Divide&Classify: Fine-Grained Classification for City-Wide Visual Place Recognition

Divide&Classify: Fine-Grained Classification for City-Wide Visual Place Recognition

Gabriele Trivigno*[1] Gabriele Berton*[1] Carlo Masone[1] Juan Aragon[1] Barbara Caputo[1]
[1] Politecnico di Torino
{gabriele.trivigno, gabriele.berton, carlo.masone, barbara.caputo}@polito.it juanm.aramas@gmail.com

Abstract

Visual Place recognition is commonly addressed as an image retrieval problem. However, retrieval methods are impractical to scale to large datasets, densely sampled from city-wide maps, since their dimension impact negatively on the inference time. Using approximate nearest neighbour search for retrieval helps to mitigate this issue, at the cost of a performance drop. In this paper we investigate whether we can effectively approach this task as a classification problem, thus bypassing the need for a similarity search. We find that existing classification methods for coarse, planet-wide localization are not suitable for the fine-grained and city-wide setting. This is largely due to how the dataset is split into classes, because these methods are designed to handle a sparse distribution of photos and as such do not consider the visual aliasing problem across neighbouring classes that naturally arises in dense scenarios. Thus, we propose a partitioning scheme that enables a fast and accurate inference, preserving a simple learning procedure, and a novel inference pipeline based on an ensemble of novel classifiers that uses the prototypes learned via an angular margin loss. Our method, Divide&Classify (D&C), enjoys the fast inference of classification solutions and an accuracy competitive with retrieval methods on the fine-grained, city-wide setting. Moreover, we show that D&C can be paired with existing retrieval pipelines to speed up computations by over 20 times while increasing their recall, leading to new state-of-the-art results.

1. Introduction

Visual Place Recognition (VPR) is the task of recognizing the location where a photo was taken with an accuracy of a few meters [43, 1, 25, 42, 28, 11, 13, 5, 18, 49, 31, 15, 44, 52, 6, 8, 7, 12, 24, 16]. This problem, also known as visual geo-localization [3, 4, 25] or image localization [28, 13], is commonly approached as an image retrieval problem: the query to be localized is compared to a database of geo-tagged images, typically via a k-nearest neighbour (kNN) in features space, and the most similar images retrieved from the database are the predictions of the query's location. Even though retrieval methods work remarkably well when the VPR task is limited to a small map [4, 52, 3], scaling them to large and densely mapped areas, such as an entire city, is impractical because both the time and memory required to execute the kNN grow with the dimension of the database [52, 4, 35, 2].

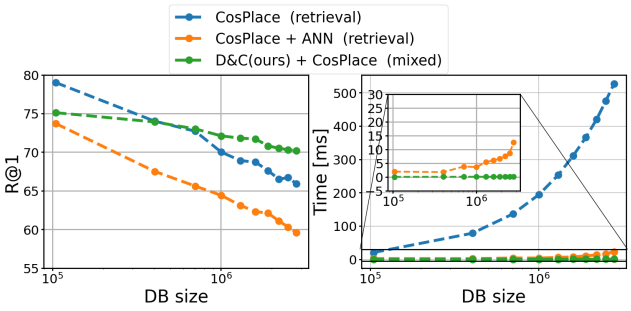

Figure 1: **Experiments** demonstrating the scalability problem of retrieval-based VPR methods, using the state-of-the-art CosPlace (with exhaustive kNN and with Approximate Nearest Neighbor - ANN - search through Inverted File Index with Product Quantization, IVFPQ). Combining our Divide&Classify method with the retrieval approach yields an optimal performance-efficiency trade-off when scaling up to the city-wide setting.

Certainly, the space and time requirements of the kNN can be reduced by resorting to Approximate Nearest Neighbors (ANNs) algorithms [22, 29, 2, 39], which grant impressive speed ups while simultaneously reducing memory footprint. However, their accuracy may be significantly worse than the one obtained with the exhaustive kNN, which is their upper bound. This trade-off is easily demonstrated with an experiment. In Fig. 1 we show the behavior obtained using the state-of-the-art retrieval method CosPlace [3] on SF-XL [3], a dataset for urban VPR that cov-

ers the whole city of San Francisco, with an area of nearly 170km^2 and over 40M images. Starting from a reduced version of SF-XL and gradually increasing the size of the database up to its full version, we can observe that the inference time quickly explodes with the exhaustive kNN. On the other hand, with an ANN the inference time grows much more slowly (yet still linearly w.r.t. the size of the database), at the cost of a big drop in accuracy.

A different route to address the scalability problem is to frame VPR as a classification task, so that predictions of the query's location can be obtained without a similarity search over a database. So far, this formulation has been applied in the setting of planet-wide, coarse geolocalization [33, 50, 38, 27]. All these methods are designed to divide the globe in very coarse classes, spanning up to hundreds of kilometers each and following the sparse and uneven distribution of photos in the planet. We question whether these classification approaches can be adapted to the fine-grained city-scale VPR setting, where the required localization accuracy is in the order of a few meters. We find that the global-scale classification methods [33, 50, 38, 27], albeit being much faster than their retrieval counterparts, are unsuitable for this setting because they are far too imprecise and because they do not account for the visual overlap among classes that arises when the photos are densely sampled from the map.

Therefore, we propose a novel classification-based VPR technique that is specifically designed for the dense and homogeneous configurations of large urban areas. Our method, called **Divide&Classify** (D&C), has the speed advantage of classification approaches while being competitive with retrieval methods in terms of accuracy. Most importantly, we demonstrate that the predictions from D&C can be used to restrict the search space of retrieval methods, combining both in a unique pipeline with more accurate results than previous works and a faster (and constant) inference time (see Fig. 1). We also show that such a combination of classification and retrieval is not effective with currently viable classification methods for planet-wise geolocalization due to their lack of accuracy.

Contributions. To summarize our contributions:

- We are the first to tackle the problem of fine-grained (error $\leq$ 25m) and city-wide (map area $>$100 km^2) VPR through classification, demonstrating the inadequacy of existing global-scale approaches in this setting and proposing a first feasible solution (D&C).
- In D&C we propose a new classifier, named Additive Angular Margin Classifier (AAMC), which uses the learned prototypes from a Additive Angular Margin Loss to classify new images. The AAMC is scalable and produces remarkably robust results.
- We show that our method not only achieves competitive results with retrieval methods, but above all it can be combined to create a fast, scalable and accurate pipeline that harnesses the best of both worlds.

Code and trained models will be made publicly available upon acceptance of this paper.

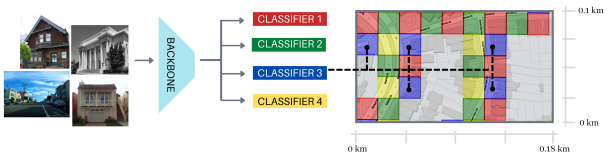

Figure 2: **Architecture of Divide&Classify**.The different groups, and their relative classifiers, are color coded. The picture explicates how cells are distributed into groups.

2. Related Work

Retrieval-based VPR. Most commonly, approaches for VPR are retrieval-based, where the predicted coordinates are obtained with a similarity search over a database of pre-computed embeddings of either local [9, 20, 23, 34] or global features [1, 25, 28, 51, 52]. In recent works, deep feature-extractors have become the de facto standard, complemented with an aggregation or pooling layer. A notable example of the former is NetVLAD [1], in some cases equipped with attention mechanisms [25]. Popular pooling strategies are [37, 41, 36]. A complete discussion on retrieval-based algorithms can be found in [4, 30]. All these approaches leverage contrastive learning via triplet loss, which requires a mining procedure to compute a cache containing suitable triplets, to be updated periodically. Due to this formulation, all retrieval pipelines suffer from poor scalability: at *training time*, as the database size increases, the computation required for mining becomes predominant with respect to the actual training; at *test time*, the time required to perform the similarity search grows linearly with the database size, thus leading to potentially unacceptable delays for applications. Recently, CosPlace [3] has addressed the training time scalability problem by using an alternative approach to contrastive learning, allowing to learn from large scale databases and achieving state of the art results across many datasets. Yet, it uses retrieval for inference, thus the scalability problem at test time still persists.

Classification-based VPR. There is another branch of literature that casts the place recognition problem as a classification task, but on a world-wide scale. Seminal works in this setting are RevIm2GPS [45] and PlaNet [50]. These global-scale classification methods partition the earth into a set of geo-classes, and the geographic center of the predicted class is used as output geolocation. This approach to VPR has many advantages, mainly in terms of space and time complexity. However, the final accuracy is highly de-

pendent on the adopted partitioning scheme. Fine-grained partitions are necessary to increase the resolution of the localization, but scaling up the number of classes is not trivial as it causes the number of parameters to grow and reduces the number of samples per class. In Hierarchical Geolocation Estimation (HGE) the authors exploit a geographic hierarchy of classes to mitigate this issues [33], whereas CPlaNet [38] is based on a combinatorial partitioning of multiple geoclass sets. Other works [27, 19] aim at learning the classes centers. For the fine-grained urban setting, the idea of addressing it as a classification problem has been explored back in 2013 [14] using SVM classifiers, however only in a small geographical area (≈ 1.56 km^2) and not on a city-wide scale.

Relationship with prior works. Our work is related to both retrieval and classification methods; we leverage intuitions from previous state-of-the-art (SOTA) in both fields and build a new technique specifically suited to tackle the city-wide localization task.

With respect to CosPlace [3] (**retrieval-based** SOTA), we use the same concept of **Groups** *i.e.* a set of geoclass partitionings. In CosPlace, only a subset of groups is used, as the target is learning to extract features meaningful for localization. Instead, in this work we aim at learning a distribution that covers geographically the entire city, so we train on all the Groups. Moreover, in CosPlace the learnt classifiers (one for each group) are discarded at inference time, and predictions are obtained via similarity search in a database, thus incurring in costly memory and time requirements. Contrarily, we keep all the classifiers and we exploit their "collective wisdom" to quickly obtain predictions.

Regarding **classification-based** methods, there is not a clear SOTA: the most popular works PlaNet [50] and CPlaNet [38] train on private datasets, thus resulting in unfair comparisons to other methods, and they have no public implementation. Comparisons in previous literature are further hindered by the fact that methods rely on different backbones. Therefore, ours is the first fair comparison of existing classification methods in a fine-grained and city-scale setting. In terms of similarities with our method, CPlaNet also uses the idea of merging predictions from multiple classifiers: the authors create 5 discrete partitions of the earth, and perform inference on the intersection of the classes in these partitions, with a combinatorial scheme. These partitions are overlapping and each one contains geographically adjacent classes. On the contrary, we show that in the geographically dense city-wide setting (as opposed to the sparse planet scale one) a discrete partitioning of adjacent classes seriously impedes the learning capabilities of a classifier and our partition is designed to prevent this phenomenon. In Sec. 3.1 we further detail this reasoning, and experimental results in Sec. 4 confirm this claim. Another key difference is that CPlanet's classes are irregularly shaped, formed as intersection of overlapping partitions and logits are assigned using a combinatorial scheme. As a result, inference is costlier. Instead, our method uses a simple partitioning scheme that makes it faster, while also largely outperforming competitors.

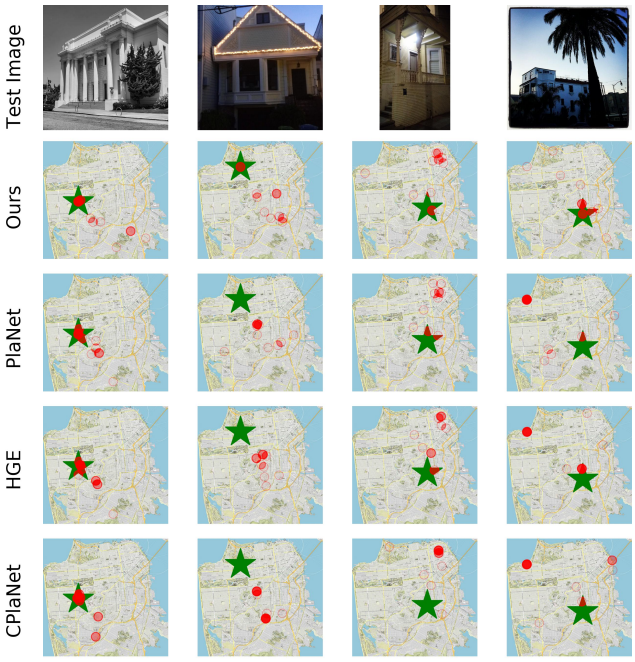

Figure 3: **Qualitative examples of predictions for each method.** The green star shows the ground truth of each test image, while the red circles represent the first 10 predictions. Brighter red indicates multiple predictions close to each other.

Finally, we experimentally demonstrate in Sec. 4.2.1 that our approach can be combined with retrieval methods into a single pipeline with faster inference time and better accuracy. This is possible thanks to D&C's accuracy which allows to use it as a distractor-filter for the retrieval, whereas the existing classification-based approaches are too imprecise and combining them with retrieval worsens results.

3. Method

We consider the VPR task in a large urban environment (*i.e.* a city covering > 100km^2) and we deem a place correctly recognized if its predicted location is within 25m from the ground truth (as commonly done in literature [1, 49, 28, 13, 5, 15, 47, 4, 52]). Furthermore, we assume to have a training set of geo-tagged urban images $\mathcal{T} = \{(I_i, east_i, north_i)\}$, where I_i is an image and the pair $(east_i, north_i) : \mathbb{UTM}_e \times \mathbb{UTM}_n$ represents its UTM coordinates (which are an approximation of GPS coordinates of a local geographical area over a flat surface). The training set $\mathcal{T}$ is created by sampling densely over the city

(*i.e*. roughly one image per meter of road), because we want to achieve precise localization.

With this setting, our goal is to learn from $\mathcal{T}$ a classifier to perform the VPR task. In pursuit of this objective, we rely on two key ingredients. Firstly, we partition $\mathcal{T}$ in disjoint sets of cells (*i.e*. classes) so that there is no visual overlap among distinct classes included in the same split. Secondly, we use an ensemble of classifiers, one for each split of $\mathcal{T}$. In particular, we propose a novel type of classifier, called Additive Angular Margin Classifier (AAMC), which inherits the discriminative power of the Additive Angular Margin Loss [46]. We thoroughly explain our partitioning method in Sec. 3.1 and the AAMC in Sec. 3.2.

3.1. Partitioning method

Splitting the training set $\mathcal{T}$ in classes is non-trivial given that the label space (GPS or UTM coordinates) is continuous. Moreover, in light of the high, uniform density that we deal with, learning over a discretized geographical partitioning with a vanilla classification framework is an ill-posed problem. The reason for this is that images at the boundaries of neighboring cells can have very similar appearance, and consequently similar embeddings, while pertaining to different classes. This results in noisy gradients and overall it hinders a model's learning ability. To address this issue, we adopt a partitioning method that produces multiple disjoint splits of the training set, preventing any classes in the same split from sharing a boundary. In practice, this avoids *perceptual aliasing* between different classes in the same split.

The core idea of our partitioning method is to *build different splits made of* ***non-adjacent*** *geographic cells, and learn a classifier on each split* (see Fig. 2). This way each cell (akin to a class) is assigned to a exactly one classifier, while its neighboring cells are assigned to separate classifiers.

The scheme that we adopt to split the dataset into cells is straightforward and it is inspired by CosPlace [3]: we divide the map into equal-sized square cells, that are defined by a single hyper-parameter M, representing the length of the side of the square. Each cell corresponds to a class. Since the geo-localization task requires to output a set of coordinates, we define a simple function Class2UTM that maps each class to a set of coordinates (the center of its corresponding cell), which will be used as prediction. Formally, a class $C_{e_i,n_j} \in \mathcal{S}$ is defined as :

$$C_{e_i,n_j} = \left\{ (east, north) : \left\lfloor \frac{east}{M} \right\rfloor = e_i, \left\lfloor \frac{north}{M} \right\rfloor = n_j \right\} \tag{1}$$

and the function Class2UTM is

$$\begin{aligned} \text{Class2UTM} : &\mathcal{S} \to \mathbb{UTM}_e \times \mathbb{UTM}_n, \\ &C_{e_i,n_j} \mapsto \big((e_i + 0.5) \cdot M,\ (n_j + 0.5) \cdot M\big) \end{aligned} \tag{2}$$

Finally, we partition the set of cells into separate Groups: we therefore set a separate hyperparameter N and define each Group as

$$G_{uv} = \left\{ C_{e_i n_j} : (e_i \bmod N = u) \ \wedge \ (n_j \bmod N = v) \right\} \tag{3}$$

The parameter N defines the minimum distance (in number of cells) between two nearby classes within the same split. Moreover the value of N determines the number of groups $|G|$ that are created from the dataset, with $|G| = N^2$. In Sec. 4.3 we empirically find the best values for the hyperparameters to be $M = 20$ meters and $N = 2$. Note that, unlike the partitioning used by CosPlace [3], we do not need the training images to be labeled with a heading angle. This makes our solution more widely applicable.

3.2. Additive Angular Margin Classifier (AAMC)

Using the Groups formulation we effectively obtain a set of $|G|$ independent partitions of the cells, so we can use a *Mixture-of-Experts* approach and assign a separate classifier to each partition (see Fig. 2). Recently, several studies [10, 46, 3] have found that using large-margin losses when training siamese architectures for retrieval leads to more discriminative embeddings and a better structured latent space [32, 48]. However, these approaches have only ever been used to train a feature extractor; thus the learnt prototypes were never applied to perform classification at test time. Motivated by the fact that we want to enjoy the benefits that these losses provide, we propose a novel classifier that allows to exploit the highly informative prototypes learnt during training.

To this end, we build upon the Additive Angular Margin Loss (ArcFace) [10], which has been used in prior large scale retrieval works [46, 40, 3]. The ArcFace allows to learn highly discriminative embeddings by maximizing the inter-class angular distance, measured via the cosine similarity between a matrix of learnable class prototypes and the normalized embeddings. Formally, the ArcFace loss is defined as

$$\mathcal{L}_{arc} = \frac{1}{N} \sum_i -\log \frac{e^{s \cos(\theta_{y_i} + m)}}{e^{s \cos(\theta_{y_i} + m)} + \sum_{i \neq j} e^{s \cos \theta_j}} \tag{4}$$

subject to

$$\begin{aligned} \cos \theta_j &= W_j^T x_i \\ W = \frac{W^*}{||W^*||}, \quad x &= \frac{x^*}{||x^*||} \end{aligned} \tag{5}$$

where x_i is the i-th embedding corresponding to the ground-truth class of y_i and W_j is the prototype vector of the j-th class.

Previous works [46, 40, 3] only use the ArcFace (or similar angular margin losses) to learn to extract embeddings

that can be used in a retrieval pipeline, discarding the prototypes. Instead, we argue that these prototypes learn a meaningful mapping from the embeddings to each class and can therefore be exploited directly to classify. To the best of our knowledge, this is the first work that uses the prototypes learnt with the ArcFace at inference time.

We call this approach Additive Angular Margin Classifier (AAMC). Namely, the generic k-th classifier (*i.e.* the classifier assigned to the k-th Group) is characterized by a matrix $W_k \in \mathbb{R}^{S_k \times d}$ of d-dimensional class prototypes, where S_k is the number of classes in group k and d is the dimensionality of the backbone's embeddings. At inference time, given the embedding of a test image $x \in \mathbb{R}^d$, we extract a set of $|G|$ normalized predictions $\{p_k = \text{softmax}(W_k^T x) \in \mathbb{R}^{S_k}\}_{k=1}^{|G|}$.

Among them, we choose the one with maximum confidence across all classifiers

$$\begin{aligned} &\text{D\&C} : \mathbb{R}^{S_1} \times \ldots \times \mathbb{R}^{S_{|G|}} \rightarrow \mathbb{UTM}_e \times \mathbb{UTM}_n, \\ &\{p_k\}_{k=1}^{|G|} \mapsto \text{Class2UTM}(\underset{k \in 1..|G|}{\text{argmax}} \; \underset{c \in S_k}{\text{argmax}} \; p_k(c)) \end{aligned} \tag{6}$$

where $p_k(c)$ denotes the c-th element of the vector p_k. In summary, each classifier provides the logits distribution on the classes pertaining to its group. To obtain the final prediction on the entire geographical support for a given image, we choose the single prediction with the highest confidence across all AAMC classifiers.

The intuition behind our *mixture-of-AAMC* approach is that while the class containing a given test image only belongs to a single classifier, the other classifiers are likely to predict nearby classes, given that classes close to each other share similar visual features. In Fig. 7 we further detail this behavior, studying the agreement and the correlation among prototypes in different AAMC classifiers.

4. Experiments

4.1. Experimental Setting

Implementation details. The split of the dataset in classes is performed using $M = 20$, meaning that each class is a square cell of 20 meters per side. The classes are then grouped together using $N = 2$, which leads to the creation of 4 groups and 4 classifiers within our model. This creates over 113k classes for the SF-XL training set; Fig. 4 gives a visual representation of the outcome. Further analysis on the number of classes generated over the various datasets can be found in the Supplementary.

The classifiers are trained independently one per epoch. We train for 1M iterations with batch size 64. Given the total of 4 classifiers, each ons is therefore trained once every 4 epochs using Adam [26] as optimizer with learning rate of $1e^{-4}$. Following [27], we use an EfficientNet as backbone.

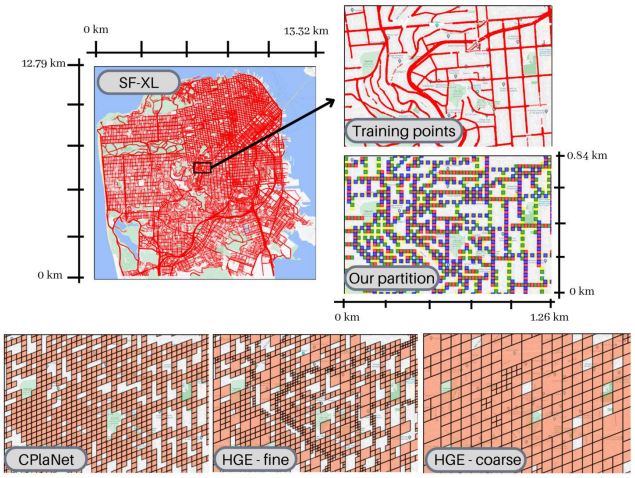

Figure 4: **Maps showing multiple partition methods over the training data of SF-XL.**

For fairness, we compute results using the same batch size, data augmentation, backbone and training set for all classification methods. Note that this is in contrast with previous works [50, 38, 33, 27, 19] that use different backbones (and different training dataset) for all methods (see Sec. 2), which makes it difficult to discern if the reported performance improvements was given by the method, a stronger backbone or better training data. For D&C, the backbone is followed by an average pooling and a whitening layer [36], which provides the input for the classifiers.

To assess the quality of different techniques on the task of city-wide visual place recognition, we run our experiments on the large-scale dataset of San Francisco eXtra Large [3], which consists of a training set of 41.2M images that densely maps the whole urban area of San Francisco ($>$100 km^2). The dataset provides two test sets, namely *test set v1* and *test set v2*, both of which are collections of photos taken with smartphones. To this date, this is the only dataset well representative of a fine-grained and city-wide VPR application, that is our target setting. Nevertheless, in Sec. 4.4 we also discuss the use of Divide&Classify with much smaller urban datasets, covering less than 3 km^2, to discuss its limited effectiveness for small scale problems.

Metrics. Visual place recognition methods are assessed using different metrics, depending on the type of method. Traditionally, retrieval methods for fine-grained VPR are evaluated using the recall@N (R@N) with a threshold of 25 meters [1, 28, 25, 13, 3, 4] for correct matches, whereas classification methods for planet-scale localization use the Great Circle Distance (GCD) [17], which measures accuracy within a given threshold distance. In order to properly compare retrieval and classification methods, we introduce a new metric called **Localization Radius@N** (LR@N) that

summarizes both: *Given the top-N predictions for a certain localization algorithm, it evaluates the percentage of queries which are correctly localized within 25 meters of the ground truth, in at least one of the top-N predictions.*

Although the LR@N is equivalent to the R@N @25m used in retrieval, its definition makes it compatible with a classification pipeline. In fact, the LR@1 is also equivalent to GCD@25m.

Baselines overview. We split baselines in three categories:

1) Previous *classification methods*, designed for world-wide geolocalization, that can be readily repurposed for city-wide geolocalization. Among these, we use PlaNet [50], Hierarchical Geolocation Estimation (HGE) [33], CPlaNet [38] and MvMF [19]. Note that, unlike Divide&Classify, all these methods rely on the S2 Sphere library, and use hyperparameters for class partitions that have been tuned on global-scale classification: for fair comparison between methods, we performed a thorough grid search of such hyperparameters on SF-XL, which we used to compute the results in Tab. 1. The final splits for each method are reported in Fig. 4. To compute results with classification methods (some of which do not have open source implementations), we reproduced their results on world-wide datasets, and then we used the code to train on the city-scale setting. We will release the code also for the implementations of such methods like PlaNet, CPlaNet, and MvMF. We provide further analysis on the optimal hyperparameters for each partitioning method in the Supplementary.

2) *Retrieval methods* for visual place recognition, using deep neural networks to produce descriptors which are then matched to the query's. For our experiments we use the ever-green NetVLAD [1], CRN [25], SARE [28], SFRS [13] and CosPlace [3].

3) *Retrieval methods* combined with *Approximate Nearest Neighbor* (ANN), in a pipeline that uses the best retrieval method, namely CosPlace, with the ANNs that have shown best results, namely Hierarchical Navigable Small Worlds (HNSW) [29] and Inverted File Index with Product Quantization [39, 22]. We chose the two optimal configurations, one designed for speed or LR@1. Extensive experiments with other ANN methods, as well as different hyperparameters, are shown in the Supplementary.

4.2. Main Results

Comparison with previous works. We report quantitative comparisons between D&C and existing baselines in Tab. 1. The results show some clear trends, which can be summarized in a few points:

- Previous **classification methods**, being designed for the uneven distribution of world-wide datasets, fail to achieve competitive results on SF-XL. Qualitative results are reported in Fig. 3.

Method	Infer. time	LR@1	
		test v1	test v2
Classification			
PlaNet [38]	12 ms	24.5	53.1
HGE [33]	15 ms	27.0	56.4
CPlaNet [38]	17 ms	27.4	64.1
MvMF [19]	12 ms	22.6	52.2
D&C(ours)	12 ms	61.0	79.1
Retrieval			
NetVLAD [1]	12117 ms	40.0	71.1
CRN [25]	12117 ms	45.8	76.4
SARE [28]	12117 ms	45.5	78.8
SFRS [13]	12117 ms	51.2	83.1
GeM [36]	1514 ms	21.7	43.1
CosPlace [3]	1514 ms	64.7	83.4
Retrieval + Approximate Nearest Neighbor			
CosPlace [3] + HNSW [29]	4 ms	52.5	77.8
CosPlace [3] + IVFPQ [22] *	8 ms	55.1	82.6
CosPlace [3] + IVFPQ [22] *	141 ms	63.7	83.3
Mixed pipeline			
D&C(ours) + CosPlace [3]	30.8 ms	**71.4**	**87.6**

Table 1: **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. The FAISS library [21] is used for all nearest neighbor implementations. *Mixed pipeline* is the best configuration from Tab. 2, which performs retrieval on the top-100 classes obtained through D&C. *We show two versions of the Inverted File Index with Product Quantization, one tuned for speed and one for recall.

- D&C, being specifically designed for learning in a dense urban setting is able to outperform previous classification methods on city-wide geolocalization, and it is almost competitive with the retrieval-based state of the art, (LR@1 of 3.7 points lower on the *test v1*).
- **Approximate Nearest Neighbor** (ANN) search algorithms provide different implementations, some of which are able to speed up retrieval methods by 10x with a slight drop in LR@1, and others speeding up retrieval by almost 400x at the price of a 12 points drop in LR@1 on the *test v1*.
- The **Mixed Pipeline** of SOTA classification (D&C) and retrieval (CosPlace) methods reaches speed on pair with classification methods, while providing a large improvement over any other results. More details on this mixed pipeline are presented in the next section.

4.2.1 Classification + retrieval

In this section we analyze how classification methods can be pipelined with retrieval ones into a single system, with the

Retrieval Method	Top-N	kNN Time (*ms*)	Classification Method & Time D&C (12 ms) LR@1	HGE (15 ms) LR@1	CPlanet (17 ms) LR@1
NetVLAD	All	12117	40.0	40.0	40.0
	1000	42	50.6	42.8	38.7
	100	4	56.7	41.1	37.4
	10	0.6	**62.6**	35.6	34.6
	1	0.06	56.1	24.8	26.3
CosPlace	All	1514	65.9	65.9	65.9
	1000	8	70.3	62.0	57.5
	100	1	**71.4**	52.7	51.0
	10	0.1	70.2	40.9	42.4
	1	0.03	57.0	25.0	26.7

Table 2: **Results of classification + retrieval pipelines** on SF-XL test v1. The *Top-N* column represents the number of cells within which we compute retrieval. The rows with *Top-N: All* are equivalent to using retrieval only, while the other rows employ the classification filter, reducing the search space by orders of magnitude. For retrieval, we use a VGG16 backbone. NetVLAD's dimensionality is 4096-D PCA (extraction time 2.1 ms), while CosPlace has 512-D (extraction time 5.1 ms).

aim of improving accuracy and speed of results. The idea is to restrict the search space for the retrieval's kNN search only to the cells that have been predicted with higher confidence by the classification model. For example, when using only the first 10 classes (Top-10), images within the 10 cells where the model predicts the location of the test image with the highest confidence are then used for retrieval. Figure 5 visually exemplifies how the filtering on the Top-N classes effectively removes distractors from the search space. To provide a relevant analysis, we adopt different classifications models, namely HGE, CPlaNet and D&C, while for retrieval we use models trained with NetVLAD [1] and CosPlace [3], which are respectively the most popular and the most recent SOTA. We report the results in Tab. 2: we separately show the multiple components of the total inference time using a two-stages pipeline, namely the classification, the descriptors extraction and kNN. Regarding descriptors extraction, we only consider the extraction of the test image (query) descriptors, given that the ones from the database can be extracted offline. In the table, when Top-N=All it means that there is no filtering and the pipeline is the same as the pure retrieval method.

The results clearly show the huge benefit of using a two-stages classification + retrieval pipeline for large-scale visual geo-localization. We find that such a pipeline leads to faster inference and better results w.r.t. a standard retrieval system. The former is mostly due to the reduction in search space, which reduces the number of database descriptors by the kNN. The latter is due to the filtering of cells to which

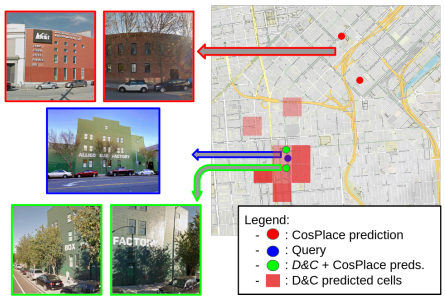

Figure 5: Example of our **mixed pipeline**. Thanks to the reduction in the search space obtained via the predictions of D&C, the retrieval module correctly localizes the query

the classification model assigns a low probability of containing the test image, therefore simplifying the retrieval task by eliminating distractors.

Among the most notable results, using the Top-100 predictions from D&C, CosPlace achieves a new SOTA (+ 6% LR@1), while being 500 times faster than the retrieval only version. Similarly, the NetVLAD model can achieve a speedup by 4 orders of magnitude and an increase in LR@1 by 20% when the retrieval is performed only on the Top-10 predicted classes. Figure 1 visually shows how D&C provides a much more scalable alternative to SOTA retrieval methods, even when retrieval is sped up by the best approximate nearest neighbor search.

4.3. Ablations

Ablation on class partitions. Figure 6 reports an ablation study on the values of M and N. Considering that the number of classifiers is $|G| = N^2$ (Sec. 3.1), each one trained independently once every $|G|$ epochs (Sec. 4.1), we can see that having a large number of classifiers leads to each one of them being trained too seldom for it to reach good performances; on the other hand when using $N = 1$ we incur in the learnability issues discussed in Sec. 3.1, due to adjacency among different classes of the same group. Finally, $N = 2$ stands out as the obvious best choice.

Regarding the value of M (defining the side of a cell), 20 m turns out to be the best choice. Values of 50 m and 100 m produce worse results. Understandably, this is because their classes encapsulate a greater variability wrt their finer counterparts, and thus are harder to learn.

Ablation on the loss. Once the dataset is split into groups and classes, a natural choice of loss would be the Cross-Entropy (CE) loss. This can be easily implemented by using one linear layer for each group, training each classifier sequentially group by group. However we empirically found (see Tab. 3) that the AAMC constantly outperforms a set of linear classifiers trained with a cross-entropy loss. This is thanks to the formulation of the loss as margin maximiza-

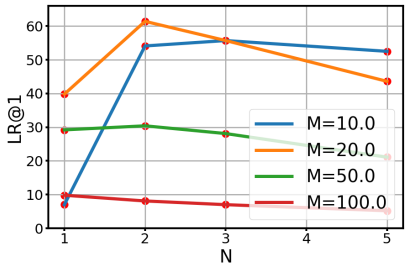

Figure 6: **Ablation** on the values of M and N. M determines cell size, N is the distance between cells in a group.

Classifier	LR@N (at 25 m)			
	LR@1	LR@5	LR@10	LR@20
Cross-Entropy	48.2	62.6	68.0	72.0
AAMC	**61.4**	**73.6**	**77.1**	**79.6**

Table 3: **Ablation AAMC vs Cross-Entropy classifier.** This table clearly presents the benefit of our AAMC classifiers, which largely outperform standard linear classifiers trained with a cross-entropy loss.

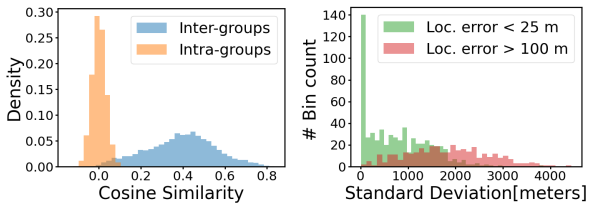

Figure 7: **Coordination of prototypes across groups.** (a) The left plot samples 500 neighboring prototypes (across all 4 groups), and shows their inter- and intra-group cosine similarity. It shows high correlation among inter-groups neighboring prototypes. Prototypes within a single classifier (intra-group) are well separated. (b) In the right plot we study the standard deviation (STD) among the prediction of each classifier. We can see that when the $N^2 = 4$ predictions are close to each other, the localization error is likely to be low (< 25 meters), proving that the STD between D&C's predictions from each expert is a good confidence measure, which is a very important value in real-world applications.

tion problem, that not only asks for classes separation (like in the vanilla CE), but also pushes them further away up to a margin. This results in a better-structured feature space, as shown in the Supplementary where we analyze the t-SNE of the learnt embeddings.

Behavior of multiple classifiers. Our method employs distinct AAMC classifiers, learned on disjoint sets, and merges their prediction to obtain the final logits. Seeing that prototypes of adjacent classes are learnt by different classifiers, it raises the question of "if and how they are related". A desirable property would be that prototypes become geographically correlated, on account of the fact that we want the different classifiers predictions to be consistent with one another. In Fig. 7 we study this aspect. In particular, the leftmost image samples 500 neighboring prototypes evenly distributed across groups, and plots the inter- and intra-group similarity distributions. The evidence indicates that the desired behavior is verified in practice. The explanation for this self-emerging property is the same that motivates our use of non-adjacent class partitioning: images from neighboring cells share many visual features, which would confuse a single classifier trying to discern among them. Instead, in our partition, prototypes assigned to neighboring cells can easily fit their distribution, given that they are learnt independently from one another in their respective groups. At the same time, since these cells are similar, their prototypes end up being similar as well; and in the end the result is that prototypes are geographically correlated providing more robust predictions.

Furthermore, having several classifiers predicting on disjoint sets also raises the question of what is their behaviour on samples that technically do not belong to any of their classes. In Fig. 7 (right) we examine this aspect by studying the distribution of the Standard Deviation (STD) of the coordinates predicted by each classifier. Interestingly, the histogram clearly shows that when our system is able to precisely localize a sample (<25m), all the classifiers concentrate their probability mass around the same area. On the contrary, on wrongly classified samples, the distribution reflects the uncertainty of the prediction. This indicates that classifier's agreement represents a good proxy of prediction reliability, which is an important feature in real-world applications.

4.4. Limitations

Divide&Classify is tailored for applications that aim at localizing in relatively large areas (e.g. SF-XL covers a surface of 170 km^2) mapped with a dense training set. We find that with smaller and less dense datasets like Pitts30k [43] and Tokyo 24/7 [42] (both are smaller than 3 km^2 and their density is less than half w.r.t. SF-XL) retrieval methods are able to achieve very high recalls (over 80% of recall@1 [13, 3], whereas classification methods fail to achieve acceptable results (LR@1 lower than 50%, regardless of the method). This can be explained by the fact that for a given query, a single positive matching image in the database is enough for retrieval (the model matches the most similar image to the query), whereas classification methods need a larger number of "positives", *i.e.* an inadequate number of images for any given class leads to poor performances. Thorough empirical experiments confirming this limitation of classification methods are shown in the Supplementary.

5. Conclusions

In this work we show the potential of framing the fine-grained VPR task in urban environments as a classification problem. We are the first to address this challenging scenario, proving that it is possible to achieve fine grained localization while obtaining a reliable measure of confidence in the predictions. We propose a novel inference pipeline to leverage the collective knowledge of a set of learnt classifiers that outperforms all the existing classification-based methods for localization. Finally, we show how our proposed framework can be combined with retrieval methods obtaining an ideal trade-off between inference cost and localization performance, paving the way for faster and more accurate VPR systems.

References

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

[2] Artem Babenko and Victor S. Lempitsky. The inverted multi-index. In *CVPR*, pages 3069–3076. IEEE Computer Society, 2012.

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

[4] Gabriele Berton, Riccardo Mereu, Gabriele Trivigno, Carlo Masone, Gabriela Csurka, Torsten Sattler, and Barbara Caputo. Deep visual geo-localization benchmark. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022.

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

[6] Z. Chen, A. Jacobson, N. Sünderhauf, B. Upcroft, L. Liu, C. Shen, I. Reid, and M. Milford. Deep learning features at scale for visual place recognition. In *2017 IEEE International Conference on Robotics and Automation*, pages 3223–3230, 2017.

[7] Z. Chen, L. Liu, I. Sa, Z. Ge, and M. Chli. Learning context flexible attention model for long-term visual place recognition. *IEEE Robotics and Automation Letters*, 3(4):4015–4022, 2018.

[8] Z. Chen, F. Maffra, I. Sa, and M. Chli. Only look once, mining distinctive landmarks from ConvNet for visual place recognition. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 9–16, 2017.

[9] Gabriela Csurka, Christopher Dance, Lixin Fan, Jutta Willamowski, and Cédric Bray. Visual categorization with bags of keypoints. In *European Conference on Computer Vision*, volume Vol. 1, 01 2004.

[10] Jiankang Deng, J. Guo, and S. Zafeiriou. ArcFace: Additive angular margin loss for deep face recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4685–4694, 2019.

[11] A.-D. Doan, Y. Latif, T.-J. Chin, Y. Liu, T.-T. Do, and I. Reid. Scalable place recognition under appearance change for autonomous driving. In *IEEE International Conference on Computer Vision*, pages 9319–9328, October 2019.

[12] S. Garg, N. Suenderhauf, and M. Milford. Semantic–geometric visual place recognition: a new perspective for reconciling opposing views. *The International Journal of Robotics Research*, 2019.

[13] Yixiao Ge, Haibo Wang, Feng Zhu, Rui Zhao, and Hongsheng Li. Self-supervising fine-grained region similarities for large-scale image localization. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 369–386, Cham, 2020. Springer International Publishing.

[14] Petr Gronát, Guillaume Obozinski, Josef Sivic, and Tomá Pajdla. Learning and calibrating per-location classifiers for visual place recognition. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 907–914, 2013.

[15] Stephen Hausler, Sourav Garg, Ming Xu, Michael Milford, and Tobias Fischer. Patch-netvlad: Multi-scale fusion of locally-global descriptors for place recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 14141–14152, 2021.

[16] S. Hausler, A. Jacobson, and M. Milford. Multi-process fusion: Visual place recognition using multiple image processing methods. *IEEE Robotics and Automation Letters*, 4(2):1924–1931, 2019.

[17] James Hays and Alexei A. Efros. im2gps: estimating geographic information from a single image. In *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[18] Sarah Ibrahimi, Nanne van Noord, Tim Alpherts, and Marcel Worring. Inside out visual place recognition. In *British Machine Vision Conference*, 2021.

[19] Mike Izbicki, Evangelos Papalexakis, and Vassilis Tsotras. Exploiting the earth's spherical geometry to geolocate images. In Ulf Brefeld, Élisa Fromont, Andreas Hotho, Arno J. Knobbe, Marloes H. Maathuis, and Céline Robardet, editors, *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2019, Würzburg, Germany, September 16-20, 2019, Proceedings, Part II*, volume 11907 of *Lecture Notes in Computer Science*, pages 3–19. Springer, 2019.

[20] H. Jégou and Andrew Zisserman. Triangulation embedding and democratic aggregation for image search. *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3310–3317, 2014.

[21] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547, 2019.

[22] Hervé Jégou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(1):117–128, 2011.

[23] Hervé Jégou, Matthijs Douze, Jorge Sánchez, Patrick Perez, and Cordelia Schmid. Aggregating local image descriptors into compact codes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34, 12 2011.

[24] A. Khaliq, S. Ehsan, Z. Chen, M. Milford, and K. McDonald-Maier. A holistic visual place recognition approach using lightweight CNNs for significant viewpoint and appearance changes. *IEEE Transactions on Robotics*, 36(2):561–569, 2020.

[25] Hyo Jin Kim, Enrique Dunn, and Jan-Michael Frahm. Learned contextual feature reweighting for image geo-localization. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3251–3260, 2017.

[26] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 12 2014.

[27] Giorgos Kordopatis-Zilos, Panagiotis Galopoulos, S. Papadopoulos, and Y. Kompatsiaris. Leveraging efficientnet and contrastive learning for accurate global-scale location estimation. *ACM International Conference on Multimedia Retrieval*, 2021.

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

[29] Yu A. Malkov and D. A. Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42:824–836, 2020.

[30] Carlo Masone and Barbara Caputo. A survey on deep visual place recognition. *IEEE Access*, 9:19516–19547, 2021.

[31] Riccardo Mereu, Gabriele Trivigno, Gabriele Berton, Carlo Masone, and Barbara Caputo. Learning sequential descriptors for sequence-based visual place recognition. *IEEE Robotics and Automation Letters*, 7(4):10383–10390, 2022.

[32] Jong Hak Moon, Wonjae Kim, and E. Choi. Correlation between alignment-uniformity and performance of dense contrastive representations. In *British Machine Vision Conference*, 2022.

[33] 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.

[34] Florent Perronnin, Yan Liu, Jorge Sánchez, and Herve Poirier. Large-scale image retrieval with compressed fisher vectors. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3384–3391, 06 2010.

[35] Noé Pion, Martin Humenberger, Gabriela Csurka, Yohann Cabon, and Torsten Sattler. Benchmarking image retrieval for visual localization. In *2020 International Conference on 3D Vision (3DV)*, pages 483–494, 2020.

[36] F. Radenović, G. Tolias, and O. Chum. Fine-tuning CNN Image Retrieval with No Human Annotation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

[37] A. Razavian, J. Sullivan, A. Maki, and S. Carlsson. Visual Instance Retrieval with Deep Convolutional Networks. *CoRR*, abs/1412.6574, 2015.

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

[39] Josef Sivic and Andrew Zisserman. Video google: A text retrieval approach to object matching in videos. In *ICCV*, pages 1470–1477. IEEE Computer Society, 2003.

[40] Yash Srivastava, Vaishnav Murali, and Shiv Ram Dubey. A performance comparison of loss functions for deep face recognition, 2019.

[41] Giorgos Tolias, R. Sicre, and H. Jégou. Particular object retrieval with integral max-pooling of CNN activations. *CoRR*, abs/1511.05879, 2016.

[42] A. Torii, R. Arandjelović, J. Sivic, M. Okutomi, and T. Pajdla. 24/7 place recognition by view synthesis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(2):257–271, 2018.

[43] A. Torii, J. Sivic, M. Okutomi, and T. Pajdla. Visual place recognition with repetitive structures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(11):2346–2359, 2015.

[44] A. Torii, Hajime Taira, Josef Sivic, M. Pollefeys, M. Okutomi, T. Pajdla, and Torsten Sattler. Are large-scale 3d models really necessary for accurate visual localization? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43:814–829, 2021.

[45] Nam Vo, Nathan Jacobs, and James Hays. Revisiting im2gps in the deep learning era. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2640–2649, 2017.

[46] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, and Wei Liu. Cosface: Large margin cosine loss for deep face recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 5265–5274. Computer Vision Foundation / IEEE Computer Society, 2018.

[47] Ruotong Wang, Yanqing Shen, Weiliang Zuo, Sanping Zhou, and Nanning Zheng. Transvpr: Transformer-based place recognition with multi-level attention aggregation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13648–13657, June 2022.

[48] Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. *Advances in Neural Information Processing System*, 2020.

[49] Frederik Warburg, Soren Hauberg, Manuel Lopez-Antequera, Pau Gargallo, Yubin Kuang, and Javier Civera. Mapillary street-level sequences: A dataset for lifelong place recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, June 2020.

[50] Tobias Weyand, Ilya Kostrikov, and James Philbin. Planet - photo geolocation with convolutional neural networks. In *European Conference on Computer Vision*, 2016.

[51] Mubariz Zaffar, Shoaib Ehsan, Michael Milford, and K. Mcdonald-Maier. Cohog: A light-weight, compute-efficient,

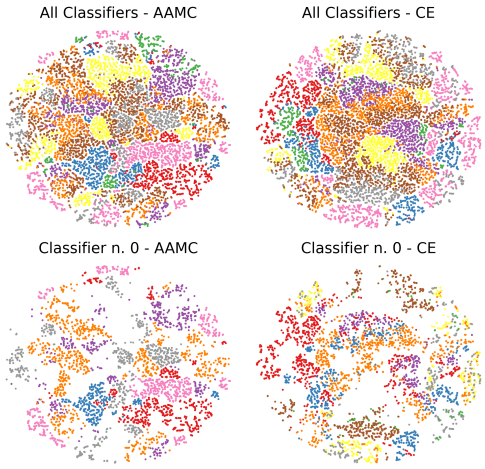

Figure 8: **t-SNE analysis** of embeddings in a 100m x 100m square. Each color codifies a different 20m cell.

and training-free visual place recognition technique for changing environments. *IEEE Robotics and Automation Letters*, 5:1835–1842, 2020.

[52] Mubariz Zaffar, Sourav Garg, Michael Milford, Julian Kooij, David Flynn, Klaus McDonald-Maier, and Shoaib Ehsan. VPR-Bench: An open-source visual place recognition evaluation framework with quantifiable viewpoint and appearance change. *International Journal of Computer Vision*, 129(7):2136–2174, 2021.

A. Experiments

In this Supplementary Material we report details that could not fit in the main paper. In Appendix A.1 we provide further ablations to better understand how our proposed method functions.

In Appendix B we provide a thorough discussion into how we adapted the partitioning scheme of previous works, that originally targeted planet-scale localization, for the proposed task of city-wide localization.

A.1. Further Ablations

Embedding learnt with our AMCC vs Cross Entropy. The first row in Fig. 8 reports the t-SNE of all embeddings in a 100m square, with a model trained either with AAMC or a fully connected layer with cross-entropy loss; each color codifies a different 20m cell. Even though some structures are visible, there is an amount of overlap which is understandable given that adjacent cells at such fine resolution can present high appearance similarities. The second row shows why in D&C each classifier is able to learn a meaningful distribution: inside each group, thanks to the non-adjacency of cells, classes are well defined. In particular, the two plots show how the AAMC yields a better-structured embedding space thanks to the concept of large margin.

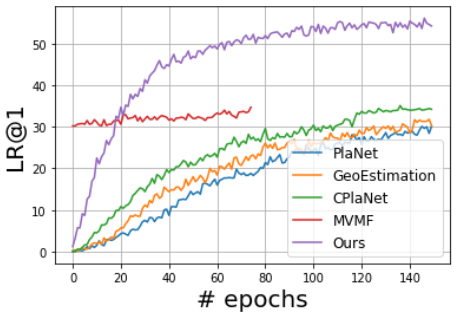

Figure 9: **Behaviour of LR@1 during training** for each of the methods. Note that MVMF [27] starts with a high LR because it uses the weights of a trained PlaNet model.

Behaviour of LR during training with different methods. In Fig. 9 we analyze how the LR@1 changes after each epoch for different methods (given the huge size of the dataset we define an epoch as 2k iterations). We find that most previous works, namely PlaNet [50], CPlaNet [38] and Hierarchical Geolocation Estimation [33] present very mild improvements on LR within the first few epochs w.r.t. our D&C, which on the other hand grows very steeply right from the beginning. MvMF initializes its mixture assignment weights from a pretrained PlaNet model, and it terminates the training after less than 100 epochs.

Behaviour of classification accuracy during training using different N. To better understand how the value of N affects training stability, we built a plot using $N = 2$ and $N = 3$ and showing the accuracy on the train set at the end of each epoch. The plot (Fig. 10) shows that in the first epochs of training the accuracy forms waves with a period length of size $|G| = N \times N$, where $|G|$ represents the number of groups and the number of classifiers. This is due to the fact that each classifier is trained once every $|G|$ epochs, meaning that at the $|G|th$ epoch the model will for the first time reuse a classifier that has been previously trained, resulting in a steep increase in accuracy every $|G|$ epochs.

Qualitative results. In Figs. 12 and 13 we show some qualitative results of challenging queries and the retrieved Top-3 candidates by some retrieval-only methods (namely CosPlace and NetVLAD) and by some classification-retrieval pipelines (using respectively our D&C and CPlanet as classification modules).

Approximate Nearest Neighbor Search. In Fig. 11 we report the results with the best combinations of methods / hyperparameters for our experiments with Approximate Nearest Neighbor search algorithms. The plot shows only the best performing configurations. Among other ANNs that we tried are standard Product Quantization [22], Inverted

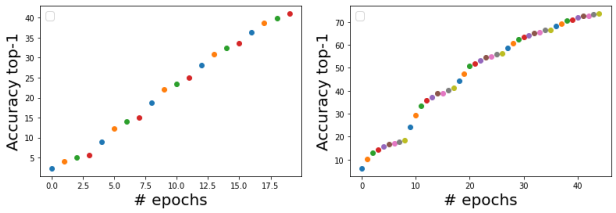

Figure 10: **Evolution of classification accuracy during training with different values of N.** We can see that in the first epochs of training, the accuracy on the train set presents waves with period length of size $|G|$. Each color represents a different classifier being trained at the given epoch for a total of $|G|$ colors.

File Indexes (these two methods can be combined in the IVFPQ), and Inverted File MultiIndex [2]. We didn't report these results as they performed poorly w.r.t. their counterparts in the plot.
For Table 1 of the main paper we chose two configuration from this pareto-optimal curve, one being optimized for performances and one for speed. For performances, we picked the configuration that grants at least 10x speed, with the maximum performances, and this turned out to be IVFPQ(128,50). For speed, we selected the methods that provided a speedup of at least 100x. This resulted in choosing IVFPQ(128,2) and HNSW(512).

A.2. Experiments on small datasets

In the main paper we discussed how classification methods are outperformed by retrieval approaches for small datasets due to the lack of enough positives during training. On the other hand, the inference time gap between both procedures loses relevance when dealing with smaller datasets. In Tab. 4 it is presented a quantitative analysis on how the proposed methods behave on datasets that are 1000x smaller than SF-XL, covering geographical areas less than $3km^2$ and having half the density of SF-XL.

B. Baselines Implementation Details

Although previous works use different partitioning methods of the dataset in classes, we carefully tuned the partitioning hyperparameters to ensure fair comparisons among different methods. While some methods split the geographical area according to the density of the training points [50, 33] others fix the dimension of the cells into a predefined value and merge them until the number of geographical regions satisfies the desired condition [38].

The optimal number of classes generated with each method is shown in Tab. 5, and in the next paragraphs we detail how we empirically found such values for each partitioning method.

Method	C-Pitts30k (30k images)		C-Tokyo 24/7 (76k images)	
	LR@1	Inf. time	LR@1	Inf. time
Classification				
PlaNet [38]	31.5	12 ms	19.5	12 ms
HGE [33]	33.6	15 ms	22.0	15 ms
CPlaNet [38]	33.0	17 ms	21.5	17 ms
MvMF [19]	31.5	12 ms	19.9	12 ms
D&C (ours)	40.5	12 ms	33.7	12 ms
Retrieval		(kNN time)		(kNN time)
NetVLAD [1]	86.1	58 ms	62.2	130 ms
CRN [25]	86.3	58 ms	62.8	130 ms
SARE [28]	87.2	58 ms	74.8	130 ms
SFRS [13]	88.7	58 ms	78.5	130 ms
GeM [36]	77.9	16 ms	46.4	25 ms
CosPlace	88.5	16 ms	82.8	25 ms
Mixed pipeline				
D&C(ours) + CosPlace	81.9	1 ms	74.9	3.5 ms

Table 4: **Comparison of LR@1** of different methods for *Pitts30k* and *Tokyo24/7* using *EfficientNet-B0* as backbone

Partitions of HGE, PlaNet and MVMF. The three methods of PlaNet [50], Hierarchical Geolocation Estimation (HGE) [33] and MVMF [19] all use the same partitions, with the only difference that HGE also uses two coarser splits (*medium* and *coarse*) besides the regular partition (*fine*) used by the other two. The partitions are built using Google S2 Sphere library, and take as input two parameters, namely τ_{min} and τ_{max}, which define the minimum and maximum number of images within each cell. We empirically search for the best values for the parameters on the San Francisco eXtra Large (SF-XL) dataset, and we report the results in Tab. 6.

We choose the partitions that lead to the best LR@1 using HGE, and, following their implementation, we use the finer HGE partition also as training set for PlaNet and MVMF. In practice, this leads to a value of $\tau_{min} = 100$ and $\tau_{max} = 2500$, as shown in Tab. 7 (where we also report the value of τ for other partitions. Note that we use proportions between different partitions size according to [33].

We tuned cells density on SF-XL since it is the most representative dataset for the studied setting. Remembering that these partitioning schemes are based on keeping cell density constant, to extend the comparison to the other adopted datasets (C-Pitts30k, C-Tokyo24/7), we scaled τ_{min} and τ_{max} according to the relative density of the other datasets with respect to SF-XL. In our method, instead, the partitioning only depends on the desired granularity of localization, so we kept the same $20m$ cells across all datasets.

Partitions of CPlaNet.

Regarding CPlaNet's [38] partitions, we carefully followed the authors' implementation: we created five *geoclass sets* for each of the experiments, where *geoclass set*$_1$ and *geoclass set*$_2$ evaluate the proximity distance using only the geographical and visual properties of the images respec-

Partition method	SF-XL	C-Pitts30k	C-Tokyo 24/7
PlaNet / MVMF	65k	486	1840
HGE	19k / 35k / 65k	158 / 272 / 486	508 / 961 / 1840
CPlaNet	54k	369	1236
Ours	114k	687	2492

Table 5: **Number of classes in different datasets** using different partitioning methods.

HGE Num. Classes			
coarse	medium	fine	LR@1
65.3k	119k	200k	19.0
35.0k	65.3k	119.0k	21.2
18.5k	35.0k	65.3k	27.0
9.4k	18.5k	35.0k	25.3
3.8k	9.4k	18.5k	19.2
1.8k	3.8k	9.4k	10.6

Table 6: **Results with different partitions** using HGE on SF-XL.

hyperparams	fine	HGE-medium	HGE-coarse
τ_{min}	100	100	100
τ_{max}	2500	5000	10000

Table 7: **Chosen hyperparameters** for previous methods partitioning. Note that Planet, HGE-fine and MvMF use the same partitioning.

Cells per geoclass						
# classes	gcs 1	gcs 2	gcs 3	gcs 4	gcs 5	LR@1
58233	30k	30k	39k	36k	33k	27.6
54144	20k	20k	26k	24k	22k	27.7
47412	10k	10k	13k	12k	11k	25.7

Table 8: **CPlaNet preliminary results** on SF-XL.

tively, while the remaining *geoclass sets* were generated by considering the distance as a stochastic linear combination of these two modalities. We refer the reader to their paper for more details about how each *geoclass set* is formed. In their method, an additional hyperparameter is the number of classes in each *geoclass set* (*i.e.* their partition algorithm stopping condition). Finally, at inference time, the granularity considered for prediction is given by the intersections of the 5 *geoclass sets*. In Tab. 8 we report results using different values for each and using the same parameters α and β, which define the differences between the 5 *geoclass sets*. The table also reports in the first column the number of distinct cells obtained by the intersection of the different partitions. Also in this case we choose from the table the split which gave the best results for LR@1.

To export these hyperparameters to the other datasets, we kept the same average size of the cells in each *geoclass set*.

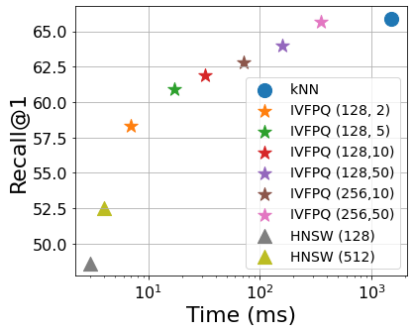

Figure 11: **Comparisons of best-performing Approximate Nearest Neighbor search algorithms.** We show only the pareto-optimal results, which are computed with an Inverted File Index with Product Quantization (IVFPQ) [22] and Hierarchical Navigable Small Worlds (HNSW) [29]. The parameters in parenthesis for IVFPQ indicate the number of subquantizers and the *nprobe*, *i.e.* the number of Voronoi cells to be searched (out of 1000). The parameters in parenthesis for HNSW indicates the number of connections each vertex has within the HNSW graph.

Figure 12: **Qualitative results** using different pipelines on challenging queries.

Figure 13: **Qualitative results** using different pipelines on challenging queries.