# Image Retrieval for Visual Geolocalization

**TA: Gabriele Berton, Gabriele Trivigno** ()

**Q&A: link**

## OVERVIEW

Visual geo-localization (VG) is the task of coarsely finding the geographical position where a given photograph was taken. This task is commonly addressed as an image retrieval problem: given an unseen image to be localized (query), it is matched against a database of geo-tagged images that represent the known world. The N-top retrieved images, with their geo-tag (typically GPS coordinates), provide the hypothesis for the query's geographical location.

The retrieval is performed as a k Nearest Neighbour search in a learned embedding space that well represents the visual similarity of places. Namely, each image is passed through a network composed of a feature extraction backbone and a head that aggregates or pools the features to create a global descriptor of the image. By using a contrastive learning approach. The model learns to extract descriptors that are discriminative of locations. The similarity search is then implemented as a pairwise comparison among descriptors, e.g. using a cosine similarity.

In this project, you will become familiar with the task of visual geo-localization and with how a visual geo-localization system works. You will start by experimenting with the baselines using a codebase provided by us that contains all the most common methods. Subsequently you will focus on improving one or more aspects of the system of your choice, from robustness to illumination changes (e.g. pictures taken at night), different perspectives and occlusions.

We already provide you with the code to implement the baseline, so that you do not have to waste time in re-implementing them; in this way you can focus on understanding the code, and then try to tackle real research problems. If you obtain any significant results, if you are interested you can bring the project to a publication.

## GOALS

1. Get acquainted with the field of Visual Geo-Localization; understand similarities and differences with Image/Landmark Retrieval

2. Run some experiments with the most popular methods in the literature, GeM [2] NetVLAD [1]  train them on the Pitts30k dataset and understand how the mining and the contrastive learning works.
3. Propose your own extension to improve the system or to overcome some of the challenges for the task. We propose some approaches, however you are free to identify other issues you want to focus on..

## DATASETS

At this link you are provided with 3 datasets to experiment with: **pitts30k** [1], San Francisco small (**sf-xs**) [19] and Tokyo  small (**tokyo-xs**). The datasets sf-xs and tokyo-xs are respectively subsets of San Francisco eXtra Large [20] and Tokyo 24/7 [16]: we purposely reduced the size of their database (hence the name small) for your convenience.

You should also create a new dataset, called **tokyo-night**, which is a subset of tokyo-xs: to create this you can copy the tokyo-xs dataset, and then remove all the queries that are not taken at night. In the tokyo-xs dataset, there are 315 queries, of which only 1 out of 3 (so 105) are taken at night. You should do this manually, it won't take long and it will force to visually inspect your dataset. To recap, tokyo-xs should contain 315 queries, and tokyo-night 105 queries.

You will mainly use pitts30k for training. Testing of all models should be performed on pitts30k (test), sf-xs (test), tokyo-xs and tokyo-night. Note that sf-xs also has a train and val set, but you will not use them in this project, and you can simply ignore them.

The reason for using different test sets is to understand how your changes to the model affect the geolocalization for different datasets: for example, some extensions might improve results for tokyo-night, but worsen results on pitts30k (test)

## STEPS

### 1. Study the literature and get familiar with the task

As a preliminary step to get familiar with the task, start by reading about the most popular methods in literature, NetVLAD [1] and GeM [2], which serve as the basis for most of the recent architectures. Especially read carefully the whole NetVLAD paper, given that the training technique from NetVLAD  is the one that you will use throughout this project: make sure to fully understand the weakly supervised triplet loss and how positive and negative images are chosen (i.e. the mining). You can also refer to this survey [3] for a broader overview of the task and its evolution through the years.

## 2. Experiment with NetVLAD[1] and GeM [2] as Baselines

Once you are familiar with the theory of visual geo-localization, you can start to run some experiments to understand how the training procedure works. You will use a ResNet-18 pre-trained on ImageNet and use as head both the NetVLAD layer [1] and the GeM pooling [2].

You will need to train both baselines using a triplet loss as shown in [1] and using the Pitts30k dataset [1]. Before getting started with the code, make sure to understand how the 3 datasets are built, how dense it is, which labels are available, plot some diagram with the distribution of the labels and visualize some images.

Test the models trained on Pitts30k also on sf-xs, Tokyo-xs and Tokyo- night.

To get started, you are provided with a GitHub repository that will provide a skeleton of the project (including the mining procedure that is required to train the model, as explained in [1], and some details about the choice of hyperparameters).

At test time, the query image is deemed correctly localized if at least one of the top N retrieved database images is within d = 25 meters from the ground truth position of the query. To assess these baselines, you will have to report the values of recall@N (for N=1,5), that is the percentage of correctly recognized queries (Recall) using the N top retrieved images.

Your first table should look something like this

| | Pitts30k (test) | sf-xs | Tokyo-xs | Tokyo-night |
|---|---|---|---|---|
| GeM | R@1/R@5 | ... | | |
| NetVLAD | ... | ... | | |

Note that Tokyo-xs has 315 queries, and Tokyo night only 105 (which are a subset from Toky-xs).

## 3. Add personal contribution

It is now time for you to put in practice what you learned in the previous steps and propose an additional study or an improvement of one of the baselines.

The objective of you contributions should be to improve results on the more challenging datasets (most important are sf-xs and Tokyo-night), improve usability of the system (e.g. speed up training time), or simply understand why some of these methods might not work. It is okay if a method provides an improvement on a dataset while decreasing results on others. You should take inspiration from the examples of extensions presented below as well as from the listed referencences. **Each extension below provides a certain amount of points: you should work on enough extensions to reach at least 50 points. . We don't want all groups to choose the same extensions, therefore as soon as you choose your extensions, add your group to this sheet. Changes can be made later, if you want to change extension, as long as there are available slots**

Always make sure to deeply understand what you are doing, make some guesses on how your changes could influence the neural net, and, when applying data augmentation, visually inspect the augmented images. The extensions should be computed with the best performing architecture (i.e. NetVLAD).

## 4. Deliverables

To conclude the project you will need to:
- Deliver PyTorch scripts for all the required steps.
- Write a complete pdf report. The report should contain a brief introduction, a related works section, a methodological section for describing the algorithms you are going to use, an experimental section with all the results and discussions. End the report with a brief conclusion.

# POSSIBLE EXTENSIONS

## a. Improve robustness to Night domain

Traditionally, resilience to domain shift, especially regarding the Day/Night domains, are critical for VG models. For this task, the objective will be to improve the recalls on Tokyo-night (but show recall on all datasets). We suggest you some possible strategies:

- **[10 points] "Smart" Data Augmentation**: using Data Augmentation in a clever way is also a good opportunity to increase robustness. For example, the standard 'torchvision.transforms.ColorJitter' transform, given a value of brightness, uses a uniform sampling inside "[max(0, 1 - brightness), 1 + brightness]", and therefore the effect is to obtain only mild changes. Try instead to use "torchvision.transforms.functional.adjust_brightness" that lets you specify an exact value. One idea would be to ensure that the parameter passed to the function adjust_brightness is often close to 0, given that the closer its value is to 0, the darker the image. However this is just an example and you can experiment with many of the other transforms that torch offers.

- **[20 points] Domain Adaptation:** you can borrow some techniques from the field of Domain Adaptation, like using a GRL (Gradient Reversal Layer), or methods that use a loss to impose the same distribution of the feature's norm across domains. In this case you can use the queries from Tokyo-night for training (without labels!) as well
- **[20 points] Synthetic images as data augmentation**: you can try to use GANs or style transfer methods to transform day pictures to night pictures. You can use these generated synthetic images to train your model, as these images will act as an improved data augmentation. You can try to train your model only with these synthetic (night) images, or you could use together day and synthetic (night) images. To convert images from day to night you could use this GitHub repo, this one, or other repos that you find online.

### b. Improve robustness to perspective changes and occlusions

VG neural nets have a hard time to work properly in cases where the queries have a strong change in perspective w.r.t. the database (e.g. the query is taken from the sidewalk, and the database from the middle of the street). Similarly, occlusions (i.e. when people or vehicle occlude part of the photo) also represent an unresolved challenge in the field. To overcome these challenges, you can try the following techniques, and report recalls on all datasets, given that all contain a significant number of images with perspective changes or occlusions.

- **[10 points] Data Augmentation:** Again, some clever data augmentation can be helpful to make the network robust to perspective changes. The RandomPerspective transform performs a warping of the image. Understand the meaning of this transformation and why it can be useful, and then find a way to remove the black background that is generated around the image due to the warping. Likewise, for robustness to occlusions study the RandomErasing transform and its possible application

- **[30 points] Spatial Verification:** Spatial verification is a technique for prediction refinement based on local-features that tries to find geometrically consistent matches between the detected features. You can try methods such as D2-Net or Patch-NetVLAD (google them!)

### c. General improvements

- **[10 points] Ensembles:** To improve the recalls you can try to concatenate the descriptors obtained with different models (ensemble). For example, you can use the models trained in the previous step, and check if a combination of those can lead to better results.
- **[10 points] Multi-scale testing:** To compute an image's descriptor, you can pass it through your model at different resolutions (multiscale) and then concatenate or average the obtained descriptors.
- **[20 points] Re-Ranking / Post-Processing methods:** You can try to implement any strategies to refine the top-k candidates retrieved with the kNN. For example, for a given query, if the first candidate comes from place A, but the next 9 candidates come from place B, we can be confident that the query is from place B, and therefore we can shift the candidates from place B before the one from place A (this is called re-ranking). Another method would be to simply remove redundant candidates if they are from the same place (and keep only one from a given place). Note that the concept of a place can't be well defined and its definition will influence the results. For example, you can define two candidates to come from the same place if they are less than 5 meters apart.
- **[10 points] Optimizer & Schedulers:** As the base parameter for training, the default is the Adam optimizer with LR 1e-5. Try to experiment with different recently proposed optimizers (beyond SGD) available in torch, like AdamW, ASGD. Try to find the better LR, weight decay parameters and understand what happens changing them. You can also try different schedulers like ReduceLrOnPlateau, or CosineAnnealingLR. These should be especially useful if you decide as well to swap the backbone moving to Transformer-based ones.

- **[20 points] Domain adaptation:** you can try to perform domain adaptation on any set of queries from the test sets (without labels!), and see if that helps with the final results

## QUESTIONS YOU SHOULD BE ABLE TO ANSWER IN THE END

- What is contrastive learning and how it relates to image retrieval?
- What is the meaning of the 'mining' procedure and what are its downsides?
- What is the relationship of NetVLAD and GeM methods with the recall-scalability trade-off?

## LITERATURE

[1] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla and J. Sivic, *NetVLAD: CNN architecture for weakly supervised place recognition*, TPAMI 2018

[2] F. Radenovic, G. Tolias, and O. Chum, *Fine-tuning CNN Image Retrieval with No Human Annotation*, TPAMI 2018.

[3] C. Masone and B. Caputo, *A survey on deep visual place recognition*, IEEE Access 2021

[4] L. Liu, H. Li, and Y. Dai, *Stochastic Attraction-Repulsion Embedding for Large Scale Image Localization*, ICCV 2019.

[5] Y. Xia, Y. Xu, S. Li, R. Wang, J. Du, D. Cremers and U. Stilla, *SOE-Net: A Self-Attention and Orientation Encoding Network for Point Cloud Based Place Recognition*, CVPR 2021

[6] T. Ng, V. Balntas, Y. Tian and K. Mikolajczyk, *SOLAR: Second-Order Loss and Attention for Image Retrieval*, ECCV 2020

[7] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*, ICLR 2021

[8] A. Hassani, S. Walton, N. Shah, A. Abuduweili, J. Li and H. Shi, *Escaping the Big Data Paradigm with Compact Transformers*, preprint 2021

[9] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, *BERT: Pre-training of deep bidirectional transformers for language understanding*, In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human LanguageTechnologies

[10] H. J. Kim, E. Dunn, and J.-M. Frahm, *Learned contextual feature reweighting for image geo-localization*, CVPR 2017

[11] G. Berton, V. Paolicelli, C. Masone and B. Caputo, *Adaptive-Attentive Geolocalization From Few Queries: A Hybrid Approach*, WACV 2021

[12] S. Woo, J. Park, J.-Y. Lee and I. S. Kweon, *CBAM: Convolutional block attention module*, ECCV 2018

[13] X. Wang, R. Girshick, A. Gupta and K. He, *Non-local neural networks*, CVPR 2018.

[14] F. Warburg, S. Hauberg, M. Lopez-Antequera, P. Gargallo, Y. Kuang and J. Civera, *Mapillary Street-Level Sequences: A Dataset for Lifelong Place Recognition*, CVPR 2020

[15]D. M. Chen, G. Baatz, K. K ̈oser, S. S. Tsai, R. Vedantham,T. Pylv ̈an ̈ainen, K. Roimela, X. Chen, J. Bach, M. Pollefeys, B. Girod, and R. Grzeszczuk. City-scale landmark identification on mobile devices. In IEEE Conference on ComputerVision and Pattern Recognition, pages 737–744, 2011.

[16]A. Torii, R. Arandjelovi ́c, J. Sivic, M. Okutomi, and T.Pajdla. 24/7 place recognition by view synthesis. IEEETransactions on Pattern Analysis and Machine Intelligence,40(2):257–271, 2018.

[17]Yang, Min, Dongliang He, Miao Fan, Baorong Shi, Xuetong Xue, Fu Li, Errui Ding and Jizhou Huang. “DOLG: Single-Stage Image Retrieval with Deep Orthogonal Fusion of Local and Global Features.” *2021 IEEE/CVF International Conference on Computer Vision (ICCV)* (2021): 11752-11761.

[18] Cao, Bingyi, Andre F. de Araújo and Jack Sim. “Unifying Deep Local and Global Features for Image Search.” *ECCV* (2020).

[19] Berton, G. and Mereu, R. and Trivigno, G. and Masone, C. and Csurka, G. and Sattler, T. and Caputo, B. “Deep Visual Geo-localization Benchmark.” CVPR (2022).

[20] Berton, G. and Masone, C. and Caputo, B. “Rethinking Visual Geo-localization for Large-Scale Applications.” CVPR (2022).