Visual Place Recognition: Experiments

Luca Catalano
Politecnico di Torino
s308658@studenti.polito.it

Razvan Nicolae Constantinescu
Politecnico di Torino
s319921@studenti.polito.it

Andrea Giordano
Politecnico di Torino
s307764@studenti.polito.it

Abstract

Visual Place Recognition (VPR), also known as Visual geolocalization, is a field of research that aims to determine the geographic location of an image or a video frame. It plays a main role in various applications, including augmented reality, autonomous navigation, urban planning, and tourism. This article presents a review of the most famous researches on visual place recognition. Its purpose is to gain insight into some specific uses of particular features that can help improve or at least better understand how the underlying frameworks and design choices work in a VPR pipeline. We run a large suite of experiments that provide criteria for choosing architectures, losses and optimizers to improve the outcome. The dataset and code are available for research purposes at https://github.com/LucaCatalano13/VPR.

1. Introduction

Visual place recognition is a fundamental task in computer vision and aims to identify and localize a specific place or scene using visual information captured by cameras or sensors. It plays a crucial role in various applications such as autonomous navigation, augmented reality, robotics, and location-based services. With the advent of deep learning techniques, visual place recognition has witnessed significant advancements, leading to remarkable improvements in accuracy and robustness. Traditional place recognition methods often rely on handcrafted features and classical machine learning algorithms, which are limited in their ability to capture complex visual patterns and handle variations in illumination, viewpoint, and environmental conditions. However, deep learning has revolutionized the field by enabling the automatic learning of hierarchical representations from raw visual data, allowing for more discriminative and robust feature extraction. Deep learning models, particularly convolutional neural networks (CNNs), have demonstrated remarkable success in various computer vision tasks. In this work, we present some of the most common descriptor aggregators, loss functions and optimizers studied for this purpose and combine their performance to obtain the optimal one with respect to the studied data-sets. Moreover we propose a new globally informed mini-batch sampling technique, which instead of randomly sampling places at each iteration, it uses a proxy index to construct mini-batches containing visually similar places.

2. Related Work

In recent years, there has been significant research on various aspects of visual place recognition, ranging from feature extraction and representation to deep learning-based methods. In this section, we provide an overview of the relevant literature in this field. With the success of deep learning, almost all recent VPR techniques make use of learned representations. This usually involves using features extracted from a backbone network pretrained on image classification datasets [7] [18], followed by a trainable aggregation layer that transforms these features into robust compact representations. One notable aggregation technique is NetVLAD [2], which is a trainable variant of the VLAD descriptor, where local features are softly assigned to a learned set of clusters. As a result of the success of NetVLAD, many variants have been proposed in literature. Kim et al. [13] introduced Contextual Reweighting Network (CRN) which estimates a weight for each local feature from the backbone before feeding it into a NetVLAD layer; their approach introduced a slight but consistent performance boost. Another notable aggregation technique is the Generalized Mean (GeM) [16] which is a learnable generalized form of global pooling. Building on GeM, Berton et al. [8] recently proposed CosPlace, a lightweight aggregation technique that combines GeM with a linear projection layer. Their method showed impressive performance on the task of VPR, outperforming GeM and NetVLAD and achieving state-of-the-art results on multiple benchmarks. In addition to the aforementioned works, Ali-bey et al. [5] also proposed MixVPR, a feature mixing approach for visual place recognition, which combines local and global features for improved accuracy. While there have been many works [5, 6, 8, 13, 16] in designing effective CNN fea-

ture map aggregation methods, they almost all exclusively use triplet [11] or contrastive [12] embedding objective to supervise CNN training. Both of these two objectives pull the L2 distance of matchable image pair while push the L2 distance of non-matching image pair.

3. Methodology

This section describes the VPR pipeline and our experimental setup.

3.1. Visual Place Recognition System

A typical visual place recognition pipeline in deep learning begins with capturing images or frames from a camera or sensor mounted on the machine. The acquired images are then processed to extract relevant visual features that capture the distinguishing characteristics of the scene with Convolutional Neural Networks (CNNs) [7, 10, 18], which automatically learn and extract discriminative features from images. The extracted features are typically represented in a compact and informative manner to facilitate efficient matching and recognition. Popular techniques include encoding features into descriptors. Then, the extracted features are used to train a recognition model. This step involves training a deep learning model to learn a similarity metric that can discriminate between different places. Once the recognition model is trained, it can be deployed to perform recognition of new or unseen places. This involves extracting features from the current scene and comparing them to the reference set using the trained model. This abstract model contains several components that can be modified, both during training and test time: the backbone (section 3.2), feature aggregators (section 3.3), losses (section 3.4) and mining (section 3.5). We also tried a different approach which we called Soft Self-Supervised approach (section 3.6). We conduct a series of tests focused individually on each of these elements to systematically show each component's influence.

3.2. Backbone

The CNN backbone is an essential component of any Visual Geo-localization (VG) system, as it is responsible for extracting highly informative feature maps from images. In our experiments we used a ResNet18 [10] pretrained on ImageNet. Berton et al. [9] showed that deeper ResNets, such as ResNet-50 and ResNet-101, achieve better results, but ResNet-18 allows for much faster and lighter computation, making it the most efficient, lightweight backbone.

3.3. Aggregators

In VPR, various pooling techniques have been developed to deal with the results of backbone interactions. These pooling techniques play an important role in features combined and summarized extracts to form a discriminatory representation. A number of aggregation strategies have been proposed [5, 6, 8, 16] over the years from simple pooling levels to incredibly complex modules. In our research we examined different aggregators considering the aforementioned strategies about the issue, in order to determine the most effective method for our particular project.

3.3.1 Generalized-Mean pooling layer

GeM pooling [16] is a pooling layer based on a generalized-mean that has learnable parameters, either one global or one per output dimension. It takes as input the output of the backbone and produces a vector $\boldsymbol{f}$ as output of the pooling process.

$$\boldsymbol{f}(x) = [f_1 ... f_k ... f_K], f_k = \left(\frac{1}{K} \sum_{i=1}^{K} x_i^p \right)^{\frac{1}{p}} \quad (1)$$

where $x = [x_1, x_2, ..., x_K]$ is the set of features extracted from the image, K is the total number of features, and p is the power parameter controlling the degree of the mean. The GeM operation applies the power parameter p element-wise to each feature before computing the mean. Finally, the result is raised to the power of $\frac{1}{p}$ to obtain the final pooled feature representation.

3.3.2 CosPlace aggregator

The CosPlace aggregator [2, 6, 8] combines the GeM [16] pooling operation with a linear layer to generate compact and discriminative representations for place recognition. More precisely, it normalizes the input (i.e. output of the backbone, that we call *descriptors*) using L2 normalization and applies the GeM pooling operation. Then it passes the flattened feature vector through the fully connected layer and renormalizes it.

3.3.3 MixVPR aggregator

The MixVPR pooling layer showed in Figure 1 combines local and global features to enhance the discriminative power of the pooled representation. We start by extracting feature maps $F \in \mathbb{R}^{c \times h \times w}$ from the backbone. We reshape each 2D feature map into a 1D representation, resulting in flattened feature maps $F \in \mathbb{R}^{c \times n}$, where $n = h \times w$. Then these flattened feature maps pass through a cascade of L Multi-Layer Perceptron (MLP) blocks, which we refer to as Feature-Mixer blocks. Each Feature-Mixer block consists of two fully connected layers followed by a ReLU activation function. The output, at the end of the L Feature-Mixer blocks, is then dimensionally reduced using two fully connected layers.

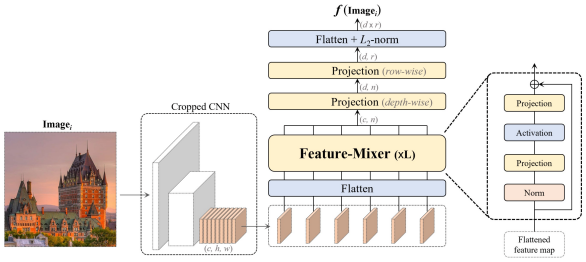

Figure 1. MixVPR aggregator from [5] for place recognition. MixVPR takes as input flattened feature maps of a pretrained backbone. It incorporates spatial relationship in each individual feature map through a succession of Feature-Mixer blocks. The resulting output is then projected into a compact representation space and used as global descriptor.

3.4. Loss Functions

In Visual Place Recognition (VPR), the choice of appropriate loss functions plays am important role in training models to effectively deal with the results of backbone interactions. These loss functions are essential for guiding the learning process and optimizing the feature representations to enable accurate and discriminative place recognition. Over the years, several loss functions have been proposed [11, 12, 19], each with its own strengths and characteristics. In our research, we focused on examining different loss functions and found that the contrastive and multi similarity resulted in better results, thanks to their effectiveness and relevance to our particular project.

3.4.1 Contrastive Loss

The contrastive loss aims to learn discriminative feature representations by maximizing the similarity between similar instances and minimizing the similarity between dissimilar instances. It encourages the model to effectively separate instances belonging to different places in the feature space. The contrastive loss can be defined as follows:

$$\mathcal{L}(i,j) = \begin{cases} \frac{1}{2}\|\bar{f}(i) - \bar{f}(j)\|^2 & \text{if } Y(i,j) = 1, \\ \frac{1}{2}\max\left\{0, \tau - \|\bar{f}(i) - \bar{f}(j)\|^2\right\} & \text{if } Y(i,j) = 0 \end{cases} \tag{2}$$

In this system of equations, $L(i, j)$ represents the loss between image pairs $(i,\ j)$, $\bar{f}(i)$ denotes the average feature representation of instance i, $\bar{f}(j)$ represents the average feature representation of instance j, $Y(i, j)$ is the label indicating whether instance i and j are similar or dissimilar (1 for similar instances, 0 for dissimilar instances), $\|\cdot\|$ denotes the Euclidean norm, and τ is a margin parameter defining when non-matching pairs have large enough distance in order to be ignored by the loss.

3.4.2 Multi Similarity Loss

Multi Similarity loss [19] aims to encourage instances with similar characteristics to have higher similarity scores while pushing dissimilar instances apart. Its formula is given by:

$$\mathcal{L} = \frac{1}{m}\sum_{i=1}^{m}\left(\frac{1}{\alpha}\log\left[1 + \sum_{k\in P_i}\left(e^{-\alpha(s_{ik}-\lambda)}\right)\right] + \frac{1}{\beta}\log\left[1 + \sum_{k\in N_i}\left(e^{-\beta(s_{ik}-\lambda)}\right)\right]\right) \tag{3}$$

Here, L represents the multi similarity loss, m is the total number of instances, and i is the index of the current instance. The formula consists of two terms, each corresponding to a different set of instances.

The first term of the formula penalizes instances that have similarity scores lower than the desired threshold λ. In this term, α is the weight factor that controls the sensitivity of the penalty for similar instances. The summation calculates the sum of exponential penalties for each similar instance k in the set P_i (positives), representing the set of similar instances to instance i. Similarly, the second term of the multi similarity loss formula penalizes instances that have similarity scores higher than the desired threshold λ. In this term, β is the weight factor that controls the sensitivity of the penalty for dissimilar instances. The summation calculates the sum of exponential penalties for each dissimilar instance k in the set N_i (negatives), representing the set of dissimilar instances to instance i.

3.5. Mining and Sampling

In visual place recognition, the training of models is commonly performed using pairwise [12, 19] or triple loss [11] functions that highly depend on the hardness of the examples sampled at each training iteration. Existing techniques address this challenge by employing computationally and memory expensive offline hard mining. This process involves identifying, at each iteration, the hardest samples from the training set. In our research, we introduce a recent technique [1] that utilizes global hard mini-batch sampling based on proxies. This approach aims to mitigate the computational and memory costs associated with offline hard mining.

3.5.1 Global Proxy-based Hard Mining

This approach [1], shown in figure 2, involves learning a function, typically a deep neural network composed of a backbone network [7,10,18] and a pooling layer [5,6,8,16], to generate representation vectors $X_i \in \mathbb{R}^d$ given as input images I_i . We then project these representations through the proxy head H, which maps the d-dimensional inputs to d'-dimensional outputs. The proxy head can be a fully

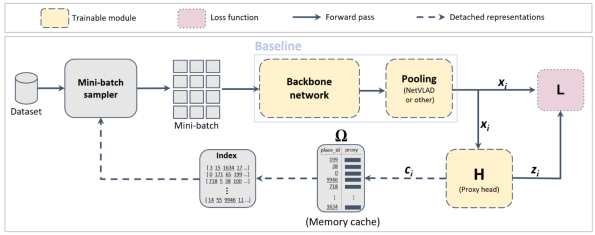

Figure 2. A diagram of the Global Proxy-based Hard Mining from [1]. We add a new end-to-end trainable branch to the network (proxy head H) that projects highly dimensional vectors x_i into very compact representations z_i ; we use the latter to compute one proxy descriptor c_i for each place in the mini-batch. We detach each proxy from the computation graph and cache it into a memory bank Ω. Then, at the beginning of each epoch, we construct an index upon Ω, in which places are gathered together according to the similarity of their proxies. This index is used to sample mini-batches containing similar places, which yields highly informative pairs or triplets.

connected layer, followed by L2 normalization. The representations produced by the proxy head H are utilized to compute compact descriptors (c_i) for each place in the mini-batch. The descriptor c_i is computed as the average of the proxy representations of the images depicting the place P_i. These descriptors, along with their identities, are cached in a memory bank Ω. At the end of each epoch, an index is built upon Ω using k-NN, grouping similar places together based on the similarity of their proxies.

3.6. Soft Self Supervised Approach

Self supervised learning (SSL) approach is a powerful technique that permits to auto-generate psuedo-labels without having them. The goal is to generate a couple of images starting from one: the two representations are obtained exploiting augmentation tools. Having such couples, the system is trained in order to understand the analogies in the images belonging to the same couple and the differences between different pairs. State-of-the-art approaches for VPR are trained in a supervised manner and therefore hardly capture the information needed for generalizing to unusual conditions. Thus, self-supervised learning may help abstract the place representation so that it can be foreseen, irrespective of the conditions. This allows the model to learn useful representations from the data and not to struggle against adverse seasonal and illumination conditions, without requiring any human annotated labels. We decided to use a new method of SSL which consists of merging the self supervised approach with the labels, that we called soft-self-supervised learning (SSSL). In the soft supervised approach the couples of images are created with respect to the labels, which means that we take images from the class: one is augmented while the other is used as an anchor/reference

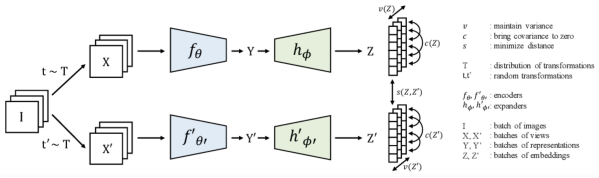

Figure 3. Self Supervised Augmentation approach from [4]

embedding. In order to achieve such avantages from the SSL approach, we need to use a specific loss that does not use labels: that is the VICReg loss. In Figure 3 we can see how the SSL architecture works.

3.6.1 VICReg Loss

Variance-Invariance-Covariance Regularization (VICReg) is a method that explicitly avoids the collapse problem with a simple regularization term on the variance of the embeddings along each dimension individually. VICReg combines the variance term with a decorrelation mechanism based on redundancy reduction and covariance regularization. The variance term encourages the embeddings to have high variance along each dimension individually, which helps to avoid the collapse problem where the encoder outputs constant vectors. The invariance term encourages the embeddings to be invariant to different views of the same image, which helps to learn a representation that captures the underlying structure of the data. The covariance term encourages the embeddings to have low covariance between different dimensions, which helps to learn a decorrelated representation that captures independent factors of variation in the data. The great advantage of VICReg Loss is that it does not require techniques such as batch normalization, feature-wise normalization, and achieves results on par with the state of the art on several downstream tasks.

The VICReg loss function consists of three terms: variance, invariance and covariance.

The variance regularization term is defined as:

$$v(Z) = \frac{1}{d}\sum_{j=1}^{d} max(0, \gamma - S(z^j, \epsilon))$$

where S is the regularized standard deviation defined by: $S(z, \epsilon) = \sqrt{Var(z) + \epsilon}$; γ is a constant target value for the standard deviation, ϵ is a small scalar preventing numerical instabilities [4]. The invariance regularization term is simply:

$$s(Z, Z') = \frac{1}{n}\sum_{i} ||z_i - z'_i||^2$$

The third one is made starting from the covariance matrix:

$$C(Z) = \frac{1}{n-1}\sum_{i=1}^{n}(z_i - \bar{z})(z_i - \bar{z})^T,$$

where $\bar{z} = \frac{1}{n}\sum_{i=1}^{n} z_i$. Then define the covariance regularization term c as the sum of the squared off-diagonal coefficients of C(Z), multiplied by a factor $1/d$ to scale the criterion as a function of the dimension d:

$$c(z) = \frac{1}{d}\sum_{i \neq j}[C(z)]_{i,j}^2$$

Therefore we define the final loss as the sum of the terms above:

$$\ell(Z, Z') = \lambda s(Z, Z') + \mu[v(Z) + v(Z')] + \nu[c(z) + c(Z')]$$

where λ, μ and ν are the hyper-parameters that regulate the relevance of the terms in the loss [4]. The final loss we want to minimize, over all unlabelled images of a dataset D is:

$$\mathcal{L} = \sum_{I \in \mathcal{D}} \sum_{t,t' \sim \mathcal{T}} \ell(Z^I, Z^{'I})$$

where Z^I and $Z^{'I}$ are the batches of descriptors that correspond to the batch of images I transformed by t (basic transformation) and t' (advanced augmentations).

4. Experiments

We investigate the impact of various components on the results obtained from our methodology. Building upon an existing codebase, we sought to overcome the baseline performance by introducing modifications as explained in our methodology section 3. By analyzing and comparing different aggregation methods, loss functions, miners, and a soft self-supervised approach, we aim to understand their individual contributions and optimize our methodology for improved outcomes. In all experiments, 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 R@5 and, following common practice in recent literature [2, 14], we use 25 meters as a distance threshold. In the following sections, we will delve into the details of each component and present our findings.

4.1. Datasets

We explore the datasets of TokyoXS, San FranciscoXS and Google Street ViewXS [6] (GSV) dataset, which provide valuable resources for computer vision and geospatial analysis.

4.1.1 Google Street ViewXS (GSV) as train set

The Google Street View (GSV) dataset offers an extensive collection of images captured by Google's Street View cars, covering diverse cities and locations worldwide. This dataset provides a global perspective, encompassing both urban and rural areas. Each image in the GSV dataset is associated with geolocation information. It consists of 529506 photos from 23 cities.

Table 1. Baseline Results for San Francisco and Tokyo Datasets

Dataset	R@1	R@5	R@10	R@20
SF	32.6	47.1	52.5	57.0
Tokyo	42.5	58.4	64.8	71.4

4.1.2 San FranciscoXS as validation and test set

San Francisco dataset provides a comprehensive set of images showcasing the city's unique features. It includes iconic landmarks, as well as various neighborhoods, parks, and urban landscapes. It contains 28191 photos divided into database and queries (only 1000) folder for the validation and test set.

4.1.3 TokyoXS as test set

Tokyo datasetXS offers a diverse collection of images capturing different locations within the city. It encompasses a wide range of neighborhoods, landmarks, streets, and urban scenes. It contains 12771 photos for the test set and 315 for the queries.

4.2. Baseline

The baseline code employed a ResNet18 [10] as backbone pretrained on ImageNet365. The features were aggregated using an average pooling, and the model was trained with contrastive loss [12]. Stochastic Gradient Descent (SGD) optimizer was used to optimize the model parameters. Table 1 shows the results of the baseline code.

4.3. Aggregators

Pooling techniques play a crucial role in combining and summarizing features extracted from backbone interactions, leading to the creation of discriminative representations. In our research, we aimed to investigate and compare different aggregation methods, including GeM pooling [16], CosPlace Aggregator [8], and MixVPR [5]. Table 2 provides an overview of the results obtained from these experiments. Specifically, we focused on modifying the aggregator module while keeping other components of the baseline code unchanged. After conducting extensive experimentation, we observed that the best performance among the aggregation methods was achieved by combining MixVPR with the baseline code. This combination yielded superior results in terms of accuracy and discriminative representation. These findings highlight the importance of selecting an effective aggregator for visual place recognition tasks, and demon-

strate the efficacy of the MixVPR approach when integrated with the baseline code.

4.4. Mining and Sampling

In our experiments, we incorporated the Proxy Miner framework [1] for visual place recognition (VPR). In our code we present the implementation of the ProxyHead, ProxyBank, and ProxyBankBatchSampler classes. The Proxy Miner framework enables the creation and maintenance of a proxy bank, where each proxy represents a cluster of similar visual descriptors. The ProxyBankBatchSampler, instead, is responsible for generating batches of training data by sampling proxies from the bank. To evaluate the effectiveness of the Proxy Miner, we conducted experiments on our system. We compared the performance of the system with and without the Proxy Miner, using MixVPR as aggregator since it proved to be the best one in the previous results. The results are summarized in Table 3. We observed that the inclusion of the Proxy Miner significantly improved the performance of the VPR system, as evidenced by the higher scores. This demonstrates the efficacy of the Proxy Miner framework in enhancing the discriminative power of the visual descriptors and improving the accuracy of place recognition.

4.5. Losses for Supervised Learning

In our experiments, we studied the impact of incorporating a Proxy into the MixVPR framework using different loss functions and the best ones found are Contrastive [12] and Multisimilarity [19] Losses. As presented in Table 3, the inclusion of a Proxy combined with the Multisimilarity loss resulted in notable performance improvements for MixVPR with a Proxy. Specifically, this configuration exhibited significantly higher scores for both the SF and Tokyo datasets. These findings demonstrate that the integration of a Proxy in conjunction with suitable loss functions can effectively augment the overall performance of the MixVPR framework.

4.6. Optimizers and Schedulers

In our experiments, shown in table 4, we compared two popular optimization algorithms, Stochastic Gradient Descent (SGD) [17] and Adam [15]. SGD is a classic optimization method that updates the model parameters based on the gradients computed on mini-batches of the training data. On the other hand, Adam is an adaptive optimization algorithm that combines ideas from both momentum and RMSprop. It computes adaptive learning rates for individual parameters, which helps in converging faster and handling sparse gradients effectively. During our experimentation, we found that both SGD and Adam (with a learning rate of 1e-4) performed well on their own, but we wanted to further improve the results by incorporating a StepScheduler. The StepScheduler adjusts the learning rate at specified milestones during training, which can be beneficial for fine-tuning the model and achieving better performance. After simulations, we identified the best configuration, in

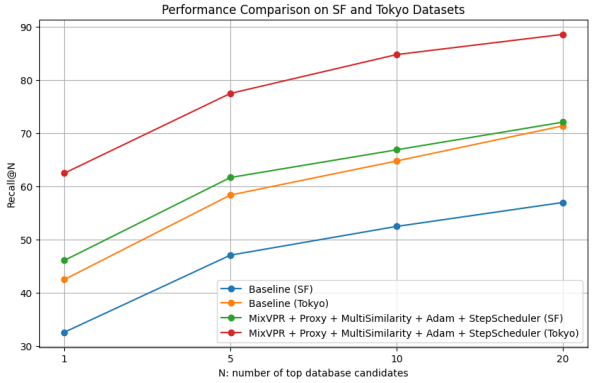

Figure 4. Comparison of performance on the SF and Tokyo datasets

(a) Image used as anchor

(b) Augmented image representing the same place

Figure 5. Image transformation

Table 2. Performance Comparison of Different Aggregation Methods on the SF and Tokyo Datasets

	SF Dataset				Tokyo Dataset			
	R@1	**R@5**	**R@10**	**R@20**	**R@1**	**R@5**	**R@10**	**R@20**
Base	32.6	47.1	52.5	57.0	42.5	58.4	64.8	71.4
GeM Pooling	31.7	45.4	51.6	58.3	45.4	62.5	69.8	77.8
CosPlace Aggregator	31.7	49.2	55.9	61.9	49.2	69.2	75.9	85.7
MixVPR	33.4	49.9	55.7	61.1	50.8	70.5	77.5	84.1

Table 3. Performance Comparison of MixVPR

	SF Dataset				Tokyo Dataset			
	R@1	**R@5**	**R@10**	**R@20**	**R@1**	**R@5**	**R@10**	**R@20**
MixVPR	33.4	49.9	55.7	61.1	50.8	70.5	77.5	84.1
MixVPR + proxy + Contrastive	38.1	54.1	59.4	65.3	56.5	72.7	81.0	85.7
MixVPR + proxy + MultiSimilarity	38.7	55.5	60.5	67.3	58.7	75.9	84.1	87.9

figure 4, that yielded the highest performance. This configuration involved utilizing a Proxy in combination with the Adam optimizer, MixVPR framework, Multisimilarity loss function, and StepScheduler. The results obtained with this configuration outperformed other variations and show the effectiveness of our proposed approach.

4.7. Soft Self Supervised Learning

For further analysis, we explored the use of Soft Self-Supervised Learning to see how it compares to a standard supervised approach. One of the main aspects of the self-supervised approach is the choice of the augmentations to use and the loss to implement. Considering the effectiveness of the VICReg loss [4], we decided to focus our efforts in understanding which transformations can achieve the best results. It is well known that for the task of Visual Place Recognition a significant problem is the change of brightness between the images because the model struggles recognizing that the place in analysis is the same if appears with a different light. Therefore, augmentations like `ColorJittering` or `GreyScale` could have been interesting to test. Unfortunately, results showed us that changing this aspect in the images leads only to a loss of information. Achieved this result, we moved forward thinking about changing the perspective of the place in several ways: augmentations like `RandomHorizontalFlip`, `RandomCrop` and `RandomPerspective` proved interesting improvements and the gain in generalization. The figures 5a and 5b shows the comparison between an anchor image and an augmented one belonging to the same place. We decided to use the maximum number of `images_per_place` which is 4, therefore we have 2 images that will be used as anchor (reference embeddings) and the other 2 that will be the augmented ones. In this way the model is more robust to overfitting and is improved respect to using only 2 images in total. In the table 5 we can see some results regarding SSSL approach. The results are well over the baseline with a +7% despite the fact that an SSSL approach in this specific task can't reach R@1 levels of a supervised approach. This is mainly because we already have a lot of images that are correctly labelled based on their GPS coordinates and the test sets, in particular the San Francisco XS set, is more challenging due to its bigger size and different domains shifts.

4.8. Visualizations and final considerations

Figure 6. Sample of visualizations. The first image of each row represents the query, while the others are the images retrieved

At the end of the test training, it could be interesting to see what images in the database (or better, the feature representations of the image) resulted to be more similar to the query picture (Figure 6). In the figures reported it is possible to understand where the model struggles and inference what

Table 4. Performance comparison of MixVPR with Proxy, MultiSimilarity loss, and different optimizers on the SF and Tokyo datasets.

	SF				Tokyo			
	R@1	R@5	R@10	R@20	R@1	R@5	R@10	R@20
MixVPR + proxy + MultiSimilarity + SGD	38.7	55.5	60.5	67.3	58.7	75.9	84.1	87.9
MixVPR + proxy + MultiSimilarity + Adam	42.9	58.4	64.0	69.7	58.7	75.9	84.7	88.3
MixVPR + proxy + MultiSimilarity + Adam + StepScheduler	**46.1**	**61.7**	**66.9**	**72.1**	**62.5**	**77.5**	**84.8**	**88.6**

Table 5. Performance of SSSL at 10 Epochs. The best augmentations found are `RandomHorizontalFlip`, `RandomPerspective` and `RandomCrop`.

	SF				Tokyo			
	R@1	R@5	R@10	R@20	R@1	R@5	R@10	R@20
AVGpooling + VICReg + SGD + StepScheduler	19.4	33.9	40.5	46.8	29.5	51.1	59.4	68.9
AVGpooling + VICReg + Adam + StepScheduler	36.7	53.2	57.8	63.1	45.1	60.0	**67.9**	**76.2**
AVGpooling + proxy + VICReg + Adam + StepScheduler	**39.3**	**53.8**	**58.6**	**63.7**	**47.3**	**60.0**	65.4	74.9

are the issues that leads to a mismatch, but also to understand where the model achieve good result, in other words how the representation on the place is more conforming to the aim of the architecture. The first query is well predicted for all of the four photos retrieved from the database: this is because the image is very clear and the principal subject, the house, is visualized properly. The second scenario is similar to the first one, but given the similar structures of the houses found in the dataset folder, the model struggles to distinguish between the details; although the error is incontrovertible, it can at least be justified considering the difficulties that also an individual could have. The worst mismatch regards for sure the third query, where the shadow covers a good part of the structure and this bring a confusion to the model so the most similar images predicted are blurred-like or characterized by an high contrast of light and dark colours. This confirm that the influence of a good illumination in the geo-localization topic is relevant. Eventually the assurance of having good quality images in the datasets is one of the most significant aspects of VPR tasks.

5. Conclusion

In conclusion, this paper explores the influence of different components on the results obtained from our methodology. Through the analysis of various aggregation methods, loss functions, miners, and a soft self-supervised approach, we uncover their individual contributions. The soft self supervised approach resulted in improving the performances and as future studies could be interesting to augment the images with more complex transformations that work again with the perspective [3]. Finally, the best configuration, that uses a Proxy in combination with the Adam optimizer, MixVPR framework, Multisimilarity loss function, and StepScheduler outperformed other variations and show the effectiveness of our proposed approach.

References

[1] Amar Ali-bey, Brahim Chaib-draa, and Philippe Giguère. Global proxy-based hard mining for visual place recognition. *BMVC*, 2023. 3, 4, 6

[2] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic. Netvlad: Cnn architecture for weakly supervised place recognition. *TPAMI*, 2018. 1, 2, 5

[3] Yalong Bai, Mohan Zhou, Yuxiang Chen, Wei Zhang, Bowen Zhou, and Tao Mei. Augmentation pathways network for visual recognition. *arXiv preprint arXiv:2107.11990*, 2021. 8

[4] Adrien Bardes, Jean Ponce, and Yann LeCun. Vicreg: Variance-invariance-covariance regularization for self-supervised learning, 2022. 4, 5, 7

[5] Ali bey Amar, Brahim Chaib-draa, and Philippe Giguère. Mixvpr: Feature mixing for visual place recognition. *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2023. 1, 2, 3, 5

[6] Ali bey Amar, Brahim Chaibdraa, and Philippe Giguère. Gsv-cities: Toward appropriate supervised visual place recognition. *Neurocomputing*, 2022. 1, 2, 3, 5

[7] 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. *IEEE*, 2017. 1, 2, 3

[8] Berton G., Masone C., and Caputo B. Rethinking visual geo-localization for large-scale applications. *CVPR*, 2022. 1, 2, 3, 5

[9] Berton G., Mereu R., Trivigno G., Masone C., Csurka G., Sattler T., and Caputo B. Deep visual geo-localization benchmark. *CVPR*, 2022. 2

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CVPR*, 2015. 2, 3, 5

[11] E. Hoffer and N. Ailon. Deep metric learning using triplet network. *SIMBAD*, 2015. 2, 3

[12] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *CVPR*, 2020. 2, 3, 5, 6

[13] H. J. Kim, E. Dunn, and J.-M. Frahm. Learned contextual feature reweighting for image geolocalization. *CVPR*, 2017. 1

[14] C. Masone and B. Caputo. A survey on deep visual place recognition. *IEEE Access*, 2021. 5

[15] Kingma Diederik P and Ba Jimmy. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 6

[16] Filip Radenovic, Giorgos Tolias, and Ondrej Chum. Fine-tuning cnn image retrieval with no human annotation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018. 1, 2, 3, 5

[17] Ruder Sebastian. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016. 6

[18] Tsun-Hsuan Wang, Hung-Jui Huang, Juan-Ting Lin, Chan-Wei Hu, Kuo-Hao Zeng, and Min Sun. Omnidirectional cnn for visual place recognition and navigation. *IEEE*, 2018. 1, 2, 3

[19] Xun Wang, Xintong Han, Weilin Huang, Dengke Dong, and Matthew R. Scott. Multi-similarity loss with general pair weighting for deep metric learning. *CVPR*, 2020. 3, 6