanded into Equation 2.11:

$$w\left(\mathbf{x}_t\right) = \frac{cp\left(\mathbf{y}_t \mid \mathbf{x}_t\right) \int p\left(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{y}_{1:t-1}\right) p\left(\mathbf{x}_{t-1} \mid \mathbf{y}_{1:t-1}\right) d\mathbf{x}_{t-1}}{\int q\left(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{y}_{1:t-1}\right) q\left(\mathbf{x}_{t-1} \mid \mathbf{y}_{1:t-1}\right) d\mathbf{x}_{t-1}} \tag{2.11}$$

As mentioned previously these integrals must be approximated. To do this a set of M particles (random samples from the distribution) and weights $\left\{\mathbf{x}_{t-1|t-1}^{(i)}, w_{t-1}^{(i)}\right\}_{i=1}^{M_s}$ are defined to characterise the posterior PDF up until $t-1$. The previous posterior

PDF, $p\left(\mathbf{x}_{t-1} \mid \mathbf{y}_{1:t-1}\right)$, can therefore be approximated by Equation 2.12.

$$p\left(\mathbf{x}_{t-1} \mid \mathbf{y}_{1:t-1}\right) \approx \sum_{i=1}^{M_s} w_{t-1}^{(i)} \delta\left(\mathbf{x}_{t-1} - \mathbf{x}_{t-1|t-1}^{(i)}\right) \tag{2.12}$$

If the particles $\mathbf{x}_{t-1|t-1}$ were drawn from the importance density, $q\left(\mathbf{x}_{t-1} \mid \mathbf{y}_{1:t-1}\right)$ Equation 2.9 can be rewritten as Equation 2.13.

$$w_{t-1}^{(i)} = \frac{p\left(\mathbf{x}_{t-1|t-1}^{(i)}\right)}{q\left(\mathbf{x}_{t-1|t-1}^{(i)}\right)} \tag{2.13}$$

As the weights are being estimated recursively sequential importance sampling (SIS) [143] is used. At each iteration the random measure $\left\{\mathbf{x}_{t-1|t-1}^{(i)}, w_{t-1}^{(i)}\right\}_{i=1}^{M_s}$ constitutes an approximation to $p\left(\mathbf{x}_{t-1} \mid \mathbf{y}_{1:t-1}\right)$ and is used to approximate $p\left(\mathbf{x} \mid \mathbf{y}_{1:t-1}\right)$ with a new set of samples and weights. By substituting Equation 2.12 in Equation 2.11, and using a similar formulation for $q\left(\mathbf{x}_{t-1} \mid \mathbf{y}_{1:t-1}\right)$, the weight update for each particle becomes Equation 2.14.

$$\begin{aligned} w_t^{(i)} &\propto \frac{p\left(\mathbf{y}_t \mid \mathbf{x}_{t|t-1}^{(i)}\right) p\left(\mathbf{x}_{t|t-1}^{(i)} \mid \mathbf{x}_{t-1|t-1}^{(i)}, \mathbf{y}_{1:t-1}\right) p\left(\mathbf{x}_{t-1|t-1}^{(i)}\right)}{q\left(\mathbf{x}_{t|t-1}^{(i)} \mid \mathbf{x}_{t-1|t-1}^{(i)}, \mathbf{y}_{1:t-1}\right) q\left(\mathbf{x}_{t-1|t-1}^{(i)}\right)} \\ &= w_{t-1}^{(i)} \frac{p\left(\mathbf{y}_t \mid \mathbf{x}_{t|t-1}^{(i)}\right) p\left(\mathbf{x}_{t|t-1}^{(i)} \mid \mathbf{x}_{t-1|t-1}^{(i)}\right)}{q\left(\mathbf{x}_{t|t-1}^{(i)} \mid \mathbf{x}_{t-1|t-1}^{(i)}\right)}. \end{aligned} \tag{2.14}$$

Chapter 3's approach specifically uses the bootstrap particle filter [144] to calculate particle weights (Equation 3.5) which makes an approximation that the importance density is equal to the prior density $p\left(\mathbf{x}_{t-1|t-1}^{(i)}\right) = q\left(\mathbf{x}_{t-1|t-1}^{(i)}\right)$ to cancel out two PDFs and simplify the weight update to Equation 2.15, where $\mathbf{x}_{t|t-1}^{(i)} = \mathbf{f}_t\left(\mathbf{x}_{t-1|t-1}^{(i)}, \mathbf{w}_{t-1}^{(i)}\right)$, from Equation 2.4.

$$w_t^{(i)} = w_{t-1}^{(i)} p\left(\mathbf{y}_t \mid \mathbf{x}_{t|t-1}^{(i)}\right) \tag{2.15}$$

The posterior filtered PDF $p\left(\mathbf{x}_t \mid \mathbf{y}_{1:t}\right)$ can then be approximated by Equation 2.16, where the updated weights are generated recursively using Equation 2.15.

$$p\left(\mathbf{x}_t \mid \mathbf{y}_{1:t}\right) \approx \sum_{i=1}^{M_s} w_t^{(i)} \delta\left(\mathbf{x}_t - \mathbf{x}_{t|t}^{(i)}\right) \tag{2.16}$$

2.5.3 Visual SLAM

Particle filters can be used by SLAM algorithms to estimate the position of a robot from noisy sensor observations, examples of this are proposed by Karkus et al. [145] and FastSLAM 2.0 [146]. Brubaker et al. [147] use a particle filter to identify noise in visual odometry using the KITTI dataset [148].

2.5.4 Visual Place Recognition

One of the earliest attempts at probabilistic VPR uses Bayesian filtering with Markov chains to match sequences of indoor image signatures that it viewed as 'words' in a sentence of places [149]. This approach was remarkably ahead of its time but was limited to indoor use and the Gist [87] descriptors which are much older than state-of-the-art VPR descriptors.

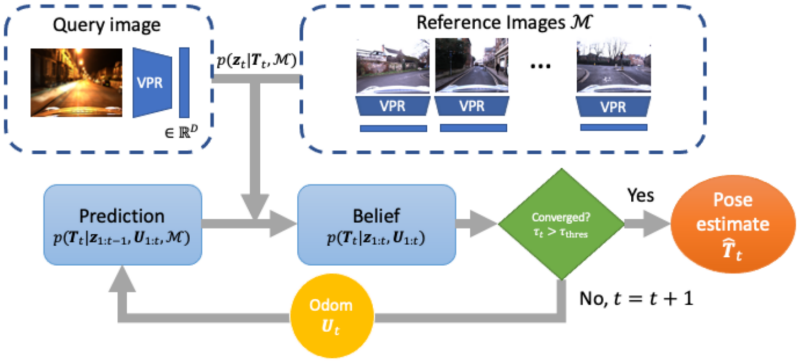


Figure 2.8: Particle filter pipeline used for VPR [142]. The belief state is recursively updated using given query images and odometry data, if the particle filter converges a pose estimate is predicted.

The introduction of deep learnt VPR descriptors allowed Xu et al. [142] to compare consecutive query images with a single reference database of images and filter the best matches of each result using a particle filter which converges to a single localisation estimate on the longer and more visually challenging Oxford dataset [115] (Figure 2.8). Combining deep learning with traditional particle filters allows unreliable predictions to be removed that cause problems in practical applications. A similar approach uses a directed graph to deal with confusing intersections and junctions represented in the dataset [150]. Probabilistic filtering has also been used for aerial navigation [151].

2.6 Scene Classification

Scene classification is heavily related to VPR, but as a classification task the main difference is that classes are defined at training time, which is not the case for image retrieval. Scene classification classifies a scene image into a pre-defined scene category and has applications for content based image retrieval [152], robot navigation [153] [154] and disaster detection [155]. Scene classification is a well-defined area of computer vision research with a wide range of approaches described in comprehensive survey papers [19, 20], an overview is presented here. As in many other computer vision tasks, CNNs have recently been used to increase classification accuracy. There are five broad categories of CNN-based approaches for state-of-the-art scene classification [19].

Figure 2.9: Two samples per category of the 365 scene classes presented by [156], showing three macro classes: Indoor, Nature and Urban.

1. *Global* descriptors typically use a generic CNN trained on a task such as object detection which is then fine-tuned for scene classification. Zhou et al. [156] do exactly this on 365 scene classes, samples of which can be seen in Figure 2.9. Global descriptors contain spatial correlations between objects and global scene properties and therefore provide enriched spatial information but are more vulnerable to background noise [157].

2. *Spatially* invariant descriptors are usually extracted from multiple local patches [158] using VLAD or Fisher encoding [159]. These approaches are an efficient way to achieve geometric robustness, but perform poorly when a scene includes objects with variable sizes or aspect ratios.

3. *Semantic* descriptor approaches based on object detection aim to identify salient regions of the scene, which provide distinctive information about a scene's contexts [160]. However, the lack of detailed scene labels and the computational

requirements of searching for salient regions [161] limits development of these approaches.

4. *Multi-layer* descriptors combine different resolution descriptors from different CNN layers [162]. A specific example of this approach is DAG-CNN [163] which integrates descriptors from different levels of a CNN in a directed acrylic graph. Feature fusion is necessary for this approach and is usually done early or late. Early fusion extracts multi-layer descriptors and merges them into a comprehensive descriptor whereas late fusion directly uses supervised learning to ensure the multi-layer descriptors are sensitive to each target scene class [164].

5. *Multi-view* descriptors (from multiple complementary CNN models trained on different datasets) can be used to create comprehensive scene representations. For example, FOSNet [165] introduces scene coherence loss to fuse object and scene data while Sun et al. [166] separately extracts three complimentary representations using object semantics, contextual information and global appearance. However, all current fusion approaches generally fuse information from networks trained for classification.

Some scene classes described in this manuscript overlap with autonomous driving events, such as stopping at a junction [167]. However, events usually detected such as lane changing, overtaking and rear-ending [168] are not directly linked to the scene classes considered in this work.

2.7 Deep Learning on Embedded Hardware

Many of the approaches already highlighted in this section use deep learning, but there are some remaining areas of research within the field of deep learning itself that are relevant to this work.

2.7.1 Computational Requirements

2.7.1.1 Specialised Processing Units (SPUs)

A significant reason for the limited deployment of deep neural networks (DNNs) on embedded hardware is the associated computational requirement. SPUs such as Tensor Processing Units (TPUs) have recently been introduced to allow more efficient

data-center implementations of hardware dedicated to deep learning [169]. These innovations have also enabled the development of embedded hardware capable of running DNNs [170]. A consumer solution for this is Google's Coral development board [2] which allows for rapid prototyping of different deep learning solutions. Specialised hardware for running CNNs includes Intel's Deep Learning Processing Unit (DPU) [171]. This technology is critical for enabling deep learning for embedded applications such as robotics, but remains uncommon.

2.7.1.2 Compression

DNN compression is the reduction in capacity of the network while retaining the original task performance [172]. Compression of CNNs is particularly important as they have the most obvious opportunities for embedded application. Some CNN architectures have been specifically designed for efficiency, such as MobileNets [118] which use depthwise separable convolutions and DenseNet [173] which connects each layer to every other layer in a feed-forward fashion. A number of compression techniques have been proposed.

1. *Weight Sharing* between layers or structures, such as CNN filters reduces the size of a network. One approach to do this is to bin the CNN weights and use them for re-training [174].

2. *Pruning* low priority network weights allows direct compression. The simplest strategy uses a threshold to decide which weights should be removed [175].

3. *Tensor Decomposition* of weight tensors into a lower rank approximation enables compression. For CNNs this can be performed filter-wise [176], channel-wise [177] or a combination of the two [178].

4. *Knowledge Distillation* uses a large pre-trained network to train a smaller network on the same task and to minimise the difference between their predictions [179].

5. *Quantization* compresses neural networks by representing network values with fewer bits [180].

[2] https://coral.ai/products/dev-board

2.8 Open Set Recognition (OSR)

The task of OSR for image classification is to recognise test images that have not been defined during training as members of an open set class. This is a critical problem for deploying deep learning approaches in the real world where they will often be presented with open class data, but has rarely been addressed for image classification, let alone image retrieval. An open set class is sometimes described as being out-of-distribution with respect to the known classes. This is an emerging and challenging area of research [181] that is largely limited to simple computer vision datasets such as CIFAR-10 [182] and MNIST [183], although more recent work [21] addresses the more complex ImageNet dataset [106]. Geng et al. [181] identify approaches to OSR as generative, or discriminative.

2.8.1 Discriminative

The majority of deep neural networks trained for classification use a typical softmax cross-entropy loss which is inherently closed set because of its normalised output. Discriminative approaches revolve around finding an empirical threshold that can be used to either reject or categorize the input samples as an unknown members of an 'open' class [184], the disadvantage of this is that it requires knowledge of a validation set to be chosen optimally.

Similarity Learning uses losses such as lifted [185], focal [186] and range [187] to cluster known classes together in an embedded space, which then allows outlier classes to be identified. Active learning has also been explored with similarity learning [188] to detect open set inputs and label them for addition to training data.

Calibration can be applied to a DNN to improve the accuracy of the confidence of its predictions. For example, an Openmax calibration layer [21], explained further in Chapter 4, is trained normally using cross-entropy, then each class is represented as a mean activation vector (MAV) based on correctly classified samples and fitted to a separate Weibull distribution and used to compute a pseudo-activation for unknown classes. Once the class embeddings are fitted to the statistical distribution it can be used to estimate whether test image embeddings are likely to be outliers. Liang et

al. [189] propose ODIN to use temperature scaling and small input perturbations to separate the softmax score distributions of closed and open set image classes.

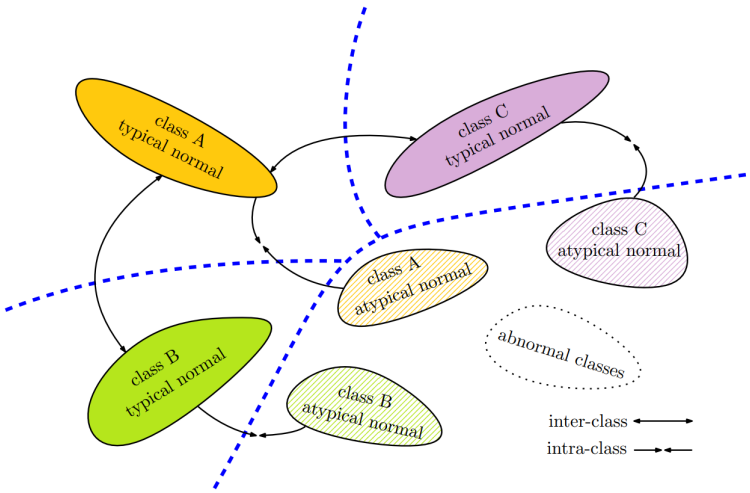


Figure 2.10: Intra-class splitting [190] for open set recognition trains for classification on the original classes (A, B and C). The training data is evaluated on the trained classifier and examples which are misclassified or classified with a low confidence are identifed as atypical examples of the original data (the hatched classes A, B and C). This atypical data is used as synthetic examples of the open class, which then allows the network to be retrained for open set recognition.

2.8.2 Generative

Generative methods attempt to synthesize likely examples of unknown classes and then learn to classify them as an additional open class.

Generative Adversarial Networks are a common choice for this approach. For example, OpenGAN [22] generates synthetic open set data to train a discriminator for open set recognition. Openmax [191] has also been extended by using a generative adversarial network (GAN) to synthesize unknown classes, which can be investigated visually, and then explicitly estimate probability over them. Encoder-decoder GANs are also used to augment a training dataset to include synthetic open set images that lie on the opposing side of the true decision boundary, between the known classes and the open set [192].

In contrast, Vaze et al. [193] improve training of a closed set classifier with data augmentation, longer training and ensembling, this is combined with maximum logit scoring and demonstrates competitive open set recognition performance compared to the state-of-the-art. Recently Schlachter et al. [190] suggested intra-class splitting

(Figure 2.10) which trains a classifier on the closed set problem. Unlearnt training data and correctly learnt training data that is classified with a low confidence threshold is used to generate potential open set data by relabelling it as an open set for re-training the network for OSR.

Long-Tailed open set recognition scenarios combine the problem of imbalanced training class samples with OSR. Liu et al. [194] introduces this problem and addresses it using a meta embedding to combine a feature descriptor, a memory feature which captures visual concepts from training classes and a feature that relates the feature descriptor to the other training classes to transfer knowledge between class representations and enable open set recognition. Cai et al. [195] expands on this approach by proposing a distribution-sensitive loss, to increase the weight of tail classes and decrease the intra-class distance in the descriptor space, and a local-density-based similarity to measure the novelty of a testing sample.

Bayesian deep learning [196], although not explicitly labelled OSR, as has been used to predict uncertainty for visual place recognition. Bayesian deep learning can be used indirectly for OSR by identifying open set classes with a high degree of epistemic uncertainty [197] or unifying a Dirichlet process mixture model with a DNN [198]. Warburg et al. [199] propose Bayesian triplet loss to train a DNN to produce stochastic descriptors rather than point estimates. By considering a posterior distribution over possible features direct uncertainty estimates can be calculated and used to assign probabilities to descriptor matches. This approach also uses estimated uncertainty for out of distribution detection, but does not aim for OSR.

2.9 Summary

The introduction of deep learning, in particular CNNs, has enabled significant advances in generating appearance invariant image descriptors but these are yet to have found practical application in robotics. Two reasons for this is that current approaches rarely consider the practical challenges of prediction uncertainty and open set recognition which are critical for use in real-world applications. Furthermore these tasks have typically only been considered for image classification rather than visual place recognition, which is an image retrieval task. To address these problems from a robotics

perspective this thesis considers a wide variety of relevant research topics and uses them to make three contributions towards improved navigation in open and uncertain environments.

Part II

Contributions

Chapter 3

Particle Filtering for Robust Real-Time Visual Teach and Repeat

3.1 Introduction

Visual SLAM has many important applications, but for real-world tasks navigation along a pre-defined route is often sufficient, this task is a subset of visual SLAM known as visual teach and repeat (VT&R). In this case a route is defined as a single path between two geographic points. This chapter addresses two key challenges for state-of-the-art VT&R:

- Appearance invariant visual place recognition descriptors specialised for each route that can be generated faster on embedded hardware.
- Real-time route navigation that is robust to variations in speed, and large visual changes.

Visual teach and repeat consists of a 'teach' phase that pre-records reference traversals of a route and then a 'repeat' phase for real-time, autonomous repetition of that same route with variations in appearance (Figure 3.1). VT&R compares single test images from a route repetition with the reference images using image descriptors. The closest match from this comparison is used to localise the query image and therefore relies heavily on those image descriptors being appearance invariant. Current approaches OpenSeqSLAM 2.0 (OSS2) [132] and Online Place Recognition (OPR) [131] use patch-normalised descriptors and deep neural network (DNN) descriptors taught

on general image classification [136] respectively which have not been taught for appearance invariant VPR. Recently large DNNs have been taught to generate appearance invariant VPR descriptors that generalise across many environments and visual conditions [112, 7], but these are not specialised for a unique route. The smaller the DNN required, the lower the computational requirements and the better it is for mobile robotics, which prioritises lower power usage and inference time.

Figure 3.1: Examples of the reference repetition traversals used from the Oxford and UAH datasets showing the variations in appearance.

For route repetition OSS2 [132] and OPR [131] compensate for descriptors not designed for the task by making associations between sequences of consecutive images, but these fail during real-time operation, which is an integral aspect of real-world, outdoor navigation. Two significant causes of failure in these algorithms are repetition of the route at variable speed and large visual changes. Variable speed repetition causes query images to be unevenly sampled along the route (Figure 3.2), which disrupts the sequences of images that OSS2 and OPR depend on for localisation during repetition.

Large visual changes impact the ability of an accurate VPR descriptor to be generated and therefore also disrupts the comparisons between sequences of consecutive place images, examples of this can be seen in Figure 3.3.

Figure 3.2: Example of the difference between comparing reference images with test traversals at constant or variable speeds using the Oxford RobotCar Dataset [115]. Variable speeds result in an uneven distribution of test images across the route which makes them more challenging to localise, especially for approaches that explicitly use sequential information.

This chapter's first contribution uses reference traversals of a single route collected in the teaching phase of VT&R to train a compact TinyVPR network with reduced inference time to generate appearance invariant VPR descriptors. The different reference traversals represent the single route in a number of appearance variations and are used to train TinyVPR to generate VPR descriptors that generalise to unseen test appearance variations, but are also specialised for that route for reduced localisation error.

Figure 3.3: Examples of the large visual changes that can disrupt descriptor matching in the Oxford Robotcar Dataset [115] and the UAH-DriveSet [200] highlighted in red. Examples include camera occlusion, construction work altering building appearance and extreme light changes.

Experiments show TinyVPR generates descriptors 11x faster with 30x fewer parameters than state-of-the-art visual place recognition network NetVLAD [10] and reduces localisation error for route repetition between 32.3% and 4.3% on two challenging outdoor datasets: Oxford RobotCar [115] and UAH-DriveSet [200] respectively.

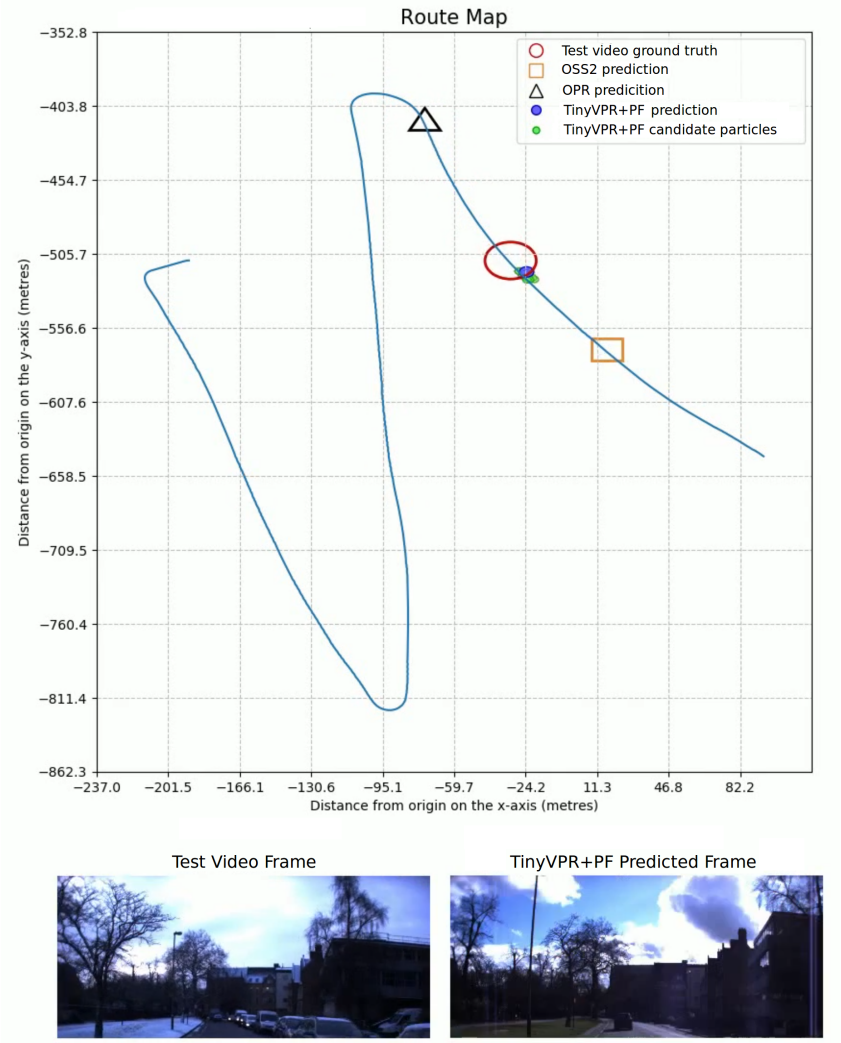


Figure 3.4: Visualisation of the proposed particle filter (with 30 particles) and TinyVPR descriptor localising during a route repetition when alternative approaches OSS2 [132] and OPR [131] fail on the Oxford RobotCar dataset [115] using a test repetition traversal in Snow conditions.

The second contribution is a novel combination of TinyVPR descriptors and accumulated visual odometry in a particle filter (Figure 3.4) to remove incorrect matches that occur when a test image is being compared to the reference traversal images for localisation. Accumulated visual odometry is used to keep track of the approximate progress along the route, if used for VT&R alone VO would drift but it can be used to help reject unlikely descriptor matches. For example, if visual odometry indicates the

robot has moved 20m along a route but a couple of descriptor comparisons suggest the robot is localised 100m along the route then these matches can be identified as outliers. The TinyVPR descriptor comparisons are generally reliable, but the particle filter is able to remove noisy matches affected by large visual changes. Other approaches use sequences of data to address this problem but these are disrupted when the camera is traversing the route at variable speed. The particle filter still uses a distribution of previous localisation predictions but does not require fixed sequences of data which are unavailable during variable speed repetition. Experiments show the proposed approach reduces localisation error by 94.3% and 99.1% compared to OSS2 and OPR on the Oxford RobotCar [115] and UAH-DriveSet [200] datasets respectively.

1. A training approach and TinyVPR network with reduced parameters to generate appearance invariant descriptors for a single route (Section 3.2.1).

2. A particle filter to combine TinyVPR embedded descriptors with accumulated visual odometry for route repetition that is robust to variable repetition speed and large visual changes (Section 3.2.2).

3.2 Methodology

In this section the network architecture and training scheme for generating the proposed TinyVPR descriptors is described. Secondly, a particle filter for route repetition at variable speeds and with large visual changes is proposed.

3.2.1 TinyVPR

3.2.1.1 Basic Hypothesis

State-of-the-art approaches use large datasets and deep neural networks to provide a general solution for generating appearance invariant VPR descriptors. However, the large capacity of these networks increases their computational requirements and the lack of route-specific training data does not ensure the descriptors are best suited for the target environment. This section explores the possibility of leveraging the reference traversal data collected in the teach phase of visual teach and repeat to train a compact DNN architecture to generate specialised VPR descriptors for visual teach and repeat on a per route basis.

3.2.1.2 Teach Phase & Data Pre-Processing

The teaching phase of visual teach and repeat collects different traversals of a route in different appearances throughout different times of the day and weather conditions, amongst others. Images from repetitions of the route can then be localised by comparing them to the reference traversals to enable autonomous navigation. Reference traversals are recorded as videos which can be separated into geotagged images (using simultaneously recorded GPS data) with associated geographical positions and cover different appearance conditions (Figure 3.1). It is proposed that this data is used to train a compact network to generate a VPR descriptor specialised on a per-route basis.

For each dataset the geotagged images from one reference route traversal were sampled at dataset-specific distance intervals (2.25m for Oxford and 18m for UAH) or at camera rotations of more than 10 degrees. These intervals were chosen to reduce the number of the frames in the dataset without compromising the coverage of the route, including the rapid visual changes from turning corners. For each sampled frame of the reference route a matching frame was found in each of the remaining reference routes (Figure 3.5). The reference route data is sourced from different videos associated with different visual variations so the data can be used for training and as the reference database for comparing test images against for VT&R. Synchronisation also helps to ensure that every image has an equivalent with a visual variation available for triplet training, as described in Section 3.2.1.4. Test images from previously unseen routes could be synchronised against the reference images, or only at a time interval, to reduce the number of frames from the original video but retain its real-time characteristics, such as pauses in traffic. Both reference and test images were cropped to remove potentially misleading visual information, such as the front of the recording vehicle.

3.2.1.3 Network Architecture

NetVLAD [10] is a state-of-the-art approach for generating appearance invariant VPR descriptors. NetVLAD is based on the VGG16[12] convolutional neural network pre-trained on general image classification using the ImageNet dataset frozen down to, but not including, the Conv5 layer. The network is cropped at the final convolutional layer and a custom VLAD layer is appended. The network is then trained using a version of similarity learning called triplet loss and semi-hard triplet mining. The

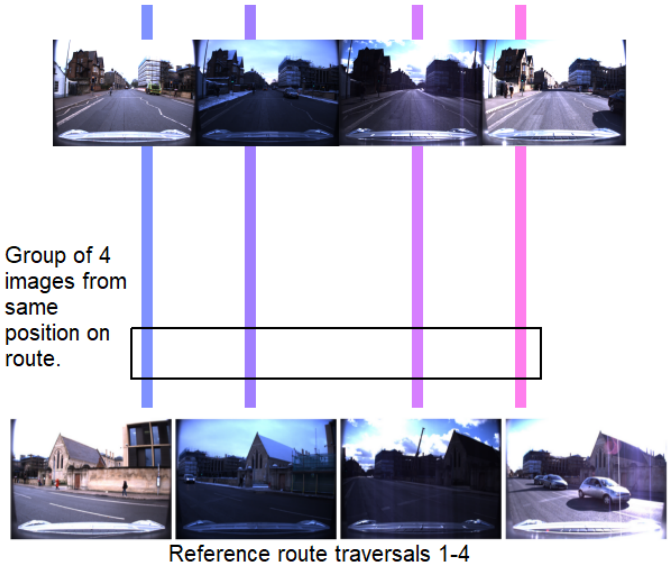


Figure 3.5: Example of the Oxford dataset sampled and synchronised across four reference traversals showing the four examples of each single place.

training images are sourced from large, feature-laden city environments at minimum intervals of 12 metres and in other environments at discrete locations.

This manuscript proposes a compact alternative to NetVLAD called TinyVPR which uses MobileNetV1 [118] as the base CNN and replaces the final layer with global averaging pooling and two fully connected layers. MobileNetV1 was originally trained for the same image classification task as VGG16, but is approximately 30x smaller and 10x faster. The biggest problem with training on a per route-basis is overfitting to the training appearance variations, dropout (ignoring a proportion of the output neurons during training) and L2 normalisation (normalising the inputs to sum to 1) were used to help prevent this (Figure 3.6). These final layers are inspired by FaceNet [201] which uses similarity learning to cluster face images also using triplet loss. MobileNetV1 is frozen and the remaining layers are trained with triplet loss. TinyVPR significantly reduces network parameters and consequently training and inference time.

3.2.1.4 Triplet Learning

Triplet learning (a variation of similarity learning introduced in Section 2.2) is used in a similar manner to NetVLAD to train TinyVPR to generate appearance invariant VPR descriptors using the multiple reference traversals recorded during the VT&R teaching phase. Specifically, the triplet loss function (Equation (3.1)) is minimised for each batch of triplets (Equation (3.2)) to:

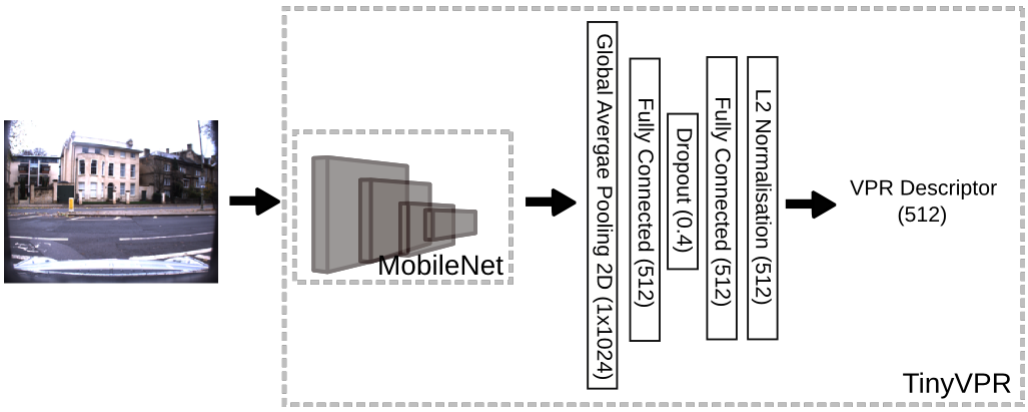


Figure 3.6: TinyVPR CNN network architecture with the output shape of each layer, the L2 normalization factor was 0.1. The architecture for TinyVPR and values for dropout were determined heuristically, but with reference to FaceNet's architecture [201]. The architecture took inspiration from FaceNet, specifically a different backbone network designed for mobile applications.

1. Minimize the Euclidean distance between the VPR descriptors of a randomly selected 'anchor' image and nearby 'positive' image, but with different appearances (Figure 3.7). This teaches the network that, despite some appearance variations, two images are of the same place. Some difference between the positive images in terms of distance and rotation can be beneficial as it helps to encourage viewpoint invariance, but a significant overlap is necessary.

2. Maximize the Euclidean distance between the VPR descriptors of the above anchor and 'negative' images of places from far away and with any example appearance (Figure 3.7). This teaches the network that other appearance variations indicate two images are from different places.

For training the network a siamese configuration is used: three instances of TinyVPR are used to generate the descriptors for each of the positive, anchor and negative images in preparation for calculating the triplet loss between them. The weights of each network instance are shared so after training one network instance can be used for VPR descriptor generation, given an input image, as shown in Figure 3.6.

$$L(a,p,n) = max(||f(a) - f(p)||^2 - ||f(a) - f(n)||^2 + m, 0) \tag{3.1}$$

where:

a = random anchor place image

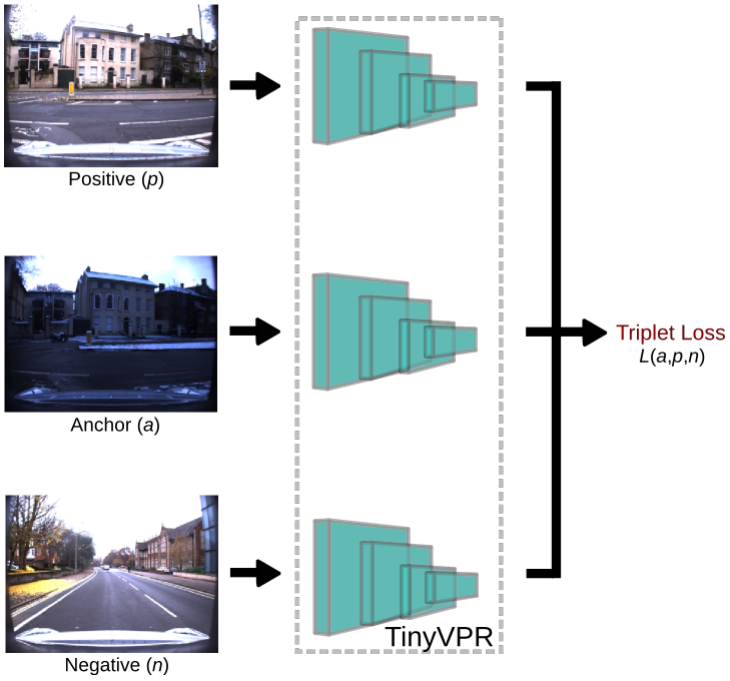


Figure 3.7: Example of an image triplet and the siamese network setup used for training. Three instances of TinyVPR are shown here with shared weights and used to produce the descriptor representations for the anchor, positive and negative images which can then be minimised in the triplet loss.

p = positive place image (appearance variation of anchor place image)

n = negative image (different place with random appearance variation)

m = margin between positive and negative pairs

f = network output (descriptor)

Note Equation (3.1) uses the $max()$ function to set $L(a, p, n) = 0$ if the distance between the anchor and negative is larger than the distance between the anchor and the positive because this is already the desired output. The cost function to **minimise** for each triplet batch is therefore:

$$cost = \sum_{i}^{batchsize} L(a^i, p^i, n^i) \tag{3.2}$$

3.2.1.5 Triplet Mining

The strategy for selecting appropriate triplets of images for training is called 'mining' and has been shown to have a significant affect on network performance [202]. For visual place recognition triplets are traditionally mined using their geographical

positions, for example NetVLAD defines positive images as images within a boundary of 10 meters of the anchor, and negative images as further away than a 25 meter boundary.

Alternatively, consecutive frames could be used for triplet mining, for example positive images can be defined as being within 10 frames of the anchor and negative images more than 25 frames away. This frame-wise mining took advantage of the pre-processing described in Section 3.2.1.2 to prevent anchor images at corners being associated with positive images that did not look similar but were recorded nearby. For example, using geographical mining an anchor image may have a positive image mined from one meter away that looks very different because it was taken while turning a sharp corner, but because the dataset is collected at distance and rotation intervals frame-wise mining a positive image from one frame away would ensure that the positive image is similar and maintain the relationship between the triplet of training images shown in Figure 3.7. For initial selection a total of three approaches are compared for training TinyVPR:

1. **Geographical**: positive images lie within X metres and negative images beyond X metres.

2. **Frame-wise**: positive images lie within X consecutive frames of the anchor, while negative images are further than X consecutive frames away.

3. **Hybrid**: positive images lie within X metres of the anchor and negative images are further than X consecutive frames away.

Combining frame-wise and geographical approaches in a hybrid method allows more overlap between positive and negative images, which may be useful for generating more difficult triplets for learning. Triplet difficulty for training a network falls into three categories:

1. **Easy** triplets which have a loss of 0, because: $||f(a)-f(p)||^2 \leqq ||f(a)-f(n)||^2+m$.

2. **Semi-hard** triplets where the negative is not closer to the anchor than the positive, but which still have positive loss: $||f(a) - f(p)||^2 + m > ||f(a) - f(n)||^2$.

3. **Hard** triplets where the negative is closer to the anchor than the positive: $||f(a)-f(p)||^2 > ||f(a) - f(n)||^2$.

To ensure efficient training easy triplets can be removed as they do not contribute towards the network learning, but a triplet needs to be passed through the network and its loss calculated before its difficulty is known, which can be time consuming and increase computational time. Ultimately including easy triplets for training does not change the result because they return 0 loss (Equation (3.1)). Different triplet mining strategies and frame/distance boundaries for positive and negative images were explored for training TinyVPR, the results of which can be seen in Section 3.3.2:

3.2.2 Particle Filter

3.2.2.1 Basic Hypothesis

State-of-the-art approaches to visual teach and repeat make associations between fixed sequences of image descriptors to enable accurate route repetition. However, in real-time operation these sequences are disrupted by variable speed route repetition and large visual changes, such as occlusions which causes current approaches to fail. The hypothesis of this chapter is that appearance invariant VPR descriptors can be combined with accumulated visual odometry in a probabilistic filter to prevent these failure cases. In the case of the Oxford dataset the visual odomoetry is provided by the dataset, for the UAH dataset the visual odometry was substituted with interpolated GPS data with added noise, the same method used by Xu et al. [142]. As discussed previously VPR descriptors are vulnerable to large visual changes from scenarios such as buses cutting across the camera and accumulated visual odometry is vulnerable to drift, by combining the two sensor measurements they can compensate for each others weaknesses. The advantage of this approach is that it uses prior information without requiring explicit associations between fixed sequences of query and reference images.

3.2.2.2 Algorithm

Particle filters are introduced in Chapter 2 as a Sequential Monte Carlo algorithm that estimates the posterior distribution of a system's state, given partial or noisy measurements of previous states. In this case *the state being estimated is the robot's pose*.

To begin with image descriptors extracted from a sequence of query images are defined $\{\mathbf{z}_s\}_{s=1}^{t}$ with unknown poses $\{\mathbf{T}_s\}_{s=1}^{t}$. The pose, $\mathbf{T}$, is a 4x4 matrix (Equation 3.3), containing the frame's translation, $\mathbf{t} \in \mathbb{R}^3$, and rotation $\mathbf{R} \in \mathrm{SO}(3)$ which repre-

sents a rotation in 3-D in a right-handed Cartesian coordinate system. Visual odometry, $\mathbf{U}$, is represented as a change of pose in the same format. Therefore $\mathbf{T} \in \mathrm{SE}(3)$, where $\mathrm{SE}(3)$ is the special Euclidean group which represents 6 degrees of freedom, 3 from translation and 3 from orientation between the world reference frame and the camera.

$$\mathbf{T} = \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{pmatrix}, \mathbf{U} = \begin{pmatrix} \Delta\mathbf{R} & \Delta\mathbf{t} \\ \mathbf{0} & 1 \end{pmatrix} (Oxford) \tag{3.3}$$

Unfortunately pose data was not available for the UAH Driveset, so the raw GPS values were used as a substitute $\mathbf{T}, \mathbf{U} \in \mathbb{R}^3$ (Equation 3.4) in a 3x1 column matrix.

$$\mathbf{T} = \begin{pmatrix} latitude \\ longitude \\ altitude \end{pmatrix}, \mathbf{U} = \begin{pmatrix} \Delta latitude \\ \Delta longitude \\ \Delta altitude \end{pmatrix} (UAHDriveset) \tag{3.4}$$

Only descriptors ($\mathbf{z}_s \in \mathbb{R}^D$) extracted using TinyVPR or NetVLAD networks are considered for evaluation, but others could be used. The proposed approach also uses visual odometry $\{\mathbf{U}_s\}_{s=1}^{t}$ to provide an estimated change in pose between consecutive frames. As the task is route repetition and the starting position is known an approximate measure of how far along the route each frame is, with respect to the starting point, is calculated for each image $\{\mathbf{d}_s\}_{s=1}^{t}$ by adding the magnitude of the translation from the odometry to the accumulated distance from the previous query frame $\mathbf{d}_s = (d_{s-1} + u_s)$, where $d_{s-1} = 0$ when $s = 0$. Additionally d_{s-1} may be adjusted for drift, as described below. Accumulated changes in rotation are discarded as they do not reflect a frame's distance moved along the route. The available data from each dataset means that the pose and visual odometry for Oxford is $\mathbf{T}_s, \mathbf{U}_s \in \mathrm{SE}(3)$, where $\mathrm{SE}(3)$ is the special Euclidean group which represents 6 degrees of freedom, 3 from translation and 3 from orientation and for UAH is $\mathbf{T}_s, \mathbf{U}_s \in \mathbb{R}^3$.

The multiple reference traversals of the target route collected during the the teach phase and previously used for training TinyVPR, are formed into a 'map'. The accumulated distance for each reference traversal frame is calculated by summing the magnitude of the translations between the ground truth poses $\mathbf{T}^r$ of every previous frame, the use of ground truth prevents drift. Each frame is then included in the map with a pose and accumulated distance $\mathcal{M} = \{(\mathbf{T}_s^r, \mathbf{z}_s^r, \mathbf{d}_s^r)\}_{s=1}^{N}$.

A finite set of M weighted particles $\mathcal{X}_t = \left\{ w_t^{(i)}, \mathbf{T}_t^{(i)} \right\}_{i=1}^{M}$ are then initialised around the known starting point with poses, $T_t^{(i)}$, from the map, $\mathcal{M}$, that are known to exist on the route, thereby also associating each pose with a reference image descriptor and accumulated distance. During the measurement update particle weights, $w_t^{(i)}$, are calculated using the visual and accumulated distance measurement models (Equations 3.7 & 3.8). The particle filter can therefore be described by the following recursion (Equation 3.5) which seeks to estimate the posterior distribution over the current pose $\mathbf{T}_t$ given the map and the sensor measurements from the VPR descriptor and accumulated distance.

$$p\left(\mathbf{T}_t \mid \mathbf{z}_{1:t}, \mathbf{d}_{1:t}, \mathcal{M}\right) = p\left(\mathbf{z}_t \mid \mathbf{T}_t, \mathcal{M}\right) p\left(\mathbf{d}_t \mid \mathbf{T}_t, \mathcal{M}\right) p\left(\mathbf{T}_{t-1} \mid \mathbf{z}_{1:t-1}, \mathbf{d}_{1:t-1}, \mathcal{M}\right) \quad (3.5)$$

Equation 3.5 is derived in Section 2.5.2 and reiterated in Equation 3.6 for convenience, where the weight, w, at time, t, of each, i, particle $w_t^{(i)}$ $\left(p\left(\mathbf{T}_t \mid \mathbf{z}_{1:t}, \mathbf{d}_{1:t}, \mathcal{M}\right)\right)$ is equal to the weight at the previous time step, $w_{t-1}^{(i)}$ $\left(p\left(\mathbf{T}_{t-1} \mid \mathbf{z}_{1:t-1}, \mathbf{d}_{1:t-1}, \mathcal{M}\right)\right)$, multiplied by the probability of the observation, y, given the state vector x, $p\left(\mathbf{y}_t \mid \mathbf{x}_{t|t-1}^{(i)}\right)$ $\left(p\left(\mathbf{z}_t \mid \mathbf{T}_t, \mathcal{M}\right) p\left(\mathbf{d}_t \mid \mathbf{T}_t, \mathcal{M}\right)\right)$. The probability of each observation is multiplied together, as done by Xu et al. [142].

$$w_n^{(i)} = w_{t-1}^{(i)} p\left(\mathbf{y}_t \mid \mathbf{x}_{t|t-1}^{(i)}\right) \quad (3.6)$$

Visual Measurement Model: The likelihood of observing a query image descriptor at each particle pose is calculated using the Euclidean distance between it and the reference descriptor associated with the particle's pose in the map, $\left\| \mathbf{z}_t - \mathbf{z}_s^{r(i)} \right\|$. The likelihood is drawn from a half-normal distribution $p_{hn}(0, \sigma_{vm})$, because Euclidean distances can only be positive.

$$p\left(\mathbf{z}_t \mid \mathbf{T}_t^{(i)}, \mathcal{M}\right) = p_{hn}(\left\| \mathbf{z}_t - \mathbf{z}_s^{r(i)} \right\|; 0, \sigma_{vm}) \quad (3.7)$$

Accumulated Distance Measurement Model: The likelihood of observing a particular accumulated distance at each particle pose is drawn from a normal distribution $p_n(\mathbf{d}_s^{r(i)}, \sigma_{dm})$

$$p\left(\mathbf{d}_t \mid \mathbf{T}_t^{(i)}, \mathcal{M}\right) = p_n(\mathbf{d}_t; \mathbf{d}_s^{r(i)}, \sigma_{dm}) \tag{3.8}$$

Drift Compensation: If the particle with the highest weight has a visual measurement from Equation 3.5 above a user defined threshold then the measured accumulated distance, $\mathbf{d}_t$, is set to the accumulated distance ground truth associated with the maximum a posteriori (MAP) reference descriptor from the map. This allows highly likely descriptor matches to be used to correct visual odometry drift.

Standard deviations are defined by the user to model the distribution of likely values. For example, if 85% of the Euclidean distances between test and particle VPR descriptors is between 0 and 1 then a suitable value for σ_{vm} would be 1. The value of σ_{dm} should reflect the standard deviation of the distance moved between repetition images and can be estimated from the reference traversals of the route collected during the teaching phase. Automatically establishing parameters for these distributions remains an outstanding problem, but can be approximated heuristically.

As $p\left(\mathbf{T}_t \mid \mathbf{z}_{1:t}, \mathbf{d}_{1:t}, \mathcal{M}\right)$ is calculated using each particle the filter's execution time scales directly with the number of particles used, M, and is therefore $O(N)$. Another advantage of the proposed approach is that particles can be instantiated simultaneously across multiple reference traversals of the route, a similar concept was explored by Churchill et al. [75].

Algorithm 1: Particle Filter Route Repetition with VPR Descriptors and Visual Odometry

initialise a geographical starting point on the target route and choose an approach for generating VPR descriptors, in this case TinyVPR.
for each reference traversal of the target route:
    for each frame:
        Extract a descriptor, $\mathbf{z}^r$, and calculate the accumulated distance, $\mathbf{d}^r$, by summing the magnitude of the translations between the ground truth poses, $\mathbf{T}^r$, of every previous frame.
    end for
end for
initialise a 'map', $\mathcal{M} = \{(\mathbf{T}_s^r, \mathbf{z}_s^r, \mathbf{d}_s^r)\}_{s=1}^N$, with all reference poses, descriptors and accumulated distances.
initialise a set of equally weighted M particles $\mathcal{X}_t = \left\{w_t^{(i)}, \mathbf{T}_t^{(i)}\right\}_{i=1}^M$ with weight, $w_t^{(i)}$ and pose $\mathbf{T}_t^{(i)}$ at time, t=0, centred around the start point.
end initialisation
Generate a new set of particles at time $t+1$, $\mathcal{X}_{t+1}$ by randomly sampling the previous particle set $\mathcal{X}_t$ in proportion to their weights and with replacement.
At an unknown pose, $\mathbf{T}_t$, measure a query image descriptor, $\mathbf{z}_t$, and accumulated distance, $\mathbf{d}_t$.
for particles $i, 1:M$ in $\mathcal{X}_t$:
    Update each particle by adding a small amount, d_{up}, to the accumulated distance $\mathbf{d}_s^{r(i)}$ associated with each particle drawn randomly from a normal distribution $N\overline{D}, \sigma_{dm})$, where $\overline{D}$ is a user defined constant associated with mean accumulated distance per query frame.
    The particle's pose, $\mathbf{T}_t^{(i)}$, is then updated by finding the pose that corresponds to the new accumulated distance, $\mathbf{d}_s^{r(i)}$, in the map. The particle is then associated with a new image descriptor $\mathbf{z}_s^{r(i)}$ that corresponds to the new pose. This step allows the particles to be updated without leaving the route.
    Calculate the weight of each particle by calculating its likelihood using the visual and accumulated distance measurement models as given by the general form of the filter's recursion (Equations (3.5) - (3.8):
    $$w_t^{(i)} = p\left(\mathbf{z}_t \mid \mathbf{T}_t^{(i)}, \mathcal{M}\right) p\left(\mathbf{d}_t \mid \mathbf{T}_t^{(i)}, \mathcal{M}\right)$$
end for
if the particle with the highest weight, j has a visual measurement greater than a user defined threshold, $p\left(\mathbf{z}_t \mid \mathbf{T}_t^{(i)}, \mathcal{M}\right) > threshold$, then perform drift correction as described previously: $\mathbf{d}_t = \mathbf{d}_s^{r(j)}$
end if
Normalise the weights of all particles in $\mathcal{X}_t$ to a total of 1.
Calculate the pose estimate, $\mathbf{T}_t$, with a mean of particle poses weighted by their normalised weights.
if the end of the route has been reached end, otherwise loop to 5.

3.3 Evaluation

In this section the proposed combination of TinyVPR descriptor and particle filter is compared against state-of-the-art visual teach and repeat algorithms Open SeqSLAM 2.0 (OSS2) [132] and Online Place Recognition (OPR) [131]. State-of-the-art off-the-shelf VPR descriptor NetVLAD was also combined with the proposed particle filter for further comparison. Secondly, specific examples are used to compare the robustness of these approaches to variable route repetition speed and large visual changes. OSS2 and OPR do not use visual odometry, unlike the proposed approach, but the comparison remains fair as the the proposed visual odometry is derived only from images which are equally available to the alternatives.

3.3.1 Experimental Setup

Evaluation was done on two datasets. The first route was a 2km section of the Oxford RobotCar dataset [115] which was collected over the course of a year in different lighting conditions and seasons. The second route was a 16km section of the UAH-DriveSet [200] collected in Spain on a semi-rural single lane road over 3 months in a large variety of weather and lighting conditions.

Table 3.1: Details of each reference route traversal with the original dataset details. Timestamps are in the format: year-month-day-hour-minute-second

Orig. Dataset Ref.	Label	Sampled Frames
Oxford		
2014-12-10-18-10-50	Night	884
2015-10-29-12-18-17	Rain	884
2014-12-16-09-14-09	Morning	884
2015-03-10-14-18-10	Afternoon	884
2015-03-03-11-31-36	Midday	884
2014-11-25-09-18-32	Overcast	884
UAH		
2015-11-11-13-45-42	Blue Sky	2951
2015-12-11-16-28-29	Grey Clouds	2951
2015-12-03-17-43-23	Sunset	2951
2015-11-26-12-42-08	Light Cloud	2951

3.3.1.1 Teaching Phase

For Oxford six reference traversals (Figure 3.1) of the route were collected, sampled and the data pre-processed as described in Section 3.2.1.2 for 884 images per traversal at mean intervals of approximately 2.25m for a total of 5304 reference images. For the UAH-DriveSet four reference traversals of the route (Figure 3.1) were pre-processed similarly resulting in 2951 images per traversal, for a total of 11804 reference images at mean intervals of approximately 18m. A summary of these details and their original dataset labels/timestamps can be found in Table 3.1.

3.3.1.2 Repetition Phase

For testing VT&R traversals of the same route in previously unseen appearance variations of the Oxford and UAH datasets were used (Figure 3.8) and the mean localisation error of each frame was recorded.

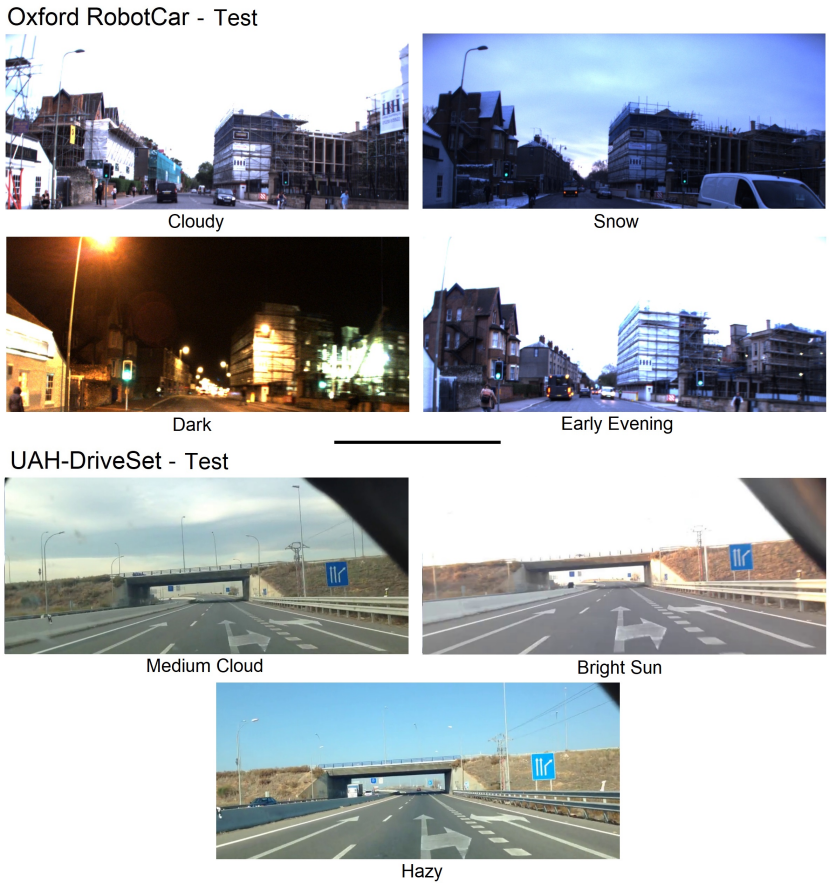


Figure 3.8: Examples of the test repetition traversals used for the Oxford and UAH datasets showing the variations in appearance.

These traversals were originally real-time videos, but were sampled at time inter-

vals merely to reduce the test dataset frames because comparing many test frames against all reference images was very computationally intensive, for ease of reference these are referred to as unsampled. However during testing OSS2 and OSR were found to fail very quickly when presented with this test data, mainly because they relied on fixed sequences but the reasons for this are explored in more detail later. To enable comparison the test traversals used for OSS2 and OPR had to be pre-processed according to Section 3.2.1.2 to remove the variations in speed. A summary of these details and their original dataset labels/timestamps can be found in Table 3.2. Furthermore the proposed particle approach requires visual odometry data which is included in the Oxford dataset, but not in the UAH-DriveSet so interpolated GPS data with noise added was used as a substitute, a similar substitution was done by Xu et al. [142].

Table 3.2: Details of each test repetition route traversal with the original dataset labels. Timestamps are in the format: year-month-day-hour-minute-second.

Orig. Dataset Label	Label	Sampled Frames	Unsampled Frames
Oxford			
2015-07-03-15-23-28	Cloudy	884	3485
2015-02-03-08-45-10	Snow	884	3132
2014-12-16-18-44-24	Dark	884	3486
2014-12-02-15-30-08	Early Evening	884	3942
UAH			
2015-11-20-16-09-03	Medium Cloud	2951	3451
2015-12-03-17-17-59	Bright Sun	2951	3893
2015-12-21-11-24-44	Hazy	2951	4153

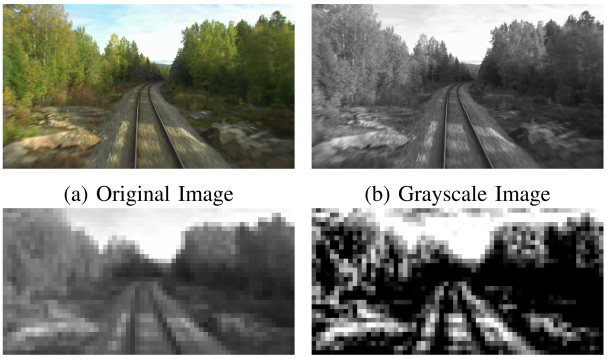
There is some apparent overlap between the reference and test appearance variations as shown in Figures 3.1 & 3.8, but this is because the range of appearance variation across the whole route is difficult to capture across one location on the route. As well as weather and lighting conditions the appearance variations are also a result of long term changes, evidence for this can be found in the timestamps associated with each of the datasets: many traversals were recorded months apart (Table 3.1 & 3.2). A sample of these further appearance variations is shown in Figure 3.9.

Figure 3.9: Examples of the further appearance variations between locations in traversals of the Oxford and UAH routes.

3.3.1.3 Comparison Approaches

OpenSeqSLAM 2.0 (OSS2) [132] and the original SeqSLAM paper remain popular state-of-the-art approaches and have been used for comparison with state-of-the-art approaches [133, 134, 142]. OSS2 is designed to use small sequences of consecutive images to reduce the impact of appearance variation on route repetition. OSS2 begins by downsampling images from a single reference traversal, converting them to

grayscale, dividing them into patches and then performing patch normalisation to enhance contrast and create an image 'template' that acts as a VPR descriptor (Figure 3.10).

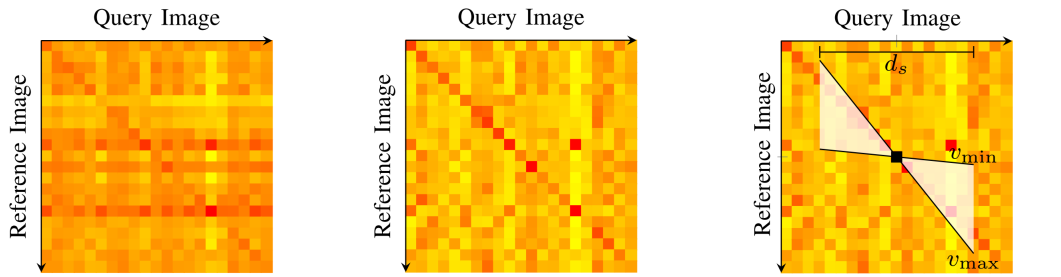


(a) Original Image (b) Grayscale Image

(c) Cropped and Resized Image (d) Enhanced Contrast Image

Figure 3.10: The four image pre-processing steps used in OpenSeqSLAM 2.0 [132].

The same pre-processing is applied to incoming test images from a route repetition. A Sum of Absolute Differences (SAD) matcher is used to compare the test and reference VPR descriptors, which is then plotted in a descriptor comparison plot. Local neighbourhood normalisation is applied to the descriptor comparison plot and a number of search strategies are used to find trajectories of differences that represent the most likely reference image match for each test image (Figure 3.11).



(a) Before local normalisation (b) After local normalisation (c) General search configuration

Figure 3.11: Three processing steps for refining the descriptor comparison plot of test and reference images and then using windows of d_s images for searching for trajectories between v_{min} and v_{max} used in OpenSeqSLAM 2.0 [132] where yellow represents a large difference between images and red a small difference.

OSS2's trajectory matching approach allows a basic invariance to variable repetition speeds, but ultimately relies on large stretches of the route to have only brief visual changes that do not disrupt the trajectories. OSS2 is also vulnerable to routes with a relatively homogeneous appearance because this leads to many plausible tra-

jectories being calculated as different parts of the route are similar to each other and trajectories across the entire descriptor comparison plot are considered.

Online Place Recognition (OPR) [131] is an approach that specifically aims to operate in real-time. It uses a directed graph with no graph cycles (acylic) that is built as each test image becomes available by initialising a node based on the similarity between the test image and the reference database. Similarity is measured by extracting image descriptors and measuring the distance between them using the cosine distance. HOG [86] or generic CNN descriptors [136] were originally used but were updated in this implementation to CNN descriptors extracted from the penultimate layer of VGG16 [12]; VGG16 was originally trained on the same image classification task as the original CNN, but is more recent and performs better. NetVLAD descriptors were also experimented with but the dimension of the descriptors was not natively supported. If the similarity between descriptors is high then that node will be expanded to a new set of nodes that represent comparisons between the next test image and the reference images. To prevent exponential expansion lazy data association is used to predict the cost of future expansions and prevent expansion of the least promising nodes. Localisation predictions are found by searching the shortest path through the graph.

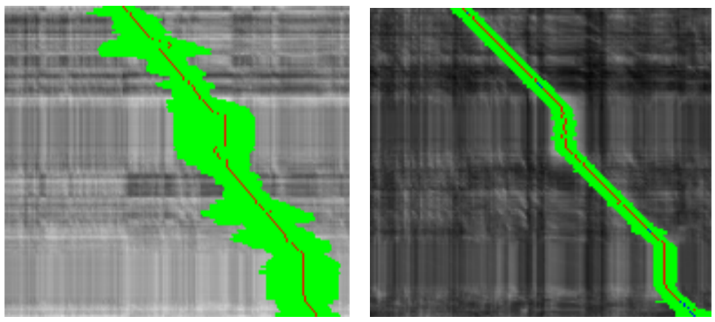


Figure 3.12: Descriptor comparison plot from the original OPR paper [131] showing the spread of graph nodes across potential reference image matches in green. Overfeat CNN (right) descriptors are more distinctive than HOG (left) descriptors and therefore require fewer nodes for matching. Variable speed traversal is also shown here, with vertical lines indicating a stationary camera.

OPR works well to reduce the amount of reference image comparisons done at each time step of a route repetition, but is vulnerable to large visual changes which disrupt the generation of appropriate nodes. This weakness is exacerbated when variable repetition speeds are combined with visual changes and causes a significant fail-

ure point in the algorithm. One observed example was when the input camera became stationary at a junction and cars passed in the opposite direction this introduced noise which corrupted the creation of new nodes and moved them away from the correct position. When the camera started moving again all the newly created nodes were equally dissimilar to the incoming test image and the algorithm was unable to recover.

3.3.2 TinyVPR Training

Once the teach phase has been used to collect reference traversals of each route TinyVPR can be trained on that route. For Oxford 51,000 triplets were mined and used for training over 10 epochs in batches of 16 with a margin of 1. For UAH TinyVPR was trained using 177,060 triplets over 15 epochs in batches of 16 with a margin of 0.1. In both cases 5% of the generated triplets were used for validation, 10 and 15 epochs were used for training because it was observed that the validation loss increased beyond these amounts of epochs. Training parameters were established using a grid search which exhaustively searches a manually specified subset of the hyperparameter space, NetVLAD was trained in a similar way so it was used as the basis for the parameter subset. Training took approximately an hour on a mobile Nvidia GTX 1070.

3.3.2.1 Which Triplet Mining Approach Worked Best?

As part of the grid search for establishing training parameters different triplet mining approaches described in Section 3.2.1.5 were evaluated for training TinyVPR. Triplet hardness was not calculated during training as this increased training time by up to a factor of 10, but this did reduce training efficiency because approximately 75% of the mined triplets were easy. For further comparison MobileNet was replaced by VGG16 in TinyVPR by removing its final layer, freezing the remainder of the network and replacing it with the layers shown in Figure 3.6. The following experiments were performed on it as well.

For triplet mining two boundaries are typically defined, for example NetVLAD defines positive images as being within 10 meters of the anchor, and negative images as further away than 25 metres. However, for ease of evaluation these two boundaries were collapsed into one, positive images were defined as being within this boundary and negative images beyond it; this increased the likelihood of harder triplets being mined (the negative image being more similar to the anchor image than the positive

image). All numbers used for these boundaries are defined in terms appropriate to the mining approach. For example, a boundary of 5 would be 5 metres for geographical mining and 5 frames for frame-wise mining. Not all anchor images contained a positive image within 5 meters so this boundary was not used for hybrid and geographical mining.

This test was only performed on the Oxford dataset because it included the most challenging appearance variations and training all the networks with different triplet mining approaches was limited by training and testing times. The unsampled test repetitions were used for this experiment. Each cropped test image was passed through the trained model to produce a VPR descriptor which was compared using the Euclidean distance to the reference image VPR descriptors. The 20 reference images most similar to the query image were found and the one with the smallest localisation error used to evaluate each mining approach. This evaluation metric was used because the proposed approach aims to filter the results and find the lowest error localisation estimate, this metric is also related to image retrieval recall [10, 199] which counts a result as a true positive if a true match is included in a set of top matches. The mean results across the four test route traversals are collected in Table 3.3.

Table 3.3: Comparison of triplet mining techniques (Section 3.2.1.5) using mean localisation error (metres) across four test route traversals. The positive/negative boundary is given in metres/metres (Geographical), frames/frames (Frame-wise) and metres/frames (Hybrid).

Mining Approach	**Hybrid**				**Geographical**				**Frame-wise**				
Pos./Neg. Boundary	10	20	40	60	10	20	40	60	5	10	20	40	60
TinyVPR (M.Net)	8.96	4.33	4.76	5.06	5.78	6.06	6.44	5.67	5.98	**4.10**	5.20	5.11	4.79
TinyVPR (VGG16)	7.20	12.1	6.41	5.88	8.14	9.71	8.66	7.87	**5.40**	8.83	8.38	6.54	6.57

Frame-wise mining performed better than geographical mining. This may be because a positive image needs to be from a very similar viewpoint to the anchor, but geographical mining can select a positive image within a couple of metres but with a very different viewpoint, for example from around a corner. However, because the training set is pre-processed at distance and rotation intervals (as discussed in Sections 3.2.1.2 and 3.2.1.5) using consecutive frames to define mining boundaries reduces the risk of this. Hybrid mining outperforms frame-wise mining for larger boundaries but never performs the best. More analysis to investigate this is required but a potential explanation for this is that hybrid mining enables greater overlap between positive and negative images which improves training by generating harder triplets, but ulti-

mately is outweighed by the aforementioned advantages of frame-wise mining. These results suggest the importance of mining appropriate triplets compared to prioritising 'hard' training data. Given these results all TinyVPR networks were therefore trained using frame-wise mining with a boundary of 10.

3.3.3 Real-Time Route Repetition

Once TinyVPR descriptors had been taught on the reference route traversals of the Oxford and UAH datasets they could be combined with the proposed particle filter and evaluated for route repetition against OpenSeqSLAM 2.0 (OSS2) and Online Place Recognition (OPR). For Oxford six reference traversals (Table 3.1) collected during the teaching phase were used for reference when repeating each of the four test traversals 3.2. One advantage of the proposed approach is that it can compare multiple reference traversals simultaneously with a single test traversal which is not possible for OSS2 and OPR because they are only designed to consider a single reference route for comparison, whereas the particle filter can distribute candidate particles across as many reference routes as required. Some reference routes visual variations may be easier to match against a test traversal so prior knowledge would be needed to select the best reference route for OSS2 and OPR, whereas the proposed approach handles this automatically, although may require more particles. Only using a single reference traversal for the proposed approach would put more emphasis on the descriptor matching experiments on this are presented in Sections 3.3.4.1 and 3.3.5.1. For OPR and OSS2 each test traversals was compared with one reference traversal at a time and the mean of the localisation results was recorded. A brief heuristic search of the main parameters used in OSS2 [1] and OPR [2] was done to see whether any major improvements could be made but in the end both approaches were used with default parameters. with one exception For OSS2 $v_{min} = 0.8$ and $v_{max} = 1.2$ were left unchanged but d_s, as seen in Figure 3.11, was increased to 30 to improve matching - a parameter value shown to be effective in [132].

For this test a 100 particle filter was used for the Oxford dataset and a 200 particle filter for the UAH dataset. These amounts were chosen because they represented less than 2% of the total reference frames for each route: 5304 for Oxford and 11804 for

[1] https://github.com/qcr/openseqslam2
[2] https://github.com/PRBonn/online_place_recognition

UAH.

In addition to the proposed TinyVPR descriptor the pre-trained NetVLAD descriptor was also used with the particle filter for further comparison. Attempts were made to re-train the final VLAD layer of NetVLAD for this task as well, but ended in triplet collapse - a phenomenon where all descriptors collapse into a single point [203], this may be caused by a lack of variety in the training data from only training on a single route. Unlocking further layers for training would have removed any pre-trained layers generated by the original model removing any of its specialisation for VPR.

Initial testing showed that OPR and OSS2 failed significantly on the unsampled test data, so to enable comparison the sampled test data was used for OSS2 and OPR. This biased the test in their favour, but only reinforced the significance of the results with respect to the advantage of using the proposed approach for real-time test data.

Results are shown in Tables 3.4 & 3.5. Localisation error is calculated by comparing the distance between the test image and the output of the particle filter, according to Algorithm 1. For each test frame in each of the four test routes (Table 3.2) the mean, median and standard deviation of the localisation errors was recorded.

Table 3.4: Mean, median and standard deviation of localisation error (metres) for visual teach and repeat on Oxford Robotcar Dataset. TinyVPR and NetVLAD descriptors are used with a 100 particle filter and compared against OSS2 and OPR.

	OSS2			**OPR**			**NetVLAD + PF**			**TinyVPR + PF**		
Test Traversal	*Mean*	*Med*	σ	*Mean*	*Med*	σ	*Mean*	*Med*	σ	*Mean*	*Med*	σ
Cloudy	49.0	37.1	48.2	68.4	73.9	44.9	5.83	5.29	3.01	4.04	3.39	2.52
Dark	132	132	56.6	87.8	70.3	74.5	2.66	2.31	1.65	3.39	2.65	2.33
Snow	60.6	51.6	42.9	60.6	40.9	56.3	4.50	4.69	1.69	2.70	1.95	2.18
Early Evening	32.4	27.7	29.5	52.0	48.3	36.6	10.7	11.6	6.22	6.09	5.17	3.96
Mean	68.5	62.1	44.3	67.2	58.3	53.1	5.92	5.97	3.14	**4.01**	**3.29**	**2.75**

Table 3.5: Mean, median and standard deviation of localisation error (metres) for visual teach and repeat on UAH-DriveSet. TinyVPR and NetVLAD descriptors are used with a 200 particle filter and compared against OSS2 and OPR.

	OSS2			**OPR**			**NetVLAD + PF**			**TinyVPR + PF**		
Test Traversal	*Mean*	*Med*	σ	*Mean*	*Med*	σ	*Mean*	*Med*	σ	*Mean*	*Med*	σ
Medium Cloud	3620	3380	1980	4090	3040	2840	27.7	26.0	13.9	20.3	17.9	13.0
Bright Sun	2970	2750	1830	4840	4380	3410	31.9	31.3	16.0	37.9	37.5	17.2
Hazy	2610	1850	2360	2210	1950	1827	27.2	25.9	15.4	25.0	24.2	12.5
Mean	3070	3380	2060	3710	3120	2690	28.9	27.7	15.1	**27.7**	**26.5**	**14.2**

3.3.3.1 Is TinyVPR + Particle Filter Better Than OSS2 and OPR for VT&R?

TinyVPR+PF reduces mean localisation error compared to OSS2 and OPR by a minimum of 94.3% from 67.2 to 4.01 (Table 3.4) for the Oxford dataset. For the UAH dataset mean localisation error is reduced compared to OSS2 and OPR by a minimum of 99.1% from 3070 to 27.7 (Table 3.5). Localisation error in metres is related to the overall size of the route as smaller routes create an upper limit on the the localisation error.

This significant improvement of the proposed TinyVPR+PF approach in comparison to the state-of-the-art alternatives can be attributed firstly to the specialised TinyVPR descriptors, which outperform the competing approaches' descriptors by being more appearance invariant and returning more distinct results, as seen in Figure 3.13. Secondly, the particle filter enables a distribution of localisation predictions to be established without establishing an explicit dependency on sequential information used by OPR and OSS2, which makes them vulnerable to disruptions in these sequences. Two failure points that TinyVPR+PF helps to address are large visual changes and variable speed route repetitions. Detailed investigations into the robustness of these algorithms to large visual changes and variable speed repetitions can be found in Section 3.3.4 & Section 3.3.5, but initial evidence to support this claim is that OPR and OSS2 failed when presented with unsampled test images.

Additionally, OSS2 compares sequences of query and reference images to find recurring similarities between them for localisation and was designed to operate offline. However, as OSS2 uses a fixed window to search for matches cached images could be used to provide localisation predictions at distance intervals, whereas OPR and the particle filter operate in real-time.

3.3.3.2 Are TinyVPR Descriptors Better Than NetVLAD's?

TinyVPR reduces mean localisation error 32.3% from 5.92 to 4.01 (Table 3.4) for the Oxford dataset and 4.3% from 28.9 to 27.7 (3.5)) for the UAH dataset in comparison to using NetVLAD descriptors, when both are implemented in an identical particle filter. This decrease in localisation error is due to the TinyVPR descriptors being taught specifically for each route using the traversals collected during the teach phase. This has a larger effect on the Oxford dataset because it is a more complex environment with elements such as construction work (Figure 3.9, Cloudy vs. Rain) and unique architecture that are not present in the more rural UAH dataset, which lends itself to

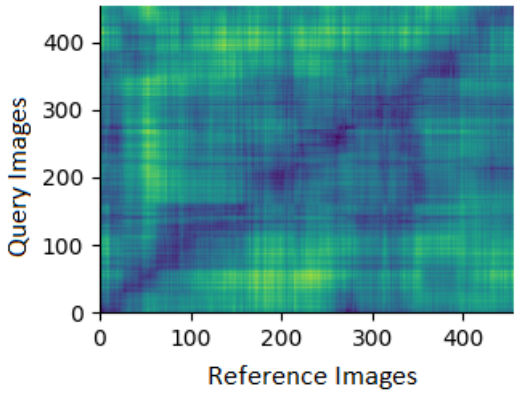


Figure 3.13: Sample of a TinyVPR descriptor comparison plot of Night and Cloudy traversals of the Oxford RobotCar dataset where more similar comparisons are dark blue and less similar comparisons are in green. Note the improved distinctiveness of the matching diagonal of descriptors compared to OSS2 (Figure 3.11) and OPR (Figure 3.12).

a more generic VPR descriptor.

NetVLAD's performance on these unseen and challenging routes is a testament to its original training and its implementation in the proposed particle filter still significantly outperforms the state-of-the-art. However, it is a large and complex network with $2.4 * 10^7$ parameters and an inference time of 80ms compared to TinyVPR's $8.0 * 10^5$ parameters and inference of time 7ms on an Nvidia GTX 1070. TinyVPR is therefore 11x faster with 30x fewer parameters.

Once the test image VPR descriptor had been generated by the network it was compared to the reference VPR descriptor state associated with each particle in the particle filter using the Euclidean distance (Equation 3.5). NetVLAD descriptors were vectors of 4096 in length and TinyVPR descriptors were vectors of length 128. 100 (Oxford) or even 200 (UAH) of these comparisons added a negligible amount of time which could be further decreased by using the Faiss library [3] designed to speed up these operations by up to 8.5x [204].

3.3.3.3 How Important is the Particle Filter?

The proposed combination of the TinyVPR descriptors and particle filter has been shown to reduce localisation error for route repetition, but it can be difficult to attribute this improvement to either the particle filter or the descriptor. This experiment therefore compares the localisation error from using just the TinyVPR descriptor in a

[3] https://github.com/facebookresearch/faiss

naive global search to implementing it in the particle filter to see the specific effect of using the particle filter. Localisation error is calculated by comparing the distance between the test image and the output of the particle filter, according to Algorithm 1. For each test frame in each of the four test routes (Table 3.2) the mean, median and standard deviation of the localisation errors was recorded. The results for this are shown in Table 3.6 and Table 3.7.

Table 3.6: Mean localisation error (metres) on Oxford RobotCar Dataset of TinyVPR descriptors used with or without the proposed particle filter.

	TinyVPR		**TinyVPR + PF**	
Test Traversal	*Mean*	σ	*Mean*	σ
Cloudy	21.8	60.2	4.04	2.52
Dark	9.28	28.7	3.39	2.33
Snow	6.76	18.8	2.70	2.18
Early Evening	6.28	24.0	6.09	3.96
Mean	11.0	32.9	**4.01**	**2.75**

Table 3.7: Mean localisation error (metres) on UAH-DriveSet of TinyVPR descriptors used with or without the proposed particle filter.

	TinyVPR		**TinyVPR + PF**	
Test Traversal	*Mean*	σ	*Mean*	σ
Medium Cloud	248	1190	20.3	13.0
Bright Sun	232	769	37.9	17.2
Hazy	145	848	25.0	12.5
Mean	208	936	**27.7**	**14.2**

The particle filter reduces mean localisation error 63.7% from 11.0 to 4.01 (Table 3.4) for the Oxford dataset and 86.7% from 208 to 27.7 for the UAH dataset (Table 3.5). This demonstrates that the improved VPR descriptors in a naive global search perform better than the state-of-the-art on this task, but still require the particle filter to reduce large errors and significantly increase reliability. Further evidence for this can be seen in the reduction in mean standard deviation of the localisation error of 91.6% from 32.9 to 2.75 for Oxford and 98.5% from 936 to 14.2 for UAH.

Overall these results show the specialised VPR descriptors NetVLAD and TinyVPR are significantly better than OPR for this task, despite its use of similar neural network descriptors. They localise so accurately in a naive global search that they only require a probabilistic filter to remove noise from large visual changes to reduce localisation in comparison to state-of-the-art alternatives. The particle filter is able to remove incorrect predictions caused by visual changes, or general descriptor noise by leveraging

visual odometry and the distribution of previous localisation predictions. Increased visual odometry noise would affect this, but the visual odometry is corrected for drift upon particularly close descriptor (Algorithm 1).

3.3.4 Large Visual Changes

3.3.4.1 Natural

Large visual changes are naturally present throughout both datasets, to investigate their affect on the localisation error on each algorithm the Oxford RobotCar *cloudy* query traversal was compared against only the *morning* reference traversal. These traversals were chosen because their similar appearances meant large visual changes could be considered in isolation. The sampled versions of these traversals were used to remove any changes in repetition speed. The mean localisation error when comparing each query image query against the reference traversal using OSS2, OPR, TinyVPR (naive global search) and TinyVPR+Particle Filter was plotted and specific large visual changes highlighted in Figure 3.14.

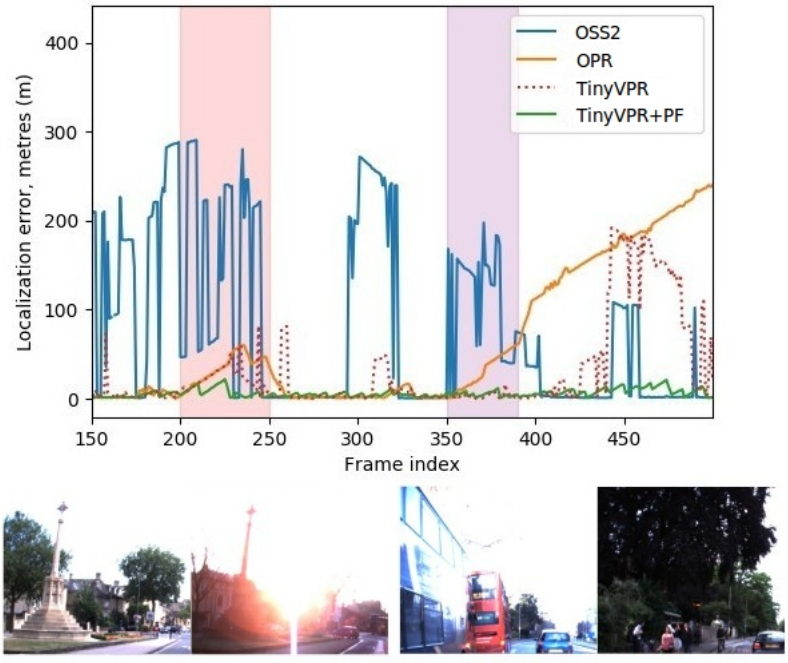


Figure 3.14: Mean localisation error of the cloudy query traversal compared against the morning reference traversal from the Oxford RobotCar Dataset [115]. Particular examples of large visual changes are shown between frames 200 - 250 and 350 - 390, specific images at these points are shown below the plot.

3.3.4.2 Artificial

For further evaluation a 200m section of the reference and test traversals of the route, represented by 100 frames, was artificially manipulated to examine the algorithm's robustness and compared against a route repetition. The 100 images used were 750-849 (inclusive) of the sampled 884 images of each traversal and were chosen because of their lack of visual changes. Images of cars from the Stanford Cars dataset [205] were cropped into increasing proportions of the reference and test traversal frames to realistically simulate cars passing at different rates and the contrast and brightness of the frames was also randomly varied, samples of these images can be found in Figure 3.15. Both test and reference frames were manipulated because large visual changes could also appear during the teach phase. To reflect these visual changes noise was added to the dataset's visual odometry using a normal distribution with mean and standard deviation equal to 10% of the original measurement. The test repetition was then compared against the reference data using the three algorithms. The mean localisation error when comparing each query image query against the reference traversal using OSS2, OPR, TinyVPR (naive global search) and TinyVPR+Particle Filter was plotted in Figure 3.16.

Figure 3.15: Samples of the artificial large visual changes introduced by cropping data from the Stanford Cars dataset [205] into the reference and query frames.

3.3.4.3 Is TinyVPR+PF More Robust to Large Visual Changes?

An example of a naturally occurring visual change (Figure 3.14) is shown to be between frames 200 - 250 of the query traversal and is due to lighting conditions, whereas frames 350 - 390 show a large visual change from an occlusion by two buses (Figure 3.14). In these scenarios OSS2 localisation error is higher than OPR, but OSS2 recovers

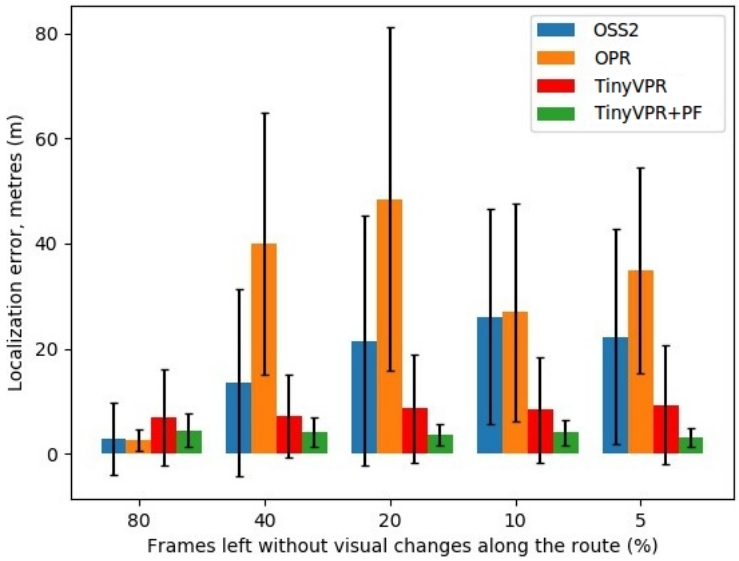


Figure 3.16: Mean localisation error and standard deviation of the cloudy query traversal compared against the morning reference traversal from the Oxford RobotCar Dataset [115] with increasing amounts of frames with artificial visual changes added.

from its incorrect predictions. OPR handles the first lighting change well, but cannot recover from the second occlusion. This chapter's approach remained resilient in both scenarios.

Artificially increasing the amount of frames with visual changes (Figure 3.16) show OPR is the worst affected. Using TinyVPR descriptors in a global search significantly outperforms OSS2 even when extremely disrupted, although the standard deviation of the error increases. These occasional, large errors are dealt with effectively by adding the particle filter.

OSS2 compensates for using descriptors that are not specialised for appearance invariant VPR by detecting recurring similarities that are available over long distances in a constant environment, but become progressively harder to establish over short distances in a dynamic environment. The particle filter improves on this by using more accurate TinyVPR descriptors specialised for the task which provide better global matching as seen when comparing the descriptor comparison plots of OSS2 (Figure 3.11), OPR (Figure 3.12) and the TinyVPR descriptors (Figure 3.13).

OPR relies solely on VPR descriptor comparisons when building its acyclic graph which are inherently vulnerable to large visual changes. The combination of a particle filter with TinyVPR descriptors improves on this by explicitly estimating the uncertainty of descriptor comparisons and taking that into account when making a locali-

sation prediction.

3.3.5 Variable Repetition Speed

3.3.5.1 Natural

Variable repetition speeds are naturally present in all the unsampled route traversals. For this investigation the Oxford RobotCar *cloudy* query traversal was again compared against only the *morning* reference traversal. The query traversal was unsampled to include all traffic light stops and other speed variations, but the reference route remained sampled. The mean localisation error when comparing each query image query against the reference traversal using OSS2, OPR, TinyVPR (naive global search) and TinyVPR+Particle Filter was plotted and specific changes in repetition speed highlighted in Figure 3.17.

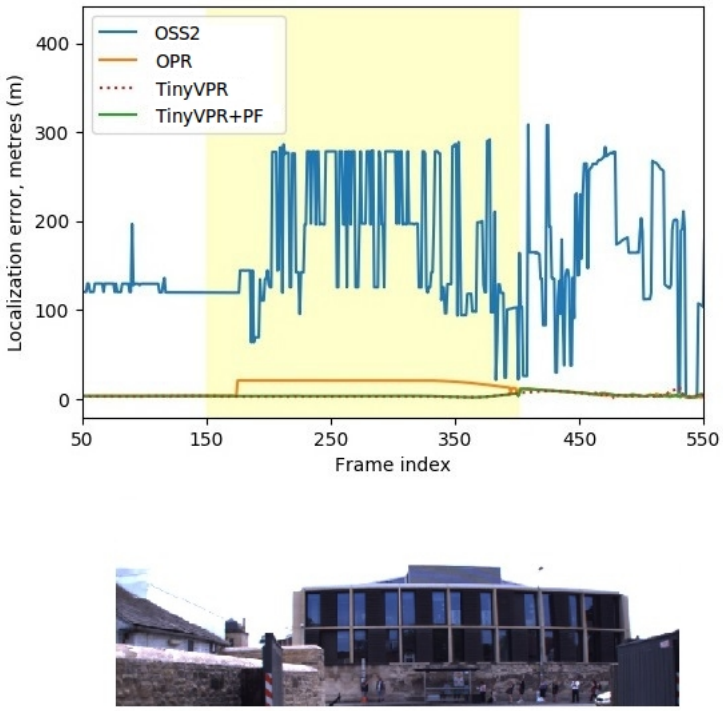


Figure 3.17: Mean localisation error of the cloudy query traversal compared against the morning reference traversal from the Oxford RobotCar Dataset [115]. A particular example of variable speed change is shown between frames 150 - 400. A specific image at this points is shown below the plot. Note that TinyVPR in a global search performs identically to its usage in the particle filter.

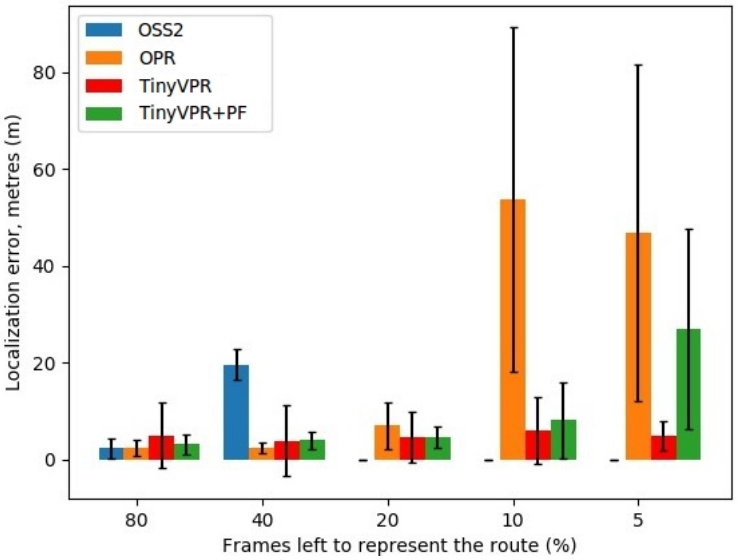


Figure 3.18: Mean localisation error and standard deviation of the cloudy query traversal compared against the morning reference traversal from the Oxford RobotCar Dataset [115] with increasing amounts of frames removed to simulate changes in repetition speed.

3.3.5.2 Variable Speed Traversal - Artificial

For further evaluation a 200m section of the reference and test traversals of the route represented by 100 frames was artificially manipulated to examine the algorithm's robustness. The 100 images used were 750-849 (inclusive) of the sampled 884 images of each traversal and were chosen because of their lack of visual changes which would be otherwise disruptive. To reflect these visual changes noise was added to the dataset's visual odometry using a normal distribution with mean and standard deviation equal to 10% of the original measurement. The number of frames in the reference traversal were fixed, while progressively more of the test traversal's frames were removed manually to simulate large and small accelerations. The test repetition was then compared against the reference data using the three algorithms. The mean localisation error when comparing each query image query against the reference traversal using OSS2, OPR, TinyVPR (naive global search) and TinyVPR+Particle Filter was plotted in Figure 3.18. Unfortunately OSS2 produced no matches for some parameters as it relied on sequences of test images which were simply not available.

3.3.5.3 Is TinyVPR+PF More Robust to Variable Speed Traversals?

A natural variation in repetition speed occurs between frames 150 - 400 when the camera pauses before pulling out into traffic (Figure 3.17). Results show OPR's error rises to 20m, while this chapter's approach error remains at 3m. OSS2 breaks down entirely in this scenario.

When artificial variable repetition speeds are introduced OSS2 only performs well for minor changes. A global search using the TinyVRP descriptors produced the lowest localisation error once 20% of the frames had been removed because it uses no sequential information which could be disrupted, but a significant difference was only available at extreme variations in speed. Localisation error eventually increased because particle updates after each prediction (Algorithm 1, line 8) is drawn from a distribution which is unlikely to produce updates that keep pace with such accelerations.

OSS2 relies on a constant relationship between query and reference images because its trajectory search can only identify straight lines of similarities between images in a descriptor comparison plot, if this is disrupted by variable speeds or large visual changes then it fails. For example, Figure 3.11 shows the straight diagonal line trajectory search and Figure 3.12 shows how this straight line is disrupted by variable speed. The particle filter reduces this vulnerability by relaxing the need for explicit sequence matching and only using a previously calculated distribution of localisation predictions from previous results to inform subsequent predictions.

OPR also claims to be robust to route repetitions at variable speeds but, particularly when these are combined with visual changes, it fails because the descriptor comparisons become so unreliable that the graph expands away from the correct matches. When the camera resumes motion the isolated graph nodes are left isolated in incorrect predictions and cannot recover. The proposed particle filter updates and resamples particle states to prevent predictions becoming isolated in poor predictions in addition to using visual odometry to take into account variations in repetition speed.

3.4 Discussion

This chapter proposes a contribution towards robust, real-time visual teach and repeat. To enable this a number of challenges had to be overcome. Firstly a descriptor had to be learnt that was specialised for each route, but was still capable of pro-

viding state-of-the-art appearance invariance while reducing inference time to allow real-time implementation on robotic hardware. The second challenge was to use information from sequences of localisation results to reduce localisation error during route repetition without introducing dependencies that prevent practical real-time operation, particularly for large visual changes and variable route repetition speed.

To address these challenges data collected during the teaching phase of visual teach and repeat was used to train a compact TinyVPR CNN architecture using triplet loss to generate VPR descriptors specialised for each route. By learning to generate appearance invariant descriptors across only a single route the network capacity could be reduced which enables faster descriptor generation and makes it easier to implement on embedded hardware typical in robotics. A particle filter was then used to filter localisation results for route repetition using the TinyVPR descriptor by assigning a probability to each prediction based on measured descriptor comparisons and accumulated visual odometry. The result is a system that implicitly uses sequences of images to reduce localisation error when repeating a route without introducing dependencies that increase localisation error when large visual changes and variable speed repetitions are present.

This approach was validated through a series of experiments on two state-of-the-art outdoor visual teach and repeat datasets. TinyVPR descriptors were shown to reduce localisation error and decrease inference time when implemented in the particle filter compared to using state-of-the-art alternative NetVLAD descriptors. Furthermore the proposed combination of TinyVPR descriptors and particle filter was shown to significantly reduce localisation error in comparison to state-of-the-art alternatives for VT&R. A series of further experiments and analysis was done to highlight the robustness of the proposed approach to visual changes and variable repetition speeds.

A paper describing an early version of the proposed TinyVPR descriptor was accepted for peer-reviewed publication at UK Robotics & Autonomous Systems Conference in 2020. Further evidence of the significance and novelty of this chapter's contribution can be found in subsequent work which has used similar approaches. For example, Xu et al. use a particle filter with visual odometry for VPR [142], rather than VT&R, for localisation of a single place image using a series of images, Gridseth et al. also train a CNN to predict keypoints and descriptors for appearance invariant VT&R [206]. Most recently Rozsypálek et al. [207] used contrastive loss to teach a

CNN to generate appearance invariant VPR descriptors also for VT&R and Garg et al. formulate a triplet loss specifically for VPR using sequences of images [208].

3.4.1 Limitations and Future Work

One limitation of this approach is that the parameters for the particle filter are difficult to estimate and can significantly affect performance. Section 3.2.2 specifies three normal distributions for estimating the likelihood of a correct localisation prediction using the VPR descriptor, the likelihood of correct visual odometry information and the distance moved along the route to update each particle with at each time step. These parameters are dependant on a number of different factors such as reliability of the VRP descriptor and visual odometry, given the difficulty of the route and the likely speed of the traversal. The mean and standard deviation of these distributions can be roughly estimated but does ideally require some prior knowledge, such as likely speed of the traversal. For example, if the movement update is not large enough the particles in the filter will not be able to keep up with route repetition and the approach will fail. In the future the reliability of the VPR descriptor could be integrated into the filter directly by using a Bayesian DNN [199] that would generate a descriptor with an associated uncertainty.

Another potential limitation is that the data gathered during the teaching phase may not be sufficiently varied to train the VPR descriptor, preferably fewer teach traversals of a route would be required to train TinyVPR to generate appropriate descriptors. One shot learning focuses on a very constrained training set to learn a task and may be able to address this. For deep learning generally more training data is better, which suggests as many frames from the route's reference videos should be sampled as possible. However, the reference data is also compared against the test frames which can take a long time if there is a lot of reference data, or the descriptors are very large. An advantage of the particle filter is that the descriptor from each candidate particle only needs to be compared against the test frame's descriptor which increases efficiency over comparing against every reference descriptor in a global search.

Traditional visual teach and repeat includes a control system for physically repeating the route, but focus on this has reduced due to the significant increase in time required to develop and test this, particularly outdoors. One potential area of future work would be to implement and evaluate this approach on a physical robot with

the aim of releasing this approach as an open source Robot Operating System (ROS) package. Implementing this approach in real-world conditions opens up further challenges for the robot leaving the pre-defined route and getting lost. This scenario of navigation in an open environment is explored in subsequent chapters.

Chapter 4

OpenSceneVLAD: Appearance Invariant, Open Set Scene Classification

4.1 Introduction

Scene classification is a well-established area of computer vision research that aims to classify a scene image into pre-defined categories such as playground, beach and airport. Research into this topic has applications in areas such as content based image retrieval [152], robot navigation [153] [154] and disaster detection [155].

Current approaches use deep neural networks to focus on increasing the variety of pre-defined categories for classification. For example, state-of-the-art datasets can include up to 365 different scene classes [156] but have so far failed to consider two major challenges which, to the author's knowledge, are addressed for the first time in this chapter:

- Invariance to changes in scene appearance due to lighting, weather and seasonal conditions.
- Open set classification to classify scene images as one of the training classes, or an 'open' class undefined at training time.

Scene classification is important for enabling intelligent robotic navigation. For example, a service robot may need to identify a steep flight of stairs where a person

with reduced mobility needs assistance, or an autonomous car may need to identify a pedestrian crossing and drive more cautiously. Appearance invariance is important because external influences such as weather and lighting are an unavoidable aspect of outdoor scene classification and can dramatically affect accuracy. Recent research into the related field of visual place recognition (VPR) has focused heavily on appearance invariance [10, 101, 13] but VPR is an image retrieval task which is at odds with image classification. Specifically, image retrieval aims to create consistently unique image descriptor representations for comparing a query image against a reference set of images to find an appropriate match, this is a very different task to image classification which aims to associate images with a single class so approaches for the two tasks cannot trivially be combined.

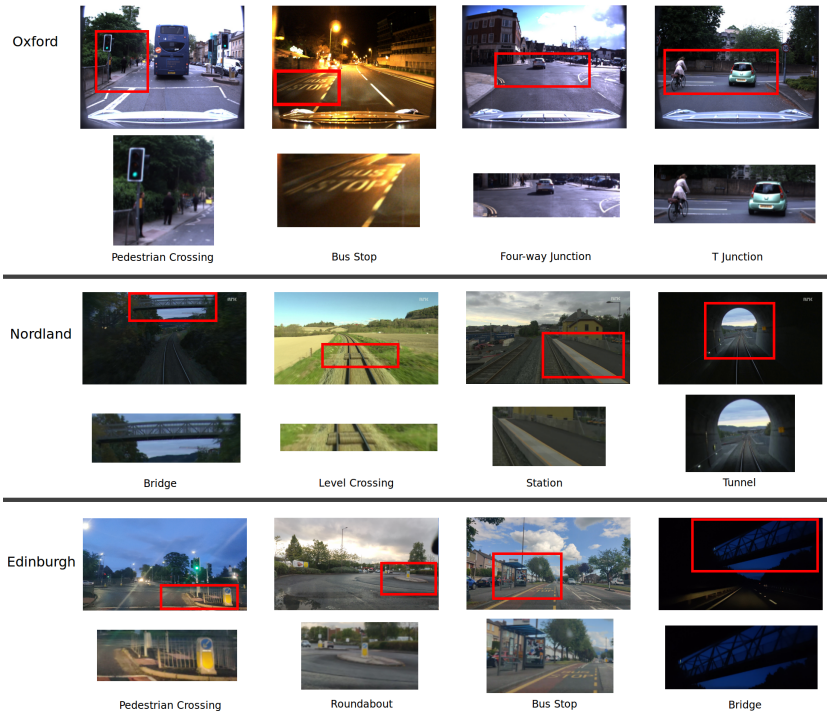


Figure 4.1: Examples of each dataset's classes and the complexity of their appearance variations, as listed in Table 4.1. Oxford (overcast, night, sunny, overcast), Nordland (spring, winter, summer, spring) and Edinburgh (evening, overcast, sunny, evening). Class relevant image sections are highlighted in red and enlarged for the reader's convenience.

In real world scenarios an autonomous robot needs to identify known scene classes of interest from a continuous environment of unknown scene classes, this task is an example of open set recognition (OSR). OSR is defined as recognising test data as a member of an unknown open class that is not defined in the training dataset. In

contrast closed set data is previously unseen examples of known classes. State-of-the-art approaches typically only aim to discriminate between open and closed set data [181] and limit themselves to simple computer vision datasets such as CIFAR-10 and MNIST [190] [191]. The work presented here considers this problem on a significantly more challenging scene classification dataset. Furthermore, this chapter frames the problem as an $N+1$ classification problem, where N is the number of closed set classes and the open set class is defined as the $+1$ class, meaning classification of closed set data and recognition of open set data is attempted simultaneously, which is defined here as 'open set classification' (OSC).

This chapter contains three main contributions to address the two previously highlighted challenges. Firstly, a new scene classification dataset that includes visual variation and open set data is synthesised from large-scale visual localisation datasets. Specifically, four different scene classes (Figure 4.1) are identified within each dataset and the remaining images are designated as an open class (Figure 4.2) across three outdoor localisation datasets: Oxford RobotCar [115], Nordland [133] and a third Edinburgh dataset collected specifically.

Figure 4.2: Examples of each dataset's open class in different appearance variations, as listed in Table 4.1 showing the potential overlap with the scene classes shown in Figure 4.1. Oxford (overcast, night, sunny, overcast), Nordland (spring, winter, summer, spring) and Edinburgh (evening, overcast, sunny).

Secondly, scene classification and visual place recognition CNNs are fused to create SceneVLAD and then trained for appearance invariant scene classification on each dataset's four scene classes. For comparison scene classification and visual place recognition networks are trained separately for this task.

Finally the second contribution is extended to open set classification by using intra-class splitting which modifies SceneVLAD's architecture and training approach to create OpenSceneVLAD. For evaluation open set images are added to the test data

and OpenSceneVLAD's classification performance is compared with SceneVLAD and baseline scene classification networks using confidence thresholding to identify the open class.

Experiments show the proposed SceneVLAD network increases mean appearance invariant scene classification F1 scores by up to 0.07. For open set classification experiments show improvements in mean F1 classification scores between baseline scene classification and SceneVLAD of up to 0.11 and between baseline scene classification and OpenSceneVLAD of up to 0.22. In summary, this chapter's contributions are:

1. A visually variant, open set scene classification dataset that is made publicly available [1], containing 4 scene classes derived from three traversals of each of the Oxford RobotCar, Nordland and our Edinburgh datasets. The raw Edinburgh visual place recognition dataset covering a 9km traversal of different environments around Edinburgh in three different visual conditions with GPS data is also released (Section 4.2.1).

2. A 'SceneVLAD' CNN that combines scene classification and visual place recognition for improved appearance invariant scene classification (Section 4.2.2).

3. An investigation into the significance of open set scene classification as a problem and an extension to SceneVLAD 'OpenSceneVLAD' that uses intra-class splitting for open set classification (Section 4.2.3).

4.2 Methodology

In this section, firstly the method used to generate the scene class data is outlined. Secondly, an approach for combining scene classification and visual place recognition for visually invariant scene classification is described. Finally, an extension is proposed to SceneVLAD to enable open set, visually invariant scene classification.

4.2.1 Scene Class Labelling

This is the first attempt at appearance invariant scene classification so there are no pre-existing datasets available for this task. This chapter's first contribution is therefore a scene class dataset compiled from three large outdoor visual localisation datasets.

[1] https://github.com/WHBSmith

These datasets cover routes around large geographical areas, each traversal of the route is done in different appearance conditions (Figure 4.1) which makes them perfect for extracting scene classes in a variety of different conditions, the remaining images provide a challenging open set of images.

Table 4.1: Dataset, appearance condition traversals used for training and testing with average number of frames per scene class, per traversal.

Dataset	Traversals	Scene Class	Frames
Oxford	Overcast (*train*)	Pedestrian Crossing	170
	Night (*test*)	Bus Stop	129
	Sunny (*test*)	Four-way Junction	152
		T-junction	103
		Open Set	2739
		Total	**3293**
Nordland	Spring (*train*)	Bridge	81
	Winter (*test*)	Level Crossing	87
	Summer (*test*)	Station	292
		Tunnel	186
		Open Set	3970
		Total	**4626**
Edinburgh	Evening (*train*)	Pedestrian Crossing	53
	Overcast (*test*)	Roundabout	135
	Sunny (*test*)	Bus Stop	46
		Bridge	47
		Open Set	843
		Total	**1124**

The first dataset is the 9km urban Oxford RobotCar dataset [115]. The three traversals used are: 2015-05-22-11-14-30 (overcast), 2014-12-16-18-44-24 (night) and 2015-03-24-13-47-33 (sunny). The second dataset is Nordland [133], a 763km train journey through rural Norway. The traversals used are: spring, winter and summer. The final Edinburgh dataset was collected from three traversals of 19km of urban, rural and motorway environments: 20210524 (overcast), 20210526 (evening) and 20210804 (sunny) using a dash-mounted OnePlus 7T recording 4k video at 30fps and a GPS logger app. For each dataset one is randomly chosen for training/reference, the remaining two are used for testing (Table 4.1) and these are fixed for all experiments. Each dataset was sampled using a minimum distance between consecutive frames (0.1, 80 and 2 meters for each dataset respectively) to prevent oversampling single scenes, for example when the raw video data stops at a pedestrian crossing.

For each dataset four scene classes were identified based on how frequently they occurred, how evenly they were spread across the dataset and their potential significance for tasks such as autonomous navigation. Images from each dataset's traversal were then labelled according to each scene class, the remaining unlabelled images comprise that traversal's open set. In dynamic environments discretion is required

when hand-labelling scenes, but all possible efforts were made to select the labels consistently. Samples of the scene class data and open set data are shown in Figure 4.1 and Figure 4.2, but for completeness a brief description of each is provided below:

- **Oxford**
  - Pedestrian crossing: any zebra crossing or set of traffic lights dedicated to pedestrian crossing.
  - Bus stop: a bus stop on either side of the road with associated road markings.
  - Four-way junction: any perpendicular crossroads.
  - T-junction: joining a road perpendicular to the current one.
- **Nordland**
  - Bridge: passing underneath a pedestrian bridge.
  - Level crossing: a crossing for pedestrian or vehicle over the railway tracks.
  - Station: any stop with raised passenger platform.
  - Tunnel: approaching and passing through a tunnel with daylight visible at all times.
- **Edinburgh**
  - Pedestrian crossing: any zebra crossing or set of traffic lights dedicated to pedestrian crossing.
  - Roundabout: view and traversal of a roundabout.
  - Bus Stop: any bus stop on either side of the road with a bus shelter.
  - Bridge: passing underneath a vehicle bridge.

A further summary of the scene classes and their frame counts is shown in Table 4.1. Note that some scene frames represent multiple views of a single scene instance. For example, two consecutive scene frames may represent the approach and traversal of the same roundabout, it is difficult to state the exact number of unique scenes as they can overlap but a single scene is rarely represented by more than five frames.

4.2.2 SceneVLAD: Appearance Invariant Scene Classification

4.2.2.1 Basic Hypothesis

Scene classification neural networks are not taught to consider appearance invariance, but neural networks taught to generate VPR descriptors are, however the two tasks are framed very differently which makes combining the output from both difficult. For example, classification outputs a single class given a query image, whereas VPR is an image retrieval task which outputs an embedded image descriptor to compare against a reference set of image descriptors. The hypothesis of this chapter is that incorporating a visually invariant representation of a scene image into the final layers of a scene classification network encourages a visually invariant scene representation to be learnt.

4.2.2.2 Architecture

To combine two neural networks, one taught for classification and the other for descriptor generation (image retrieval), inspiration is taken from recent work on supervised [209] and self-supervised [210] contrastive loss.

Contrastive loss is a similarity learning loss function that Khosla et al. [209] adapt to partially train a network on a 'pretext task' to represent data in a way that would be helpful as input to a later part of the network taught on a 'downstream task', in this case classification. Usually the pretext task and downstream task would be the same, but in this case the objective is to combine the knowledge from two different tasks so NetVLAD VPR descriptors [10] are used, which have been shown to generalise well to a variety of environments and remain the backbone of state-of-the-art approaches [13]. NetVLAD consists of a VLAD layer appended to a partially frozen VGG16 network pre-trained on the ImageNet dataset [12] and is retrained for appearance invariant VPR using triplet loss, a variation of contrastive loss, rather than scene classification. A downstream classifier would normally then be appended and trained on the task, but as the pretext and downstream tasks are different it is fused with the lower levels of an entire scene classification network and the whole network is partially re-trained for scene classification.

In parallel the same training image is passed to the '365' or '1365' network pre-trained on scene classification [156]. '365' is taught to classify 365 scene classes from

the Places datasets [156] and '1365' is taught to classify 365 Places and the 1000 ImageNet classes [106]. These CNNs were selected because of their performance on a wide variety of scene classes, the availability of their pre-trained weights and the ease with which they can be integrated into the proposed architecture. The classification network's final layer is removed, the top 16 layers frozen and the 365 or 1365 dimensional output is concatenated with NetVLAD's. A small, trainable 1x1 convolutional layer is added before concatenation to reduce the NetVLAD descriptor output dimensions from 4096 to 256 to make them more comparable in size to the output of the scene classification network. Finally, the output is passed through two fully connected layers of size 4096 to a 4-class softmax classification layer (Figure 4.3). This architecture forces SceneVLAD to learn scene classification with respect to NetVLAD's appearance invariant image descriptors, thereby improving appearance invariance for scene classification.

The number of filters used for dimension reduction, the number of final fully connected layers and their width were optimised using a grid search. 365 and 1365 networks were re-trained with varying amounts of layers frozen, and it was found that freezing the top 16 layers produced the best results in all cases. Additionally, using Khosla et al's supervised contrastive approach [209] to explicitly re-train NetVLAD was explored for this task. However, NetVLAD is only taught for appearance invariance down to the conv5 layer so retraining these layers significantly compromised its performance for the task. The width of the NetVLAD dimension reduction layer was varied to allow multiplication of the two networks' output rather than concatenation, as suggested by the literature [162], but this did not improve performance.

SceneVLAD was then evaluated (Section 4.3.1) before it was expanded to consider an open set scenario.

4.2.3 OpenSceneVLAD: Open Set Appearance Invariant Scene Classification

4.2.3.1 Basic Hypothesis

Scene classification for use in scenarios such as visual navigation requires rejection of unknown scenes that cannot be defined at training time, this problem is known generally as open set recognition. Current approaches for this task use relatively simple data [190] [191], the hypothesis explored in this section is that these techniques can be

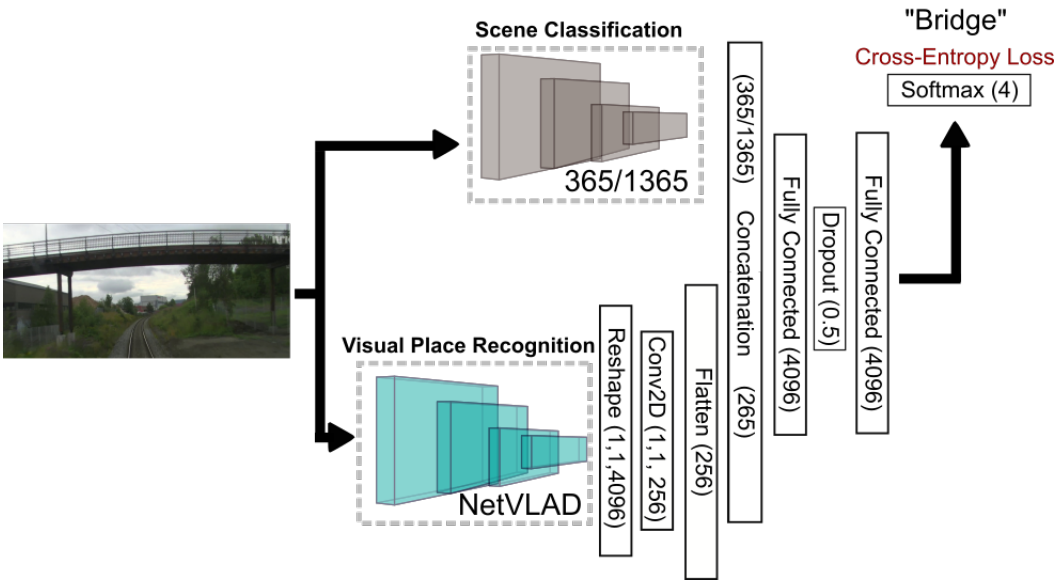


Figure 4.3: SceneVLAD network architecture fusing of scene classification and visual place recognition for appearance invariant scene classification.

used on more complex scene data. Using these techniques SceneVLAD can therefore be extended to a new network, OpenSceneVLAD, that can reject incoming images from a set of unknown classes. To do this intra-class splitting is used which trains SceneVLAD normally, identifies and gives a secondary label of a potential open class to the training data from each class that is incorrectly classified, or correctly classified, but with a low confidence. The network is then re-trained to simultaneously classify the original classes and the open class. An alternative approach that uses a statistical test to identify images from a potential open class, called openmax, is also explored as an alternative.

4.2.3.2 Openmax Layer

The state-of-the-art openmax layer [21] introduced in Chapter 2 leverages the Extreme Value Theory (EVT), a branch of statistics that models the likelihood of extreme events using the tails of probability distributions, to identify open set data. The intuition behind this approach is that the penultimate layer of a classification network produces activation vectors of a class, the distances from a mean activation vector for that class can then be fit to probability distributions for each class. An activation vector of an open set image is assumed and its distance from the mean activation vector of the known classes can be used to calculate whether it is likely to be a member of one of

the pre-defined classes. The end result is a layer that replaces a softmax layer (the softmax layer in Figure 4.3) with one extra output to represent the open set with no further training.

Calculation of Mean Activation Vectors. A mean activation layer for each class is calculated using penultimate activation values. Given an input, **x**, each element of the penultimate activation values for N classes is given as $\mathbf{v}(\mathbf{x}) = v_1(\mathbf{x}) \dots v_N(\mathbf{x})$. For each class, j, let $S_{i,j} = v_j\left(x_{i,j}\right)$ and for each correctly classified training example $x_{i,j}$ the mean activation vector can be calculated $\mu_j = \text{mean}_i\left(S_{i,j}\right)$.

Distribution Fitting. For each class j a Weibull distribution is fit to the largest η distances from the mean activation vector to find a limit on the likely representation of a known class from its mean representation. A value of $\eta = 20$ was used as specified in the original work. The use of a Weibull distribution for EVT has been established by previous work on meta-recognition [211]. To fit the Weibull distribution libMR's FitHigh [211] function is used $\rho_j = (\tau_j, \kappa_j, \lambda_j) = \text{FitHigh}\left(\left\|\hat{S}_j - \mu_j\right\|, \eta\right)$, where τ_i is for shifting the data and κ_i, λ_i are the Weibull shape and scale parameters.

Inference. Openmax then recalibrates each activation value of the penultimate layer by estimating the Weibull CDF probability on the distance between a test sample x each known class' mean μ_i activation vectors, which servers as the core of the rejection estimation. This function is only assumed to provide a meaningful probability for the classes that are predicted most likely as the input has already been established as being relatively dissimilar to the remaining predicted classes. The weights for the α largest activation classes are then calculated and used to scale the Weibull CDF probability and compute a revised activation vector with the top scores changed (line 6 of Alg. 2). The α parameter was set using a grid search to 4. A final classification prediction is then made using Equation 4.1). A comprehensive description of Openmax can be found in [21] .

4.2.3.3 OpenSceneVLAD

Intra-class splitting identifies poorly classified images from pre-defined classes and splits them into typical and atypical examples. Atypical examples are given secondary labels as members of a potential open class and the original training data, with its new secondary labels, is used to re-train the network for classification with an extra open class. The intuition being that by selecting some 'atypical' images from the original

Algorithm 2: Openmax probability estimation for open set classification

Require: number of classes N.
Require: activation vector for a test input $\mathbf{v}(\mathbf{x}) = v_1(x) \dots v_N(x)$
Require: a mean activation vector μ_j and associated Weibull distribution $\rho_j = (\tau_j, \kappa_j, \lambda_j)$ for each class.
Require: α the number of most confidently predicted classes to revise.
Let $s(i) = argsort(v_j(x))$; Let $\omega_j = 1$
for $i = 1, \dots, \alpha$ **do** calculate the weights for each class
    $\omega_{(i)}(x) = 1 - \frac{\alpha - i}{\alpha} e^{-\left(\frac{||x - \tau_{s(i)}||}{\lambda_{s(i)}}\right)^{\kappa_{s(i)}}}$
end for
Recalibrate the activation vector. $\hat{v}(x) = \mathbf{v}(\mathbf{x}) \circ \omega(\mathbf{x})$
Define $\hat{v}_0(x) = \sum_i v_i(x)(1 - \omega_i(x))$

$$P(y = j \mid \mathbf{x}) = \frac{e^{\hat{\mathbf{v}}_j(\mathbf{x})}}{\sum_{i=1}^{N+1} e^{\hat{\mathbf{v}}_\mathbf{i}(\mathbf{x})}} \tag{4.1}$$

Let the openmax open class be $N + 1$ and $y^* = \text{argmax}_j P(y = j|\mathbf{x})$
if $y^* == N + 1$ or $P(y = y^*|\mathbf{x}) < \epsilon$, where ϵ is a user-defined confidence threshold **then** the input is predicted to be the open set class.
end if
else predict one of the closed set classes, y^*.

training dataset as members of a potential open set some basic assumptions about likely open set scene images could be leveraged, such as position of the ground plane and orientation of ambient scenery for improved performance over openmax. Intra-class splitting is summarised in Algorithm 3.

Identify Atypical Class Examples. Firstly, SceneVLAD is trained for closed set classification on N number of classes using scene images x_i and the corresponding closed set labels $y_{i,cs}$. The trained network is then used to classify the training images. A user defined proportion of images per class, in this case 30% [190], that cannot be classified correctly, or are classified with the lowest confidence are identified as 'atypical' examples. A small change from the original technique was made by identifying atypical images proportionally across all training classes. This was necessary because of the small class sizes and class imbalance.

Generate Open Set Labels. A new set of scalar labels is then generated $y_{i,os}$. For every image, x_i, if deemed atypical $y_{i,os} = N + 1$, otherwise $y_{i,os} = y_{i,cs}$.

Create Architecture and Re-train. Two separate softmax layers are used as network outputs (Figure 4.4), one with N outputs trained for closed set regularization using cross-entropy loss $\mathcal{L}_{cs}$ (Equation 4.4), the other with $N + 1$ outputs trained for

OSC also using cross-entropy loss $\mathcal{L}_{os}$ (Equation 4.3). OpenSceneVLAD is then trained using both losses with the corresponding labels of each image. Closed set regularisation helps maintain a high closed-set accuracy by forcing the atypical samples to be correctly classified to their original classes, this prevents the network learning to classify all data as the open class, for example. A user determined scaling factor γ, is applied to $\mathcal{L}_{cs}$ and the two losses are then minimised as a joint optimisation problem (Equation 4.2).

Loss Functions The objective of intra-class splitting is a joint optimisation problem consisting of two individual loss terms for the open set layer and closed set layer

$$\mathcal{L} = \mathcal{L}_{os} + \gamma * \mathcal{L}_{cs}, \tag{4.2}$$

where $\mathcal{L}_{os}$ is the loss function for the open set layer and $\mathcal{L}_{cs}$ is the loss function for the closed set layer. γ is a hyperparameter to tune the focus on closed set regularisation.

Let B be the minibatch size during training. Moreover, $1_{y_i \in y^{(n)}}$ is an indicator function which returns 1 if a given sample x_i with a scalar label y_i belongs to the class $y^{(n)}$ and otherwise returns 0. Based on these notations $\mathcal{L}_{os}$ is a simple $(N+1)$-class categorical cross-entropy loss

$$\mathcal{L}_{os} = -\frac{1}{B}\sum_{i=1}^{B}\sum_{n=1}^{N_{os}} 1_{y_i \in y^{(n)}} \log[P(\hat{y}_i \in y^{(n)})] \tag{4.3}$$

where $N_{os} = N + 1$ and $P(y_i \in y^{(n)})$ denotes the predicted probability that sample x_i belongs to the class $y^{(n)}$, i.e. the value of the n-th element of the network's output vector.$\mathcal{L}_{cs}$ is an N-class categorical cross entropy loss

$$\mathcal{L}_{cs} = -\frac{1}{B}\sum_{i=1}^{B}\sum_{n=1}^{N_{cs}} 1_{y_i \in y^{(n)}} \log[P(\hat{y} \in y^{(n)})], \tag{4.4}$$

where $\mathcal{L}_{cs}$ shares the same notation as $\mathcal{L}_{os}$ and $N_{cs} = N$ is the number of the given known classes.

Algorithm 3: Algorithm for training OpenSceneVLAD.

Train SceneVLAD, as detailed in Section 4.2.2 on N classes, with I scene images using softmax loss. Each scene image, $\mathbf{x_i}$ has a corresponding label $y_{i,cs}$
for $i = 1, \ldots, I$
    Use SceneVLAD to calculate each image's ($\mathbf{x_i}$) class probability prediction $P(y = j \mid \mathbf{x_i})$ using softmax.

$$P(y = j \mid \mathbf{x_i}) = \frac{e^{j(\mathbf{x_i})}}{\sum_{k=1}^{N} e^{k(\mathbf{x_i})}} \tag{4.5}$$

end for
for $i = 1, \ldots, I$
    if $\mathbf{x_i}$ with class label $y_{i,cs} = n$ is in class n's 30% incorrect, or lowest confidence predictions, **then** let open set label $y_{i,os} = N + 1$
    else $y_{i,os} = y_{i,cs}$.
    end if
end for
Create OpenSceneVLAD architecture (Figure 4.4) and train on $N + 1$ classes, with I scene images by minimising the loss function in Equation 4.2.

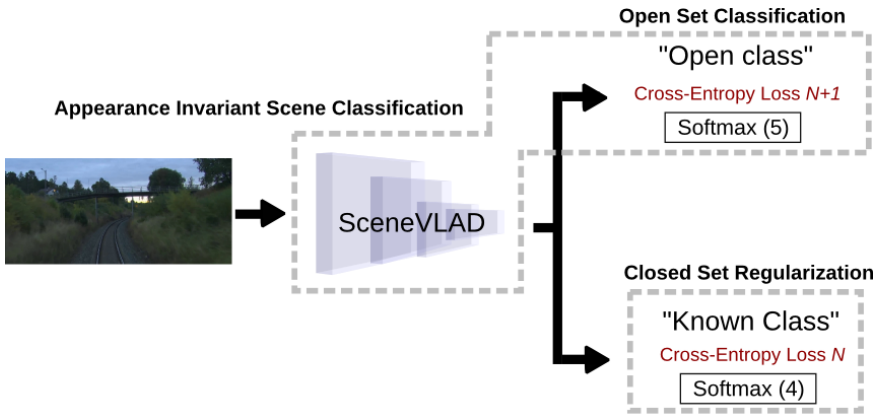


Figure 4.4: OpenSceneVLAD network architecture, demonstrating training using intra-class splitting on an atypical image.

Inference. At test time the closed set regularization is removed and output from the remaining open set layer is used for open set classification, as shown in Algorithm 4 and Figure 4.5.

Algorithm 4: Algorithm for using OpenSceneVLAD for inference.

1: A test image, $\mathbf{x}$, has a label, y from $N+1$ classes, including the open class.
2: Use OpenSceneVLAD to calculate the test image class probability prediction $P(y = j \mid \mathbf{x})$ using Equation 4.6.
3:

$$P(y = j \mid \mathbf{x}) = \frac{e^{\hat{\mathbf{v}}_j(\mathbf{x})}}{\sum_{i=1}^{N+1} e^{\hat{\mathbf{v}}_\mathbf{i}(\mathbf{x})}} \tag{4.6}$$

4: Let the open class be $N+1$ and $y^* = \text{argmax}_j P(y = j \mid \mathbf{x})$
5: **if** $y^* == N+1$ or $P(y = y^* | \mathbf{x}) < \epsilon$, where ϵ is a user-defined confidence threshold the input is predicted to be the open set class.
6: **else** predict one of the closed set classes, y^*.
7: **end if**

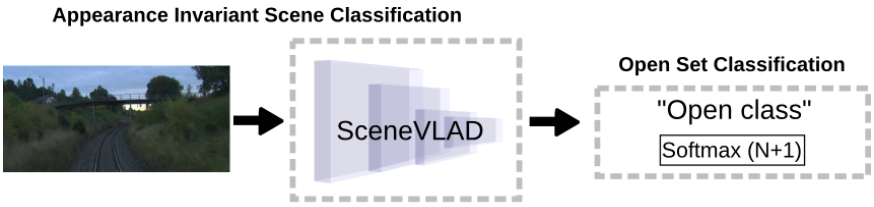


Figure 4.5: Inference using OpenSceneVLAD with the output softmax layer with an extra output for the open class ($N+1$), given N original closed classes.

4.2.3.4 OpenSceneVLAD + Openmax

The final contribution of this chapter is the combination of OpenSceneVLAD with an openmax layer (Figure 4.6). Openmax is designed to use out-of-distribution activation vectors to identify images from an open set, it is therefore well suited to scenarios where open set images are likely to include images that create activation vectors far away from the mean activation vectors of the known classes. Environments where known scene classes are quite distinct from their surroundings. In contrast intra-class splitting uses 'atypical' examples of known classes to learn to recognise examples of a potential open set class so is better suited to scenarios where the the open set images are more likely to overlap with the known classes because it has been trained to recognise open set images using parts of the original training data. The intuition behind this combination is that it can be used to account for both types of open set data more effectively.

The two approaches are combined to formulate open set classification as an $N+2$ classification problem, where N is the original number of closed set classes and intra-class and openmax open set classification contribute an output each for the open set as shown in Algorithm 5.

Algorithm 5: Algorithm for training OpenSceneVLAD + OpenMAX and using it for inference.

1: Train OpenSceneVLAD (Algorithm 3).
2: At inference time apply the openmax layer to $N+1$ classes (Algorithm 2 lines 1 - 11), to create $N+2$ outputs.
3: Let the openmax open class be $N+1$, the intra-class open class be $N+2$ and $y^* = \text{argmax}_j P(y=j|\mathbf{x})$.
4: **if** $y^* > N$ or $P(y=y^*|\mathbf{x}) < \epsilon$, where ϵ is a user-defined confidence threshold the input is predicted to be the open class.
5: **else** predict one of the closed set classes, y^*.
6: **end if**

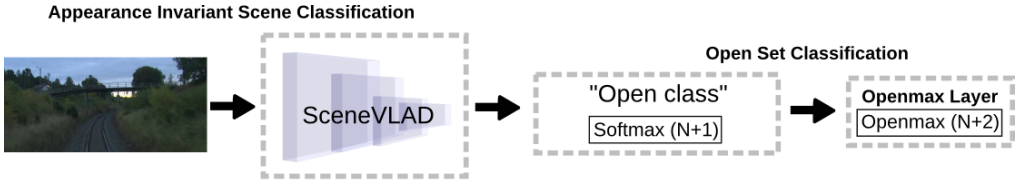


Figure 4.6: Inference using OpenSceneVLAD with the output softmax layer with an extra output for the open class ($N+1$) and openmax layer with another extra output ($N+2$), given N original closed classes.

4.3 Evaluation

In this section an evaluation of the three proposed methods is done on the collected datasets. The experiments take the form of an extended ablation study. The proposed contributions are systematically added to the scene classification CNN to specifically demonstrate that SceneVLAD improves appearance invariance and then open set classification.

4.3.1 SceneVLAD: Appearance Invariant Scene Classification

The purpose of this experiment was to test the hypothesis that SceneVLAD's fusion of NetVLAD VPR descriptors with scene classification networks 365 or 1365 improves appearance invariance for scene classification.

4.3.1.1 Baselines

For a baseline the VPR and scene classification networks that make up SceneVLAD (NetVLAD, 365 and 1365) were trained individually for scene classification on each dataset (Section 4.2.1). Each network has the same architecture as they do in SceneVLAD (Figure 4.3) before the concatenation layer, simply with a softmax layer appended. The

top 16 layers of networks 365 and 1365 were frozen for training and NetVLAD was frozen up to its final normalization layer. The key difference between 365 and 1365 is that 1365 was also taught for general image classification, as well as specific scene classification.

4.3.1.2 SceneVLAD

The base networks are then combined into two versions of SceneVLAD using the architecture described in Section 4.2.2.2, each based on different scene classification networks: SceneVLAD365 (NetVLAD+365) and SceneVLAD1365 (NetVLAD+1365). For SceneVLAD the top 16 layers of networks 365 and 1365 were also frozen for training and NetVLAD was frozen up to its final normalisation layer.

4.3.1.3 Training

All 5 networks were trained to classify the four scene classes from the training traversal of each dataset using the same approach: image augmentation (featurewise normalisation, width and height shifting, horizontal flipping and brightness changes), a minibatch size of 8, a learning rate of 1e-5 and a stratified training/validation split of 80/20. Training was done for 100 epochs with early stopping using a patience of 20. For early stopping validation classification accuracy was monitored rather than loss because the validation data was not representative of the test data's visual conditions so a coarser measure of training success was needed to prevent overfitting. For each dataset the network was trained five times and the mean of the results was calculated.

4.3.1.4 Testing

The classifiers were tested on the scene classes from the remaining two traversals of the same dataset. Because the traversals cover the same area the instances of the scene classes remain similar but only the visual variations change, which allows any increase in performance to be attributed solely to improved appearance invariance. For evaluation the F1 score is used to compensate for the class imbalance. The F1 scores for each class are reported to specifically highlight the score differences between the base scene classification network and SceneVLAD in Table 4.2, which shows a mean increase in F1 score between 0.04 - 0.10 for SceneVLAD versus the base scene classification networks. These results are further summarised across all datasets in Table 4.3,

which shows a mean increase in F1 scores across all datasets of between 0.05 and 0.07 for SceneVLAD compared to the base scene classification networks.

Table 4.2: Mean F1 scores of appearance invariant scene classification comparing SceneVLAD's fusion of VPR and scene classification with its constituent parts trained on the same task. The standard deviation of the class means from each of the 5 trials is reported. Changes in mean F1 scores (Δ) are positive and negative.

	Oxford RobotCar					**Class**
	Ped. Cross	Bus Stop	Four Junct.	T Junct.	**Class Mean**	**Mean Std. Dev.**
NetVLAD	0.64	0.69	0.67	0.48	**0.62**	0.0063
365	0.67	0.43	0.63	0.63	**0.59**	0.024
SceneVLAD365	0.67	0.67	0.61	0.63	**0.65**	0.014
Δ	**0.00**	**0.24**	**-0.02**	**0.00**	**0.06**	
1365	0.67	0.44	0.67	0.65	**0.61**	0.019
SceneVLAD1365	0.67	0.62	0.70	0.68	**0.67**	0.016
Δ	**0.00**	**0.18**	**0.03**	**0.03**	**0.06**	
	Nordland					**Class**
	Bridge	Lvl. Cross.	Station	Tunnel	**Class Mean**	**Mean Std. Dev.**
NetVLAD	0.77	0.81	0.96	0.95	**0.87**	0.0097
365	0.84	0.69	0.96	0.92	**0.85**	0.11
SceneVLAD365	0.80	0.86	0.97	0.95	**0.89**	0.0095
Δ	**-0.04**	**0.17**	**0.01**	**0.03**	**0.04**	
1365	0.69	0.86	0.96	0.96	**0.87**	0.077
SceneVLAD1365	0.85	0.87	0.97	0.96	**0.91**	0.011
Δ	**0.16**	**0.01**	**0.01**	**0.00**	**0.04**	
	Edinburgh					**Class**
	Ped. Cross.	Round.	Bus Stop	Bridge	**Class Mean**	**Mean Std. Dev.**
NetVLAD	0.00	0.64	0.00	0.00	**0.16**	0.00
365	0.40	0.84	0.46	0.47	**0.54**	0.020
SceneVLAD365	0.48	0.80	0.65	0.47	**0.60**	0.028
Δ	**0.08**	**-0.04**	**0.19**	**0.00**	**0.06**	
1365	0.32	0.89	0.37	0.59	**0.54**	0.039
SceneVLAD1365	0.44	0.89	0.60	0.63	**0.64**	0.043
Δ	**0.12**	**0.0**	**0.23**	**0.04**	**0.10**	

Table 4.3: Summary of mean F1 scores of ablation study for appearance invariant scene classification comparing SceneVLAD's fusion of VPR and scene classification with its constituent parts trained on the same task. Changes in scores (Δ) are positive and negative.

	Dataset Class Average			
	Oxford	Nordland	Edinburgh	**Total Mean**
NetVLAD	0.62	0.87	0.16	**0.55**
365	0.59	0.85	0.54	**0.66**
SceneVLAD365	0.65	0.89	0.60	**0.71**
Δ	**0.06**	**0.04**	**0.06**	**0.05**
1365	0.61	0.87	0.54	**0.67**
SceneVLAD1365	0.67	0.91	0.64	**0.74**
Δ	**0.06**	**0.04**	**0.10**	**0.07**

4.3.1.5 Does SceneVLAD Improve Appearance Invariance?

The results in Table 4.3 show a mean increase in F1 scores of up to 0.07 across a variety of scenes using SceneVLAD compared to state of the art scene classification networks. This experiment's results demonstrate networks 365 and 1365, despite being taught in exactly the same way, are outperformed by SceneVLAD's fusion of them with NetVLAD in all but three individual classes (Table 4.2). As the only difference between SceneVLAD and the baseline scene classification networks is the addition of NetVLAD for training this confirms the hypothesis that combining the two does improve appearance invariance.

Table 4.2 shows NetVLAD's improved performance on specific classes translates directly to its addition in SceneVLAD, increasing F1 scores up to 0.24. However, any reductions in F1 scores are limited to a decrease of up to 0.04. For the Edinburgh dataset NetVLAD failed to classify all but one class. The final layers of NetVLAD are trained for appearance invariance so unfreezing them to improve training would remove any advantage the network has from its pre-training. However, SceneVLAD improves the F1 score of each class, apart from one, which indicates SceneVLAD's successful and necessary use of VPR as a pretext task - re-training more layers of NetVLAD would overwrite the part of the network that had been trained for appearance invariance.

4.3.1.6 How Does SceneVLAD Improve Appearance Invariance?

To understand in more detail how SceneVLAD improves appearance invariance t-SNE plots were used to investigate the inter-class relationships generated by SceneVLAD and the baseline scene classification networks. t-SNE [212] was chosen over PCA for visualisation because of its ability to represent non-linear relationships between data and the local relationships between data points, which is useful for assessing the tightness of each cluster - an important metric for examining a classifier's performance. However, PCA may have advantages because of its ability to preserve global relationships between different clusters of data points and warrants future investigation. Additionally t-SNE plots are not presented as results alone and are meant to support the other results with a visualisation.

Traditionally, the characteristics of a well-trained classifier are that it creates greater separability between class representations Figure 4.7 shows that SceneVLAD's class

representations are able to improve classification by doing this. For example, in Figure 4.7 the Oxford and Nordland t-SNE plots indicate that SceneVLAD is able to better distinguish bus stop and bridge/level crossing classes respectively, compared to standard scene classification. These results are further supported in Table 4.2. For the Edinburgh dataset SceneVLAD365 is able to better separate the bus stop and bridge classes, but the inter-class distance is reduced as a result. Meanwhile SceneVLAD1365 is able to distinguish between the bus stop and pedestrian crossing classes, which 1365 cannot. The t-SNE plots in Figure 4.7 show apparently separable classes, which is not always reflected in the F1 scores in Tables 4.3 & 4.2. This is a limitation of the 2D t-SNE plots which in this case tightly groups class predictions and prevents their overall separability being totally assessed, but still gives a good indication of the relative separability between different approaches. Evidence for this can be seen in Oxford 365/SceneVLAD365, Edinburgh 365/SceneVLAD365 and Oxford 1365/SceneVLAD1365 where classes are represented in very small clusters which are then spread out by SceneVLAD.

Heatmaps are used to further examine the difference between false classifications made by scene classifiers 365 and 1365 with SceneVLAD's true classifications on the same images. The classes investigated were those identified as having the biggest difference using the t-SNE plots: bus stops (Oxford), level crossings (Nordland) and pedestrian crossings (Edinburgh). To build the heatmaps Grad-CAM [213] was used which computes the gradient of the most likely predicted class with respect to the last convolutional layer, this is averaged and then multiplied with the last convolutional layer. The final result is normalised between 0 and 1 and then overlaid on the original image.

Figure 4.8 shows SceneVLAD activates more on more class relevant features than the 365 and 1365 scene classification networks, enabling correct classification that would otherwise be impossible. In Oxford bus stops are characterised by road markings and bus shelters which SceneVLAD is shown to activate more on than the scene classifiers. Nordland level crossings vary for use with large vehicles down to pedestrians, SceneVLAD has a higher activation on relevant areas such as the railway track for classification. Pedestrian crossings in the Edinburgh dataset vary between zebra crossings and larger crossings with dedicated traffic lights, SceneVLAD focuses more on the associated road markings and traffic lights.

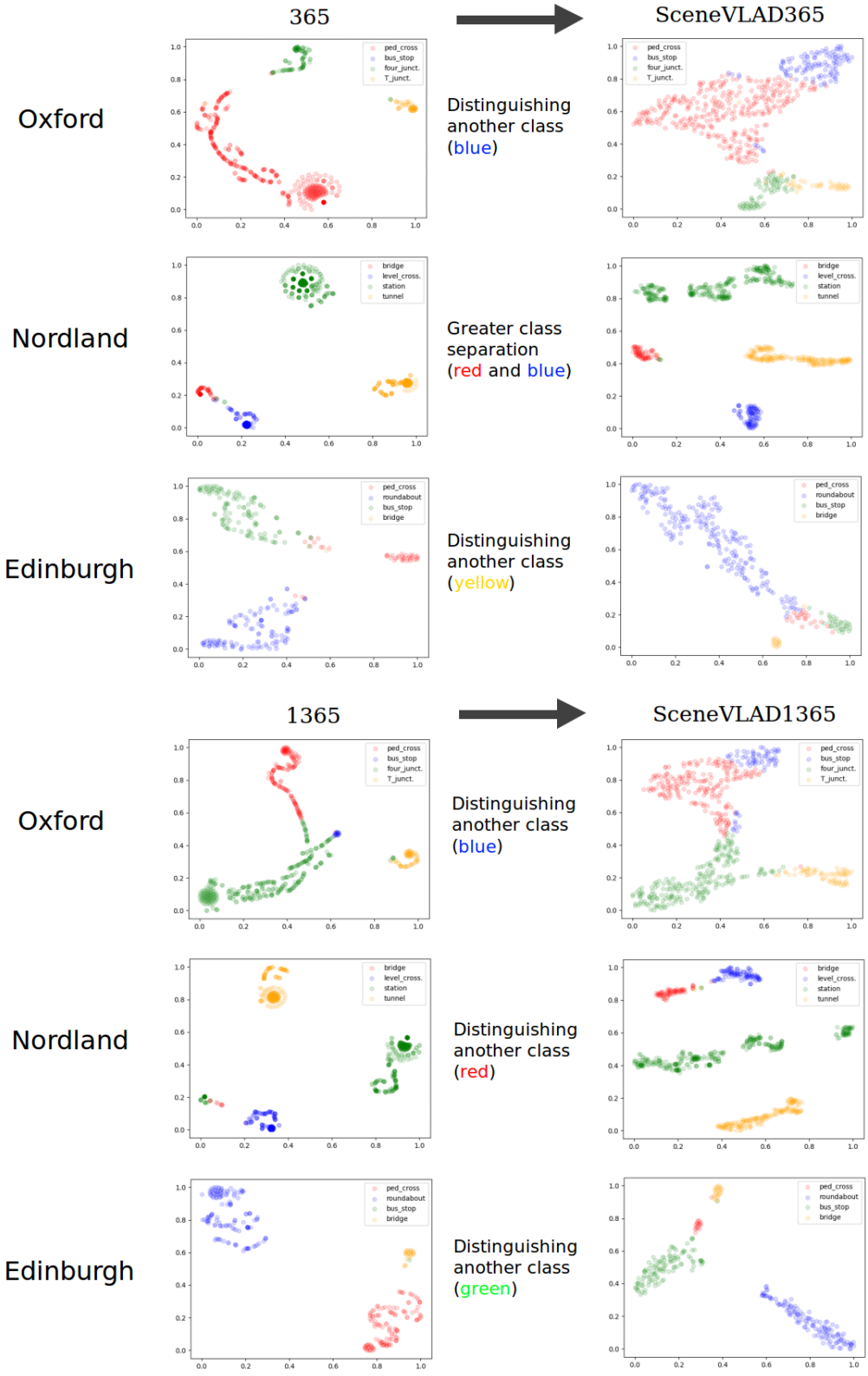


Figure 4.7: Samples of scene classification and SceneVLAD t-SNE plots from the test results in Table 4.2. SceneVLAD is shown to be able to distinguish between classes that baseline scene classification networks cannot.

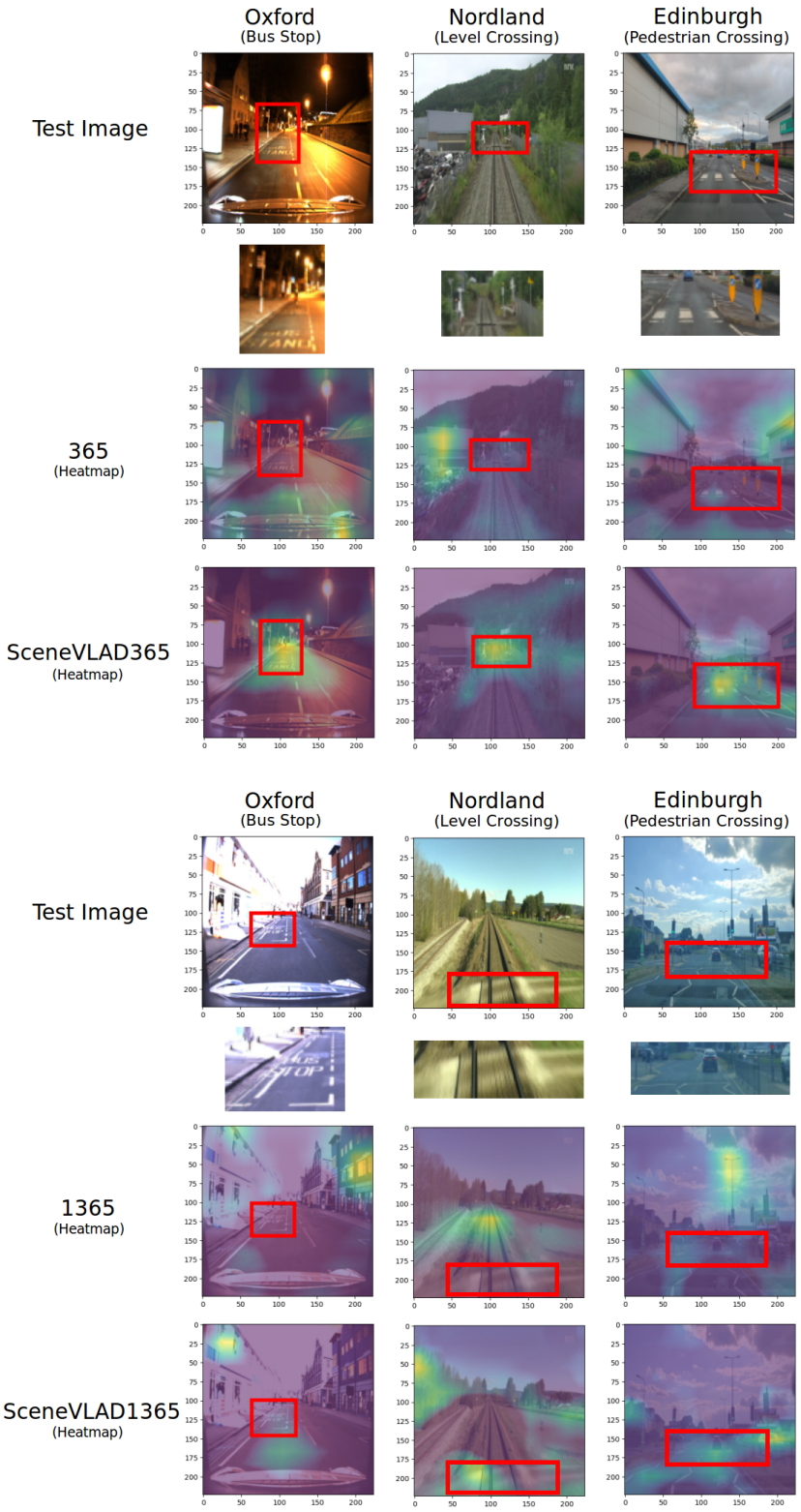


Figure 4.8: Samples of scene classification false positive and false negative classifications vs SceneVLAD true positives from the test results in Table 4.2. SceneVLAD's true positives are shown to be due to it focusing on more relevant features despite visual changes. Class relevant image sections are highlighted in red. Activation amounts range from low (purple) to high (yellow).

4.3.2 OpenSceneVLAD: Open Set Appearance Invariant Scene Classification

The purpose of this experiment was to test the hypothesis that scene classification was negatively affected by an open set scenario and to examine whether intra-class splitting used in OpenSceneVLAD could improve performance against the openmax and softmax baselines.

4.3.2.1 Baseline

For this experiment the best performing 1365 base scene classification from the previous experiment was compared with its version of SceneVLAD and OpenSceneVLAD with or without the state-of-the-art openmax [21] (Omax) layer used for open set classification and introduced in Section 4.2.3.2. To do this 10% of the test images for each class were used as a validation set for the test dataset to calculate a confidence threshold optimised to maximise the dataset class mean F1 score. In this way networks with outputs equal to the closed set number of classes (1365 and SceneVLAD) predict an open set image if the prediction confidence is below the calculated threshold and networks with an output that includes the open set (OpenSceneVLAD and any network with a final openmax layer) predict an open set image if the open set is predicted, or any prediction is below the calculated threshold. 1365 and SceneVLAD networks with a threshold of 0 (base) were also included. These networks are unable to classify any open set images, but were considered to examine how the open set scenario affects scene classification and whether confidence thresholding affects classification of known classes.

4.3.2.2 OpenSceneVLAD

An OpenSceneVLAD network is created using 365 or 1365 and NetVLAD with or without an openmax layer, per Section 4.2.2. As before, the top 16 layers of 1365 were frozen for training and NetVLAD was frozen up to its final normalization layer.

4.3.2.3 Training

All networks were trained following the same procedure in our previous experiment (Section 4.3.1.3), the only change being that OpenSceneVLAD was trained using intra-

class splitting (as described in Section 4.2.3.3), with 30% of training images being selected as 'atypical'.

Table 4.4: Mean F1 Scores of appearance invariant, open set scene classification using thresholding. Baseline scene classification networks are compared with SceneVLAD and OpenSceneVLAD with or without an openmax layer. Best class mean results are orange.

	Oxford RobotCar						
	Ped. Cross	Bus Stop	Four Junct.	T Junct.	Open	**Class Mean**	**Class Mean Std. Dev.**
365 - base	0.20	0.14	0.19	0.33	0.00	**0.17**	0.014
365	0.038	0.033	0.041	0.083	0.73	**0.19**	0.0046
365+Omax	0.00	0.00	0.00	0.00	0.91	**0.18**	0.00
SceneVLAD365	0.17	0.14	0.26	0.35	0.46	**0.27**	0.011
SceneVLAD365+Omax	0.19	0.15	0.31	0.35	0.76	**0.34**	0.0078
OpenSceneVLAD365	0.17	0.2	0.26	0.31	0.84	**0.36**	0.017
OpenSceneVLAD365+Omax	0.15	0.16	0.34	0.26	0.83	**0.35**	0.0084
1365 - base	0.15	0.12	0.14	0.23	0.00	**0.13**	0.0037
1365	0.10	0.12	0.18	0.32	0.27	**0.20**	0.0091
1365+Omax	0.00	0.00	0.00	0.00	0.91	**0.18**	0 .00
SceneVLAD1365	0.16	0.23	0.33	0.37	0.47	**0.31**	0.0016
SceneVLAD1365+Omax	0.19	0.13	0.40	0.38	0.78	**0.37**	0.015
OpenSceneVLAD1365	0.15	0.15	0.33	0.33	0.82	**0.35**	0.0049
OpenSceneVLAD1365+Omax	0.15	0.13	0.39	0.29	0.87	**0.36**	0.016

	Nordland						
	Bridge	Lvl. Cross.	Station	Tunnel	Open	**Class Mean**	**Class Mean Std. Dev.**
365 - base	0.32	0.053	0.51	0.32	0.00	**0.24**	0.044
365	0.20	0.038	0.35	0.39	0.36	**0.27**	0.0076
365+Omax	0.00	0.00	0.00	0.00	0.92	**0.18**	0.00
SceneVLAD365	0.46	0.075	0.61	0.49	0.29	**0.39**	0.017
SceneVLAD365+Omax	0.48	0.11	0.59	0.55	0.57	**0.46**	0.0092
OpenSceneVLAD365	0.50	0.17	0.65	0.73	0.89	**0.59**	0.013
OpenSceneVLAD365+Omax	0.43	0.22	0.61	0.76	0.92	**0.59**	0.016
1365 - base	0.14	0.05	0.47	0.23	0.00	**0.18**	0.014
1365	0.34	0.069	0.51	0.35	0.00	**0.25**	0.00962
1365+Omax	0.00	0.00	0.00	0.00	0.92	**0.18**	0.00
SceneVLAD1365	0.29	0.089	0.61	0.49	0.40	**0.38**	0.0016
SceneVLAD1365+Omax	0.37	0.14	0.59	0.56	0.70	**0.47**	0.013
OpenSceneVLAD1365	0.47	0.14	0.67	0.74	0.80	**0.56**	0.016
OpenSceneVLAD1365+Omax	0.48	0.18	0.6	0.83	0.88	**0.59**	0.016

	Edinburgh						
	Ped. Cross.	Round.	Bus Stop	Bridge	Open	**Class Mean**	**Class Mean Std. Dev.**
365 - base	0.15	0.43	0.21	0.29	0.00	**0.22**	0.017
365	0.090	0.36	0.17	0.25	0.17	**0.21**	0.025
365+Omax	0.00	0.00	0.00	0.00	0.86	**0.17**	0.00
SceneVLAD365	0.34	0.27	0.27	0.48	0.11	**0.29**	0.021
SceneVLAD365+Omax	0.26	0.38	0.11	0.00	0.72	**0.29**	0.020
OpenSceneVLAD365	0.25	0.34	0.22	0.13	0.53	**0.29**	0.024
OpenSceneVLAD365+Omax	0.37	0.33	0.25	0.0093	0.61	**0.31**	0.019
1365 - base	0.07	0.36	0.20	0.21	0.00	**0.17**	0.014
1365	0.13	0.34	0.12	0.26	0.17	**0.20**	0.023
1365+Omax	0.00	0.00	0.00	0.00	0.86	**0.17**	0 .00
SceneVLAD1365	0.17	0.65	0.074	0.32	0.35	**0.31**	0.0017
SceneVLAD1365+Omax	0.044	0.60	0.063	0.00	0.82	**0.31**	0.012
OpenSceneVLAD1365	0.27	0.67	0.15	0.28	0.66	**0.41**	0.0021
OpenSceneVLAD1365+Omax	0.19	0.66	0.11	0.0048	0.78	**0.35**	0.012

4.3.2.4 Testing

Testing was exactly the same as in Section 4.3.1.4 with one major difference: the test set was expanded to include all the open set images from each dataset's traversal, as seen in Table 4.1. The full results of this experiment can be seen in Table 4.4 and a summary of the results in Table 4.5. The individual class F1 scores are reported, but only for completeness as they are heavily affected by the choice of confidence threshold used for classification. In other words increasing the threshold value may allow more accurate classification of closed set classes, but the corresponding reduction in open set classification may result in an overall lower mean class F1 score. This leads to scenarios where the 365 scene classification network with an openmax layer (365+Omax) is reported with only the open class having a non-zero F1 score.

4.3.2.5 How Difficult is Open Set Scene Classification?

Base networks (-base) only attempt closed set classification but the results in Table 4.4 demonstrate that the addition of open set data significantly decreases the F1 score of even known classes, sometimes by more than 50%, confirming the hypothesis that the introduction of open set scene images makes this task very challenging due to false predictions introduced by the additional open set class. The addition of confidence thresholding to the base networks results in an improvement in open set detection but reduces the F1 score on the known classes even further, resulting in a similar class mean for all datasets. Some classes are affected more than others. For example, Nordland level crossings only have a slight difference in F1 score in closed set classification than tunnels (Table 4.2). However in open set classification the level crossing class is far more challenging, because that scene class overlaps more with the open class. In effect, level crossings look more like a generic railway scene than a bridge does.

4.3.2.6 Does OpenSceneVLAD Improve Open Set Scene Classification?

The results in Table 4.5 show SceneVLAD alone outperforms the original scene classification networks by an increase in mean F1 score of 0.10 (365) and 0.11 (1365). This suggests that the VPR descriptors improve open set classification as well as appearance invariance. The use of intra-class splitting in OpenSceneVLAD further increases performance over the original scene classification networks by mean F1 scores of 0.19

(365) and 0.22 (1365). This result is particularly significant given that the training data and majority of the underlying network architecture is the same used for SceneVLAD.

Table 4.5: Summary of mean F1 scores for appearance invariant, open set scene classification using confidence thresholding. Original baseline networks are compared with SceneVLAD and OpenSceneVLAD. Best total results are orange.

	Dataset Class Mean			
	Oxford	Nordland	Edinburgh	**Total Mean**
365 - base	0.17	0.24	0.22	**0.21**
365	0.19	0.27	0.21	**0.22**
SceneVLAD365	0.27	0.39	0.29	**0.32**
OpenSceneVLAD365	0.36	0.59	0.29	**0.41**
1365 - base	0.13	0.18	0.17	**0.16**
1365	0.20	0.25	0.20	**0.22**
SceneVLAD1365	0.31	0.38	0.31	**0.33**
OpenSceneVLAD1365	0.35	0.56	0.41	**0.44**

Table 4.6: Mean F1 scores for appearance invariant, open set scene classification using confidence thresholding. Original baseline networks, SceneVLAD and OpenSceneVLAD are compared when used with an openmax layer (+Omax), or not. Changes in mean F1 scores between best mean results (Δ) are positive and negative.

	Dataset Class Mean			
	Oxford	Nordland	Edinburgh	**Total Mean**
365	0.19	0.27	0.21	**0.22**
365+Omax	0.18	0.18	0.17	**0.18**
Δ				**-0.04**
SceneVLAD365	0.27	0.39	0.29	**0.32**
SceneVLAD365+Omax	0.34	0.46	0.29	**0.36**
Δ				**0.04**
OpenSceneVLAD365	0.36	0.59	0.29	**0.41**
OpenSceneVLAD365+Omax	0.35	0.59	0.31	**0.42**
Δ				**0.01**
1365	0.20	0.25	0.20	**0.22**
1365+Omax	0.18	0.18	0.17	**0.18**
Δ				**-0.04**
SceneVLAD1365	0.31	0.38	0.31	**0.33**
SceneVLAD1365+Omax	0.37	0.47	0.31	**0.38**
Δ				**0.05**
OpenSceneVLAD1365	0.35	0.56	0.41	**0.44**
OpenSceneVLAD1365+Omax	0.36	0.59	0.35	**0.43**
Δ				**-0.01**

4.3.2.7 Does Openmax Improve Open Set Scene Classification?

Table 4.6 shows that using a scene classification network with an openmax layer and confidence thresholding results in all test data being collapsed into open class predictions, which performs worse than simply a softmax output decreasing the F1 score by -0.04 in both cases. However, when combined with SceneVLAD openmax does improve results by a mean of 0.04 (365) and 0.05 (1365), which provides supporting evidence for improved class representations. Including openmax with OpenSceneVLAD

changes results marginally with a mean F1 score increase of 0.01 (365) and -0.01 (1365), the results in Table 4.4 show that combining OpenSceneVLAD with openmax maintains or improve the F1 score on the open class in all cases, but reduces the F1 scores on the closed set classes.

4.3.2.8 Does OpenSceneVLAD Outperform SceneVLAD with an Openmax Layer?

The results in Table 4.7 confirm that OpenSceneVLAD outperforms SceneVLAD with an openmax layer by 0.05 (365) and 0.06 (1365). This demonstrates that even the best performing version of SceneVLAD with the openmax layer is outperformed by OpenSceneVLAD. Table 4.4 shows that versions of OpenSceneVLAD with or without the openmax layer perform the best on the datasets with only one exception (Oxford, SceneVLAD1365+Omax).

Table 4.7: Mean F1 cores for appearance invariant, open set scene classification using confidence thresholding. OpenSceneVLAD is compared to using SceneVLAD with an openmax layer (+Omax). Changes in mean F1 scores between best mean results (Δ) are positive and negative.

	Dataset Class Mean			
	Oxford	Nordland	Edinburgh	**Total Mean**
SceneVLAD365+Omax	0.34	0.46	0.29	**0.36**
OpenSceneVLAD365	0.36	0.59	0.29	**0.41**
Δ				**0.05**
SceneVLAD1365+Omax	0.37	0.47	0.31	**0.38**
OpenSceneVLAD1365	0.35	0.56	0.41	**0.44**
Δ				**0.06**

4.3.2.9 How Does OpenSceneVLAD Improve Open Set Scene Classification?

To investigate in detail how OpenSceneVLAD compares to the baseline networks for open set classification t-SNE plots are used. Firstly, samples of plots that correspond to results in Table 4.5 are presented. OpenSceneVLAD based on 365 and 1365 have similar results so for the sake of space only the results for OpenSceneVLAD1365 (Figure 4.9) are examined. This figure shows how the class representation develops from the baseline scene classification networks to SceneVLAD and then to OpenSceneVLAD. Scene classification network predictions collapse all test data into the open, or closed set classes with, or without an openmax layer. SceneVLAD begins to distinguish between open and closed set classes, it performs particularly well on the Nordland dataset. OpenSceneVLAD is able to further separate the closed class representations

from each other and the open set, although there remains a significant overlap, which is supported by the results in Table 4.4.

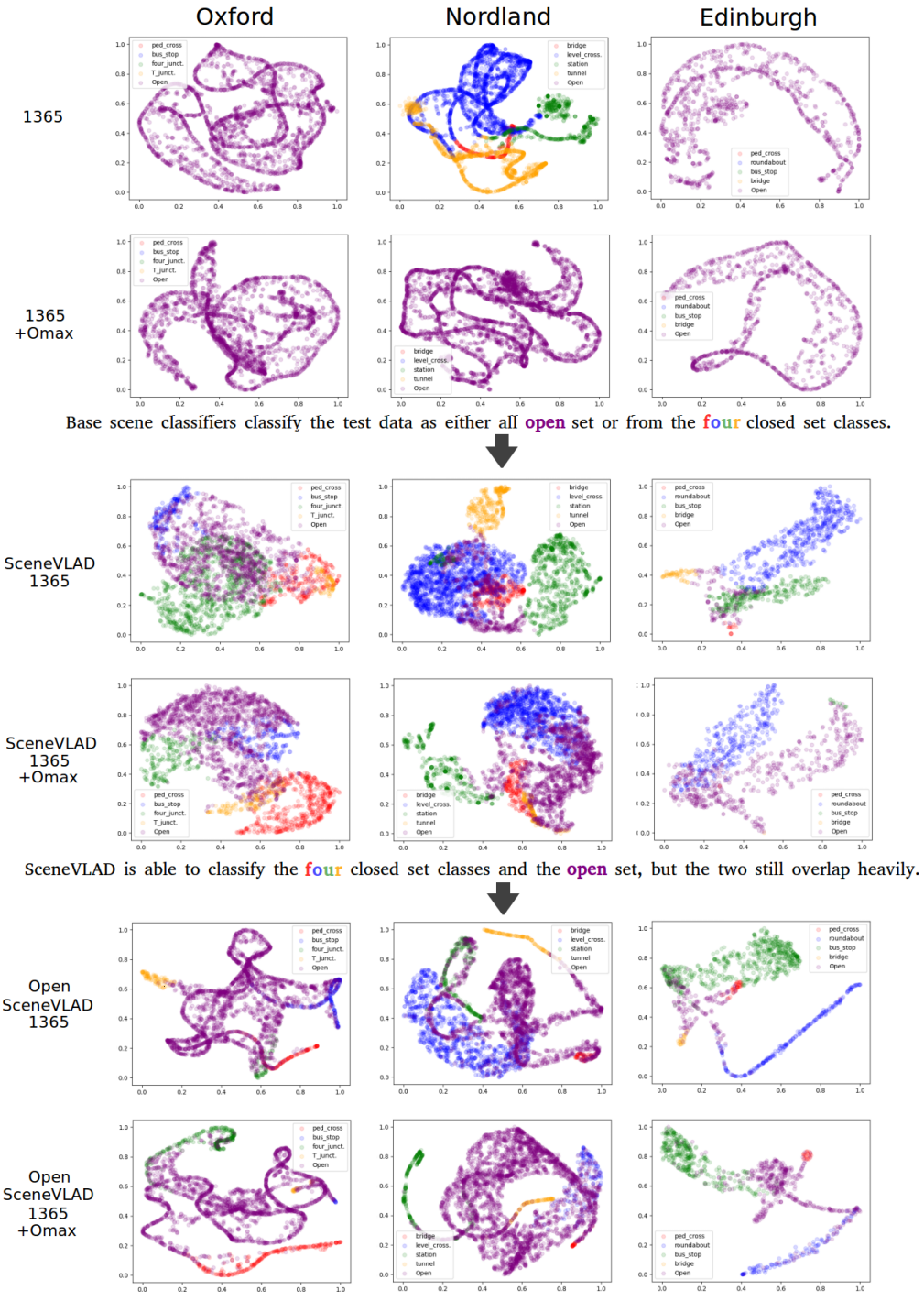


Figure 4.9: Samples of scene classification, SceneVLAD and OpenSceneVLAD t-SNE plots from the test results in Table 4.4. The progressive separation of the open set (purple) class from the closed set (red, blue, green, yellow) classes and the closed set classes from each other are shown from the 1365 scene classification network to SceneVLAD1365 and OpenSceneVLAD1365.

Additionally, in Figure 4.10 the advantage of OpenSceneVLAD over SceneVLAD is investigated. Two examples are selected from Figure 4.9 and a third from SceneVLAD365. The open class is removed from the plot to explicitly show how OpenSceneVLAD is able to better separate the closed from the open set classes and the individual closed set classes from each other.

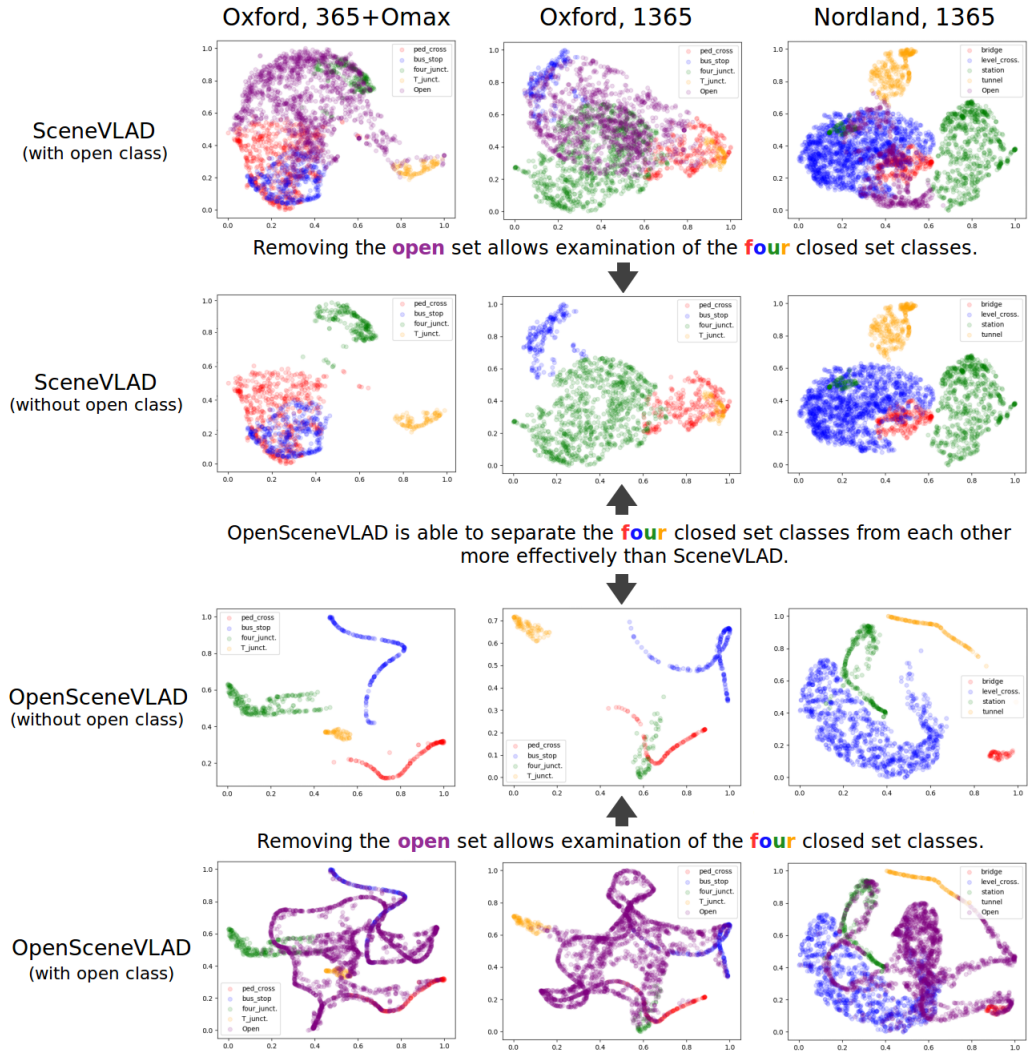


Figure 4.10: Samples of scene classification, SceneVLAD and OpenSceneVLAD t-SNE plots from the test results in Table 4.4. The overlap between the open (purple) and four (red, blue, green, yellow) closed classes is highlighted and the difference between SceneVLAD and OpenSceneVLAD's closed set class representation compared.

4.4 Discussion

This chapter proposes a series of contributions towards appearance invariant, open set scene classification. To the author's knowledge appearance invariance and open set classification have not previously been addressed specifically for scene classification.

This presented a number of challenges. Firstly, a number of scene classes had to be identified and extracted from three different visual place recognition datasets. A variety of outdoor scenes also had to be classified, despite significant changes in lighting, weather and seasonal conditions. To perform open set classification the original closed set of scene classes then had to be classified from a test dataset containing many open set images that frequently overlapped heavily with the closed set images.

To address these challenges visually invariant visual place descriptors were leveraged by fusing them with a scene classification network and forming a new network called SceneVLAD to achieve improved visual invariance. SceneVLAD was then extended to OpenSceneVLAD using intra-class splitting and an openmax layer to enable open set, visually invariant scene classification.

This approach was validated through a series of experiments. The first contribution, SceneVLAD, is shown to improve F1 classification scores for visually invariant scene classification, particularly for scene classes that are most vulnerable to visual changes, such as bus stops which are sometimes only defined by road markings which may be obscured in different lighting and weather conditions. The second set of experiments used remaining images from the dataset not included in the closed set of scene classes to test open set classification. SceneVLAD alone was demonstrated to improve the mean F1 classification score for open set classification and OpenSceneVLAD improves this even further with the further addition of an openmax layer only improving performance in some scenarios.

Training for the smallest dataset (Edinburgh) took roughly half an hour, but stretched to 2 hours on the larger Nordland dataset using an Nvidia RTX3070 and i5 CPU. A grid search was used for parameter tuning, as described in Section 4.2.2, for finding the number of filters used for dimension reduction, the number of final fully connected layers and their width. Additionally a small heuristic search was done to determine training parameters such as batch size, number of epochs and learning rate used for training the network, further parameter tuning is left for future work.

4.4.1 Limitations and Future Work

4.4.1.1 Additional Classes

Although a wide variety of scene classes are used for evaluation the total class number is strictly constrained by the available data. As this is the first time this problem has been addressed a wide variety of scene class data with large visual changes is very difficult to find for evaluation. The results also show that some scene classes are more vulnerable to visual changes than others, but fortunately the scene classes we use did prove to be vulnerable to visual change which made them useful for assessing the efficacy of our approach. In the future increasing the number of classes, particularly those vulnerable to visual change, would enable better evaluation. Increasing the variety of data sources would also enable the network's ability to generalise across classes from different datasets to be tested. A wider variety of data sources would also allow a wider variety of open set data to be used to evaluate open set classification. Intuitively SceneVLAD and OpenSceneVLAD should be able to be applied to more classes because NetVLAD and 365/1365 are based on networks originally taught to classify hundreds of different classes indicating a large remaining model capacity. Intra-class splitting was originally used for 10 classes, increasing the classes and therefore the open set data, may improve performance further, but this is reserved for future work.

4.4.1.2 Other Limitations

A second limitation of this work is that the NetVLAD descriptors are a very relevant, but slightly older visually invariant descriptor. More recent approaches, such as Patch-NetVLAD [13] may offer more visual invariance, but are not purely neural network approaches, so a different approach to fusing the visual place descriptors and scene classification may be required. More advanced approaches to visual invariance may also affect the network's use for open set classification.

Open set classification is still an emerging area of research, but the work in this chapter goes some way towards showing that it is possible in more challenging scenarios than previously recognised. Intra-class splitting works by using poorly or incorrectly classified training images as an open set, however this is problematic because some training data, although challenging, may be necessary for strong generalisation and therefore using it as open set data may be inappropriate. If all images are classified

with a high confidence or the training data is particularly high quality there may also not be many suitable images for use as the open class. However, intra-class splitting demonstrates that manipulating training data can be useful in creating data for learning open set classification so further work could examine using image augmentation based on classification performance or heatmaps to create an open set of data from the training images that did not rely on unaltered training data being appropriated for use as an open class.

Chapter 5

Descriptor Comparison Classification for Open Set Recognition and Outcome Classification

Image retrieval is established in previous chapters as a computer vision task that finds a match for a query image by comparing it against a database of reference images. Visual place recognition (VPR) is a variation of image retrieval that uses place images and specially designed VPR descriptors for this comparison. In this chapter two major new problems associated with image retrieval are identified and addressed in the context of VPR, to the author's knowledge, for the first time:

- Outcome classification to predict whether the top reference database matches include a true match for the query image.
- Open set recognition to predict whether a true match for the query image exists in the reference database.

Related problems have been addressed for image classification, as discussed in Chapter 2, but so far remain unexplored for image retrieval. Image classification returns a single label for an input image based on classes pre-defined at training time, whereas image retrieval only returns a relative similarity between previously unknown query and reference images. This fundamental difference in task renders

image classification approaches to uncertainty and open set recognition irrelevant for image retrieval. At first glance these problems appear almost intractable for image retrieval, but this chapter proposes that underlying characteristics of the relative similarities alone may be sufficient for training a neural network to address both problems.

Image retrieval compares a query image against a reference set of images, typically by measuring the distance between descriptors extracted from the images. The descriptors extracted by the neural networks considered here are in the form of a 1D vector of real numbers. Descriptors are used because they allow key information to be extracted from the images, they also take up less space and can be compared with each other using the Euclidean distance. These comparison values are also in the form of a 1D vector of real, positive numbers, one for each comparison between the query image and a reference image, this is henceforth known as a **descriptor comparison**. The descriptor comparison values can then be used to order the reference images according to how similar they are to the query images, these ordered reference images are henceforth known as the **nearest neighbours**. The pipeline for descriptor comparison is shown in Figure 5.1.

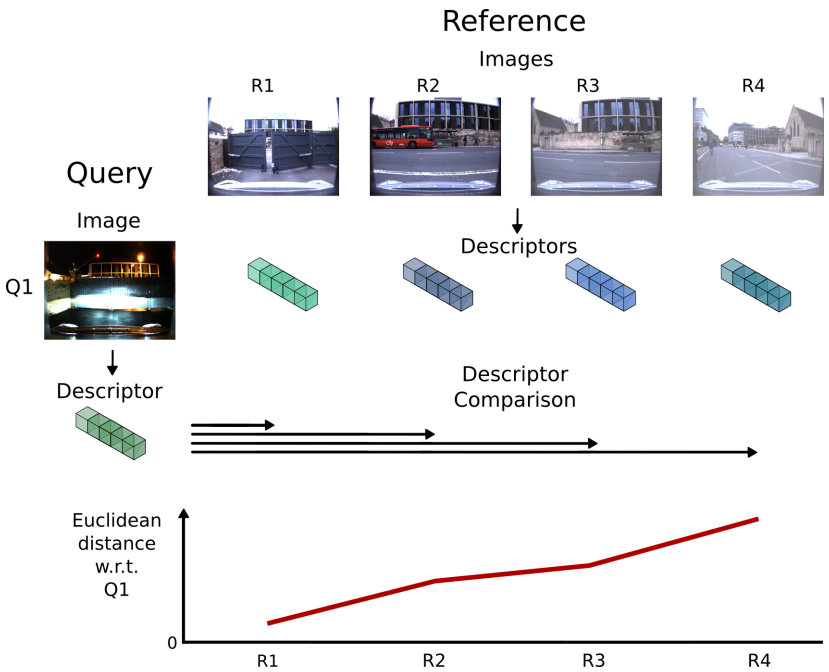


Figure 5.1: A diagram showing the descriptor comparison pipeline of a single query image against a set of reference images, note the size of the reference image set is only used for illustrative purposes. Initially the query image and reference image set is established, descriptors (each a 1D vector of numbers) are extracted and then used to compare the query image to each reference image. In this case the resulting comparison (a 1D vector of numbers) is plotted using a simple line graph.

Image retrieval is evaluated using a metric called **recall** ([10]), which differs from the traditional machine learning definition of recall. In this case a true positive result is defined if the top X nearest neighbours contain a **true match**, which *is a reference image within a dataset-specific number of meters from the query image*. For this work X is set to 10, a common threshold for determining true positives [10, 199]. Currently there is no way of knowing whether an image retrieval result is likely to be a true positive, which poses a significant problem for the real-world utility of image retrieval. Note, because image retrieval orders the reference set of images with respect to the query image it always returns a result and is unable to produce negative predictions. An example of this problem in the context of VPR is identifying reliable VPR results for updating localisation predictions during autonomous navigation.

This chapter defines, for the first time, the task of predicting **true positive**, or **false positive** VPR results as **outcome classification (OC)** and frames it as a binary classification problem (Figure 5.2). This chapter proposes descriptor comparison classification (DCC) to address this problem, which trains a neural network on this classification task using descriptor comparison data. The taught model can then classify unseen descriptor comparisons as resulting in true or false positive VPR results.

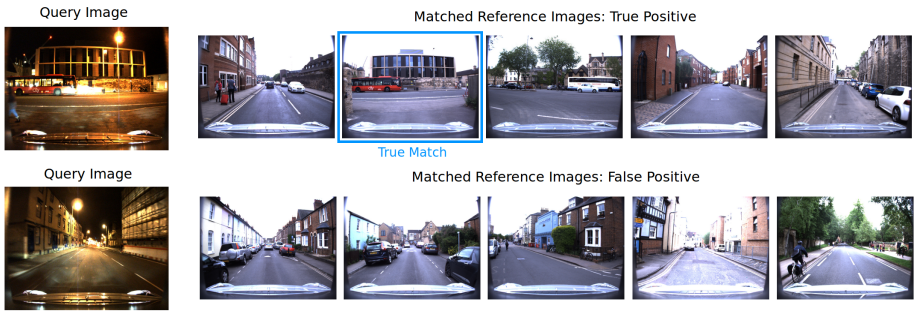


Figure 5.2: Illustrated example of outcome classification task using images from the Oxford RobotCar dataset. The top query image's 5 nearest reference image neighbours include a true match (true positive) and the bottom query image's do not (false positive). Please note, the usage of 5 closest matches is shown here for illustrative purposes only.

A fundamental assumption of image retrieval approaches is that a true match for the query image exists in the reference image database, however this assumption is often untrue for many practical scenarios. For example, during visual navigation a robot may leave a pre-mapped area and produce query images with no true match. Identifying these query images enables more intelligent navigation, for example a lost robot can signal for help or begin mapping the new area.

This chapter defines the task of predicting whether a query image has a potential true match (**closed set**) in the reference database, or not (**open set**) as an example of **open set recognition (OSR)** and a binary classification problem. This chapter proposes that DCC is suitable for addressing this problem as well because descriptor comparisons contain enough relevant information to train a neural network for this task. An illustrated example of open set recognition is shown in Figure 5.3

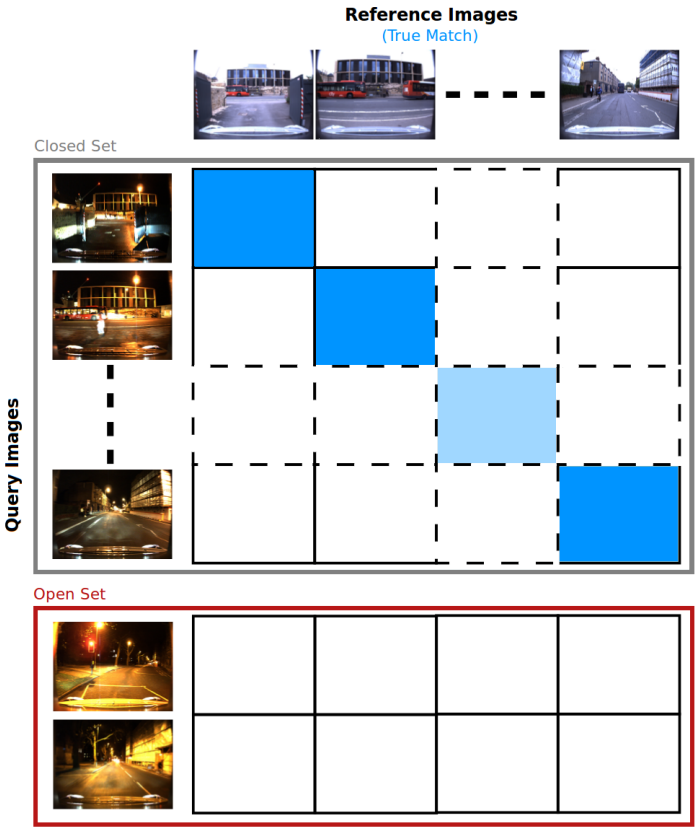


Figure 5.3: Illustrated example of open set recognition task using images from the Oxford RobotCar dataset. showing examples of closed set query images with a potential match in the reference image database and open set query images that do not.

To the author's knowledge this is the first time OC and OSR have been addressed for image retrieval. One related approach is out of distribution detection using Bayesian neural networks [199] which is used to estimate the uncertainty of comparisons between a query image and single reference images, but does not attempt open set recognition, or to predict an uncertainty related to true or false positive results. A small amount of work has been done on open set object recognition, but this approach only aims to find a known object in an otherwise unfamiliar image, which is in fact an example of image retrieval, not open set image retrieval [214, 215].

This chapter's final contribution is the application of contrastive supervised learning to both of these problems. Contrastive learning was used in Chapter 4 as the inspiration for the descriptor fusion, but in this chapter is used as originally proposed. This chapter will later show the descriptor comparison data for the two classes for both OC and OSR tasks overlaps. Contrastive learning trains a portion of the deep neural network classifier to separate the classes to improve learning. The remainder of the network is then trained on the original binary classification problem.

Experiments show that a baseline approach that only uses the distance between descriptors for classification cannot address OC and OSR reliably when compared to the proposed approaches. The baseline approach used in this chapter was inspired by confidence thresholding for binary classification used by other open set recognition work [21]. The proposed application of supervised contrastive learning is shown to improve mean F1 classification scores by up to 0.04 and 0.05 for OC and OSR respectively. In summary, this chapter's contributions are:

1. An introduction and formulation of outcome classification and open set recognition for visual place recognition (Section 5.1.2).

2. A descriptor comparison classification (DCC) approach applied to four deep learning architectures to address both problems (Section 5.1.3).

3. The use of supervised contrastive learning to separate overlapping classes in the embedded space for better classification performance (Section 5.1.4).

5.1 Methodology

This section describes a contribution towards outcome classification and open set recognition for visual place recognition. Firstly, the intuition behind outcome classification and open set recognition is explained and then the problems are formally defined. Secondly, an approach using descriptor comparisons to train four different deep neural network architectures for each classification task is described. Finally, the addition of supervised contrastive loss is proposed to separate overlapping classes and improve classification performance.

5.1.1 Investigation & Intuition

The introduction to this chapter establishes the pipeline required for descriptor comparisons: establish reference image set and query image, extract descriptors from the image, compare the query image descriptor with each reference image descriptor (Figure 5.13). Please note, the descriptor used for all the comparisons in this section is NetVLAD [10] with query images from 2015-03-24-13-47-33 (sunny) and reference images 2015-05-22-11-14-30 (overcast) from the Oxford RobotCar dataset [115]. In the following figures the reference and query images are considered as they were collected: in chronological order during the route traversal. A true match is defined here as a reference image from the same location as the query image.

Descriptor comparisons can be examined in a number of ways. One of these is a descriptor comparison plot, which is used in Chapter 3 (Figures 3.11, 3.12 & 3.13), another example is shown in Figure 5.4. These plots show the descriptor comparison values for each query image when compared to each reference image. There is a one-to-one correspondence between query and reference images so a distinct diagonal line of small comparison values (a high similarity) indicates true matches between corresponding query and reference images. VPR aims to represent each place uniquely so line's width and distinction gives an indication of the performance of a descriptor for the task. This plot can also be visualised in 3D (Figure 5.5).

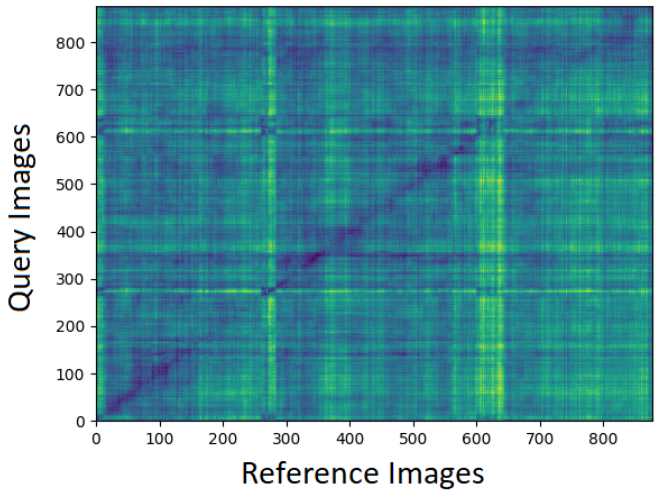


Figure 5.4: 2D descriptor comparison plot of query and reference images where smaller descriptor comparison values (higher similarity) are dark blue and higher values (lower similarity) are in green. This figure is for illustrative purposes only.

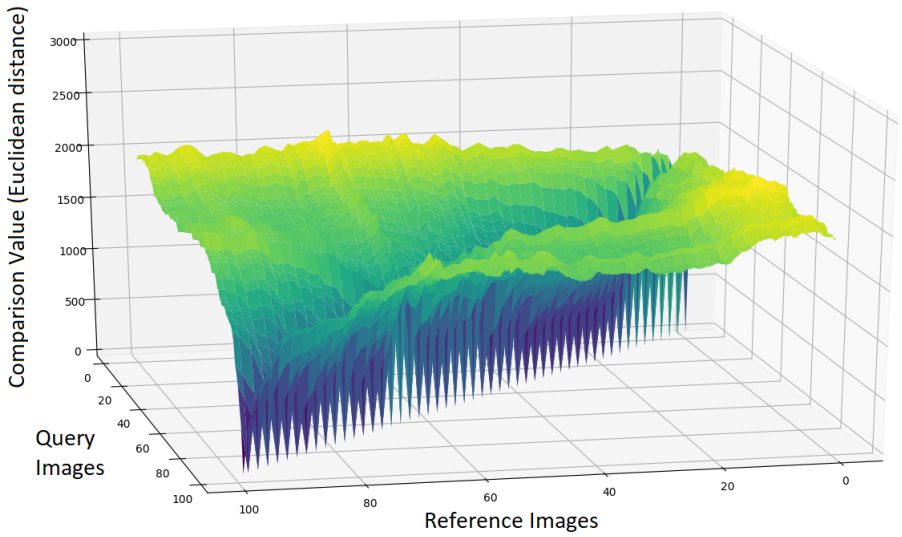


Figure 5.5: 3D descriptor comparison plot of a subset of query and reference images, where smaller descriptor comparison values (higher similarity) are dark blue and higher values (lower similarity) are in green. This figure is for illustrative purposes only.

5.1.1.1 Outcome Classification

Outcome classification considers the problem of overconfident descriptor matches. An example of these can be seen in Figure 5.6, where single descriptor comparisons that lead to false positives, when the true matching diagonal is poorly defined, are a similar value to single descriptor comparisons that lead to true positives. This means that a low comparison value is no guarantee of a true match which makes them difficult to differentiate from false positives by placing a threshold on single comparison values alone.

However, all of the descriptor comparison values for a query image may contain more useful information. Specific plot descriptor comparisons were then examined to confirm their characteristics varied for false and true positive results by looking at specific query images. The following samples represent the general observations made between the classes. A sample of a true positive descriptor comparison in Figure 5.7 shows a distinct and concentrated fall in comparison values where all of the nearest neighbours are collected around the true match. In contrast the false positive sample in Figure 5.7 comes from a descriptor comparison with multiple, small local minima where the nearest neighbours are distributed. This provides evidence that the descriptor comparisons and the distribution of nearest neighbours contain

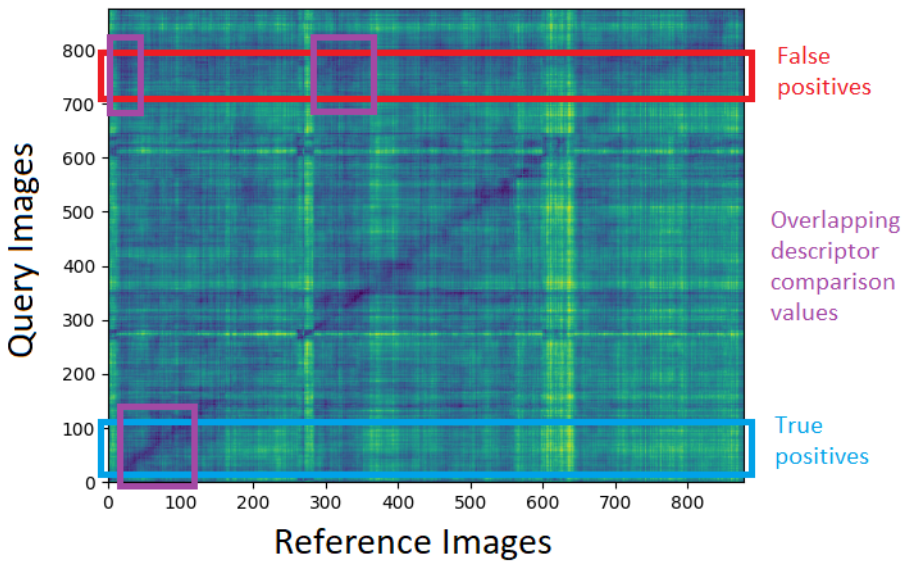


Figure 5.6: Annotated version of Figure 5.4 showing the descriptor comparison plot of query and reference images, where smaller descriptor comparison values (higher similarity) are dark blue and higher values (lower similarity) are in green. Additionally, descriptor comparisons leading to true positives (include distinct dark blue values as part of the larger diagonal across the image) and descriptor comparisons leading to false positives (include no distinct dark blue values as part of the larger diagonal across the image) are highlighted in blue and red respectively. The overlapping descriptor comparison values from each are highlighted in purple.

information which can be used to learn to classify true and false positives. It's also worth noting that the the true positive result is achieved despite some occlusion from a passing bus, however the more significant occlusion from a bus may lead to the false positive.

5.1.1.2 Open Set Recognition

Open set recognition considers the problem of being able to identify query images with no true match in the reference image set. Figures 5.4 & 5.4, the literature [10, 13, 14, 70, 71] and intuition determine that, given a reference set of unique images and a query image with a potential true match. The aim of VPR is to generate a descriptor comparison that represents the query image as closely as possible to a single reference image (the true match) and to represent the remaining reference images as far away as possible. If the true match was to be removed from the reference image set, making the query image open set, ideally all remaining reference images should still be represented far away from it, enabling a threshold to be used to easily identify open set query images.

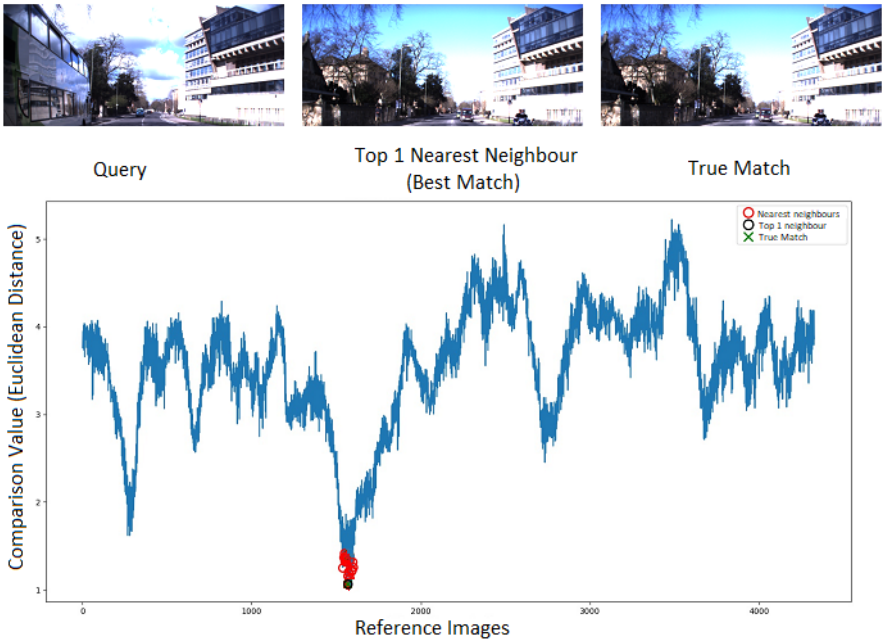


Figure 5.7: Descriptor comparison plot between a query image and reference image set from the Oxford dataset leading to a true positive match. The query image, the top match and the true match from the reference set are displayed and plotted on the descriptor comparison along with the next 24 nearest neighbours.

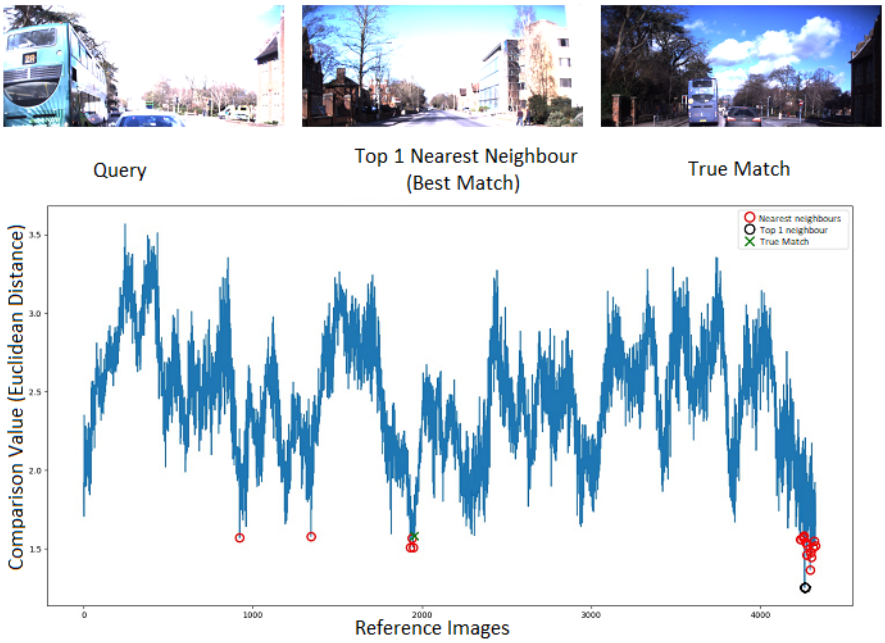


Figure 5.8: Descriptor comparison plot between a query image and reference image set from the Oxford dataset leading to a false positive match. The query image, the top match and the true match from the reference set are displayed and plotted on the descriptor comparison along with the next 24 nearest neighbours.

An investigation to explore this was undertaken (Figure 5.9) by removing all frames beyond the first 1500 frames from the route used for reference image set. This made all the query images beyond this cut-off point open set. Each query image was compared with the reduced reference set and the minimum single comparison value was plotted against the corresponding localisation error. Because there was no true match for the query images beyond frame 1500, as expected, the localisation error promptly increased beyond the cut-off. The minimum single comparison value increased on average, but there was still a large overlap between the values. In other words there is no reliable way to determine whether a match for a query image is likely to have a potential true match in the reference database using single descriptor comparison values, they need to be considered as a whole.

However, as previously, all of the descriptor comparison values for a query image may contain more useful information. Specific plot descriptor comparisons were then examined to confirm their characteristics vary for closed and open set results by looking at specific query images and the distribution of nearest neighbours. The following samples represent the general observations made between the classes. A sample of a closed set descriptor comparison is shown in Figure 5.10 where a true positive result with the nearest neighbours is clustered around the true match, closed set results can also include false positives (Figure 5.8). The open set descriptor comparison sample in Figure 5.11 shows several local minima with nearest neighbours distributed across them, but with fewer local minima than the false positive, which intuitively makes sense if the query image is from a very different place. This provides evidence that the descriptor comparisons and the distribution of nearest neighbours contain information which can be used to learn to classify query images as open and closed set.

5.1.1.3 Descriptor Comparison Pre-processing

Extracting information from the descriptor comparison for use in a classifier is difficult for two main reasons. Firstly, the length of the descriptor comparison is completely dependent on the number of reference images which is highly variable and the network therefore needs to be capable of accepting a variable size. This is possible, but has implications for generalisation to different input sizes. For example, some reference image sets may be hundreds of images, while others could be thousands. Secondly, there is a lot of variation in the descriptor comparison values which is essentially noise

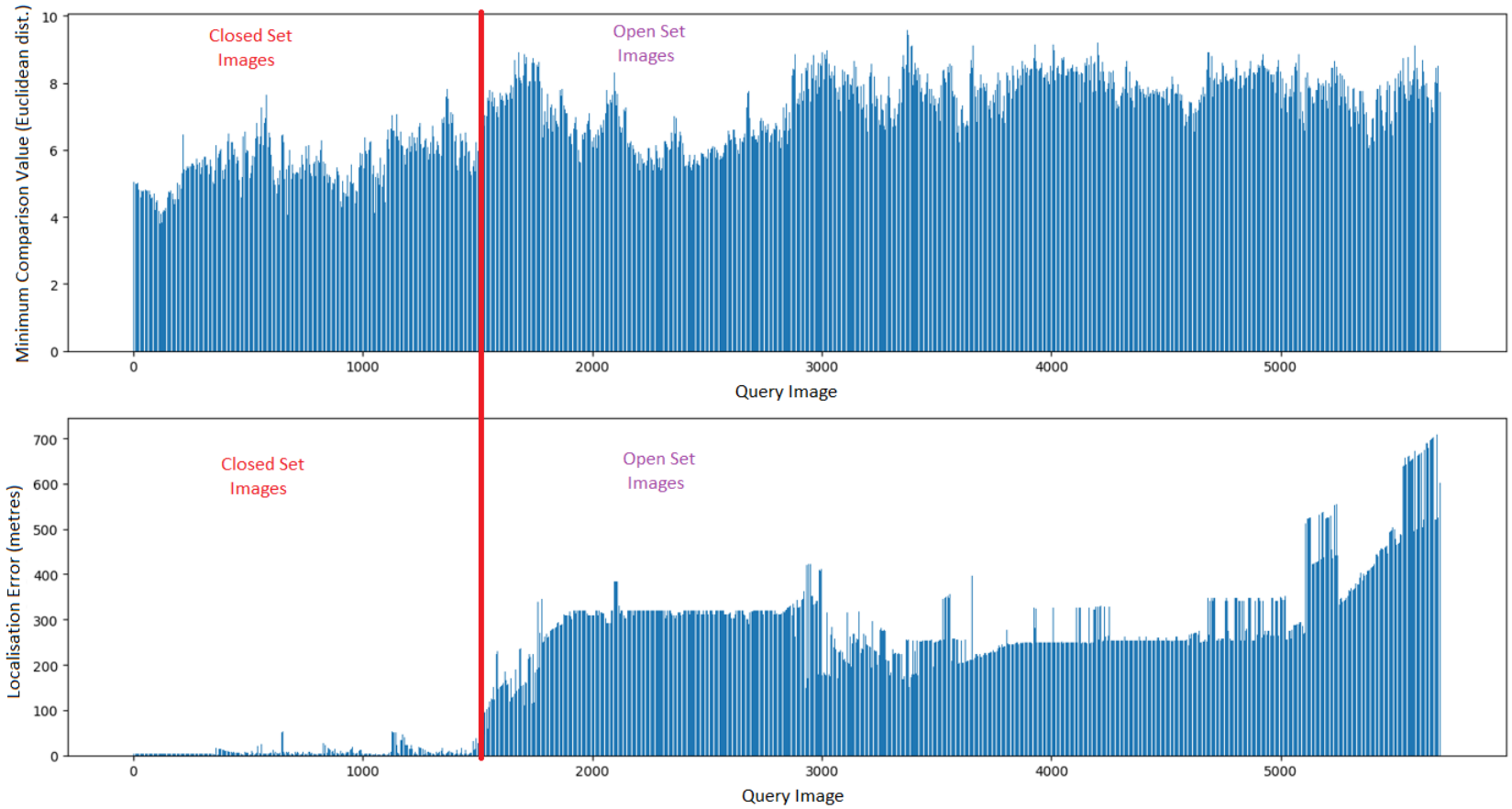


Figure 5.9: A plot associating localisation error and minimum descriptor comparison values for a series of query images, the first 1500 of which have a true match that exists in the reference image set (closed set) and the frames beyond which do not (open set). The 1500 frame cutoff for this division is denoted by a purple line. The increase in localisation error is shown to not necessarily correspond to a corresponding and proportional increase in descriptor comparison values.

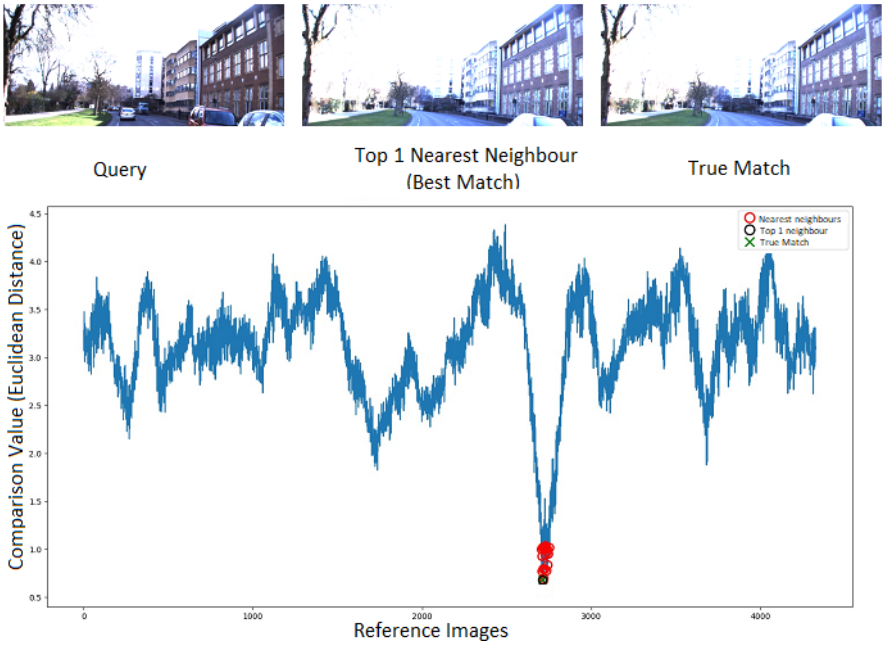


Figure 5.10: Descriptor comparison plot between a query image and reference image set from the Oxford dataset corresponding to a closed set query image. The query image, the top match and the true match from the reference set are displayed and plotted on the descriptor comparison along with the next 24 nearest neighbours.

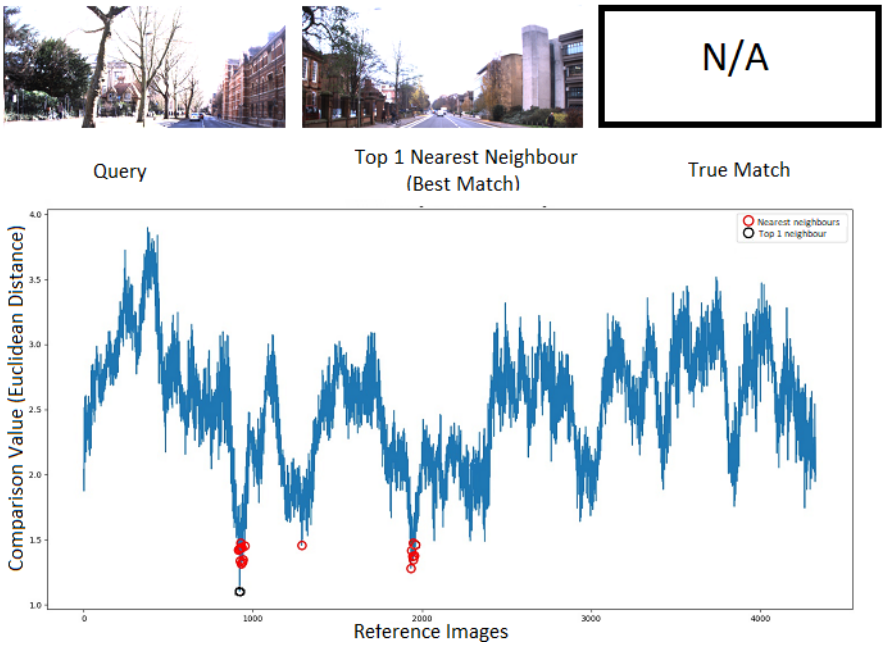


Figure 5.11: Descriptor comparison plot between a query image and reference image set from the Oxford dataset corresponding to an open set query image. The query image, the top match from the reference set are displayed and plotted on the descriptor comparison along with the next 24 nearest neighbours.

and is not useful for this task and can even be misleading, it is therefore important to try and extract the key parts of the descriptor comparison for input to the classifier.

The final approach decided on for pre-processing was to take the descriptor comparison values from the 25 nearest neighbours and then exploit the sequential nature of the route data to take the comparison values from 3 geographically nearby images. This kept the input size to a vector of length 100, the number of nearby images to consider was varied heuristically to establish this value, any less than 3 were found to negatively impact performance with more providing no significant increase. The top 25 nearest neighbours were chosen as this is a common threshold for considering relevant image retrieval results [10]. The idea behind this approach was to encode some geographical data into the processed descriptor comparison while also capturing the spread and size of the local minima which from observations seemed to be some of the major variables associated with each class. However, there may be some overlap between classes, as seen in some of the similarities between Figures 5.7, 5.8, 5.10 & 5.11. Contrastive learning is proposed to help separate the data, as proposed in Section 5.1.4.

5.1.1.4 Summary

The main difference between image retrieval and classification is that retrieval does not define traditional classes for training and testing. Instead image retrieval outputs a vector of descriptor comparison values of unknown length, rather than a traditional classifier's output of known class probabilities. Traditional image classification approaches for calculating prediction uncertainty and open set recognition are therefore not applicable. As the only given for image retrieval are the relative descriptor comparison values they need to be used to solve these two tasks. The intuition behind this chapter's approach is that, despite descriptor comparison noise, valuable information can still be extracted and used to address outcome classification and open set recognition tasks.

5.1.2 Problem Definition

5.1.2.1 Pre-Processing

The result of an image retrieval query, Q, compared to N reference images, R, is a descriptor comparison, DC. The DC is the Euclidean distance between the query image descriptor, Q_{desc}, and the reference descriptors, $[R_{desc_1}, ...R_{desc_N}]$ as shown in Equation 5.1. Where $MinMaxScaler()$ is an sklearn function that scales input between 0 and 1 [1]. Scaling is necessary because the absolute values of the descriptor comparisons can be misleading. For example, comparing a query image taken in summer to a winter reference image set may lead to larger absolute values than comparing to a spring reference image set but this does not reflect the relative changes between the values which have been established as being relevant for learning each task.

$$DC = MinMaxScaler([\|Q_{desc} - R_{desc_1}\|, \dots \|Q_{desc} - R_{desc_N}\|]) \tag{5.1}$$

The smaller a value in DC is the better the match between the query image and that reference image. Once the descriptor comparison has been calculated it is used to sort the reference image indexes from best to worst match (Equation 5.2), this is a permutation of the original indexes $[1..N]$, also referred to as the nearest neighbours of the query image in this chapter.

$$matches = argsort(DC) \tag{5.2}$$

Each reference image match has a corresponding GPS location. For evaluation the localisation errors, $errors$ are calculated by measuring the geographical distance, $GeoDist()$, between the GPS locations of the query image, Q and the reference image matches (Equation 5.3).

$$errors = [GeoDist(Q, R_{matches[1]}), \dots GeoDist(Q, R_{matches[N]}] \tag{5.3}$$

5.1.2.2 Outcome Classification

Recall, in the case of image retrieval, has a different definition from its traditional usage in machine learning. For evaluation recall [10, 199] is defined as the index, i,

[1] https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html

of the first element of $errors$ that is a true match for the query image, i.e. less than a geographic, dataset-specific localisation error threshold, $Thresh_{error}$. This is fully described in Equation 5.4 and a visual example is given in Figure 5.12. Note that conventional image retrieval, and therefore VPR, assumes a true match exists for each query image and therefore a true or false negative outcome is impossible.

$$recall = min(i \,|\, errors[i] < Thresh_{error}) \tag{5.4}$$

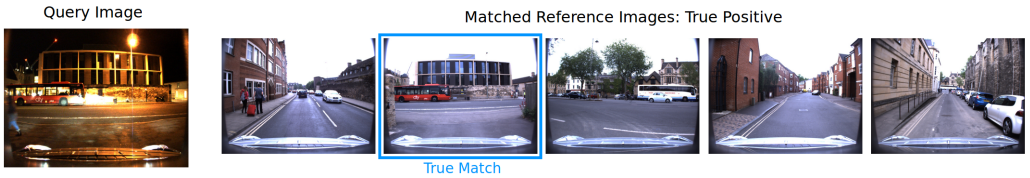


Figure 5.12: Illustrated representation of $recall = 2$, as described by Equation 5.4

Outcome classification (OC) aims to predict whether a DC is a true or false positive, i.e. whether its $recall$ value is above or below a threshold value, $Thresh_{recall}$ which is set to 10 for this work (Equation 5.5). This is done before evaluation, when the localisation error is calculated and $recall$ is explicitly measured.

$$OC = \begin{cases} True\ Positive, & \text{if } recall \leqq Thresh_{recall} \\ False\ Positive, & \text{if } recall > Thresh_{recall} \end{cases} \tag{5.5}$$

5.1.2.3 Open Set Recognition

Open set recognition (OSR) is defined for classification as the recognition of 'unknown classes submitted to an algorithm during testing' [181]. As image retrieval defines known classes at test time in the reference image database this is a particularly challenging problem and not compatible with current open set image classification approaches (Chapter 2). OSR is therefore defined as the binary classification of DC as including a true match for the query image, or not. For image retrieval it is proposed that an open set query image is defined as one without a true match in the reference image database, which would mean i is undefined for Equation 5.4.

$$OSR = \begin{cases} Open\ Set, & \text{if } recall \text{ is undefined} \\ Closed\ Set, & \text{otherwise} \end{cases} \tag{5.6}$$

5.1.3 Descriptor Comparison Classification

5.1.3.1 Basic Hypothesis

The main hypothesis explored in this chapter is that a deep neural network (DNN) can be used to identify underlying characteristics of descriptor comparisons which can then be used to classify whether they are likely to include a true match (OC) in the top results, or if a true match is likely to exist in the reference image database at all (OSR).

5.1.3.2 Data Pre-processing

The comparison of query and reference descriptors, DC as defined above, is a one dimensional vector that must be pre-processed before it can be used as data for learning OC and OSR because of the noise from dissimilar matches described in Section 5.1.1. To create this data the required visual place recognition dataset and descriptor required and used in this chapter are described in Section 5.2.1.1.

The first 25 elements of $matches$, as defined previously, are selected. For each element the three geographically nearest images in the reference set are selected and the DC values associated with all four images are sampled for a total of 100 values (Figure 5.13). This is done to provide information about the similarity of the matched place and nearby places, with respect to the query image for richer training data, not including it was found to significantly decrease classification accuracy. This sampling removes noise and fixes the input data size, which allows for generalisation to reference image databases as small as 25 images and is compatible with the input shape of a wider variety of deep learning approaches. A grid search was used to set the variables for number of best matches and adjacent descriptor comparisons.

5.1.3.3 Baseline Calibrated Threshold

This is the first time OC and OSR problems have been addressed so there are no state-of-the-art approaches for comparison. As discussed in Section 5.1.1 larger comparison values between query and reference descriptors should intuitively suggest for OC that a query image is a false positive ($y = 1$), rather than a true positive ($y = 0$) and for OSR that a query image is open set ($y = 1$), rather than closed set ($y = 0$). A threshold approach inspired by the confidence thresholding baseline used for binary

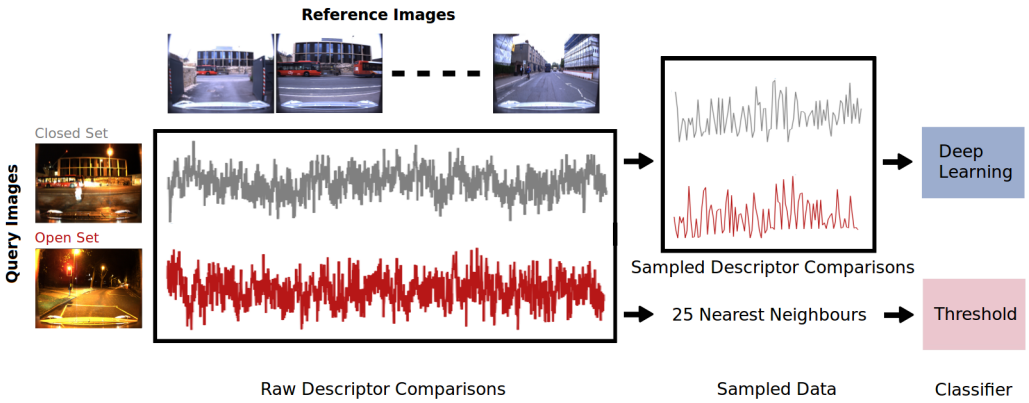


Figure 5.13: Illustrated example of the proposed classification pipeline for query and reference image descriptor comparisons, including dataset pre-processing. This example uses OSR classes.

classification by Bendale et al. [21] was therefore used to discriminate between larger and smaller descriptor comparison DC values and enable classification for each task accordingly. Training data for each task was used to calibrate the threshold $Thresh_{class}$ by using the DC values of the first 25 $matches$ elements of each training sample to optimise $Thresh_{class}$ for binary classification on that task:

$$\text{predicted labels, } \hat{y} = \begin{cases} 1, & \text{if } max([DC_{matches[1]}, \dots DC_{matches[25]}]) > Thresh_{class} \\ 0, & \text{otherwise} \end{cases} \tag{5.7}$$

The threshold is optimised (Equation 5.8) to maximise the F1 score of the predicted, $\hat{y}$ and true training labels y.

$$\begin{aligned} \text{maximise} \quad & \text{F1 score}(y, \hat{y}) \\ \text{where} \quad & y = \text{true labels} \\ & \hat{y} = \text{predicted labels} \end{aligned} \tag{5.8}$$

Only the the first 25 $matches$ are used as input (Equation 5.7) because the sampling approach from Section 5.1.3.2 introduces noise that is potentially useful for training a DNN, but disruptive for classification with this calibrated threshold. The calibrated threshold can then be used to make predictions on test data.

5.1.3.4 Deep Neural Networks

The data pre-processed according to Section 5.1.3.2 is a one dimensional vector so four architectures inspired by state-of-the-art signal classification approaches [216] were taught for both binary classification tasks using sigmoid loss and the pre-processed descriptor comparisons.

1. Fully connected models are naturally well suited to one dimensional input and have been used for classifying ECG signals [217] and hand gestures [218].

2. Convolutional models have been used for classifying ECG [219], high power engine [220] and high power circuitry [221] signals.

3. ResNET models take inspiration from computer vision research to introduce skip layers and have been used to classify electrodermal activity [222] and radio signals [223].

4. Transformers have been used to great effect for natural language processing and have recently been applied to time series classification [224]. Transformers have not been commonly explored for signal classification, however their use with time series data has shown that they are relevant for usage with sequential data.

Each architecture is used to build an encoder that is appended to a small, fully connected classifier network (Figure 5.14). The distinction between encoder and classifier architectures is made to enable an ablation study that compares training the models as a whole against the use of supervised contrastive loss (Section 5.1.4) which trains the encoder and classifier separately. The specific architectures were inspired by the referenced works and adapted for the task and ease of comparison. For example, all the final fully connected layers of all encoders (Figure 5.15) have a dimension of 128 and both ResNet and Convolutional architectures use layers with 32 and 16 filters.

5.1.4 Supervised Contrastive Loss

5.1.4.1 Basic Hypothesis

This section aims to use supervised contrastive loss to train an encoder network to use similarity learning to maximise the separation between overlapping classes of pre-processed descriptor comparison data, which can then be used to improve training

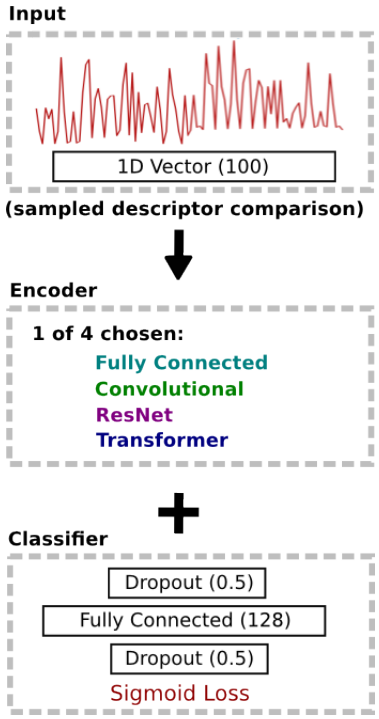


Figure 5.14: Diagram showing overall network architecture, as described in Section 5.1.3.4. For supervised contrastive loss each encoder was alternately separately pre-trained and then frozen before the addition of the classifier, as described in Section 5.1.4.2. Each layer shown includes its output shape.

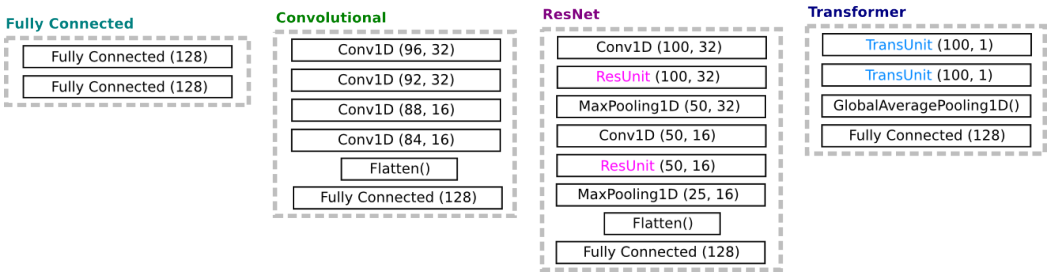


Figure 5.15: This figure shows the encoder architectures, as described in Section 5.1.3.4 which are then used in the larger DCC pipeline (Figure 5.14). Each layer is shown along with its output shape. The individual architectures used for the ResUnit and TransUnit layers are shown in Figures 5.16 & 5.17

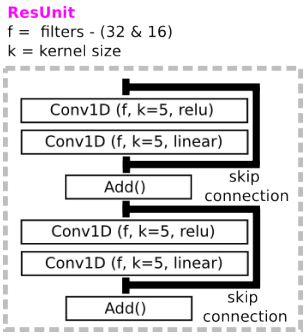


Figure 5.16: ResUnit architecture used in the ResNET encoder (Figure 5.15), the first layer in the encoder uses 32 filters and the second layer 16 filters.

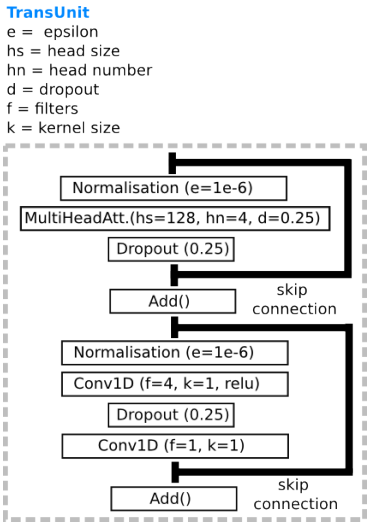


Figure 5.17: TransUnit architecture, used in the Transformer encoder (Figure 5.15).

for classification. The intuition here is that descriptor comparison data for true and false positives used in the outcome classification task and the open and closed descriptor comparison data for open set recognition can overlap, so separating them may improve classification performance.

5.1.4.2 Training

A specific example of overlapping class data can easily be imagined for outcome classification which defines two classes: true positives ($recall \leqq 10$) and false positives ($recall > 10$). Intuitively there will be significant overlap between descriptor comparison data with a recall of 9 and 11. This chapter therefore aims to use supervised contrastive loss [209] (Equation 5.9) to pull the network encoder's embedded output from the same classes together and away from that of different classes.

The encoder section of each architecture is therefore pre-trained using contrastive loss on a pretext task to separate the classes and frozen, before the final classifier network is appended and trained with the improved embedded representation. The advantage of using supervised contrastive learning is that the labels remove the need to mine the hardest combinations of embeddings for learning, which can be difficult and time-consuming when unsupervised.

The generalised form of supervised contrastive loss in Equation 5.9 was introduced by Khosla et al. [209] and is a variation of contrastive loss for similarity learning presented by Chen et al. [225]. A minibatch of N randomly sampled sample/label pairs

$\{\boldsymbol{x}_k, \boldsymbol{y}_k\}_{k=1\ldots N}$ is used and the contrastive prediction task is applied to each sample where all other samples in the mini batch of the same class become positives and the remaining samples are negatives, resulting in N data points. Let $i \in I \equiv \{1 \ldots N\}$ be the index of an arbitrary sample. The set of indices of all positives in the batch distinct from i is $P(i) \equiv \{p \in A(i) : \tilde{\boldsymbol{y}}_p = \tilde{\boldsymbol{y}}_i\}$, and $|P(i)|$ is its cardinality. Additionally, $\boldsymbol{z}_\ell = Clas\left(Enc(\tilde{\boldsymbol{x}}_\ell)\right) \in \mathcal{R}^{D_P}$, the $\cdot$ symbol denotes the inner (dot) product which is used to measure the similarity between embedded class representations, $\tau \in \mathcal{R}^+$ is a scalar temperature parameter, and $A(i) \equiv I \backslash \{i\}$.

$$\mathcal{L}_{\text{cont}}^{\text{sup}} = \sum_{i \in I} \mathcal{L}_{\text{cont},i}^{\text{sup}} = \sum_{i \in I} \frac{-1}{|P(i)|} \sum_{p \in P(i)} \log \frac{\exp\left(\boldsymbol{z}_i \cdot \boldsymbol{z}_p / \tau\right)}{\sum_{a \in A(i)} \exp\left(\boldsymbol{z}_i \cdot \boldsymbol{z}_a / \tau\right)} \tag{5.9}$$

The intuition behind supervised contrastive loss is that normalised embeddings from the same class are drawn together and pushed away from embeddings from different classes. Supervised contrastive loss considers many positives per anchor in addition to many negatives, as opposed to triplet loss which only uses a single positive and negative. These positives are drawn from samples of the same class as the anchor, while the negatives come from any other class.

5.2 Evaluation

This section presents an evaluation of the proposed contributions. Firstly, the datasets and VPR descriptors necessary for this evaluation are introduced. Secondly, the proposed descriptor comparison classification approach is used to train deep learning architectures for outcome classification in comparison to a calibrated threshold. The addition of supervised contrastive loss is then tested in an ablation study. Finally this evaluation is repeated for open set recognition.

5.2.1 Experimental Setup

5.2.1.1 Datasets

This is the first time outcome classification and open set recognition have been addressed, for image retrieval generally and specifically VPR, so datasets for these tasks had to be created.

To begin with two traversals in different appearance conditions of a single route

from four VPR datasets were used for this with associated geolocation data for each image. The first dataset is the Oxford RobotCar dataset [115] and the two traversals used are: 2015-05-22-11-14-30 (overcast), 2014-12-16-18-44-24 (night). The second dataset is Nordland [133] and the two traversals used are winter and summer. The third dataset is St. Lucia [226] and the two traversals used are 100909_0845 (morning) and 180809_1545 (afternoon). The fourth dataset is the UAH-DriveSet [200] and the two traversals used are UAH 4 (sunset) and UAH 6 (haze). As the datasets were originally raw video recordings each dataset was sampled using a minimum distance between consecutive frames of 0.1 (Oxford), 80 (Nordland) and 5 (UAH) meters to prevent oversampling of single locations. The St. Lucia dataset required some additional synchronising between routes to ensure that the same place images shared a single GPS point, it therefore had to be sampled with a mean distance between frames of 10 meters. For each dataset one traversal was randomly designated as a reference and the other as a query traversal, this was kept constant for all experiments. Samples of each dataset are shown in Figure 5.18.

Figure 5.18: Samples of the four VPR datasets used for experimentation and the associated appearance invariance. From top to bottom, left to right: Oxford RobotCar (night, overcast), UAH DrivSet (sunset, haze), Nordland (winter, summer), St. Lucia (afternoon, morning).

For open set recognition the reference traversal was then split into two halves and

only one half was included in the reference image database. For each split query images with a possible true match were designated as *closed set* and the rest as *open set*. The descriptor comparison data between query and reference images was generated using one of the four VPR descriptors described in Section 5.2.1.2.

For outcome classification descriptor comparisons between the reference and closed set query descriptors with a recall of up to and including 10 were labelled as *true positives*. The remainder of the comparisons between the reference and closed set query descriptors were labelled as *false positives*. This data was descriptor dependent because higher VPR performance meant more true positives existed. The average OC data produced for each of the four descriptors is reported in Table 5.1, along with a detailed summary of the datasets and a visualisation of a dataset split in Figure 5.19.

Table 5.1: A detailed description of the VPR datasets used to generate the OC and OSR datasets, as described in Section 5.2.1.1. True and false positive comparisons are derived from the closed set comparisons and therefore their total data samples equal the number of closed set samples. Note OC data depends on the accuracy of the descriptors used for comparison, the mean result using the four descriptors described in Section 5.2.1.2 is shown here

					OC Data		OSR Data	
Dataset	**Traversals**	**Frames**	**Type**	**Length (km)**	**False Positive**	**True Positive**	**Closed Set**	**Open Set**
Oxford	Overcast (*ref.*) Night (*query*)	3235 3280	Urban	9	2416	1281	3697	2863
Nordland	Summer (*ref.*) Winter (*query*)	4619 4617	Rural	763	1859	2760	4619	4615
St. Lucia	Afternoon (*ref.*) Morning (*query*)	1619 1628	Suburban	18	1530	1239	2768	488
UAH	Haze (*ref.*) Sunset (*query*)	2066 2077	Rural	16	919	1128	2047	2107
				Total:	6724	6408	13131	10073

5.2.1.2 VPR Descriptors

To experiment whether the proposed approach generalised to different descriptors the four different state-of-the-art VPR descriptors were used to generate descriptor comparisons. These descriptors are listed below, from oldest to newest.

AMOS and HybridNet [14]. AMOS-Net is a modified version of Caffe-Net with all parameters trained for VPR, whereas Hybrid-Net's top 5 convolutional layers were initialized from Caffe-Net. 'Conv5' layer features from both networks are extracted and encoded using Spatial Pyramidal Pooling to an output size of 2543.

NetVLAD [10]. NetVLAD appends a VLAD layer to a partially frozen VGG16

network pretrained on the ImageNet dataset [12] and retrained for VPR using triplet learning. NetVLAD features are reduced using PCA to an output shape of 4096.

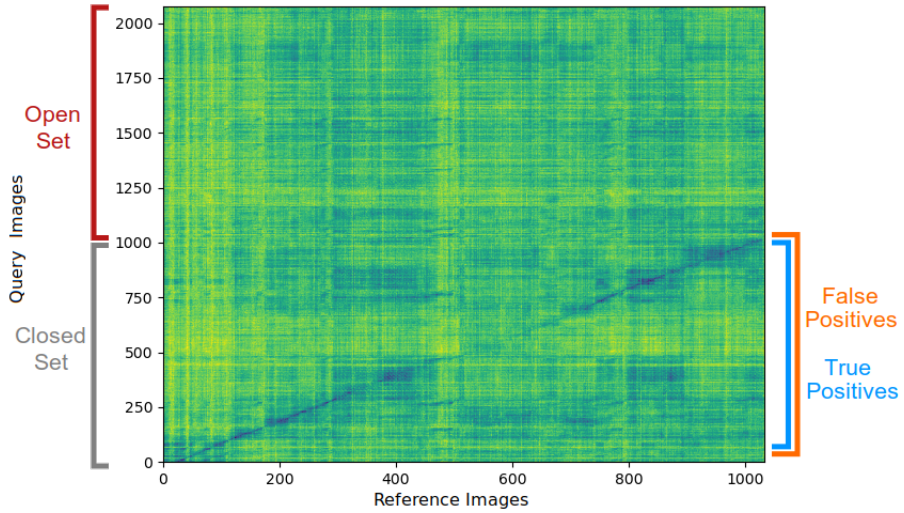


Figure 5.19: A matrix showing comparisons between the UAH Driveset [200] query and reference images using NetVLAD descriptors [10]. For open set recognition comparisons of open set images with no potential true match in the reference image set are shown alongside comparisons of closed set images with a possible true match. For outcome classification closed set comparisons can then be divided into true positives (top 10 nearest neighbours include a true match for the query image and false positives which (no true match in the top 10 nearest neighbours). The proposed approach uses these descriptor comparisons to train a neural network for these two binary classification tasks. Descriptor similarity ranges from blue (high) to yellow (low).

Patch-NetVLAD [13]. Patch-NetVLAD, introduced in Chapter 2 uses multi-scale fusion of patch-level features from NetVLAD residuals to rearrange the top 50 descriptor comparisons done using NetVLAD, re-trained specifically for urban (Oxford and St. Lucia) or rural (Nordland and UAH) environments. For example, the first match, according to the Patch-NetVLAD comparison, is given the value of the first match from the original NetVLAD descriptor comparison, this is done to maintain a single scale across the entire descriptor comparison. For experimentation performance configuration of Patch-NetVLAD was impractical so the speed configuration was used, which used more memory than than the NetVLAD features by a factor of 100. The dimensions (*num_pcs*) of the speed configuration Patch-NetVLAD patch descriptors were increased to 512 to improve performance, resulting in a final descriptor shape of (936x512).

Descriptor performance for localisation varies across each dataset split and the average performance is shown in Table 5.2. Recall at k (R@k) measures the percentage

of queries that have at least one positive among their closest k descriptor comparison neighbours, as described in Section 5.1.2. Although PatchNetVLAD [13] claims an improvement over vanilla NetVLAD these results suggest that hyperparameter selection significantly affects its performance.

Using descriptors trained on each route specifically, such as the TinyVPR descriptors proposed in Chapter 3 is an interesting possibility as they would be specialised for each route and might be more sensitive to false positives or open set images, but this is left for future work. Pre-trained descriptors were used in this case because of their wider availability.

Table 5.2: Mean Recall (R) of AMOS-Net (AMOS), Hybrid-Net (Hybrid), NetVLAD (NVLAD) and Patch-NetVLAD (P-NVLAD) descriptors on VPR dataset splits described in 5.2.1.2.

Dataset	Descriptors	R@1	R@5	R@10
Oxford	AMOS	0.04	0.10	0.14
	Hybrid	0.04	0.11	0.16
	NVLAD	0.34	0.57	0.66
	P-NVLAD	0.22	0.35	0.42
Nordland	AMOS	0.29	0.45	0.54
	Hybrid	0.32	0.50	0.59
	NVLAD	0.48	0.65	0.73
	P-NVLAD	0.40	0.47	0.53
St. Lucia	AMOS	0.18	0.34	0.42
	Hybrid	0.21	0.38	0.48
	NVLAD	0.27	0.43	0.51
	P-NVLAD	0.22	0.32	0.38
UAH	AMOS	0.13	0.29	0.38
	Hybrid	0.16	0.35	0.45
	NVLAD	0.38	0.67	0.76
	P-NVLAD	0.31	0.51	0.61

5.2.2 Outcome Classification

This section contains experiments to test three hypotheses. The first is that DCC outperforms a baseline threshold approach for outcome classification, the second is that this approach is agnostic to the choice of VPR descriptor and the third is that the use of contrastive learning helps to separate these three overlapping classes to improve outcome classification.

5.2.2.1 Baseline

As described in Section 5.1.3.3, the descriptor comparison data from the top 25 nearest neighbours $[DC_{matches[1]}, \ldots DC_{matches[25]}]$ were compared against a threshold value

$Thresh_{class}$ and the results used to predict the descriptor comparison as a true or false positive.

5.2.2.2 Descriptor Comparison Classifiers

The four DNN encoders detailed in Section 5.1.3.4: Fully Connected (FC), Convolutional (Conv.), ResNET (Res.) and Transformer (Trans.) were prepended to the classifier and the whole network was trained for the task. For supervised contrastive learning the encoder was trained and frozen, then the classifier was appended and trained for classification.

5.2.2.3 Training

For experimentation one dataset was selected for testing and the remaining three were used for training data. For the baseline thresholding approach the top 25 nearest neighbours are extracted from the training descriptor comparisons and a baseline threshold is calibrated according to Section 5.1.3.

For the deep neural network models each training descriptor comparison was pre-processed according to Section 5.1.3, 10% of the stratified training data was used to create a validation set and the remaining 90% used to train each neural network for 50 epochs using sigmoid loss on batches of size 8 with a learning rate of 1e-5 and early stopping according to the validation loss, with a patience of 20.

For the contrastive approach each encoder was trained to separate class embeddings using supervised contrastive loss for 100 epochs with a temperature of 0.05 using a learning rate of 1e-4 and a batch size of 64. The classifier was then appended to the frozen encoder and trained with the same approach used for plain classification. Supervised contrastive loss requires pre-training for the encoder so it was difficult to ensure equivalent training for both deep neural network approaches. Early stopping was used to help offset this by giving each classifier as many epochs as required for the validation loss to stop decreasing. This process was repeated for each dataset and VPR descriptor.

5.2.2.4 Testing

Descriptor comparison samples from one of two test data splits were pre-processed and then classified using one of the deep neural networks. For the baseline thresh-

old approach $[DC_{matches[1]}, \ldots DC_{matches[25]}]$ were extracted and if one of these values was above $Thresh_{class}$ the test sample was classified as a false positive, otherwise it was classified as a true positive, in other words if one of the top 25 reference image matches includes a particularly poor match then it is considered a false positive. This approach was used because particularly large comparison values in the nearest neighbours was a good indication of the unreliability of the other matches. This process was repeated for the second test dataset split and the mean F1 score of each class is then calculated for both test dataset splits and reported. For completeness the raw experimental results are presented in Table 5.3, but for clarity these results are reinterpreted in Tables 5.4, 5.5 and 5.6. The Transformer architectures taught plainly and with supervised contrastive learning (Trans & Trans + C) failed because they collapsed all predictions into one class for all descriptors other than NetVLAD, this network's results were therefore only displayed in this table and not included in calculating the mean scores for the other tables.

5.2.2.5 Is DCC Better Than Thresholding for Outcome Classification?

Thresholding attempts to calculate an absolute descriptor comparison value that can be used to identify false positives, but because descriptor comparison data is the result of a relative comparison this absolute value becomes irrelevant if the relationship between the query and reference test data is significantly different from the training data, despite all descriptor comparison data being normalised between 0 and 1. This leads to the threshold being too low and classifying all the test data as a false positive, or too high and classifying all the test data as a true positive thereby collapsing all of the outcome classification predictions into one class. These results were therefore omitted and replaced with a dash (-).

Table 5.4 shows thresholding collapsed on 7 of the 16 dataset and descriptor combinations for outcome classification. Furthermore, in five of the nine examples DCC outperforms thresholding by a mean F1 score of up to 0.21 (AMOS descriptor, St. Lucia dataset, Thresh. vs. DNN). This result confirms the hypothesis that this chapter's DCC approach outperforms the threshold approach for outcome classification, although there is still significant room for improvement on this new and challenging task.

Table 5.3: Raw mean true positive, false positive F1 outcome classification scores of baseline thresholding, DNN and DNN with contrastive learning (+C) classification approaches on four datasets using four VPR descriptors, as described in Section 5.2.2. The best scores between networks with or without contrastive loss are shown in **bold** (higher the better).

		Datasets												Dataset
		Oxford			Nordland			St. Lucia			UAH			
Descriptor	**Classifier**													**Mean**
AMOS	Thresh.	-	-	-	0.52	0.56	0.54	0.58	0.10	0.34	-	-	-	-
	FC	0.25	0.80	**0.53**	0.16	0.65	0.40	0.46	0.65	**0.56**	0.08	0.77	**0.44**	0.48
	FC + C	0.18	0.85	0.52	0.31	0.68	**0.49**	0.48	0.60	0.54	0.06	0.76	0.41	**0.49**
	Conv.	0.25	0.84	**0.54**	0.25	0.65	**0.45**	0.56	0.53	0.55	0.10	0.77	0.43	0.49
	Conv. + C	0.22	0.84	0.53	0.17	0.67	0.42	0.45	0.67	**0.56**	0.12	0.76	**0.44**	**0.49**
	Res.	0.29	0.80	**0.54**	0.27	0.63	0.45	0.55	0.52	**0.54**	0.13	0.77	0.45	0.50
	Res. + C	0.26	0.76	0.51	0.36	0.63	**0.50**	0.53	0.52	0.52	0.26	0.75	**0.50**	**0.51**
Hybrid	Thresh.	-	-	-	0.65	0.41	0.53	0.63	0.25	0.44	0.56	0.55	0.56	-
	FC	0.29	0.56	0.43	0.30	0.61	0.45	0.59	0.41	**0.50**	0.05	0.71	0.38	0.44
	FC + C	0.28	0.77	**0.52**	0.39	0.60	**0.50**	0.57	0.42	**0.50**	0.06	0.71	**0.39**	**0.48**
	Conv.	0.30	0.64	0.47	0.45	0.60	**0.53**	0.62	0.28	0.45	0.07	0.71	0.39	0.46
	Conv. + C	0.28	0.74	**0.51**	0.38	0.58	0.48	0.58	0.42	**0.50**	0.14	0.70	**0.42**	**0.48**
	Res.	0.30	0.66	0.48	0.40	0.59	0.50	0.62	0.25	0.43	0.08	0.71	0.39	0.45
	Res. + C	0.28	0.70	**0.49**	0.48	0.57	**0.52**	0.60	0.42	**0.51**	0.13	0.70	**0.42**	**0.49**
NVLAD	Thresh.	-	-	-	-	-	-	0.67	0.26	0.46	0.85	0.33	0.59	-
	FC	0.79	0.31	0.55	0.56	0.49	0.52	0.68	0.20	0.44	0.86	0.20	0.53	0.51
	FC + C	0.76	0.45	**0.61**	0.64	0.47	**0.56**	0.67	0.30	**0.48**	0.86	0.32	**0.59**	**0.56**
	Conv.	0.79	0.40	**0.59**	0.60	0.49	0.54	0.68	0.26	0.44	0.87	0.15	0.51	0.52
	Conv. + C	0.78	0.32	0.55	0.73	0.44	**0.59**	0.68	0.21	**0.47**	0.84	0.30	**0.57**	**0.55**
	Res.	0.80	0.33	0.57	0.58	0.49	0.53	0.68	0.19	0.44	0.87	0.29	0.58	0.53
	Res. + C	0.74	0.43	**0.58**	0.64	0.45	**0.55**	0.68	0.33	**0.50**	0.83	0.36	**0.60**	**0.56**
	Trans.	0.79	0.34	0.56	0.59	0.49	0.54	0.68	0.19	0.39	0.87	0.19	0.53	0.51
	Trans. + C	0.77	0.48	**0.63**	0.64	0.51	**0.57**	0.68	0.25	**0.46**	0.86	0.33	**0.60**	**0.57**
P-NVLAD	Thresh.	-	-	-	-	-	-	0.51	0.10	0.30	0.14	0.49	0.32	-
	FC	0.52	0.45	0.48	0.19	0.57	0.38	0.48	0.54	**0.51**	0.50	0.44	**0.47**	**0.46**
	FC + C	0.53	0.47	**0.50**	0.32	0.52	**0.42**	0.49	0.49	0.49	0.46	0.40	0.43	**0.46**
	Conv.	0.53	0.38	0.45	0.26	0.54	0.40	0.50	0.48	**0.49**	0.45	0.43	**0.44**	**0.45**
	Conv. + C	0.47	0.49	**0.48**	0.37	0.50	**0.44**	0.49	0.42	0.45	0.44	0.44	**0.44**	**0.45**
	Res.	0.55	0.19	0.37	0.30	0.53	0.41	0.50	0.36	0.43	0.37	0.46	0.42	0.41
	Res. + C	0.45	0.50	**0.47**	0.42	0.48	**0.45**	0.44	0.49	**0.46**	0.49	0.38	**0.44**	**0.46**

Table 5.4: Comparison of mean F1 outcome classification scores of baseline thresholding vs. all deep neural network (DNN) approaches (plain and contrastive) for DCC. Changes in mean F1 scores (Δ) are positive and negative.

		Datasets				Dataset
Descriptor	**Classifier**	**Oxford**	**Nordland**	**St. Lucia**	**UAH**	**Mean**
AMOS	Thresh.	-	**0.54**	0.34	-	-
	DNN	**0.53**	0.45	**0.55**	**0.45**	**0.50**
	Δ	**n/a**	**-0.09**	**0.21**	**n/a**	**n/a**
Hybrid	Thresh.	-	**0.53**	0.44	**0.56**	-
	DNN	**0.49**	0.50	**0.48**	0.40	**0.47**
	Δ	**n/a**	**-0.03**	**0.04**	**-0.16**	**n/a**
NVLAD	Thresh.	-	-	**0.46**	**0.59**	-
	DNN	**0.58**	**0.56**	**0.46**	**0.57**	**0.54**
	Δ	**n/a**	**n/a**	**0.00**	**-0.02**	**n/a**
P-NVLAD	Thresh.	-	-	0.30	0.32	-
	DNN	**0.46**	**0.42**	**0.48**	**0.44**	**0.45**
	Δ	**n/a**	**n/a**	**0.18**	**0.12**	**n/a**

5.2.2.6 Does DCC for Outcome Classification Work with Different Descriptors?

The results in Table 5.5 show that, despite the large changes in VPR performance between the descriptors (Table 5.2), comparable performance for outcome classification is achieved. For example, the mean VPR recall of the best performing NetVLAD descriptors across all 4 datasets is 0.67 at R@10, compared to the worst performing AMOS descriptor with a mean R@10 of 0.37. However, despite this decrease of 45% in recall performance AMOS descriptors produce the second highest average F1 score across all DNN approaches (Table 5.5), which is only 7% lower than the OC results achieved with NetVLAD. Additionally, the range of mean F1 scores between descriptors is only 0.09. These results confirm the hypothesis that DCC can be used for outcome classification using data from different descriptors.

However, DCC is not totally agnostic to the VPR performance of each descriptor because NetVLAD descriptors provide the best VPR recall and the best OC performance in all cases. This makes sense intuitively because better VPR descriptors are likely to include more consistent results for true positive results in the top nearest neighbours, which should make them easier to differentiate from false positives.

Table 5.5: Comparison of mean F1 outcome classification scores of all deep neural network (DNN) approaches (plain and contrastive) for DCC.

	Descriptor			
Classifier	**AMOS**	**Hybrid**	**NVLAD**	**P-NVLAD**
DNN	0.50	0.47	**0.54**	0.45

5.2.2.7 Which Deep Neural Network Architecture Performs the Best for Outcome Classification?

Different DNN architectures only slightly affect performance for outcome classification, as shown in Table 5.6. The convolutional network performs slightly better than the others (+0.01) when trained normally, perhaps because the convolutional network is best at processing the local coherence present in the descriptor comparison data as a result of the pre-processing detailed in Section 5.1.3.2. The addition of supervised contrastive learning results allows ResNET to slightly outperform the other networks (+0.01). Interestingly, the transformer architecture was only able to reliably produce results using NetVLAD descriptors. The reason for this requires further research, but suggests that it benefits from high accuracy VPR descriptors. When the Transformer

does work it provides the best F1 classification score on the Oxford and UAH datasets 0.63 and 0.60 respectively in Table 5.3 (NVLAD, Trans+C., Oxford and UAH).

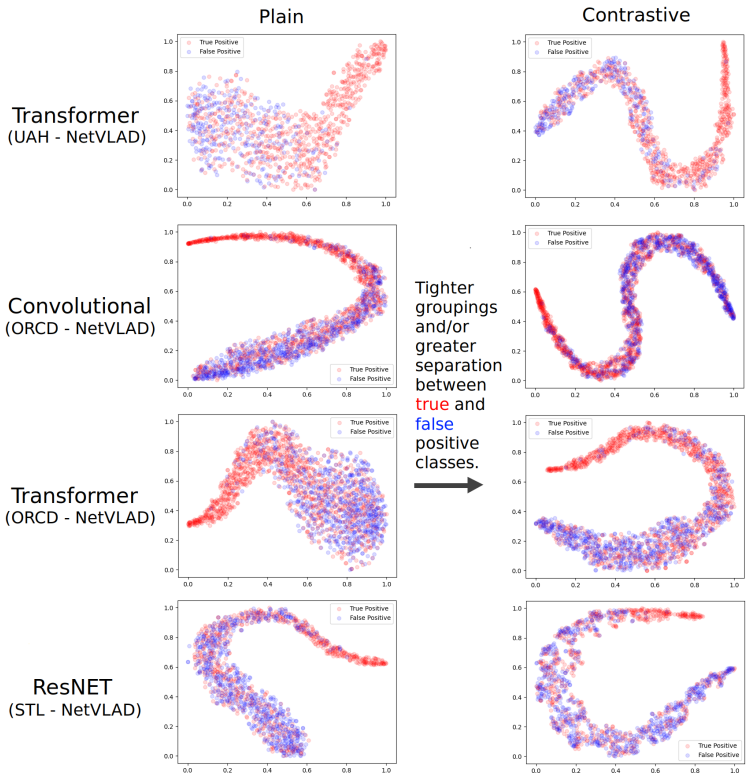


Figure 5.20: t-SNE plots showing the improved separation and class grouping between true and false positive classes when using supervised constrastive learning for outcome classification.

5.2.2.8 Does Supervised Contrastive Learning Improve Descriptor Comparison Classification?

Table 5.6 shows that supervised contrastive learning improves mean F1 scores by up to 0.04. Supervised contrastive learning relies on separating descriptor comparisons of true and false positives, which these results suggest is more advantageous with Hybrid and NetVLAD descriptors. This is not because Hybrid and NetVLAD descriptor comparisons are the most accurate, because Hybrid descriptors have lower mean recall performance at R@10 than Patch-NetVLAD for Nordland and UAH datasets as shown in Table 5.2. One possibility is that Hybrid and NetVLAD networks are both only trained down to their conv5 layers for visual place recognition, leaving the remainder of the network pre-trained for general feature extraction. Comparisons between descriptors that prioritise more general features may therefore be more pre-

dictable in their representation of the true and false positive classes used for outcome classification. Supporting evidence for this theory is that using supervised contrastive learning with AMOS data provides the smallest mean improvement (+0.01) and the only difference between Hybrid and AMOS descriptors is that AMOS descriptors have been taught purely for VPR. Figure 5.20 uses t-SNE plots of class features to illustrate the effects of supervised contrastive learning on the class representations.

Table 5.6: Comparison of different DNN architectures for DCC and the effect of supervised contrastive learning on outcome classification. The mean F1 scores are calculated across four datasets. Changes in mean F1 scores (Δ) are positive and negative.

	Descriptor				Descriptor
Classifier	**AMOS**	**Hybrid**	**NVLAD**	**P-NVLAD**	**Mean**
FC	0.48	0.44	0.51	**0.46**	0.47
FC + C	**0.49**	**0.48**	**0.56**	**0.46**	**0.50**
Δ	**0.01**	**0.04**	**0.05**	**0.00**	**0.03**
Conv.	**0.49**	0.46	0.52	**0.45**	0.48
Conv. + C	**0.49**	**0.48**	**0.55**	**0.45**	**0.49**
Δ	**0.00**	**0.02**	**0.03**	**0.00**	**0.01**
Res.	0.50	0.45	0.53	0.41	0.47
Res. + C	**0.51**	**0.49**	**0.56**	**0.46**	**0.51**
Δ	**0.01**	**0.04**	**0.03**	**0.05**	**0.04**

5.2.3 Open Set Recognition

This section contains experiments to test three hypotheses. The first is that DCC outperforms a baseline threshold approach for open set recognition. The second is that this approach is agnostic to the choice of VPR descriptor and the third is that the use of contrastive learning helps to separate overlapping closed and open set classes to improve open set recognition.

5.2.3.1 Baseline, Descriptor Comparison Classifiers, Training and Testing

These sections were repeated exactly from Sections 5.2.2.1 to 5.2.2.4, the one difference is that OC training and test data were substituted for OSR data. For completeness the raw experimental results are presented in Table 5.7, but for clarity these results are reinterpreted in Tables 5.8, 5.9 and 5.10. The Transformer architectures taught plainly and with supervised contrastive learning (Trans & Trans + C) failed because they collapsed all predictions into one class for all descriptors other than NetVLAD, this network's results were therefore only displayed in this table and not included in calculating the mean scores for the other tables.

Table 5.7: Raw mean closed set, open set and mean F1 open set recognition scores of baseline thresholding, DNN and DNN with contrastive learning (+C) classification approaches on four datasets using four VPR descriptors, as described in Section 5.2.3. The best scores between networks with or without contrastive loss are shown in **bold** (higher the better)

		Datasets												Dataset
		Oxford			Nordland			St. Lucia			UAH			
Descriptor	**Classifier**													**Mean**
AMOS	Thresh.	0.19	0.58	0.38	0.66	0.44	0.55	-	-	-	0.19	0.66	0.42	-
	FC	0.72	0.03	0.37	0.64	0.01	0.32	0.72	0.23	**0.48**	0.66	0.01	0.33	0.38
	FC + C	0.67	0.24	**0.45**	0.66	0.03	**0.34**	0.69	0.24	0.46	0.66	0.03	**0.34**	**0.40**
	Conv.	0.59	0.36	**0.47**	0.67	0.01	0.33	0.69	0.24	**0.47**	0.66	0.01	0.34	0.40
	Conv. + C	0.62	0.31	**0.47**	0.62	0.14	**0.38**	0.61	0.23	0.42	0.64	0.23	**0.43**	**0.43**
	Res.	0.53	0.42	**0.47**	0.66	0.02	0.34	0.74	0.23	**0.48**	0.66	0.01	0.34	0.41
	Res. + C	0.58	0.36	**0.47**	0.62	0.22	**0.42**	0.66	0.23	0.44	0.59	0.36	**0.47**	**0.45**
Hybrid	Thresh.	-	-	-	0.66	0.43	0.55	0.87	0.14	0.51	0.61	0.54	0.58	-
	FC	0.72	0.02	0.38	0.64	0.01	0.33	0.54	0.25	0.39	0.66	0.01	0.33	0.36
	FC + C	0.71	0.08	**0.39**	0.65	0.03	**0.34**	0.66	0.25	**0.45**	0.66	0.02	**0.34**	**0.38**
	Conv.	0.66	0.27	0.46	0.66	0.01	0.33	0.48	0.26	0.37	0.66	0.71	0.33	0.37
	Conv. + C	0.59	0.36	**0.48**	0.65	0.10	**0.38**	0.55	0.26	**0.41**	0.60	0.29	**0.45**	**0.43**
	Res.	0.71	0.05	0.38	0.65	0.02	0.34	0.58	0.24	0.41	0.66	0.01	0.34	0.37
	Res. + C	0.58	0.41	**0.49**	0.62	0.30	**0.46**	0.60	0.26	**0.43**	0.57	0.44	**0.51**	**0.47**
NVLAD	Thresh.	0.67	0.03	0.35	0.67	0.29	0.34	0.14	0.27	0.20	0.67	0.26	0.46	0.34
	FC	0.65	0.57	**0.61**	0.64	0.54	**0.59**	0.56	0.29	0.43	0.69	0.37	0.53	0.54
	FC + C	0.63	0.54	0.59	0.61	0.47	0.54	0.69	0.28	**0.48**	0.69	0.43	**0.56**	**0.54**
	Conv.	0.58	0.61	**0.60**	0.65	0.47	**0.56**	0.56	0.29	0.43	0.70	0.48	**0.59**	**0.55**
	Conv. + C	0.55	0.60	0.58	0.64	0.42	0.53	0.73	0.25	**0.49**	0.69	0.45	0.57	0.54
	Res.	0.57	0.62	**0.6**	0.64	0.49	**0.56**	0.63	0.28	0.46	0.69	0.38	**0.53**	**0.54**
	Res. + C	0.59	0.58	0.58	0.61	0.47	0.54	0.72	0.27	**0.50**	0.68	0.39	**0.53**	**0.54**
	Trans.	0.63	0.58	0.60	0.63	0.58	**0.61**	0.66	0.30	**0.48**	0.69	0.50	0.60	**0.57**
	Trans. + C	0.62	0.59	**0.61**	0.64	0.57	**0.61**	0.53	0.29	0.41	0.70	0.50	**0.62**	0.56
P-NVLAD	Thresh.	-	-	-	0.68	0.13	0.40	0.89	0.12	0.50	0.35	0.72	0.54	-
	FC	0.66	0.37	0.51	0.64	0.06	0.35	0.36	0.27	0.32	0.66	0.07	0.37	0.39
	FC + C	0.59	0.44	**0.52**	0.64	0.17	**0.40**	0.57	0.24	**0.41**	0.66	0.12	**0.39**	**0.43**
	Conv.	0.49	0.52	**0.51**	0.65	0.11	**0.38**	0.42	0.26	0.34	0.66	0.08	0.37	0.40
	Conv. + C	0.49	0.51	0.50	0.65	0.10	**0.38**	0.59	0.24	**0.41**	0.67	0.24	**0.45**	**0.44**
	Res.	0.44	0.55	0.49	0.65	0.09	0.37	0.47	0.25	0.36	0.66	0.06	0.36	0.40
	Res. + C	0.52	0.52	**0.52**	0.61	0.31	**0.46**	0.58	0.24	**0.41**	0.66	0.31	**0.49**	**0.47**

5.2.3.2 Is DCC Better Than Thresholding for Open Set Recognition?

As discussed in Section 5.2.2, the main weakness of thresholding is that it uses an absolute value, which can result in a collapse of classification predictions into one class if the test data is very different to the training data. Table 5.8 shows thresholding failed in this way on 3 of the 16 dataset and descriptor combinations, these results were omitted and replaced with a dash (-).

Thresholding works better for OSR than OC, with it outperforming DCC in 8 out of 16 tests. Intuitively this makes sense because comparing an open set query image with reference images should result in generally higher descriptor comparison values than a closed set image, which makes thresholding with an absolute value easier. However, DCC outperformed thresholding in all cases for NetVLAD descriptors, which performed the best for VPR (Table 5.2). These results suggest an improved ability to match query and reference images for VPR may result in a tendency to overconfidently match open set query images which makes an absolute threshold less reliable, but the underlying descriptor comparisons are more consistent and therefore easier to classify.

Although thresholding and DCC perform the best on an equal number of dataset and descriptor combinations thresholding fails on three and is therefore an unreliable approach to this problem. Furthermore DCC outperforms thresholding on the best performing VPR descriptor NetVLAD in all cases. This evidence supports the hypothesis that DCC is better than thresholding for open set recognition, but that thresholding works better for descriptors that are less specialised for the task.

Table 5.8: Comparison of mean F1 outcome classification scores of baseline thresholding vs. all deep neural network (DNN) approaches (plain and contrastive) for DCC. Changes in mean F1 scores (Δ) are positive and negative.

		Datasets				Dataset
Descriptor	**Classifier**	**Oxford**	**Nordland**	**St. Lucia**	**UAH**	**Mean**
AMOS	Thresh.	0.38	**0.55**	-	**0.42**	-
	DNN	**0.45**	0.36	**0.46**	0.38	**0.42**
	Δ	**0.07**	**-0.19**	**n/a**	**-0.04**	**n/a**
Hybrid	Thresh.	-	**0.55**	**0.51**	**0.58**	-
	DNN	**0.43**	0.37	0.41	0.39	**0.40**
	Δ	**n/a**	**-0.18**	**-0.10**	**-0.19**	**n/a**
NVLAD	Thresh.	0.35	0.34	0.20	0.46	0.34
	DNN	**0.60**	**0.57**	**0.46**	**0.57**	**0.55**
	Δ	**0.25**	**0.24**	**0.27**	**0.11**	**0.21**
P-NVLAD	Thresh.	-	**0.40**	**0.50**	**0.54**	-
	DNN	**0.51**	0.39	0.38	0.41	**0.43**
	Δ	**n/a**	**-0.01**	**-0.09**	**-0.10**	**n/a**

5.2.3.3 Does DCC for Open Set Recognition Work with Different Descriptors?

The results in Table 5.9 show that DCC for open set recognition is less agnostic to the VPR performance descriptors than outcome classification because there is a larger range of F1 scores: 0.09 for OC and 0.15 for OSR. However, despite a decrease of 45% between the mean VPR recall of the best performing NetVLAD descriptors across all 4 datasets from 0.67 compared to AMOS's 0.37 at R@10 there is only a 24% decrease (Table 5.9) in open set recognition from 0.55 (NetVLAD) to 0.42 (AMOS). These results confirm the hypothesis that the proposed approach works with different descriptors.

Table 5.9: Comparison of mean F1 open set recognition scores of all deep neural network (DNN) approaches (plain and contrastive) for DCC.

	Descriptor			
Classifier	**AMOS**	**Hybrid**	**NVLAD**	**P-NVLAD**
DNN	0.42	0.40	**0.55**	0.43

5.2.3.4 Which Deep Neural Network Architecture Performs the Best for Open Set Recognition?

Different DNN architectures only slightly affect open set recognition, as shown in Table 5.10. When trained normally convolutional and ResNET networks achieve a slightly higher mean F1 classification score than the fully connected network (+0.01), but when trained with supervised contrastive loss the ResNet network achieves a slight increase in the best mean F1 classification score (+0.02). As in outcome classification, the Transformer was only able to reliably produce results using NetVLAD descriptors but also provides the best F1 classification score on the Oxford (0.61), Nordland (0.61) and UAH (0.62) datasets, but the worst for St. Lucia (0.41) as shown in Table 5.7 (NVLAD, Trans+C., all datasets).

5.2.3.5 Does Supervised Contrastive Learning Improve Descriptor Comparison Classification?

Table 5.10 shows supervised contrastive learning improves mean F1 classification scores for OSR by up to 0.05. Supervised contrastive learning relies on separating descriptor comparisons of closed and open set images, which these results suggest is more advantageous with AMOS, Hybrid and Patch-NetVLAD descriptor comparisons.

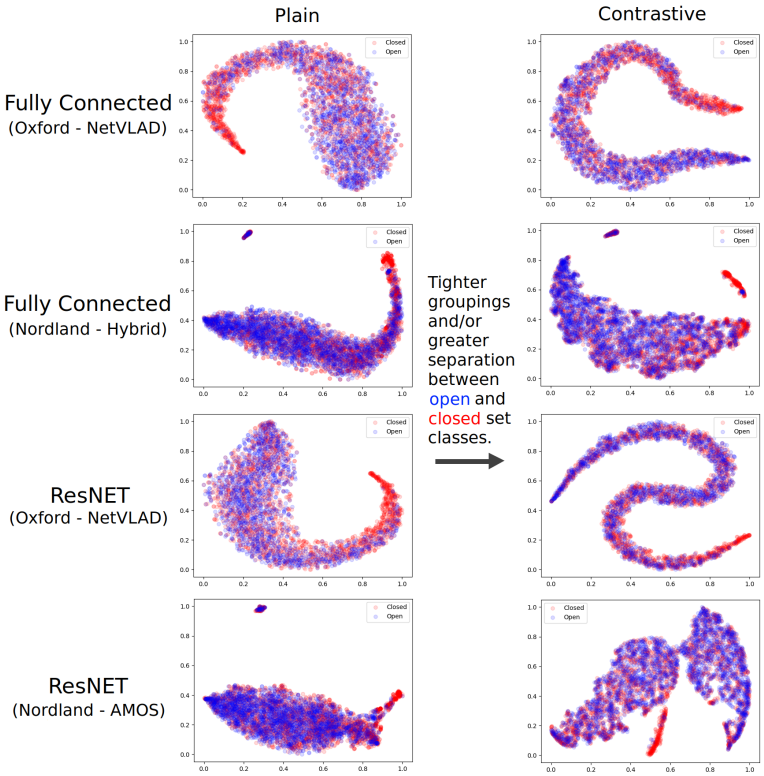


Figure 5.21: t-SNE plots showing the improved separation and class groupings between open and closed set classes when using supervised contrastive learning for open set recognition.

One reason for this may be found in Table 5.8 where thresholding is shown to be better for classifying AMOS, Hybrid and Patch-NetVLAD than NetVLAD data for this task and therefore implies some separability between the open and closed descriptor comparisons which supervised contrastive learning might be able to leverage for better performance.

Figure 5.21 uses t-SNE plots of class features to illustrate the effects of supervised contrastive learning on the class representations.

Table 5.10: Comparison of different DNN architectures for DCC and the effect of supervised contrastive learning on open set recognition. The mean F1 scores are calculated across four datasets. Changes in mean F1 scores (Δ) are positive and negative.

	Descriptor				Dataset
Classifier	**AMOS**	**Hybrid**	**NVLAD**	**P-NVLAD**	**Mean**
FC	0.38	0.36	**0.54**	0.39	0.42
FC + C	**0.40**	**0.38**	**0.54**	**0.43**	**0.44**
Δ	**0.02**	**0.02**	**0.00**	**0.04**	**0.02**
Conv.	0.40	0.37	**0.55**	0.40	0.43
Conv. + C	**0.43**	**0.43**	0.54	**0.44**	**0.46**
Δ	**0.03**	**0.06**	**-0.01**	**0.04**	**0.03**
Res.	0.42	0.37	**0.54**	0.40	0.43
Res. + C	**0.45**	**0.47**	**0.54**	**0.47**	**0.48**
Δ	**0.03**	**0.10**	**0.00**	**0.07**	**0.05**

5.3 Discussion

This chapter formulates two new problems for visual place recognition, and by extension image retrieval. Firstly, classification of image retrieval results as being true or false positives is defined as outcome classification. Secondly, open set recognition is extended for image retrieval to recognise query images that have no true match in the reference image database.

Image retrieval is particularly complex because it compares representations of unknown images with each other, rather than simply classifying images into a pre-defined set of classes. The main challenge associated with these two problems was therefore to classify not the images, but the relationship between them.

To address this challenge descriptor comparison classification is proposed. The top results from the comparisons' nearest neighbours and their adjacent descriptors are used to train a deep neural network to classify these comparisons as resulting from true or false positive results, or from closed or open set query images. Deep neural networks designed for signal classification are used for this task and supervised con-

trastive learning is used to separate overlapping descriptor comparisons to improve classification on both tasks.

The proposed approach was evaluated on four state-the-art descriptors, datasets and deep neural network architectures. The use of deep neural networks for descriptor comparison classification is shown to be reliable compared to using a threshold on the descriptor comparison values, which is shown to be unsuccessful for a number of descriptor and dataset combinations. These experiments also show supervised contrastive learning improves results on outcome classification and open set recognition.

5.3.1 Limitations and Future Work

As this final chapter introduces two challenging new problems and a benchmark approach to address each one there is a large amount of work still to be done.

Real-world applications of deep learning are increasingly being required to produce an associated confidence with their prediction, for example in [196]. The results of image retrieval are inherently difficult to interpret so tasks such as outcome classification are progressively becoming more important. Open set recognition is a key problem for open world deployment of deep learning classification systems, which is particularly important for robotic applications. Recognising previously unknown images for image retrieval also potentially enables more advanced deep learning techniques, such as continual learning.

Combining open set recognition and outcome classification into one network was attempted and found to be impossible at this stage, but may be possible in the future as currently they both use very similar data. Furthermore, outcome classification only attempts to classify descriptor comparisons as binary true or false positive states, but this could be extended to include more precise recall predictions. Further unsuccessful experiments confirmed this was also currently impossible.

Overfitting was observed when training the deep neural network classifiers for descriptor comparison classification, which indicates that the training data is too dissimilar to the test data for generalisation. The relative similarity in classification performance between different deep learning classifiers suggests that the underlying data is a more significant factor in deciding performance. Adding noise to the descriptor comparisons was experimented with for data augmentation, but did not improve generalisation, which suggests that specific transforms need to be applied for effective aug-

mentation. The relative similarity of image descriptors within the reference database significantly affects the descriptor comparisons, sampling the reference database at various intervals and varying splits of open and closed data would reflect this and therefore perhaps allow for more effective data augmentation. Alternative approaches to sampling key information from the descriptor comparisons, which these results suggest overlapped significantly for both tasks, may also be effective.

One weakness of the approaches proposed here is that the classifiers are descriptor dependent, more work needs to be done to establish whether a classifier could be trained that generalised to different descriptors, but this seems unlikely given the wide range of descriptor characteristics observed ion this chapter. A better approach may be to use a statistical approach similar to the one used by openmax which would be descriptor agnostic. Although this approach may be difficult to apply to outcome classification.

Descriptor comparison classification is shown to work for both open set recognition and outcome classification, but is dependent on descriptor characteristics that are hard to quantify. Considering outcome classification and open set recognition when creating descriptors for end-to-end training may be beneficial for both tasks and also result in improved visual place recognition. An ensemble approach of descriptors trained for complimentary tasks may also be effective.

Another limitation of this approach is that it only leverages relationships between descriptors available at test time. A strength of this is that it requires no other information, but a weakness is that it fails to consider fixed descriptor characteristics that could be derived from a calibration procedure. For example, if a distance between descriptors is particularly high for some descriptor comparisons then they may be intrinsically unreliable. A combination of calibrated thresholding and descriptor comparison classification could help to improve this situation.

Part III

Conclusions

Chapter 6

Conclusion

This thesis began by introducing visual place recognition (VPR) as the localisation of a query place image by comparing it against a reference database of place images. A number of outstanding problems with state-of-the-art VPR for robotic navigation in open and uncertain environments were then identified:

- **Computational requirements** of DNNs used to generate state-of-the-art VPR descriptors [10, 13] make them impractical for use in robotics [16].
- **Integration** of deep learnt descriptors into visual SLAM pipelines is difficult [6] because they were designed for descriptors with different characteristics [4].
- **Ambiguous** predicted similarities between place images make VPR results difficult to interpret.
- **Non-transferable** appearance invariant VPR descriptors cannot be used for appearance invariance in other navigation tasks such as scene classification [19, 20].
- **Open set** scenarios, fundamental to robotic navigation in an open world, have not been considered for visual place recognition.

6.1 Contributions and Limitations

Three novel contributions were presented to address these problems.

6.1.1 Particle Filtering for Robust Real-Time Visual Teach and Repeat

Chapter 3 presents a compact 'TinyVPR' deep neural network that can be taught on a per-route basis to generate appearance invariant descriptors in real-time for a subset of

visual SLAM called visual teach and repeat. A particle filter is also proposed to combine the TinyVPR descriptors with visual odometry for route repetition that is robust to variations in speed and large visual changes. This approach was found to significantly reduce localisation error when compared against the state-of-the-art [132, 131]. However, the parameters of the probability distributions used in the particle filter are difficult to estimate precisely as they depend on the performance of the TinyVPR descriptor and visual odometry in an unknown environment. Another weakness of this approach is that training a network on a per-route basis requires reference examples of it in multiple appearance variations which may not be available.

6.1.2 OpenSceneVLAD: Appearance Invariant, Open Set Scene Classification

Chapter 4 presents a fusion of scene classification [156] and appearance invariant visual place recognition descriptor [10] deep neural networks called SceneVLAD for appearance invariant scene classification. This is then extended to OpenSceneVLAD which uses atypical class examples to retrain the network for an open set visual navigation scenario where scene classes must be classified amongst an open set of previously undefined classes. These approaches are compared to state-of-the-art scene classification [156] and open set classification [21] approaches and shown to increase performance. For evaluation, a new dataset for open set and appearance invariant scene classification was extracted from new and existing visual localisation datasets. This approach is principally limited by the small number of scene classes with appearance variations available for evaluation. Additionally, training for OpenSceneVLAD assumes a suitable range of atypical class examples that can be used as synthetic members of an open class which may not always be available.

6.1.3 Descriptor Comparison Classification for Open Set Recognition and Outcome Classification

Chapter 5 formulates and addresses the problem of open set recognition and outcome classification for the first time in the context of visual place recognition. Open set recognition is the problem of predicting whether query images have a true match in the reference database, or not and is analogous to detecting if a robot is lost. Outcome classification is the problem of predicting whether the top 10 matches of a query image

include a true match, or not and is analogous to estimating the reliability of a visual place recognition result. To address these tasks classification of the actual comparisons between descriptors is proposed and compared against confidence thresholding the raw comparisons using state-of-the-art descriptors [14, 10, 13]. A number of deep neural networks are proposed for classification and supervised contrastive learning is also used to improve separation between the target classes. Results show that this is the only approach that reliably solves both problems. The main weakness of this approach is that it is the first attempt to solve these problems so classification accuracy can still be significantly improved, particularly by improving generalisation between training and test data. Another weakness of this approach is that separate networks are required for each classification task.

6.2 Future Work

Deep learnt image descriptors specialised for visual place recognition are shown to significantly decrease localisation error when combined with a probabilistic filter for visual teach and repeat which indicates their potential for improving visual SLAM. However, they are yet to be commonly implemented in state-of-the-art approaches despite their increased robustness to appearance variations, which is a major limitation of current approaches. Possible future work could implement specialised visual place recognition descriptors on embedded robotic hardware in a specially designed visual SLAM pipeline for improve appearance invariance.

Open set recognition and outcome classification for visual place recognition are introduced in this thesis and represent key problems for robotic navigation, and image retrieval generally, that justify future work. At the heart of these problems is the need to estimate the uncertainty for descriptor comparisons. Using Bayesian neural networks (introduced in Section 2.8) to explicitly predict descriptor uncertainties, could allow these problems to be addressed without a secondary classifier. Calibration of the deep neural network used for descriptor generation could also be a promising future research direction that is descriptor agnostic.

Throughout this thesis relatively simple, but well-trained deep convolutional neural networks, such as NetVLAD [10], have been shown to be remarkably versatile for a variety of visual place recognition scenarios and remain competitive against more

complex approaches. Recent advances in deep neural networks for image classification, such as the introduction of convolutional vision transformers [227], opens up the possibility of creating improved networks that use a learned attention mechanism for improved VPR descriptor generation.

This thesis gives an insight into the overlap between image classification and retrieval that may have implications for future research into more general areas of deep learning. For example, image retrieval uses similarity learning to learn a relationship between data that generalises to comparisons between previously unknown test and reference data. Using similarity learning to train a network to cluster general image classes may allow test images from previously undefined classes to be classified by calculating their similarity to samples of previously undefined image classes and may therefore have applications for zero shot learning.

Part IV

Appendix

Appendix A

Source Code and Dataset

In this appendix links are provided to the software (which has, or will be released) in addition to the datasets developed and collected for this thesis.

A.1 Source Code

The open-source Python implementation for SceneVLAD and OpenSceneVLAD, the method proposed in Chapter 4, can be found here:

https://github.com/WHBSmith

A.1.1 Edinburgh Visual Navigation and Scene Classification Dataset

The Edinburgh dataset used for the evaluation in Chapter 4 is available here:

https://github.com/WHBSmith/Edinburgh_VPR

It consists of three traversals of one 19.5km route around Edinburgh, the Scottish capital city in urban, rural and motorway environments: 20210524 (overcast), 20210526 (evening) and 20210804 (sunny) using a dash-mounted OnePlus 7T recording 4k video at 30fps and a GPS logger app. This dataset is designed for visually invariant place recognition but has also been hand annotated to identify four classes within them and all remaining images are labelled as open set images.

A.1.2 Nordland and Oxford Visual Navigation and Scene Classification Dataset Subset

The subset of the Nordland [133] and Oxford [115] datasets along with the scene labels used for evaluation in Chapter 4 are available here:

https://github.com/WHBSmith/Nordland_Oxford_VPR

The first dataset subset is of the Nordland dataset, a 763km train journey through rural Norway. The traversals used are: spring, winter and summer. The second subset is of the 9km urban Oxford RobotCar dataset. The three traversals used are: 2015-07-03-15-23-28 (overcast), 2014-12-16-18-44-24 (night) and 2015-03-24-13-47-33 (sunny). These dataset subsets were designed for visually invariant place recognition but have also been hand annotated to identify four classes within them and all remaining images are labelled as open set images:

Bibliography

[1] S. Lowry, N. Sünderhauf, P. Newman, J. J. Leonard, D. Cox, P. Corke, and M. J. Milford, "Visual place recognition: A survey," *IEEE Trans. Robotics*, vol. 32, no. 1, pp. 1–19, 2015.

[2] S. Garg, T. Fischer, and M. Milford, "Where is your place, visual place recognition?" in *Intl. Joint Conf. on Artificial Intelligence*, 2021.

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

[4] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "Monoslam: Real-time single camera slam," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 29, no. 6, pp. 1052–1067, 2007.

[5] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras," *IEEE Transactions on Robotics*, 2017.

[6] S. Sumikura, M. Shibuya, and K. Sakurada, "Openvslam: a versatile visual slam framework," in *ACM International Conference on Multimedia*, 2019, pp. 2292–2295.

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

[8] P. Mirowski, R. Pascanu, F. Viola, H. Soyer, A. J. Ballard, A. Banino, M. Denil, R. Goroshin, L. Sifre, K. Kavukcuoglu *et al.*, "Learning to navigate in complex environments," in *Proc. Int. Conf. Learn. Represent.*, 2017, pp. 1–16.

[9] T. D. Kulkarni, K. Narasimhan, A. Saeedi, and J. Tenenbaum, "Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation," *Adv. in Neural Inf. Proc. Systems*, vol. 29, 2016.

[10] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "Netvlad: Cnn architecture for weakly supervised place recognition," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, 2016, pp. 5297–5307.

[11] P. Panphattarasap and A. Calway, "Automated map reading: image based localisation in 2-d maps using binary semantic descriptors," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 6341–6348.

[12] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Intl. Conf. on Learning Representations (ICLR)*, 2015.

[13] S. Hausler, S. Garg, M. Xu, M. Milford, and T. Fischer, "Patch-netvlad: Multi-scale fusion of locally-global descriptors for place recognition," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, 2021, pp. 14 141–14 152.

[14] Z. Chen, A. Jacobson, N. Sünderhauf, B. Upcroft, L. Liu, C. Shen, I. Reid, and M. Milford, "Deep learning features at scale for visual place recognition," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 3223–3230.

[15] Y. Kong, W. Liu, and Z. Chen, "Robust convnet landmark-based visual place recognition by optimizing landmark matching," *IEEE Access*, vol. 7, pp. 30 754–30 767, 2019.

[16] K. Tateno, F. Tombari, I. Laina, and N. Navab, "Cnn-slam: Real-time dense monocular slam with learned depth prediction," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, 2017, pp. 6243–6252.

[17] Z. Chen, F. Maffra, I. Sa, and M. Chli, "Only look once, mining distinctive landmarks from convnet for visual place recognition," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 9–16.

[18] Z. Wang, J. Li, S. Khademi, and J. van Gemert, "Attention-aware age-agnostic visual place recognition," in *Intl. Conf. on Computer Vision (ICCV)*, 2019, pp. 0–0.

[19] D. Zeng, M. Liao, M. Tavakolian, Y. Guo, B. Zhou, D. Hu, M. Pietikäinen, and L. Liu, "Deep learning for scene classification: A survey," *arXiv*, 2021.

[20] L. Xie, F. Lee, L. Liu, K. Kotani, and Q. Chen, "Scene recognition: A comprehensive survey," *Pattern Recognition*, vol. 102, p. 107205, 2020.

[21] A. Bendale and T. E. Boult, "Towards open set deep networks," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, 2016, pp. 1563–1572.

[22] S. Kong and D. Ramanan, "Opengan: Open-set recognition via open data generation," in *Intl. Conf. on Computer Vision (ICCV)*, 2021, pp. 813–822.

[23] A. Macario Barros, M. Michel, Y. Moline, G. Corre, and F. Carrel, "A comprehensive survey of visual slam algorithms," *Robotics*, vol. 11, no. 1, p. 24, 2022.

[24] Y. Matsumoto, M. Inaba, and H. Inoue, "Visual navigation using view-sequenced route representation," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, vol. 1. IEEE, 1996, pp. 83–88.

[25] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Intl. J. of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[26] H. Bay, T. Tuytelaars, and L. V. Gool, "Surf: Speeded up robust features," in *Eur. Conf. on Computer Vision (ECCV)*. Springer, 2006, pp. 404–417.

[27] S. Se, D. Lowe, and J. Little, "Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks," *Intl. J. of Robotics Research*, vol. 21, no. 8, pp. 735–758, 2002.

[28] S. Se, D. G. Lowe, and J. J. Little, "Vision-based global localization and mapping for mobile robots," *IEEE Trans. Robotics*, vol. 21, no. 3, pp. 364–375, 2005.

[29] G. Klein and D. Murray, "Parallel tracking and mapping for small ar workspaces," in *IEEE and ACM international symposium on mixed and augmented reality*. IEEE, 2007, pp. 225–234.

[30] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "Orb-slam: a versatile and accurate monocular slam system," *IEEE Trans. Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[31] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *Intl. Conf. on Computer Vision (ICCV)*. Ieee, 2011, pp. 2564–2571.

[32] E. Rosten, R. Porter, and T. Drummond, "Faster and better: A machine learning approach to corner detection," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 32, no. 1, pp. 105–119, 2008.

[33] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "Brief: Binary robust independent elementary features," in *Eur. Conf. on Computer Vision (ECCV)*. Springer, 2010, pp. 778–792.

[34] T. Taketomi, H. Uchiyama, and S. Ikeda, "Visual slam algorithms: A survey from 2010 to 2016," *IPSJ Transactions on Computer Vision and Applications*, vol. 9, no. 1, pp. 1–11, 2017.

[35] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon, "Kinectfusion: Real-time dense surface mapping and tracking," in *IEEE International Symposium on Mixed and Augmented Reality*. IEEE, 2011, pp. 127–136.

[36] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. Kelly, and A. J. Davison, "Slam++: Simultaneous localisation and mapping at the level of objects," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, 2013, pp. 1352–1359.

[37] K. Tateno, F. Tombari, and N. Navab, "When 2.5 d is not enough: Simultaneous reconstruction, segmentation and recognition on dense slam," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 2295–2302.

[38] T. D. Barfoot, C. McManus, S. Anderson, H. Dong, E. Beerepoot, C. H. Tong, P. Furgale, J. D. Gammell, and J. Enright, "Into darkness: Visual navigation based on a lidar-intensity-image pipeline," in *Robotics research*. Springer, 2016, pp. 487–504.

[39] A. Cherubini, F. Spindler, and F. Chaumette, "Autonomous visual navigation and laser-based moving obstacle avoidance," *IEEE Transactions on Intelligent Transportation Systems*, vol. 15, no. 5, pp. 2101–2110, 2014.

[40] M. Labbé and F. Michaud, "Rtab-map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation," *J. of Field Robotics*, vol. 36, no. 2, pp. 416–446, 2019.

[41] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. Montiel, and J. D. Tardós, "Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam," *IEEE Trans. Robotics*, vol. 37, no. 6, pp. 1874–1890, 2021.

[42] F. Zhu, Y. Zhu, V. Lee, X. Liang, and X. Chang, "Deep learning for embodied vision navigation: A survey," *arXiv*, 2021.

[43] A. Kendall, M. Grimes, and R. Cipolla, "Posenet: A convolutional network for real-time 6-dof camera relocalization," in *Intl. Conf. on Computer Vision (ICCV)*, 2015, pp. 2938–2946.

[44] H. Yin and S. J. Pan, "Knowledge transfer for deep reinforcement learning with hierarchical experience replay," in *AAAI Conf. on Artificial Intelligence*, 2017.

[45] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[46] D. Fried, R. Hu, V. Cirik, A. Rohrbach, J. Andreas, L.-P. Morency, T. Berg-Kirkpatrick, K. Saenko, D. Klein, and T. Darrell, "Speaker-follower models for vision-and-language navigation," *Adv. in Neural Inf. Proc. Systems*, vol. 31, 2018.

[47] U. Özaydın, T. Georgiou, and M. Lew, "A comparison of cnn and classic features for image retrieval," in *International Conference on Content-Based Multimedia Indexing (CBMI)*. IEEE, 2019, pp. 1–4.

[48] A. Sharif Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "Cnn features off-the-shelf: an astounding baseline for recognition," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, 2014, pp. 806–813.

[49] A. S. Razavian, J. Sullivan, S. Carlsson, and A. Maki, "Visual instance retrieval with deep convolutional networks," *ITE Transactions on Media Technology and Applications*, vol. 4, no. 3, pp. 251–258, 2016.

[50] L. Zheng, S. Wang, J. Wang, and Q. Tian, "Accurate image search with multi-scale contextual evidences," *Intl. J. of Computer Vision*, vol. 120, no. 1, pp. 1–13, 2016.

[51] A. Babenko, A. Slesarev, A. Chigorin, and V. Lempitsky, "Neural codes for image retrieval," in *Eur. Conf. on Computer Vision (ECCV)*. Springer, 2014, pp. 584–599.

[52] A. Babenko and V. Lempitsky, "Aggregating local deep features for image retrieval," in *Intl. Conf. on Computer Vision (ICCV)*, 2015, pp. 1269–1277.

[53] S. Pang, J. Xue, J. Zhu, L. Zhu, and Q. Tian, "Unifying sum and weighted aggregations for efficient yet effective image representation computation," *IEEE Transactions on Image Processing*, vol. 28, no. 2, pp. 841–852, 2018.

[54] Y. Liu, Y. Guo, S. Wu, and M. S. Lew, "Deepindex for accurate and efficient image retrieval," in *Proceedings of the 5th ACM on International Conference on Multimedia Retrieval*, 2015, pp. 43–50.

[55] J. Cao, Z. Huang, and H. T. Shen, "Local deep descriptors in bag-of-words for image retrieval," in *Proceedings of the on Thematic Workshops of ACM Multimedia*, 2017, pp. 52–58.

[56] T.-T. Do, T. Hoang, D.-K. L. Tan, H. Le, T. V. Nguyen, and N.-M. Cheung, "From selective deep convolutional features to compact binary representations for image retrieval," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 15, no. 2, pp. 1–22, 2019.

[57] J. Song, T. He, L. Gao, X. Xu, and H. T. Shen, "Deep region hashing for efficient large-scale instance search from images," *AAAI Conf. on Artificial Intelligence*, 2018.

[58] B. Cao, A. Araujo, and J. Sim, "Unifying deep local and global features for image search," in *Eur. Conf. on Computer Vision (ECCV)*. Springer, 2020, pp. 726–743.

[59] Y. Kalantidis, C. Mellina, and S. Osindero, "Cross-dimensional weighting for aggregated deep convolutional features," in *Eur. Conf. on Computer Vision (ECCV)*. Springer, 2016, pp. 685–701.

[60] H. J. Kim, E. Dunn, and J.-M. Frahm, "Learned contextual feature reweighting for image geo-localization," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*. IEEE, 2017, pp. 3251–3260.

[61] M. Boudiaf, J. Rony, I. M. Ziko, E. Granger, M. Pedersoli, P. Piantanida, and I. B. Ayed, "A unifying mutual information view of metric learning: cross-entropy vs. pairwise losses," in *Eur. Conf. on Computer Vision (ECCV)*. Springer, 2020, pp. 548–564.

[62] J. L. Suárez, S. García, and F. Herrera, "A tutorial on distance metric learning: Mathematical foundations, algorithms, experimental analysis, prospects and challenges," *Neurocomputing*, vol. 425, pp. 300–322, 2021.

[63] I. Melekhov, J. Kannala, and E. Rahtu, "Siamese network features for image matching," in *Intl. Conf. on Pattern Recognition (ICPR)*. IEEE, 2016, pp. 378–383.

[64] A. Gordo, J. Almazan, J. Revaud, and D. Larlus, "End-to-end learning of deep visual representations for image retrieval," *Intl. J. of Computer Vision*, vol. 124, no. 2, pp. 237–254, 2017.

[65] P. Wu, S. C. Hoi, P. Zhao, C. Miao, and Z.-Y. Liu, "Online multi-modal distance metric learning with application to image retrieval," *IEEE transactions on knowledge and data engineering*, vol. 28, no. 2, pp. 454–467, 2015.

[66] R. Cao, Q. Zhang, J. Zhu, Q. Li, Q. Li, B. Liu, and G. Qiu, "Enhancing remote sensing image retrieval using a triplet deep metric learning network," *International Journal of Remote Sensing*, vol. 41, no. 2, pp. 740–751, 2020.

[67] M. Donoser and H. Bischof, "Diffusion processes for retrieval revisited," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, 2013, pp. 1320–1327.

[68] C. Chang, G. Yu, C. Liu, and M. Volkovs, "Explore-exploit graph traversal for image retrieval," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, 2019, pp. 9423–9431.

[69] M. Caron, P. Bojanowski, A. Joulin, and M. Douze, "Deep clustering for unsupervised learning of visual features," in *Eur. Conf. on Computer Vision (ECCV)*, 2018, pp. 132–149.

[70] T. Barros, R. Pereira, L. Garrote, C. Premebida, and U. J. Nunes, "Place recognition survey: An update on deep learning approaches," *arXiv*, 2021.

[71] X. Zhang, L. Wang, and Y. Su, "Visual place recognition: A survey from deep learning perspective," *Pattern Recognition*, vol. 113, p. 107760, 2021.

[72] J. Košecká, F. Li, and X. Yang, "Global localization and relative positioning based on scale-invariant keypoints," *Journal of Robotics and Autonomous Systems*, vol. 52, no. 1, pp. 27–38, 2005.

[73] A. Gil, O. Reinoso, O. M. Mozos, C. Stachniss, and W. Burgard, "Improving data association in vision-based slam," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2006, pp. 2076–2081.

[74] M. Calonder, V. Lepetit, M. Ozuysal, T. Trzcinski, C. Strecha, and P. Fua, "Brief: Computing a local binary descriptor very fast," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 34, no. 7, pp. 1281–1298, 2011.

[75] W. Churchill and P. Newman, "Experience-based navigation for long-term localisation," *Intl. J. of Robotics Research*, vol. 32, no. 14, pp. 1645–1661, 2013.

[76] M. Cummins and P. Newman, "Appearance-only slam at large scale with fab-map 2.0," *Intl. J. of Robotics Research*, vol. 30, no. 9, pp. 1100–1123, 2011.

[77] D. Filliat, "A visual bag of words method for interactive qualitative localization and mapping," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2007, pp. 3921–3926.

[78] C. Cadena, D. Gálvez-López, J. D. Tardós, and J. Neira, "Robust place recognition with stereo sequences," *IEEE Trans. Robotics*, vol. 28, no. 4, pp. 871–885, 2012.

[79] P. Furgale and T. D. Barfoot, "Visual teach and repeat for long-range rover autonomy," *J. of Field Robotics*, vol. 27, no. 5, pp. 534–560, 2010.

[80] C. Valgren and A. J. Lilienthal, "Sift, surf & seasons: Appearance-based long-term localization in outdoor environments," *Robotics and Autonomous Systems*, vol. 58, no. 2, pp. 149–156, 2010.

[81] P. Corke, R. Paul, W. Churchill, and P. Newman, "Dealing with shadows: Capturing intrinsic scene appearance for image-based outdoor localisation," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2013, pp. 2085–2092.

[82] C. McManus, W. Churchill, W. Maddern, A. D. Stewart, and P. Newman, "Shady dealings: Robust, long-term visual localisation using illumination invariance," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 901–906.

[83] A. Ranganathan, S. Matsumoto, and D. Ilstrup, "Towards illumination invariance for visual localization," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*. IEEE, 2013, pp. 3791–3798.

[84] N. Carlevaris-Bianco and R. M. Eustice, "Learning visual feature descriptors for dynamic lighting conditions," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2014, pp. 2769–2776.

[85] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah, "Signature verification using a" siamese" time delay neural network," *Adv. in Neural Inf. Proc. Systems*, vol. 6, 1993.

[86] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, vol. 1. Ieee, 2005, pp. 886–893.

[87] A. Oliva and A. Torralba, "Building the gist of a scene: The role of global image features in recognition," *Progress in brain research*, vol. 155, pp. 23–36, 2006.

[88] A. C. Murillo and J. Kosecka, "Experiments in place recognition using gist panoramas," in *Intl. Conf. on Computer Vision (ICCV)*. IEEE, 2009, pp. 2196–2203.

[89] C. Azzi, D. C. Asmar, A. H. Fakih, and J. S. Zelek, "Filtering 3d keypoints using gist for accurate image-based localization." in *British Machine Vision Conf. (BMVC)*, 2016.

[90] J. Yue-Hei Ng, F. Yang, and L. S. Davis, "Exploiting local features from deep networks for image retrieval," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, 2015, pp. 53–61.

[91] S. Garg, N. Suenderhauf, and M. Milford, "Don't look back: Robustifying place categorization for viewpoint-and condition-invariant place recognition," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 3645–3652.

[92] T. Naseer, G. L. Oliveira, T. Brox, and W. Burgard, "Semantics-aware visual localization under challenging perceptual conditions," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 2614–2620.

[93] N. Sünderhauf, S. Shirazi, F. Dayoub, B. Upcroft, and M. Milford, "On the performance of convnet features for place recognition," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 4297–4304.

[94] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, 2015, pp. 1–9.

[95] H. Jégou and O. Chum, "Negative evidences and co-occurences in image retrieval: The benefit of pca and whitening," in *Eur. Conf. on Computer Vision (ECCV)*. Springer, 2012, pp. 774–787.

[96] C. L. Zitnick and P. Dollár, "Edge boxes: Locating object proposals from edges," in *Eur. Conf. on Computer Vision (ECCV)*. Springer, 2014, pp. 391–405.

[97] M.-M. Cheng, Z. Zhang, W.-Y. Lin, and P. Torr, "Bing: Binarized normed gradients for objectness estimation at 300fps," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, 2014, pp. 3286–3293.

[98] N. Sünderhauf, S. Shirazi, A. Jacobson, F. Dayoub, E. Pepperell, B. Upcroft, and M. Milford, "Place recognition with convnet landmarks: Viewpoint-robust, condition-robust, training-free," *Robotics: Science and Systems (RSS)*, pp. 1–10, 2015.

[99] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Adv. in Neural Inf. Proc. Systems*, vol. 25, 2012.

[100] J. Sivic and A. Zisserman, "Video google: A text retrieval approach to object matching in videos," in *Intl. Conf. on Computer Vision (ICCV)*, vol. 3. IEEE Computer Society, 2003, pp. 1470–1470.

[101] A. Khaliq, S. Ehsan, Z. Chen, M. Milford, and K. McDonald-Maier, "A holistic visual place recognition approach using lightweight cnns for significant viewpoint and appearance changes," *IEEE Trans. Robotics*, vol. 36, no. 2, pp. 561–569, 2019.

[102] M. Kaya and H. Ş. Bilge, "Deep metric learning: A survey," *Symmetry*, vol. 11, no. 9, p. 1066, 2019.

[103] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, vol. 2. IEEE, 2006, pp. 1735–1742.

[104] E. Hoffer and N. Ailon, "Deep metric learning using triplet network," in *International workshop on similarity-based pattern recognition*. Springer, 2015, pp. 84–92.

[105] W. Chen, X. Chen, J. Zhang, and K. Huang, "Beyond triplet loss: a deep quadruplet network for person re-identification," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, 2017, pp. 403–412.

[106] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*. Ieee, 2009, pp. 248–255.

[107] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, "Object retrieval with large vocabularies and fast spatial matching," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*. IEEE, 2007, pp. 1–8.

[108] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, "Lost in quantization: Improving particular object retrieval in large scale image databases," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*. IEEE, 2008, pp. 1–8.

[109] H. Jegou, M. Douze, and C. Schmid, "Hamming embedding and weak geometric consistency for large scale image search," in *Eur. Conf. on Computer Vision (ECCV)*. Springer, 2008, pp. 304–317.

[110] J. Yu, C. Zhu, J. Zhang, Q. Huang, and D. Tao, "Spatial pyramid-enhanced netvlad with weighted triplet loss for place recognition," *IEEE transactions on neural networks and learning systems*, vol. 31, no. 2, pp. 661–674, 2019.

[111] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[112] K. Qiu, Y. Ai, B. Tian, B. Wang, and D. Cao, "Siamese-resnet: implementing loop closure detection based on siamese network," in *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2018, pp. 716–721.

[113] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, vol. 1. IEEE, 2005, pp. 539–546.

[114] D. Olid, J. M. Fácil, and J. Civera, "Single-view place recognition under seasonal changes," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2018.

[115] W. Maddern, G. Pascoe, C. Linegar, and P. Newman, "1 year, 1000 km: The oxford robotcar dataset," *Intl. J. of Robotics Research*, vol. 36, no. 1, pp. 3–15, 2017.

[116] M. Cummins and P. Newman, "FAB-MAP: Probabilistic localization and mapping in the space of appearance," *Intl. J. of Robotics Research*, vol. 27, no. 6, pp. 647–665, 2008.

[117] L. Wu and Y. Wu, "Deep supervised hashing with similar hierarchy for place recognition," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 3781–3786.

[118] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv*, 2017.

[119] D. Blalock, J. J. Gonzalez Ortiz, J. Frankle, and J. Guttag, "What is the state of neural network pruning?" *Proceedings of machine learning and systems*, vol. 2, pp. 129–146, 2020.

[120] S. Hausler, A. Jacobson, and M. Milford, "Filter early, match late: Improving network-based visual place recognition," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 3268–3275.

[121] Y. Latif, R. Garg, M. Milford, and I. Reid, "Addressing challenging place recognition tasks using generative adversarial networks," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 2349–2355.

[122] N. Merrill and G. Huang, "Lightweight unsupervised deep loop closure," *arXiv*, 2018.

[123] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[124] L. Tang, Y. Wang, Q. Luo, X. Ding, and R. Xiong, "Adversarial feature disentanglement for place recognition across changing appearance," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 1301–1307.

[125] M. Milford and G. Wyeth, "Persistent navigation and mapping using a biologically inspired slam system," *Intl. J. of Robotics Research*, vol. 29, no. 9, pp. 1131–1153, 2010.

[126] M. Warren, M. Greeff, B. Patel, J. Collier, A. P. Schoellig, and T. D. Barfoot, "There's no place like home: Visual teach and repeat for emergency return of multirotor uavs during gps failure," *IEEE Robotics & Automation Letters*, vol. 4, no. 1, pp. 161–168, 2018.

[127] M. Paton, K. MacTavish, C. J. Ostafew, and T. D. Barfoot, "It's not easy seeing green: Lighting-resistant stereo visual teach & repeat using color-constant images," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 1519–1526.

[128] M. Paton, F. Pomerleau, and T. D. Barfoot, "Eyes in the back of your head: Robust visual teach & repeat using multiple stereo cameras," in *Conference on Computer and Robot Vision (CRV)*. IEEE, 2015, pp. 46–53.

[129] T. Krajník, P. Cristóforis, K. Kusumam, P. Neubert, and T. Duckett, "Image features for visual teach-and-repeat navigation in changing environments," *Journal of Robotics and Autonomous Systems*, vol. 88, pp. 127–141, 2017.

[130] J. Dequaire, C. H. Tong, W. Churchill, and I. Posner, "Off the beaten track: Predicting localisation performance in visual teach and repeat," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 795–800.

[131] O. Vysotska and C. Stachniss, "Lazy data association for image sequences matching under substantial appearance changes," *IEEE Robotics & Automation Letters*, vol. 1, no. 1, pp. 213–220, 2015.

[132] B. Talbot, S. Garg, and M. Milford, "Openseqslam2. 0: an open source toolbox for visual place recognition under changing conditions," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 7758–7765.

[133] N. Sünderhauf, P. Neubert, and P. Protzel, "Are we there yet? challenging seqslam on a 3000 km journey across all four seasons," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2013.

[134] S. M. Siam and H. Zhang, "Fast-seqslam: A fast appearance based place recognition algorithm," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 5702–5708.

[135] D. Bai, C. Wang, B. Zhang, X. Yi, and X. Yang, "Sequence searching with cnn features for robust and fast visual place recognition," *Computers & Graphics*, vol. 70, pp. 270–280, 2018.

[136] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "Overfeat: Integrated recognition, localization and detection using convolutional networks," in *Intl. Conf. on Learning Representations (ICLR)*, 2014.

[137] S. Thrun, "Probabilistic robotics," *Communications of the ACM*, vol. 45, no. 3, pp. 52–57, 2002.

[138] S. Thrun, "Particle filters in robotics." in *Conference on Uncertainty in Artificial Intelligence (UAI)*, vol. 2. Citeseer, 2002, pp. 511–518.

[139] A. J. Haug, "A tutorial on bayesian estimation and tracking techniques applicable to nonlinear and non-gaussian processes," *Technical Report*, 2005.

[140] R. E. Kalman, "A new approach to linear filtering and prediction problems," 1960.

[141] P. S. Maybeck, *Stochastic models, estimation, and control*. Academic press, 1982.

[142] M. Xu, N. Snderhauf, and M. Milford, "Probabilistic visual place recognition for hierarchical localization," *IEEE Robotics & Automation Letters*, vol. 6, no. 2, pp. 311–318, 2020.

[143] G. Kitagawa, "A monte carlo filtering and smoothing method for non-gaussian nonlinear state space models," in *Proceedings of the 2nd US-Japan joint seminar on statistical time series analysis*, vol. 2, 1993, pp. 110–131.

[144] N. J. Gordon, D. J. Salmond, and A. F. Smith, "Novel approach to nonlinear/non-gaussian bayesian state estimation," in *IEE proceedings F (radar and signal processing)*, vol. 140, no. 2. IET, 1993, pp. 107–113.

[145] P. Karkus, D. Hsu, and W. S. Lee, "Particle filter networks with application to visual localization," in *Conference on robot learning (CoRL)*. PMLR, 2018, pp. 169–178.

[146] M. Montemerlo, S. Thrun, D. Koller, B. Wegbreit *et al.*, "Fastslam 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges," in *International Joint Conference on Artificial Intelligence (IJCAI)*, vol. 3, 2003, pp. 1151–1156.

[147] M. A. Brubaker, A. Geiger, and R. Urtasun, "Map-based probabilistic visual self-localization," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 38, no. 4, pp. 652–665, 2015.

[148] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *Intl. J. of Robotics Research*, 2013.

[149] M. Dubois, H. Guillaume, F. Emmanuelle, and P. Tarroux, "Visual place recognition using bayesian filtering with markov chains," in *European Symposium on Artificial Neural Networks (ESANN)*, 2012.

[150] E. Pepperell, P. Corke, and M. Milford, "Routed roads: Probabilistic vision-based place recognition for changing conditions, split streets and varied viewpoints," *Intl. J. of Robotics Research*, vol. 35, no. 9, pp. 1057–1179, 2016.

[151] K. Hong, S. Kim, J. Park, and H. Bang, "Particle filter approach to vision-based navigation with aerial image segmentation," *Journal of Aerospace Information Systems*, vol. 18, no. 12, pp. 964–972, 2021.

[152] J. Vogel and B. Schiele, "Semantic modeling of natural scenes for content-based image retrieval," *Intl. J. of Computer Vision*, vol. 72, no. 2, pp. 133–157, 2007.

[153] J. Hou, H. Zeng, J. Zhu, J. Hou, J. Chen, and K.-K. Ma, "Deep quadruplet appearance learning for vehicle re-identification," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 9, pp. 8512–8522, 2019.

[154] A. Pal, C. Nieto-Granda, and H. I. Christensen, "DEDUCE: Diverse scEne Detection methods in Unseen Challenging Environments," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 4198–4204.

[155] K. Muhammad, J. Ahmad, and S. W. Baik, "Early fire detection using convolutional neural networks during surveillance for effective disaster management," *Neurocomputing*, vol. 288, pp. 30–42, 2018.

[156] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba, "Places: A 10 million image database for scene recognition," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 40, no. 6, pp. 1452–1464, 2017.

[157] L. Herranz, S. Jiang, and X. Li, "Scene recognition with cnns: objects, scales and dataset bias," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, 2016, pp. 571–579.

[158] M. Cimpoi, S. Maji, and A. Vedaldi, "Deep filter banks for texture recognition and segmentation," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, 2015, pp. 3828–3836.

[159] Y. Li, M. Dixit, and N. Vasconcelos, "Deep scene image classification with the mfafvnet," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, 2017, pp. 5746–5754.

[160] X. Cheng, J. Lu, J. Feng, B. Yuan, and J. Zhou, "Scene recognition with objectness," *Pattern Recognition*, vol. 74, pp. 474–487, 2018.

[161] Z. Xiong, Y. Yuan, and Q. Wang, "Rgb-d scene recognition via spatial-related multi-modal feature learning," *IEEE Access*, vol. 7, pp. 106 739–106 747, 2019.

[162] P. Tang, H. Wang, and S. Kwong, "G-ms2f: Googlenet based multi-stage feature fusion of deep cnn for scene recognition," *Neurocomputing*, vol. 225, pp. 188–197, 2017.

[163] S. Yang and D. Ramanan, "Multi-scale recognition with dag-cnns," in *Intl. Conf. on Computer Vision (ICCV)*, 2015, pp. 1215–1223.

[164] Y. Dong, S. Gao, K. Tao, J. Liu, and H. Wang, "Performance evaluation of early and late fusion methods for generic semantics indexing," *Pattern Analysis and Applications*, vol. 17, no. 1, pp. 37–50, 2014.

[165] H. Seong, J. Hyun, and E. Kim, "Fosnet: An end-to-end trainable deep neural network for scene recognition," *IEEE Access*, vol. 8, pp. 82 066–82 077, 2020.

[166] N. Sun, W. Li, J. Liu, G. Han, and C. Wu, "Fusing object semantics and deep appearance features for scene recognition," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 6, pp. 1715–1728, 2018.

[167] E. Yurtsever, J. Lambert, A. Carballo, and K. Takeda, "A survey of autonomous driving: Common practices and emerging technologies," *IEEE access*, vol. 8, pp. 58 443–58 469, 2020.

[168] J.-R. Xue, J.-W. Fang, and P. Zhang, "A survey of scene understanding by event reasoning in autonomous driving," *International Journal of Automation and Computing*, vol. 15, no. 3, pp. 249–266, 2018.

[169] N. P. Jouppi, C. Young, N. Patil, D. Patterson, G. Agrawal, R. Bajwa, S. Bates, S. Bhatia, N. Boden, A. Borchers *et al.*, "In-datacenter performance analysis of a tensor processing unit," in *Proceedings of the 44th annual international symposium on computer architecture*, 2017, pp. 1–12.

[170] N. Jouppi, C. Young, N. Patil, and D. Patterson, "Motivation for and evaluation of the first tensor processing unit," *IEEE Micro*, vol. 38, no. 3, pp. 10–19, 2018.

[171] J. Zhu, L. Wang, H. Liu, S. Tian, Q. Deng, and J. Li, "An efficient task assignment framework to accelerate dpu-based convolutional neural network inference on fpgas," *IEEE Access*, vol. 8, pp. 83 224–83 237, 2020.

[172] J. O. Neill, "An overview of neural network compression," *arXiv*, 2020.

[173] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, 2017, pp. 4700–4708.

[174] J. Garland and D. Gregg, "Low complexity multiply accumulate unit for weight-sharing convolutional neural networks," *IEEE Computer Architecture Letters*, vol. 16, no. 2, pp. 132–135, 2017.

[175] M. Hagiwara, "Removal of hidden units and weights for back propagation networks," in *Proceedings of 1993 International Conference on Neural Networks (IJCNN-93-Nagoya, Japan)*, vol. 1. IEEE, 1993, pp. 351–354.

[176] R. Rigamonti, A. Sironi, V. Lepetit, and P. Fua, "Learning separable filters," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, 2013, pp. 2754–2761.

[177] M. Jaderberg, A. Vedaldi, and A. Zisserman, "Speeding up convolutional neural networks with low rank expansions," in *British Machine Vision Conf. (BMVC)*, 2014.

[178] X. Yu, T. Liu, X. Wang, and D. Tao, "On compressing deep models by low rank and sparse decomposition," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, 2017, pp. 7370–7379.

[179] G. Chen, W. Choi, X. Yu, T. Han, and M. Chandraker, "Learning efficient object detection models with knowledge distillation," *Adv. in Neural Inf. Proc. Systems*, vol. 30, 2017.

[180] A. Kouris, S. I. Venieris, and C.-S. Bouganis, "Cascadeˆ cnn: Pushing the performance limits of quantisation in convolutional neural networks," in *International Conference on Field Programmable Logic and Applications (FPL)*. IEEE, 2018, pp. 155–1557.

[181] C. Geng, S.-j. Huang, and S. Chen, "Recent advances in open set recognition: A survey," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 43, no. 10, pp. 3614–3631, 2020.

[182] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," *Technical Report*, 2009.

[183] L. Deng, "The mnist database of handwritten digit images for machine learning research," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, 2012.

[184] D. Miller, N. Sunderhauf, M. Milford, and F. Dayoub, "Class anchor clustering: A loss for distance-based open set recognition," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2021, pp. 3570–3578.

[185] H. Oh Song, Y. Xiang, S. Jegelka, and S. Savarese, "Deep metric learning via lifted structured feature embedding," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, 2016, pp. 4004–4012.

[186] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Intl. Conf. on Computer Vision (ICCV)*, 2017, pp. 2980–2988.

[187] X. Zhang, Z. Fang, Y. Wen, Z. Li, and Y. Qiao, "Range loss for deep face recognition with long-tailed training data," in *Intl. Conf. on Computer Vision (ICCV)*, 2017, pp. 5409–5418.

[188] B. J. Meyer and T. Drummond, "The importance of metric learning for robotic vision: Open set recognition and active learning," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 2924–2931.

[189] S. Liang, Y. Li, and R. Srikant, "Enhancing the reliability of out-of-distribution image detection in neural networks," in *Intl. Conf. on Learning Representations (ICLR)*, 2018.

[190] P. Schlachter, Y. Liao, and B. Yang, "Open-set recognition using intra-class splitting," in *European signal processing conference (EUSIPCO)*. IEEE, 2019, pp. 1–5.

[191] S. D. Zongyuan Ge and R. Garnavi, "Generative openmax for multi-class open set classification," in *British Machine Vision Conf. (BMVC)*. BMVA Press, September 2017, pp. 42.1–42.12.

[192] L. Neal, M. Olson, X. Fern, W.-K. Wong, and F. Li, "Open set learning with counterfactual images," in *Eur. Conf. on Computer Vision (ECCV)*, 2018, pp. 613–628.

[193] S. Vaze, K. Han, A. Vedaldi, and A. Zisserman, "Open-set recognition: A good closed-set classifier is all you need," in *Intl. Conf. on Learning Representations (ICLR)*, 2021.

[194] Z. Liu, Z. Miao, X. Zhan, J. Wang, B. Gong, and S. X. Yu, "Large-scale long-tailed recognition in an open world," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, 2019, pp. 2537–2546.

[195] J. Cai, Y. Wang, H.-M. Hsu, J.-N. Hwang, K. Magrane, and C. Rose, "Luna: Localizing unfamiliarity near acquaintance for open-set long-tailed recognition," *Artificial Intelligence*, 2022.

[196] A. Kendall and Y. Gal, "What uncertainties do we need in bayesian deep learning for computer vision?" *Adv. in Neural Inf. Proc. Systems*, vol. 30, 2017.

[197] M. Mundt, I. Pliushch, S. Majumder, and V. Ramesh, "Open set recognition through deep neural network uncertainty: Does out-of-distribution detection require generative classifiers?" in *Intl. Conf. on Computer Vision (ICCV)*, 2019, pp. 0–0.

[198] X. Ye and J. Zhao, "Open set deep learning with a bayesian nonparametric generative model," in *ACM International Conference on Multimedia*, 2019, pp. 2133–2141.

[199] F. Warburg, M. Jørgensen, J. Civera, and S. Hauberg, "Bayesian triplet loss: Uncertainty quantification in image retrieval," in *Intl. Conf. on Computer Vision (ICCV)*, 2021, pp. 12 158–12 168.

[200] E. Romera, L. M. Bergasa, and R. Arroyo, "Need data for driver behaviour analysis? presenting the public uah-driveset," in *International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2016, pp. 387–392.

[201] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition*, 2015, pp. 815–823.

[202] B. Yu, T. Liu, M. Gong, C. Ding, and D. Tao, "Correcting the triplet selection bias for triplet loss," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 71–87.

[203] K. Sohn, W. Shang, X. Yu, and M. Chandraker, "Unsupervised domain adaptation for distance metric learning," in *Intl. Conf. on Learning Representations (ICLR)*, 2018.

[204] J. Johnson, M. Douze, and H. Jégou, "Billion-scale similarity search with gpus," *IEEE Transactions on Big Data*, vol. 7, no. 3, pp. 535–547, 2019.

[205] J. Krause, M. Stark, J. Deng, and L. Fei-Fei, "3d object representations for fine-grained categorization," in *Intl. Conf. on Computer Vision (ICCV)*, 2013, pp. 554–561.

[206] M. Gridseth and T. D. Barfoot, "Keeping an eye on things: Deep learned features for long-term visual localization," *IEEE Robotics & Automation Letters*, vol. 7, no. 2, pp. 1016–1023, 2021.

[207] Z. Rozsypálek, G. Broughton, P. Linder, T. Rouček, J. Blaha, L. Mentzl, K. Kusumam, and T. Krajník, "Contrastive learning for image registration in visual teach and repeat navigation," *Sensors*, vol. 22, no. 8, p. 2975, 2022.

[208] S. Garg, M. Vankadari, and M. Milford, "Seqmatchnet: Contrastive learning with sequence matching for place recognition & relocalization," in *Conference on Robot Learning*. PMLR, 2022, pp. 429–443.

[209] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, and D. Krishnan, "Supervised contrastive learning," *Adv. in Neural Inf. Proc. Systems*, vol. 33, pp. 18 661–18 673, 2020.

[210] A. Jaiswal, A. R. Babu, M. Z. Zadeh, D. Banerjee, and F. Makedon, "A survey on contrastive self-supervised learning," *Technologies*, vol. 9, no. 1, p. 2, 2020.

[211] W. J. Scheirer, A. Rocha, R. J. Micheals, and T. E. Boult, "Meta-recognition: The theory and practice of recognition score analysis," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 33, no. 8, pp. 1689–1695, 2011.

[212] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of Machine Learning Research*, vol. 9, no. 11, 2008.

[213] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," in *Intl. Conf. on Computer Vision (ICCV)*, 2017, pp. 618–626.

[214] A. Tüzkö, C. Herrmann, D. Manger, and J. Beyerer, "Open set logo detection and retrieval," *arXiv*, 2017.

[215] M. Bastan, H.-Y. Wu, T. Cao, B. Kota, and M. Tek, "Large scale open-set deep logo detection," *arXiv*, 2019.

[216] S. Kiranyaz, O. Avci, O. Abdeljaber, T. Ince, M. Gabbouj, and D. J. Inman, "1d convolutional neural networks and applications: A survey," *Mechanical systems and signal processing*, vol. 151, p. 107398, 2021.

[217] H. Wang, H. Shi, K. Lin, C. Qin, L. Zhao, Y. Huang, and C. Liu, "A high-precision arrhythmia classification method based on dual fully connected neural network," *Biomedical Signal Processing and Control*, vol. 58, p. 101874, 2020.

[218] A. A. Neacsu, G. Cioroiu, A. Radoi, and C. Burileanu, "Automatic emg-based hand gesture recognition system using time-domain descriptors and fully-connected neural networks," in *Int. Conf. on Telecommunications and Signal Proc. (TSP)*. IEEE, 2019, pp. 232–235.

[219] D. Li, J. Zhang, Q. Zhang, and X. Wei, "Classification of ecg signals based on 1d convolution neural network," in *IEEE Int. Conf. on e-Health Networking, Applications and Services*. IEEE, 2017, pp. 1–6.

[220] T. Ince, S. Kiranyaz, L. Eren, M. Askar, and M. Gabbouj, "Real-time motor fault detection by 1-d convolutional neural networks," *IEEE Transactions on Industrial Electronics*, vol. 63, no. 11, pp. 7067–7075, 2016.

[221] S. Kiranyaz, A. Gastli, L. Ben-Brahim, N. Al-Emadi, and M. Gabbouj, "Real-time fault detection and identification for mmc using 1-d convolutional neural networks," *IEEE Transactions on Industrial Electronics*, vol. 66, no. 11, pp. 8760–8771, 2018.

[222] R. Sánchez-Reolid, F. L. de la Rosa, M. T. López, and A. Fernández-Caballero, "One-dimensional convolutional neural networks for low/high arousal classification from electrodermal activity," *Biomedical Signal Processing and Control*, vol. 71, p. 103203, 2022.

[223] T. J. O'Shea, T. Roy, and T. C. Clancy, "Over-the-air deep learning based radio signal classification," *IEEE Journal of Sel. Topics in Signal Proc.*, vol. 12, no. 1, pp. 168–179, 2018.

[224] H. Ismail Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, "Deep learning for time series classification: a review," *Data mining and knowledge discovery*, vol. 33, no. 4, pp. 917–963, 2019.

[225] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Intl. Conf. on Learning Representations (ICLR)*. PMLR, 2020, pp. 1597–1607.

[226] A. J. Glover, W. P. Maddern, M. J. Milford, and G. F. Wyeth, "Fab-map+ ratslam: Appearance-based slam for multiple times of day," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*. IEEE, 2010, pp. 3507–3512.

[227] H. Wu, B. Xiao, N. Codella, M. Liu, X. Dai, L. Yuan, and L. Zhang, "Cvt: Introducing convolutions to vision transformers," in *Intl. Conf. on Computer Vision (ICCV)*, 2021, pp. 22–31.