POLITECNICO DI TORINO

Master of Science in Data Science and Engineering

Master's Degree Thesis

Approaching visual geo-localization through classification

Supervisors:
Prof. Barbara Caputo
Prof. Carlo Masone

Co-Supervisors:
Dott. Gabriele Berton
Dott. Gabriele Trivigno

Candidate
Juan Manuel
Aragon Armas
s.291466

Academic Year 2022-2023

Summary

The increasing interest towards visual geolocation (or visual place recognition) has been noticeable in recent years. A fast and efficient system able to identify a particular place around the globe, that uses only the visual content of a query image, has been requested in many different fields, going from virtual reality to self driving cars and exploratory robots. Advances in artificial intelligence and dense open source datasets boosted the research community on proposing a set of alternative ways to tackle this challenge, being *retrieval* and *classification* two of the most diffused approaches, each of which propose its own advantages and weaknesses. Although, there is no absolute winner when comparing the existing approaches, more efficient systems can be obtained when combining more than one technique through the processing pipeline.

The objectives of this project were focused on the analysis of the *classification* approach. Primarily consisting of partitioning the geographical area of interest into geographical cells and developing machine learning models able to associate a particular set of images to its correspondent geo-class. The goal of this project was to study, implement and deploy existing approaches, popular among the existing literature, but whose authors in some cases never released the source codes to replicate their results. A second part of the project consisted of developing a deep, complete and updated comparison of the existing methods, studying their performances on highly dense datasets of urban scenarios.

Contents

1 Introduction 5
- 1.1 General Description . . . 6
- 1.2 Objectives . . . 7
- 1.3 My Contribution . . . 7

2 Related works 9
- 2.1 Visual Geolocation . . . 9
- 2.2 Approaches to VG . . . 10
  - 2.2.1 Image Retrieval . . . 11
  - 2.2.2 Classification . . . 13
  - 2.2.3 Hybrid and Paired Methods . . . 14
- 2.3 Image Retrieval Framework . . . 15
  - 2.3.1 Feature Extraction . . . 15
  - 2.3.2 Similarity Search . . . 19
  - 2.3.3 Indexing & Clustering . . . 22
  - 2.3.4 Reranking . . . 23
- 2.4 Classification Framework . . . 23
  - 2.4.1 Discrete Global Grid Systems . . . 26
  - 2.4.2 PlaNet . . . 30
  - 2.4.3 CPlaNet . . . 31
  - 2.4.4 Hierarchical Geo-Estimation . . . 33
  - 2.4.5 MvMF . . . 34

I 37

3 Datasets 38
- 3.1 YFCC100M . . . 39

3.1.1 MP16 . . . 39
3.1.2 YFCC_26k . . . 41
3.2 Im2GPS & Im2GPS3k . . . 41
3.3 Pitts250k . . . 44
3.3.1 Pitts30k . . . 44
3.4 Tokyo 24/7 . . . 45
3.5 San Francisco eXtra Large . . . 46

II **51**

4 Replicating the results 52

5 Experiments on SF-XL 54
5.1 PlaNet . . . 54
5.2 HGE . . . 55
5.3 CPlaNet . . . 57
5.4 Comparison with retrieval approaches . . . 58

6 Experiments on smaller datasets 61
6.1 Pitts250k . . . 62
6.1.1 PlaNet on Pitts250k . . . 62
6.1.2 HGE on Pitts250k . . . 63
6.1.3 CPlaNet on Pitts250k . . . 63
6.2 Pitts30k . . . 64
6.2.1 PlaNet on Pitts30k . . . 64
6.2.2 HGE on Pitts30k . . . 66
6.2.3 CPlaNet on Pitts30k . . . 66
6.3 Tokyo 24/7 . . . 68
6.3.1 PlaNet on Tokyo24/7 . . . 68
6.3.2 HGE on Tokyo24/7 . . . 68
6.3.3 CPlaNet on Tokyo 24/7 . . . 69

7 Conclusions 71

"A place is more than just its physical attributes, it is the essence of its history, culture, and traditions that gives it its true identity."
[ChatGPT, the essence of a place]

Chapter 1

Introduction

Location systems are technologies that allow for the determination of the physical position of an object or person. These systems have become increasingly important in a wide range of applications, from navigation and tracking to emergency response and logistics. There are several different types of location systems, including global positioning systems (GPS), cellular triangulation, WiFi-based systems, and newer emerging approaches like visual geo-location.

Traditional systems follow a satellite-based approach, on which the main idea is to use a network of satellites in orbit around the Earth and a receiver on the ground to determine its precise location. In practice, the satellites broadcast a signal that contains information about their precise location, as well as the time the signal was transmitted. When the receiver on the ground receives signals from the satellites, it can use the time-stamped signals to calculate its precise position through a process called trilateration.

Satellite-based location systems are highly accurate, with GPS providing location information with an accuracy of a few meters. Other satellite-based location systems, such as the European Galileo system and the Chinese Beidou system, are also becoming more widely available and increasing their accuracy. However, there are several disadvantages to these systems:

- **Dependency on technology**: Satellite-based location systems rely on technology and infrastructure that can be vulnerable to disruption or failure. For example, if a GPS satellite fails, it can affect the accuracy and availability of location services.

- **Limited coverage**: Satellite-based location systems require a clear line of sight to the sky, which can be obstructed in urban areas with tall buildings, tunnels, or in densely forested areas. This can limit their effectiveness in certain situations.

- **Accuracy**: While GPS is generally accurate, it can sometimes be affected by environmental factors such as atmospheric conditions or interference from other devices, which can affect its accuracy.

- **Security**: Satellite-based location systems can be vulnerable to hacking or other security threats, which can compromise the privacy and security of users.

- **Battery**: Using satellite-based location services can consume a lot of battery power on mobile devices, which can limit their usefulness in situations where power is limited or not available.

- **Cost**: While GPS is freely available for civilian use, some satellite-based location systems can be expensive to use or require special equipment, making them less accessible to some users.

At this extent and given the astonishing advances in machine learning, deep learning and computer vision, we could ask ourselves if it is possible to develop alternative location systems, that do not depend on a complex and expensive infrastructure of satellites orbiting around the globe, but exploiting the large amounts of images coming from highly diffused social networks, and by using only visual cues such as landmarks, street signs, and building facades, could we teach a computer to accurately recognize the location of where the scene was taken?

1.1 General Description

This document is intended to work as a guide to explore the latest advances in deep-learning to the visual geolocation task, relative to the classification approach. Aiming to provide a fair comparison of the most recent ways to tackle the problem of understanding the location of a scene only by analyzing its visual content. Until now, the existing benchmarks lack key aspects that were targeted to be solved through this project:

1. **Homogeneity**: Evaluating the performance of the different methods on the same datasets and computational conditions.

2. **City-wide performances**: Applying the existing methods on highly dense datasets from large city-wide maps, where classes are intended to cover areas of just a few blocks, and the visual gap between adjacent classes is minimum.

3. **Multiple-scenarios analysis**: Testing their efficiency on a variety of different highly diffused datasets, well known among the research community, including visual elements from different sources, environmental conditions, locations, and dataset-lengths.

4. **Replicability**: Releasing the source code of the implemented methods, providing clear documentation of either the hyper-parameters used in the partitioning and in the training process, as the computational resources employed during the analysis.

1.2 Objectives

The precise roadmap of the project consisted of:

1. Implementing a number of existing papers as baselines.

2. Use the best baseline method to train a large neural network on large-scale datasets.

3. Collect a bunch of well known datasets on which to evaluate the speed/accuracy trade-off between retrieval-based VG and classification-based VG.

1.3 My Contribution

This project aims to present a deep analysis in the visual geolocation task, focusing on the most significant progress done in recent years under the classification-based framework. The main part of this project consisted of developing machine learning algorithms capable of classifying an image into a geographic cell, generated by a variety of partitioning strategies at different granularity, following existing baselines to which the source code was not released.

Then, by comparing their performance with new SotA methods, this document provides a fair and updated benchmark through a variety of datasets, at either large city-wide scale and planet-wide scale.
Finally, releasing the code to the open public for further research activities, specially those requiring the comparison of the implemented approaches with newer and more efficient methodologies.

Chapter 2

Related works

The following chapter aims to describe the general overview of the Visual Geolocation task, describing its uses and limitations, as well as the most common approaches discussed in the literature, which includes the classification approach.

2.1 Visual Geolocation

Visual Geolocation (VG) or Visual Place Recognition (VPR) could be described as the process of finding any kind of geographical information, such as geographical coordinates or orientation, from a visual media source, that can be defined as a single image or a sequence of frames taken on a relatively close period of time. In this process, only the visual cues and the content of the visual elements are considered.
Although some authors propose different definitions of this complex process, influenced by the kind of approach they use to solve it, they all coincide with the main idea of the task. E.g. Hays and Efros [1] give a classification-based definition, referring to this process as obtaining the probabilistic distribution of where a picture was taken in a limited geographical space, over a set of predefined regions or classes. On the other hand, Zamir and Shah [2] describe this process as retrieving the location of a visual scene by analyzing the content-based similarities it shares with a set of well-known items.

This challenging process has been gaining more interest inside the research community, which seeks for alternative ways to build faster, more reliable, and more robust autonomous navigation systems, that is at the

moment highly dependent on satellite-based technology, such as GPS sensors and space communication devices, constantly sharing data with far away transponders that constitute a complex and expensive infrastructure. In some cases, VPR seems to be the most promising solution to apply, especially in environments where satellite communication is limited, mostly affected by the presence of physical obstructions (buildings or tunnels), atmospheric inference (solar flares, ionospheric scintillation), electronic interference with other electromagnetic waves, and weather conditions (rain, fog or snow).

This particular approach takes the advantage of highly diffused RGB cameras worldwide as well as the growing media community that makes it easier to find accurate geo-tagged visual elements, with no significant additional costs. And of course, the recent discoveries in machine learning and artificial intelligence constitute a key aspect of the development of these promising methods.

Although modern technologies had helped efficiently in achieving extraordinary results in this field, this task is still vulnerable to visual limitations that add an extra notch to its complexity. For example, dynamic environments influenced by weather conditions, and significant architectural modifications, in the case of urban scenarios, are difficult to address and have a direct impact on the performance of the proposed solutions. Not to mention the large computational resources required either in the development of the model or during the inference process. These problems together with the demand for fast inference solutions have shaped the development of modern approaches which are discussed later in this chapter.

2.2 Approaches to VG

Inside the research community, plenty solutions have been proposed in recent years, addressing the problem of visual geolocation through different approaches, each of them focused on solving a specific problem of the task. Recent discoveries have been addressed to improve this process along the following aspects:

1. **dynamic environments**: moving objects that do not constitute part

of a specific place but can be found on a particular scene, as pedestrians or vehicles as in urban scenarios or any fauna and flora element that modifies its appearance based on seasonal stages.

2. **scene conditions**: The best iconic example is well described by the different appearance a scene can show during day and night. Without introducing substantial changes in the urban architecture of a place, just by varying the position and direction of light sources can impact negatively on the performance of VG solutions. Not to mention distortions introduced by weather conditions can not only modify the appearance of the surroundings but can interfere in the collection of accurate visual elements as in the application of autonomous vehicles.

3. **inference time**: It will be discussed how some approaches introduced in the literature achieve SotA results in terms of accuracy but are not salable to larger applications, since their performance has share a linear dependency with the amount of visual elements used to provide the accurate results, either at a World-wide or city-wide scale.

4. **variable orientations**: One of the most crucial aspects in achieving high performance has to do with models being tolerant to variations in the location at which a scene is captured. The complexity of this particular task grows with the granularity of the desired results.

As any challenging task, it is difficult to find a solution that rules over all the challenging aspects that have been described. Some approaches are faster during inference time but are less scalable than others and viceversa. And sometimes the best results are achieved when different approaches are fused at different stages of the processing pipeline.

2.2.1 Image Retrieval

The idea behind the Image Retrieval process is to analyze a user's query and to retrieve, usually from inside a large database, a particular image that best fits with the requested output. The input sent by the user could be either a textual description of the image to be retrieved or another image from which it is asked to obtain the most similar one from the whole dataset, measuring their similarity only based on their visual content. These systems are

typically used to facilitate the navigation and browsing into the database on the client side.

As mentioned before, the main ways to process the user's input are: content-based and text-based. The former type, Content-based image retrieval (CBIR), considers only the visual features during the search, these may include color, texture, shape, or even spatial layout. The latter type, Text-based image retrieval (TBIR), uses instead a set of textual items associated with the image, these may include captions, keywords, or descriptions relevant to the desired output. In both types, the first stage consists of converting the input features into numerical vectors used then to compare and retrieve images that are similar in their content.
Figure X shows a general overview of a CBIR pipeline, where it can be seen the 4 main stages are listed below:

1. Feature extraction: obtaining a numerical representation of the features that best characterize the images. Typically the features of the input query are extracted at the running time while the features for the entire dataset are extracted offline and used during the comparison.

2. Similarity measurement: By using a distance measure such as the Euclidean distance or cosine similarity it is obtained a numerical indicator of the likeliness between the input query and each of the images stored in the dataset.

3. Ranking and re-ranking: This similarity measure is used to select those images that best fit with the query. At this stage, the output scores are ordered and the top results are selected. In some cases, the previous steps are repeated with the top results as an ulterior analysis to obtain a higher precision.

In the VG setting, the initial dataset is intended to contain verified geotagged images, i.e. pictures to whom its geographical coordinates are already known. Then, by following a CBIR approach, the system extracts the relevant features from its visual content. Finally, once the ranking process is done and the best matches have been selected, it is retrieved the geographical information (or an aggregated version) of the top-N results.

These systems are well-known for their high-tolerance capabilities. The geographical coordinates retrieved are highly exact, with a tolerance of only

a few meters, when they accurately match the input query with its counterpart on the backend side. However, in order to get a greater rate of successful matches, these systems require large datasets, which increases inference time because the number of comparisons in the backend for each query grows, in most cases, directly proportional to the dimension of the dataset.

2.2.2 Classification

The image classification problem in computer vision entails training a machine learning model to associate a user's query image with a certain category or label retrieved from a set of predefined classes. The ultimate goal is for machines to be capable of identifying and comprehending visuals in the same way that humans do.

During the process, the machine learning model is trained on a dataset of previously annotated images. It then learns to find patterns in photos that distinguish one category from another by extracting vectors of numerical information from the images. These characteristics are derived from the image's visual content, which may contain numerical representations of colors, textures, forms, and other visual clues. After the teaching process is completed, the model is used to classify unseen images into their appropriate class. This procedure is required for a variety of applications, including object recognition, facial recognition, and medical diagnosis.

The classes in the VG setup are intended to be a group of predetermined geographical regions, generated by partitioning the geographical area of interest and defining the geo-tag of each class using the geographical coordinates of their associated photos. Once the geographical area of interest has been divided properly, the model receives image pixels as input, and the goal output is a one-hot encoded vector indicating the cell holding the image geotag. Given a test image, the output of this model is a probability distribution for the entire area. This formulation is superior to one that just regresses from pixels to latitude/longitude coordinates because it allows the model to indicate its uncertainty about a picture by assigning a confidence score to each cell that the image was taken there. In contrast, a regression model would be compelled to select a single site and would lack a natural manner of expressing concern about its predictions.

When compared to mage retrieval methods, this methodology provides an entirely different perspective. Their key advantage is their reduced latency during inference and their excellent scalability. Once trained, the inference time is nearly constant and independent of the dimension of the training dataset. To achieve more accurate predictions, one possible option would be to expand the number of positives during training, increasing the amount of photos associated to each cell and making the model more robust in detecting the right label of an input image. However, as it was discussed before, once the cell has been identified the coordinates retrieved by the model are the ones of the entire cell, extracted by aggregating the coodinates of the training images associated to it. As a result, the final prediction accuracy is determined by the granularity of the designated cells. Increasing the number of classes by constructing them at a finer granularity is connected with a decrease in accuracy as well, because by decreasing the area of the cell, fewer possibilities are examined during training.

2.2.3 Hybrid and Paired Methods

More recent studies have shown extraordinary results when combining the most diffused approaches so far, classification and retrieval. There are a variety of ways these two approaches can be combined to achieve better performances in the VG task. In this project, it were identified two main types, hybrid, and paired methods. For a hybrid method, it is intended a system that changes completely its approach in the transition from training to inference, during training it uses the dataset to tune a set of parameters but during inference, these parameters are extracted from the trained model and used as vectors of features. On the other hand, paired methods place the two procedures one on top of another, using the output obtained after the first layer to reduce the searching space on the second layer.

In the previous sections, we have seen how the two main approaches offer different advantages and limitations when applied to the VG task. However, in recent years, researchers have found ways of exploiting the best of both worlds, building highly scalable systems with low tolerance predictions. The hybrid procedure introduced on [6] uses the classification approach only as a proxy to train the model avoiding the introduction of mining techniques as in contrastive learning. After the training phase, the model is used to

extract a set of image descriptors which are then used during the inference phase as intended on the image retrieval approaches.

2.3 Image Retrieval Framework

The preceding section provided an overview of how the image retrieval framework approaches the VG challenge. The primary benefits and drawbacks of the VG systems that use this method were highlighted. Finally, the section addressed how newer methods link the image retrieval pipeline with the classification methodology. As a result, the retrieval strategy is presented in greater detail in this part, a system that first showed promising results in the early 2000s and has gained increasing relevance in the field of deep learning with the launch of AlexNet in 2012.

2.3.1 Feature Extraction

Until now, it has been stated that retrieval systems determine the similarity of two images by comparing their image descriptors, which are numerical representations of an image's content in the form of vectors. Nevertheless, no mention has been made of how this procedure works.

An image descriptor is an algorithm that collects important aspects from an image and encodes them in a form that allows them to be easily compared to other images. Based on the type of features the descriptor is extracting from the image the following classes can be made:

- Color-based image descriptors: These image descriptors use statistical methodologies to describe a picture's color distribution, which is the only information considered to represent its visual content. Color moments (such as mean, standard deviation, and skewness) are calculated from a picture's color histogram, which is a graph that shows the frequency of each color value in the image. The color histogram is typically displayed in three dimensions, representing the Red, Green, and Blue color channels, respectively.

- Texture-based descriptors: They primarily encode the image's local texture information by comparing the gray-scale intensity levels of a pixel and its surrounding neighbors. These descriptors are robust to noise

and illumination changes, and insensitive to geometric transformations. Specifically, a binary code is assigned based on whether each neighbor's intensity value is more than or less than the intensity value of the center pixel. The generated binary code is then utilized to represent the pixel's local texture pattern.

- Shape-based descriptors: They use the forms and contours of an image to describe its content. Its main virtue is that they are unaffected by translation, rotation, or scaling, allowing them to compare shapes that have been altered in different ways. Fourier descriptors, for example, aim to characterize the boundary of an item in a picture as a set of sine and cosine waves. The Zernike moments, which are based on the Zernike polynomials, are another well-known approach that can capture more complex and irregular structures.

- Deep learning-based descriptors: Unlike traditional feature extraction approaches that rely on craftsman features, deep learning-based descriptors learn how to extract them directly from data via neural networks. Learning a mapping between the input image and a high-dimensional feature space, with each feature representing a different component of the image. The input image is convolved using a collection of learnable filters that capture distinct features of the image at different scales in a Convolutional Neural Network (CNN). To minimize the dimensionality of the feature space, the generated feature maps are passed through a sequence of non-linear activation functions and pooling layers. Another type of deep learning-based descriptor is the Siamese Network, consisting of two identical neural networks that share the same weights, with each network taking as input one of the two images to be matched.

At this point, it has been shown a general overview on the different types of features introduced in the existing literature. The descriptors have been associated based on the type of information they are able to extract from the input pixels. However, often the literature proposes an alternative way of classifying them, based on the granularity or the level of details they are able to express from a single picture. While some descriptors tell a general overview of the whole image, others focus only on few portions of the image and treat them in a separate manner. Specifically, they group them into local and global descriptors.

- Local descriptors: They consist of techniques used in image processing

to identify highly interesting regions or points within an image. These points are locations that stand out from their surroundings in terms of intensity, texture, or other visual characteristics. The idea is to determine if two images refer to the same visual content by analyzing the number of matches in terms of their extracted keypoint. Typically, local features are more robust to changes in lighting, orientation, and scale, than global approaches. There are various types of local descriptors used in image processing, with some of the most diffused algorithms listed below:

- Scale-invariant feature transform: SIFT descriptors were introduced in 1999 by David G. Lowe [11]. The way it extracts features is based on the scale and orientation of an image by using gradient magnitudes and orientations within rotated image patches. They use a similar approach to the Difference of Gaussian (DoG) algorithm, which identifies key points based on the differences in the Gaussian function associated with the compared images. The input image is convolved using two Gaussian filters with varying standard deviations. The difference between these two filtered images yields a new image that emphasizes locations where the intensity changes at a specific scale. Then, the gradient orientation of patches surrounding each of the keypoints is extracted. These gradient orientations are then histogramed, and the orientation with the highest value is chosen as the dominant orientation.
- Speeded Up Robust Features: A few years later, in 2006, Herbert Bay, Tinne Tuytelaars, and Luc Van Gool introduced SURF descriptors [5] based on the same principles of SIFT , by making some modifications to the previous approach to make it more robust and computationally efficient. The algorithm uses a box filter to approximate the Laplacian of the Gaussian function and identify key points based on the local maxima of the determinant of the Hessian matrix. The use of the Hessian matrix for scale space representation was crucial in achieving faster feature detections. SURF also uses a unique descriptor that captures information about the distribution of Haar wavelet responses around a feature point. The Haar wavelet function is used to analyze and decompose signals into their frequency components, in this case, it is used to detect edges in an image.

  - Oriented FAST and Rotated BRIEF: In 2011 Ethan Rublee and Vincent Rabaud introduced the ORB descriptor [19]. By taking an improved version of the BRIEF (Binary Robust Independent Elementary Features [13]) descriptor and joining it together with a Features from Accelerated Segment Test (FAST) module. While in BRIEF, binary feature descriptors are computed by comparing the intensity values of pairs of pixels in a neighborhood around a keypoint being affected by images with significant orientation changes, Rotated BRIEF overcomes this problem by allowing the descriptors to be computed with respect to an arbitrary orientation of the patch around the keypoint, rather than being limited to horizontal and vertical directions. On the other hand, FAST module compares the intensity of pixels in a circular pattern around a central pixel, making this approach more efficient than previously existing methods, as it only requires a few comparisons per pixel.

- Global descriptors: These extraction techniques used in image processing describe an entire image rather than just a local region or point. Their goal is to capture the overall properties of the image such as color, texture, shape, and spatial layout. Global approaches are often made by grouping local descriptors extracted from an image into a set of clusters, each of which represents a distinct visual feature. This is often done using clustering techniques such as k-means or hierarchical clustering. Once the local descriptors have been clustered into visual words or codewords, a global descriptor can be constructed by counting the number of local descriptors that belong to each cluster in the image. Some of the most diffused methods that follow this approach are:

  - Bag of Visual Words: The technique was first introduced in the early 2000s by Sivic and Zisserman [21]. The BoVW algorithm surged with the observation that images can be represented as sets of small patches each of which capture specific visual features such as edges, corners, or textures. Once the visual words have been determined, the frequency of each visual word is counted in the image to create a histogram of visual word frequencies. This histogram serves as the fixed-length representation of the image.
  - Gist: The GIST method was applied for representing global descriptors in 2006 by Aude Oliva and Antonio Torralba [18]. The algorithm starts by dividing the image into a grid of cells and computing

a set of statistics for each cell. These statistics include color histograms, gradient orientation histograms, and texture energy measures. Next, a spatial pyramid matching scheme is applied to the image, by dividing the image into increasingly fine subregions and computing histograms of the GIST features within each subregion. The histograms are then concatenated into a single feature vector that captures the global spatial layout and color properties of the image.

- Pyramid Match Kernels: PMK was introduced in 2007 by Kristen Grauman and Trevor Darrell [8]. This algorithm represents an image as a set of histograms at multiple levels of spatial resolution. Each histogram captures the distribution of visual words or features within each region of the image. The idea is to divide the whole image into multiple spatial bins at each level of the pyramid, and to compute the histograms independently for each bin, so that the similarity between two images is given by comparing the histograms at each level of the pyramid using a kernel function, such as the Chi-squared distance or the intersection kernel.
- Vector of Locally Aggregated Descriptors: [12] The idea behind the VLAD methodology was based on the bag-of-words model. Once the histogram of visual word frequencies is generated, a residual vector is calculated for each visual word, computed as the difference between the feature vector and the nearest visual word. Finally, the residuals are aggregated into a single vector by summing up the residuals for each visual word.

2.3.2 Similarity Search

Previously it was explained that the idea behind image retrieval approaches in VG is to identify images that are visually similar to a given query based on their associated features. The previous section showed the general overview of how features are extracted, what kind of information they can represent, and at what level of granularity they can explain the visual content of the analyzed image. At this point, it is necessary to introduce the next fundamental part of the retrieval pipeline that is in charge of effectively comparing two vectors of features extracted by the same descriptor when applied to different pictures.

In order to evaluate the similarities or dissimilarities between two numerical vectors it is crucial to define an adequate indicator capable of measuring this property in terms of distance. Often, in literature, the following mathematical functions are implemented:

- Euclidean distance: It is one of the most commonly used distance measures for similarity search. It computes the distance between two Euclidean points. It is defined as the square root of the sum of the squared differences between the two vectors' corresponding elements. The Euclidean distance is always non-negative, and it is zero if and only if the two features are identical. The Euclidean distance between two n-dimensional vectors X and Y is calculated as follows:

$$d_E(X,Y) = \sqrt{\sum_{i=1}^{N} (x_i - y_i)^2}$$

- Cosine distance: The cosine distance is used to compare two vectors in a high-dimensional space. It computes the cosine of the angle formed by the two vectors. The cosine distance is always between 0 and 1, and it is 0 if and only if the two vectors are identical. The larger the cosine distance, the less similar the two vectors are. It's a popular distance metric in natural language processing and text mining.

$$d_C(X,Y) = \frac{X \cdot Y}{\|X\|_2 \|Y\|_2}$$

- Manhattan distance: Also known as the L1 norm, it computes the distance between two points by adding the absolute differences between their corresponding elements.

$$d_E(X,Y) = \sqrt{\sum_{i=1}^{N} |x_i - y_i|}$$

Intuitively, one may assume that in order to have more accurate results, retrieval systems should extract as many features as possible in order to increase the matching rate of the local features or to introduce more complex global approaches by concatenating all at once the outputs of different types of local descriptors. However, in many fields of machine learning

and mathematics in general, it has been demonstrated that moving toward high-dimensional vectors does not always improve the system's performance in finding similarities. Specifically, it has been noticed that distance measures lose their ability to represent similarities and dissimilarities in high-dimensional spaces. A phenomenon commonly described as "The curse of dimensionality".

When dealing with high-dimensional spaces the number of possible data points grows exponentially with the number of dimensions, making it difficult and computationally expensive to search for similar points. The distance between any two points tends to converge towards a constant value as the number of dimensions increases, making it difficult to distinguish between similar and dissimilar points. This means that in high-dimensional spaces, distance-based similarity measures become less informative, which can lead to poor search performance.

Various techniques have been developed to address the curse of dimensionality in similarity search. Dimensionality reduction is one of the most widely used approaches. The number of dimensions can be reduced by transforming high-dimensional data into a lower-dimensional space, making similarity search more efficient. PCA (Principal Component Analysis) and other dimensionality reduction techniques can be used to convert data into a lower-dimensional space while retaining the most important information.

In the PCA framework, the initial dimensions are transformed into a new coordinate system in which the data points are represented by a smaller number of variables called principal components. The first principal component is chosen to be the linear combination of variables that has the largest possible variance. In a similar way, the second principal component is then chosen to be, not only, the linear combination of variables that has the next largest possible variances, but also subject to the constraint that it must be orthogonal to the previous one. This is how the dimensionality of the feature space can be reduced while retaining the majority of the variance in the data by retaining only the top k principal components, where k is much smaller than the original number of variables.

2.3.3 Indexing & Clustering

It was discussed in the previous section how similarity searching techniques can quantify the degree to which two feature vectors are similar when they exhibit high coincidences in the features extracted. However, the retrieval pipeline discussed at this point consists of comparing the query features with all the features available on the backend side, which is inefficient and not suitable for large datasets. In this section, it is presented a clever solution that has been introduced in the literature, and whose implementation helps to solve this problem.

In machine learning, clustering is an unsupervised learning technique whose goal is to group objects in a way that maximizes similarity within each group and minimizes similarity between groups. Clustering and indexing are techniques used in similarity-searching tasks to improve the efficiency and scalability of the search process. The main idea is to avoid searching into the entire dataset for the best match of a query object but to search only within the clusters that are likely to contain the nearest neighbors. This reduces the number of distance calculations required and can significantly speed up the search process.

Indexing, on the other hand, involves creating a data structure that enables fast search based on some similarity metric or distance measure. An index is a precomputed data structure that summarizes the dataset and can be used to quickly find the nearest neighbors of a query object. However, clustering and indexing can be used together to further improve the efficiency and scalability of similarity search, by clustering the dataset into smaller groups and then creating an index for each cluster to speed up the search within each group.

In line with this topic, in recent years, the Faiss library developed by Facebook has gained relevance. The Facebook AI Similarity Search open-source library was designed to work with high-dimensional vectors, such as those used in computer vision and natural language processing applications. It applies the KNN algorithm to cluster the dataset into groups that are then indexed.

2.3.4 Reranking

On the last stage of the retrieval pipeline, it is common to add a post-processing step. The goal is to improve the accuracy of the retrieved output by applying deeper analysis into a reduced version of the searching space, considering only the elements that achieved better results in the previous stages. Reranking is the process of using a secondary ranking algorithm to reorder the top N results produced by a search engine or machine learning model based on additional criteria.

In the existing literature, some of the most diffused approaches apply spatial verification methods to verify the correctness of correspondences between points or features in two or more images, the goal is to improve the accuracy of matching algorithms by verifying that the matches satisfy certain geometric constraints. Some versions apply algorithms like RANSAC [17], which randomly samples a small set of point correspondences, fits a model to them, and then verifies the model by checking how many other correspondences agree with it.

2.4 Classification Framework

We have already seen how machine learning algorithms are used in the analysis of the visual content of an image in order to determine the geographical coordinates of where the picture was taken. The previous chapter provided a brief overview on the main structure of one of the most widely used methodologies in this subject, the image retrieval pipeline. This section of the project describes an alternative approach for tackling the challenging task of visual geolocation, changing completely the previous paradigm. Although there is not a clear winner when comparing these approaches, the classification framework is more demanded in the development of VG applications for large-scale and dense datasets, which are often applied in city-wide scenarios and for which the image retrieval approach loses efficiency, mainly in terms of inference time.

Image classification is the process of categorizing and assigning pre-defined labels to images based on the vectors of features extracted from their pixels. Approaching visual geolocation through classification surged from the interest of the research community after the great results observed in a variety

of fields. In medical imaging analysis, for example, image classification has been used in detecting tumors, cancer, and other abnormalities in X-rays. Autonomous driving cars are using image classification to identify and classify objects such as pedestrians, traffic lights, and road signs. And in the information-security industry image classification is used in security systems for facial recognition, object detection, and tracking.

The idea behind the image classification approach applied to the visual geolocation task is to develop machine learning models capable of associating images to geographical classes based on the visual content extracted from its pixels. The term geographical class coincides with the same definition of a geographical area, and in some cases, this term is extended into a wider idea, as a collection of geographical areas (not necessarily adjacent) that are linked together based on visual elements that are common to their geographical environments. Once the area of interest has been divided into the desired amount of classes, obtaining an adequate distribution of the training images, the labels are generated for each class by aggregating the latitude and longitude information of the images from the training dataset. To this extent, a classification-based VG system is able to associate the input query to a specific class and retrieve an accurate but approximated version of its latitude and longitude, obtained through the class label.

In the image classification framework, the reliability of a large and high-quality training dataset is key. The greater the dataset, the more likely the model's predictions are to be accurate. This is due to the fact that a larger dataset contains more representative examples of the various classes and variants within those classes, making it easier for the model to learn the patterns and characteristics that differentiate them. Larger datasets, on the other hand, help to avoid overfitting. When a model is trained on a tiny dataset, it gets overly specialized to the training data, resulting in poor performance on fresh data. A larger dataset can help to prevent overfitting by giving the model more examples to learn from, lowering the danger of it memorizing the training data rather than generalizing to new data. In order to improve the model's ability to generalize the different environments it is necessary to provide not only a large number of images for each class but also images taken from a wide variety of conditions, such as light intensity, seasonal factors, visual noise and camera orientation.

The research community, proposing classification-based VPR solutions, has found a tremendous chance in terms of having access to accurate and large-scale datasets. In recent years, the research community has benefited from access to a large variety of datasets designed to train, test and compare their proposals without incurring any significant additional costs. The role of the growing and already large media community has been crucial. Thanks to a variety of sources, mostly consisting of social media platforms, millions of people from all around the world are allowed to upload and share with the public a large number of high-quality pictures, all taken from a wide spectrum of environmental conditions that have been accurately labeled with their geographical information. The largely diffused YFCC (Yahoo Flickr Creative Commons) dataset serves as a clear example. A substantial multimedia dataset that includes pictures and videos as well as the tags and geolocation data that go with them was produced by Yahoo and Flickr and is accessible for research under a Creative Commons Attribution license.

Another important aspect to consider when developing classification-based VG systems relies on the proper delimitation of the classes, choosing their right dimensions according to the use case has a similar impact on the accuracy of the model as the one already discussed regarding the length of the training dataset. Finer approaches may require denser training datasets in order to provide a sufficient amount of images per class during the teaching phase, otherwise, models may perform poorly in generalizing the common features of the class. On the other hand, coarser solutions may not be sufficiently accurate during the inference phase, achieving good results in associating images to classes but performing poorly when comparing the ground truth with the retrieved geographical label.

In recent years, researchers have been studying different ways of handling and generating efficient geographical classes either for the planet or city-wide applications. In general, the latest proposals have been done by choosing different approaches under two main aspects. The proper delimitation of the geographical classes through the proper selection of a **partitioning schema**, which aids in reducing the complexity of the implemented system, and the **output aggregation mechanism**, used to efficiently improve the probabilistic distribution of the predicted classes during inference, defining an effective approach of combining classes and their outputs to improve predictions.

2.4.1 Discrete Global Grid Systems

Before diving into the description of how each of the proposed classification-based VG methods operates, in this section it is going to be discussed the most diffused libraries and methodologies used to generate the partitioning of Earth or any geographical area of interest. To this purpose, it is necessary to introduce the term of *geographical cell*, whose definition does not always coincide with the previously mentioned geographical class. A geographical cell, is the finest output obtained from an algorithm in charge of splitting the area of interest only based on the geographical information, such as latitude and longitude. In some of the proposed classification-based approaches, the cells are grouped together based on the visual content of their associated images in order to generate clusters of cells which are then called classes.

In the existing literature, this approach can also be referred to as Geohashing, which by definition, it converts any location's latitude and longitude into a short string of letters and digits. A geohash is a hierarchical spatial index, which means that it divides the Earth's surface into a grid of cells, each with a unique representative code. Each geohash code corresponds to a regular-shaped area on the map, such as rectangles or hexagons. The longer the code, the more precise the area it represents. The main idea behind the discrete global grid systems is to generate a discrete "digital" model of the Earth.

Although there is an infinite number of ways to split the surface of the Earth, the most diffused approaches follow a similar idea to the one proposed many centuries ago, through projections. During the 16th century mathematicians studied and developed algorithms to transform the globe into accurate flat maps. Popular rectangular maps, that are still used today like the Mercator's map (developed in 1569 by Gerardus Mercator) use a cylindrical projection, an algorithm that puts a theoretical cylinder over the globe and projects each of the points of the Earth onto the cylinder's surface. However, thanks to the Theorema Egregium proposed by Karl Gauss in 1827, it is demonstrated that it is impossible to project a spheroid into a flat surface without introducing any kind of distortion. Indeed, although the Mercator's map is accurate in terms of directions (the angle between two lines on the map is the same angle between the projected lines on the earth), it is highly inaccurate in terms of dimensions (areas closer to the

pole are larger than the rest of the map).

In order to reduce the distortions while achieving the goal of a discrete global grid, which is to cover the globe with each cell having an equal area. Modern approaches use 3D polyhedrons to project into the globe. Although any polyhedral solid can be mapped to the surface of the planet, only the Platonic solids (like cube, tetrahedron, icosahedron, octahedron or dodecahedron) can divide a sphere's surface into uniform, equal-sized cells. Hence, the Platonic solids are employed in the design of the most accurate DGGS in terms of greatest equal area, frequently by mapping the polyhedral faces to the surface model of the Earth. The coordinate reference system produced by this technique maps the faces of a base unit polyhedron to the surface model of the Earth.

In the existing literature, there exists different libraries and methodologies to perform the partitioning of the globe. The followings are some of the most diffused approaches in the classification-based VG field:

UTM

The UTM (Universal Transverse Mercator) projection is a cartographic projection used to represent the Earth's surface on a flat map. It divides the Earth into 60 time zones of 6 degrees each, numbered from 1 to 60, and centered on the meridians of longitude that are multiples of 6 degrees. Each UTM zone is projected onto a cylinder tangent to the Earth's surface along the central meridian of the zone. The cylinder is then unrolled into a flat map, resulting in a rectangular grid of coordinates. The grid lines are placed on the map at every km so that UTM coordinate values are displayed on the grid as labels, where the East-West position is determined by the vertical grid lines, whereas the North-South position is determined by the horizontal ones.

Since UTM follows a similar approach to the one described previously for the Mercator's map, one of the main distortions introduced by the UTM projection is that distances and areas are not accurately represented on the map. In particular, areas that are far away from the equator are disproportionately enlarged, while areas close to the equator are compressed. This is because the UTM projection is based on a cylindrical model of the Earth,

which stretches the Earth's surface as it moves away from the equator.

Another type of distortion introduced by the UTM projection is the distortion of shape. As you move away from the central meridian of a UTM zone, the shapes of features on the map become increasingly distorted. UTM also introduces angular distortion, which means that angles between lines on the map are not accurately represented.

S2 Geometry

S2 geometry is a mathematical framework originally designed to perform computations on the surface of a sphere. It was developed by Google as a key component of their mapping and geospatial tools, such as Google Maps, Google Earth, and Google Street View. Unlike UTM projection, S2 geometry does not rely on projecting the Earth's surface onto a flat plane, but instead represents the Earth as a sphere and makes the projection onto the six faces of a cube, allowing for more accurate representations of geographic data.

The S2 geometry is based on a hierarchical spatial indexing system that divides the surface of a sphere into a series of progressively smaller cells, using a quadtree structure. It starts by projecting the spherical representation of the earth into the six faces of a cube, then for each of the projected faces it applies a non-linear transformation in order to correct the problem of same-area cells on the cube having different sizes on the sphere. In total there are 30 levels in the hierarchy, which by using a 64-bit integer the finest granularity of the cells can reach every cm^2 on Earth.

In order to identify each of the cells in the s2 framework, the library implements at each level of the hierarchy an indexing system based on the Hilbert Curve. The Hilbert curve is constructed by recursively dividing a square into four smaller squares, and then connecting the centers of those squares in a specific order. At each level of recursion, the Hilbert curve visits each point in the square exactly once, and the resulting curve fills the entire square as the recursion depth goes to infinity. In doing so this algorithm assures that cells sharing similar indexes appear close to each other on the globe.

One of the key features of S2 geometry is its ability to accurately represent the curvature of the Earth's surface at any scale, from the entire globe down to a single city block. This makes it a powerful tool for geospatial computations, such as distance calculations, nearest neighbor searches, and spatial queries. Another advantage of the s2 geometry projection is that it provides an efficient way of storing and retrieving geographic data. Because the cells are all the same size and shape, they can be easily indexed and searched, making it possible to quickly retrieve data for a particular location or region. https://github.com/sidewalklabs/s2sphere

level	Min Area $[km^2]$	Max Area $[km^2]$
0	85,011,012	85,011,012
1	21,252,753	21,252,753
12	3.31	6.38
30	48×10^{-12}	93×10^{-12}

Table 2.1: Areas of the S2-sphere cells at different hierarchy levels.

H3

H3 is a hierarchical geospatial indexing system developed by Uber for mapping and location-based services. It is similar to S2 geometry since it uses a hierarchical system of cells to represent the surface of a sphere, but with some important differences. H3 cells are hexagons and pentagons, rather than squares. There are always exactly 12 pentagons at every resolution, designed to provide more uniform coverage of the surface of the Earth. This approach was introduced in order to minimize the impact of projection distortions, that in s2 geometry were already reduced by applying a non-linear transformation.

In H3 the hierarchy of the cells is also designed to be more flexible, with different levels of resolution depending on the specific use case. One of the key advantages of H3 is its ability to accurately represent complex urban environments, where traditional geospatial indexing systems can struggle

with irregular shapes and varying densities of data. H3 cells can be used to accurately represent neighborhoods, city blocks, and other areas of interest with a high degree of precision.

Although the shapes of the cells in the H3 framework helps reducing the distortion generated from the globe's curvature, this frameworks looses one important property that is insted preserved on S2 geometry, the area covered by a single cell is not the same as the sum of the areas covered by its children. In the classification-based VG framework, later on, it will be shown that this property is crucial for the implementation of some strategies. https://h3geo.org/

level	Min Area $[km^2]$	Max Area $[km^2]$
0	4,106,166	4,977,807
1	447,684	729,486
12	0.000185933	0.000370527
15	0.000000542	0.000001080

Table 2.2: Areas of the H3 cells at different hierarchy levels.

2.4.2 PlaNet

Developed in 2016, by Tobias Weyand, Ilya Kostrikov and James Philbin [24], the PlaNet framework was one of the first classification-based VG algorithms to be applied on a planet-wide scale. This means that the goal of this method was to identify the geographical coordinates of any type of photo taken at any possible location on Earth, including nature scenes whose visual cues are very similar world wide, such as beaches, waterfalls or mountains. The authors highlighted the fact that at that time many of the proposed VG methods were focused only on restricted fields of applications, for example some works treated only the landmark-building scenarios like [4],[23]. Others were focused on cities or other places were the available datasets could provide denser areas, but only a few proposals were done at the global scale. The authors also proposed a way of exploiting the temporal coherence between images in order to improve the accuracy by working with

photo albums instead of just single images.

In line with the description of classification-based VG algorithms, this method proposes an adaptive partitioning schema, meaning that the way it generates the geographical cells for the entire world uses the advantage of hierarchical DGGSs to regulate the density for each class. To this extent, the authors decided to work with the *s2 geometry* framework, partitioning the surface of the earth into a set of non-overlapping cells. Instead of fixing the depth in the hierarchy of the cells, arguing that this approach could impact the imbalance distribution of the classes, the authors introduce two parameters, τ_1 and τ_2, to indicate the minimum and maximum amount of images each class is allowed to contain in order to be considered a class. The partitioning schema starts from the root of the quadtree as implemented on the s2 geometry library, then it recursively descends by splitting the current leaves into smaller cells until the distribution of the images is in line with the constraints, so that larger cells cover sparsely populated areas.

The way this algorithm calculates the output prediction of an image whose features have been passed through the trained model is by simply applying the SoftMax function to the output layer, obtaining the probability distribution within the classes. This output probability distribution is considered by the authors an important advantage for the classification-based VG methods with respect to other approaches since it represents the model's confidence for possible locations, useful in cases where the model is not able to determine with low uncertainty the location of the image.

2.4.3 CPlaNet

Developed in 2018 [20], by Paul Hongsuck Seo, Tobias Weyand, Jack Sim, and Bohyung Han, the CPlaNet framework aimed to improve the performance of the existing classification-based VG algorithms by combining the output of multiple partitioning schemes, achieving highly accurate results while reducing the number of parameters with respect to fine-granularity models. In the paper, the authors highlighted the importance of choosing a proper partitioning schema to handle the crucial trade off related to the granularity of the classes. As it was already discussed in the previous section, usually finer approaches are prone to overfitting since the fixed number of training images are distributed into a larger number of classes. On the

other hand, coarser approaches may not provide a proper accuracy output since the retrieved label of the class (containing an aggregated version of the geographical information) is far from the ground truth of the query image.

The idea behind CPlaNet is to generate a large number of fine-grained classes by intersecting multiple coarse-grained partitions, also known as geoClassSets. Each partition is then used to train a different classifier during the training phase. However, during inference, a normalized version of the probability distribution of each classifier is added cell-wise across the different partitioning schemes, generating finer predictions without incurring training data deficiency issues commonly observed when reducing the number of images per class.

The way CPlaNet generates each geoClassSet starts by splitting the geographical area of interest into a set of non-overlapping cells. The authors proposed to use the s2 geometry library since it allows to access cells with very similar properties at every level of the quadtree. In this process, the depth of the quadtree is fixed to a specific value at which all cells are considered. Then, after associating each of the training images to its corresponding cell, the empty ones are randomly merged with one of their non-empty neighbors. At this point, the algorithm merges the existing cells into clusters or geo-Classes, one at a time, until the number of classes reaches a predefined number. During the merging process, cells are grouped by considering both geolocational and visual distances. At each iteration, the cell or group to be merged is selected by applying a scoring function ($\omega(v_i)$) that considers the number of images present in the group (α_1), the number of non-empty s2-cells (α_2) and the number of empty cells (α_3). Once the cell or group has been selected it is merged with its closest neighbor in terms of the edge scoring function $\nu(v_i, v_j)$. In this formula, the visual distance refers to the cosine similarity between the visual features of the classes, while the geolocational distance is the euclidean distance between the centers of the two nodes.

$$\omega(v_i) = \alpha_1 * n_{images} + \alpha_2 * n_{emptyS2} + (1 - \alpha_2) * n_{nonEmptyS2}$$

$$\nu(v_i, v_j) = \beta_1 * dist_{visual}(v_i, v_j) + \beta_2 * dist_{geographical}(v_i, v_j)$$

Once each of the partitioning schemes has reached the desired number of

classes, a classifier is trained independently for each geoClassSet. On the other hand, during inference once the probability distributions have been obtained for each of the geoClassSets, the authors proposed to normalize the obtained scores by the number of s2-cells available for each class before adding the cell-wise probability distributions across the geoClassSets. This strategy assures that each classifier contributes equally to the final prediction.

In particular, the authors quote three major benefits of combinatorial partitioning:

- Fine-grained classes with fewer parameters: The authors demonstrate that by using a combinatorial approach to generate finer classes from coarse partitions, it is possible to build VG classifiers that require fewer parameters.

- More training data per class: By considering fewer classes the number of training images associated to each class is sufficient to make the training process robust and avoid overfitting.

- More reasonable class sets: At some partitioning, the geographical cells are combined considering the information of their visual environments by comparing an aggregated version of the visual features of their associated images. For this reason the obtained classes are likely to share common visual characteristics that are well represented on each class.

2.4.4 Hierarchical Geo-Estimation

Also in 2018, a group of researchers from Hannover, Germany, developed a hierarchical model [15] for the Visual Place Recognition of photos. The group conformed by Eric Muller-Budack, Kader Pustu-Iren, and Ralph Ewerth proposed to exploit the hierarchical knowledge of multiple partitioning schemes to improve the accuracy of classification-based VG methods at inference time. Their method was demonstrated to perform well on the planet-wide scenario, even by considering a reduced number of training images per class, and considering instead a vast variety of visual environments into account, such as indoor pictures, and natural and urban settings.

As in previous cases, this work dedicates particular attention to describe the existing tradeoff between coarser and finer partitioning schemes. However, on this occasion, this phenomenon is tackled from a different perspective. The authors propose that the main reason why finer partitionings perform better on city-scale scenarios while coarser divisions are more efficient at the country scale is because of the huge diversity caused by different environmental settings. For this reason, the authors propose that taking information about the type of environmental setting into account could address efficiently the requirement of specific features to distinguish different locations.

On the other hand, the authors proposed to develop a multi-partitioning approach for incorporating the hierarchical knowledge at different geographical granularities. Similar to PlaNet, the authors proposed using the s2 geometry library to generate a set of non-overlapping cells in order to build multiple partitioning schemes by varying the parameters τ_1 and τ_2 that share the same meaning discussed before. With this approach at each partitioning, the authors argue that it prevented any kind of dataset bias by allowing the creation of classes with almost the same amount of training images. On the other hand, more accurate predictions can be obtained at interesting regions such as landmarks or cities.

In order to calculate the output prediction during inference, the authors propose multiplying the respective output probabilities for each of the partitioning schemas. Consequently, the finer subdivision is refined by the output probabilities of the coarser partitions.

2.4.5 MvMF

This method was proposed in 2019 by Mike Izbicki, Evangelos E. Papalexakis, and Vassilis J. Tsotras. In comparison to the previous methods, where the main contribution of the proposed solutions was aimed to solve the negative impact of the granularity tradeoff, in the paper [14], the authors argue that the used cross-entropy loss is inadequate for the classification-based VG framework because it does not take advantage of the earth's spherical geometry. Instead, they propose an alternative loss function based on the von-Mises Fisher distribution which demonstrates to improve the geolocation accuracy.

The authors build their proposal on top of the PlaNet framework, applied into a planet-wide scenario, highlighting the fact that while some images are strongly localizable because of their unique visual content, such as famous buildings, bridges or monuments, other images are hard to localize since different places worldwide can show very similar visual environments (pictures taken from a mountain, beach or at indoors). At the same time, they assure that by exploiting the spherical geometry of the Earth it is possible to represent the visual ambiguity of the images.

In traditional classification tasks, including the previously discussed solutions, it is common to used the cross-entropy loss to measure the dissimilarity between the predicted probability distribution and the actual probability distribution of the target variable. This loss is minimized when the predicted probability for the true class is close to 1, and for all other classes close to 0. The cross-entropy loss works by applying the softmax function to the output vector of the classifier and then computing the loss value from the true label, y, and the predicted probability , $\hat{y}_i$. However, on this occasion, the authors criticize the fact that this approach does not consider the Earth's spherical geometry.

$$L(y, \hat{y}) = -\sum y_i * log(\hat{y}_i)$$

The MvMF framework applies the *von-Mises Fisher* distribution (*vMF*) which is used in directional statistics to study spherical distributions. The easiest way to understand it is to consider it as the spherical analog of the Gaussian distribution. The normal (or Gaussian) distribution is a continuous probability distribution that is symmetric, bell-shaped, and characterized by two parameters: the mean μ and the variance σ^2, which control the location and spread of the distribution. On the other hand, the vMF distribution is used to model directional data on the surface of a sphere. It is characterized by two parameters: the mean direction μ and the concentration parameter κ, which control the location and concentration of the distribution, respectively.

In the *mixture of vMF* there are considered c component *vMF* distributions and the parameters are considered as collections of multiple items. In particular it is considered the set of mean directions $M = (\mu_1, ..., \mu_c)$, the set of concentration parameters $K = (\kappa_1, ..., \kappa_2)$ and a vector of weights $\Gamma \in \mathbb{R}^c$.

The following equation describes the density function for all $y \in \mathbb{S}^2$.

$$vMF(y; \mu, \kappa) = \frac{\kappa}{sinh\kappa} exp(\kappa_i \mu^T y) \tag{2.1}$$

so that the obtained loss function is the following:

$$l_{MvMF}(x, y; M, K, W) = -log \sum (\Gamma_i * vMF(y, M_i, K_i)) \tag{2.2}$$

Part I

Chapter 3

Datasets

In the previous section, it was presented the general overview of the VG framework and how the classification-based VG approaches tackles this challenging task, highlighting the main requirements in terms of available data in order to provide highly accurate results at fast inference time. In particular, it was explained the existing trade off regarding the granularity of the partitioning schema, focusing on how large and dense datasets are more likely to achieve better results, specially in the city-wide scenario, and how they avoid the overfitting of the model by offering more representative samples for the various geographical classes, making it easier for the model to learn the patterns and characteristics that differentiate them, while lowering the danger of it memorizing the training data rather than generalizing to new data.

On the other hand, it was discussed how the research community, proposing classification-based VPR solutions, has found a tremendous chance by having access to a large clusters of high quality geo-tagged images, used to build large-scale datasets designed to train, test and compare their proposals without incurring any significant additional cost. In particular, it was mentioned the positive impact of the growing and already large media community, where millions of people share with the public a large number of high-quality pictures, all taken from a wide spectrum of environmental conditions that have been accurately labeled with their geographical information.

In line with the previous discussion, this section aims to describe the list

of the datasets that were used at the different stages of this project, including the evaluation and fair comparison of the most diffused classification-based methods.

3.1 YFCC100M

The Yahoo Flickr Creative Commons 100 Million Dataset was published in 2016 by Bart Thomee, David A. Shamma, and Gerald Friedland [22], comprising a large collection of almost 100M images and videos made free and publicly available. This media dataset was considered one of the largest collection of media elements useful for a variety of research activities oriented in the image analysis and computer vision fields.

The dataset contains around 99.2 million photos and 0.8 million videos extracted from the Flickr social media platform, in the temporal range between 2004 and 2014. For each item in the collection, the dataset includes a set of tags (introduced for a variety image classification purposes), timestamp, locations and other useful information, such as the Flickr identifier, the camera description, the title, and general information about the user.

In line with the purpose of this project, we were interested only on the images associated with their geographical metadata. To this goal, the dataset provides around 48 million pictures annotated with geographical information, either obtained automatically from the GPS of the capturing device or manually introduced by the user during the uploading of the files. In total, the geographical items make reference to 249 territories world wide, including a vast variety of visual scenarios such as landmarks, indoor and outdoor pictures, artistic images, ecc.

3.1.1 MP16

The MediaEval Placing Task 2016 dataset is a subset extracted from the original YFCC100M. The subset includes around 5 million images with geographical information extracted from Flicker on a world-wide basis.

In this project we considered in our experiments the same subset as the one introduced in https://github.com/TIBHannover/GeoEstimation.

Figure 3.1: Dataset images randomly sampled from YFCC100M

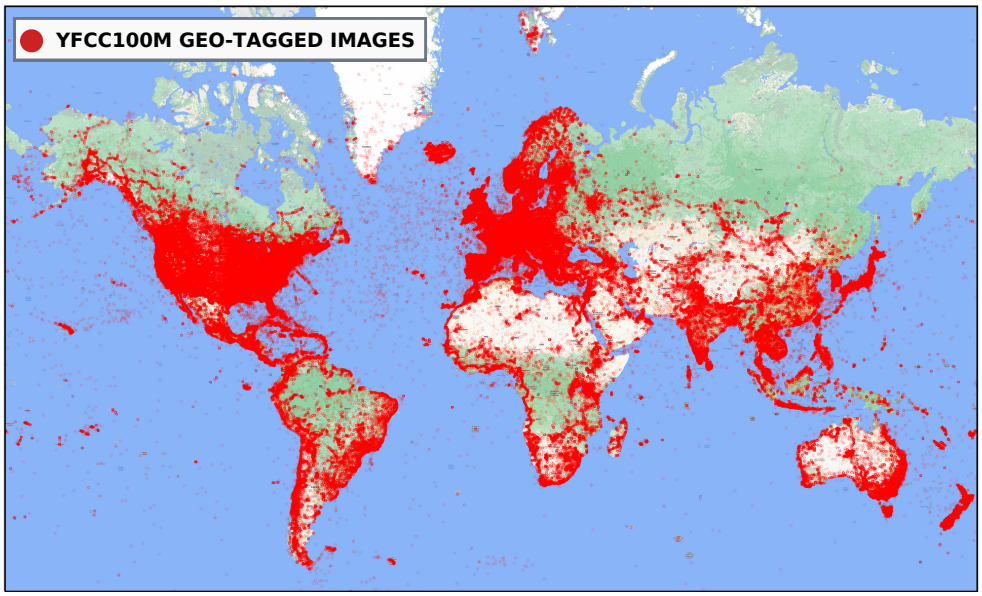


Figure 3.2: Geographical distribution of the geo-tagged images of YFCC100M Dataset.

However, the MediaEval Placing Task was an anual research challenge that aimed to develop algorithms for automatically geolocating images and videos based on their visual content and other metadata. In order to determine the geographic location where a particular media item was captured or produced, the task involved using a dataset of geotagged multimedia content, along with other metadata and contextual information, to develop a system

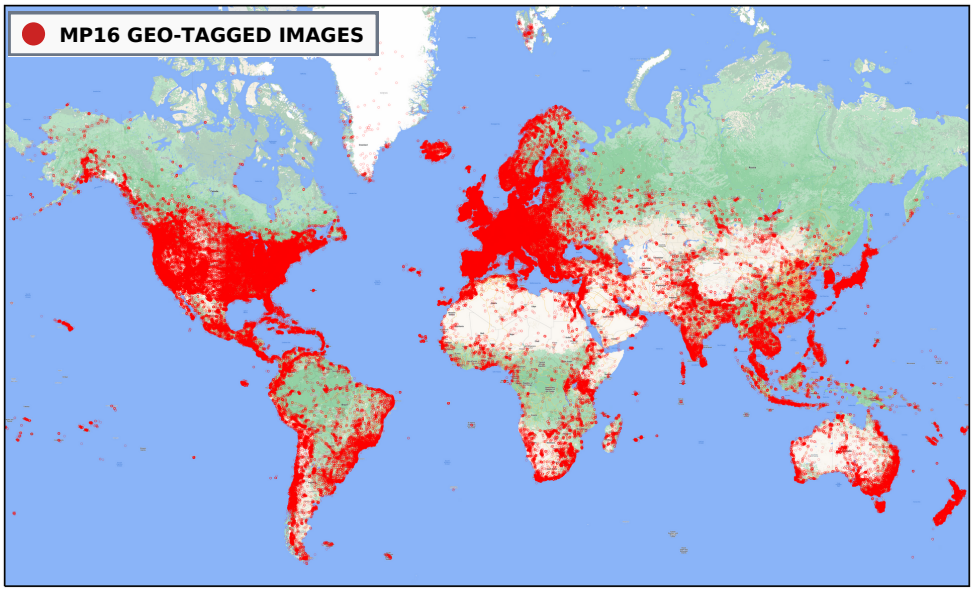


Figure 3.3: Geographical distribution of the geo-tagged images of MP-16 Dataset

that can accurately predict the location of a new media item.

3.1.2 YFCC_26k

Commonly used as validation set [15][10], this dataset was generated by randomly extracting 25,600 geo-tagged images from the original YFCC100M.

3.2 Im2GPS & Im2GPS3k

Introduced in 2008 by James Hays and Alexei A. Efros [9], this dataset was proposed with one of the first and simplest algorithms for VG applied to the planet scale. The idea behind the proposed algorithm was to provide a probability distribution for a query image belonging to a particular region worldwide, by extracting and aggregating features from the images in order to find its k-nearest-neighbors (k = 120) around the globe. For this reason,

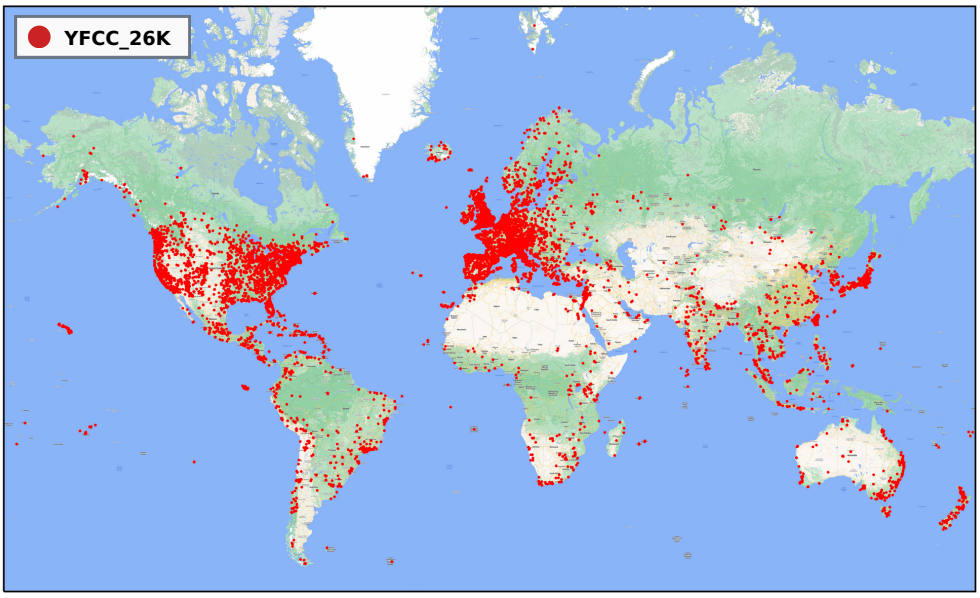


Figure 3.4: Geographical distribution of the geo-tagged images of YFCC_26K dataset.

the authors were interested in defining a sufficiently large dataset with labeled images providing their geographical information.

In the paper, the authors highlighted the importance of a rapidly growing media community and proposed to take advantage of widely diffused social media platforms such as *flickr.com* to collect high-quality labeled images. The training dataset consists of 6 million images extracted from Flickr, which were filtered based on their provided textual tags in order to rely only on useful and high-quality data. The authors described the existence of poor-quality images, mainly affected by noise and low resolution, that were efficiently filtered by considering both the geographical coordinates and additional geographic keywords.

On the other hand, the authors provided a test set, primarily to evaluate the performance of their method. In this case, 400 images were randomly sampled from the original dataset, to which they manually discarded any

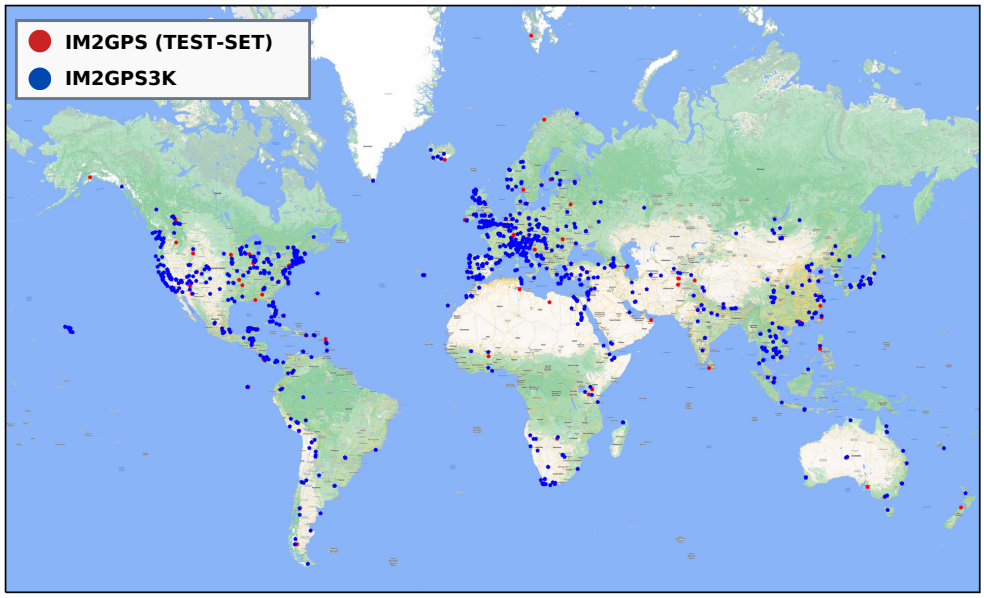


Figure 3.5: Geographical distribution of the geo-tagged images of im2GPS test-set and im2GPS3k Dataset

kind of undesirable photos, evaluating either their visual content (abstract art, black and white, ecc.) and the presence of significant artifacts (blur and extreme noise). Finally, by discarding images that may affect someone's own privacy and removing from the training data all those images whose authors are already present in the test set, it was provided a final dataset with 237 items.

The authors Nam Vo, Nathan Jacobs and James Hays [16] provided a new dataset consisting of 3,000 samples randomly extracted from im2GPS training-dataset, called im2GPS3k, arguing that the previous test-set was not sufficiently large for significant evaluations on a planet-scale in VG.

3.3 Pitts250k

This dataset was introduced in 2013, by Akihiko Torii, Josef Sivic, Tomas Pajdla, and Masatoshi Okutomi [2]. Originally designed for retrieval-based VG applications, it is a large-scale image dataset covering a portion of the geographical area of Pittsburgh, Pennsylvania, USA. The dataset coontains precisely 254,064 geotagged images that were collected using Google Street View, a feature of Google Maps that provides panoramic views of various locations around the world, covering a large amount of urban scenarios, including downtown, residential neighborhoods, and commercial areas.

Each image in the Pitts250k dataset is associated with a geographical label, which provides the geographical information, such as latitude and longitude, of where the picture was taken. Initially, for the development of this dataset, there were extracted 10,586 panoramas from Google Street View, each of them at very high resolution (6,656x3,328 pixels). Each panorama was then cropped into 24 perspective images with dimensions 640x480, considering 60° of the horizontal field of view (FoV), using two pitch directions (4° and 26.5°) and 12 yawn directions (at each 30°)

On the other hand, the test data originally consisted of 24,000 images generated using a similar approach to the training data, starting from 1,000 panoramas collected with different timestamps. These panoramas were randomly selected from the Google Pittsburgh Research Data Set.

3.3.1 Pitts30k

Introduced in 2016 by Relja Arandjelovic, in the same paper on which NetVlad [3] was proposed, Pitts30k contains a subset of 30,000 images from the larger Pitts250k dataset, with the images selected to form a representative sample of Pittsburgh's urban and suburban areas. The variant was introduced in order to facilitate faster training for some experiments.

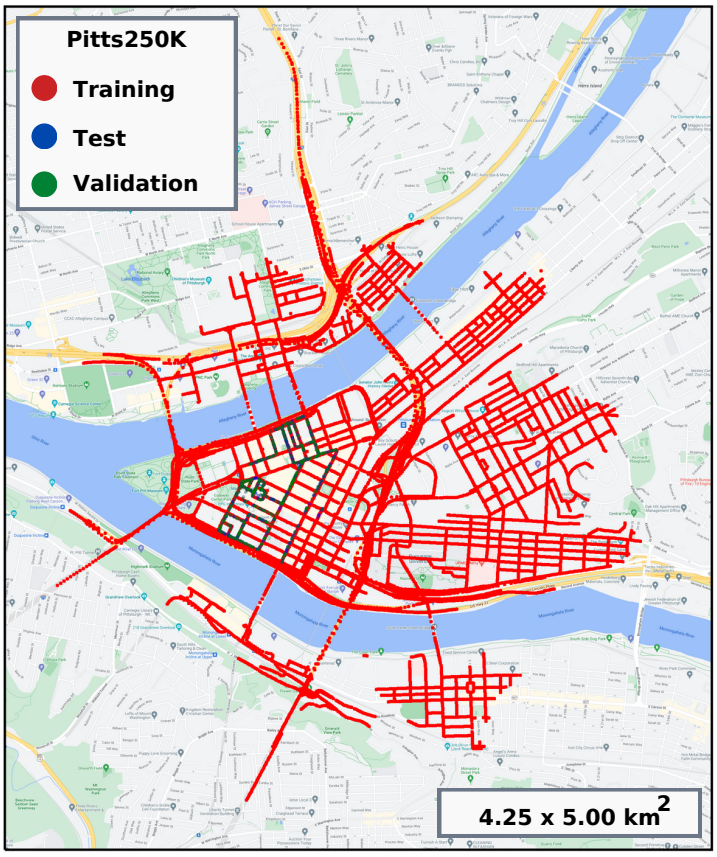


Figure 3.6: Geographical distribution of the geo-tagged images of Pitts30k dataset.

3.4 Tokyo 24/7

Introduced in 2015 by Akihiko Torii and Relja Arandjelovi [1], the Tokyo 24/7 dataset contains 75,985 street-level images with their corresponding geographical coordinates, covering a significant part of the urban area in Tokyo, Japan. The authors introduced the following dataset focusing on the existing challenge of large-scale VG applications, on which appearance variations in the query images, such as changes in the illumination or the presence of environmental noise due to seasonality represent one of the major

Figure 3.7: Sampled images from Pitts250k training dataset.

vulnerabilities of this task. For this reason, the authors propose an efficient approach to extend existing datasets, commonly sampled on a regular grid basis, by introducing synthetic views which is demonstrated to have a positive impact in the inference accuracy. In the mentioned dataset, each image was generated with a resolution of 640x480 pixels and representing different lighting conditions and from various angles and perspectives.

This images were generated by applying virtual transformations to 6,332 street-view panoramas. During the development of the dataset, there were generated 597,744 synthesized views generated at 49,812 virtual camera positions combined with depth maps downloaded from Google maps, each of the maps encoded as a set of 3D plane parameters. Originally the panoramas captured 360° by 180° horizontal and vertical viewing angle with a standard resolution of 13,312x6,656 pixels. However, the panoramas were then cropped into 12 perspective images 1,280x960 pixels each, by partitioning each of the items across 12 yawn directions (30°, 60°, ..., 360°) at a fixed pitch direction (12°).

Regarding the evaluation dataset the authors proposed a subset of 315 images covering a smaller area of 1600x1600 m^2.

3.5 San Francisco eXtra Large

Firstly introduced in 2022 by Gabriele Berton, Carlo Masone, and Barbara Caputo, [6] the SF-XL dataset was proposed specifically for evaluating the performance of VG applications in urban environments. The authors proposed that the existing geo-tagged datasets were not suitable for the

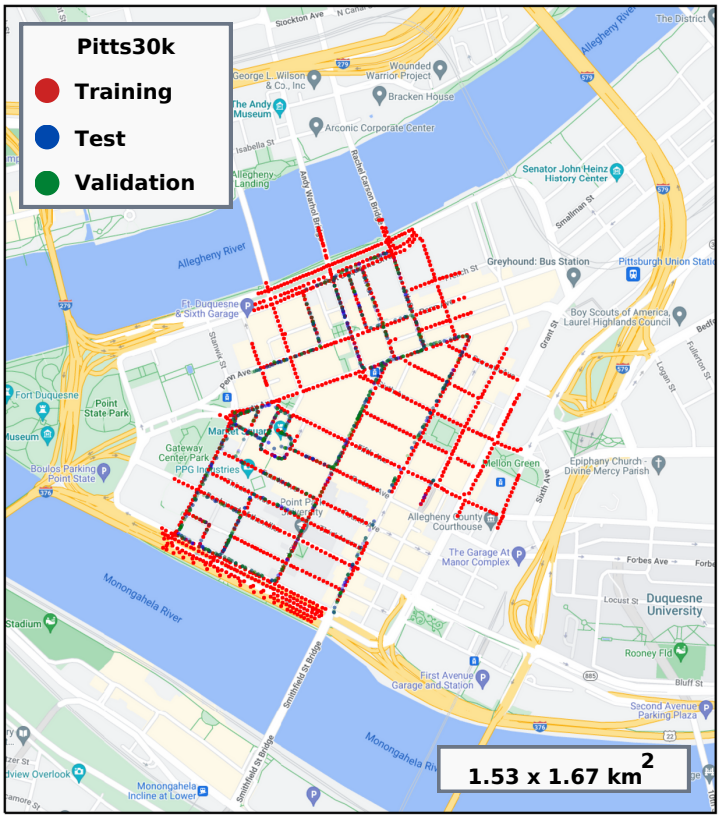


Figure 3.8: Geographical distribution of the geo-tagged images of Pitts30k dataset.

city-wide scenarios, on which VG methods require higher granularity levels to provide accurate results (with just a few meters of tolerance). On the other hand, the majority of the large geo-tagged datasets were planet-wide oriented, covering more extensive geographical areas with few visual characteristics in common within items, instead of covering entire urban environments. Thus, this dataset is considered to be the first highly-dense dataset city-wide oriented, comprising a collection of images at different timestamps, taken in the range from 2009 to 2021, to offer the most realistic conditions requested on a possible use case for VG in large urban environments.

Figure 3.9: Sampled images from Tokyo 24/7 training dataset

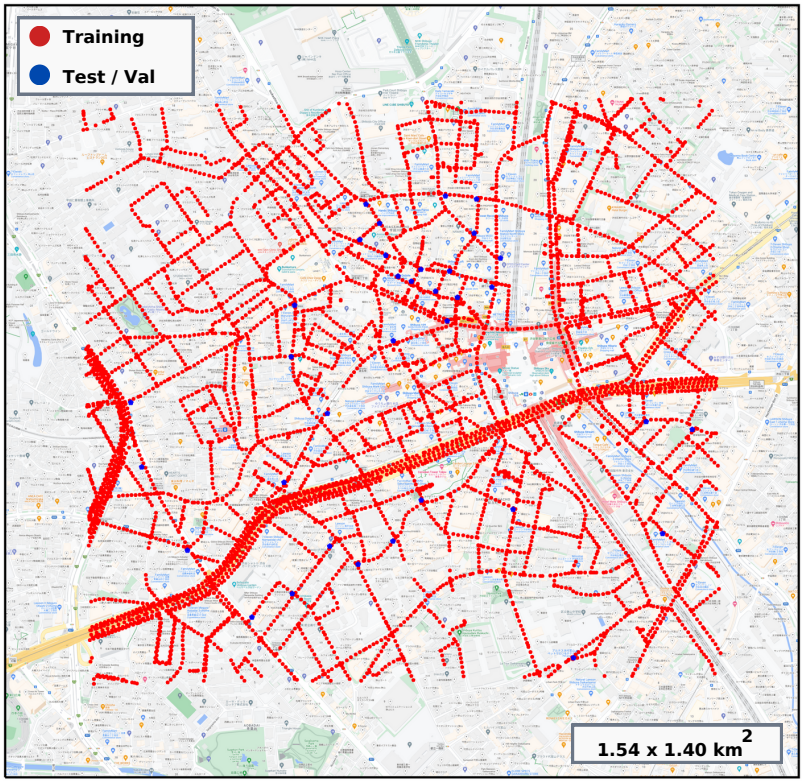


Figure 3.10: Geographical distribution of the geo-tagged images of Tokyo 24/7 dataset.

The authors proposed three different sets of images specifically designed for training, and testing/evaluating VG methods at a city-wide scale. To this scope, the different sets come from different sources, offering images from different visual domains, aiming to simulate real-world scenarios while covering the whole area of the city of San Francisco, USA,. The training-dataset contains images extracted from Google StreetView imagery, a feature of Google Maps that provides panoramic 360° views from various positions along streets and roads around the world. The additional sets, designed for testing the trained model on unseen samples were extracted from Flickr, an online photo management and sharing application, and the San Francisco Landmark Dataset [7], a collection of data and images related to historical landmarks and buildings in the city of San Francisco.

Regarding the training-set, SF-XL offers 3.43M panoramas (images covering 360° scene view) from which there were extracted 41.2M frames, by splitting the panoramas horizontally into 12 crops each, labeling them with geographical information including latitude, longitude, and camera orientation. On the other hand, the two additional subsets proposed as test queries (v1 and v2) consists of 1,000 images collected from Flick and 598 images from the San Francisco Landmark Dataset , respectively.

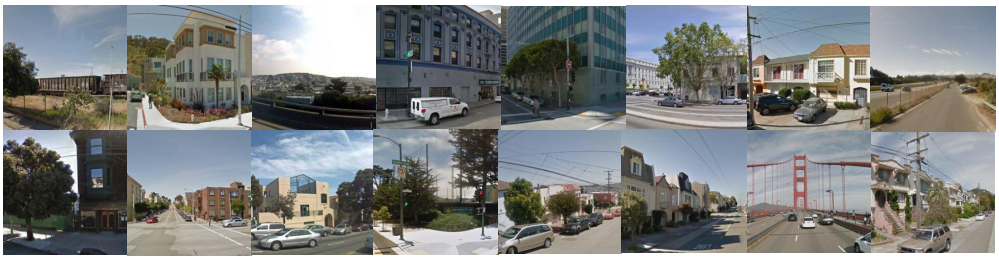


Figure 3.11: Training-set samples randomly extracted from San Francisco eXtra Large.

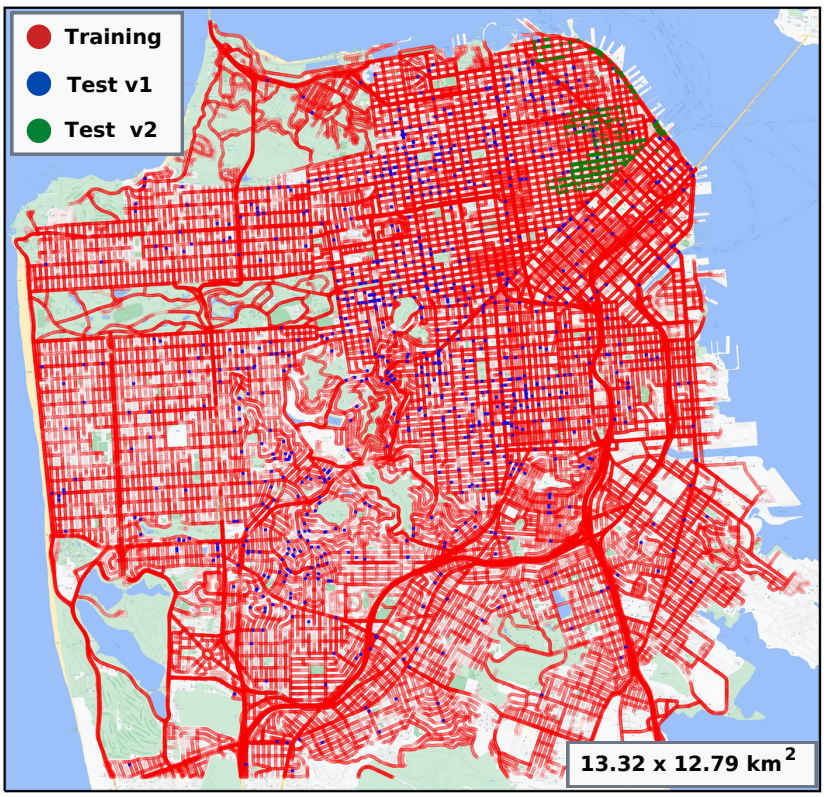


Figure 3.12: Geographical distribution of the San Franciso eXtra Large dataset

Part II

Chapter 4

Replicating the results

We discussed the most common approaches to the classification-based VG challenge in previous sections. The operation of the suggested methods was explained, as well as how they primarily contribute to the development of faster and more accurate VG applications, focusing on their primary differences regarding the generation of the partitioning schema and how some of them even aggregate the model's output to improve the prediction's accuracy. The datasets that were used in this research were discussed in the previous chapter. These datasets were used to assess the performances of the suggested methods in order to provide a fair comparison in the city-wide scenario, which is closer to a real and practical case of application.

However, many of the demonstrated classification-based pipelines did not include source code that could be used to assess their performance in challenging scenarios. As a result, one of the primary goals of this project was to comprehend the concept underlying each proposal and to develop our own implementation based on the theoretical description given by the available documentation. Along this process, thanks to the great work done by Eric Müller-Budack, Kader Pustu-Iren, and Ralph Ewerth, who did provide a well documented repository to replicate their results, it was possible to modify their proposed algorithm [15] to implement the rest of the desired methods.

In this chapter it is presented the obtained results from testing the pretrained model proposed by the authors and comparing it with a model trained from scratch, using the same datasets as those provided in their

documentation. The experiments were computed on a 24GB NVIDIA TITAN RTX.

Experimental setup:

- Training-set: MP-16
- Validation-set: yfcc_25600
- Test-set: im2gps & im2gps3k
- Backbone: ResNet50
- Batch size: 128
- τ_{mins}: [50, 50, 50]
- τ_{maxs}: [5000, 2000, 1000]

Test-set	Partitioning	Great Circle Distance (GCD)				
		1m	25m	200m	750m	2500m
im2GPS	fine	0.0928	0.3122	0.4937	0.6667	0.7890
	medium	0.1392	0.346	0.481	0.6793	0.7890
	coarse	**0.1561**	**0.3882**	0.4895	0.6624	0.7848
	hierarchy	0.1477	0.3797	**0.4979**	**0.6878**	**0.7932**
im2GPS3k	fine	0.0617	0.2429	0.3604	**0.5155**	**0.6680**
	medium	0.0824	0.2616	0.3577	0.5125	0.6637
	coarse	0.0991	0.2723	0.3610	0.5105	0.6627
	hierarchy	**0.1011**	**0.2796**	**0.3680**	0.5105	0.6710

Table 4.1: **Test on a pretrained version for HGE** proposed by [15]. Evaluating its performance on the proposed test-sets

Test-set	Partitioning	Great Circle Distance (GCD)				
		1m	25m	200m	750m	2500m
im2GPS	fine	0.0844	0.3038	0.4768	**0.6751**	**0.7890**
	medium	0.1308	0.3418	0.4726	0.6329	0.7764
	coarse	0.1392	0.3629	0.4684	0.6371	0.7679
	hierarchy	**0.1435**	**0.3797**	**0.4937**	0.6498	0.7848
im2GPS3k	fine	0.0617	0.2252	0.34	**0.4948**	**0.6527**
	medium	0.0767	0.2379	0.3327	0.4818	0.642
	coarse	0.0868	0.2506	0.3297	0.4751	0.634
	hierarchy	**0.0928**	**0.2589**	**0.3427**	0.4892	0.6446

Table 4.2: **Test on a trained-from-scratch version for HGE**. Evaluating its performance on the proposed test-sets [15]

Chapter 5

Experiments on SF-XL

Based on the work done by Eric Müller-Budack, Kader Pustu-Iren, and Ralph Ewerth, it was possible to adjust the HGE framework to replicate the pipelines for the rest of the widely diffused classification-based VG methods, whose implementation were not publicly available. Specifically, this project presents the comparison between HGE and two additional proposed methods, PlaNet and CPlaNet. All of the proposed solutions exploit the s2sphere partitioning framework to generate the geographical cells which are then combined in different ways to generate the classes according to the authors proposals.

This section compares the effectiveness of these three classification-based VG methods in the context of a large and highly dense city-wide dataset, SF-XL.

5.1 PlaNet

The main goal of these experiments was to evaluate the PlaNet framework on SF-XL. For this reason there were tested different partitioning schemes by varying the granularity level of the obtained classes, and evaluating the model's robustness against the granularity trade-off discussed in the previous sections.

Experiment setup:

- Training-set: SF-XL training set

- Validation-set: SF-XL queries v2
- Test-set: SF-XL queries v1
- Backbone: EfficientNet_B0
- batch size: 64
- optimizer: adam
- learning rate: 0.0001
- τ_{min}: 100

τ_{max}	num. classes	Great Circle Distance (GCD)				
		25m	50m	100m	500m	1000m
80,000	1,169	1.0	2.7	10.4	28.5	38.2
40,000	2,336	2.45	6.7	16.2	31.3	40.8
20,000	4,714	5.43	15.5	24.3	34.6	42.9
10,000	9,359	11.2	22.7	28.5	36.6	44.0
5,000	18,505	20.0	28.5	**31.6**	**38.1**	**45.8**
2,500	34,978	22.4	27.9	30.1	37.2	45.7
1,250	65,330	**24.5**	**28.6**	30.2	37.0	45.2
625	115,226	20.7	23.7	25.4	32.4	40.2

Table 5.1: Evaluating the performance of PlaNet on SF-XL by varying the granularity of the partitioning schema

From the results showed on Tab 5.1 and by analyzing the GCD@25m it is evident that the granularity trade-off is properly balanced around $\tau_{max} = 1{,}250$.

5.2 HGE

The experiments discussed on this section were designed to evaluate the performance of the HGE framework on SF-XL. By considering the same parameter-space as in the previous section it was decided to evaluate how

the previous results could be improved when aggregating multiple partitioning schemes at different granularity (τ_{max}).

Experiment setup:

- Training-set: SF-XL training set
- Validation-set: SF-XL queries v2
- Test-set: SF-XL queries v1
- Backbone: EfficientNet_B0
- batch size: 64
- optimizer: adam
- learning rate: 0.0001
- τ_{min}: 100

τ_{max}			Great Circle Distance (GCD)				
coarse	medium	fine	25m	50m	100m	500m	1000m
80,000	40,000	20,000	4.8	13.6	22.0	31.8	40.4
40,000	20,000	10,000	11.1	22.0	27.3	36.3	44.3
20,000	10,000	5,000	20.1	28.7	32.0	39.4	47.4
10,000	5,000	2,500	26.1	29.5	33.3	39.4	47.8
5,000	2,500	1,250	**27.0**	**32.2**	**33.9**	**40.1**	**48.0**
2,500	1,250	625	23.2	26.6	28.1	35.6	44.7

Table 5.2: Evaluating the performance of HGE on SF-XL by varying the granularity of the partitioning schema

By analyzing the highlighted results with the ones obtained in the previous section it is possible to see that the best accuracy on the PlaNet framework (at τ_{max} was improved by considering the model's predictions on coarser partitions. However, it is possible to identify that the main drawback of this approach is the increasing number of classes and models used along the process which is translated to more computational resources required to infer the geographical location of the image.

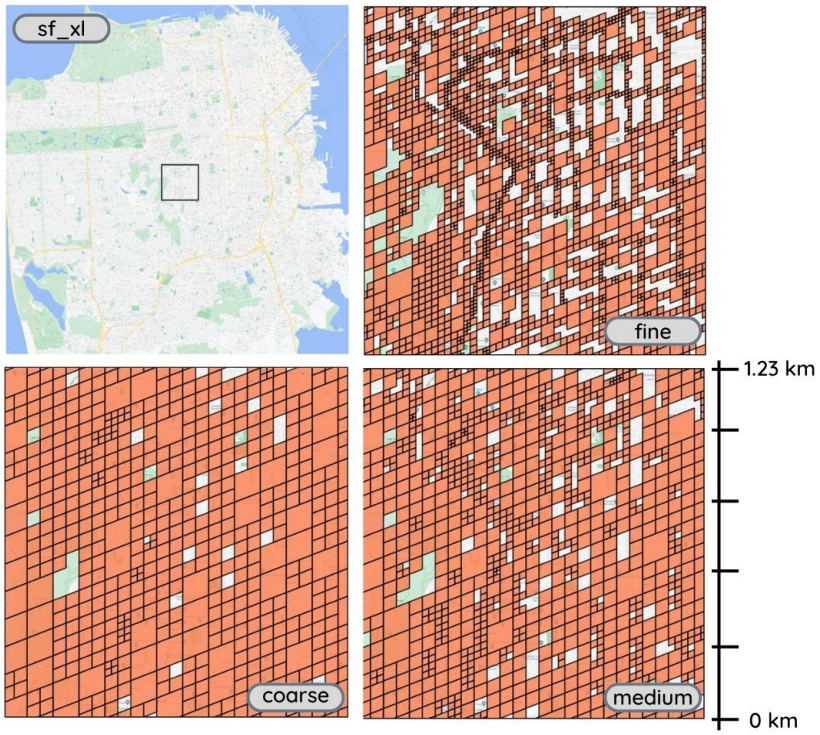


Figure 5.1: Example of the partitioning schema for HGE. It shows the partitionings: coarse, medium & fine, computed by varying the τ_{max} parameter into $[5000, 2500, 1250]$ respectively.

5.3 CPlaNet

As it was discussed in the previous sections, the CPlaNet framework aims to provide predictions at a fine granularity scale by combining the outputs of multiple models trained on coarser partitions. As it was already discussed, this combinatorial approach aims to reduce the number of computational resources with respect single-model approaches at the same level of granularity.

Experiment setup:

- Training-set: SF-XL training set

- Validation-set: SF-XL queries v2
- Test-set: SF-XL queries v1
- Backbone: EfficientNet_B0
- batch size: 64
- optimizer: adam
- learning rate: 0.0001
- s2sphere level: 17

Param. Group	Parameters	1	2	3	4	5
N/A	Num. of geoclasses	20,000	20,000	26,000	24,000	22,000
	Image Feature Dim	512	0	307	256	358
Node Score	Weight num. images	1.0	1.0	0.501	0.953	0.713
	Weight num. cells	0.0	0.0	0.499	0.047	0.287
Edge weight	Visual Distance	1.0	0.0	0.421	0.628	0.057
	geographical distance	0.0	1.0	0.579	0.372	0.943

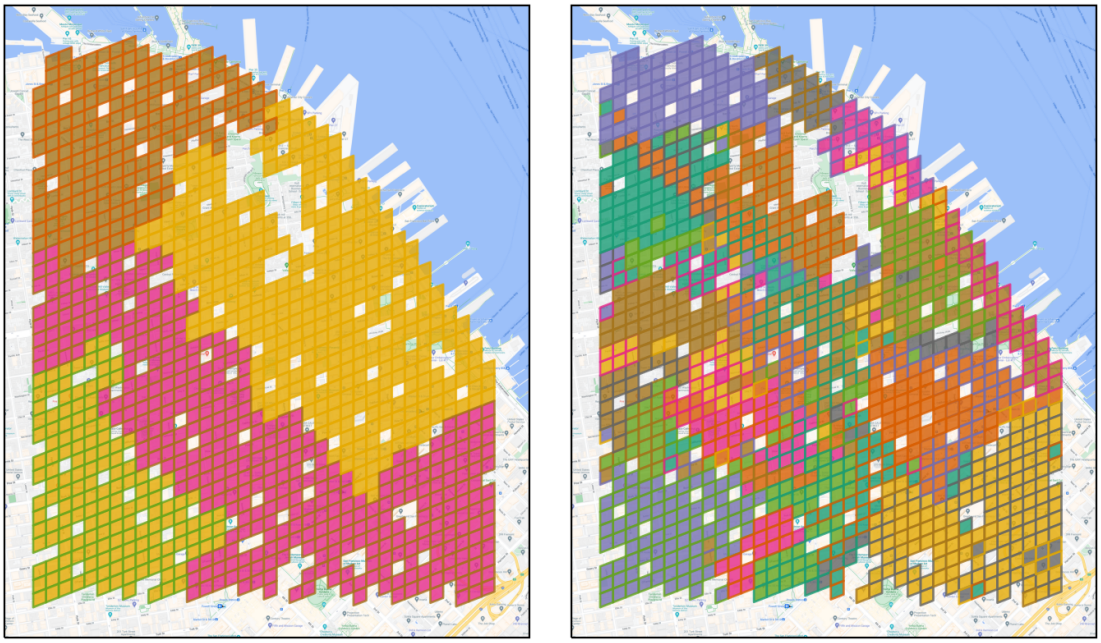
Table 5.3: CPlaNet parameters for SF-XL

N. geoclasses	N. intersections	Great Circle Distance (GCD)			
		25m	50m	100m	500m
10K/10K/11K/12K/13K	47,412	25.7	31.5	**32.9**	**38.9**
20K/20K/22K/24K/26K	54,144	**27.4**	**31.9**	32.8	37.9
30K/30K/33K/36K/39K	58,233	27.2	31.7	32.9	37.2

Table 5.4: Evaluating the performance of CPlaNet on SF-XL by varying the granularity of the partitioning schema

5.4 Comparison with retrieval approaches

At this point of the project we have seen how different classification-based VG approaches behave in large and highly dense datasets. However, we



(a) geoclasset 1 (b) combinatorial

Figure 5.2: Demonstration of the CPlaNet framework applied on a small portion of SF-XL considering 5 geoclassSets of 4 classes each. (a) The obtained geoclasset 2 applied to the area of interest, where the cells are iteratively combined based solely on geographic distance. (b) The combinatorial result showing the intersections between all the geoclassSets produced.

haven't seen a comparison with other traditional approaches to highlight the advantages of the classification-based approach, which we have discussed earlier to be ideal for this kind of applications. In Tab 5.5, it is presented a summary of the previous results, focusing at the GCD@25m metric, and a comparison with two of the most diffused methods in the retrieval framework, NetVlad [3] and CosPlace [6].

From these results it is possible to see that although the retrival-based methods are more accurate at the fine-granularity scale, their main drawback relies on the inference time, being from 100 to 10,000 times slower than classification-based solutions.

Method	Infer. time	GCD@25m	
		queries v1	queries v2
Classification:			
PlaNet [24]	12 ms	24.5	53.1
HGE [15]	15 ms	27.0	56.4
CPlaNet [20]	17 ms	27.4	64.1
Retrieval:			
NetVLAD [3]	12117 ms	40.0	71.1
CosPlace [6]	1514 ms	**64.7**	**83.4**

Table 5.5: **Comparison of results for a large number of methods using different approaches.** All inference times measures are averaged over 1000 queries, on a system with a RTX 3090 GPU and i9-10940X CPU.

Chapter 6

Experiments on smaller datasets

In the previous chapter, the performance of the various classification-based VG methods was evaluated using the large and highly dense dataset SF-XL. The goal was to simulate a realistic use case of the VG framework at a city-wide scale. At this point, we have already discussed how the granularity trade-off have an impact on the parameters of the partitioning schema, in particular, datasets on which there is a sufficiently large amount of images to achieve highly accurate results allow us to reduce the granularity of the classes without affecting the learning process.

On the other hand, this project seeks to evaluate the performance of the proposed classification-based VG methods on a wider range of urban scenarios, considering datasets with a smaller number of images to consider during training while covering low density areas. The goal of this section is test how by adapting the granularity of the partitioning schema it is possible to achieve good performances on these kind of scenarios. In particular, it was decided to conduct a series of experiments on a city-wide scale, aiming to test the already proposed classification-based VG methods on other popular urban and geo-tagged datasets, including different visual domains: Pitts250k[2], Pitts30k and Tokyo 24/7[1].

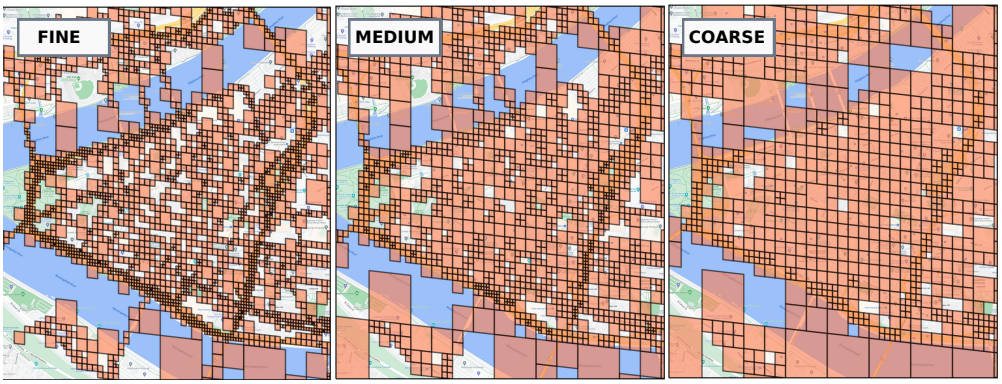


Figure 6.1: **Pitts250k partitioning schema for HGE**. It shows the partitions: coarse, medium & fine, computed by varying the τ_{max} parameter into $[500, 220, 120]$ respectively ($\tau_{min} = [4, 4, 4]$).

6.1 Pitts250k

6.1.1 PlaNet on Pitts250k

In Tab 6.1 it is presented the results of evaluating the performance of the PlaNet framework applied to the Pitts250k dataset. For this analysis a total of 4,578 classes were considered

Experiment setup:

- batch size: 64
- learning rate: 0.0001
- optimizer: adam
- τ_{min}: 4
- τ_{max}: 120

Backbone	Great Circle Distance (GCD)			
	25m	50m	100m	500m
EfficientNet-B0	26.9	34.5	40.7	71.5
EfficientNet-B5	29.6	38.0	44.4	74.5

Table 6.1: Evaluating the performance of PlaNet on Pitts250k using different versions for the feature extraction model.

6.1.2 HGE on Pitts250k

In Tab 6.2 it is presented the results of evaluating the performance of the HGE framework applied to the Pitts250k dataset. By following a similar to the previous chapter it was decided to improve the accuracy of the finest partition (same partition used in PlaNet) by aggregating coarser cells obtained by increasing teh τ_{max} by a factor of 2. The number of classes obtained were 4578, 2667 and 1355 (fine, medium & coarse)

Experiment setup:

- batch size: 64
- learning rate: 0.0001
- optimizer: adam
- τ_{min}: [4,4,4]
- τ_{max}: [500, 220, 120]

6.1.3 CPlaNet on Pitts250k

By looking at the results shown in Tab 6.3 we are able to see an improvement with respect the PlaNet baseline. In this case, it was considered the number of classes to have the same order of magnitude as in the coarse partition (1355 classes).

Experiment setup:

Backbone	Great Circle Distance (GCD)			
	25m	50m	100m	500m
EfficientNet-B0	28.5	37.6	44.7	76.2
EfficientNet-B5	32.0	40.2	47.7	73.7

Table 6.2: Evaluating the performance of HGE on Pitts250k using different versions for the feature extraction model.

- batch size: 64
- learning rate: 0.0001
- optimizer: adam
- num. classes: [1000, 1000, 1300, 1200, 1100]

Backbone	Great Circle Distance (GCD)			
	25m	50m	100m	500m
EfficientNet-B0	26.8	36.2	43.2	73.9
EfficientNet-B5	32.5	45.7	52.4	76.1

Table 6.3: Evaluating the performance of CPlaNet on Pitts250k using different versions for the feature extraction model.

6.2 Pitts30k

6.2.1 PlaNet on Pitts30k

Knowing that Pitts30k is a subset of Pitts250k, and considering that the PlaNet partitioning schema is based on the density of the cells it was decided to mantain the same parameter configuration as in the previous experiments, achieving a total of 486 classes. The PlaNet baseline for Pitts30k is provided on Tab 6.4

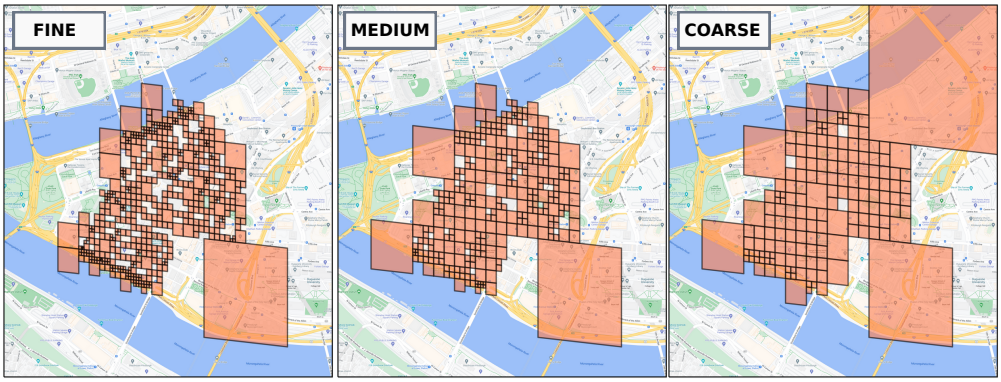


Figure 6.2: **Pitts30k partitioning schema for HGE**. It shows the partitionings: coarse, medium & fine, computed by varying the τ_{max} parameter into $[500, 220, 120]$ respectively.

Experiment setup:

- batch size: 64
- learning rate: 0.0001
- optimizer: adam
- τ_{min}: 4
- τ_{max}: 120

Backbone	Great Circle Distance (GCD)			
	25m	50m	100m	500m
EfficientNet-B0	30.3	41.5	51.5	88.9
EfficientNet-B5	38.9	49.4	58.5	89.5

Table 6.4: Evaluating the performance of PlaNet on Pitts30k using different versions for the feature extraction model.

6.2.2 HGE on Pitts30k

By looking at the results in Tab 6.5 we are able to see that the performance of the single-partitioning schema is improved when the ouptuts of coarser models are agregated into the final inference of the system. For this experiment there were obtained the following number of classes: 486, 272, and 158 (fine, medium and coarse respectively)

Experiment setup:

- batch size: 64
- learning rate: 0.0001
- optimizer: adam
- τ_{min}: [4,4,4]
- τ_{min}: [500, 220, 120]

Backbone	Great Circle Distance (GCD)			
	25m	50m	100m	500m
EfficientNet-B0	31.9	45.0	56.0	90.4
EfficientNet-B5	40.3	51.7	60.3	88.7

Table 6.5: Evaluating the performance of HGE on Pitts30k using different versions for the feature extraction model.

6.2.3 CPlaNet on Pitts30k

The last experiment on Pitts30k consisted of testing the CPlaNet with a distribution of the classes similar to the orders of magnitude as in the medium partition. In this case it was decided to work with these parameters in order to obtain a cell distribution similar to the previous scenario.

Experiment setup:

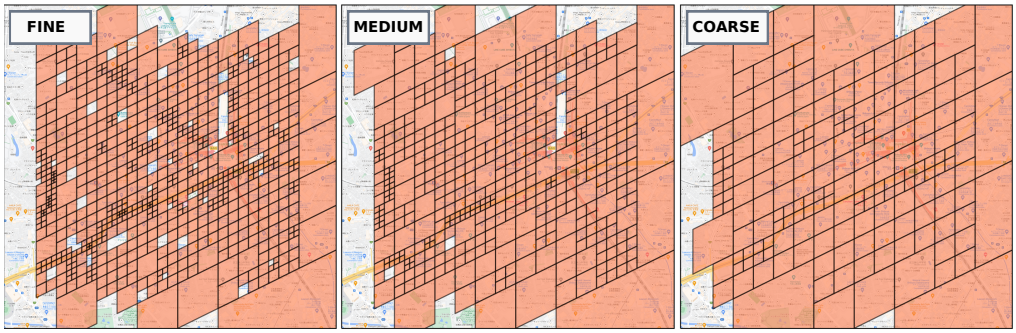


Figure 6.3: **Tokyo 24/7 partitioning schema for HGE**. It shows the partitionings: coarse, medium & fine, computed by varying the τ_{max} parameter into $[360, 170, 90]$ respectively.

- batch size: 64
- learning rate: 0.0001
- optimizer: adam
- num. classes: [270, 270, 310, 300, 290]

Backbone	Great Circle Distance (GCD)			
	25m	50m	100m	500m
EfficientNet-B0	33.5	48.5	58.8	91.2
EfficientNet-B5	47.5	61.7	69.8	91.0

Table 6.6: Evaluating the performance of CPlaNet on Pitts30k using different versions for the feature extraction model.

6.3 Tokyo 24/7

6.3.1 PlaNet on Tokyo24/7

Moving into the last dataset, the PlaNet baseline (Tab 6.7) was computed considering a partitioning schema with a similar distribution to the previous cases showed before. In this case, the number of classes obtained was 1840.

Experiment setup:

- batch size: 64
- learning rate: 0.0001
- optimizer: adam
- τ_{min}: 4
- τ_{max}: 90

Backbone	Great Circle Distance (GCD)			
	25m	50m	100m	500m
EfficientNet-B0	22.3	27.4	31.3	68.7
EfficientNet-B5	22.9	30.4	35.1	70.4

Table 6.7: Evaluating the performance of PlaNet on Tokyo 24/7 using different versions for the feature extraction model.

6.3.2 HGE on Tokyo24/7

Even in this case the previous baseline was improved by considering the output of coarser partitions at the inference process (Tab 6.8). In this case, the number of classes produced for each of the partitions were the following: 1840, 961, and 508.

Experiment setup:

- batch size: 64
- learning rate: 0.0001
- optimizer: adam
- τ_{min}: [4,4,4]
- τ_{max}: [360, 170, 90]

Backbone	Great Circle Distance (GCD)			
	25m	50m	100m	500m
EfficientNet-B0	22.5	31.8	38.8	73.6
EfficientNet-B5	24.9	33.4	38.0	73.1

Table 6.8: Evaluating the performance of HGE on Tokyo 24/7 by using different versions for the feature extraction model.

6.3.3 CPlaNet on Tokyo 24/7

Lastly, in Tab 6.9 it is shown the results of the CPlaNet framework applied to the Tokyo 24/7 dataset. In this case it was noticed a slight drop in the performance with respect the PlaNet baseline.

Experiment setup:

- batch size: 64
- learning rate: 0.0001
- optimizer: adam
- num. classes: [450, 450, 480, 470, 460]

Backbone	Great Circle Distance (GCD)			
	25m	50m	100m	500m
EfficientNet-B0	18.7	25.2	32.3	72.3
EfficientNet-B5	23.2	31.4	37.5	67.3

Table 6.9: Evaluating the performance of CPlaNet on Tokyo 24/7 by using different versions for the feature extraction model.

Chapter 7

Conclusions

On this project we examined the Visual Geolocation task, focusing on the most widely diffused classification-based approaches. In particular it was discussed the general overview and main idea behind the PlaNet, CPlaNet and HGE frameworks, explaining what are their main contributions in efficiently achieving accurate results in either city and planet-wide scenarios. It was also discussed why these types of solutions are faster than other widely diffused VG approaches, such as retrieval-based solutions, specially when applied to large and highly dense datasets. On the other hand, we discussed the main limitations for classification VG methods, highlighting the differences between finer and coarser approaches, and the importance of working at the right balance in the granularity trade-off.

In the second part of the project, it was listed a wide range of datasets commonly used in the development of VG applications, covering from small portions of a single city to extensive areas at the world wide scale. These datasets were then used to test and compare the efficiency of the classification approaches, implementing and testing a wide set of experiments aiming to highlight the advantages and disadvantages between classification and retrieval-based solutions.

Bibliography

[1] R. Arandjelovic A. Torii and J. Sivic. "24/7 place recognition by view synthesis". IEEE Conference on Computer Vision and Pattern Recognition, 2015.

[2] T. Pajdla A. Torii, J. Sivic and M. Okutomi. "visual place recognition with repetitive structures". page 883–890. IEEE Conference on Computer Vision and Pattern Recognition, 2013. doi: 10.1109/CVPR.2013.119.

[3] Relja Arandjelović, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. NetVLAD: CNN architecture for weakly supervised place recognition. 40(6):1437–1451, 2018. doi: 10.1109/TPAMI.2017.2711011.

[4] Y. Avrithis, Y. Kalantidis, G. Tolias, and E. Spyrou. Retrieving landmark and non-landmark images from community photo collections. In *in Proceedings of ACM Multimedia (Full paper) (MM 2010)*, Firenze, Italy, October 2010.

[5] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. pages 404–417. Springer Berlin Heidelberg, 2006. doi: 10.1007/11744023_32.

[6] Gabriele Berton, Carlo Masone, and Barbara Caputo. Rethinking visual geo-localization for large-scale applications. In *CVPR*, June 2022.

[7] D. M. Chen. "city-scale landmark identification on mobile devices,". page 737–744. IEEE Conference on Computer Vision and Pattern Recognition, 2011. doi: 10.1109/CVPR.2011.5995610.

[8] K. Grauman and T. Darrell. The pyramid match kernel: Efficient learning with sets of features. 2007.

[9] James Hays and Alexei A. Efros. "im2gps: estimating geographic information from a single image". page 1–8. IEEE Conference on Computer Vision and Pattern Recognition, 2008. doi: 10.1109/CVPR.2008.4587784.

[10] E. Muller-Budack J. Theiner and R. Ewerth. "interpretable semantic photo geolocation,". page 1474–1484, 2022. doi: 10.1109/WACV51458.2022.00154.

[11] D. G. Lowe. "distinctive image features from scale-invariant keypoints". page 91–110. International Journal of Computer Vision, 2004. doi: 10.1023/B:VISI.0000029664.99615.94.

[12] M. Douze M. Paulin. Local convolutional features with unsupervised training for image retrieval. pages 91–99, 2015.

[13] Christoph Strech Michael Calonder, Vincent Lepetit and Pascal Fua. Brief: Binary robust independent elementary features. volume 6314. Springer Berlin Heidelberg, 2010. doi: https://doi.org/10.1007/978-3-642-15561-1_56.

[14] Evangelos Papalexakis Mike Izbicki and Vassilis Tsotras. Exploiting the earth's spherical geometry to geolocateimages. page 3–19. Springer Berlin Heidelberg, 2019.

[15] Eric Müller-Budack, Kader Pustu-Iren, and Ralph Ewerth. Geolocation estimation of photos using a hierarchical model and scene classification. In *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XII*, volume 11216 of *Lecture Notes in Computer Science*, pages 575–592. Springer, 2018. doi: 10.1007/978-3-030-01258-8_35. URL https://doi.org/10.1007/978-3-030-01258-8_35.

[16] N. Jacobs N. Vo and J. Hays. "revisiting im2gps in the deep learning era.". 2017.

[17] J. Matas O. Chum and S. Obdrzalek. Enhancing ransac by generalized model optimization. pages 812–817, 2004.

[18] A. Oliva and A. Torralba. Building the gist of a scene: The role of global image features in recognition. page 23–36, 2006.

[19] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *2011 International Conference on Computer Vision*, pages 2564–2571, 2011. doi: 10.1109/ICCV.2011.6126544.

[20] Paul Hongsuck Seo, Tobias Weyand, Jack Sim, and Bohyung Han. Cplanet: Enhancing image geolocalization by combinatorial partitioning of maps. In *ECCV*, 2018.

[21] Sivic and Zisserman. Video google: a text retrieval approach to object matching in videos. In *Proceedings Ninth IEEE International Conference on Computer Vision*, pages 1470–1477 vol.2, 2003. doi: 10.1109/ICCV.2003.1238663.

[22] Bart Thomee. "yfcc100m: the new data in multimedia research,". volume 59, page 64–73, 2019. doi: 10.1145/2812802.

[23] Tobias Weyand and Bastian Leibe. Visual landmark recognition from internet photo collections: A large-scale evaluation. Elsevier BV, jun 2015. doi: 10.1016/j.cviu.2015.02.002. URL https://doi.org/10.1016%2Fj.cviu.2015.02.002.

[24] Tobias Weyand, Ilya Kostrikov, and James Philbin. Planet - photo geolocation with convolutional neural networks. 2016.