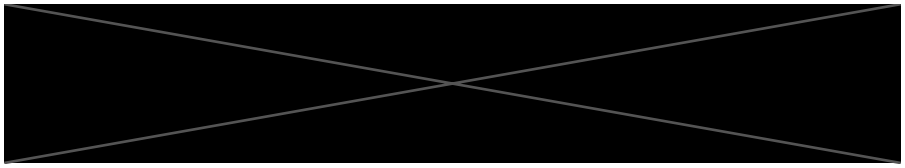
Classification for Visual Geolocalization



***Abstract*—Visual Geo-localization (VG) is the process of determining the location where a photograph was taken by comparing it with a database of images from known locations. In this study, a new highly scalable training technique called CosPlace is used, which performs the training part as a classification problem preventing expensive mining by commonly used contrastive learning. Our research focuses on three main areas of improvement: adapting the model to perform more efficiently in the nighttime domain, correcting distortion, and utilizing re-ranking or post-processing techniques for overall improvement.**

I. Introduction

A. Main Model

The problem of approximately identifying the geographical location where a photo was taken is called Visual (Image) Geo-localization (VG) or Visual Place Recognition (VPR) [4], [5]. The typical approach is to address a VPR task as an image retrieval problem, searching for the closest matches of a query (i.e., the image to be localized) in an image database, and using the retrieved images metadata to determine the location of the query, usually accepting an error margin of a few meters [1].

Specifically, recent methods for VPR are based on machine learning: a neural network is trained to perform the image transformation into an embedded space which effectively captures the similarity of the image geographical positions, allowing for efficient retrieval. Until recently, research on VPR has primarily focused on identifying the location of images within relatively small geographical regions, such as neighborhoods. During the last year, a new method, called CosPlace [1], has been proposed to address more practical applications, where it is often required to operate on a much larger scale (e.g., cities or metropolitan areas). This approach allows one to take full advantage of the availability of large databases of geo-tagged images, not only during the execution of the location retrieval, but also in the training phase of the model.

This new method addresses the two main limitations of previous research:

- *Non-representative datasets*: Previously available datasets for VG were not suited for large-scale application problems due to the limited geographical coverage [4] or sparse distribution of the population [6]. Additionally, images were divided into separate sets for training and inference but that separation did not reflect real-world scenarios where the goal is to use images from the specific area being targeted for training the model.
- *Scalability of training*: large amount of data becomes a challenge in how to effectively utilize it for the training. The majority state-of-the-art methods in VG make use of contrastive learning [5], [7]. Most of them rely on triplet loss which heavily depends on finding negative examples throughout the training dataset. This process is computationally expensive and becomes infeasible when the dataset is very large. A possible workaround is to rely on lightweight mining strategies which reduce the time but slow down convergence and waste available data.

B. Contributions

The development process for some extensions is as follows:

- *Domain adaptation*: Deep architectures are trained on enormous volume of labeled data. In case of lack of labeled data, given that labeled data of a comparable kind but from a different domain (such as synthetic photographs) are available, domain adaptation frequently offers an appealing alternative in the absence of labeled data for a particular job. We use GRL (Gradient Reversal Layer) method which uses a backpropagation loss to enforce the same distribution of the feature's norm across domains [3].
- *Re-ranking / post-processing*: There are two strategies to reorder the top-k candidates of model predictions. In terms of re-ranking, the goal is to put groups of candidates with a larger majority on top of those with smaller ones. On the other hand, when it comes to post-processing, we only maintain one candidate from each group and eliminate redundant candidates if they originate from the same location.
- *Undistortion*: This is a widely utilized technique in the field of computer vision that aims to correct for image distortion and produce a more accurate representation of the scene. The process of undistortion includes the analysis of distortion present in the image, followed by the application of correction to each individual pixel, resulting in a unbending image with straight lines and minimal distortion. The ability to produce rectilinear images with minimal distortion enhances the accuracy of these applications, ultimately leading to improved performance and usability.

II. Related works

In this section, we concentrate on examining previous research that is related to our own findings and contributions.

A. Domain Adaptation

A number of approaches to domain adaptation has been suggested in the context of shallow learning, e.g. in the situation when data representation/features are given and fixed. The proposed approaches then build the mappings between the source (training-time) and the target (test time) domains, so that the classifier learned from the source domain can also be applied to the target domain, when composed with the learned mapping between domains. A large number of domain adaptation methods have been proposed over the recent years, and here we focus on the most related ones. Multiple methods perform unsupervised domain adaptation by matching the feature distributions in the source and the target domains. Some approaches perform this by reweighing or selecting samples from the source domain [8], while others seek an explicit feature space transformation that would map source distribution into the target ones [9].

B. Re-ranking/post-processing

There have been several studies on re-ranking and post-processing techniques for geo-localization [10]. These techniques aim to improve the accuracy of geo-localization predictions by using additional information or by refining the predictions made by a primary model. One example of a re-ranking technique is using language-based features to re-rank the top-n predictions made by a primary model. In this approach, the primary model makes a set of predictions and the re-ranking model uses features such as the presence of location-specific keywords in the text to re-rank the predictions. One example of a post-processing technique is to use contextual information to refine the predictions made by a primary model. This approach involves using information such as the user's location history or the locations of nearby users to adjust the predictions made by the primary model.

C. Undistortion

Due to the prevalence of 360-degree consumer cameras and the numerous applications that might benefit from the broad field of view, omnidirectional photography is growing in popularity. Convolutional neural networks (CNNs) have never been a more popular deep learning technique for computer vision problems. However, using CNNs to process spherical pictures is more complex than just using spherical projections onto an image plane to train a standard CNN [11].

III. Datasets

There are three types of datasets to be implemented for our different purposes as follows: San Francisco eXtra Small (SF-XS), San Francisco v1 small (SF v1 small), Tokyo small. SF-XS is mainly used for training and the other two are used for test time. It should be mentioned that all datasets are the short version of the biggest datasets for the matter of convenience and less complexity. Our model performance in terms of domain shifts, occlusion and perspective changes are evaluated based on these datasets. There is another dataset which is a subset of Tokyo small named Tokyo night including just night pictures. We use this dataset for domain adaptation techniques as an additional branch. For all experiments, results are computed on all mentioned datasets.

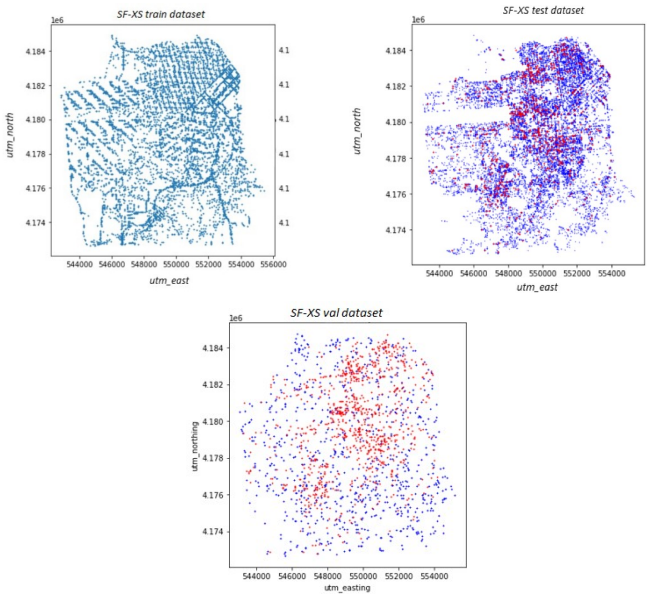


Fig. 1. Distribution of SF-XS subsets and test datasets

It is worth noting that unlike earlier techniques, CosPlace does not need the train set to be divided into database and queries. We believe that this decision significantly affects the outcomes since they depend on database/queries division for training which requires selecting best split that best meets their needs.

Regarding test queries, they should not originate from the same domain as the database since, in the majority of real-world cases, test time queries could come from unknown domain.

TABLE I
Rounded image numbers for each subset of SF-XS and test datasets

	SF-XS	SF-v1 val	SF-v1 test	Tokyo xs	Tokyo night
Database	60k	8k	27k	13k	13k
Queries		8k	1k	315	105

IV. Model

A. Main Method

To achieve high degree of scalability for VG at train and test time, main method addresses two important qualities which other methods are suffering from.

First one is related to the space and time complexity at train time since other state of art require to compute the features of all database images periodically and store them in a cache.

Second, using NetVLAD layer [4] or its variants produces high dimensional vectors requiring huge amount of memory for inference. Although some reduction techniques such as PCA are used but they lead to quick deterioration of results.

To tackle these problems, cosFace and arcFace are used [2], by taking motivation from domain of face recognition. To utilize these losses, we require to divide training set into classes considering that label space is continuous in VG.

In order to divide database into classes, one simple approach is used in which database is split into square geographical cells using UTM coordinates east, north and then slice each cell into a set of classes based on each image's heading. A set of images can be assigned to a class as follows:

$$\{x : [\frac{east}{M}] = e_i, [\frac{north}{M}] = n_j, [\frac{heading}{\alpha}] = h_k\} \quad (1)$$

where parameters M (meter) and (degree) indicate the extent of each class in position and heading. There exists a problem related to identical images which may be allocated to different classes due to quantization error causing confusion for classification-based training algorithm. To avoid this, groups of non-adjacent classes (two classes are adjacent if an infinitesimal difference in position or heading can bring an image from one class to the other) are used for training instead of all classes at once, Fig.2.

These groups, called CosPlace groups, are generated by fixing minimum spatial separation that two classes of the same group should have, either in terms of translation or orientation. Therefore, two hyperparameters are introduced: N controls the minimum number of cells between two classes of the same group, and L is equivalent for orientation, Fig.3. CosPlace groups as disjoint sets of classes are defined by Eq.(2) with following properties:

$$\{\mathrm{G}_u vw = \{C_{e_i n_j h_k} : (c_i mod N = u) \wedge (n_j mod N = v) \wedge (h_k mod L = w)\} \quad (2)$$

- each class belongs to exactly one group
- within a given group, if two images belong to different classes, they are at least M · (N - 1) meters apart or α · (L - 1) degrees apart (see Fig. 3)
- the total number of CosPlace Groups is N × N × L
- no two adjacent classes can belong to the same group (unless N = 1 or L = 1)

To train the network, Large Margin Cosine Loss (LMCL) known as cosFace is used. We cannot directly apply LMCL to any VG dataset since images cannot be divided in finite number of classes. Therefore, LMCL is performed sequentially over each CosPlace group thanks to the proposed partitioning and iterate over the any groups. This procedure is called CosPlace. It should be mentioned that not all groups have to be used for training, but simply one single group. Although, using more than one group improve the final results. For each group:

$$L_{cosPlace} = L_{lmcl}(G_{uvw}) \quad (3)$$

Fig. 2. **Grid of the map** showing how cells are formed based on UTM coordinates. The side length M of each cell is specified by a pair of values (for readability only a few are shown on the right). UTM east, UTM north, and heading are written in white between braces on each of the four photos. Since each cell is divided into various classes based on the heads of the photos, each cell contains images with significantly varied headers/orientations (see the two images on the right), which should belong to distinct classes. We can also observe that, while depicting the same landscape or structure, the two photos at the bottom belong to distinct cells (and, hence, to separate classes), which would be perplexing for a simple classification-based algorithm.

where $u \in \{0, ..., N\}, v \in \{0, ..., N\}, w \in \{0, ..., L\}$, and L_{lmcl} is the LCML loss. In reality, we train each epoch on a different group, starting with G000, moving on to G001, and so on. This approach has the notable characteristic of not requiring mining nor caching, making it a considerably more scalable alternative to methods based on contrastive losses that are frequently employed in visual geo-localization. At validation and test times, we employ the model to extract picture descriptors as in for a conventional database retrieval rather than categorize the query. This enables the model to be applied on additional datasets from uncharted geographic regions.

B. Undistortion Method

To implement an undistortion task, we first need to address our dataset. It should be noted that images from SF-XS and SF-v1 small are provided through cropped 360° panoramas from street view. Although, queries of SF-v1 small are normal images taken by users being in a real deployed system. To do so, we undistort the dataset of SF-v1 small during test time to be similar to query ones then evaluate the effect on the recall metric. (See Fig.4)

C. Re-ranking/Post-processing Methods

In this section, the goal is to refine model's predictions, which are indexed through faiss library, by comparing them with the top-k candidates retrieved from the kNN method

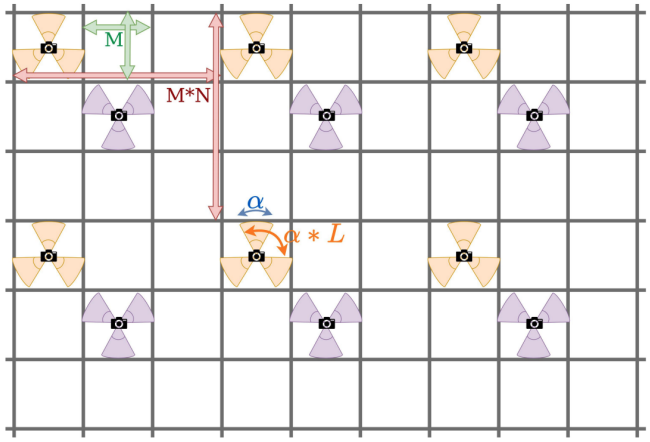


Fig. 3. **Visual representation of CosPlace groups**. One of the N × N × L distinct groups is represented by the orange triangles, while another is represented by the purple triangles. Only two groups are displayed for simplicity (namely G000 and G111). Each triangle represents a class, which includes all pictures in the corresponding cell that are oriented correctly. Visual representations are used to illustrate the significance of each of the four hyperparameters (M,, N, and L) that determine how groups are constructed. In the illustration, N = 3, L = 2, and = 60°. As a result, there are 3 × 3 × 2 = 18 groups in total with this design, and each cell has 6 classes from 2 groups.

Fig. 4. Sample images of SF-v1 small database before and after undistortion.

applied on test database during test time. Concerning re-ranking, the idea is to shift candidates with more majority before the ones with less majority. For instance, for a given query, if the first candidate comes from place A and the next 9 candidates come from place B, one could be confident that the target query is from place B so candidates from B are placed before candidate from place A. On the other hand, regarding post-processing, we remove redundant candidates if they come from the same place and keep only one from a given place. To implement these methods, we first classify our test database by kNN method using their positions (UTM), and then for each specific query, we retrieve all photos as positive images which are truly in the circle with 25 meters radius and the query position as its center (all the images with distance at most 25 meters far from target query). Afterwards, for a given query, we use the generated descriptors of the test dataset, which are our model's outputs, as the input of faiss library to produce indices. In the next step, the predictions are computed by faiss index whose input is query descriptors. Finally, the first n images of these predictions are chosen. (See Fig.5)

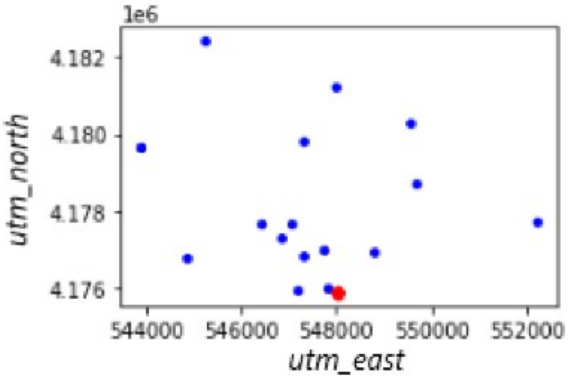


Fig. 5. Red dots are images from kNN which are 25 meters apart from given query, and blue dots are the first 20 un-ordered predictions of the model. The positions are based on UTM coordinates east, north

Now, our methods are able to re-order and refine these n images by comparing to positive images. In order to implement these methods, we need to classify images in meaningful groups. Therefore, DBSCAN method, by which we can put images together based on a distance threshold, is used to group these blue dot images by tuning some hyperparameters to obtain the candidates for each place and improve the results. The hyperparameters are the distance of candidates to be considered in a group and minimum percentage indicating acceptance cardinality of each group to be nominated for shifting. For example, for R@5 using 10 images, when distance is 5 and min-per is 0.4, it indicates that we should create the groups of images 5 meters apart and shift them if the image number of the group is 4. Note that we tune the hyperparameters based on meaningful numbers and some boundaries to obtain reasonable results. (For example, having a group with distance 5 and 4 members is not rational but a boundary).

D. Domain Adaptation Method

The attraction of domain adaption techniques lies in their capacity to discover a mapping across domains when the target domain data are either completely unlabeled (unupervised domain annotation) or contain few labeled samples (semi-supervised domain adaptation). Below, we concentrate on the more challenging unsupervised scenario, while the suggested method may be easily extended to the semi-supervised case. We focus on integrating domain adaptation with deep feature learning inside one training procedure. In order for the final classification decisions to be based on features that are both discriminative and invariant to the change of domains —that is, have the same or very comparable distributions in the source and the target domains— we want to include domain adaptation into the learning process of representation. Learning

a discriminative classifier or other predictor in the presence of a shift between training and test distributions is known as domain adaptation (DA).

This is accomplished by jointly optimizing the underlying features and two discriminative classifiers that operate on these features: i) the label predictor, which forecasts class labels and is used both during training and at test time, and (ii) the domain classifier, which makes a distinction between the source and the target domains during training. The parameters of the underlying deep feature mapping are adjusted to reduce the loss of the label classifier and to maximize the loss of the domain classifier, while the classifier parameters are tuned to minimize their error on the training set. The latter promotes the emergence of domain-invariant features throughout the optimization process.

Importantly, we demonstrate that all three training processes can be trained with standard backpropagation algorithms based on stochastic gradient descent or its modification and can be integrated into a properly constructed deep feedforward network (Fig. 6) using standard layers and loss functions.

For more clarity, we want the model to be trained to distinguish the domain of SF-XS images from the one of Tokyo night queries. To do so, whenever dataloader is called, 32 images (batch size) of SF-XS and certain number of Tokyo night images are loaded, then the model labels them as 0, 1. For instance, 0 for SF-XS and 1 for Tokyo night. During training, the target outputs of the model are descriptors which are used for class label branch, and another output is domain ones for domain label branch. In GRL path, we defined two losses that calculate the differences between predicted labels and true labels of SF-XS and of Tokyo night queries, respectively. After computation of these losses by CrossEntropy function, we sum them as our final loss to backpropagate after multiplication by lambda.

Lambda plays an important role in this structure since it determines how much GRL can influence our feature extractor backbone. The magnitude of lambda, which increases exponentially (Eq.4), should be controlled in a sense that the model can learn its main task without any difficulties and confusion. For this purpose, we should first allow the model to learn through its main architecture without additional branch, and then lambda intervention is added gradually to the architecture. To be precise, to find domain-invariant features at training time, we simultaneously look for the parameters θ_d of the domain classifier that minimize the loss of the domain classifier and the parameters θ_f of the feature mapping that maximize the loss of the domain classifier (by allowing the two feature distributions as similar as possible). We also want to reduce the loss of the label predictor.

$$\lambda = \frac{2}{1 + e^{gamma \times epochnum}} - 1 \tag{4}$$

V. EXPERIMENTS

A. Architecture

CosPlace may be used with almost any image-based model since it is architecture-neutral. For the majority of our studies, we use a straightforward network that consists of a typical CNN backbone, a GeM pooling step, a fully connected layer with an output size of 512, and ResNet-18 as the backbone.

B. Training

As far as hyperparameters are concerned, M = 10 meters, α = 30°, N = 5 and L = 2. For each epoch, 10k iteration with batch size 32 are performed over each group. It should be mentioned that there are some limitations due to the hardware resources and running time which cause setting number of epochs very carefully. The number of groups is set to 1 out of N × N × L groups. After each epoch, validation is performed and a model with best performance is chosen after finishing the training, then testing is implemented by this model.

Our metric is recall@N with a threshold of 25 meters i.e., the percentage of queries for which at least one of the first N predictions is within a 25 meters distance from the query, following standard procedure. After prediction, kNN method is used to compute the recall. In terms of inference time, it can be divided into: i) extraction time to extract the features of an image, which depends on model and resolution, and matching time related to the duration of the kNN to obtain the best matches in the database, which depends on parameters k (i.e., number of candidates), the size of the database, the dimension.

C. Results and Discussion

It is worth noting that we set number of groups just to one and number of epochs are less than 10 due to the mentioned limitations even using Colab with the new policy. Therefore, all results may be changed if these limitations are declined. Furthermore, all the datasets are provided as a fraction of complete and big datasets.

Table 2 shows the results for R@1 and R@5 when we apply unchanged model without any contributions. So, they are our baseline numbers to be improved.

TABLE II
OUTCOMES OF APPLYING PURE MODEL

SF-XS test	Tokyo-xs	Tokyo-night
R@1/R@5= 54.4 /68	R@1/R@5=72.7/85.4	R@1/R@5=56.2/72.4

Due to the limited downward scalability, CosPlace performance is influenced by size of training dataset and recalls vary for logarithmically decreasing sizes of the training dataset. Therefore, it is not suited to be used for training on small datasets it is not suited for training on small datasets nor datasets without orientation labels.

- Regarding **undistortion**, the undistorted test database is supposed to help us to get better results since, now, the images of database and queries have more similar structure. However, this process of undistorting only happens during the

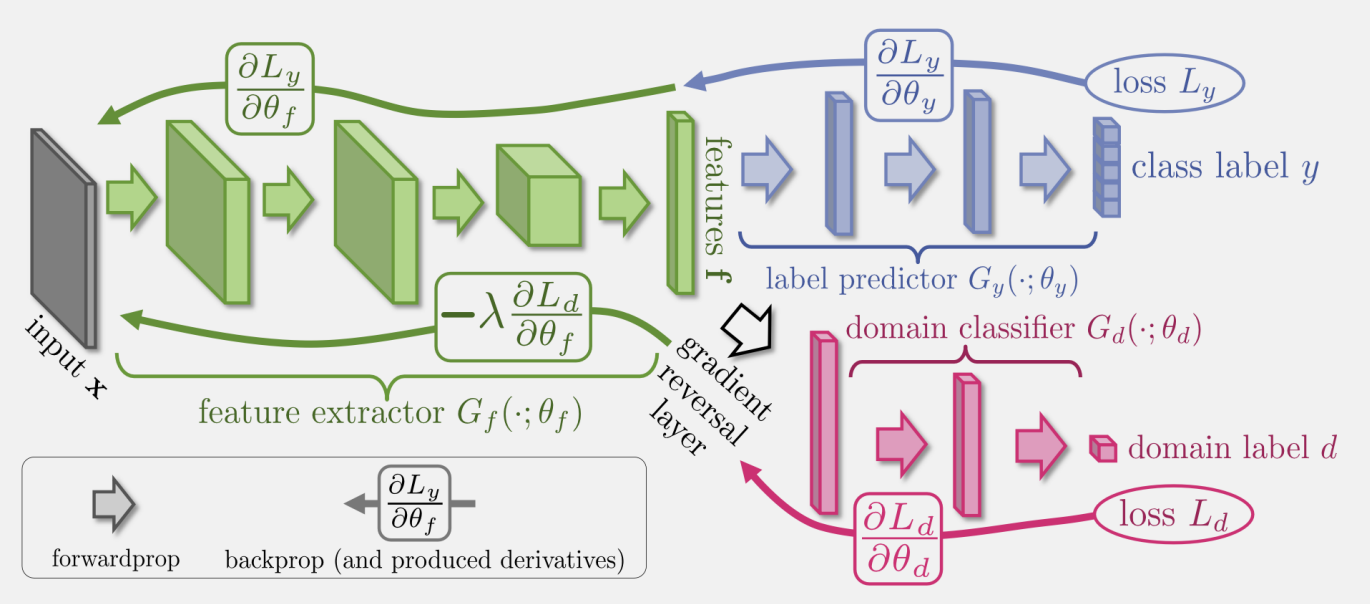


Fig. 6. A deep feature extractor (green) and a deep label predictor (blue) are included in the suggested design, which together make up a typical feed-forward architecture. By including a domain classifier (red) coupled to the feature extractor through a gradient reversal layer that multiplies the gradient by a certain negative constant during the backpropagation-based training, it is possible to accomplish unsupervised domain adaption. Otherwise, the training goes as usual and minimizes the domain classification loss and label prediction loss (for source instances) (for all samples). Gradient reversal makes sure that the feature distributions throughout the two domains are identical (as similar as the domain classifier can tell them apart), leading to the creation of domain-invariant features.

test phase, and the model isn't trained to extract important features and descriptors from undistorted images. This negative impact makes the results actually worse than they would have expected if the original images had not been distorted. (See Table 3)

TABLE III
OUTCOMES OF APPLYING UNDISTORTION

SF-XS test
R@1/R@5 = 52.9/65.4

- Concerning **re-ranking/post-processing**, we use different combinations to evaluate the impact of two hyperparameters: distance and minimum percentage of DBSCAN. The model is tested on three datasets and results vary w.r.t. combinations and type of the dataset. In a sense that postprocessing does not affect R@1 at all and min-per is not used in this case, because the first image is never replaced and re-ordered but each group is truncated by eliminating redundant images. However, these combinations are used for both R@1 and R@5 of the re-ranking method. As Table.4 shows, there are minor improvements (green numbers) in R@1 for re-ranking but more in R@5 for post-processing. We can assume three different cases for each method, unaffected, improvement and deterioration in results, respectively.

For re-ranking, first case is related to the possibility that no group is created based on our combinations or there is not any replacement so the results remain unchanged, but cases 2 and 3 indicate that the grouping replaces the best possible predictions so that the recalls improve if the best ones are placed on top and vice-versa.

Regarding post-processing, case 1 is the same as reranking. To evaluate case 2, it should be mentioned that post-processing picks the first image of each group and eliminates the rest so best possible predictions can be placed among the first 5 images resulting in R@5 improvement. However, case 3 happens when the distance of grouping increases which causes the elimination of best probable predictions if there are not the first position to be chosen.

Finally, we should mention that methods like CosPlace simply rely on a more efficient retrieval through a nearest neighbor search but not on an extra post-processing step.

- To evaluate **domain adaptation** contribution, we perform the task using variable lambda method (with gamma = 0.5) that indicates if the model first gets familiar with its actual task, it is able to integrate domain adaptation task with less negative effects. Nevertheless, due to mentioned limitations, the growth rate of lambda (from no intervention to full intervention) during 10 epochs is fairly fast (Fig.6) so we are not able to take the full advantage of gradual adding of GRL.

Noting that, overall, no further improvement is obtained using domain adaptation task compared to our base results (Table. 5). Unfortunately, we are not able to extract better and more useful features through domain adaptation method. In fact, the mode does not diverge during training but final recalls are less than the ones in Table.2.

There are three main hypothesizes for this problem. First, one can say that Tokyo night query is small that prevent CosPlace to perform efficiently. Second, it is possible that

TABLE IV
RESULTS FROM RE-RANKING AND POST-PROCESSING TESTED ON THREE DATASETS

	Distance	Min-per	SF-XS test	Tokyo-XS test	Tokyo night
Re-ranking	5	0.5	R@1: 54.4, R@5: 67.9	R@1: 72.7, R@5: 85.4	R@1: 56.2, R@5: 72.4
	10	0.6	R@1: 54.3, R@5: 67.9	R@1: 72.7, R@5: 85.1	R@1: 56.2, R@5: 72.4
	25	0.6	R@1: 54.4, R@5: 67.8	R@1: 73.0, R@5: 85.1	R@1: 56.2, R@5: 72.4
	25	0.4	R@1: 54.1, R@5: 67.5	R@1: 72.4, R@5: 83.5	R@1: 55.2, R@5: 70.5
	25	0.8	R@1: 54.4, R@5: 68.0	R@1: 72.7, R@5: 85.4	R@1: 56.2, R@5: 72.4
	50	0.7	R@1: 54.4, R@5: 68.0	R@1: **73.0**, R@5: 85.1	R@1: 56.2, R@5: 72.4
	100	0.8	R@1: 54.4, R@5: 68.0	R@1: 72.7, R@5: 85.4	R@1: 56.2, R@5: 72.4
	250	0.6	R@1: 54.5, R@5: 68.0	R@1: 72.7, R@5: 85.1	R@1: 56.2, R@5: 71.4
	250	0.8	R@1: **54.4**, R@5: 68.0	R@1: 72.7, R@5: 85.4	R@1: 56.2, R@5: 72.4
	2500	0.8	R@1: 54.4, R@5: 67.9	R@1: 72.7, R@5: 85.4	R@1: 56.2, R@5: 72.4
Post-processing	5	-	R@1: 54.4, R@5: 68.2	R@1: 72.7, R@5: 85.4	R@1: 56.2, R@5: 72.4
	10	-	R@1: 54.4, R@5: **68.4**	R@1: 72.7, R@5: 86.3	R@1: 56.2, R@5: 74.3
	15	-	R@1: 54.4, R@5: 68.2	R@1: 72.7, R@5: **86.7**	R@1: 56.2, R@5: 74.3
	20	-	R@1: 54.4, R@5: 68.1	R@1: 72.7, R@5: 86.3	R@1: 56.2, R@5: 74.3
	25	-	R@1: 54.4, R@5: 67.9	R@1: 72.7, R@5: 86.7	R@1: 56.2, R@5: **75.2**
	50	-	R@1: 54.4, R@5: 67.8	R@1: 72.7, R@5: 83.8	R@1: 56.2, R@5: 72.4
	100	-	R@1: 54.4, R@5: 67.6	R@1: 72.7, R@5: 81.6	R@1: 56.2, R@5: 70.5
	500	-	R@1: 54.4, R@5: 66.6	R@1: 72.7, R@5: 83.5	R@1: 56.2, R@5: 71.4
	1000	-	R@1: 54.4, R@5: 64.5	R@1: 72.7, R@5: 85.4	R@1: 56.2, R@5: 72.4
	2000	-	R@1: 54.4, R@5: 65.1	R@1: 72.7, R@5: 85.4	R@1: 56.2, R@5: 72.4

TABLE V
OUTCOMES OF APPLYING DOMAIN ADAPTATION

SF-XS test	Tokyo-xs	Tokyo-night
R@1/R@5=45.3/59.5	R@1/R@5 =49.5/63.5	R@1/R@5=20.2/30.8

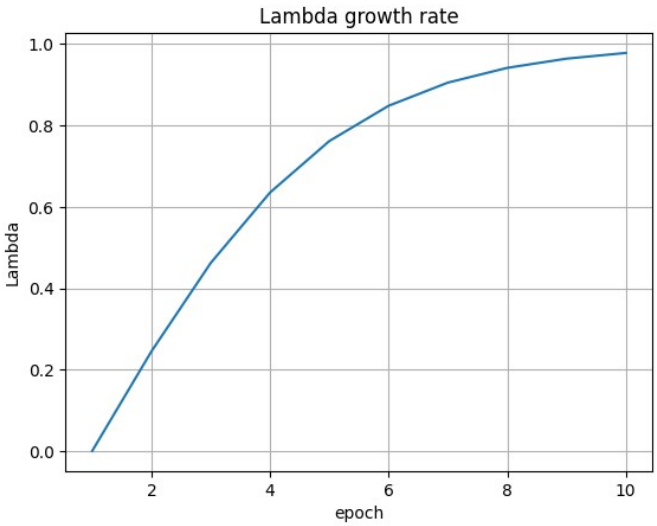


Fig. 7. Lambda growth rate as function of epoch

Fig. 8. Two sample photos of Tokyo night query

the model is not able to find precise descriptors and domain label for comparison since images of Tokyo night query are not very clear, for example, they have concentrated lights like pole illuminations which exacerbate our classification task resulting in poor recall (Fig. 8). The last assumption, the most important one, is related to applying different strategies for lambda variation. For instance, the gamma or other coefficients can be varied, added and tuned, and lambda intervention can be applied after some specific epochs so that the model learns its main task through these epochs.

VI. CONCLUSION

In this project, our main model architecture is based on CosPlace method on which we perform some extensions and compare the outcomes with our baseline results (obtained from applying raw model on test datasets).

For undistortion, the model does not obtain any progress since this process is only implemented during the test phase, and the model is not trained to extract important features and descriptors of undistorted images.

Regarding re-ranking/post-processing, we reach different results by manipulating two hyperparameters (distance and minimum percentage). By testing these combinations, we achieve some improvements for both methods, in particular, further progresses are gained for R@5 of post-processing.

Concerning domain adaptation, the results are unsatisfactory because of three main reasons based on our assumption. The main one is that we are not able to apply different strategies for Lambda intervention due to the time and hardware limitations.

Generally, the small size of our datasets does not allow the model to be trained efficiently. The size of the training dataset affects CosPlace performance because of the restricted downward scaling, and recalls change as the training dataset size decreases logarithmically.

REFERENCES

[1] Berton, Gabriele, 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, pp. 4878-4888. 2022.

[2] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, and Wei Liu. Cosface: Large margin cosine loss for deep face recognition. In IEEE Conference on Computer Vision and Pattern Recognition, pages 5265–5274. Computer Vision Foundation / IEEE Computer Society, 2018 K. Elissa, "Title of paper if known," unpublished.

[3] Ganin, Yaroslav, and Victor Lempitsky. "Unsupervised domain adaptation by backpropagation." In International conference on machine learning, pp. 1180-1189. PMLR, 2015.

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

[5] Liu Liu, Hongdong Li, and Yuchao Dai. Stochastic Attraction-Repulsion Embedding for Large Scale Image Localization. In IEEE International Conference on Computer Vision, 2019.

[6] Aayush Bansal, Hernan Badino, and Daniel F. Huber. Understanding how camera configuration and environmental conditions affect appearance-based localization. In 2014 IEEE Intelligent Vehicles Symposium Proceedings, Dearborn, MI, USA, June 8-11, 2014, pages 800–807. IEEE, 2014.

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

[8] Yu, Le, Jie Wang, and Peng Gong. "Improving 30 m global land-cover map FROM-GLC with time series MODIS and auxiliary data sets: a segmentation-based approach." International Journal of Remote Sensing 34, no. 16 (2013): 5851-5867.

[9] Baktashmotlagh, Mahsa, Mehrtash T. Harandi, Brian C. Lovell, and Mathieu Salzmann. "Unsupervised domain adaptation by domain invariant projection." In Proceedings of the IEEE international conference on computer vision, pp. 769-776. 2013.

[10] Zhang, Xuanmeng, Minyue Jiang, Zhedong Zheng, Xiao Tan, Errui Ding, and Yi Yang. "Understanding image retrieval re-ranking: a graph neural network perspective." arXiv preprint arXiv:2012.07620 (2020).

[11] Eder, Marc, and Jan-Michael Frahm. "Convolutions on spherical images." In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, pp. 1-5. 2019.