Visual Geo-localization: mapping images to GPS

Alessio Carachino
s296138
Politecnico di Torino
alessio.carachino@studenti.polito.it

Francesco Di Gangi
s301793
Politecnico di Torino
francesco.digangi@studenti.polito.it

Abstract

This work aims to implement additional features to the recent paper "Rethinking Visual Geo-localization for Large-Scale Applications" of CVPR(2022) [1]. Visual Geo-localization consists in estimating the position where a given photo, called query, was taken by comparing it with a large database of images of known locations. Briefly, this paper shows how SOTA techniques of Domain Adaptation, Re-ranking, Data Augmentation, and more, can be exploited to significantly improve the recalls by working on the most crucial problems in Visual Geo-localization such as Domain shift, due to different weather or lighting conditions, and obstruction, due to the presence of obstacles in the dataset.

1. Introduction

The Visual Geo-localization task is commonly addressed as an Image Retrieval problem: given an unseen image to be localized (query), it is matched against a database of geo-tagged images that represent the known world. The N-top retrieved images, with their geo-tag (typically GPS coordinates), provide the hypothesis for the query's geographical location.
This project focuses in an innovative approach that extends the so-called CosPlace method, described in [1]. In CosPlace, it is exploited a loss function called CosFace. Besides CosFace, in this work are shown other loss functions, which are:

- ArcFace, described in [3]
- SphereFace, described in [5]

All the work has been done by conducting the training and the experiments on three different datasets:

- *San Francisco eXtra Small (sf-xs)*: contains a train, validation and test set, and it is used to train and test the models.
- *Tokyo eXtra Small (tokyo-xs)*: only contains a test set: it is used to test the models.
- *Tokyo Night (tokyo-night)*: subset of tokyo-xs with only photos taken at night: it is used to test the models.

Tokyo-xs and sf-xs contain domain shifts, occlusions and perspective changes, meaning that the queries are very different from the database images.

1.1. CosPlace with ArcFace

ArcFace [3] is a face recognition algorithm and is not directly related to visual geolocalization, to integrate ArcFace with CosPlace it is only considered the loss function that ArcFace utilizes.
ArcFace focuses on improving the accuracy of face recognition by designing a loss function that considers the inter-class variations and intra-class compactness of the features extracted from face images. It is applied the same principle is applied to the CosPlace loss function to train the images. The common factor is that both losses (and, as it shown in the next subsection, also SphereFace) are based on the cosine loss.

1.2. CosPlace with SphereFace

SphereFace [5] is a deep learning-based face recognition algorithm, so it is not directly related, as ArcFace, to visual geo-localization. Also in this part, the paper wants to show how the loss function may be integrated with CosPlace.
SphereFace aims to improve the robustness and accuracy of face recognition by using a deep neural network architecture that maps the feature representations of face images into a hypersphere with a specific angular margin. This margin helps to separate the feature representations of different faces and improve the discriminative ability of the model.

1.3. Implementation and results

Since CosPlace is developed for *Image Retrieval and Visual Geo-Localization*, meanwhile ArcFace and SphereFace are developed for *Face Recognition*, the model

are quite different eachother.
Despite so, it is possible to see that all the three have some similarities in the loss function. CosPlace uses the Large Cosine similarity loss [4], SphereFace and ArcFace uses a cosine based loss. As shown in [3], the three losses are specific cases of the general angular margin penalty-based loss:

$$L = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{s(\cos(m_1\theta_{y_i}+m_2)-m_3)}}{e^{s(\cos(m_1\theta_{y_i}+m_2)-m_3)}+\sum_{j=1,j\neq y_i}^{n}e^{s\cos\theta_j}}$$

where $m1$ is the parameter used by SphereFace, m_2 is the parameter used by ArcFace and m_3 is the parameter used by CosFace.

Loss function	sf-xs (test)	Tokyo-xs	Tokyo-night
CosFace	**R1: 52.3** R5: 66.3	**R1: 69.5** R5: 84.8	**R1: 49.5** **R5: 73.3**
SphereFace	R1: 50.5 R5: 65.0	R1: 68.9 **R5: 88.9**	R1: 46.7 R5: 72.4
ArcFace	R1: 51.9 **R5: 66.6**	R1: 67.3 R5: 80.6	R1: 47.6 R5: 70.5

Table 1. Results obtained with different loss functions

Due to limited computational resources, here are shown the results of the models trained on sf-xs for 3 epochs and 10.000 iterations per epoch. As shown in the table, only *Recall@1* and *Recall@5* are took in consideration. From the results, it is clear that CosFace leads to better R@1, where SphereFace and ArcFace provided slightly worse results. This may be due to the low number of epochs or miscalibration of the parameters.

2. Extensions

As mentioned, the main focus of this paper is to develop and test new extensions. From now on, **the model trained on sf-xs, with 3 epochs, 10.000 iterations per epoch and CosFace loss is considered as to be the *baseline model***.
The main faced challenges concern, in terms of results, robustness of the model to domain shifts and generalization.
The main shown extensions are:

- Gradient Reversal Layer to improve robustness to domain shifts, including night domains.
- Data augmentation to improve robustness to perspective changes or occlusions.
- Re-ranking method with GeoWarp to further improve the recalls.
- Testing different backbones to improve Visual Geo localization.
- Ensembles techniques.
- Multiscale techniques.

2.1. Domain shift with Gradient Reverse Layer

Gradient Reversal Layer, as described in "Unsupervised Domain Adaptation by Backpropagation" [8], consists in placing a Gradient Reversal Layer between the feature extractor and domain classifier.

Intuitively, Gradient Reversal Layer acts as an identity function during forward propagation and, during backpropagation, the output is multiplied by a certain factor that generally is -1.

The pipeline that involves Gradient Reversal Layer is shown below:

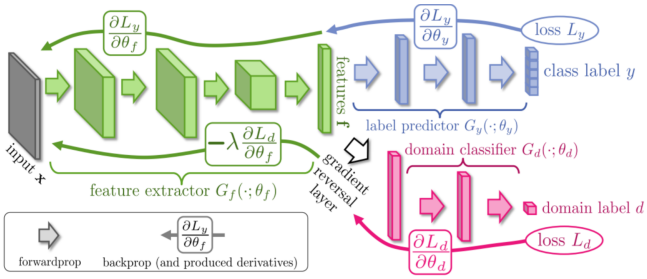

Figure 1. Domain Adaptation with GRL

In other words, during backpropagation, the output of Gradient Reversal Layer is basically leading to the opposite of Gradient descent that is performing gradient ascent on the feature extractor with respect to the classification loss of Domain predictor.

In this case, GRL is exploited to adapt the domain to a target domain made only by few night images of tokyo-night. This aims to improve the robustness of the model when the query is a night image.
It is interesting to see how, without any other changes to the original baseline, GRL alone leads to a significant improvement in terms of recalls.
A table with the results is shown as follows:

	baseline		with GRL	
Dataset	R1	R5	R1	R5
sf-xs(test)	52.3	66.3	54.7	68.1
tokyo-xs	69.5	84.8	73.0	87.3
tokyo-night	49.5	73.3	58.1	78.1

Table 2. baseline and GRL improvements

2.2. Data augmentation

Generally, Data augmentation is used to increase the size and diversity of the dataset by applying transformations to existing data in order to avoid overfitting and improve the model's generalization performance.
Specifically, in this case, two different types of Data augmentation are employed.

1. Data augmentation on night train queries to improve the recalls on Tokyo-night.
2. ...more

They are analyzed better in the following subparagraphs.

2.2.1 First type: Night Data Augmentation

When the training dataset is mostly characterized by daylight images, it is challenging to adapt the model to work with night images. In this section, it is proposed a very simple change: adjust the brightness of the image with a certain brightness factor in order to have darker images respect to the originals. Here are some examples:

(a) Original image (b) Brightness factor 0.3

(c) Brightness factor 0.2 (d) Brightness factor 0.1

Figure 2. Impact of different brightness factor

	Baseline		GRL	
	R1	R5	R1	R5
None	49.5	73.7	58.1	78.1
First type, factor=0.1	52.4	70.5	60.0	78.1
First type, factor=0.15	51.4	71.4	**60.0**	80.0
First type, factor=0.2	**55.2**	74.3	58.1	**81.0**
First type, factor=0.3	52.4	74.3	57.1	80.0
First type, factor=0.4	51.4	**75.2**	60.0	80.0

Table 3. Results on tokyo-night with First Type Data Augmentation

From now on, if not specified, when the First Type of DA is involved, the brightness factor is fixed to 0.15. The changes are performed at test time on the training queries and this allows to get slightly better performance on night domain images with almost zero additional computational costs.
In particular, since tokyo-night uses tokyo-xs dataset, this first type of data augmentation is performed only on tokyo-xs dataset (at test time).

3. Re-ranking method with GeoWarp to further improve the recalls

Visual Geo-localization is the task of recognize where a given photo was taken, this photo is usually called a query. Given a large dataset of images the goal is to find the most similar to the query. The problem is approached using local or global features, GeoWarp [2] takes the best of each of both.

3.1. GeoWarp

At test time the goal is to find a number of predictions using a CNN with global features, in this paper it is used *GeM pooling*, then re-rank this predictions using GeoWarp. In order to do this, the couple query-prediction is passed to an *Homography Regression Module* which estimates the core points in each images.
This points are then passed to an *Homographic Projection* that estimates the similarity between the two images.
Finally, given the warped images, with an encoder it is possible to extract the dense local features which are used to compute the similarity between the warped images.

Figure 3. Geoward architecture. The warping regression module estimates two quadrilaterals from the query and the prediction images. The images are then warped with a homography, and thir similarity is computed on their deep local features.

Dense local features are highly informative but do not resist to POV changing, that is the reason why a *Homographic Projection* is applied.

3.1.1 Train of the Warping Regression Module

The train is executed with a *self-supervised* way, since there are no large dataset with homography labels.

1. An image, from the training set, is taken and two intersecting trapezoids are drawn.

2. An homographic projection is applied on both to extract two images

3. It is now possible to have two images with known intersection, that we can use to train the CNN to estimate their intersection

It is significant to enline that *self-supervised training* data helps train the network but it **does not** represent real world distributions, this is because query-predictions pairs at test-time have temporal variation. To overcome this issue, it is possible to mine query-positive pairs in a *weakly supervised manner*, then use them to better train the network, introducing two losses.

- **Feature wise loss**: compute the pairwise warping, their dense local features and minimize the distance between local features.

$$L_{fw} = \sum_{i=0}^{w_f-1} \sum_{j=0}^{h_f-1} (\hat{f}_q(i,j)^T \hat{f}_g(i,j))^2$$

- **Consistency loss**: after the application of random augmentations to the pair of images, the second step is to estimate homographies for augmented pairs and use them to estimates pseudo-labels, and minimize the distance among them.

$$L_{cons} = \frac{1}{2N}(\tau_i^{-1}(\tau_i(I_q), \tau_i(I_g))) - [t_q^*, t_g^*])^2 + (\tau_i^{-1}(\tau_i(I_g), \tau_i(I_q))) - [t_g^*, t_q^*])^2$$

3.1.2 Experiments conducted

The experiments are conducted using *Resnet18* as backbone and *GeM pooling* for feature extraction. The training is performed with three epochs of 10.000 iterations each, and the training set used to train is *sf-xs*.

	sf-xs(test)		tokyo-xs		tokyo-night	
Model	R1	R5	R1	R5	R1	R5
Baseline	52.3	**66.3**	69.5	**84.8**	49.5	**73.3**
ResNet18 + GeoWarp	**60.2**	66.1	**76.2**	83.2	**60.0**	70.5

Table 4. Results with GeoWarp respect to baseline

It is possible to see that GeoWarp significantly improves the R@1 in all the three datasets, especially with *sf-xs* and *Tokyo-night* datasets but it slightly worsens R@5 in all three datasets.
This can lead to the fact that GeoWarp can achieve better results using a different network. A descriptor such as *GeM* can proves that GeoWarp is also viable for large-scale problems.

4. Testing different backbones to improve Visual Geo localization

All the experiments conducted until now are performed with *ResNet18* as backbone and *GeM pooling* as descriptor. In the following section, it is possible to find tests with different backbones.

4.1. EfficientNetV2S

EfficientNetV2 [6] are a family of image classification models, which achieve better parameter efficiency and faster training speed than prior arts. Built upon EfficientNetV1, our EfficientNetV2 models use neural architecture search (NAS) to jointly optimize model size and training speed, and are scaled up in a way for faster training and inference speed.

EfficientNetV2-S is pre-trained with ImageNet1K and it is a new version of the previous ones, with an higher complexity.

	sf-xs(test)		tokyo-xs		tokyo-night	
Model	R1	R5	R1	R5	R1	R5
Baseline	52.3	66.3	69.5	84.8	49.5	73.3
ResNet18 + GeoWarp	60.2	66.1	76.2	83.2	60.0	70.5
ResNet18 + GRL	54.7	68.1	73.0	87.3	58.1	78.1
EfficientNetV2s	62.5	74.5	80.0	90.8	73.3	**87.6**
EfficientNetV2s + Geowarp	**67.1**	73.3	**84.1**	90.5	**75.2**	83.8
EfficientNetV2s + GRL	63.0	**74.2**	81.0	**91.0**	73.3	84.8
EfficientNetV2s + GeoWarp + GRL						

Table 5. Results of different models. For tokyo-night, an additional First Type of Data Augmentation is employed with brightness_factor equal to 0.15.

5. Ensembles techniques

Ensembles is a machine learning technique that combines several base models in order to produce one optimal predictive model.

5.1. Model soups

Model Soup [7] is a new approach that instead of selecting the individual fine-tuned model which achieves the highest accuracy on the validation set, averages the weights of models fine-tuned independently, and refer to the result as a model soup.
As long as models have the same initial state, they can be combined in a *model soup*.

5.1.1 Usage of the model soup

To use model soup, different processes are needed:

- **Uniform Soup**: uniformly average all of the model weights.
- **Greedy Soup**: only using models that make an improvement.

The authors of the original paper [7] conducted a large random parameter search for fine-tuning a CLIP model on ImageNet.

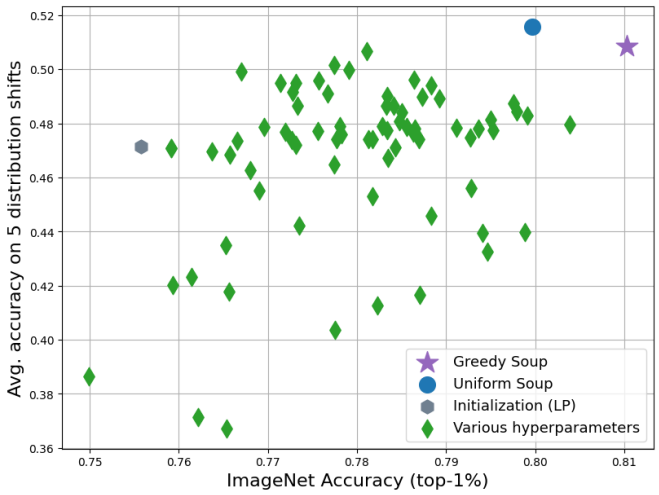

Figure 4. Tuning of the parameter.

The Greedy Soup outperforms every single fine-tuned model in terms of accuracy. *Uniform Soup*, is also good since it achieves the highest accuracy on five distribution shifts.

5.1.2 Effectiveness of the approach

Similar-initialized models with different seed or parameters might wind up on opposite sides of the optimum point of loss and error landscape, yet when compared to separate models interpolation the weights of the two solutions, can improve the accuracy.
The original research presents a novel approach to fine-tuning models that is far less heavy in terms of time and performances. Model Soups might be a terrific idea for time-saving methods.

5.1.3 Experiments conducted

The experiments have been conducted on the three model trained and mentioned earlier:

- CosPlace
- CosPlace with ArcFace
- CosPlace with SphereFace

Each model has been trained with *ResNet18* as backbone, *GeM* as descriptor, three epochs and 10.000 iterations for epoch. The training dataset is *sf-xs*.

Model Soups			
	sf-xs	*Tokyo-xs*	*Tokyo-night*
CosFace	52.3	69.5	49.5
ArcFace	51.9	67.3	47.6
SphereFace	50.5	68.9	46.7
Model Soup	52.0	**68.2**	47.7

Table 6. Results of the model soup with CosFace + ArcFace + SphereFace.

It is possible to see that the best performance are achieved with *Tokyo-xs*. In the original paper it is possible to see that there is a *large* improvement using Model Soup, this can lead to the fact that using a different backbone instead of *ResNet18* might lead to higher results, also the original weight might be changed, since in this paper default values are applied.

5.2. Ensemble

5.2.1 Ensemble with GeoWarp and GRL

In this technique it is used a combination of:

- GRL (2.1)
- GeoWarp (3.1)

Both models have been trained with *ResNet-18* as backbone and *GeM* as descriptor.
These were the two best models so an ensemble might lead to even better results.

	sf-xs(test)		tokyo-xs		tokyo-night	
Model	R1	R5	R1	R5	R1	R5
Geowarp	**67.1**	73.3	**84.1**	90.5	75.2	83.8
GRL	63.0	**74.2**	81.0	91.0	73.3	84.8
Ensembler	65.3	73.8	**84.1**	**91.1**	**76.2**	**86.7**

Table 7. Results of the ensemble

In the table above, it is possible to see that the ensemble technique actually lead to some imporvements in *tokyo-xs* and *tokyo-night*.

6. Multi-scale testing

Multi-scale testing means resize the input image into multiple scales to feed into the model, and average the scores as final prediction.
The first step was find an optimal resolution, then the images are passed through the model and the descriptors are then aggregated with one of the method proposed: **avg**, sum, max, min.
The resolution was fine tuned and the best result obtained was $[0.526, 0.588, 1, 1.7, 1.9]$ with **avg** method.

7. Conclusion

References

[1] G. Berton, C. Masone, and B. Caputo. Rethinking visual geo-localization for large-scale applications., 2022. CVRP. 1

[2] Gabriele Berton, Carlo Masone, Valerio Paolicelli, and Barbara Caputo. Viewpoint invariant dense matching for visual geolocalization, October 2021. 3

[3] J. Deng, J. Guo, J. Yang, N. Xue, I Kotsia, and S. Zafeiriou. Arcface: Additive angular margin loss for deep face recognition., 2022. TPAMI. 1, 2

[4] Z. Zhou X. Ji D. Gong J. Z. Z. Li W. Liu H. Wang, Y. Wang. Cosface: Large margin cosine loss for deep face recognition., 2018. CVRP. 2

[5] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song. Sphereface: Deep hypersphere embedding for face recognition., 2017. CVRP. 1

[6] Quoc V. Le Mingxing Tan. Efficientnetv2: Smaller models and faster training. 4

[7] Mitchell Wortsman, Gabriel Ilharco, Samir Ya Gadre, Rebecca Roelofs, Raphael Gontijo-Lopes, Ari S Morcos, Hongseok Namkoong, Ali Farhadi, Yair Carmon, Simon Kornblith, and Ludwig Schmidt. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time, 17–23 Jul 2022. 4, 5

[8] G. Yaroslav and V. Lempitsky. Unsupervised domain adaptation by backpropagation, 2015. 2