Machine Learning and Deep Learning Project Report, Summer 2024

Alessandro Coco
Politecnico di Torino
Student id: s333896
s333896@studenti.polito.it

Alessio Nebbia
Politecnico di Torino
Student id: s329335
s329335@studenti.polito.it

Pablo Torasso
Politecnico di Torino
Student id: s303196
s303196@studenti.polito.it

July 14, 2024

Abstract

In the context of visual place recognition for geolocalization, this work explores the implementation of the model proposed by the authors in [2]. Leveraging the setup proposed in their paper, we introduce our modifications on two fronts[1]. First, we adjust the behavior of miners, reclassifying certain negative images as positive, under specific hypotheses. Second, we perform a model comparison across combinations of miners and loss functions to find the best configuration. This evaluation is conducted both with the modified miner behavior and with the traditional off-the-shelf miner approach, which does not include the additional hypotheses. The results and analysis will provide insights into why our proposed approach appears promising for future research.

1 Introduction

Visual location recognition for geo-localization has become an important area of research in computer vision and robotics, attracting considerable attention due to its diverse applications in autonomous systems, immersive technologies and temporal image analysis. The challenge of accurately determining global position from visual data has far-reaching implications for areas such as self-driving vehicles, augmented reality experiences and historical urban development studies. These representations must be sufficiently detailed to distinguish between visually similar locations, while being compact enough to allow scalable solutions for large geographical areas.

Despite increasing research efforts, geolocalization remains a complex and multifaceted problem. The core of the challenge lies in developing robust algorithms that can accurately identify locations on a global scale, taking into account the large visual variability of real-world environments. This variability is due to various factors, including diurnal and seasonal light changes, weather conditions, vegetation cycles and the long-term development of cities. Basic scientific research in this field focuses on developing optimal representations of places that provide a balance between distinctiveness and computational power. This dichotomy poses a non-trivial optimization problem that requires innovative approaches for feature extraction, descriptor encoding, and matching strategies.

Recent advances in deep learning architectures based on convolutional neural networks (CNNs) have shown promise in overcoming these challenges. These approaches make use of large datasets and end-to-end training paradigms to learn hierarchical features that are more robust to changes in appearance. However,

[1] Source code available at GitHub repository

the question is whether these models can be generalized to unknown environments and whether they can deal with extreme changes in appearance.

2 Related Works

The field of geo-localization has undergone significant evolution, transitioning from traditional instance retrieval methods to more sophisticated approaches leveraging deep learning. Initially, techniques like SIFT [12], along with aggregation methods such as bag-of-visual-words [11], VLAD vectors, and Fisher vectors, were used for local feature representation [14, 8]. However, the advent of convolutional neural networks (CNNs) marked a turning point in visual place recognition. The approach in [13] represented a crucial advancement, demonstrating the superiority of end-to-end trained CNNs over pre-trained models for image representation. This work was further enhanced by [6], which introduced the GEM pooling layer that reaches significant improvements in feature extraction and representation.

Subsequent research has focused on various aspects to enhance geo-localization performance: Dataset Development: Large-scale, detailed datasets depicting locations across different timeframes with accurate ground truth have been created, enabling the training of more robust and generalizable models [13, 2, 9]. Architectural Innovations: Researchers have developed complex pipelines [13, 4], and new architectures that better utilize local and global image descriptors [13, 6, 4, 3]. Mining Techniques: Novel miners have been introduced to extract tuples from datasets more effectively [13, 10, 9]. Loss Functions: New loss functions have been developed to better evaluate how algorithms model datasets [13, 10, 9].

Despite these advancements, little attention has been paid to evaluating the range of positive instances as a hyperparameter in both model development and large-scale training datasets like GSV-Cities [2]. In real-world scenarios, query images received during actual model usage in a real application are likely to have a continuous distribution over the area covered by the database, with most queries taken some distance away from the closest database image. This realization has led to efforts in training models that are robust to non-co-habitation of query images and positive instances. The model leveraged on GSV defines as positives, instances that are taken at the exact same location of the query considering different orientations. However, this approach creates an unrealistic scenario of overlapping positions.

Later we will introduce a new approach in the training phase that considers positive instances, also those images taken within a 25-meter radius. This method aims to better prepare trained models for real-world applications by teaching them to handle more realistic spatial relationships between query and database images.

3 Datasets

3.1 Training dataset

The original GSV-Cities dataset is a comprehensive collection of images covering over 40 cities across all continents, spanning a 14-year period. It provides highly accurate ground truth data, including precise GPS coordinates and viewing directions for each image, which facilitates the creation of positive and negative pairs for training [2].

This work used a limited subset of the same dataset. GSV-Cities eXtra Small (GSV-XS) encompasses 23 cities from various continents, containing a total of 529,506 images captured over several years. The known approaches, such as the same authors one, exploit the dataset considering positive instances to be only those images taken from the exact same location as the query image. Starting from here we simulate a more permissive miners behaviour including also the images captured within a 25-meter radius of the query location as positive instances. This modification aims to align the training process more closely with the validation and test phase criteria, where the task is to retrieve images within a 25-meter range of the query.

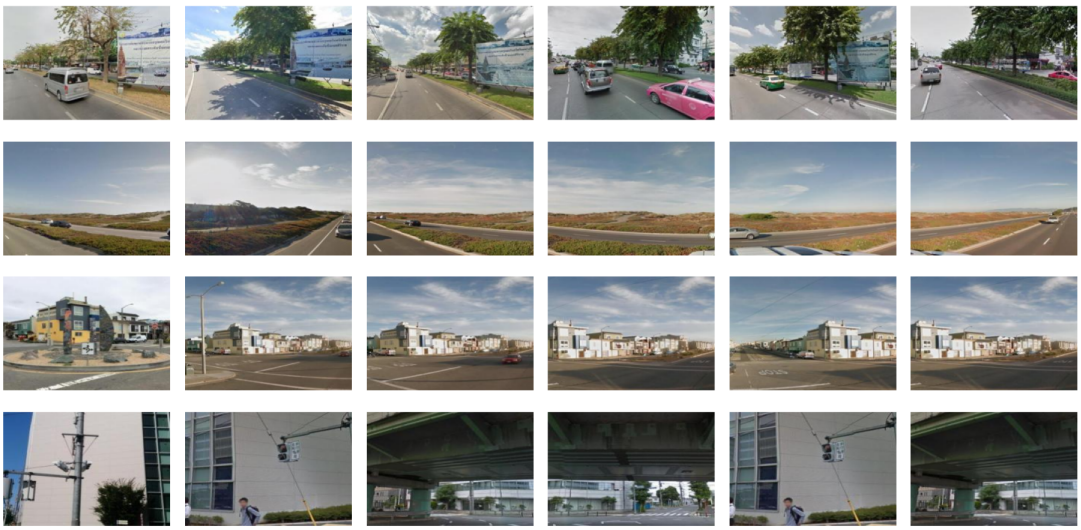


Figure 1: Examples from different datasets used in this study. Each row represents a location from a dataset. From top to bottom: GSV-Cities-XS, San Francisco-XS-Validation, San Francisco-XS-Test, and Tokyo-XS. The first column shows the query image, and the remaining columns show positive instances for that location. In GSV-Cities-XS and San Francisco-XS-Validation, query and positive images are taken from the same position with different orientations. In San Francisco-XS-Test and Tokyo-XS, query and positive images are from different locations and viewpoints, offering diverse scenarios for evaluating visual place recognition algorithms.

3.2 Validation dataset

For validation purposes, we employed the San Francisco eXtra Small (SF-XS) dataset. This dataset is a compact version of the larger San Francisco Landmark Dataset [5], which features images of San Francisco buildings accompanied by ground truth labels, geotags, and calibration data [9]. The SF-XS validation set is structured into two parts: 7,993 images designated as queries and 8,015 images serving as the reference database. This configuration allows for a comprehensive evaluation of our model's performance in a real-world urban environment, providing a robust test of its ability to recognize and localize landmarks across varying viewpoints and conditions.

3.3 Test datasets

For the testing phase, we utilized two distinct datasets as benchmarks for our models: the test set from the SF-XS dataset and the Tokyo eXtra Small dataset (Tokyo-XS). The Tokyo dataset consists of camera-phone query images of Tokyo, featuring significant variations in illumination (day, sunset, night) and structural changes in the scenes [1]. The Tokyo-XS test set, a subset of the original dataset, comprises 315 query images and 12,771 database images. The test set from the SF-XS dataset contains 1,000 query images and 27,191 database images.

Both the SF-XS test set and Tokyo-XS differ from the training and validation datasets in a crucial aspect: the majority of queries have ground truth images from the database that are not taken at the exact same location, but rather several meters apart from the query location. During the training and validation phases, the GSV-XS and SF-XS datasets are designed to teach the model to match queries with images taken at the exact same location. In contrast, the test phase using the SF-XS and Tokyo-XS

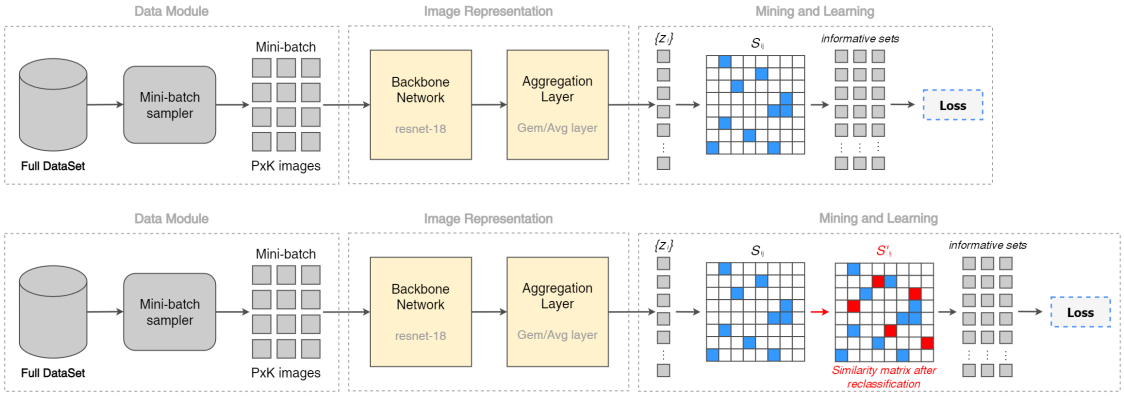


Figure 2: GSV-Cities original workflow (top) and new implementation with variations (bottom). Both workflows process batches of PxK images (P =number of places, K =images per place). The CNN generates a representation (Z_i) for each image, and a matrix (S_i) of pairwise similarities is computed. In the new implementation, additional positives are flagged in the matrix. This matrix identifies informative sets and feeds the selected loss function.

datasets evaluates the model's ability to match images and queries taken meters apart, demanding significant generalization from the model. This inconsistency between the validation sets, the final task in the test sets, and the training paradigm motivated us to introduce a modification in the mining process. During the training phase, we now consider not only images taken from the exact same location as positive instances but also images captured within a 25-meter radius. This change aims to encourage the model to generalize with respect to viewpoint and distance, better aligning with the real-world scenarios presented in the test sets.

4 Method Overview

Our work began with the implementation of the approach exposed in the research published by Ali-bey *et al.*: we developed an identical pipeline and architecture, as shown in figure 2, as proposed in [2]. Specifically, we first processed the available datasets to create ground truth data frames, enabling our sampler and miners to work effectively with our data sets. We then established a specific backbone for our model: a *ResNet-18* pre-trained on the ImageNet dataset, truncated at the convolutional layer 3, followed by an average pooling layer to generate the final embeddings.

Our approach is divided into two distinct phases as follows. The first phase is an empirical study of the results produced by various combinations of *off-the-shelf* miners, losses, and pooling layers. While, during the second phase, we introduced an offline mining technique to extend the set of positive instances during the training phase for some locations. Then, with a focus on an additional empirical stage, we evaluate the results of different combinations of miners, losses, and pooling layers while leveraging this new offline technique.

In the following sections, we will provide a detailed explanation of these two phases, including our methodology, findings, and analysis.

4.1 Phase 1: Miners, Losses and Pooling Layers

Our initial setup consisted of a pre-trained ResNet-18, truncated at the convolutional layer 3, followed by an average pooling layer. This configuration utilized the MultiSimilarity Miner and MultiSimilarity Loss [15]. Building upon this foundation, we introduced several variations on miners, losses and pooling layer:

Figure 3: Example of two locations in Bangkok from the GSV-Cities dataset, identified as additional positive instances by our methodology. The first location is represented by the first 3 images and the second location is represented by the last 3 images. These locations, 25 meters apart, show the same portion of highway from different perspectives.

Miners
- TripletMargin Miner
- PairMargin Miner

Losses
- TripletMargin Loss
- FastAP Loss [7]

Pooling Layer
- GeM layer [6]

In these setups, we opted not to freeze any layers, instead we further trained the entire model on the previously described datasets. We then conducted an empirical study, comparing the performance and accuracy of these newly trained models across various combinations of the aforementioned modules. This process resulted in a comprehensive table of results 5 for each combination, allowing us to evaluate the effectiveness of different architectural choices and training strategies.

4.2 Phase 2: A new offline mining technique

In a subsequent phase, we initiated a parallel empirical study utilizing the same architecture and combinations of modules (miners, losses, and pooling layers). However, we introduced a novel approach to labeling positive and negative instances during the training phase in the GSV-Cities dataset. Three primary factors drove this innovation:

1. The desire to create models more robust to variations in location appearances
2. Consideration of the end task in the test sets
3. Potential real-world applications

To achieve this, we implemented an offline mining strategy on the training dataset. In an offline environment, we calculated which other locations within the GSV-Cities dataset were within a 25-meter radius of each location. This process allowed us to create a comprehensive dictionary of additional positive instances for each location.

During training, we utilized this dictionary of additional positives to add new pairs of positive instances to the similarity matrix, as depicted in 2. This mechanism enabled the online miner to select positive instances from a significantly larger pool compared to the traditional approach used in our first empirical study 4.1. This implementation effectively enhanced the miner's ability to select more informative and varied image tuples.

Out of the 64,394 unique places in the training dataset, 4,317 places (6.7% of the locations) benefited from new positive instances. The impact of these additional positive instances during training is significantly influenced by the specific miner used. Even if these additional instances appear in the similarity matrix S_i, it doesn't guarantee they will be used as informative tuples for the loss function. Thus, the selection of miners is crucial in leveraging these additional instances. To increase the probability of using tuples formed by these new instances, an ad hoc sampler could have been implemented to feed specific locations into each batch. We focused primarily on the role and behavior of miners, leaving the implementation of a sampler for future work.

	San Francisco eXtra Small Test Dataset							
	Average Pooling Layer				GeM Pooling Layer			
	Old Methodology		Our Methodolgy		Old Methodology		Our Methodolgy	
	R@1	R@5	R@1	R@5	R@1	R@5	R@1	R@5
	Multi Sim. Loss		Multi Sim. Loss		Multi Sim. Loss		Multi Sim. Loss	
Multi S. Miner	22.10	37.00	19.50	35.50	25.10	40.30	24.20	37.70
Triplet M. Miner	20.10	32.70	20.30	31.10	21.1	36.5	23.6	38.5
Pair M. Miner	20.00	33.00	20.40	33.90	26.3	41.3	26.3	40.3
	Triplet M. Loss		Triplet M. Loss		Triplet M. Loss		Triplet M. Loss	
Multi S. Miner	20.70	33.50	20.40	33.20	26.6	39.6	26.5	40.5
Triplet M. Miner	16.10	31.00	15.80	30.30	23.8	37.9	22.6	36.9
	FastAp Loss		FastAp Loss		FastAp Loss		FastAp Loss	
Multi S. Miner	25.20	40.40	26.30	40.50	31.5	45.3	33.6	47.3
Triplet M. Miner	24.90	38.30	24.00	37.90	30.7	45.6	29.5	45.7
Pair M. Miner	25.30	40.80	26.00	41.70	31.7	46.7	32.1	47.9

	Tokyo eXtra Small Test Dataset							
	Average Pooling Layer				GeM Pooling Layer			
	Old Methodology		Our Methodolgy		Old Methodology		Our Methodolgy	
	R@1	R@5	R@1	R@5	R@1	R@5	R@1	R@5
	Multi Sim. Loss		Multi Sim. Loss		Multi Sim. Loss		Multi Sim. Loss	
Multi S. Miner	40.95	57.14	40.00	55.87	37.46	58.41	34.29	52.06
Triplet M. Miner	33.33	55.24	34.29	50.79	32.6	58.1	36.19	57.46
Pair M. Miner	36.83	55.87	31.11	53.65	39.05	59.05	42.54	61.9
	Triplet M. Loss		Triplet M. Loss		Triplet M. Loss		Triplet M. Loss	
Multi S. Miner	32.70	53.33	34.60	52.70	37.78	56.51	40.32	56.83
Triplet M. Miner	26.03	47.62	25.71	47.62	36.19	53.97	33.97	53.97
	FastAp Loss		FastAp Loss		FastAp Loss		FastAp Loss	
Multi S. Miner	37.46	54.92	37.46	54.60	48.89	64.49	45.71	65.08
Triplet M. Miner	37.46	58.73	40.00	57.78	46.03	65.4	49.52	66.35
Pair M. Miner	41.59	61.59	40.00	60.32	45.71	65.71	43.49	66.03

Table 1: Results obtained on the San Francisco eXtra Small dataset (top table) and on the Tokyo eXtra Small test dataset (bottom table), for every combination of miner, loss, pooling layer, and methodology of positive instances computation (old and our methodology). Only Recall@1 and Recall@5 are shown in these tables. Best results in red. For more complete tables of results refer to appendix B.

5 Results

In this section, we will discuss the results obtained from the two empirical studies introduced earlier and compare their scores. Recall@K, a standard metric for this task, is used to measure the proportion of relevant instances successfully retrieved within the top K results returned by the model. We will specifically focus on Recall@1 and Recall@5 in this section. However, more comprehensive tables, including Recall@10, Recall@15, Recall@20, and Recall@25, are presented in the appendix B. Tables 1 show the results for both test datasets and for all combinations of pooling layers, miners, and losses. These tables also facilitate easy comparison between results obtained using the old methodology for computing positive instances and those obtained using our new methodology. It should be noted that results from the combination of Triplet Margin Loss and Pair Margin Miner are not included in any results tables. The pairs selected by the Pair Margin Miner might not always form the most informative triplets for the Triplet Margin loss, and this combination would have required additional work to produce a competitive model compared to the other combinations. We decided not to invest time in this particular implementation.

Regarding the first empirical study, the GeM layer

consistently demonstrated superior accuracy compared to the average pooling layer across all combinations in both test datasets, confirming a well-known superiority[6]. Additionally, the FastAP loss achieved the best results across all combinations of miners, pooling layers, methodologies, and test datasets.

Regarding the second empirical study, our methodology achieved the best results for both Recall@1 and Recall@5 across both test datasets. Specifically, for the SF-XS test dataset, the best score achieved using the old methodology is 31.7 for Recall@1 and 46.7 for Recall@5. In contrast, using our new methodology, the best score is 33.6 for Recall@1 and 47.9 for Recall@5, surpassing the old methodology by almost 2 points for Recall@1 and more than 1 point for Recall@5. Similarly, for the Tokyo-XS test dataset, the best score using the old methodology is 48.49 for Recall@1 and 65.71 for Recall@5. With our new methodology, the best score is 49.52 for Recall@1 and 66.35 for Recall@5, surpassing the old methodology by 1 point for Recall@1 and nearly 1 point for Recall@5.

However, even though our methodology achieved the best results across all combinations, it did not outperform the old methodology in every combination, not demonstrating a clear superiority in this instance across all scenarios. We believe this limitation is due to our initial, basic implementation of this proposed novelty. As we will discuss in more detail in section 7, we believe that our approach can significantly outperform the old methodology with more refined implementations, utilizing advanced techniques and more informative datasets.

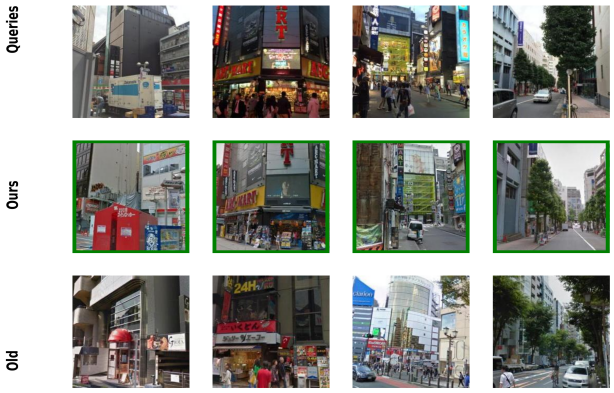


Figure 4: Examples of retrieval results for challenging queries on Tokyo. Each column corresponds to one test case: the query is shown in the first row, the top retrieved image using our best method (GeM pooling layer + NewTripletMargin miner + FastAP loss) in the second, and the top retrieved image using one of the best baseline (GeM pooling layer + TripletMargin miner + FastAP loss) in the last row. The green border corresponds to correct retrievals.

6 Conclusions

In this work, we have analyzed two different empirical studies. In the first instance, we examined results from state-of-the-art CNNs, pooling layers, miners, and losses. In the second part, we proposed an approach that modifies the positive images selection considering the miners behavior more permissive: this is possible by acting directly and a-posteriori on the output of the miners themselves. Based on the results obtained and reported in the Appendix B, our implementation shows interesting premises for future work, making us think that too stringent conditions have often been placed when selecting the positive ones. We believe that this innovation can further improve results on the geolocalization task when paired with larger state-of-the-art models. Additionally, we strongly believe that this new methodology performs best when large datasets are used to train the model: the high density of locations represented in a larger dataset increases the likelihood that additional positive instances could be formed from two different but close locations, both representing the same areas, enhancing the model's performance.

7 Future Work

Future works could explore several promising avenues and further improve models based on this new way of classifying positive instances. The following improvements should be explored:

Scaling to Larger Models and Datasets: We

hypothesize that applying our approach to larger, state-of-the-art models (such as ResNet-101) and more extensive datasets (like the full GSV-Cities dataset) will yield even more significant improvements. The increased capillarity of larger datasets should enhance the probability of intersection between additional positive instances, potentially leading to more robust and generalizable models.

Adaptive Radius for Positive Instance Selection: Future research could investigate the impact of an adaptive radius for selecting positive instances. Instead of a fixed 25-meter radius, the distance could be adjusted based on factors such as urban density, terrain type, or the visual complexity of the environment, or could be considered as a simple additional hyper-parameter. This could potentially lead to more context-aware and flexible models.

Computational Efficiency Analysis: A detailed study on the computational trade-offs of our approach would be valuable. While expanding the positive instance pool may improve accuracy, it likely increases computational demands during training. Understanding these trade-offs could help in developing optimized implementations for real-world applications.

Dedicated Sampler: a dedicated sampler to increase the probability of encountering our newly defined positive instances in each training batch that:

- Prioritizes query images with additional positive instances.
- Balances traditional and new positive instances in each batch.
- Dynamically adjusts sampling probabilities based on model performance.
- Implements curriculum learning, gradually introducing more challenging positives.

Penalization of pairs pointing in opposite directions: prioritize query-positive pairs where one image is pointed towards the location of the other, while excluding pairs where both images face away from each other (so, in which the two images do not depict same landscapes/buildings/objects).

References

[1] J. Sivic M. Okutomi T. Pajdla A. Torii, R. Arandjelović. 24/7 place recognition by view synthesis. 2015. 3

[2] A. Ali-bey and Giguère P. Chaib-draa, B. Gsv-cities: Toward appropriate supervised visual place recognition. 2022. 1, 2, 4

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

[4] Andre F. de Araújo Jack Sim Cao, Bingyi. Unifying deep local and global features for image search. 2020. 2

[5] D. M. Chen et al. City-scale landmark identification on mobile devices. *CVPR 2011*, 2011. 3

[6] O. Chum F. Radenovic, G. Tolias. Fine-tuning cnn image retrieval with no human annotation. 2018. 2, 5, 7

[7] Xide Xia Brian Kulis Stan Sclaroff Fatih Cakir, Kun He. Deep metric learning to rank. 2019. 5

[8] Jorge S´anchez Herv´e Poirier Florent Perronnin, Yan Liu. Large-scale image retrieval with compressed fisher vectors. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010. 2

[9] Barbara Caputo Gabriele Berton, Carlo Masone. Rethinking visual geo-localization for large-scale applications. 2022. 2, 3

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

[11] Andrew Zisserman Josef Sivic. Video google: A text retrieval approach to object matching in videos. *IEEE International Conference on Computer Vision*, 2003. 2

[12] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 2004. 2

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

[14] Andrew Zisserman Relja Arandjelovic. All about vlad. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2013. 2

[15] Weilin Huang Dengke Dong Matthew R. Scott Xun Wang, Xintong Han. Multi-similarity loss with general pair weighting for deep metric learning. 2019. 4

The below appendices describe the implementation details (A), and provide additional results (B).

A Implementations details

In this appendix, we provide a comprehensive overview of the implementation details for our model used in the geolocalization task. The following details outline the parameters and configurations employed in our experiments.

The data module was configured with a batch size of 200, ensuring efficient processing of data during training. Each place in the dataset is represented by a minimum of four images, and we ensured that images were shuffled within each city rather than across all cities. Images were resized to 320x320 pixels. We employed eight worker threads to speed up data loading.

The model was built using a ResNet18 backbone architecture pre-trained on ImageNet. We did not freeze any layers of the backbone, allowing the entire network to be fine-tuned during training. For the aggregation layer, we employed Average Pooling layer and Generalized Mean Pooling (GeM) layer.

Training hyperparameters were carefully chosen to optimize the model's performance. We set the learning rate to 0.0002 and used the Adam optimizer for its efficiency in handling sparse gradients. No weight decay was applied, but a momentum of 0.9 was included to accelerate gradient vectors in the right direction. The learning rate schedule included warmup steps and milestone epochs at 5, 10, 15, and 25, with a learning rate multiplier of 0.3 to adjust the learning rate dynamically during training.

For the loss function, we opted for the Multi Similarity, Triplet Margin, and FastAP loss. The mining strategies employed were the Multi Similarity, Triplet Margin, and Pair Margin Miner.

Training was set to run for a maximum of 10 epochs, with validation checks after every epoch. The dataloaders were reloaded every epoch to shuffle the data order, ensuring a diverse training process.

B Complete Results

In this section, we present tables displaying the complete results for all combinations of miners, losses, test datasets, pooling layers, and methodologies. These tables include Recall@K metrics with $K \in [5, 10, 15, 20, 25]$.

Following the tables, we provide a graphical representation of the results. This visualization categorizes the results by pooling layers, losses, and miners, showing outcomes achieved with the same miner but different methodologies on two test sets.

	San Francisco eXtra Small Test Dataset											
	Average Pooling Layer											
	Old Methodology						Our Methodolgy					
	R@1	**R@5**	**R@10**	**R@15**	**R@20**	**R@25**	**R@1**	**R@5**	**R@10**	**R@15**	**R@20**	**R@25**
	Multi Similarity Loss											
Multi S. Miner	22.10	37.00	43.80	46.70	49.40	51.00	19.50	35.50	40.90	45.20	48.50	50.80
Triplet M. Miner	20.10	32.70	38.80	42.50	45.90	48.60	20.30	31.10	37.10	41.60	44.60	47.50
Pair M. Miner	20.00	33.00	39.00	42.80	46.00	49.10	20.40	33.90	39.70	43.10	46.00	48.20
	Triplet Margin Loss											
Multi S. Miner	20.70	33.50	39.30	43.50	46.70	49.90	20.40	33.20	39.20	43.70	46.90	49.20
Triplet M. Miner	16.10	31.00	36.60	41.60	45.00	47.50	15.80	30.30	36.40	40.30	43.50	46.40
	FastAp Loss											
Multi S. Miner	25.20	40.40	47.50	51.80	53.50	54.70	26.30	40.50	46.90	50.50	53.50	55.60
Triplet M. Miner	24.90	38.30	44.60	48.70	52.00	54.30	24.00	37.90	44.70	49.10	52.00	54.30
Pair M. Miner	25.30	40.80	47.30	51.70	54.70	56.40	26.00	41.70	48.60	52.50	54.10	55.50

Table 2: Results for all the combinations of miners and losses, using Average pooling layer for San Francisco test dataset.

	San Francisco eXtra Small Test Dataset											
	GeM Pooling Layer											
	Old Methodology						Our Methodolgy					
	R@1	**R@5**	**R@10**	**R@15**	**R@20**	**R@25**	**R@1**	**R@5**	**R@10**	**R@15**	**R@20**	**R@25**
	Multi Similarity Loss											
Multi S. Miner	25.10	40.30	47.60	50.60	52.40	54.40	24.20	37.70	45.40	49.60	52.10	54.30
Triplet M. Miner	21.1	36.5	41.6	45.8	48.8	50.5	23.6	38.5	45	48.8	51	53.3
Pair M. Miner	26.3	41.3	47.4	51.6	54	55.8	26.3	40.3	46.9	51.4	53.5	55.6
	Triplet Margin Loss											
Multi S. Miner	26.6	39.6	46.3	50.3	53.8	56.5	26.5	40.5	47.5	51.9	54.1	56.8
Triplet M. Miner	23.8	37.9	43.6	48.4	51.1	53	22.6	36.9	43.2	47	49.8	52.3
	FastAp Loss											
Multi S. Miner	31.5	45.3	51.2	55.7	58.6	61.8	33.6	47.3	53.3	56.8	59.5	61.4
Triplet M. Miner	30.7	45.6	51.5	56.4	59.5	62	29.5	45.7	51.1	54.8	58.5	61
Pair M. Miner	31.7	46.7	53	57.2	59.7	61.7	32.1	47.9	54.4	57.7	60.9	62.5

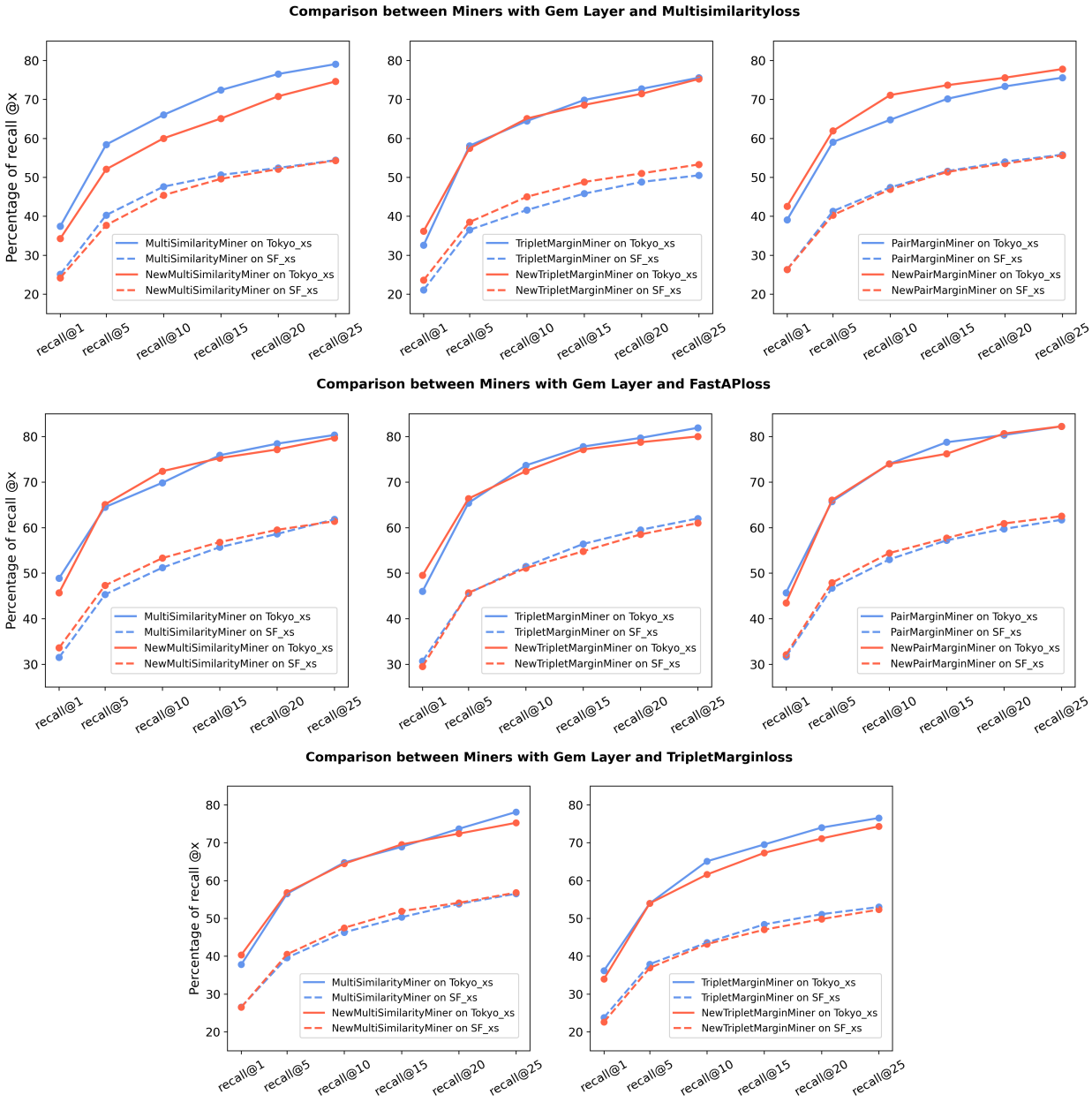
Table 3: Results for all the combinations of miners and losses, using GeM pooling layer for San Francisco test dataset.

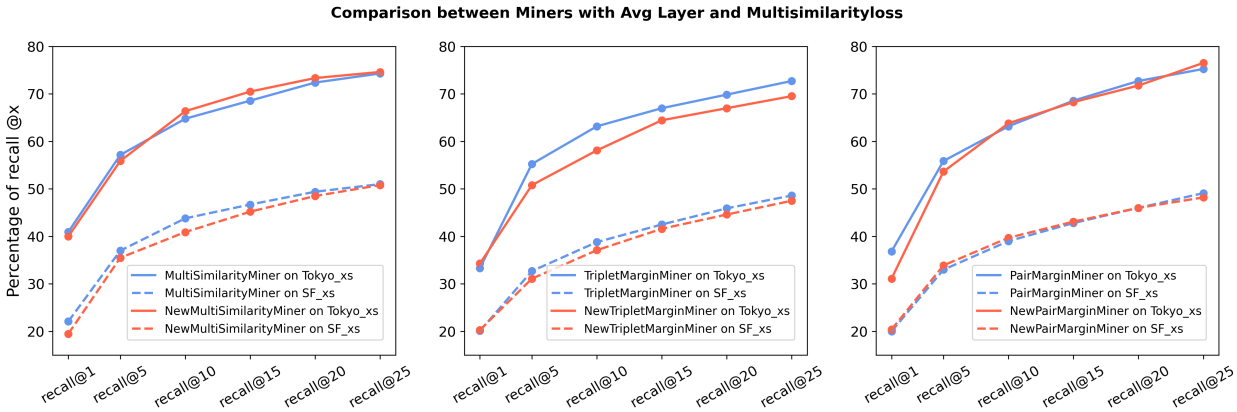
	Tokyo eXtra Small Test Dataset											
	Average Pooling Layer											
	Old Methodology						Our Methodolgy					
	R@1	**R@5**	**R@10**	**R@15**	**R@20**	**R@25**	**R@1**	**R@5**	**R@10**	**R@15**	**R@20**	**R@25**
	Multi Similarity Loss											
Multi S. Miner	40.95	57.14	64.76	68.57	72.38	74.29	40.00	55.87	66.35	70.48	73.33	74.60
Triplet M. Miner	33.33	55.24	63.17	66.98	69.84	72.70	34.29	50.79	58.10	64.44	66.98	69.52
Pair M. Miner	36.83	55.87	63.17	68.57	72.70	75.24	31.11	53.65	63.81	68.25	71.75	76.51
	Triplet Margin Loss											
Multi S. Miner	32.70	53.33	61.59	66.98	71.11	74.60	34.60	52.70	61.59	66.98	71.11	75.24
Triplet M. Miner	26.03	47.62	59.05	64.13	67.94	71.75	25.71	47.62	58.73	65.08	70.48	73.30
	FastAp Loss											
Multi S. Miner	37.46	54.92	63.49	67.94	71.43	74.29	37.46	54.60	64.44	68.89	71.75	75.56
Triplet M. Miner	37.46	58.73	69.52	73.97	75.87	78.10	40.00	57.78	66.67	72.38	76.51	77.14
Pair M. Miner	41.59	61.59	70.16	73.97	78.73	82.54	40.00	60.32	68.57	75.56	78.73	81.27

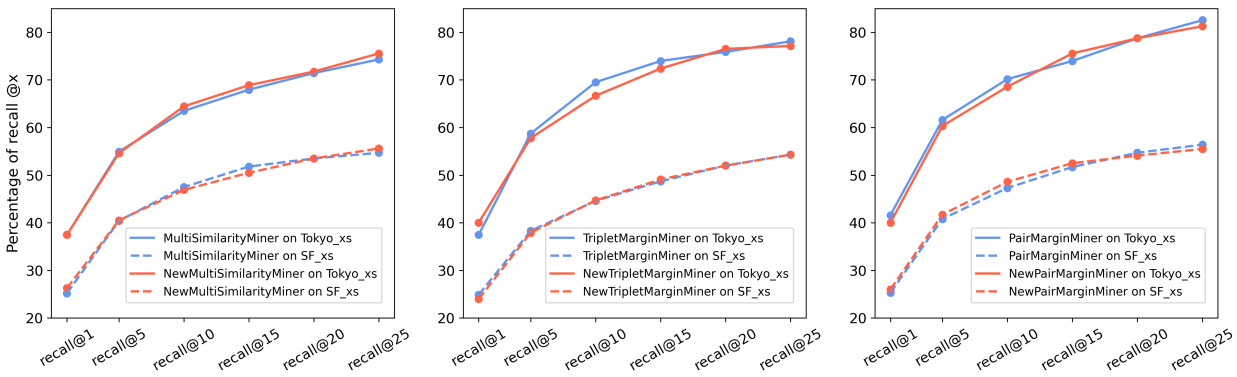
Table 4: Results for all the combinations of miners and losses, using Average pooling layer for Tokyo test dataset.

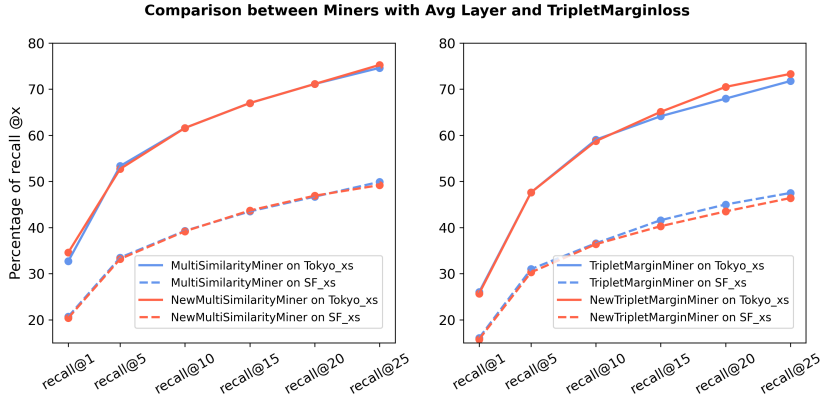
	Tokyo eXtra Small Test Dataset											
	GeM Pooling Layer											
	Old Methodology						Our Methodolgy					
	R@1	**R@5**	**R@10**	**R@15**	**R@20**	**R@25**	**R@1**	**R@5**	**R@10**	**R@15**	**R@20**	**R@25**
	Multi Similarity Loss											
Multi S. Miner	37.46	58.41	66.03	72.38	76.51	79.05	34.29	52.06	60.00	65.08	70.79	74.60
Triplet M. Miner	32.6	58.1	64.44	69.84	72.7	75.56	36.19	57.46	65.08	68.57	71.43	75.24
Pair M. Miner	39.05	59.05	64.76	70.16	73.33	75.56	42.54	61.9	71.11	73.65	75.56	77.78
	Triplet Margin Loss											
Multi S. Miner	37.78	56.51	64.76	68.89	73.65	78.1	40.32	56.83	64.44	69.52	72.38	75.24
Triplet M. Miner	36.19	53.97	65.08	69.52	73.97	76.51	33.97	53.97	61.59	67.3	71.11	74.29
	FastAp Loss											
Multi S. Miner	48.89	64.49	69.84	75.87	78.41	80.32	45.71	65.08	72.38	75.24	77.14	79.68
Triplet M. Miner	46.03	65.4	73.65	77.78	79.68	81.9	49.52	66.35	72.38	77.14	78.73	80
Pair M. Miner	45.71	65.71	73.97	78.73	80.32	82.22	43.49	66.03	73.97	76.19	80.63	82.22

Table 5: Results for all the combinations of miners and losses, using GeM pooling layer for Tokyo test dataset.


Comparison between Miners with Gem Layer and Multisimilarityloss
Percentage of recall @x
MultiSimilarityMiner on Tokyo_xs
MultiSimilarityMiner on SF_xs
NewMultiSimilarityMiner on Tokyo_xs
NewMultiSimilarityMiner on SF_xs
TripletMarginMiner on Tokyo_xs
TripletMarginMiner on SF_xs
NewTripletMarginMiner on Tokyo_xs
NewTripletMarginMiner on SF_xs
PairMarginMiner on Tokyo_xs
PairMarginMiner on SF_xs
NewPairMarginMiner on Tokyo_xs
NewPairMarginMiner on SF_xs
recall@1
recall@5
recall@10
recall@15
recall@20
recall@25
Comparison between Miners with Gem Layer and FastAPloss
Comparison between Miners with Gem Layer and TripletMarginloss


Comparison between Miners with Avg Layer and Multisimilarityloss
Percentage of recall @x
MultiSimilarityMiner on Tokyo_xs
MultiSimilarityMiner on SF_xs
NewMultiSimilarityMiner on Tokyo_xs
NewMultiSimilarityMiner on SF_xs
TripletMarginMiner on Tokyo_xs
TripletMarginMiner on SF_xs
NewTripletMarginMiner on Tokyo_xs
NewTripletMarginMiner on SF_xs
PairMarginMiner on Tokyo_xs
PairMarginMiner on SF_xs
NewPairMarginMiner on Tokyo_xs
NewPairMarginMiner on SF_xs
recall@1
recall@5
recall@10
recall@15
recall@20
recall@25


Comparison between Miners with Avg Layer and FastAPloss
Percentage of recall @x
MultiSimilarityMiner on Tokyo_xs
MultiSimilarityMiner on SF_xs
NewMultiSimilarityMiner on Tokyo_xs
NewMultiSimilarityMiner on SF_xs
TripletMarginMiner on Tokyo_xs
TripletMarginMiner on SF_xs
NewTripletMarginMiner on Tokyo_xs
NewTripletMarginMiner on SF_xs
PairMarginMiner on Tokyo_xs
PairMarginMiner on SF_xs
NewPairMarginMiner on Tokyo_xs
NewPairMarginMiner on SF_xs
recall@1
recall@5
recall@10
recall@15
recall@20
recall@25


Comparison between Miners with Avg Layer and TripletMarginloss
Percentage of recall @x
MultiSimilarityMiner on Tokyo_xs
MultiSimilarityMiner on SF_xs
NewMultiSimilarityMiner on Tokyo_xs
NewMultiSimilarityMiner on SF_xs
TripletMarginMiner on Tokyo_xs
TripletMarginMiner on SF_xs
NewTripletMarginMiner on Tokyo_xs
NewTripletMarginMiner on SF_xs
recall@1
recall@5
recall@10
recall@15
recall@20
recall@25