Exploring Visual Place Recognition: a survey on methods and approaches

Mansillo Daniele
Politecnico Di Torino
s319297@studenti.polito.it

Picone Umberto Emmanuele
Politecnico Di Torino
s296496@studenti.polito.it

Varriale Luca
Politecnico Di Torino
s300795@studenti.polito.it

Abstract

Visual geo-localization (VG), also called Visual Place Recognition (VPR), consists of determining the location depicted in a query image by referring to a database of reference geo-tagged images. The N-top retrieved images, with their geo-tag (typically GPS coordinates), provide the hypothesis for the query's geographical location. In this paper we aim to investigate how the combination of different miners, loss functions and aggregator layers perform on challenging datasets. We also introduce a brand new aggregator layer (ConvGeM) by leveraging key features of the state-of-the-art aggregators. We then remark and formalize how the domain and the dataset heavily influence the choice of the "right" model and which components better fit in specific contexts. Our code and model are available at `https://github.com/danielemansillo/Simple_VPR_codebase`.

1. Introduction

Visual place recognition (VPR) refers to the capability of a system to determine the geographical location of where a photo was taken. It plays a pivotal role in various domains, enabling systems to determine whether a given query image corresponds to a previously visited location. VPR aims to bridge the gap between visual perception and spatial understanding by leveraging the visual content of images to establish associations with specific places.

The emergence of deep learning and convolutional neural networks (CNNs) revolutionized computer vision tasks, including image classification, object detection, and semantic segmentation.

Consequently, researchers have proposed training CNNs directly for place recognition [4] [12] [14], yielding remarkable success in large-scale benchmark evaluations [20]. The strategy entails employing a neural network to project the images into an embedding space that accurately depicts the similarity of their locations, and then use such similarity for the retrieval.

In recent years bigger and more specific datasets, such as Tokyo [18], Mapillary [20], San Francisco [5] and GSV-Cities [1], were developed in order to tackle a new challenge of this field: the domain adaptation. Recognizing an image after a domain shift such as season change, different light condition, different angle of the picture or human activity on the area can be really challenging for classic models.

This led to an approach shift: using these challenging datasets also to train the model rather than just for the execution of the retrieval [5]. New training techniques were developed to face the challenge [5] exploring the influence of aggregator layers in feature extraction [1] [2] and how miners and loss techniques previously employed in classical image recognition tasks do not perform adequately in this domain shift case.

Contribution: In this paper, relying on the state-of-the-art datasets for Visual Place Recognition (San Francisco [5], Tokyo [18], GSV-Cities [1]), we aim to explore different possibilities concerning the combination of mining, loss and aggregation techniques to gain insight into the optimal synergies.

We then develop and present a brand new aggregator, the ConvGeM, which employs key features from the existing state-of-the-art aggregators such as CosPlace [5] and ConvAP [1] and some interesting concepts from the Generalized-Mean Pooling [16] which manages to generate highly efficient and compact representations while maintaing remarkable performance of the domain adaptation task.

2. Related Works

The aim of Visual Place Recognition task is to determine the location or place depicted in an image by comparing it with a reference database of visual data. This task of recognition encompasses not only the identification of the place but also the estimation of the perspective, orientation, and additional contextual particulars associated with the image. This stimulates development of novel techniques for robust algorithms and retrieval strategies. Convolutional Neural Networks are typically employed as the backbone network in Visual Place Recognition systems: they are trained on large-scale image datasets, such as ImageNet [13], and they are adapted to VPR using a a trainable aggregation layer. Significant contributions to the field of Visual Place Recognition have been achieved by Arandjelovic *et al*. [3] introducing an end-to-end trainable version of the VLAD descriptor and incorporating it into a CNN [4]. Thanks to its features aggregation into a single descriptor, NetVLAD has been used as a baseline for further advancements in the field. For istance, Kim *et al*. [12] projected a method for reweighting local features based on their contextual information, reducing the influence of less informative features. In another work Yu *et al*. [21] introduced SPE-VLAD, an enhanced version of the NetVLAD algorithm integrating a weighted triplet loss and a spatial pyramid structure, which divides the image into multiple regions extracting local features from each specific region. Thereafter Zhan *et al*. [22] proposed Gated NetVLAD, incorporating attention weights into the computation of residuals, through the gating mechanism: this ensured an improvement of the discriminative power of the NetVLAD, leading to better performance in VPR. One important factor that hindered the progress of new research [9] is the lack of large, various and public available datasets, with accurate ground truth. VPR suffers from this bottleneck, so to address this challenge several new image datasets providing wider variety of perceptual changes and geographic coverage have been introduced (for example GSV-Cities [1] and Mapillary Street-Level Sequences [20]). Other works instead focused on aggregation techniques: for example Radenovic *et al*. [16] gave a valuable contribution to the field of CNN-based image retrieval. In particular, the implementation in this work of the Generalized Mean (GeM), a learnable global pooling, will pave the way for various future works. Indeed recently Berton *et al*. [5] proposed CosPlace, a lightweight aggregation technique obtained combining GeM pooling with a linear projection layer, leading to an enhancement of the overall performance of the model. Recently, new cutting-edge aggregation techniques which make use of convolutions have been developed, such as ConvAP [1] and MixVPR [2]. The latter, in particular, employs a stack of isotropic MLP blocks to incorporate global relationships into each individual feature map, instead of focusing on local features. These techniques have proven to be very reliable especially in the domain shift use-case.

3. Methodology

In the paragraphs below, we present a comprehensive analysis of the methods we explored for the survey.

3.1. Loss function

Loss functions play a pivotal role in training deep learning models. The choice of an appropriate loss function is crucial, because it directly impacts the learning process and the ability of a model to generalize to unseen data.

Contrastive loss: This function [10] penalizes similar image pairs if they are far apart in feature space and penalizes dissimilar pairs if they are too close, as follows:

$$\mathcal{L}_C = (1 - \mathcal{I}_{ij})[S_{ij} - m]^+ - \mathcal{I}_{ij} S_{ij} \tag{1}$$

where $\mathcal{I}_{ij}$ is the binary label indicating whether the pair of samples are similar ($\mathcal{I}_{ij} = 1$) or dissimilar ($\mathcal{I}_{ij} = 0$). The margin m is an hyperparameter that avoids not needed optimization for negative pairs already dissimilar enough.

Multi-Similarity loss: This function [19] implements a new pair weighting scheme using two iterative steps: mining and weighting. Its formulation involves logarithmic scaled sums of similarities as follows:

$$\begin{aligned}\mathcal{L}_{MS} = \frac{1}{N} \sum_{i=1}^{N} \Bigg\{ \frac{1}{\alpha} log \Bigg[1 + \sum_{j \in \mathcal{P}_i} e^{-\alpha(S_{ij} - m)} \Bigg] \\ + \frac{1}{\beta} log \Bigg[1 + \sum_{k \in \mathcal{N}_i} e^{-\beta(S_{ik} - m)} \Bigg] \Bigg\}\end{aligned} \tag{2}$$

where elements in $\mathcal{P}_i$ and $\mathcal{N}_i$ represents the set of indices that form a positive or a negative pair with the selected istance, while α, β and m are hyperparameter. With respect to the Contrastive loss, which focuses on pairwise comparisons, Multi-Similarity loss is able to consider multiple similarities and in this way to capture the overall distribution of similarities within the batch.

FastAp loss: Most common loss functions used to train the model compare similarities between pairs or triplets of examples in the training set. However, as the size of the training set increases, the number of possible pairs or triplets grows exponentially, leading to a significant increase in the number of training instances that need to be considered. The implementation of FastAP loss [7] avoids a high-order explosion of the training set, which is a common problem in existing losses defined on pairs or triplets. It optimizes the rank-based Average Precision measure over ranked lists of examples, using an approximation based on the probabilistic interpretation of AP as the area under precision-recall curve.

3.2. Aggregation module

The aggregation function receives as input the feature maps extracted from the last layer of the ResNet 18 or, in the case of the MixVPR, from an intermediate layer. The aggregator takes these input features and performs a fusion or aggregation operation to generate a condensed representation that can effectively capture the discriminative information necessary for place recognition tasks.

Aggregator layers help achieve spatial invariance, which means the network becomes less sensitive to the exact location of features. By downsampling and summarizing the input, pooling layers make the subsequent layers more robust to variations in position, scale, or orientation of the features.

Adaptive Average Pooling: This aggregator dynamically adapts the pooling size based on the input's spatial dimensions. In adaptive average pooling, the input tensor is divided into a grid of equal-sized regions (specifically $s1 \times s2$ equal subregions), and the average value within each region is computed. The number of regions and their sizes are determined based on the desired output size. This allows for flexible pooling operations that can accommodate input tensors of varying sizes.

GeM (Generalized-mean pooling): [16] This aggregator exploits the generalized mean:

$$\mathbf{f} = \left[\mathrm{f}_1^{(g)} \dots \mathrm{f}_k^{(g)} \dots \mathrm{f}_K^{(g)}\right] \top, \mathrm{f}_k^{(g)} = \left(\frac{1}{|\mathcal{X}_k|} \sum_{x \in \mathcal{X}_k} x^{p_k}\right)^{\frac{1}{p_k}} \tag{3}$$

For instance average pooling is a special case of GeM pooling with $p_k = 1$. The pooling is applied to each feature map, so the resulting feature vector consists of a single value per feature map. The resulting feature vector could be flattened and L_2 normalized in order to be used as a solo aggregator for our model.

CosPlace: [5] The CosPlace aggregator combines GeM with a linear projection layer (indicated by the symbol σ):

$$\mathbf{z}_i = \sigma(\mathsf{GeM}_{p_k}(\mathbf{F}_i)) \tag{4}$$

CosPlace aggregator is very lightweight but has proven to achieve very high accuracy, specifically in the domain adaptation use-case.

Conv-AP: [1] The initial step of the aggregator performs a channel-wise weighted pooling on the input feature map $\mathbf{F} \in \mathbb{R}^{h \times w \times c}$. This pooling operation linearly projects each spatial descriptor $\mathbf{f}_{ij} \in \mathbf{F}$ into a compact d-dimensional representation space, achieving a dimensionality reduction along the channel dimension. This is accomplished by employing, a 1×1 convolution with parameters $\mathbf{W} \in \mathbb{R}^{d \times c \times 1 \times 1}$ such as:

$$\mathbf{F}' = \mathbf{W} \circledast \mathbf{F} = \mathsf{Conv}_{1 \times 1}(\mathbf{F}) \tag{5}$$

where $\circledast$ is the convolution operation and $\mathbf{F}' \in \mathbb{R}^{h \times w \times f}$ is the resulting feature map with depth d.

In the second step, the spatial dimensionality of F' is decreased using adaptive average pooling (AAP). The resulting output features have then dimension $s1 \times s2 \times d$. Summarizing all the steps so far and assigning the resulting representation to $\mathbf{z}_i$:

$$\mathbf{z}_i = \mathsf{AAP}_{s_1 \times s_2}\left(\mathsf{Conv}_{1 \times 1}(\mathbf{F}_i)\right) \tag{6}$$

The resulting representation $\mathbf{z}_i \in \mathbb{R}^{s_1 \times s_2 \times d}$ is then flattened and L_2 normalized in order to be used as a solo aggregator for our model.

Conv-GeM: Taking inspiration from the Conv-AP and the GeM techniques above we decided to generalize the behaviour of the Conv-AP aggregator by replacing the AAP with a GeM pooling and introducing a linear transformation, indicated here by the symbol σ. Using the same symbols used above for the Conv-AP and indicating with GeM_{p_k} the p_k-th grade generalized mean:

$$\mathbf{z}_i = \sigma\left(\mathsf{GeM}_{p_k}\left(\mathsf{Conv}_{1 \times 1}(\mathbf{F}_i)\right)\right) \tag{7}$$

The resulting representation $\mathbf{z}_i \in \mathbb{R}^{1 \times 1 \times d}$ is also then flattened and L_2 normalized in order to be used as a solo aggregator for our model. A limitation of our aggregator at the moment is that we can only obtain a 1×1 representation from the GeM layer instead of the $s_1 \times s_2$ dimension that's easily obtained by the AAP.

MixVPR: [2] The architecture of the MixVPR is very different from the other aggregators shown above. The MixVPR aims to learn a global compact representations that integrates features in a holistic way. As a first step the Convolutional neural network is cropped, in our specific case we removed from the ResNet18 architecture the last two convolutional layers, the average pooling and the linear projection. Each 2d feature map, which we will call $X^i \in \mathbf{F}$, from this intermediate layer is then flattened and fed to the *Feature-Mixer* layer which is composed by a cascade of L MLP blocks of identical structure. In particular, each of the L MLP blocks that make up the *Feature-Mixer* presents the following structure:

$$X^i \leftarrow \mathbf{W}_2(\sigma(\mathbf{W}_1 X^i)) + X^i, \quad i = \{i, \dots, c\} \tag{8}$$

Where $\mathbf{W}_1$ and $\mathbf{W}_2$ are the weights of two fully connected layers that compose the MLP and σ is a non-linearity (ReLU in this specific case). The input X^i of the MLP is then added back to the resulting projection in a skip connection.

For a given input $\mathbf{F} \in \mathbb{R}^{c \times n}$, the Feature Mixer (FM) generates an output $\mathbf{Z} \in \mathbb{R}^{c \times n}$ of the same shape (due to its isotropic architecture), which is then fed to a second Feature-Mixer and so on recursively until L blocks are executed:

$$\mathbf{Z} = FM_L(FM_{L-1}(\dots FM_1(\mathbf{F}))) \tag{9}$$

To reduce the dimensionality of $\mathbf{Z}$ from $\mathbb{R}^{c \times n}$ to $\mathbb{R}^{d \times n}$ a depth-wise convolution is applied:

$$\mathbf{Z}' = \mathbf{W}_d(\mathit{Transpose}(\mathbf{Z})) \tag{10}$$

where $\mathbf{W}_d$ are the weights of the fully connected layer. Then a row-wise convolution is applied to $\mathbf{Z}'$ to reduce its dimension from $\mathbb{R}^{d \times n}$ to $\mathbb{R}^{d \times r}$:

$$\mathbf{O} = \mathbf{W}_r(\mathit{Transpose}(\mathbf{Z}')) \tag{11}$$

where $\mathbf{W}_r$ are the weights of this fully connected layer. The final output $\mathbf{O}$ is then flattened and L_2-normalized.

3.3. Miners

The selection of informative positive and negative pairs within a batch is fundamental for robust feature learning. An informative pair refers to a pair that generates a significant loss value, so an effective mining strategy not only enhances performance but also accelerates the training process [11]. In particular, if the pairs are excessively easy, the network fails to develop the ability to properly distinguish between the images, as proven in [8]. Conversely, allowing the network to learn excessive challenging examples may result in overfitting [15] and encountering bad local minima [17]. Furthermore, there exist two main methods for selecting the pairs. Offline mining refers to the process of selecting a predetermined set of training data before the learning phase begins. This method typically involves gathering a substantial dataset of images accompanied by geo-location information and annotating them with labels or coordinates. However, offline mining comes with a significant computational cost. It requires pausing the training procedure to compute and store representations of a large subset of images in a cache memory. In contrast, online mining takes a different approach by dynamically selecting and incorporating challenging negative samples into the training process based on the model's performance during training. It does not imply a lot of computations, for this reason we have chosen online mining, analyzing the application of MultiSimilarityMiner and comparing its performance against Pair and Triplet miners.

The **PairMarginMiner** selects positive and negative pairs based on the violation of specified margins. There are two margins involved: the positive one represents the threshold below which positive pairs are considered, while the negative margin represents the threshold above which negative pairs are considered.

The **TripletMarginMiner**, instead, selects triplets based on a margin, which is the difference between the distance of the anchor to the positive sample and the distance of the anchor to the negative sample. It operates based on different types of triplets. In particular, in our experiments the considered triplets are "semihard": used for the first time in [17], they includes all the triplets that violate the margin and where the negative sample is further away from the anchor than the positive sample.

Finally, the **MultisimilarityMiner**, chooses negative pairs if their similarity exceeds the similarity of the hardest positive pair by a certain little margin, epsilon. Similarly, it selects positive pairs if their similarity is less than the similarity of the hardest negative pair plus the margin (epsilon).

4. Experimental Results

In the following section, we firstly outline the key features of the employed datasets. Then, we provide a comparative analysis of the results obtained through iterative implementations of the previously introduced methods.

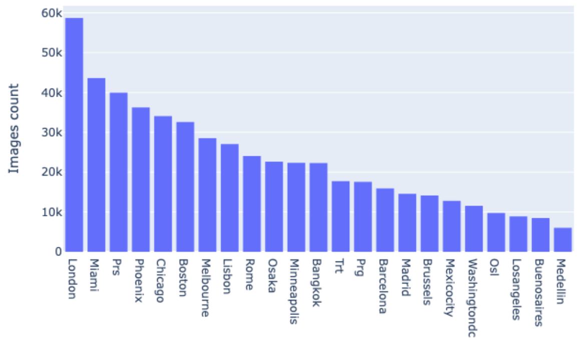

Figure 1. Images distribution by city in GSV-Cities dataset.

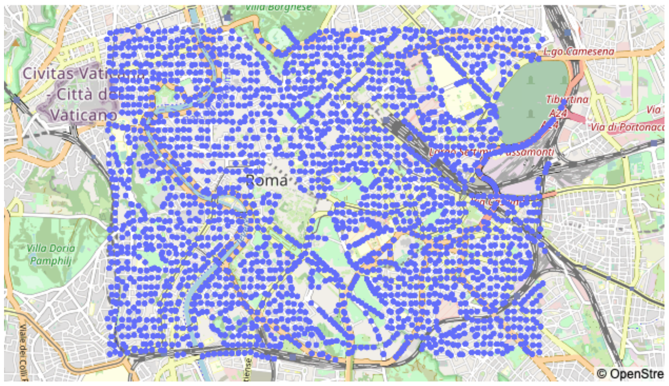

Figure 2. Locations captured in the city of Rome. Each location is provided with a minimum of four images.

4.1. Datasets and Experimental Setting

We leveraged the GSV-Cities [1] dataset, which stands out for its wide geographical coverage. Moreover, it offers highly accurate ground truth annotations and it spans over a 14-year period, encompassing more than 40 cities across all continents. The dataset comprises approximately 530,000 images, capturing over 62,000 different places located in various cities worldwide. As we can see from the Fig. 1, the representation of cities is not uniform or equally distributed. Notably, all places in the dataset are physically distant from each other, with a minimum separation of 100 meters between any pair of places.

To fit within our computational constraints we utilized images with lower resolution with respect to the original dataset.

In order to assess the model's performance in different domains, we conducted tests using reduced portions of other image geolocation datasets, namely Tokyo-XS [18]

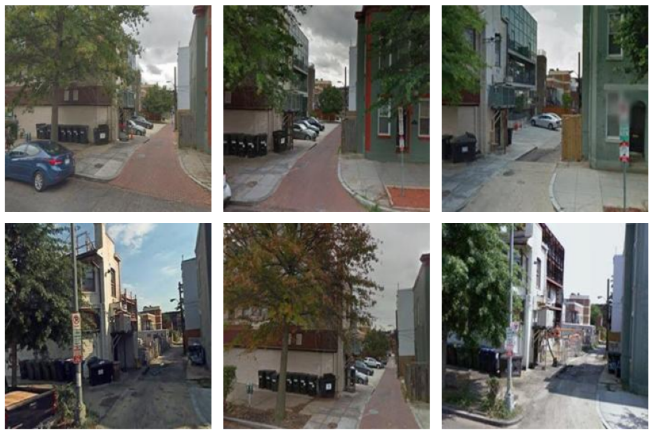

Figure 3. A challenging example from GSV-cities where the same place varies due to human activity, angle, season and light change.

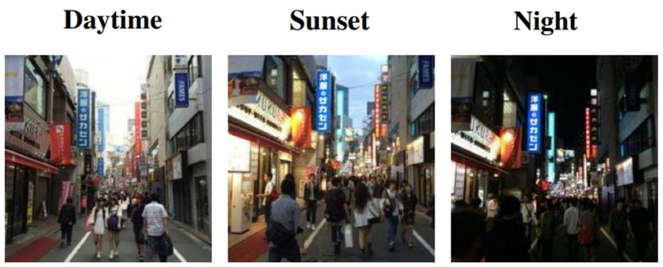

Figure 4. Every image in the Tokyo Dataset is captured at different times of day.

and SF-XS [6]. The SF dataset encompasses the entire city, offering dense coverage and temporal variability, while the Tokyo dataset focuses on a small geographical area and exhibits diverse changes in appearance, such as variations in illumination (day/night), seasonal transformations and aging effects. Moreover, there are provided multiple perspectives and a continuous temporal coverage, enabling the algorithm to understand different spatial relationships and perspectives. Also, the queries of Tokyo-XS are split into three equally sized sets (day, sunset and night) in order to analyze how the model performs with different time shifts (Fig. 4).

We initiated our experimentation by employing Resnet-18 as our baseline model by training it over 20 epochs. Additionally, we adopted the top-1 and top-5 accuracy metric on the test sets to present our results. Furthermore, our entire methodology was implemented with PyTorch, using Google Colab as our development environment and exploiting its GPU support.

Loss	Pooling	SF-XS		Tokyo-XS	
		R@1	R@5	R@1	R@5
Contrastive	Average	32.9	46.3	42.9	63.2
Contrastive	CosPlace	33.9	50.1	51.4	69.5
Contrastive	ConvAp	35.9	50.3	52.4	73.0
Multi Similarity	Average	30.0	45.0	43.2	59.0
Multi Similarity	CosPlace	33.5	49.4	53.3	71.1
Multi Similarity	ConvAp	**36.3**	**51.5**	**57.1**	**75.9**
FastAp	Average	26.1	43.2	40.7	57.2
FastAp	CosPlace	26.9	42.0	46.3	63.5
FastAp	ConvAp	32.6	49.0	48.3	67.0

Table 1. Results obtained using different combinations of loss functions and aggregation modules.

4.2. Comparing different losses and aggregation modules

We conducted a comparative analysis examining the behaviour of the model using several metric learning loss functions employed in training of VPR tasks. Moreover, in order to investigate potential improvements achievable by combining different loss functions and aggregators, we analyzed various possible couplings.

As shown in Table 1, performance is significantly affected by different combinations. Indeed certain combinations may enhance the model's ability to capture complex patterns and relationships that might be challenging for a single loss or aggregation module to capture independently. The choice of the Multisimilarity Loss, in combination to ConvAP, achieves the best result in both test datasets. It can be explained by the fact that this loss metric allows the selection of more informative and discriminative pairs, implementing a dynamic weighting that leads the model to focus on more challenging examples. In addition to this elaborate pair mining and weighting strategy, ConvAP aggregator is characterized by adaptive pooling which enables a dynamically adaptation to the characteristics of the input data, focusing on relevant features of images. High values of recall on Tokyo dataset could be justified by the adaptation properties of this model, which is able to address variations of illuminations and presence of occlusions that characterize Tokyo images. Moreover, it is important to emphasize that these results need to be interpreted in the context of the available training dataset size. For instance, as demonstrated by Berton *et al*. [5], Cosplace is specifically designed for large-scale datasets, and its performance tends to decrease when applied to smaller datasets, indicating limited downward scalability. Therefore, with larger scale dataset, higher recall could have been achieved.

Loss	Pooling	SF-XS		Tokyo-XS	
		R@1	R@5	R@1	R@5
Contrastive	Average Pooling	21.5	34.9	33.0	45.1
Contrastive	CosPlace	30.1	46.7	49.9	68.6
Contrastive	ConvAp	**37.7**	**54.3**	54.3	70.2
Multi Similarity	Average	21.4	36.1	35.6	50.5
Multi Similarity	CosPlace	29.3	46.1	47.0	67.3
Multi Similarity	ConvAp	33.6	49.8	**56.2**	**70.5**
FastAp	Average	20.1	34.6	39.0	58.7
FastAp	CosPlace	22.1	36.5	41.6	58.1
FastAp	ConvAp	28.5	44.1	49.8	66.3

Table 2. Results obtained implementing MultiSimilarity Miner on previous combinations of loss functions and aggregators.

Miner	Loss	SF-XS		Tokyo-XS	
		R@1	R@5	R@1	R@5
MultiSimilarity	Contrastive	37.7	**54.3**	54.3	70.2
MultiSimilarity	MultiSimilarity	33.6	49.8	56.2	70.5
MultiSimilarity	FastAP	28.5	44.1	49.9	66.3
Triplet	Contrastive	29.2	46.1	48.6	68.6
Triplet	MultiSimilarity	34.2	49.8	52.1	71.1
Triplet	FastAP	27.4	41.9	44.4	64.7
Pair	Contrastive	**38.0**	53.7	**55.6**	**74.6**
Pair	MultiSimilarity	34.2	50.7	54.9	75.9
Pair	FastAP	30.3	46.5	51.1	67.0

Table 3. Comparison of different mining strategies and loss functions using ConvAP as aggregator.

4.3. Mining strategy

To ensure optimal results, it is important to adopt a well-planned mining strategy. Without such strategy, a large amount of pair-wise samples could be generated, characterized by highly redundance making them uninformative: for this reason we avoid pure random sampling, that could have degraded model capability. Different mining functions are available, which, taking batch of n embeddings, return k pairs or triplets to be used for the loss evaluation. In our specific application we opted for online miners, that process data in real-time, handling changing and variations in environments, orientation and illuminations.

The aim of these analysis is to highlight how an appropriate combination of loss functions and pooling layers with the design of the online miner, can improve the accuracy, robustness, and efficiency of the model. Referring to other works [1], Multi-Similarity miner achieves great results in different frameworks. For this reason we conducted a preliminary evaluation implementing this miner in combination with previous coupling of loss functions and pooling layers, and all the results are shown in Table 2.

The choice of the loss function directly impacts the performance of the online miner because it guides the learning process and enables update of the internal model parameters of the miner, improving in this way recognition accuracy. Moreover also the pooling layer plays a crucial role, since it summarizes informations from the miner's outputs affecting model's performance in multiple ways. As it can be noticed, the implementation of MS-Miner in combination with the Contrastive loss and ConvAp produces a significant improvement in terms of accuracy (from 35.9 % to 37.7% for SF-XS, and from 52.4% to 54.3% for Tokyo-XS). It can be due to the synergy that arises between them. Conv-Ap dynamically adapts to the provided inputs with a selective focus on significant features capturing the most salient information while the Contrastive loss guides the learning process enforcing desired similarities: all this properties helps the online miner to select additional challenging samples.

However, by combining a miner and a loss function or aggregator, it is possible for the performance to deteriorate due to several reasons. In particular in most of cases the implementation of MS-miner produces a worsening of the accuracy: for example in the case of Fast-Ap loss, it could be attributed to the fact that it does not provide sufficient discrimination between positive and negative samples or fails to capture the desired similarity, making it difficult for the miner to distinguish between different places accurately. For this reason we decided to carry out further experimentation and comparative analysis which can help to identify the most effective combination and mitigate possible performance degradation.

In order to achieve a model with the highest possible level of accuracy, we deemed it better to conduct further analysis on models that seem to yield better results. From previous experiments, Conv-Ap, despite being coupled with various loss functions, outperforms all other combinations. This should not be so surprising, indeed as further explained by Ali-bey *et al*. [1] Conv-Ap surpasses many existing techniques. Therefore, as shown in Table 3, it could be interesting to investigate the effects of other types of miners coupled with this aggregator. In particular, the combination of Pair Margin Miner with the Contrastive loss leads to the temporary higher accuracy on SF-XS (38.0%). Since the Pair Margin Miner enhances the discriminative power of the model by explicitly considering the margins between positive and negative pairs, it seems to be more compatible with pairwise comparison adopted by Contrastive loss, and for this reason it produces a further improvement with respect to MS-Miner.

4.4. MixVPR

Afterwards, referring to the remarkable results achieved by Ali-bey *et al*. [2] we decided to further investigate the MixVPR aggregator and explore its potential in combination with different losses, verifying if the usage of the MS-miner could lead to further improvement; all the results are

Miner	Loss	SF-XS		Tokyo-XS	
		R@1	R@5	R@1	R@5
MultiSimilarity	MultiSimilarity	34.1	47.5	52.4	69.8
MultiSimilarity	FastAp	25.8	39.9	44.4	65.7
MultiSimilarity	Contrastive	34.3	49.1	55.6	72.7
NoMiner	Contrastive	37.5	49.3	54.3	71.4
NoMiner	FastAp	32.1	47.1	48.3	67.3
NoMiner	MultiSimilarity	**38.7**	**51.6**	**58.1**	**75.2**

Table 4. Results achieved by MIXVPR in combination with various loss functions and, eventually, with the usage of MS-miner.

Loss	Pooling	SF-XS		Tokyo-XS	
		R@1	R@5	R@1	R@5
MultiSimilarity	ConvGeM	30.8	46.4	51.7	70.0
Contrastive	ConvGeM	32.7	48.7	52.4	70.5
MultiSimilarity	CosPlace	**33.2**	**49.5**	**54.6**	**72.7**

Table 5. ConvGEM and CosPlace aggregator comparison.

reported in Table 4.

Due to its capability to aggregate features in a holistic way, MixVPR proves to have remarkable performances, especially in San Francisco dataset where the images are not strongly characterized by distinctive features or structures such as typical buildings and poles. Therefore, the skyline and repetitive structures, constitute an important signature of the place and MixVPR better manages to recognize tight differences. While, on the other hand, other techniques retrieve images of different places that are overly similar to the query. In addition, MultiSimilarity loss further improves the ability to distinguish between places with very little differences due to its design that aims to emphasize the difference between similar images, which however depict different places.

4.5. ConvGeM

In Tab. 5 we see how in our new aggregator the insertion of a convolutional layer, following the example of the ConvAP, does not improve performances.

The main reason to which we attribute this poor results obtained by the aggregator is that convolution operations can have a smoothing effect on image features and this can lead to a loss of relevant details. For example, if the goal is to recognize fine-grained objects, the presence of one further convolution might reduce the model's ability to capture fine details.

5. Qualitative comparison

The best performance during the previous analysis have been achieved by MixVPR with Multisimilarity as loss function. Fig. 5 shows two examples of challenging queries extracted from Tokyo-XS: these are images of locations taken at nighttime, characterized by poor lightining. It's possible to compare the best predictions obtained using MixVPR and the model trained using ConvAP as aggregator (both with Multisimilarity loss). These are the two combinations among the ones that reach highest level of accuracy, but with this further qualitative analysis it's possible to highlight the greater robustness of MixVPR in challeng-

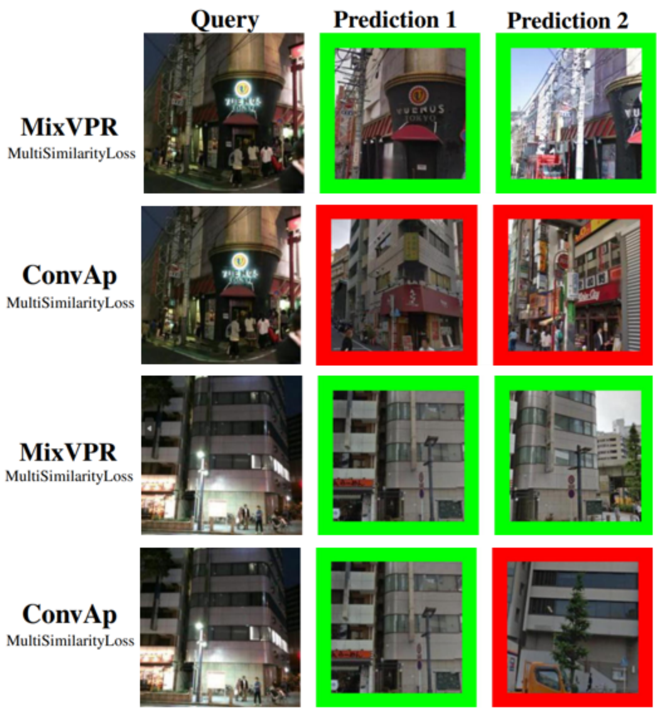

Figure 5. Comparison between MixVPR and ConvAP (using Multisimilarity loss) retrieval of challenging images with low lightning conditions.

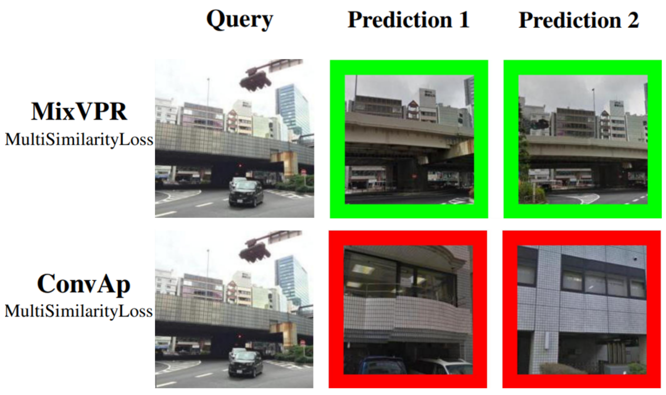

Figure 6. Comparison between MixVPR and ConvAP when an easily recognizable pattern (the geometric pattern, in this case) is present.

ing scenarios. This is a pivotal aspect in image retrieving because lightning variations occur multiple times throughout the day, and they are also influenced by weather conditions as well.
Although ConvAp doesn't encounter particular problems in retrieving the right references in the case of images taken during the day, in case of illumination changes sometimes it struggles to match them correctly. Instead MixVPR proves to be more reliable and robust in these extreme conditions. Moreover another important improvement to point out is the capacity to overcome eventual occlusions that can obstruct the reference image, as exemplified by the tree in the wrong matching of the second example.

Another serious problem for VPR techniques is the presence of repetitive pattern since different places may contain the same type of building or structure with the same layout or texture and this can fool the recognition system and induce a lot of false positives. As shown in Fig. 6, MixVPR has managed to retrieve the right reference, contrarily to ConvAP which instead retrieved images similar to the query depicting different locations.

6. Conclusions

In this work we surveyed different approaches to address the VP Recognition. In particular, we analyzed possible synergies that could arise by combining different aggregator modules, loss functions and mining strategies. By examining the various combinations, we discovered valuable insights into how these elements can work in harmony to enhance the performance and efficiency of Visual Place Recognition systems. In particular the best result in terms of accuracy is achieved by MixVPR, in combination with MultiSimilarityLoss. Moreover, we developed a brand new aggregator layer (ConvGeM) by leveraging key features of the state-of-the-art aggregators. Through this experimentation, we gained a deeper understanding of the complexities associated with VPR and identified some crucial factors that influence the overall performance of the system.

Our exploration in search of synergies among the different modules that make up a VPR model aims to pave the way for more powerful and accurate recognition models.

References

[1] Amar Ali-bey, Brahim Chaib-draa, and Philippe Giguère. Gsv-cities: Toward appropriate supervised visual place recognition. *Neurocomputing*, 513:194–203, 2022. 1, 2, 3, 4, 6

[2] Amar Ali-bey, Brahim Chaib-draa, and Philippe Giguère. Mixvpr: Feature mixing for visual place recognition, 2023. 1, 2, 3, 6

[3] Relja Arandjelovic and Andrew Zisserman. All about vlad. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1578–1585, 2013. 2

[4] Relja Arandjelović, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. Netvlad: Cnn architecture for weakly supervised place recognition, 2016. 1, 2

[5] Gabriele Berton, Carlo Masone, and Barbara Caputo. Rethinking visual geo-localization for large-scale applications. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4878–4888, 2022. 1, 2, 3, 5

[6] Gabriele Berton, Riccardo Mereu, Gabriele Trivigno, Carlo Masone, Gabriela Csurka, Torsten Sattler, and Barbara Caputo. Deep visual geo-localization benchmark. In *CVPR*, June 2022. 5

[7] Fatih Cakir, Kun He, Xide Xia, Brian Kulis, and Stan Sclaroff. Deep metric learning to rank. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1861–1870, 2019. 2

[8] Yin Cui, Feng Zhou, Yuanqing Lin, and Serge Belongie. Fine-grained categorization and dataset bootstrapping using deep metric learning with humans in the loop, 2016. 4

[9] Jose M Facil, Daniel Olid, Luis Montesano, and Javier Civera. Condition-invariant multi-view place recognition. *arXiv preprint arXiv:1902.09516*, 2019. 2

[10] R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742, 2006. 2

[11] Mahmut KAYA and Hasan Şakir BİLGE. Deep metric learning: A survey. *Symmetry*, 11(9), 2019. 4

[12] 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

[13] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012. 1

[14] Liu Liu, Hongdong Li, and Yuchao Dai. Stochastic attraction-repulsion embedding for large scale image localization, 2019. 1

[15] Filip Radenović, Giorgos Tolias, and Ondřej Chum. CNN image retrieval learns from BoW: Unsupervised fine-tuning with hard examples. In *ECCV*, 2016. 4

[16] Filip Radenović, Giorgos Tolias, and Ondřej Chum. Fine-tuning cnn image retrieval with no human annotation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(7):1655–1668, 2019. 1, 2, 3

[17] Florian Schroff, Dmitry Kalenichenko, and James Philbin. FaceNet: A unified embedding for face recognition and clustering. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, jun 2015. 4

[18] Akihiko Torii, Relja Arandjelovic, Josef Sivic, Masatoshi Okutomi, and Tomas Pajdla. 24/7 place recognition by view synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015. 1, 4

[19] Xun Wang, Xintong Han, Weilin Huang, Dengke Dong, and Matthew R Scott. Multi-similarity loss with general pair weighting for deep metric learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5022–5030, 2019. 2

[20] Frederik Warburg, Søren Hauberg, Manuel López-Antequera, Pau Gargallo, Yubin Kuang, and Javier Civera. Mapillary street-level sequences: A dataset for lifelong place recognition. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2623–2632, 2020. 1, 2

[21] Jun Yu, Chaoyang Zhu, Jian Zhang, Qingming Huang, and Dacheng Tao. Spatial pyramid-enhanced netvlad with weighted triplet loss for place recognition. *IEEE Transactions on Neural Networks and Learning Systems*, 31(2):661–674, 2020. 2

[22] Jian Zhang, Yunyin Cao, and Qun Wu. Vector of locally and adaptively aggregated descriptors for image feature representation. *Pattern Recognition*, 116:107952, 2021. 2