Deep Visual Geo-localization Benchmark

Gabriele Berton Politecnico di Torino
Riccardo Mereu Politecnico di Torino
Gabriele Trivigno Politecnico di Torino
Carlo Masone CINI
Gabriela Csurka NAVER LABS Europe
Torsten Sattler CIIRC, Czech Technical University in Prague
Barbara Caputo Politecnico di Torino

Abstract

In this paper, we propose a new open-source benchmarking framework for Visual Geo-localization (VG) that allows to build, train, and test a wide range of commonly used architectures, with the flexibility to change individual components of a geo-localization pipeline. The purpose of this framework is twofold: i) gaining insights into how different components and design choices in a VG pipeline impact the final results, both in terms of performance (recall@N metric) and system requirements (such as execution time and memory consumption); ii) establish a systematic evaluation protocol for comparing different methods. Using the proposed framework, we perform a large suite of experiments which provide criteria for choosing backbone, aggregation and negative mining depending on the use-case and requirements. We also assess the impact of engineering techniques like pre/post-processing, data augmentation and image resizing, showing that better performance can be obtained through somewhat simple procedures: for example, downscaling the images' resolution to 80% can lead to similar results with a 36% savings in extraction time and dataset storage requirement. Code and trained models are available at dataset storage requirement. https://deep-vg-bench.herokuapp.com/.

1. Introduction

The task of coarsely estimating the place where a photo was taken based on a set of previously visited locations is called *Visual (Image) Geo-localization* (VG) [35, 40, 81] or *Visual Place Recognition* (VPR) [20, 42] and it is addressed using image matching and retrieval methods on a database of images of known locations. We are witnessing a rapid growth of this field of research, as demonstrated by the increasing number of publications [2, 10, 14, 21–23, 27, 29, 34, 35, 40, 42, 44, 55, 58, 69, 70, 73–76, 78, 82], but this expansion is accompanied by two major limitations:

	Vanilla	Resize (80%)	Data augm. (brightness = 2)	Pred. refinement (*nearest crop*)	PCA (2048)	CRN [35]
R@1	63.4	64.3	68.6	67.0	56.6	68.8

Table 1. Example of how results can be influenced by little train or test time changes to the VG pipeline. Recall@1 for a ResNet-18 with NetVLAD trained on Pitts30k and tested on Tokyo24/7. Results are thoroughly discussed in later sections.

i) **A focus on single metric optimization**, as it is common practice to compare results solely based on the recall on chosen datasets and ignoring other factors such as execution time, hardware requirements, and scalability. All these aspects are important constraints in the design of a real-world VG system. For instance, one might gladly accept a 5% drop in accuracy if this leads to a 90% decrease of descriptors size as the resulting reduction in memory requirements enables a better scalability. Similarly, computational time and descriptor dimensionality are crucial constraints in real-time applications, given a target hardware platform.
ii) **A lack of a standardized framework** to train and test VG models. It is common practice to perform direct comparisons among off-the-shelf methods that use different setups (*e.g.*, data augmentation, initialization, training dataset, *etc.*) [35, 64, 80], which can hide the improvement (or lack thereof) obtained by algorithmic changes and it does not allow to pinpoint the impact of each individual component. Table 1 shows how some simple engineering choices can have big effects on the recall metric.

Although previous benchmarks for VPR [80] and the related task of Visual Localization [56, 62] offer interesting insights, they do not address the aforementioned issues. For these reasons, we propose a new open-source benchmark that provides researchers with an all-inclusive tool to build, train, and test a wide range of commonly used VG architectures, offering the flexibility to change each component of a geo-localization pipeline. This allows to rigorously examine how each element of the system influences the final results while providing information computed on-the-fly regarding the number of parameters, FLOPs, descriptors dimensionality, *etc*.

Using our framework, we run numerous experiments aiming to understand which components are the most suitable for a real-world application, and derive good practices depending on the target dataset and one's hardware availability. For example, we find that ResNet-50 [28] provides a good trade-off between accuracy, FLOPs and model size, and that Visual Transformers can successfully replace the CNN backbones and achieve better geo-localization performances when trained on larger datasets. Furthermore, we observed that partial negative mining and reduced resolution yield important decrease in computations without significantly compromising the performance, or even yielding gains in some cases.

The benchmark's software and models are hosted at https://deep-vg-bench.herokuapp.com/.

2. Related Work

Representation learning for visual retrieval and localization. Visual Geo-localization (VG), Visual Localization (VL), and Landmark Retrieval (LR) are three well-known Computer Vision tasks that try to establish a mapping between an image and a spatial location, albeit with some nuances. In VG the goal is to find the geographical location of a given query image and the predicted coordinates are considered correct if they are roughly close to ground truth position [2, 10, 23, 35, 39, 40, 74, 75]. VL focuses on precisely estimating the 6 DoF camera pose of a query image within a known scene. VG methods can be used as a part of a VL pipeline, combined with other processing stages that reduce the differences when used in a VL task. Therefore the evaluation papers on VL [56,62,71] might not be indicative of VG performance, justifying a separate benchmark on the latter. LR is a particular case of Image Retrieval (IR) in which queries contain some landmark, and the goal is to identify all database instances depicting the same landmark, regardless of their visually overlap with the query photo. Since VG is usually addressed as a retrieval problem where the query position is estimated using the GPS tags of the top retrieved image, several methods originally proposed for LR (or IR in general) have carried over to VG. LR datasets, both on a city-scale (Oxford and Paris Buildings [53, 54]) and on a global scale (Google Landmarks [47,77]), consists of a discrete set of landmarks, whereas VG datasets usually cover a continuous geographical area.

IR [3,16,67] is traditionally performed via nearest neighbors search using fixed-size image representations [15, 30, 31,33,52,63,69] obtained from the aggregation of highly informative local [1, 8, 41] or global [48, 79] features. Convolutional neural networks (CNNs) have become the de-facto standard to extract the features for IR, using various methods to concatenate them [7, 59] or pool them [4, 5, 68] to create image descriptors. Among the deep learning representation methods, one that has proven very effective for VG is NetVLAD [2], a differentiable implementation of VLAD [33] trained end-to-end with the CNN backbone directly for place recognition. The layer has since been used in numerous works [10,19,22,23,27,40,74,75]. One downside of NetVLAD is that it outputs high-dimensional descriptors, leading to steep memory requirements for VG systems. This problem has inspired research on more compact descriptors, either using dimensionality reduction techniques [7, 11, 24, 49, 57, 83] or replacing NetVLAD with lighter pooling layers, such as GeM [58] and R-MAC [25]. It has also been shown that attention modules can be used to focus feature extraction and aggregation towards the most salient parts of the scene for the geo-localization task [11, 35, 39, 46]. The Contextual Reweighting Network (CRN) [35] is a variation of NetVLAD that adds a contextual modulation to produce a weighting mask based on semi-global context. Visual Transformers based on self-attention such as ViT [17] and DeiT [72] have also been used in IR [12,18], but not yet in VG. All these architectures used in VG to learn image representations are trained with metric embedding objectives commonly used in learning-to-rank problems, such as the contrastive loss [49, 57, 58], the triplet loss [2, 25, 35] and the SARE loss [40].

Our benchmark analyzes how the combination of popular backbone networks, pooling strategies, data augmentation, and engineering choices impacts geo-localization performance and other aspects, such as memory and computational requirements.

Benchmarking. The only available benchmark focused specifically on VG/VPR is VPR-Bench [80]. In contrast to our work, [80] (as well as [56] for VL) directly compares off-the-shelf models because it is mainly concerned with the performance of VG in practical settings, where one would likely prefer using a pre-trained model rather than having to fine-tune or train it. On the other hand, we are more interested in measuring the impact of algorithmic changes, which requires performing comparisons where all other factors are the same. To this end, we propose a modular framework that allows a fair evaluation of each element of a VG system under identical conditions, ensuring clarity and reliability of the results.

While [80] also provides insights on descriptors dimensionality and retrieval time, we focus on more general hardware-agnostic statistics, such as FLOPs and model size (Sec. 4.1), training complexity (Sec. 4.4), storage requirements (Sec. 4.6).

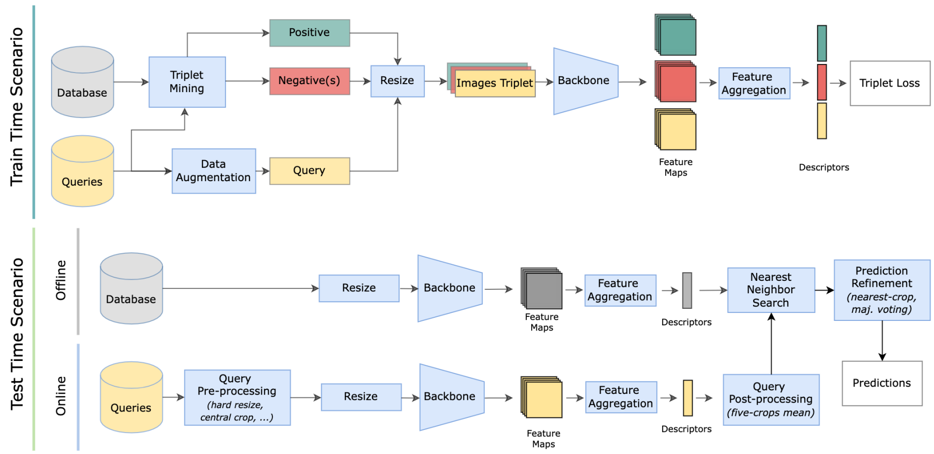


Figure 1. **Diagram of a visual geo-localization system.** Throughout this work, we rigorously and fairly analyze each component of a visual geo-localization system (the light blue blocks) comparing a variety of different implementations, both for train and test time.

3. Methodology

This section describes the VG pipeline used in our benchmark (*cf*. Fig. 1) and our experimental setup.

3.1. Visual Geo-localization System

The VG task is commonly tackled using an image retrieval pipeline: given a new photo (*query*) to be geo-localized, its location is estimated by matching it to a database of geo-tagged images. A VG system is thus an algorithm that first extracts descriptors for the database images (offline) and for the query photo (online), then it applies a nearest neighbors search in the descriptor space. The orange blocks in Fig. 1 show that a VG system is built through several design choices, including network architectures, negative mining methods, and engineering aspects such as image sizes and data augmentation. All of these choices impact the behavior of the system, both in terms of performance and required resources. We propose a new benchmark to systematically investigate the impact of the components of VG systems, using the modular architecture shown in Fig. 1 as a canvas to reproduce most VG methods based on CNN backbones and to develop new models based on Visual Transformers.

This abstract model contains several components that can be modified, both during training and test time: the **backbone** (Sec. 4.1); **feature aggregation** (Sec. 4.2); **mining** training examples (Sec. 4.4); **image resizing** (Sec. 4.6); **data augmentation** (Sec. 4.5). We conduct a series of tests focused individually on each of these elements, to systematically show each component's influence. Due to limited space, we only summarize here the results of some experiments, while detailed results and additional experiments on **pre/post-processing methods and predictions refinement**, **effect of pre-training** and many other aspects are provided in the supp. material.

The code of the benchmark follows the modular structure shown in Fig. 1, where each component can be modified. We further provide scripts to download and format a number of datasets, and to train and test the models making easy to perform a large number of experiments while ensuring consistency and reproducibility of results. Our codebase allows to easily reproduce the architectures used in a wide range of works [2, 25, 35, 40, 58, 61, 68, 75] and commonly used training protocols [2, 40, 75]. More details on the software are provided in the supp. material.

3.2. Datasets

We use six highly heterogeneous datasets (see Tab. 2 and maps in the supp. material), which together cover a variety of real-world scenarios: different scales, degree of inter-image variability, different camera types. For training, we use Pitts30k [2] and Mapillary Street-Level Sequences (MSLS) [75] datasets, as they provide a small and large amount of images, respectively. While Pitts30k is very homogeneous, *i.e.* all images share the same resolution, weather conditions and camera, MSLS represents a wide range of conditions from very diverse cities. Regarding MSLS, given the lack of labels for the test set, we follow [27] and report validation recalls computed on the validation set. To assess inter-dataset robustness, we also test all models on four other datasets: Tokyo 24/7 [69], Revisited San Francisco (R-SF) [13, 38, 71], Eynsham [16] and St Lucia [45]. Further details on these datasets, such as their geographical coverage, are included in the supp. material.

3.3. Benchmark Protocol

In all experiments, unless otherwise specified, we use the metric of recall@N (R@N) measuring the percentage of queries for which one of the top-N retrieved images was taken within a certain distance of the query location. We mostly focus on R@1 and, following common practice in

	# train/val datab./queries	# test datab./queries	Dataset size	Database type	Database img. size	Queries type
Pitts30k	20K / 15K	10K / 6.8K	2.0 GB	panorama	480×640	panorama
MSLS	934K / 514K	19K / 11K	56 GB	front-view	480×640	front-view
Tokyo 24/7	0 / 0	75K / 315	4.0 GB	panorama	480×640	phone*
R-SF	0 / 0	1.05M / 598	36 GB	panorama	480×640	phone*
Eynsham	0 / 0	24K / 24K	1.2 GB	panorama	512×384	panorama
St Lucia	0 / 0	1.5K / 1.5K	124 MB	front-view	480×640	front-view

Table 2. **Summary of the datasets:** "panorama" means images are cropped from a 360° panorama (including undistortion); "front-view" means that only one (forward facing) view is available; "phone" means photos were collected with a smartphone. "panorama" and "front-view" images were taken with car-rooftop cameras. * Variable resolution.

the literature [2, 9, 10, 27, 35, 40, 50, 51, 75], use 25 meters as a distance threshold, but we also investigate how results change varying thresholds and top-N (*cf*. supp. material). For reliability, all results are averaged over three repetitions of experiments. To avoid overloading the tables, standard deviations are shown in the supp. material, where the reported experiments are a superset of the ones in this manuscript. Training is performed until recall@5 on the validation set does not improve for 3 epochs. Given the variability in datasets size (see Tab. 2), we define an epoch as a pass over 5,000 queries. We use the Adam optimizer [36] for training, as in general it leads to faster convergence and better performance than SGD. Following the widely used training protocol defined in [2], we use a batch size of 4 triplets, where each triplet is composed of an anchor (the query), a positive and 10 negatives. Following standard practice [2, 10, 27, 40, 74, 75], at train time, the positive is selected as the nearest database image in features space among those within a 10 meters radius from the query and negative images selected from those further than 25. Due to the size of each dataset, we use full database mining when training on the Pitts30k, and partial mining when training on the MSLS (cf. Sec. 4.4 for details).

4. Results

Throughout this section, we explore how each block from Fig. 1 influences the results. Specifically, we first investigate the use of different architectures, with a focus on backbones (Sec. 4.1), aggregation methods (Sec. 4.2) and Transformers-based networks (Sec. 4.3). We then move to train-time components (*i.e*. negative mining Sec. 4.4 and data augmentation Sec. 4.5), to understanding how the resolution of the images influences a VG system (Sec. 4.6), and finally we explore the use of efficient nearest neighbor search algorithms (Sec. 4.7). Given the limited amount of space in the manuscript, a thorough extension over each one of these sections can be found in the supp. material, as well as further experiments on various metrics and more.

4.1. CNN Backbones

Tasked with extracting highly informative feature maps from images, the CNN backbone represents a fundamental component of any VG system. To understand its impact, we experiment with four CNN backbones (VGG16 [65], ResNet-18, ResNet-50 and ResNet-101 [28]), combined with two popular aggregation methods, GeM [58] and NetVLAD [2]. Note that this seemingly limited number of backbones covers several state-of-the-art architectures in VG and image retrieval [2, 25, 35, 40, 50, 51, 58, 61, 68, 75]. For all ResNets, we use the feature maps extracted from the *conv4_x* layer[1]. For VGG16, we use all the convolutional layers, excluding the last pooling before the classifier part. Table 3 shows the results of our experiments.

Discussion. We can see that deeper ResNets, such as ResNet-50 and ResNet-101, achieve better results w.r.t. their shallower counterparts. In particular, ResNet-50 shows recalls on par with ResNet-101, but with the advantage of less than half the FLOPs and model size, making the former a more practically relevant option than the latter. ResNet-18 performs worse, but allows for much faster and lighter computation, making it the most efficient, lightweight backbone. Moreover, results considerably depend on the training data: as an example, training the same network on Pitts30k or MSLS yields a 30% gap testing the model on St. Lucia, as well as a noticeable difference on other datasets too. This effect demonstrates that comparing models trained on different datasets, as done in [80], can be misleading.

4.2. Aggregation and Descriptor Dimensionality

Aggregations methods are layers tasked with processing the output features of the backbone. Over the years, a number of such methods have been proposed, from shallow pooling layers [5, 60] to more complex modules [2, 35]. Our framework allows to compute results with a number of them, namely SPOC [5], MAC [60], R-MAC [68], RRM [37], GeM [58], NetVLAD [2] and CRN [35]. While a complete list of results with all aggregation methods is shown in the supp. material, in Tab. 4 we report the performance of the best performing aggregators: GeM, NetVLAD and CRN. Given the difference in size of the outputted descriptors, we apply PCA or a fully connected (FC) layer to even their dimensionality.

Discussion. The results in Tab. 4 show that performance strongly depends on the training set. When training on the small Pitts30k, the best results are obtained globally with CRN, even when reducing its dimension to be the same as GeM. However, when training on the much larger MSLS, the advantage of CRN is reduced, and both CRN

[1] Preliminary results have shown on average better recall and efficiency rather than using until *conv5_x* (see Tab. 2 in the supp. material)

Backbone	Aggregation Method	Features Dim	FLOPs (GF)	Model Size (MB)	Extraction Time (ms)	Training on Pitts30k R@1 Pitts30k	R@1 MSLS	R@1 Tokyo 24/7	R@1 R-SF	R@1 Eynsham	R@1 St Lucia	Training on MSLS R@1 Pitts30k	R@1 MSLS	R@1 Tokyo 24/7	R@1 R-SF	R@1 Eynsham	R@1 St Lucia
VGG-16	GeM	512	188.01	56.13	12.3	78.5	43.4	39.9	40.4	70.2	46.4	70.2	66.7	43.6	32.1	80.4	79.9
ResNet-18	GeM	256	17.29	10.63	4.1	77.8	35.3	35.3	34.2	64.3	46.2	71.6	65.3	42.8	30.5	80.3	83.2
ResNet-50	GeM	1024	40.61	32.71	6.7	82.0	38.0	41.5	45.4	66.3	**59.0**	**77.4**	72.0	**55.4**	45.7	**83.9**	91.2
ResNet-101	GeM	1024	86.29	105.36	9.6	**82.4**	**39.6**	**44.0**	**52.5**	**69.0**	57.6	77.2	**72.5**	51.0	**46.9**	83.6	**91.6**
VGG-16	NetVLAD	32768	188.09	56.38	13.0	83.2	50.9	61.4	64.6	74.4	50.1	79.0	74.6	61.9	57.1	84.2	86.7
ResNet-18	NetVLAD	16384	17.27	10.76	4.4	86.4	47.4	63.4	61.4	76.8	57.6	**81.6**	75.8	62.3	55.1	87.1	92.1
ResNet-50	NetVLAD	65536	40.51	33.21	8.5	86.0	50.7	69.8	67.1	**77.7**	60.2	80.9	76.9	**62.8**	51.5	**87.2**	93.8
ResNet-101	NetVLAD	65536	86.06	105.86	11.5	**86.5**	**51.8**	**72.2**	**67.5**	74.0	**63.6**	80.8	**77.7**	59.0	**56.1**	86.7	**95.1**

Table 3. Results and computational requirements with different convolutional **backbones**. Extraction time is the average over a 1000 forward passes.

Backbone	Aggregation Method	Features Dim	Training on Pitts30k R@1 Pitts30k	R@1 MSLS	R@1 Tokyo 24/7	R@1 R-SF	R@1 Eynsham	R@1 St Lucia	Training on MSLS R@1 Pitts30k	R@1 MSLS	R@1 Tokyo 24/7	R@1 R-SF	R@1 Eynsham	R@1 St Lucia	R@1 Average
ResNet-50	GeM	1024	82.0	38.0	41.5	45.4	66.3	59.0	**77.4**	72.0	**55.4**	**45.7**	83.9	91.2	63.2
ResNet-50	NetVLAD + PCA 1024	1024	83.9	46.5	59.4	53.2	72.5	57.7	**77.4**	74.8	51.3	39.0	85.2	92.9	66.2
ResNet-50	CRN + PCA 1024	1024	**84.1**	**49.9**	**64.6**	**58.8**	**74.3**	**63.4**	77.3	**75.6**	51.8	38.8	**85.7**	**94.1**	**68.2**
ResNet-50	GeM + FC 2048	2048	80.1	33.7	43.6	48.2	70.0	56.0	**79.2**	73.5	**64.0**	**55.1**	86.1	90.3	65.0
ResNet-50	NetVLAD + PCA 2048	2048	84.4	47.9	62.6	56.0	74.1	58.9	78.5	75.4	52.8	42.6	85.8	93.4	67.7
ResNet-50	CRN + PCA 2048	2048	**84.7**	**51.2**	**67.1**	**62.3**	**75.8**	**65.0**	78.3	**76.3**	54.3	42.8	**86.2**	**94.4**	**69.9**
ResNet-50	GeM + FC 65536	65536	80.8	35.8	45.6	49.0	72.5	59.6	79.0	74.4	**69.2**	**58.4**	86.2	90.8	66.8
ResNet-50	NetVLAD	65536	**86.0**	50.7	69.8	67.1	77.7	60.2	**80.9**	76.9	62.8	51.5	87.2	93.8	72.1
ResNet-50	CRN	65536	85.8	**54.0**	**73.1**	**70.9**	**79.7**	**65.9**	80.8	**77.8**	63.6	53.4	**87.5**	**94.8**	**73.9**

Table 4. **Aggregation methods:** we report results with different aggregation methods downscaled or upscaled to equivalent dimensionality.

and NetVLAD end up being significantly outclassed on Tokyo and R-SF[2] by GeM, making it a more compelling choice. Furthermore, the dimensionality reduction via PCA yields a significant drop in performance for NetVLAD and CRN, while adding a fully connected layer on top of GeM gives best results when trained on a large scale dataset, which is the type of scenario for which GeM was proposed [58]. Note that while the CRN aggregator yields the most robust results, it has the drawbacks of requiring a two-stage training process that almost doubles the training time and three times more hyperparameters w.r.t. NetVLAD. In addition, depending on the initialization of its modulation layer, training does not always converge.

4.3. Visual Transformers

In this section we investigate how Visual Transformers compare to more traditional CNN-based methods in VG. For this analysis we use two popular Transformer architectures, the Vision Transformer (ViT) [17], which processes the images by splitting them into sequences of flattened 2D patches, and the Compact Convolutional Transformer (CCT) [26], which incorporates convolutional layers to insert the inductive bias of CNNs. Following [18], we use as a global descriptor the CLS token, which is the output state of the prepended learnable embedding to the sequence of patches [17]. Moreover, we test the use of CCT in conjunction with traditional aggregation methods, such as GeM [58] and NetVLAD [2], and with SeqPool, which was specifically introduced in [26] for Transformers.

[2]The reason could be that these two datasets have different query and database image types, *i.e.* phone-taken and panorama images, respectively.

Backbone	Aggreg. Method	Feat. Dim	FLOPs (GF)	Training on MSLS R@1 Pitts30k	R@1 MSLS	R@1 Tok. 24/7	R@1 R-SF	R@1 Eyns.	R@1 St Lucia
ResNet-18	GeM	256	17.29	71.6	65.3	42.8	30.5	80.3	83.2
ResNet-50	GeM	1024	40.61	77.4	72.0	55.4	45.7	83.9	91.2
ViT	CLS	768	82.31	**82.9**	**73.5**	**59.9**	**65.0**	84.5	93.6
CCT	CLS	384	22.34	79.6	71.1	52.0	49.9	85.6	**94.0**
CCT	SeqPool	384	26.19	81.4	71.0	59.1	60.5	**86.1**	92.4
CCT	GeM	384	22.36	78.7	72.0	48.8	48.6	83.9	92.9
ResNet-18	NetVLAD	16384	17.27	81.6	75.8	62.3	55.1	87.1	92.1
ResNet-50	NetVLAD	65536	40.51	80.9	76.9	62.3	51.5	87.2	93.8
CCT	NetVLAD	24576	18.53	**85.1**	**79.9**	**70.3**	**65.9**	**87.4**	**98.4**

Table 5. **Transformers** Comparison of traditional CNN architectures with novel Transformers-based approaches.

Discussion. Table 5 compares traditional CNN-based methods with novel Visual Transformer based approaches, never used before specifically for VG. The main findings of this set of experiments is that they represent a viable alternative to CNN-based backbones even without an additional aggregation steps using directly the compact and robust representation provided by the CLS token. Further improvements can be obtained when combined with aggregators such as GeM, SeqPool, NetVLAD, as shown in the table. Overall, the results show that these architectures possess better generalization capabilities than their CNN counterparts , and ViT proves to be competitive even with the much bigger NetVLAD descriptors, albeit with higher computational requirements. As for CCT, despite being incredibly lightweight, with a cost comparable to a ResNet-18, consistently outperforms the ResNet-18 and, in many cases, also the ResNet-50, which has roughly double the computational cost. Concluding, it seems that the SeqPool aggregator enhances the robustness of the CCT descriptors, providing better generalization and that NetVLAD coupled with CCT outperforms CNN-based methods. We observe similar behaviors when trained on Pitts30k (see supp. material).

Backbone	Aggregation Method	Mining Method	Space & Time Complexity	Training on Pitts30k R@1 Pitts30k	R@1 MSLS	R@1 Tokyo 24/7	R@1 R-SF	R@1 Eynsham	R@1 St Lucia	Training on MSLS R@1 Pitts30k	R@1 MSLS	R@1 Tokyo 24/7	R@1 R-SF	R@1 Eynsham	R@1 St Lucia
ResNet-18	GeM	Random	$\mathcal{O}(1)$	73.7	30.5	31.3	24.0	58.2	41.0	62.2	50.6	28.8	17.1	70.2	71.4
ResNet-18	GeM	Full database	$\mathcal{O}(\#db + \#q)$	**77.8**	**35.3**	**35.3**	**34.2**	**64.3**	**46.2**	70.1	61.8	**42.8**	**31.3**	79.3	81.0
ResNet-18	GeM	Partial database	$\mathcal{O}(k_{db} + k_q + \#pos)$	76.5	34.2	33.9	32.9	64.0	45.6	**71.6**	**65.3**	**42.8**	30.5	**80.3**	**83.2**
ResNet-18	NetVLAD	Random	$\mathcal{O}(1)$	83.9	43.6	55.1	53.8	76.3	53.5	73.3	61.5	45.0	34.8	84.9	79.7
ResNet-18	NetVLAD	Full database	$\mathcal{O}(\#db + \#q)$	**86.4**	**47.4**	**63.4**	61.4	**76.8**	**57.6**	-	-	-	-	-	-
ResNet-18	NetVLAD	Partial database	$\mathcal{O}(k_{db} + k_q + \#pos)$	86.2	47.3	61.2	**62.9**	76.6	57.1	**81.6**	**75.8**	**62.3**	**55.1**	**87.1**	**92.1**

Table 6. **Negative mining methods.** "Space & Time Complexity" refers to the complexity of building the cache, which normally is done after iterating over 1000 triplets [2, 75]. $\#db$ and $\#q$ are the numbers of database and query images, k_{db} and k_q are chosen constants (usually set to 1000), and $\#pos$ is the number of positives for the considered queries, which depends on the queries and database density.

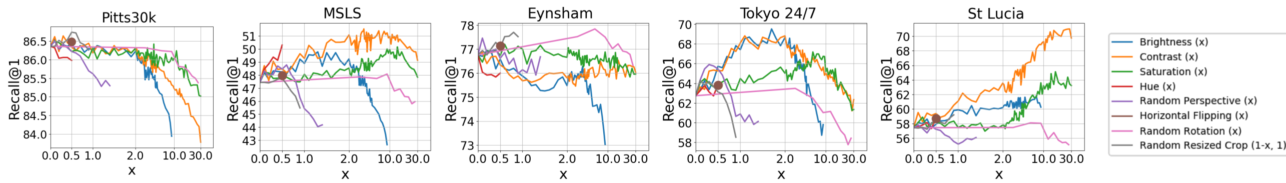


Figure 2. **Data Augmentation.** Results obtained applying popular augmentation techniques during training. We used PyTorch's transforms, and the x axis relates to the parameter passed to the class; the higher the parameter, the heavier the transform effect (*i.e.* $x = 0$ equals to the identity transformation). Refer to supp. material for further details on the transforms.

The main limitations of these architectures is the lack of an all-around best configuration. In other words, for each use case, an additional tuning on where to truncate/freeze the network was required, unlike the CNNs which were consistently used up to their *conv4* layer.

4.4. Negative Mining

An important step in a VG pipeline is the mining of negatives: ideally, we want to select images of different scenes that appear visually similar to the query to ensure that the model learns highly informative features for the task. We extensively compare three main mining strategies: full database mining [2], partial database mining [75] where only a reasonable subset of images is ranked, and random negative sampling. Details about the mining strategies, full set of results and their analyses can be found in the supp. material, here we present in Tab. 6 only a subset of our results as illustration and summarize our main findings.

Discussion. As expected, both full and partial database mining outperform the random negative sampling. The latter, in spite of its low cost yields in average 5% lower results on Pitts30k, due to the low variability of the dataset. Indeed, on the larger MSLS results drop of 10% or more. On the other hand, full database mining does not provide always best performance and on average its gain over partial mining is around 1%. Furthermore, on large scale datasets such as MSLS full mining is not feasible in a reasonable time. These results clearly show that partial mining is, in general, a great compromise between cost and accuracy.

4.5. Data Augmentation

Here we investigate if and which data augmentation are beneficial for VG methods, and if the improvements are domain-specific or can generalize to diverse datasets. We apply data augmentation to the query, with the sole exception of random horizontal flipping, for which we either flip or not flip the whole triplet. We run experiments with many popular augmentation techniques, training a ResNet-18 with NetVLAD on Pitts30k.

Discussion. Plots of the results are in Fig. 2 (shown in higher resolution in the supp. material). Depending on the test dataset, we observe different impact of these augmentations. On one hand, on Pitts30k augmentation only worsens results, probably due to dataset homogeneity between train and test. On the other hand, we see that some techniques can improve robustness on unseen datasets, in particular color jittering methods that change brightness, contrast and saturation. As an example, setting contrast[3] up to 2 can improve recall@1 by more than 3% on MSLS, 5% on Tokyo 24/7, 5% on St Lucia, with a less than 1% drop on Pitts30k and Eynsham. Although most augmentations fail to produce consistent improvements, two notable exceptions are random horizontal flipping (with probability 50%) and random resized cropping, where crops are as small as 50% of the image size (and then resized to full resolution).

4.6. Resize

While common VG datasets have images of resolutions around 480x640 pixels, it is interesting to investigate how resizing them can affect the results. To this end, we perform experiments by training and testing models on images of lower resolution, by reducing both sides of the images from 80% to 20% of their original size, both at train and test time on Pitts30k. We conduct this analysis with CNNs

[3]This refers to PyTorch's `ColorJittering()` function.

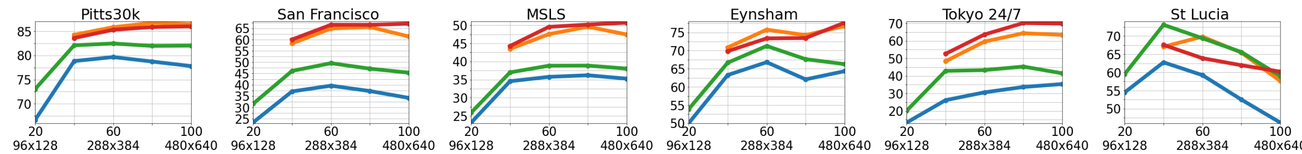


Figure 3. **Changing the images' resolution.** On the x-axis is the train and test resolution (N%), on the y-axis is the recall@1. Regarding the curves, red refers to ResNet-50 + NetVLAD, orange to ResNet18 + NetVLAD, green to ResNet-50 + GeM, and blue to ResNet-18 + GeM. In many cases, full resolution is not the optimal choice. NetVLAD's initial clusters computation breaks with low resolutions.

followed by GeM or NetVLAD, since such architectures do not require a fixed input image resolution.

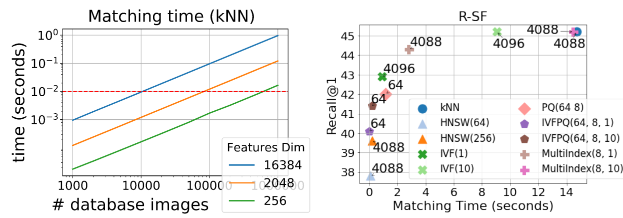
Discussion. Interestingly, it can be seen in Figure 3 that using the highest available resolution is in most cases superfluous, and often even detrimental. On average, NetVLAD's descriptors seem to better handle higher resolutions than their GeM counterparts. Lower resolutions, as low as 40%, show improved results especially when there is a wide domain gap between train and test sets: this is exemplified by the results on the St Lucia dataset, which is very different from Pitts30k (the former has only forward views) and shows best R@1 performance when using 40% of the original resolution. This behaviour can be explained by the disappearance of domain specific low-level patterns (*e.g.*, texture and foliage) when the size of the image is reduced. In general 60% is a good compromise, suggesting that for geo-localization, which is strongly related to appearance-based retrieval, fine details are not too important.

Finally, note that 40% resolution means reducing it to 192x256, with FLOPs going down to $(40\%)^2$ = 16% w.r.t. full resolution images. Storage needs also decrease in the same fashion as FLOPs, and although images are not directly needed in a retrieval system (only descriptors and coordinates are used for kNN), they can be used useful for post-processing, *e.g.*, spatial verification, or to generate a visual response for users.

4.7. Nearest Neighbor Search and Inference Time

In practical applications, one of the most relevant factors for a VG system is inference time (t_i). Once the application is deployed and has to serve the user's needs, the perceived delay depends only on t_i. Inference time can be divided into: i) **extraction time** (t_e), defined as the elapsed time to extract the features of an image, which solely depends on model and resolution; ii) **matching time** (t_m), *i.e.*, duration of the kNN to find the best matches in the database, which depends on the parameter k (*i.e.* number of candidates), the size of the database, the dimension of the descriptors, and the type of searching algorithm.

In Fig. 4a, we report a plot on how matching time linearly depends on the sizes of the database and descriptors. Figure 4b shows how the use of efficient nearest neighbor search algorithm impacts computation and memory footprint. Besides exhaustive kNN, we investigate the use of



(a) (b)

Figure 4. (a) **Matching time for one query.** The plot shows, with exact search, linear dependency on database size and features dimensionality. The red line marks the extraction time of an image for ResNet-101 + GeM; above, the bottleneck is matching time, below it is extraction time. As a rule of thumb, kNN is the bottleneck if database size times the features dimension exceeds 200M. (b) Analysis of the **Recall-Speed-Memory trade-off** using optimized indexing techniques for neighbor search. Dots refer to a ResNet-50 + GeM (feat. dim. 1024) trained on Pitts30k. On the x axis is matching time in seconds for all queries in the dataset, on the y axis recall@1. The numbers next to the dots represent the RAM requirements in MB.

inverted file indexes (IVF) [66], product quantization without and with IVF (PQ and IVFPQ) [32], inverted multi index (MultiIndex) [6] and hierarchical navigable small world graphs (HNSW) [43]. In Fig. 4b we report results computed with a ResNet-50 + GeM descriptors on R-SF. See more experiments and thorough discussions in the supp. material.

Discussion Figure 4a shows that as the database grows, inference time is dominated by matching time whereas the extraction time is generally fixed at around 10 milliseconds (see Tab. 3 and supp. material). On the other hand, Fig. 4b shows that the choice of neighbor search algorithm can bring huge benefits on time and memory footprint, with little to no loss in recalls. Among the most interesting results, IVFPQ reduces both matching time and memory footprint by 98.5%, with a drop in accuracy from 45.4% to 41.4%. Note that memory footprint is an important factor in image retrieval, since for fast computation all vectors should be kept in RAM, making large scale VG application expensive in terms of memory. For example, R-SF dataset's descriptors, with a ResNet-50 + NetVLAD, require roughly $1.05\text{M} \cdot 65536 \cdot 4\text{B} = 256\text{GB}$ of memory, thus making a RAM-efficient search technique (*e.g.* product quantization) very useful. When memory is not a critical constraint, us-

Method	Feat. Dim.	R@1 Pitts30k	R@1 Pitts250k	R@1 Tokyo 24/7
VGG16 + NetVLAD + PCA [51]	4096	85.2	86.5	68.9
VGG16 + NetVLAD [51]	32768	-	84.1	60.0
SRALNet (ICRA21) [50]	4096	-	87.8	72.1
SRALNet (ICRA21) [50]	32768	85.1	85.8	68.6
APPSVR (ICCV21) [51]	4096	87.4	88.8	77.1
APPSVR (ICCV21) [51]	32768	-	86.6	68.3
ResNet-18 + NetVLAD + PCA (Ours)	4096	86.8	87.9	72.2
ResNet-18 + NetVLAD (Ours)	16384	87.2	88.1	73.7

Table 7. Comparison between recent SOTA methods, and a simple ResNet-18+NetVLAD where we use all the insight gained from the benchmark to find its optimal configuration: training with data augmentation, resize 80%, and majority voting post-processing for Tokyo 24/7 (since queries have different resolutions).

ing a MultiIndex yields the same RAM occupancy but provides an 80% saving in matching time, losing only a 0.9 % of recall. These observations make the use of exact search hardly justified and prove that (i) recall should not be the only metric considered and (ii) for practical applications, the optimization of the neighbor search is a crucial factor that cannot be ignored.

5. Discussions and Findings

This work introduces a modular framework that allows to build, train and test a wide range of VG architectures, with the flexibility to change each component of a geo-localization pipeline. Our experiments provide valuable insights on how different engineering choices implemented at training and test time can affect both the performance and the required resources (FLOPs, storage, time).

Architecture. We found that ResNet-50 is an excellent choice as a CNN backbone, yielding close to the best results at a reasonable cost. We also demonstrate for the first time the use of Visual Transformers for VG and find that they provide compelling results compared to their CNN counterparts. Among them, CCT is particularly interesting because it is incredibly lightweight, with a cost comparable to a ResNet-18, but it performs better than a heavier ResNet-50. Regarding the feature aggregation layers, the best performance is generally obtained with CRN, nevertheless requiring a significant training cost. At the same time, the GeM pooling, which is much more efficient, has shown a better generalization power, especially when training the model on a large and heterogeneous dataset. The best results overall are obtained with CCT combined with NetVLAD.

Negative mining. In general for metric learning for retrieval, negative mining is a crucial element. This was confirmed by our experiments, where we have additionally shown that partial mining can yield similar or sometimes even better performance than full mining, but at a fraction of the (computational) cost.

Training dataset. Unsurprisingly, using a large-scale training set, with a wide range of conditions and collected from very diverse cities, leads to significantly better results. This confirms the importance of the training set and the evidence that comparisons amongst models trained on different datasets, as commonly done in many papers [35,80], are not fair and should be avoided if possible.

Image size and data augmentation. As usually observed for deep models, data augmentation generally helps. In our case we found that the effectiveness of the color jittering augmentations are highly dependent on the dataset, while horizontal flipping and resized cropping provide a slight but consistent boost in all cases. Finally, a surprising finding is that using the full resolution images (usually 480x640) is often superfluous – scaling down the images to 60% not only reduces the FLOPs, but on average yields comparable (and sometimes better) results.

Inference time and kNN search. We propose an extensive study for VG, unique in its kind, comparing advanced kNN search algorithms and compact representations. This study has shown that the choice of a good neighbor search algorithm can have a huge impact on time and memory footprint, with little impact on the performance. Furthermore, we observe that advanced kNN methods might nullify the gap in terms of both memory footprint and matching time between larger and smaller descriptors.

Final remarks All the above insights are important to design and optimize VG architectures depending on one's use case and requirements. For instance, consider again the example from Tab. 1. In light of the lessons learned, we can carefully optimize the same simple architecture to get results that are comparable with much more complex (yet not optimized) methods (see Tab. 7).

Limitations Despite its modularity and versatility, our framework has also some limitations, *e.g.*, it is focused on VG methods in outdoor urban environments, it only addresses the task of Visual Geo-localization from a single image, it does not try to analyze the viewpoint and luminosity invariance of the methods (as done in [80]). Furthermore, some recent SOTA works [23, 51] are not implemented yet, and some newer losses not yet compared [40]. However, we plan to continue supporting the software and website, expanding them to evaluate more techniques and use-cases and investigate additional elements in a VG pipeline.

Acknowledgements We acknowledge the CINECA award under the ISCRA initiative, for the availability of high performance computing resources and support. Also, computational resources were provided by HPC@POLITO, a project of Academic Computing within the Department of Control and Computer Engineering at the Politecnico di Torino (http://www.hpc.polito.it). This work was partially supported by CINI, the European Regional Development Fund under project IMPACT (reg. no. CZ.02.1.01/0.0/0.0/15_003/0000468), and the EU Horizon 2020 project RICAIP (grant agreement No 857306).

References

[1] R. Arandjelović and Andrew Zisserman. Three things everyone should know to improve object retrieval. *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2911–2918, 2012. 2

[2] Relja Arandjelović, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. NetVLAD: CNN architecture for weakly supervised place recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(6):1437–1451, 2018. 1, 2, 3, 4, 5, 6

[3] Relja Arandjelović and Andrew Zisserman. Dislocation: Scalable descriptor distinctiveness for location recognition. In Daniel Cremers, Ian D. Reid, Hideo Saito, and Ming-Hsuan Yang, editors, *Computer Vision - ACCV 2014 - 12th Asian Conference on Computer Vision, Singapore, Singapore, November 1-5, 2014, Revised Selected Papers, Part IV*, volume 9006 of *Lecture Notes in Computer Science*, pages 188–204. Springer, 2014. 2

[4] Hossein Azizpour, Ali Razavian, Josephine Sullivan, Atsuto Maki, and Stefan Carlsson. Factors of transferability for a generic convnet representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38, 11 2015. 2

[5] Artem Babenko and Victor Lempitsky. Aggregating deep convolutional features for image retrieval. *ICCV*, 10 2015. 2, 4

[6] Artem Babenko and Victor S. Lempitsky. The inverted multi-index. In *CVPR*, pages 3069–3076. IEEE Computer Society, 2012. 7

[7] Artem Babenko, Anton Slesarev, A. Chigorin, and V. Lempitsky. Neural codes for image retrieval. *ArXiv*, abs/1404.1777, 2014. 2

[8] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Computer Vision and Image Understanding*, 110:346–359, 06 2008. 2

[9] Gabriele Berton, Carlo Masone, Valerio Paolicelli, and Barbara Caputo. Viewpoint Invariant Dense Matching for Visual Geolocalization. In *IEEE International Conference on Computer Vision*, 2021. 4

[10] Gabriele Moreno Berton, Valerio Paolicelli, Carlo Masone, and Barbara Caputo. Adaptive-attentive geolocalization from few queries: A hybrid approach. In *IEEE Winter Conference on Applications of Computer Vision*, pages 2918–2927, January 2021. 1, 2, 4

[11] B. Cao, A. Araujo, and J. Sim. Unifying deep local and global features for image search. In *European Conference on Computer Vision*, pages 726–743. Springer Int. Publishing, 2020. 2

[12] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *IEEE International Conference on Computer Vision*, pages 9650–9660, October 2021. 2

[13] D. M. Chen, G. Baatz, K. Köser, S. S. Tsai, R. Vedantham, T. Pylvänäinen, K. Roimela, X. Chen, J. Bach, M. Pollefeys, B. Girod, and R. Grzeszczuk. City-scale landmark identification on mobile devices. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 737–744, 2011. 3

[14] Zetao Chen, Adam Jacobson, Niko Sünderhauf, Ben Upcroft, Lingqiao Liu, Chunhua Shen, Ian Reid, and Michael Milford. Deep learning features at scale for visual place recognition. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3223–3230, 2017. 1

[15] Gabriela Csurka, Christopher Dance, Lixin Fan, Jutta Willamowski, and Cédric Bray. Visual categorization with bags of keypoints. In *European Conference on Computer Vision*, volume Vol. 1, 01 2004. 2

[16] M. Cummins and P. Newman. Highly scalable appearance-only slam - FAB-MAP 2.0. In *Robotics: Science and Systems*, 2009. 2, 3

[17] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. *ArXiv*, abs/2010.11929, 2021. 2, 5

[18] Alaaeldin El-Nouby, Natalia Neverova, Ivan Laptev, and Hervé Jégou. Training vision transformers for image retrieval. *ArXiv*, abs/2102.05644, 2021. 2, 5

[19] Matthew Gadd, D. Martini, and P. Newman. Look around you: Sequence-based radar place recognition with learned rotational invariance. *2020 IEEE/ION Position, Location and Navigation Symposium (PLANS)*, pages 270–276, 2020. 2

[20] Sourav Garg, Tobias Fischer, and Michael Milford. Where is your place, visual place recognition? In Zhi-Hua Zhou, editor, *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 4416–4425. International Joint Conferences on Artificial Intelligence Organization, 8 2021. Survey Track. 1

[21] Sourav Garg, Ben Harwood, G. Anand, and Michael Milford. Delta descriptors: Change-based place representation for robust visual localization. *IEEE Robotics and Automation Letters*, 5:5120–5127, 2020. 1

[22] Sourav Garg and Michael Milford. Seqnet: Learning descriptors for sequence-based hierarchical place recognition. *IEEE Robotics and Automation Letters*, 6:4305–4312, 2021. 1, 2

[23] Yixiao Ge, Haibo Wang, Feng Zhu, Rui Zhao, and Hongsheng Li. Self-supervising fine-grained region similarities for large-scale image localization. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 369–386, Cham, 2020. Springer International Publishing. 1, 2, 8

[24] Albert Gordo, Jon Almazán, Jérôme Revaud, and Diane Larlus. Deep image retrieval: Learning global representations for image search. In *ECCV*, 2016. 2

[25] A. Gordo, J. Almazan, J. Revaud, and D. Larlus. End-to-end learning of deep visual representations for image retrieval. *IJCV*, 2017. 2, 3, 4

[26] Ali Hassani, Steven Walton, Nikhil Shah, Abulikemu Abuduweili, Jiachen Li, and Humphrey Shi. Escaping the Big Data Paradigm with Compact Transformers. *ArXiv*, abs/2104.05704, 2021. 5

[27] Stephen Hausler, Sourav Garg, Ming Xu, Michael Milford, and Tobias Fischer. Patch-netvlad: Multi-scale fusion of

locally-global descriptors for place recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14141–14152, 2021. 1, 2, 3, 4

[28] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. 2, 4

[29] Ziyang Hong, Yvan Petillot, David Lane, Yishu Miao, and Sen Wang. Textplace: Visual place recognition and topological localization through reading scene texts. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019. 1

[30] H. Jégou and Andrew Zisserman. Triangulation embedding and democratic aggregation for image search. *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3310–3317, 2014. 2

[31] H. Jégou, M. Douze, and C. Schmid. Hamming embedding and weak geometric consistency for large scale image search. In D. Forsyth, P. Torr, and A. Zisserman, editors, *European Conference on Computer Vision*, pages 304–317, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg. 2

[32] Hervé Jégou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(1):117–128, 2011. 7

[33] Hervé Jégou, Matthijs Douze, Jorge Sánchez, Patrick Perez, and Cordelia Schmid. Aggregating local image descriptors into compact codes. *IEEE transactions on pattern analysis and machine intelligence*, 34, 12 2011. 2

[34] 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 Transactions on Robotics*, 36(2):561–569, 2020. 1

[35] Hyo Jin Kim, Enrique Dunn, and Jan-Michael Frahm. Learned contextual feature reweighting for image geo-localization. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3251–3260, 2017. 1, 2, 3, 4, 8

[36] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 12 2014. 4

[37] Giorgos Kordopatis-Zilos, Panagiotis Galopoulos, S. Papadopoulos, and Y. Kompatsiaris. Leveraging efficientnet and contrastive learning for accurate global-scale location estimation. *ACM International Conference on Multimedia Retrieval*, 2021. 4

[38] Yunpeng Li, Noah Snavely, Daniel Huttenlocher, and Pascal Fua. Worldwide Pose Estimation using 3D Point Clouds. In *European Conference on Computer Vision*, 2012. 3

[39] Dongfang Liu, Yiming Cui, Liqi Yan, Christos Mousas, Baijian Yang, and Yingjie Chen. DenserNet: Weakly supervised visual localization using multi-scale feature aggregation. *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 6101–6109, May 2021. 2

[40] Liu Liu, Hongdong Li, and Yuchao Dai. Stochastic Attraction-Repulsion Embedding for Large Scale Image Localization. In *IEEE International Conference on Computer Vision*, 2019. 1, 2, 3, 4, 8

[41] David G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, 2004. 2

[42] Stephanie Lowry, Niko Sünderhauf, Paul Newman, John J. Leonard, David Cox, Peter Corke, and Michael J. Milford. Visual place recognition: A survey. *IEEE Transactions on Robotics*, 32(1):1–19, 2016. 1

[43] Yu A. Malkov and D. A. Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42:824–836, 2020. 7

[44] Carlo Masone and Barbara Caputo. A survey on deep visual place recognition. *IEEE Access*, 9:19516–19547, 2021. 1

[45] Michael Milford and G. Wyeth. Mapping a suburb with a single camera using a biologically inspired slam system. *IEEE Transactions on Robotics*, 24:1038–1053, 2008. 3

[46] Eva Mohedano, Kevin McGuinness, Xavier Giro i Nieto, and N. O'Connor. Saliency weighted convolutional features for instance search. *2018 International Conference on Content-Based Multimedia Indexing (CBMI)*, pages 1–6, 2018. 2

[47] Hyeonwoo Noh, Andre Araujo, Jack Sim, Tobias Weyand, and Bohyung Han. Large-scale image retrieval with attentive deep local features. In *IEEE International Conference on Computer Vision*, 2017. 2

[48] A. Oliva and A. Torralba. Building the gist of a scene: the role of global image features in recognition. *Progress in brain research*, 155:23–36, 2006. 2

[49] Eng-Jon Ong, Sameed Husain, and Miroslaw Bober. Siamese network of deep fisher-vector descriptors for image retrieval. *CoRR*, abs/1702.00338, 2017. 2

[50] Guohao Peng, Yufeng Yue, Jun Zhang, Zhenyu Wu, Xiaoyu Tang, and Danwei Wang. Semantic reinforced attention learning for visual place recognition. In *IEEE International Conference on Robotics and Automation, ICRA 2021, Xi'an, China, May 30 - June 5, 2021*, pages 13415–13422. IEEE, 2021. 4, 8

[51] Guohao Peng, Jun Zhang, Heshan Li, and Danwei Wang. Attentional pyramid pooling of salient visual residuals for place recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 885–894, October 2021. 4, 8

[52] Florent Perronnin, Yan Liu, Jorge Sánchez, and Herve Poirier. Large-scale image retrieval with compressed fisher vectors. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3384–3391, 06 2010. 2

[53] James Philbin, Ondrej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society, 2007. 2

[54] James Philbin, Ondrej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *IEEE Conference on Computer Vision and Pattern Recognition*, June 2008. 2

[55] Nathan Piasco, Désiré Sidibé, Cédric Demonceaux, and Valérie Gouet-Brunet. A survey on visual-based localization:

On the benefit of heterogeneous data. *Pattern Recognition*, 74:90–109, 2018. 1

[56] Noé Pion, Martin Humenberger, Gabriela Csurka, Yohann Cabon, and Torsten Sattler. Benchmarking image retrieval for visual localization. In *2020 International Conference on 3D Vision (3DV)*, pages 483–494, 2020. 1, 2

[57] Filip Radenović, Giorgos Tolias, and O. Chum. CNN Image Retrieval Learns from BoW: Unsupervised Fine-Tuning with Hard Examples. In *ECCV*, 2016. 2

[58] F. Radenović, G. Tolias, and O. Chum. Fine-tuning CNN Image Retrieval with No Human Annotation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018. 1, 2, 3, 4, 5

[59] A. Razavian, Hossein Azizpour, J. Sullivan, and S. Carlsson. Cnn features off-the-shelf: An astounding baseline for recognition. *2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 512–519, 2014. 2

[60] A. Razavian, J. Sullivan, A. Maki, and S. Carlsson. Visual Instance Retrieval with Deep Convolutional Networks. *CoRR*, abs/1412.6574, 2015. 4

[61] Jérôme Revaud, Jon Almazán, R. S. Rezende, and César Roberto de Souza. Learning with average precision: Training image retrieval with a listwise loss. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5106–5115, 2019. 3, 4

[62] Torsten Sattler, Will Maddern, Carl Toft, Akihiko Torii, Lars Hammarstrand, Erik Stenborg, Daniel Safari, Masatoshi Okutomi, Marc Pollefeys, Josef Sivic, Fredrik Kahl, and Tomas Pajdla. Benchmarking 6DOF outdoor visual localization in changing conditions. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8601–8610, 2018. 1, 2

[63] Grant Schindler, Matthew Brown, and Richard Szeliski. City-Scale Location Recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2007. 2

[64] Zachary Seymour, Karan Sikka, Han-Pang Chiu, S. Samarasekera, and Rakesh Kumar. Semantically-aware attentive neural embeddings for image-based visual localization. *ArXiv*, abs/1812.03402, 2018. 1

[65] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015. 4

[66] Josef Sivic and Andrew Zisserman. Video google: A text retrieval approach to object matching in videos. In *ICCV*, pages 1470–1477. IEEE Computer Society, 2003. 7

[67] Elena Stumm, Christopher Mei, and Simon Lacroix. Probabilistic place recognition with covisibility maps. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4158–4163. IEEE, 2013. 2

[68] Giorgos Tolias, R. Sicre, and H. Jégou. Particular object retrieval with integral max-pooling of CNN activations. *CoRR*, abs/1511.05879, 2016. 2, 3, 4

[69] A. Torii, R. Arandjelović, J. Sivic, M. Okutomi, and T. Pajdla. 24/7 place recognition by view synthesis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(2):257–271, 2018. 1, 2, 3

[70] A. Torii, J. Sivic, M. Okutomi, and T. Pajdla. Visual place recognition with repetitive structures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(11):2346–2359, 2015. 1

[71] A. Torii, Hajime Taira, Josef Sivic, M. Pollefeys, M. Okutomi, T. Pajdla, and Torsten Sattler. Are large-scale 3d models really necessary for accurate visual localization? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43:814–829, 2021. 2, 3

[72] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 10347–10357. PMLR, July 2021. 2

[73] O. Vysotska and C. Stachniss. Effective visual place recognition using multi-sequence maps. *IEEE Robotics and Automation Letters*, 4:1730–1736, 2019. 1

[74] Z. Wang, J. Li, S. Khademi, and J. van Gemert. Attention-aware age-agnostic visual place recognition. In *The IEEE International Conference on Computer Vision (ICCV) Workshops*, Oct 2019. 1, 2, 4

[75] Frederik Warburg, Soren Hauberg, Manuel Lopez-Antequera, Pau Gargallo, Yubin Kuang, and Javier Civera. Mapillary street-level sequences: A dataset for lifelong place recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, June 2020. 1, 2, 3, 4, 6

[76] Isaac Ronald Ward, M. Jalwana, and M. Bennamoun. Improving image-based localization with deep learning: The impact of the loss function. In *PSIVT Workshops*, 2019. 1

[77] Tobias Weyand, A. Araújo, Bingyi Cao, and Jack Sim. Google landmarks dataset v2 – a large-scale benchmark for instance-level recognition and retrieval. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2572–2581, 2020. 2

[78] Zhe Xin, Xiaoguang Cui, Jixiang Zhang, Yiping Yang, and Yanqing Wang. Visual place recognition with cnns: From global to partial. In *2017 Seventh International Conference on Image Processing Theory, Tools and Applications (IPTA)*, pages 1–6, 2017. 1

[79] Mubariz Zaffar, Shoaib Ehsan, Michael Milford, and K. Mcdonald-Maier. Cohog: A light-weight, compute-efficient, and training-free visual place recognition technique for changing environments. *IEEE Robotics and Automation Letters*, 5:1835–1842, 2020. 2

[80] Mubariz Zaffar, Sourav Garg, Michael Milford, Julian Kooij, David Flynn, Klaus McDonald-Maier, and Shoaib Ehsan. VPR-Bench: An open-source visual place recognition evaluation framework with quantifiable viewpoint and appearance change. *International Journal of Computer Vision*, 129(7):2136–2174, 2021. 1, 2, 4, 8

[81] Amir R. Zamir, Asaad Hakeem, Luc Van Gool, Mubarak Shah, and Richard Szeliski, editors. *Large-Scale Visual Geo-localization*. Advances in Computer Vision and Pattern Recognition. Springer, 2016. 1

[82] Xiwu Zhang, Lei Wang, and Yan Su. Visual place recognition: A survey from deep learning perspective. *Pattern Recognition*, 113, 2021. 1

[83] Y. Zhu, J. Wang, L. Xie, and L. Zheng. Attention-based pyramid aggregation network for visual place recognition. In *Proc. of the 26th ACM Int. Conf. on Multimedia*, MM '18, page 99–107, New York, NY, USA, 2018. Association for Computing Machinery. 2