2015 IEEE Symposium Series on Computational Intelligence

Improving SVM Training Sample Selection Using Multi-Objective Evolutionary Algorithm and LSH

Romaric Pighetti, Denis Pallez, Frédéric Precioso
pighetti@i3s.unice.fr, denis.pallez@unice.fr, precioso@i3s.unice.fr, frederic.precioso@unice.fr
University Nice Sophia-Antipolis, I3S Lab (UMR CNRS 7271)

***Abstract*—In this paper, we propose a new framework hybridizing a Support Vector Machine (SVM), a Multi-Objective Genetic Algorithm (MOGA) and a Locality Sensitive Hashing (LSH). The goal is to tackle fine-grained classification challenges which means classifying many classes with high similarities between classes and poor similarities inside one class. SVM is used for its ability to learn multi-class problems from very few training data. MOGA is used to optimize training samples used by the SVM so as to improve its learning rate. As data define a discrete set of instances and not a continuous solution space, LSH is used to map "optimal solutions" obtained by MOGA onto the closest real instances contained in the dataset. We evaluate our method for content-based image classification on the standard image database Caltech256 (i.e. 30000 images distributed in 256 classes). Experiments show that our method outperforms state-of-the-art approaches.**

I. Introduction

One of the big challenges encountered recently in classification is the so-called "fine-grained challenge". This not only refers to the task of classifying large scale datasets, which is still an open question for huge volume of data, but it has mainly to deal with considering many classes among which some are very similar with respect to the description of their content while the semantic information they hold is different and vice-versa. This problem arise often when the amount of data and the number of classes increase. It has been exhibited in particular in the context of classifying large image databases (see Fig.10 for inter-class visual ambiguities and see Fig.11 for intra-class visual dissimilarities). In such a context, to learn a classifier the larger the training set is the better the prediction will be. However, considering the amount of data the computation time required would be extremely long.

One way to deal with very large datasets and proposing low time consuming algorithms is to consider subsets of the training set during the learning process. The issue is then how to build these subsets: Most of the works are based on random subsets [1].

While this effectively solves the large scale problem, it does not guarantee the best results. In opposition to random strategies, a statistical and a spatial distribution analysis of the data may be used for building better subsets. In this case, computing geometrical information on the whole dataset is still very time consuming and is hard to apply for large datasets [2], [3]. In this paper, we propose to combine a genetic algorithm (GA) which is based on stochastic exploration mechanisms while guiding the search towards promising areas of the search space, and Support Vector Machine classifiers (SVM) [4] to exploit locally these promising areas to build a better global classifier. SVM have proven to be very effective in a wide range of problems, thus building a solution for the fine-grained challenge on SVM is pretty promising, even though training a SVM is a quadratic problem with respect to the number of training samples. In the proposed approach, the GA will search for the optimal training samples to improve the SVM classification. The training set for the SVM will incrementally grow with training samples retrieved by the GA. At each iteration the SVM classifier, trained up to the current step, is then used to evaluate new generated solutions and thus contribute to guide the GA search process.

This incremental building process of the training set requires the GA solutions to be samples of the dataset. In [5], Kawulok et al have also worked on a method using a GA to extract learning samples for SVM. However in their approach, each solution evolved by the GA is a training set so that at the end of the optimization process, the best training set is obtained and ensures better performances for the classifier. Such strategy implies to learn a classifier for each training set contained in the GA population, which is very time consuming. As opposed to the previous strategy, we design a solution to be a single point of the dataset. Thus, the proposed approach has a lower complexity.

In our approach, as the genotype is only one solution and considering that GA may apply crossover and mutation during evolution, resulting optimal solutions may not belong to the initial dataset. So, we propose to use Locality Sensitive Hashing (LSH) to facilitate the identification of an adapted solution picked from the initial dataset.

We have focused the evaluation of our apporach on the fine-grained context of large scale image classification. Several other works have been considered to combine evolutionary algorithms and local search techniques (SVM, k-Nearest Neighbor classifier...) for image classification [6]–[11]. However, most of these approaches are evaluated on very small image datasets (no more than few thousand images) or ad-hoc handcrafted databases up to 12 000 images [11], mainly owing to scalabilty limitations. We have chosen to evaluate our method on Caltech-256 dataset (30 000 images in 256 classes) which is a well-known benchmarck for fine-grained image context, classically considered in most of the recent works in image classification.

The paper is organized as follows, the next section gives some background about SVM, GA and LSH, which are the three main components of the presented framework. Section III presents the details and explanations about the proposed ap-

978-1-4799-7560-0/15 $31.00 © 2015 IEEE
DOI 10.1109/SSCI.2015.197

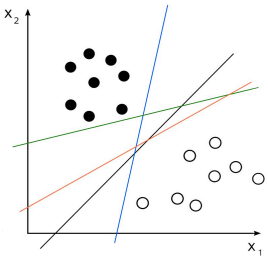

Figure 1. Possible separating planes

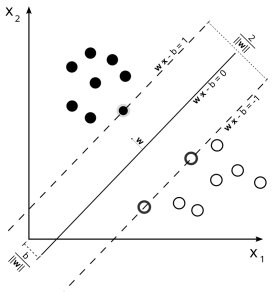

Figure 2. SVM margin illustration

proach. The experimental set-up and parameters are exposed in Section IV, and Section V presents the results. Finally, a conclusion and future works are presented in Section VI.

II. Preliminaries

The presented method combines SVM, multi-objective GA and LSH. In order to better understand our method, details about those three algorithms are given in this section.

A. Support Vector Machine

SVM has encountered a great success as a classification technique in the past twenty years for several good reasons: It has proven to be effective for a wide range of pattern recognition tasks, it is theoretically sound, and it has among other power the capacity to achieve rather impressive classification results from few training samples. While several hyperplane can separate a given set of training data (Fig.1), SVM computes the hyperplane which gives the largest margin (Fig.2). As it can be seen in Fig.2, the hyperplane is defined by a small number of vectors (those with bigger light gray and black border), called support vectors, computed from the training set. Once the hyperplane is defined, it is used to classify further unclassified data that are presented to the SVM.

A hyperplane separates the space into two parts, thus SVM are binary classifiers by design. When dealing with multi-class problems, the problem is broken down into several binary classification problems. Common strategies to then build the final multi-class decision are: learning a binary classifier between every pair of classes (one versus one strategy) or learning a binary classifier between a class and the remaining classes (one versus all strategy) for each class. In the first case, the class is assigned to a new data using the max voting wins strategy. In the second case, the classifier with the highest SVM decision (the new data is both on the positive side for this SVM and farthest from this SVM boundary than any other SVM) determines the class. However, the SVM decision (i.e. distance between the new data and the SVM boundary) has to be tuned to produce comparable results among all one-vs-all SVMs.

B. Genetic Algorithms

Genetic Algorithms (GA) are optimization methods inspired by Darwin's theory of evolution. GA evolves candidate solutions by applying successively genetic operators as selection, recombination (crossover) or mutation on candidate solutions. A candidate solution, also called individual, is often represented by a vector of genes called a genome. A gene can be a bit (0 or 1), an integer or a real value. In a multi-objective optimization process [12], a fitness function f is defined by $f(x) = (f_1(x), ..., f_n(x))$ where $f_i(x) : X \rightarrow \mathbb{R}$ represents one objective to optimize (maximize or minimize). In this context, a dominance relation is defined as given in Def.1 to compare individuals performances in solving the multi-objective function. The way candidate solutions $x \in X$ are selected for breeding is based on this fitness value. This process iterates for new generations until a stopping criteria is reached.

Definition 1: Let $\mathcal{D}$ be the space in which solutions are expressed, and $F = \{f_1, f_2, ..., f_n\}$ be n scalar objective (or fitness) functions to be minimized.
For $(s_1, s_2) \in \mathcal{D}$, s_1 dominates s_2 if and only if:
$\forall i \in [1, n]$, $f_i(s_1) \leq f_i(s_2)$ and
$\exists i \in [1, n]$, $f_i(s_1) < f_i(s_2)$

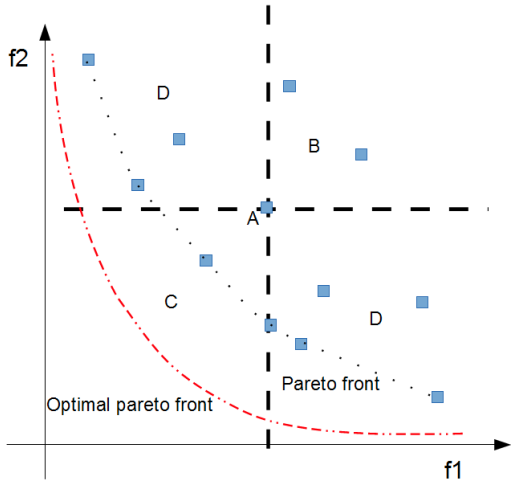

Figure 3. Solutions to a bi-objective minimisation problem

Fig.3 illustrates this dominance relation considering that the goal here is to minimize both f1 and f2. All individuals in the area marked B are dominated by the solution A, and all solutions in the area marked C dominate solution A. No comparison can be made between solution A and solutions in the area marked D using the dominance relation. Given this dominance relation, for a set of individuals, the pareto front is defined as the subset of non-dominated individuals, represented as a black dotted line in Fig.3. The optimal pareto front is the set of attainable non-dominated solutions, depicted with a red mixed dashed line in Fig.3. The goal of a multi-objective evolutionary algorithm is to evolve individuals toward the optimal pareto front, keeping as much diversity as possible along the front. The dominance relation is used as part of the evaluation process to guide the search toward this goal. One of the most known algorithm for MOGA is NSGA2 [13].

Those algorithms are known to be able to explore the search space thanks to the genetic operators, which is of great interest in the presented method.

C. Locality Sensitive Hashing

Locality Sensitive Hashing (LSH) [14] is a hashing technique aiming at giving a fast approximate solution to the k-Nearest Neighbours problem. It uses family of locality sensitive hashing functions to hash the data in small cells, commonly called buckets, see Fig.4. This partition of the space is relevant only if two data which were close before partitionning, are put in the same part of the partioned space, i.e. are in the same bucket. LSH classic algorithm [15]–[17]

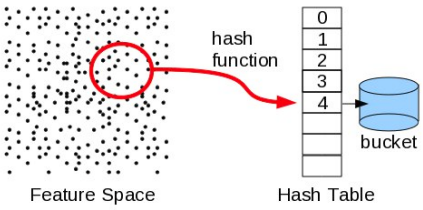

Figure 4. The input space $\mathcal{D}$ is partitioned thanks to hash functions and data are assigned to buckets.

tackles the problem $(R; cR)NN$. Indeed, LSH defines how to design a family $\mathcal{H}$ of hash functions h to be locality sensitive: Given a query Q the probability p_1, that a data A at a distance less than R from the query in the original space is put in the same bucket as the query, is higher than the probability p_2 that a data B at a distance higher than cR from the query, is put in the same bucket as the query (cf. Fig.5). This can be better formalized in Def.2.

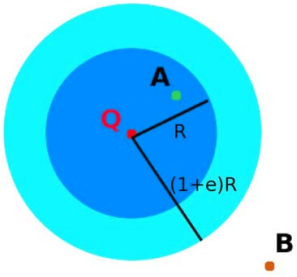

Figure 5. Q is the query and A its nearest neighbor within radius R while B is farther than cR from Q, with $c = 1 + \epsilon$ on the figure.

Definition 2: Let $\mathcal{D}$ be the space of data to be hashed, $(R, cR) \in \mathbb{R}^2$, $p_1 \in [0, 1]$ be the probability of true NN detection, $p_2 \in [0, 1]$ be the probability of false NN detection, $\mathcal{H}$ be a family of hashing functions and d be a similarity or distance function on $\mathcal{D}$.

$\mathcal{H}$ is said to be (R, cR, p_1, p_2)-sensitive if:
Given a query data $Q \in \mathcal{D}$,
$\forall h \in \mathcal{H}$, $\forall (A, B) \in \mathcal{D}^2$,
if $d(A, Q) \leq R$ then $P_{\mathcal{H}}[h(A) = h(Q)] \geq p_1$
if $d(B, Q) \geq cR$ then $P_{\mathcal{H}}[h(B) = h(Q)] \leq p_2$
with $p_2 \leq p_1$ and $R \leq cR$, $c > 1$.

The standard implementation of LSH provides a solution to this problem with a computation complexity in $O(n^{\rho})$ with $\rho < 1/c$ and n the dimension of $\mathcal{D}$ space.

Among all the families $\mathcal{H}$ proposed by the authors [17], we have chosen to use random projections (see Fig.6):

For any data $\mathbf{p} \in \mathcal{D}$ given,

$$h_{\mathbf{a},b}(p) = \lfloor \frac{\mathbf{a}.\mathbf{p} + b}{W} \rfloor$$

where $\mathbf{a}$ is a random vector in $\mathcal{D}$, b is a random value in $[0, W[$ and W specifies the size of the "slices". Fig.6 shows

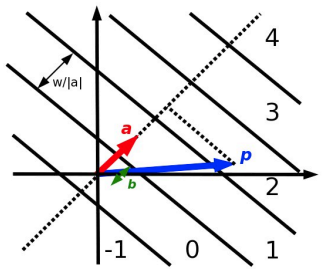

Figure 6. One hash function (random projection) for $\mathcal{D} = \mathbb{R}^2$

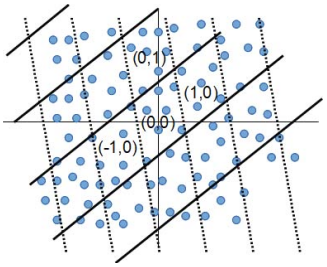

Figure 7. $\mathcal{D} = \mathbb{R}^2$ partitioned with one table of two hash functions

the graphical representation of one hash function defined as a random projection [17] and Fig.7 shows a graphical representation of hashing a two dimension space using one table with two hash functions. Parallel lines represent one hash function. In this figure, each hash function gives a result in $\mathbb{Z}$ and combined hash functions are couples of integers. If more than one table is used, then, for each new table, two new hash functions are drawn, splitting the space in other directions.

Given this structure, searching for the approximate nearest neighbours of a data d consists in searching the true nearest neighbour of d in the set of data that have the same combined hash function as d in at least one table. Fig.7 helps observing that the more hash functions are used per table, the smaller the retrieved set will be. More hash functions means more "lines", and thus smaller buckets. And, the more tables are used, the bigger the set will be. More tables means multiplying the space partition displayed on Fig.7 as many times as the number of tables with new hash functions. Then retrieving from each table the set of data with the same combined hash as d multiplies the number of sources from which data are retrieved, and so increases the number of elements in the set.

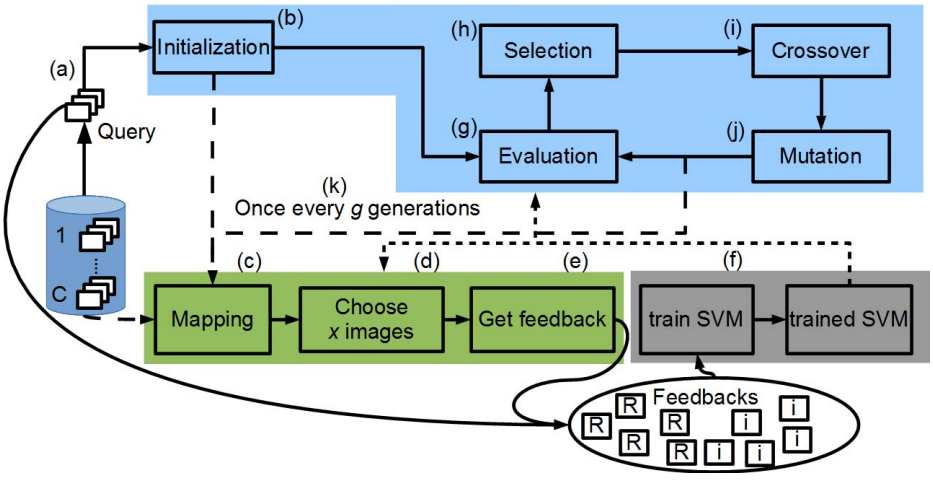

Figure 8. The framework workflow

Careful attention must be taken when choosing the number of tables and hash functions. If those are not tuned correctly, one might have difficulties to find neighbours in some part of the search space, because there are too few elements retrieved. Or spend too much time computing the true nearest neighbour because there are too much elements in the retrieved set of approximate nearest neighbours.

III. The Framework

The proposed approach aims at using a genetic algorithm to extract optimal training samples for a SVM from a very large set of possible training samples. As other sampling methods, we aim at reducing the size of the training set to improve the SVM training time, which is quadratic with respect to the training set size. As aforementioned, the SVM decision is built upon only few support vectors from the training samples. The exploration capabilities of GA will help reaching the right training samples and the framework will be able to get better performances than random sampling or local search strategies which concentrate only on training samples close to or within the SVM margin.

In our approach, the SVM training set is built iteratively by extracting training samples from the GA exploration process. Lets first introduce the global workflow before going into more details about the GA exploration.

A. Global Workflow

Fig.8 depicts the global workflow of the framework. In order to initialize the learning process, one example of training data must be provided for each category searched (Fig.8(a)). This example plays the role of query. From there, one genetic algorithm is initialized for each category (Fig.8(b)). The initialization is partially biased, using a normal law centred on the example provided to the algorithm for each category.

Let GA_i be the i^{th} GA, associated to C_i, the i^{th} searched category, and P_i its population.

The SVM decision is used as part of the evaluation process to guide the search towards interesting areas of the input data space. Therefore, it is required to learn a first SVM before the GA process can further start. To do so, once the initialization is done, a first extraction of training samples is processed (green part of Fig.8) to initialize the classifier. For each GA, each individual of its population P_i is mapped to the nearest available true training sample from the training dataset (Fig.8(c)). The mapping is applied on data that have not been yet added to the training set. This is done using an approximate nearest neighbour search using LSH. Some of the new training samples are then extracted from each mapped population, and added to the classifier training set (Fig.8(d and e)). Once it is done, a classifier is available and ready to classify unlabelled data (Fig.8(f)).

The evolutionary process can then begin. The evolution loop, consisting of evaluation, selection, crossover and mutation (Fig.8(g, h, i and j)) is processed. The classifier trained so far is used as part of the evaluation process to help guiding the search towards interesting new training samples. Every g generations (Fig.8(k)) the mapping and the extraction process (green part of Fig.8) are repeated to add samples to the classifier training set. The process stops when enough samples have been added to the training set.

B. The Genetic Algorithm

The goal of the genetic algorithm in this framework is to retrieve training samples. Therefore, each individual of the GA represents a potential training sample (defined as a real valued vector representing the image description in our experiments).

The evaluation (Fig.8(g)) is the core of the process. It defines the problem the GA must solve and thus guides the search. The GA is expected to retrieve training samples that improve the SVM decision. As we want to identify rapidly good results, one objective will be to identify the worst and best individuals to determine the center of the class. Another concern is to find a diverse set of elements and make the GA explore the space.

So as to satisfy these two requirements, two objectives have been defined. The first one is the closeness to the SVM margin, which is to be maximized to identify rapidly the center of the

class. As we are working with a multi-class SVM based on the 1 versus all strategy and we have one GA per category, each GA will evaluate the distance of its individuals to the margin of the SVM separating the category associated to the GA and the rest.

The second objective is the closeness of the individual to the query element, which is to be maximized. Since the query element is probably in the center of the class, we expect the GA to find elements in other palces of the search space by doing this.

Those two objectives are quite different. They most probably have their solutions in different regions of the search space. In order to solve those problems together, we have decided to use a multi-objective GA. As explained before, those are designed to find a diverse set of solutions, giving multiple non-dominated trade-off between objectives. This property will also ensure that the solutions identified by the MOGA of a given category are diverse enough and allow the extraction of several training samples from its population whenever we need to.

Experiments have been conducted with this framework to evaluate its performances against random sampling and classical interactive SVM. Next section brings further details about experimental set-up and the implementation used.

IV. Framework evaluation

This section first introduces content based image retrieval (CBIR) and its challenges as the test bed problem for the proposed method. Details about the dataset used and the experimental set-up are then given before some details about the implementation and used parameters are provided.

A. Content Base Image Retrieval

Within the wide variety of problems that can be seen as a classification problem, we have considered here CBIR as the test bed for the presented framework. It consists in searching images by their visual content instead of keywords or meta-data. The goal of such techniques is to be able to classify huge image databases, gathering into a same class or group, images that represent the same semantic concept. This field of research has grabbed most of the attention from computer vision and machine learning communities in the last decade, driven by the increasing growth of visual content downloaded on the internet everyday, which makes impossible to attach relevant keywords or meta-data to every single data.

The number of data makes CBIR a large scale problem by definition. It can also be considered as a fine grained problem due to the number of classes to be retrieved and the characteristics of images, which can have inter-class ambiguous similarities (as illustrated in Fig.10) and intra-class ambiguous dissimilarities (as illustrated in Fig.11). In addition, CBIR is a really active research topic, which results are applied from medical use [5] to everyday search queries [18]. In addition, state of the art techniques in the domain rely on SVM, allowing straightforward comparison of the proposed framework against classical SVM use.

CBIR systems are composed of two main parts. The first is the expression of the visual content of the images. It consists in extracting a visual description for each image of the database. The second, on which the focus is in this work, is the mining of those visual description to classify the database.

Thanks to the popularity of CBIR, there exists many datasets and competitions to which one can compare their results to.

B. Experimental Setup

Figure 9. Some categories from Caltech-256 database

Caltech-256 [19] database has been chosen as the test bed for the presented work. It is composed of about 30 000 images in 256 categories. Fig.9 shows some images from this database. It has been used in several works whom results are reported in [18]. Those works will form the baselines for results comparisons. The problem associated with Caltech-256 dataset is to be able to learn the 256 categories of the database by training a classifier from a small subset, the so-called training set, in order to assign the right class to the remaining data, the so-called test set, which have not been seen by the classifier.

As no separation is pre-defined for training and test sets in caltech-256, the standard evaluation protocol is based on randomly drawing training samples from the database, and use the remaining of the database as a test set. In this protocol, methods have to be evaluated for different training set sizes: 15, 30, 45 and 60 training samples per category are drawn respectively.

We want to point out here that following such protocol is rather unfair for our approach since we have to constrain

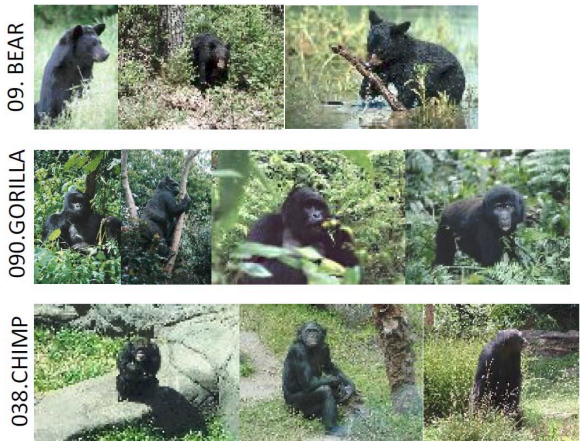

Figure 10. Inter-class ambiguities for categories: BEAR, GORILLA and CHIMP

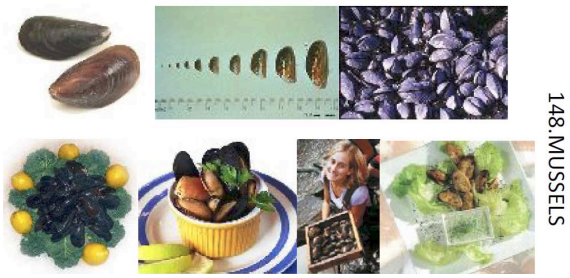

Figure 11. Intra-class ambiguities for the category: MUSSELS

the exploration potential of the GA to be sure to extract precisely 15, 30, 45 and 60 training samples per category, to remain compliant with the evaluation protocol. However, the evaluation on Caltech-256 classification was the most simple context for fine-grained challenge.

As in all the works evaluated on this same protocol, the training samples are drawn from the whole dataset, we have decided that the whole dataset will be considered as available training samples for our method. Doing so, training samples are drawn from the same pool of data as previous works, what changes is how the training samples are selected. To use LSH for the approximate nearest neighbour search in the framework, the database requires to be hashed before any learning process. In order to fully respect the standard evaluation protocol, we can select a new training sample for a given training class subset only if this subset has not yet reached the training size limit (15 or 30 or 45 or 60). Each class is thus hashed separately so that we can control the mapping to a specific class and so the amount of each class integrated in the training set.

The learning process runs and 15 training samples are added to the classifier training set at each mapping and selection processes (green part of Fig.8). The learning process is stopped when 60 training samples have been selected from each category. The SVM classification score is evaluated each time new elements are added to the training set, which ends up providing results for training sets containing 15, 30, 45 and 60 images per category. As for [18], the remaining of the database is then used as a test set. This allows a fair evaluation of what the framework brings in comparison to random sampling.

For comparison purposes, an active SVM approach [20] has also been implemented. It uses the same SVM code as our framework implementation. The learning process consists in selecting the 15 first learning samples at random for each category. Then, in each category, the 15 most uncertain data are chosen to be added to the training set. The process is repeated until enough (60 in this study) elements have been added to the training set for each category. As for the previous method, the SVM classification score is evaluated each time new elements are added to its training set.

In order to observe the impact of our training sample selection against standard methods, the same visual descriptors, i.e. Fisher Vectors, as [18] have been used in the experiments.

As the framework embeds a GA, which uses randomness during the evolution, and because the query elements are chosen at random, the learning process is repeated 10 times and the results presented are the average results over the 10 runs. For each run, the image given for each category as a query is randomly drawn from the database.

The most commonly considered measures to evaluate CBIR system are Precision and Recall. Results are compared to those of [18], therefore, the same measure is used: the Mean Average Precision (MAP). This measure consists in computing the mean of the retrieval precision obtained for each separate class with a given classifier. (1) explains how to compute the MAP.

$$MAP = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{|\text{num(correctly classified in } C_i|)}{|\text{num(classified in } C_i|)}\right) \quad (1)$$

This ends the experimental set-up. Parameters values and details about the implementation are provided in the following.

C. Implementation Details and Parameters Values

The implementation has been realized in java, using ECJ [21] for the genetic algorithm and Weka [22] for the SVM. For LSH, the library TarsosLSH[1] has been used. As explained in Section III, before any learning can be processed, each category of available training samples must be hashed separately using LSH. In order to have enough potential neighbours, and after empirical tests, it has been decided to use 10 tables with only 1 hash per table.

Table I. PARAMETERS USED IN THE IMPLEMENTATION OF THE FRAMEWORK

GA	NSGAII
Population size	20
Crossover	two points crossover
Crossover probability	1
Mutation	none
Tournament selection size	2
Genome	2048 floats
SVM library	LibLinear
C	1
Distance	L2
LSH tables	10
LSH hash per table	1

A multi-objective GA is needed for the framework. NSGA-II [13] has been chosen, using the implementation provided by ECJ. Each GA has a population of 20 individuals. The biased initialization process is done on the whole population. The crossover is a two point crossover and the crossover probability is 1. No mutation has been used in this implementation. The selection is a tournament selection of size 2. As visual descriptors extracted are normalized float vectors with 2048 dimensions, individuals in the GA are represented with float vectors with 2048 dimensions bounded to [-1, 1]. The distance used to evaluate the proximity of two vectors is the L2 distance. The evaluation process of NSGA-II has been tuned to call the classifier learning process when necessary and the two objectives explained in Section III have been defined.

The SVM used is a libLinear wrapper for Weka[2]. It is used with the complexity parameter set to 1 and the default solver.

[1]https://github.com/JorenSix/TarsosLSH (09/29/2015)
[2]https://github.com/bwaldvogel/liblinear-weka (09/29/2015)

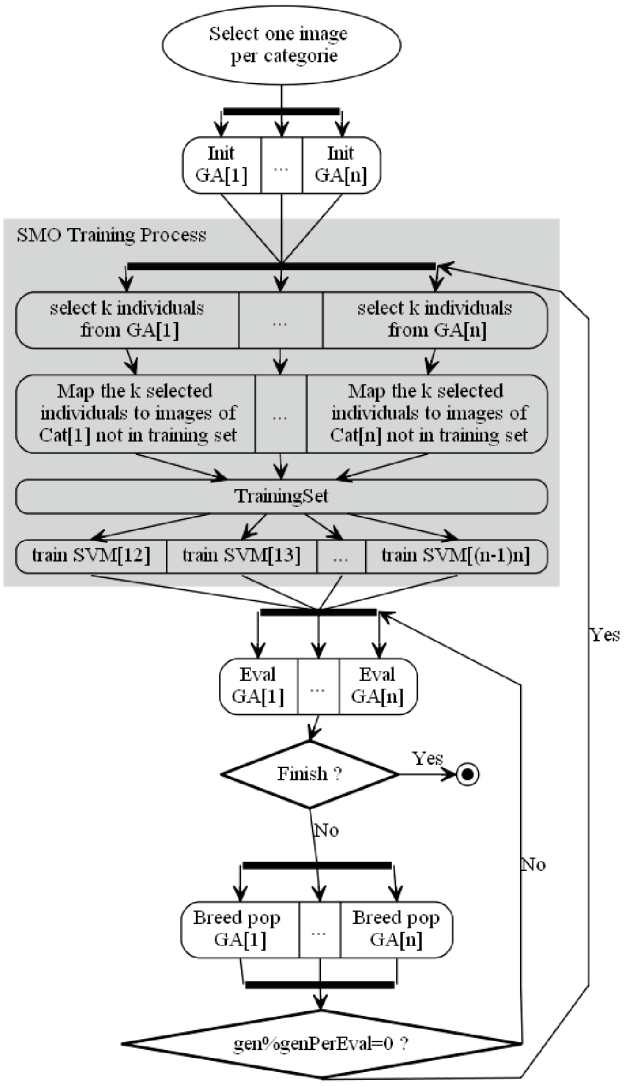

Figure 12. Detailed work flow of our implementation

Experimental results showed that the value of the complexity parameter has barely no influence on classification results, so we did not bother tuning it more. All these parameter settings are reported in the Tab.I.

Fig.12 shows a detailed workflow of our implementation. During the mapping process, individuals of each GA are mapped to true samples from the dataset that have not yet been added to the training set.

This ends the definition of the experimental set-up and parameters. The results will now be presented and analyzed in the next section.

V. Results

Tab.II presents the results obtained with the proposed framework against some of the previous works reported in [18] and the implemented active SVM [20].

All the approaches from "Kernel Codebook" down to "IFK (SIFT)" are focusing on building the most powerful visual content representations to be then input in a classifier. We have decided to use the same visual content representation as in "IFK (SIFT)" [18] since they provide the best results. In their approach, Perronnin et al. [18] consider a linear SVM to classify the images. Since our framework is based on active learning to incrementally build the training set, as it has been shown to be more efficient to train a SVM with less data [20], we also evaluate our framework against the standard active learning approach for SVM [20], in order to get a complete comparison. This approach is reported right above our results in the Tab.II.

Table II. Comparison of our approach with results from the literature on Caltech-256

Method	Nb. of images per category in the training set			
	15	**30**	**45**	**60**
Kernel Codebook		27.20		
EMK (SIFT)	23.20	30.50	34.40	37.60
Standard FK (SIFT)	25.60	29.00	34.90	37.60
Sparse Coding (SIFT)	27.70	34.00	37.50	40.10
Baseline (SIFT)		34.10		
NN (SIFT)		38.00		
NN		**42.70**		
IFK (SIFT)	34.70	40.80	45.00	47.90
Active SVM [20]	*32.22*	*35.85*	*38.59*	*40.58*
Our approach	**34.96**	42.03	**47.99**	**53.35**

For the first training set, composed of 15 training data from each category, the proposed approach have similar results to those obtained in [18] and by the active SVM. It was expected because the initialization consists in selecting images at random to create the first population for the GA, and the first training samples for the SVM are extracted from this random population.

From 30 samples per category, the MOGA process has started. As expected, the use of MOGA helps identifying good training samples and thus leads to better performances than a random selection. The presented framework has an accuracy 2% better than the random selection at 30 images per category, almost 3% better at 45 images and more than 5% better at 60 images. The gap between the proposed approach and results presented in [18] increases with the number of training samples used per category. Those results confirm that the MOGA helps finding training data that improve the classification performances of the SVM.

We can also see that the active learning scheme performed worse than the random selection. This active scheme is dedicated to binary SVM and selects the most uncertain data as new training samples to add them to the training set, one after the other. Authors of [20] proved that this was the best way to select training samples for a binary SVM. To adapt it to our context, we added at each iteration the 15 most uncertain data of each category to the training set. This means that we apply this active selection strategy in a multi-class context and we perform selection of individuals 15 by 15 instead of 1 by 1. Obtained results show that this strategy is not good and so this simple extension of the binary SVM active learning scheme is not ideal. We are also doing an active learning task so we can compare our results to those of this method. And we can see that the selection of training samples through the MOGA

improved a lot the SVM classification performances. Indeed, the MAP gains more than 6% at 30 images per category, more than 9% at 45 images per category and almost 13% at 60 images per category. As for the comparison with the random selection, we can see that the gap increases with the number of training samples used.

During the learning process, the MOGA uses the partially learnt SVM to help guiding the search and explore relevant areas. Each time that new data are added to the SVM training set, the SVM performances increase. This changes part of the MOGA fitness computation, meaning that the MOGA goals adapt to the SVM needs. This can explain the increasing gap between the presented framework results and other methods.

VI. Conclusion

This paper has introduced an innovative way to select training data for SVM from a large set of available samples. The method proposes to build the training set incrementally. The training data are extracted from a GA population, which evolves using the partially learnt classifier as part of its evaluation process to find the best training data to add to the training set.

When applied to the CBIR problems on the caltech-256 dataset, this method has shown promising results, providing performances that are far more better than the classical random sampling used in state of the art methods.

Owing to the evaluation protocol used for comparison purposes, the MOGA is stuck to only one category. This limits its exploratory performances. In addition, the objectives of the MOGA were inspired by an active learning method dedicated to binary SVM [20] because each MOGA was learning training samples for a specific class. However, we want to increase the global classification performances of the SVM. Therefore, future works will focus on releasing the category constraint on the MOGA, allowing the search of training samples in the whole set of available training samples. This will give more importance to the exploratory capacity of the GA. Crafting objectives from existing active learning methods dedicated to multi-class SVM [23] is also considered. This should highly improve the quality of the training samples retrieved by the MOGA. An adaptation of the MOGA evolving individuals in the discrete space instead of the underlying continuous space is our current on-going work. All those elements leave a big room for improving of this promising framework.

This framework will also now be tested against larger and harder datasets as the one from Large Scale Visual Recognition Challenge [24]. We consider then to use pre-trained Convolutional Neural Network features as input of our framework instead of Improved Fisher Vectors.

VII. Acknowledgement

We would like to acknowledge Florent Perronnin who kindly helped us with the Fisher Vectors he used in [18]. This work is partly funded by French National Agency for Research, VISIIR project, under contract number ANR-13-CORD-0009.

References

[1] J. Balcázar, Y. Dai, and O. Watanabe, "A random sampling technique for training support vector machines," in *Algorithmic Learning Theory*, ser. Lecture Notes in Computer Science, 2001, vol. 2225, pp. 119–134.

[2] J. Wang, S. Kumar, and S.-F. Chang, "Semi-supervised hashing for scalable image retrieval," in *IEEE CVPR*, San Francisco, USA, 2010.

[3] J. He, W. Liu, and S.-F. Chang, "Scalable similarity search with optimized kernel hashing," in *ACM SIGKDD Conference*, Washington, DC, USA, 2010.

[4] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[5] M. Kawulok and J. Nalepa, "Support vector machines training data selection using a genetic algorithm," in *Structural, Syntactic, and Statistical Pattern Recognition*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2012, vol. 7626, pp. 557–565.

[6] M. Arevalillo-Herráez, F. J. Ferri, and S. Moreno-Picot, "A hybrid multi-objective optimization algorithm for content based image retrieval," *Applied Soft Computing*, vol. 13, no. 11, pp. 4358 – 4369, 2013.

[7] S. Moreno-Picot, F. Ferri, and M. Arevalillo-Herráez, "A nsga based approach for content based image retrieval," in *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, ser. Lecture Notes in Computer Science, J. Ruiz-Shulcloper and G. Sanniti di Baja, Eds. Springer Berlin Heidelberg, 2013, vol. 8258, pp. 359–366.

[8] M. Arevalillo-Herráez, F. J. Ferri, and S. Moreno-Picot, "Distance-based relevance feedback using a hybrid interactive genetic algorithm for image retrieval," *Applied Soft Computing*, vol. 11, no. 2, pp. 1782 – 1791, 2011.

[9] C.-C. Lai and Y.-C. Chen, "A user-oriented image retrieval system based on interactive genetic algorithm," *Instrumentation and Measurement, IEEE Transactions on*, vol. 60, no. 10, pp. 3318 –3325, oct. 2011.

[10] T. Kanimozhi and K. Latha, "An integrated approach to region based image retrieval using firefly algorithm and support vector machine," *Neurocomputing*, vol. 151, Part 3, no. 0, pp. 1099 – 1111, 2015.

[11] M. Broilo and F. De Natale, "Evolutionary image retrieval," in *IEEE ICIP*, Nov 2009, pp. 1845–1848.

[12] C. Coello Coello, C. Dhaenens, and L. Jourdan, "Multi-objective combinatorial optimization: Problematic and context," in *Advances in Multi-Objective Nature Inspired Computing*, ser. Studies in Computational Intelligence. Springer Berlin Heidelberg, 2010, vol. 272, pp. 1–21.

[13] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *Trans. Evol. Comp*, vol. 6, no. 2, pp. 182–197, Apr. 2002.

[14] D. Gorisse, M. Cord, and F. Precioso, "Scalable active learning strategy for object category retrieval," in *Image Processing (ICIP), 2010 17th IEEE International Conference on*, Sept 2010, pp. 1013–1016.

[15] P. Indyk and R. Motwani, "Approximate nearest neighbors: towards removing the curse of dimensionality," *ACM*, pp. 604–613, 1998.

[16] A. Gionis, P. Indyk, and R. Motwani, "Similarity search in high dimensions via hashing," in *International Conference on VLDB*, 1999, pp. 518–529.

[17] M. Datar, N. Immorlica, P. Indyk, and V. Mirrokni, "Locality-sensitive hashing scheme based on p-stable distributions," *SCG*, pp. 253–262, 2004.

[18] F. Perronnin, J. Sánchez, and T. Mensink, "Improving the fisher kernel for large-scale image classification," in *ECCV*, 2010, pp. 143–156.

[19] G. Griffin, A. Holub, and P. Perona, "Caltech-256 object category dataset," California Institute of Technology, Tech. Rep. 7694, 2007. [Online]. Available: http://authors.library.caltech.edu/7694

[20] S. Tong and E. Chang, "Support vector machine active learning for image retrieval," in *ACM MM*, 2001, pp. 107–118.

[21] D. White, "Software review: the ecj toolkit," *Genetic Programming and Evolvable Machines*, vol. 13, no. 1, pp. 65–67, 2012.

[22] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: An update," *SIGKDD Explor. Newsl.*, vol. 11, no. 1, pp. 10–18, Nov. 2009.

[23] A. Joshi, F. Porikli, and N. Papanikolopoulos, "Scalable active learning for multiclass image classification," *IEEE Trans. on PAMI*, vol. 34, no. 11, pp. 2259–2273, Nov 2012.

[24] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "Imagenet large scale visual recognition challenge," *IJCV*, pp. 1–42, April 2015.