Deep Learning Techniques for Visual Place Recognition

Olloshukur Atadjanov
Politecnico di Torino
s274478@studenti.polito.it

Arcangelo Frigiola
Politecnico di Torino
s295406@studenti.polito.it

Maria Rosa Scoleri
Politecnico di Torino
s301841@studenti.polito.it

***Abstract*—Visual Geolocalization (VG) or Visual Place Recognition (VPR) consists in determining the location where a given query photograph was taken. This task is performed using image matching and retrieval methods on a database of images with known GPS (Global Positioning System) coordinates. Most recent works on VG still face several challenges in building models robust to night domain, occlusion, perspective changes, and require effective methodologies to combine features extracted from different models. In this work we propose several approaches to face some of these issues: to begin with, data augmentation techniques are used to improve robustness against changes in perspectives and occlusions. Afterwards, a night domain study is performed in two different ways: the first via a 'smart' data augmentation, modifying images parameters like brightness and pixel colors, and the latter through the creation of synthetic night images with the use of UNIT networks. This last technique in particular allows us to drastically improve the results on specific datasets. Eventually, multiscale testing and ensembles methods are adopted to further improve the results.**

I. Introduction

In the last few years, the research on the VPR task has seen a rapid growth, sustained by the increasing availability of large geolocalized image datasets. This growth is prompted by several services that rely on VPR such as 3D reconstruction, augmented reality, consumer photography, and and it is of interest to other fields of research like robotics and autonomous systems. Geolocalization is generally considered an image retrieval problem: given a query as input the model is required to find images that depict the same place comparing the feature extracted from the input with those obtained from a gallery of geotagged images. Recent studies use convolutional neural networks to extract the most relevant features from images, and have reached very promising results. Despite this continuous expansion VPR still has to deal with considerable challenges such as changes in weather conditions, illumination, viewpoint shifts, season variability and occlusion. In addition to problems related to the domain of the images that constitute the data, the VG task necessitates new approaches to improve the models used and/or combine the results obtained with different models.

The work presented in [1] provides us with a standardized framework that allows to build, train and test a broad range of VG techniques offering flexibility in changing the components of a geo-localization pipeline. Starting from this baseline, we propose a range of further methodologies to improve the given benchmark which focuses on different aspects of the VG task:

- **Data Augmentation for Perspective Changes and Occlusions**: a classic VPR issue is facing perspective changes and occlusions in real-life images. We provide some methodologies to improve recall on this problem.
- **Data Augmentation for Night Domain**: as mentioned above, one of the most relevant difficulties in this field of research is the resilience to domain shift, especially regarding night and day. This is the motivation behind our focus on two data augmentation techniques to improve the model's robustness to the night domain.
  - *'Smart' Data Augmentation* is implemented with the use of PyTorch transformations such as the adjust brightness and solarize functions.
  - *Synthetic Images generation* performed employing the Unsupervised Image-to-Image Translation Networks [2] to create a new dataset with synthetically generated night images. This significantly improves the performance of the original model.
- **Ensembles**: in order to further improve the recalls found in the benchmark, we perform the concatenation of the descriptors obtained with different models.
- **Multi-scale Testing**: an image can be passed through the model at different resolutions and the descriptors obtained can be used in conjunction to gain more accurate results. We performed the concatenation and a custom average of the descriptors and the outcome is satisfactory.

II. Related Works

A. Deep Visual Geo-localization Benchmark

The main related work of this study is the [1], which provides a new open-source benchmarking framework for Visual Geo-localization. [1] performs a large suite of experiments that provide criteria for choosing backbone, aggregation, and negative mining depending on the use case and the requirements. We use this benchmark as a baseline to perform our experiments and test further techniques to improve the recalls. Authors show that small but consistent improvements can be obtained by simple pre/post-processing techniques.

B. Multi-scale learning and testing

Several recent works use multi-scale features to improve VR models. Multi-scale can be used in the learning process, (*i.e.* while training the model), or as a post-processing technique.

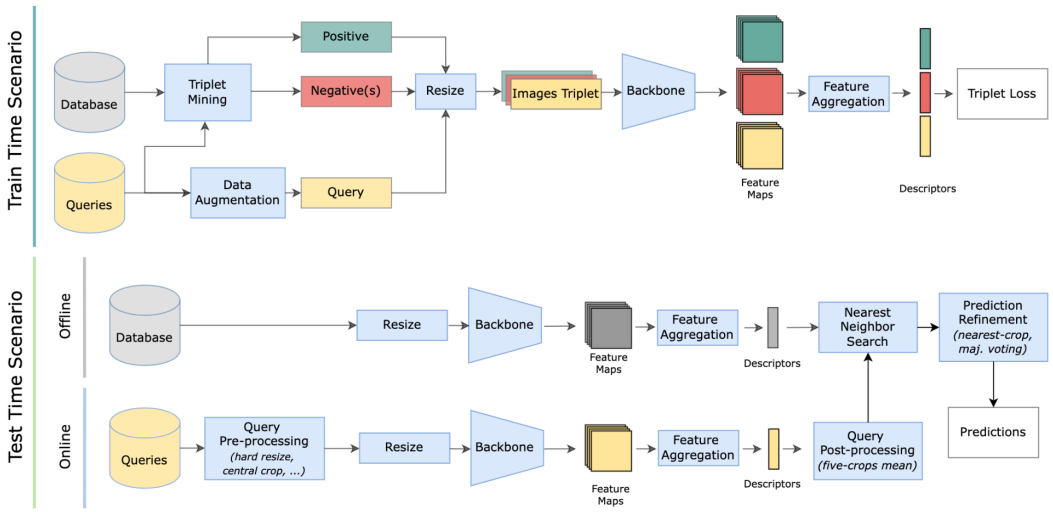

Fig. 1: **Diagram of a Visual Geo-localization system.** This diagram shows each component of a visual geo-localization system (blue boxes) comparing a variety of different implementations, both for training and test time. [1]

The work in [3] is an example of multi-scale process *after* training: the authors alter the last layer of NetVLAD with multiple patch sizes to improve local feature mapping. The work in [4] instead uses the multi-scale method to train the selected model and introduces new techniques to re-weight and select more relevant features. In this research we limit ourselves to performing a testing multi-scale approach that concatenates or averages descriptors of images in different sizes.

C. Night-to-day image translation for image retrieval

As previously stated, image retrieval across particularly different illumination conditions is still one of the most arduous aspects of VPR. Recent works are now studying the use of GANs to convert images from one domain to another reaching promising results. The work presented in [5] is based on a modified image-translation model used to transform night images into a more useful daytime representation. The aforementioned work creates synthetic day images at inference time starting from a query image. Rather than using synthetically generated day images to improve the model performance at test time, we propose to convert an entire dataset of day images to the night domain and use the combination of both these datasets to train the model.

III. Methodology

The work presented in this paper is built starting from the *Deep Visual Geo-localization Benchmark* and it is mainly targeted to improve the model robustness to night domain, perspective changes, and other general improvements. Section III-A describes the underlying architecture used for this work. Section III-B presents the datasets used to train and test the model are described, whereas Section III-C explains the techniques adopted for our research.

A. Baseline

This work is built upon the baseline [1]. It provides a VG pipeline based on image recognition: given a *query* to be geolocalized, its location is computed by matching it against a database of geotagged pictures. A more detailed description of the databases used will be reported in III-B. As shown in Fig. 1, the pipeline is built through several design choices and components, which can be changed in order to train and test different solutions, such as the **backbone**, the **feature aggregation**, the **data augmentation**.

For the purpose of this work, which is experimenting different techniques to improve the baseline results, we decide to fix the backbone to the one which performs best in terms of lighter computation, **ResNet-18**. We seek for the best performances analyzing the results between the two available feature aggregators: **NetVLAD** [6] and **GeM** [7].

The benchmark system (Fig.1) is composed of two parts: the training one, where the algorithm learns how to extract the descriptors, and the test one, where a KNN search is performed between the extracted descriptors from the database (*offline*) and the one from the query (*online*). At training time, batches of 4 triplets are created, and every triplet consists of the query, the positive and ten negatives. The positive is selected among those that within a radius of 10 meters from the query. Negative ones are instead picked among those with a distance further than 25 meters. At this point, the images pass through a CNN backbone (in this work **ResNet-18**), combined with a feature aggregator in order to extract the descriptors. At test time, the algorithm extracts the descriptors for the database images and for the query, which will be used to perform a *KNN* search in the descriptor space.

The metric used to evaluate every experiment is the recall@N (R@N) which measures the percentage of queries for which one of the top-N retrieved images was taken within a

certain distance of the query location.

B. Datasets

Our work is performed with the use of different datasets, some provided and others generated to perform specific tasks.

Provided datasets:

- **Pitts30k** is a subset of Pitts250k split in train, val and test set. It contains pictures of Pittsburgh collected from Google Street View imagery. All the pictures portray the city during the day and the database and queries folder are taken two years apart. We use this dataset for training and testing.
- **sf-xs** is a subset of San Francisco eXtraLarge and is composed of images collected by a car-mounted camera or by phone. This dataset is only used for testing.
- **tokyo-xs** is a subset of Tokyo 24/7 and presents a large dataset of images taken from Google Street View against a small number of queries. The peculiarity of this dataset consists in the fact that the queries folder is split into three equally sized sets: day, sunset and night. Similarly to sf-xs this dataset is only used for testing

Built datasets:

- **Tokyo-night** is a subset of tokyo-xs and is created by copying the tokyo-xs dataset and removing by hand all the pictures in the queries folder that are not taken at night. This dataset is used exclusively for testing.
- **Pitts30k_night** is a dataset generated with the use of the UNIT network that contains all the pictures in the Pitts30k converted to the night domain. This dataset is used for training to improve robustness to the night domain. More details on how it was created and used are available in III-C1.
- **Pitts30k_dn** is obtained by merging the two datasets Pitts30k and Pitts30k_night. Similarly to the Pitts30k_night,it is used to train the model and improve the performance on the night domain.

C. Improvements

1) Data Augmentation for the Night Domain:

Smart Data Augmentation. Data Augmentation is one of the most common yet effective ways to improve robustness in certain domains. With this idea, we experiment some image transformations with the help of PyTorch's *torchvision.transforms*. We proceed by looking for the best combinations in order to generate as output an image that is the most similar to a night-taken one, starting from a day one. The purpose is to train our model with enough night images such that it could be able to efficiently perform over a night dataset. For this experiment, we evaluate the results over the built database *tokyo-night*.

Synthetic Images as Data Augmentation. The baseline model is trained using Pitts30k which lacks night images. As a consequence, the model is not robust to the night domain and performs poorly in datasets such as tokyo-xs and especially Tokyo-night. To address this issue we resort to the generation of synthetic night images using the UNIT network proposed in [2]. This framework is based on variational autoencoders

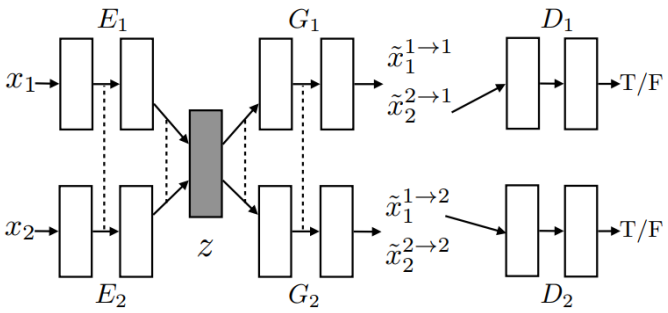

Fig. 2: The encoders-generators pairs $\{E_1, G_1\}$ and $\{E_2, G_2\}$ constitute the two VAEs used: the two encoders map the two images (x_1 and x_2) to the same latent code z and the two generators G_1 and G_2 map the code z to images. $\tilde{x}_1^{1\to1}$ and $\tilde{x}_2^{2\to2}$ are self-reconstructed images while $\tilde{x}_1^{1\to2}$ and $\tilde{x}_2^{2\to1}$ are domain-translated images. The pairs $\{D_1, G_1\}$ and $\{D_2, G_2\}$ constitute the two GANs used: G_1 creates one image reconstructed in the same domain as x_1 and one image that translates x_1 to the domain of x_2; the same applies to G_2. The adversarial discriminators D_1 and D_2 are in charge of evaluating whether the translated images are realistic.

(VAEs) [8] and GANs [9]. Starting from two sets of images belonging to two different domains (such as day and night, sunny and rainy, winter and summer) this network is able to generate two corresponding sets of images with inverted domains. The strongest feature of this network is that the training process is not required to have the same photograph taken in both domains but works using completely different pictures. Other details on this network's structure are shown in Fig. 2.

To implement our task we use a pre-trained UNIT network that is specifically trained to convert day images to night and vice versa. Several versions of the trained UNIT network are available, we choose the best one for our case by comparing them as shown in Fig. 3. The main goal of this work is to improve the recalls on tokyo-xs and tokyo-night. Examining the night pictures contained in the two mentioned datasets, we notice that most of them have several sources of illumination, objects are still very visible and discernible and they are not entirely dark. According to these considerations we choose amongst the UNIT models that produce the most realistic night images without extremely compromising the visibility of places portrayed in the pictures. These observations lead us to employ the unit_512x288_VGG_1 model.

With the aforementioned model, we generate the Pitt30k_night dataset which contains all the pictures present in Pitts30k converted to the night domain. With this new dataset we train the baseline model. The adoption of night images exclusively does not improve the benchmark results for any of the datasets.

The procedure that allows us to drastically improve the baseline results on tokyo and tokyo-night is the creation of Pitts30k_dn. This dataset is an exact merge of Pitts30k and Pitts30k_night which means that every folder contains all the

Fig. 3: **Synthetic Night Images.** The top left image represents the input image and all the others contain the night image generated with the model specified in each picture. Every image is obtained with the UNIT framework presented and with different normalization layers and encoders.

pictures present in Pitts30k and all the corresponding night images. The training with this dataset grants us the state-of-the-art results on tokyo and tokyo-night.

D. Data Augmentation for Perspective Changes

We perform another data augmentation experiment targeted to improve robustness against occlusions and perspective changes. A constant issue in place recognition is the variability of the environment. Indeed, if we take for instance a picture of a street, there could be several elements that are not of interest for the descriptors extraction, such as cars and pedestrians. Furthermore, the goal of image retrieval for place recognition is to be able to match a given query with the most similar geo-tagged picture, regardless of the angle, the height or the perspective from which the photo is taken.

To pursue this robustness we use one more time PyTorch's *torchvision.transforms*, adding fake occlusion components and changing perspective to the given pictures.

E. Ensembles

In order to further improve the recalls, we perform an ensemble of different models. Ensemble methods is a machine learning technique that combines several base models in order to produce one optimal predictive model.

A great alternative to choosing models based on their performances is to gather them together with an ensemble method. From a provided model, there exist several ways to proceed with ensemble learning, such as providing to a network different backbone algorithms, or aggregating features from a single backbone through different aggregation layers, and then merging the obtained descriptors.

For this experiment we choose two different backbone models: *AlexNet* and *ResNet-18*. As we compute feature maps, we feed them into the same aggregation layer separately and, before performing the Nearest Neighbour algorithm, we ensemble the descriptors from each model. Two different approaches are performed for the ensembling: *tensor addition* and *division*.

F. Multi-scale Testing

In order to obtain richer or more precise descriptors of an image, one possible approach is to consider different image resolutions and concatenate or average the descriptors obtained. To perform this task we start from the work proposed in [10]. An image can be represented using a tensor with dimensions *D x H x W* where D is the number of channels or feature maps, H is the height of the image and W is the width. The implementation we developed proposes to resize the images of a factor *step* that allows skipping pixels in the input image in a way that the final image has the following dimensions:

$$I^{step} = \left(3, \frac{H_{step}}{step}, \frac{W_{step}}{step}\right) \tag{1}$$

Our work allows to resize the pictures with multiple step factors. The resized pictures are then passed through the designed backbone and the new features are obtained. These features can be combined in two different ways:

- **Concatenation**: the features extracted from every resized image can simply be concatenated to obtain a tensor with a more complete description of the picture considered;
- **Average**: a true average can not be implemented since the tensor obtained from images with different sizes have different lengths and therefore the sum can not be performed. In particular, the tensors that represent the feature vectors of resized images are shorter than the tensor that represents the non-resized image. To overcome this problem we define an approach that can be considered a

naive average: we repeat the shorter tensors so that their final length reaches the one of the original tensor and finally we perform the mean over every tensor.

Finally, the aggregation layer is applied to the concatenated or averaged descriptors. Experiments with both NetVLAD and GeM are performed.

IV. Experiments

We evaluate our methods on the following four datasets: Pitts30k, sf-xs, tokyo-xs, tokyo-night. The metrics used to validate the results are the recall@1 and the recall@5 respectively defined as the percentage of queries for which the first (R@1) and the top five (R@5) predictions are not further than a certain threshold.

A. Data Augmentation for the Night Domain

- **Smart Data Augmentation.** As a very first experiment, we perform a series of data augmentation transformations on the images in the training dataset. We try to achieve the best combinations in order to transform in a more realistic fashion a day-light picture into a night-light one. To do that, we choose three transformations functions made available by PyTorch's *torchvision.transform* package:
  1) *colorJitter*: through this transformation is possible to change brightness, contrast, saturation, and hue. These components are sampled uniformly from [max(0, 1 - param_value), 1 + param_value], except for the hue value, which is sampled uniformly from [-hue,hue].
  2) *adjust_brightness*: this transformation takes as input an image and float value between 0 and 1, and precisely changes the picture brightness according to the latter parameter.
  3) *solarize*: this functional transformation inverts all pixel values above a certain threshold for an RGB/grayscale image.

  Fig. 4 displays a series of transformations performed tuning the parameters presented above, until we reach a reasonable representation of the night domain. We also test various combinations among the three functions, achieving interesting results combining the *adjust_brightness* and the *solarize* augmentations, which we choose to implement in the training phase.

  As shown in Table I, good improvements are obtained with respect to the baseline on the target dataset: *tokyo-night*. The training is performed with three epochs, and tested over all the test datasets provided. The increase of the recall on *tokyo-night* is of 3% for R@5 and about 10% for R@1, with *NetVLAD* as features aggregator. Results remain quite similar for *GeM*.
- **Synthetic Images as Data Augmentation** The synthetic image generation brings to the creation of Pitts30k_night and Pitts30k_dn. The training on both these datasets is performed with two epochs. The results in Tab. II show the R@1 and R@5 divided based on the training dataset used and show a substantial improvement with respect to the baseline results. The recalls on tokyo-xs increase by more than 5% and those on tokyo-night by more than 10%. It is interesting to notice that the recalls are slightly lower in the datasets that only contain day images.

B. Data Augmentation for Perspective Changes & Occlusions

In this second data augmentation phase, we perform some transformations to increase robustness toward perspective changes and random occlusions, as shown in Fig. III. Inter alia, we choose to adopt the following functions provided by PyTorch's library:

1) *RandomPerspective*: we use this to perform a random perspective transformation of the given image, based on a probability passed by the user. In our case we used a probability value of 0.7 as default. That value is possibly editable as desired by the user. Furthermore, this transformation offers the possibility to decide the fill value. Namely, the pixel fill value for the area outside the transformed image. By default, we set it at one, changing the black padding into white.
2) *CenterCrop*: in order to remove the padding created by the *RandomPerspective* we perform another transformation. *CenterCrop* cuts the given image at the center, with the size passed as parameter.
3) *RandomErasing*: this transformation randomly selects a rectangle region in a torch tensor and erases its pixels.

Results obtained are shown in Table III: the tests are conducted on a trained model over *Pitts30k* dataset, with 3 ephocs, a probaility of performing the transformation of 70%, and comparing results between the two different feature aggregators. As can be seen, slightly but interesting improvements are reached among experiments. In particular, combining the perspective change with the occlusion transformation seems to work better and obtain higher recalls.

C. Ensembles

As explained above in the methodology section, we perform the two experiments: simple addition of descriptors, obtained from different models (AlexNet + ResNet + Aggregation layer), and element-wise division by scalar model's descriptors.

- *Addition of descriptors*: the first approach we experiment is the tensor addition. Here, we simply add obtained descriptors from the aggregation layer, after computing feature maps of two different backbones.
- *Division by scalar model's descriptors*: in this experiment we first compute the feature maps from the two backbones. At this point, we feed the aggregation layers and sum the results. We obtain a common matrix which is then divided element-wise by 2. This handing is needed to get average values of descriptors from two different backbones.

The experiment's results are shown in Table IV. The performance of the new method does not improve the results of the vanilla model. However, we can observe that, among

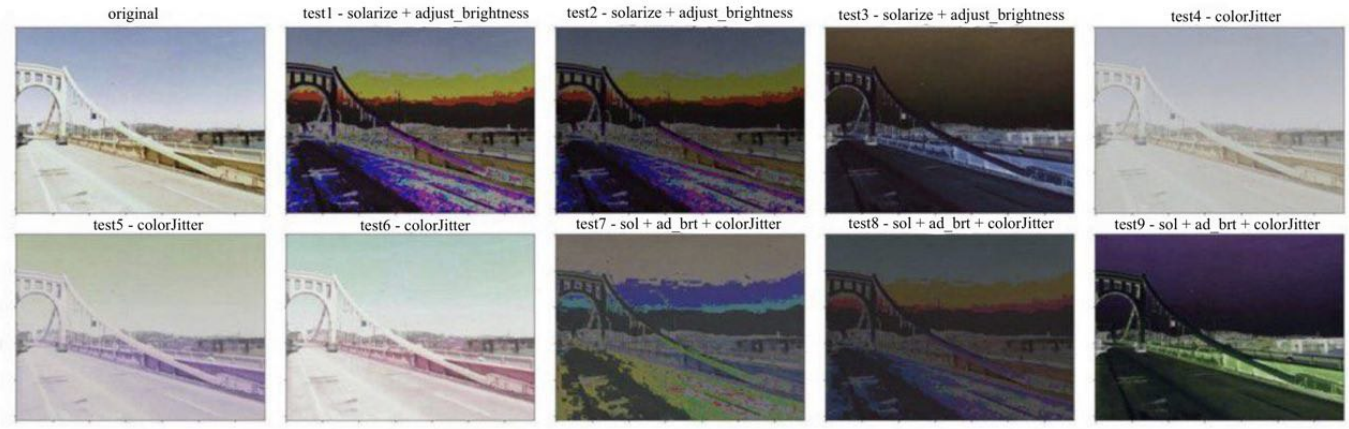

Fig. 4: **'Smart' Data Augmentation for night domain.** This picture shows on the right-upper corner the original image, while the others are the result of the cited transformations.

		pitts30k		sf-xs		tokyo-xs		tokyo-night	
'Smart' Data Augmentation	Aggregation	R@1	R@5	R@1	R@5	R@1	R@5	R@1	R@5
Yes	GeM	61.6	80.8	4.2	11.4	24.1	41.3	13.3	28.6
	NetVLAD	80.8	90.8	15.3	24.2	55.9	72.4	**31.4**	**46.7**
No (baseline)	GeM	73.5	87.4	10.0	21.6	33.0	48.6	14.3	28.6
	NetVLAD	**84.8**	**92.3**	**25.7**	**40.6**	**56.2**	**70.2**	21.0	43.8

TABLE I: This table shows the results of the **Smart Data Augmentation extension**. Although results seems to decrease among other datasets, the recalls show increasing results for the target dataset: *tokyo-night*.

		Pitts30k		sf-xs		tokyo-xs		tokyo-night	
Train Dataset	Aggregation	R@1	R@5	R@1	R@5	R@1	R@5	R@1	R@5
Pitts30k	GeM	73.5	87.4	10.0	21.6	33.0	48.6	14.3	28.6
	NetVLAD	**84.4**	**92.3**	**25.7**	**40.6**	56.2	70.2	21.0	43.8
Pitts30k_night	GeM	49.6	70.0	5.8	13.2	24.4	40.0	12.4	27.6
	NetVLAD	63.5	79.1	12.4	19.5	43.5	60.0	19.0	26.7
Pitts30k_dn	GeM	68.8	85.7	6.5	14.0	27.0	45.7	14.3	26.7
	NetVLAD	82.2	91.4	14.7	23.3	**61.3**	**78.1**	**39.0**	**58.1**

TABLE II: This table shows the recalls obtained using for the training process the datasets built with **synthetically generate images**. The training performed using Pitts30k_dn improves sensibly the recalls on the targeted datasets while slightly worsening on the others. The training on Pitts30k_night does not bring any relevant improvements.

the experiment results, GeM's performance comes better than NetVLAD aggregation, which means using this network ensembling makes NetVlad's descriptors less representative.

D. Multi-scale Testing

We analyze the performance of the multi-scale testing and compare them with the baseline model's tests. In this case, the model was trained for 5 epochs. The results are shown in Table V. The concatenation method achieves good performance and slightly improves recalls on both Pitts30k and sf-xs. The *naive* average method instead does not bring any improvement with respect to the original model.

V. Conclusion

In this paper we present some milestones techniques to address some issues regarding the VG task, showing how they can influence performance and how they can be implemented along the baseline's system. This work adds to the original benchmark some interesting extensions, which allow to increase the range of possible components to experiment within a VG architecture.

Part of the extensions we implemented bring an important help in terms of improvement in specific datasets. The synthetic image generation, for example, substantially increases the baseline results in tokyo and tokyo-night. The multiscale

		Pitts30k		sf-xs		tokyo-xs		tokyo-night	
Data Augmentation	Aggregation	R@1	R@5	R@1	R@5	R@1	R@5	R@1	R@5
Occlusion	GeM	73.5	87.6	9.7	20.0	32.1	48.9	14.3	25.7
	NetVLAD	57.5	77.4	21.1	32.5	56.8	**76.1**	28.6	43.8
Perspective Change ”fill”	GeM	22.7	45.2	2.4	6.3	12.1	24.4	5.7	16.2
	NetVLAD	84.2	**92.5**	19.4	28.0	**61.6**	73.3	**37.1**	46.7
Perspective Change ”crop”	GeM	69.9	86.0	8.0	16.4	27.0	45.7	7.6	21.9
	NetVLAD	84.6	92.4	18.6	29.0	57.8	70.8	26.7	41.0
Perspective Change ”crop” + Occlusion	GeM	69.3	85.4	9.9	19.1	31.4	49.2	11.4	24.8
	NetVLAD	84.6	**92.5**	22.6	**34.6**	59.4	74.0	33.3	**47.6**
Perspective Change ”fill” + Occlusion	GeM	69.3	85.4	9.9	19.1	31.4	49.2	11.4	24.8
	NetVLAD	84.6	**92.5**	22.6	**34.6**	59.4	74.0	33.3	**47.6**
Baseline	GeM	73.5	87.4	10.0	21.6	33.0	48.6	14.3	28.6
	NetVLAD	**84.8**	92.3	**25.7**	40.6	56.2	70.2	21.0	43.8

TABLE III: This table shows the recalls obtained by the Data Augmentation experiments. Relevant results are obtained combining occlusion and both perspective change extensions, which provide very similar results.

Fig. 5: **Data Augmentation for Perspective Changes and Occlusions** This picture shows some samples of the data augmentation performed: form top to down, occlusion, perspective change and crop.

		Pitts30k		sf-xs		tokyo		tokyo-night	
Ensembling	Aggregation	R@1	R@5	R@1	R@5	R@1	R@5	R@1	R@5
Addition(Resnet+AlexNet)	GeM	71.3	86.2	11.2	20.1	27.0	45.1	4.8	14.3
	NetVLAD	85.7	**94.8**	5.1	10.9	18.1	33.7	2.9	10.5
Division(Resnet+AlexNet)	GeM	72.2	87.5	5.2	11	10.5	24.4	7.6	13.3
	NetVLAD	**86.3**	94.6	5.1	10.9	18.1	33.7	2.9	10.5
No (Baseline)	GeM	73.5	87.4	10.0	21.6	33.0	48.6	14.3	28.6
	NetVLAD	84.8	92.3	**25.7**	**40.6**	**56.2**	**70.2**	**21.0**	**43.8**

TABLE IV: The result shown in this table indicate an **ensemble** learning performed method. GeM's descriptors while ensembling are more representative in the recall metric than NetVlad's.

		Pitts30k		sf-xs		tokyo		tokyo-night	
Multiscale	Aggregation	R@1	R@5	R@1	R@5	R@1	R@5	R@1	R@5
No (Baseline)	GeM	75.8	88.7	13.4	25.4	35.2	49.8	14.3	28.6
	NetVLAD	85.7	92.8	34.0	49.3	59.0	**74.0**	25.7	**47.6**
Concatenation	GeM	76.6	88.9	14.6	25.9	33.0	44.8	9.5	23.8
	NetVLAD	**85.9**	**93.0**	**37.0**	**51.5**	**60.6**	70.2	**29.5**	42.9
Average	GeM	68.8	85.2	6.8	15.3	16.5	29.2	4.8	10.5
	NetVLAD	75.4	88.8	14.4	25.3	27.6	46.7	7.6	21.9

TABLE V: The result shown in this table indicate that a **multiscale** pyramid can be useful to improve the baseline.Almost every recall using the concatenation method improves the given benchmark.

experiment brings some minor but stable improvements with the concatenation method in almost every dataset.

Although our proposal is unable to outperform the original baseline in some parts of the experiment, we are confident that with a deeper tuning of the possible parameters and combinations, and with higher resources, important achievements can be reached through these techniques. With regard to the multiscale testing with the average method, a new and more appropriate average could be defined and lead to a more significant description of the image.

Finally it is relevant to notice how the best results are consistently obtained using the NetVLAD aggregation layer.

For more details, we invite readers to consult the following repository where the code is available: https://github.com/arcangeloC-137/deep_learning_techniques_for_visual_place_recognition.

REFERENCES

[1] G. Berton, R. Mereu, G. Trivigno, C. Masone, G. Csurka, T. Sattler, and B. Caputo, "Deep visual geo-localization benchmark," 2022. [Online]. Available: https://arxiv.org/abs/2204.03444

[2] M.-Y. Liu, T. Breuel, and J. Kautz, "Unsupervised image-to-image translation networks," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017.

[3] S. Hausler, S. Garg, M. Xu, M. Milford, and T. Fischer, "Patch-netvlad: Multi-scale fusion of locally-global descriptors for place recognition," 2021.

[4] J. Mao, X. Hu, X. He, L. Zhang, L. Wu, and M. J. Milford, "Learning to fuse multiscale features for visual place recognition," *IEEE Access*, vol. 7, pp. 5723–5735, 2019.

[5] A. Anoosheh, T. Sattler, R. Timofte, M. Pollefeys, and L. V. Gool, "Night-to-day image translation for retrieval-based localization," pp. 5958–5964, 2019.

[6] A. Relja, G. Petr, T. Akihiko, and et al., "Netvlad: Cnn architecture for weakly supervised place recognition," 2016.

[7] R. Filip, T. Giorgios, and C. Ondrej, "Fine-tuning cnn image retrival with no human annotation," 2018.

[8] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," 2013.

[9] Y. Yu, Z. Gong, P. Zhong, and J. Shan, "Unsupervised representation learning with deep convolutional neural network for remote sensing images," in *Image and Graphics*, Y. Zhao, X. Kong, and D. Taubman, Eds. Cham: Springer International Publishing, 2017, pp. 97–108.

[10] A. Khaliq, M. Milford, and S. Garg, "MultiRes-NetVLAD: Augmenting place recognition training with low-resolution imagery," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 3882–3889, apr 2022.