UNIMORE at ImageCLEF 2013: Scalable Concept Image Annotation

Costantino Grana[1], Giuseppe Serra[1], Marco Manfredi[1], Rita Cucchiara[1], Riccardo Martoglia[2], and Federica Mandreoli[2]

[1] University of Modena and Reggio Emilia - DIEF Department
[2] University of Modena and Reggio Emilia - FIM Department
`<name.surname>@unimore.it`

Abstract. In this paper we propose a large-scale Image annotation system for the Scalable Concept Image Annotation task. For each concept to be detected a separated classifier is built using the provided textual annotation. Images are represented as a Multivariate Gaussian distribution of a set of local features extracted over a dense regular grid. Textual analysis, on the web pages containing training images, is performed to retrieve a relevant set of samples for learning each concept classifier. An online SVMs solver based on Stochastic Gradient Descent is used to manage the large amount of training data. Experimental results show that the combination of different kind of local features encoded with our strategy achieves very competitive performance both in terms of mAP and mean F-measure.

Keywords: image retrieval, image classification, multi-class, multi-label, stochastic gradient descent

1 Introduction

University of Modena and Reggio Emilia group (UNIMORE) participated at ImageCLEF 2013 [1] to the Scalable Concept Image Annotation Task [2], and identified two possible strategies to attack the problem: finding images similar to the query, and from those extract the image concepts, leveraging the provided textual annotation, or directly using the textual annotation to roughly annotate the training set and then for every concept building a classifier applicable to the query.

The first approach is the organizers baseline, while for the second we annotated every training image with a concept if the concept word was found in the scofeats file, then we used the provided BoW CSIFT features to build a classifier for each concept. This second strategy largely outperformed the baseline, so we further expanded this second approach. In our experiments we aimed at improving the features and the initial textual annotation. Instead of relying on the BoW model we propose to describe the local features as a Multivariate Gaussian Distribution, which employs a full rank covariance matrix, thus leading to a large feature vector (for a 128 dimensional SIFT descriptor a 8,384 dimensional vector for each spatial pyramid region). We partitioned the image into 1x1, 2x2, 1x3 regions, thus a total of 8 spatial regions and a 67,072 dimensional vector which becomes 201,216 dimensional for color based SIFTs.

For the textual part, stopword removal and stemming is performed on the scofeats file, then the titles of the original web pages are extracted and parsed. Moreover we built a context from the WordNet definition, in order to detect if different words fall in the same concept context. Finally a negative context is similarly built by other senses of the same word.

We used a linear SVM classifier for each concept, built using a Stochastic Gradient Descent online technique, which allowed us to provide one example at a time to the algorithm, thus allowing training within our memory limit (6 biprocessor Xeon machines with 32 GB of RAM each). Parallelization was achieved by separately training the classifiers on every machine on chunks of data read from disk. A late fusion averaging approach is used in our best run (UNIMORE5_test) with the HSVSIFT, OPPONENTSIFT, RGBSIFT, and SIFT features. We further improved the training set by querying about 100k images from Google Images with the concept name. We managed to be the best group in terms of mAP-samples, the second in terms of MF-concepts and the third in terms of MF-samples.

In Section 2 we describe the feature sumarization approach, while in Section 3 the textual annotation processing method is presented. In Section 4 the Stochastic Gradient Descent is briefly sumarized and our modifications are highlighted. Finally Section 5 describes the submitted runs in detail and reports the performance obtained on both the development and the test sets.

2 Visual Information Processing

For an image W, we first extract features through densely sampling in a regular grid. Let $F = \{\mathbf{f}_1 \dots \mathbf{f}_\mathbf{N}\}$ be a set of local features (e.g. SIFT descriptors, where $d = 128$) in W (or a sub-region of W, when Spatial Pyramid Matching is used), we describe them with a multivariate Gaussian distribution supposing that they are normally distributed. The multivariate Gaussian distribution of a set of d-dimensional vectors F is given by

$$\mathcal{N}(\mathbf{f}; \mathbf{m}, \mathbf{C}) = \frac{1}{|2\pi\mathbf{C}|^{\frac{1}{2}}} e^{-\frac{1}{2}(\mathbf{f}-\mathbf{m})^T \mathbf{C}^{-1}(\mathbf{f}-\mathbf{m})}, \tag{1}$$

where $|\cdot|$ is the determinant, $\mathbf{m}$ is the mean vector and $\mathbf{C}$ is the covariance matrix ($\mathbf{f}, \mathbf{m} \in \mathbb{R}^d$ and $\mathbf{C} \in \mathbb{S}^{d\times d}_{++}$, with $\mathbb{S}^{d\times d}_{++}$ the space of real symmetric positive semi-definite matrices) defined as follows:

$$\mathbf{m} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{f}_i, \tag{2}$$

$$\mathbf{C} = \frac{1}{N-1} \sum_{i=1}^{N} (\mathbf{f}_i - \mathbf{m})(\mathbf{f}_i - \mathbf{m})^T. \tag{3}$$

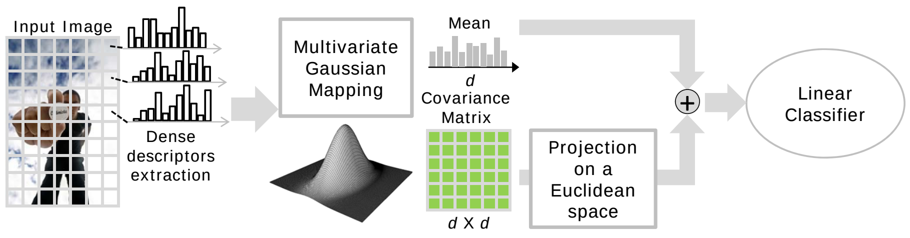
Input Image
Dense descriptors extraction
Multivariate Gaussian Mapping
Mean
d
Covariance Matrix
d X d
Projection on a Euclidean space
Linear Classifier

Fig. 1. Overview of visual information processing.

The covariance matrix encodes information about the variance of the features and their correlation. Although it is very informative, it does not lie in a vector space since the covariance space is not closed under multiplication with a negative scalar. In fact, it lies in a Riemannian manifold. Most of the common machine learning algorithms assume that the data points form a vector space, therefore a suitable transformation is required prior to their use. Since the covariance matrix is symmetric positive definite we adopt the Log-Euclidean metric. The basic idea of the Log-Euclidean metric is to construct an equivalent relationship between the Riemannian manifold and the vector space of the symmetric matrix.

An approach to map from Riemannian manifolds to Euclidean spaces is introduced in [3] and used in [4]. The first step is the projection of the covariance matrices on an Euclidean space tangent to the Riemannian manifold, on a specific tangency matrix $\mathbf{P}$. The second step is the extraction of the orthonormal coordinates of the projected vector. In the following, matrices (points in the Riemannian manifold) will be denoted by bold uppercase letters, while vectors (points in the Euclidean space) by bold lowercase ones.

More formally, the projected vector of a covariance matrix $\mathbf{C}$ is given by:

$$\mathbf{t}_{\mathbf{C}} = \log_{\mathbf{P}}(\mathbf{C}) = \mathbf{P}^{\frac{1}{2}} \log\left(\mathbf{P}^{-\frac{1}{2}}\mathbf{C}\mathbf{P}^{-\frac{1}{2}}\right)\mathbf{P}^{\frac{1}{2}} \tag{4}$$

where log is the matrix logarithm operator and $\log_{\mathbf{P}}$ is the manifold specific logarithm operator, dependent on the point $\mathbf{P}$ to which the projection hyperplane is tangent. The matrix logarithm operators of a matrix C can be computed by eigenvalue decomposition ($\mathbf{C} = \mathbf{U}\mathbf{D}\mathbf{U}^T$); it is given by:

$$log(\mathbf{C}) = \sum_{k=1}^{\infty} \frac{(-1)^{k-1}}{k}(\mathbf{C}-\mathbf{I})^k = \mathbf{U}log(D)\mathbf{U}^T. \tag{5}$$

The orthonormal coordinates of the projected vector $\mathbf{t}_{\mathbf{C}}$ in the tangent space at point $\mathbf{P}$ are then given by the vector operator:

$$\mathrm{vec}_{\mathbf{P}}(\mathbf{t}_{\mathbf{C}}) = \mathrm{vec}_{\mathbf{I}}\left(\mathbf{P}^{-\frac{1}{2}}\mathbf{t}_{\mathbf{C}}\mathbf{P}^{-\frac{1}{2}}\right) \tag{6}$$

where $\mathbf{I}$ is the identity matrix, while the vector operator on the tanget space at identity of a symmetric matrix $\mathbf{Y}$ is defined as:

$$\mathrm{vec}_{\mathbf{I}}(\mathbf{Y}) = \left[y_{1,1}\ \sqrt{2}y_{1,2}\ \sqrt{2}y_{1,3} \ldots y_{2,2}\ \sqrt{2}y_{2,3} \ldots y_{d,d} \right]. \tag{7}$$

Substituting $\mathbf{t_C}$ from Eq. 4 in Eq. 6, the projection of $\mathbf{C}$ on the hyperplane tangent to $\mathbf{P}$ becomes

$$\mathbf{c} = \mathrm{vec}_{\mathbf{I}} \left(\log \left(\mathbf{P}^{-\frac{1}{2}} \mathbf{C} \mathbf{P}^{-\frac{1}{2}} \right) \right). \tag{8}$$

Thus, after selecting an appropriate projection origin, every covariance matrix is projected to an Euclidean space. Since $\mathbf{c}$ is a symmetric matrix of size $d \times d$ a $(d^2 + d)/2$-dimensional feature vector is obtained.

As observed in [5], the projection point $\mathbf{P}$ is arbitrary and, even if it could influence the performance (distortion) of the projection, from a computational point of view, the best choice is the identity matrix, which simply translates the mapping into a standard matrix logarithm.

In short, our method (see Fig. 1) is to extract local descriptors from an image and then collect them in a spatial pyramid; each sub-region is described by a multivariate Gaussian distribution (MGD). The covariance matrix is projected on a Euclidean space and concatenated to the mean vector to obtain the final descriptor (in the case of SIFT descriptors, the dimensionality becomes 8384 per sub-region). We empirically observe that most of the values in the concatenated descriptor are low, while few are high. In order to distribute the values more evenly, we adopt the power normalization method proposed by Perronnin et al. [6], i.e. to apply to each dimension the function:

$$f(x) = sign(x)|x|^{\alpha} \quad \text{with } \alpha = 0.5 \tag{9}$$

Eventually, the concatenated descriptors are fed to a linear classifier. For a more detailed analysis of the proposed multivariate Gaussian descriptor see [7].

3 Textual Information Processing

The goal of textual information processing is, given a list $L_{conc} = \{c_1, \ldots, c_n\}$ of concepts of interest, to retrieve a relevant set of images from the ImageCLEF training set exploiting only the textual content of the web pages that referenced the images. The concepts c are expressed as WordNet[3] synsets, for instance `airplane.n.1` is the first sense of the term `airplane` as a noun; the list can include more than one synset, as they could be equally relevant, as in $L_{conc} =$ {`book.n.1, book.n.2`}. The retrieved image set I will then be used to train ad-hoc image classifiers in identifying the specific concepts in the test image set. In particular, in order for the training to be effective, the text processing techniques should be designed to retrieve a set of images:

(a) sufficiently large so to perform training (a minimum number of images threshold th_{min} should be exceeded);

[3] `http://wordnet.princeton.edu/`

(b) as relevant as possible to the concepts.

One of the most naive approaches for text processing, which also constitutes a typical baseline, could be accomplished in very few simple steps. For instance, for a given $c \in L_{conc}$ (e.g. `airplane.n.1`):

1. extract the main term t associated to c (e.g. `airplane`);
2. look for t in the "scofeats" data file, containing, for each of the training images, the processed text of the referencing web pages, and retrieve in I the images referenced from the web pages where t appears.

Following the above baseline, however, brings to very unsatisfying results due to a large number of both (a) false negatives and (b) false positives in I, thus failing to meet the above mentioned desiderata. The following are just some real examples for `airplane.n.1`:

– many relevant and useful images are missing since the original pages described the concepts using different terms (e.g. `aeroplane`, `jumbojet`, etc);
– many unrelated images are retrieved, for instance the closeup of a hat (from a web page about "airplane pilot hats"), the album covers and group shots of the "Jefferson Airplane" music band, etc.

3.1 Enhanced Text Processing Approach

In order to overcome the above mentioned issues, we exploit an enhanced text processing approach, whose steps are outlined in Figure 2. First of all, a textual information pre-processing is executed on the "scofeats" data and on the original web pages data (top part of the figure):

– *stopword removal and stemming* is performed on the "scofeats" file, thus producing a "stemmed scofeats" file;
– the titles of the original web pages are extracted and parsed (*title extraction and analysis* step), thus producing a "parsed page titles" file.

Note that, in our approach, we choose to exploit first of all the textual features already extracted in the "scofeats" file (including the term scores), complementing them with specific information extracted from the original web pages which would be otherwise unavailable. The output of pre-processing, then, enables the actual textual information search process (bottom part of the figure) that, given an input list of concepts L_{conc}, produces the associated result image set I. With respect to the baseline described in the previous section, and for each concept $c \in L_{conc}$ the processing is enhanced in two main directions, corresponding to the original desiderata:

– the number of retrieved images associated to c is significantly higher, thanks to the *candidate image search* step, which produces an expanded candidate image set CI. Each image in CI is associated to its scofeat information for further refinements;

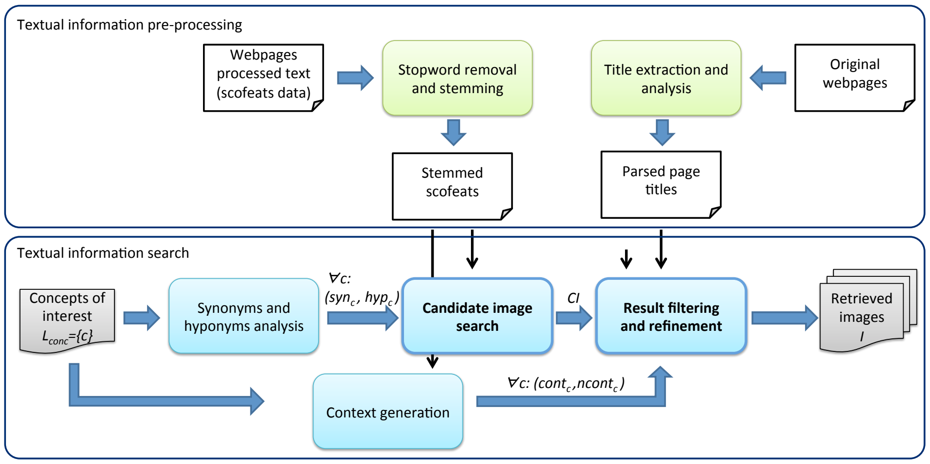
Textual information pre-processing
Webpages processed text (scofeats data)
Stopword removal and stemming
Title extraction and analysis
Original webpages
Stemmed scofeats
Parsed page titles
Textual information search
Concepts of interest $L_{conc}=\{c\}$
Synonyms and hyponyms analysis
∀c: (syn_c, hyp_c)
Candidate image search
CI
Result filtering and refinement
Retrieved images I
Context generation
∀c: ($cont_c$, $ncont_c$)

Fig. 2. Overview of textual information pre-processing (top) and search process (bottom).

– irrelevant images in CI are discarded and a refined set I is produced in a *result filtering and refinement* step.

The techniques behind these two crucial steps, along with the ones propaedeutic to them, will shortly be discussed in the following.

Candidate image search. The first key idea here is to search for term t (e.g. `airplane`) associated to c in the web page processed text of the "stemmed scofeats" file, instead of the plain "scofeats" file. In this way, stemming [8] significantly improves the recall in the image retrieval process, also retrieving pages containing different inflexions of the term (e.g., the plural `airplanes`).

Moreover, the search is performed not only for term t but also for the terms $\bar{t}$ available in its synonyms and hyponyms (i.e. more specific terms) lists, syn_c and hyp_c, respectively. Such lists are extracted from WordNet in the *synonyms and hyponyms analysis* step. For instance, for `airplane` syn_c will include `aeroplane` and hyp_c `biplane`, `jumbojet`, etc., whereas for `rodent` hyp_c will contain the more common `mouse` and `rat` terms. In this way, a set of images significantly larger than the baseline is retrieved, for instance 40% more for `airplane` and even 300% more for `rodent`.

Please note that including in the search all the synonyms and hyponyms of a term, without any discrimination, while enlarging the retrieved image set, could also bring negative effects on its precision, due to ambiguity of language. For instance, an hyponym of `newspaper` is `daily`, an hyponym of `horse` is `bay`; these are however terms which are more typically used in completely different contexts and which would, thus, produce noise in the results. For this reason, we chose the "safest" approach of selecting only those synonyms/hyponyms having a single sense in WordNet; alternatively word sense disambiguation approaches (such as

[9, 10]) could be applied to the web pages before the search (their application will be investigated in the future).

Result filtering and refinement. In this step, various refinement techiques are exploited so to reduce the number of false positives in the CI image set:

- **score threshold**: the "stemmed scofeats" file contains, for each term of the referencing web page, a score which captures the concepts of term frequency [8] and term distance to the image: the more frequent and the more close to the image is a term, the more it should be representative of the image content. Therefore, we define a th_{score} threshold which the score of term t should exceed in order for the image to be put in the final results. For instance, the scofeat for an helicopter image from a web page about various means of transport could contain the term `airplane` but with a very low score, and would thus not be considered;
- **context threshold**: a term in a web page can have very different senses. For instance, when looking for `elder.n.1`, i.e. "a person who is older than you are", one should be very careful to exclude web pages (and its images) which are about elders as "shrubs or small trees". While word sense disambiguation [9, 10] could be applied to the web pages, in order to be able to directly exploit the "stemmed scofeats" information we chose to implicitly derive the sense of a term from its co-occurent terms. In particular, for a given concept c, the *context generation* step first derives a preliminary context $pcont_c$ from the nouns of its WordNet definition, then expands it to a context $cont_c$ containing the terms which most frequently co-occur with the ones of $pcont_c$. For instance, for $c =$`elder.n.1`, $pcont_c$={`elder`, `person`}, while $cont_c$ will include additional terms as `old`, `family`, `people`, `house`, `woman` and `man`. Then, we define a th_{cont} threshold which is the minimum number of terms of $cont_c$ that an image scofeat should have in order for the image to be put in the final results. In this way, when looking for images about `elder.n.1`, images about bushes are typically excluded;
- **negative context check**: in order to enforce context filtering, we also derive a *negative context* $ncont_c$ from the definitions of the other senses of c that the image scofeat should *not* include: for instance when looking for $c =$`castle.n.2` ("a large building"), $ncont_c$ will include misleading terms such as `chess`;
- **page title check**: in this further check we exploit the "parsed page titles" information so to exclude from the results the images whose referencing pages have a title not meeting the desired criteria. In particular, term t should not be used as a deteriminer in a noun phrase, or be present in capitalized form: for instance, images from pages about "airplane pilot hats" or "Jefferson Airplane songs" are excluded as deemed not representative for `airplane`.

Please note that the above refinement techniques are applied only: (a) if they are relevant to the case (for instance, context and negative context are not needed in case of single-sense terms) and (b) if the number of images in the candidate set

CI does not fall under the minimum threshold th_{min} (in particular, thresholds th_{cont} and th_{score} are adjusted so to satisfy the th_{min}=500 limit).

4 Online learning for SVM training

Although there exist many off-the-shelf SVM solvers, such as SVMlight, SVMperf or LibSVM/LIBLINEAR, they are not feasible for training large volumes of data. This is because most of them are batch methods, which require to go through all data to compute gradient in each iteration and often need many iterations to reach a reasonable solution. Even worse, most off-the-shelf batch type SVM solvers require to pre-load training data into memory, which is impossible when the size of the training data explodes. Indeed, LIBLINEAR released an extended version that explicitly considered the memory issue, but in a recent test [11] it was shown that the performance dropped considerably and even on 80GB of training data it could not provide useful results. Therefore, a better solution may be provided by the stochastic gradient descent (SGD) algorithm recently introduced for SVM classifiers training.

We have training data that consists of T feature-label pairs, denoted as $\{\mathbf{x}_t, y_t\}_{t=1}^{T}$, where $\mathbf{x}_t$ is a $s \times 1$ feature vector representing an image and $y_t \in \{-1, +1\}$ is the label of the image. Then, the cost function for binary SVM classification can be written as

$$L = \sum_{t=1}^{T} \frac{\lambda}{2} \|\mathbf{w}\|^2 + \max\left[0, 1 - y_t(\mathbf{w}^T \mathbf{x}_t + b)\right], \tag{10}$$

where $\mathbf{w}$ is $s \times 1$ SVM weight vector, λ (nonnegative scalar) is a regularization parameter, and b (scalar) is a bias term. In the SGD algorithm, training data are fed to the system one by one, and the update rule for $\mathbf{w}$ and b respectively are

$$\begin{aligned} \mathbf{w}_t &= (1 - \lambda\eta)\mathbf{w}_{t-1} + \eta y_t \mathbf{x}_t \\ b_t &= b_{t-1} + \eta y_t \end{aligned} \tag{11}$$

if margin $\Delta_t = y_t(\mathbf{w}^T \mathbf{x}_t + b)$ is less than 1; otherwise, $\mathbf{w}_t = (1 - \lambda\eta)\mathbf{w}_{t-1}$ and $b_t = b_{t-1}$. The parameter η is the step size. We set $\eta = (1 + \lambda t)^{-1}$, following the `vl_pegasos` implementation [12]. To parallelize SVMs training, we randomize the data on disk. We load the data in chunks which fit in memory, then train the different classifiers in parallel threads on further randomizations of the chunks, so that different epochs will get the chunks data with different orderings.

In our experimental setting each classifier is trained considering all the 250,000 images given as training; as a result the data are highly unbalanced, namely the negative samples are predominant, and in addition the number of samples per concept is unevenly distributed. We observed that this leads the classifier to incorrectly estimate the hyperplane, that is shifted towards the positive samples while maintaining a proper orientation. To reduce this effect, a simultaneous optimization of the SVMs bias for all the classifiers is conducted by maximizing the F-measure on all the training set.

5 Experimental results

UNIMORE participated to the Scalable Concept Image Annotation task submitting six runs. All the different kind of SIFT descriptors are extracted at four scales, defined by setting the width of the local feature spatial bins to 4, 6, 8, and 10 pixels respectively, over a dense regular grid with a spacing of 3 pixels. We use the function `vl_phow` provided by the `vl_feat` library [12] and, apart from the spacing step, the defaults options are used. Images are hierarchically partitioned into 1×1, 2×2 and 1×3 blocks on 3 levels respectively. In the case of SIFT descriptor we obtain a 67,072 dimensional vector concatenating the MGD features of the 8 spatial windows, while for color SIFT descriptors (RGB, OPPONENT and HSV) we describe a region by concatenating the MGD computed for each color channel separately (instead of using the full covariance matrix of 384 dimensions) obtaining a 201,216 dimensional feature vector. Stochastic gradient descent is used to learn a classifier for each concept, for each feature descriptor and each training set; detection scores are thresholded at zero to obtain the decisions. Finally, a late fusion averaging approach is used.

In some runs we added another training set of about 100k images queried from Google Image Search directly using the concepts list. Each image is automatically labeled with the concept word used in the query; synonyms and hyponyms analysis is also performed in order to identify labels relationship (e.g. images labeled as "car" are also tagged with the "vehicle" label). These images are described with RGBSIFT and summarized with a Multivariate Gaussian Descriptor. All experiments are performed on six biprocessor Xeon machines with 32 GB of RAM each. Runs are described and discussed in the following:

– **UNIMORE_1:** Training images are associated to a concept using the first step of the textual information processing, called "candidate image search", on the scofeats file. For every image multiscale HSVSIFT and RGBSIFT features are extracted and summarized with a Multivariate Gaussian Descriptor.
– **UNIMORE_2:** Based on the scofeats file, two different sets of training images are associated to a concept: 1) images linked to a concept when the concept word is present in the stemmed scofeats; 2) images obtained by the candidate image search technique. For every image, multiscale HSVSIFT, OPPONENTSIFT, RGBSIFT and SIFT features are extracted and summarized with a MGD. Google Images dataset is also used for training.
– **UNIMORE_3** Training images are associated to a concept only if the concept word is present in the stemmed scofeats file. For every image multiscale HSVSIFT, OPPONENTSIFT, RGBSIFT and SIFT features are extracted and summarized with a MGD. Google Images dataset is also used for training.
– **UNIMORE_4** Training images are associated to a concept if the concept word is present in the stemmed scofeats file. For every image multiscale HSVSIFT, OPPONENTSIFT, RGBSIFT features are extracted and summarized with a MGD.

- **UNIMORE_5** Three sets of training images, based of the scofeats file, are associated to a concept : 1) an image is linked to a concept when the concept word is present in the stemmed scofeats file; 2) images obtained by the candidate image search technique; 3) images obtained applying the complete textual information processing pipeline. For every image multiscale HSVSIFT, OPPONENTSIFT, RGBSIFT and SIFT features are extracted and summarized with a MGD. In addition, Google Images dataset is used for training. Two different combination strategies are used to compute decisions and score values: decisions are computed through a late fusion averaging approach of classifiers trained with images derived by the candidate image search technique and described with HSVSIFT, OPPONENTSIFT and RGBSIFT descriptors, while the score values are obtained combining all the classifiers learned in this run.
- **UNIMORE_6** Two sets of training images, based on the scofeats file, are associated to a concept: 1) an image is linked to a concept when the concept word is present in the stemmed scofeat file; 2) images obtained by the candidate image search technique. For every image multiscale HSVSIFT, OPPONENTSIFT, RGBSIFT and SIFT features are extracted and summarized with a MGD. Google Images dataset is also used for training. Two different combination strategies are used to compute decisions and score values: decisions are computed through a late fusion averaging approach of classifier trained with images derived by the candidate image search technique and described with HSVSIFT and RGBSIFT features, while the score values are obtained combining all the classifiers learned in this run.

Tab. 1 and 2 present the results obtained for each run in term of mAP (mean average precision) and MF (mean F-measure). The MF is computed analyzing both the samples (MF-samples) and the concepts (MF-concepts), whereas the mAP is computed analyzing the samples. It can be noted that the performance reported in all the three metrics in the development and test sets are strictly related, and shows slightly lower results in the latter. This is probably due to the higher number and variability of the concepts given in the test setting. The late fusion averaging approach proves to be a good solution for combining different features and training sets and for easily learning classifiers in parallel. In particular it greatly improves the mAP value, that increases for each new feature or training set added in the system (for example see runs UNIMORE_1 and UNIMORE_2). Adding our Google Images dataset, automatically downloaded using concepts list, increases the performance in term of mAP although the high level of label noise (see runs UNIMORE_3 and UNIMORE_4). Textual information processing is also essential for the proposed method, mainly because it increases the performance in term of MF and defines new training sets to be used in the late fusion approach. It can be noted that both textual processing steps contribute to the improvement of the performance: see for example the gap in terms of MF between runs UNIMORE_4 and UNIMORE_1 mainly caused by candidate image search strategy and the difference of mAP values between

Table 1. Development set results

	MF-samples	**MF-concepts**	**mAP-Samples**
baseline_rand	6.2	4.8	10.9
baseline_sift	17.8	11.0	24.0
UNIMORE_1	33.0	34.1	39.2
UNIMORE_2	27.3	34.2	46.0
UNIMORE_3	23.1	32.4	43.7
UNIMORE_4	26.8	31.7	39.7
UNIMORE_5	33.3	33.7	47.9
UNIMORE_6	33.0	34.1	46.0

Table 2. Test set results

	MF-samples	**MF-concepts**	**mAP-Samples**
baseline_rand	4.6	3.6	8.7
baseline_sift	15.9	11.0	21.0
UNIMORE_1	31.1	32.0	36.7
UNIMORE_2	27.5	33.1	44.1
UNIMORE_3	23.1	31.5	41.9
UNIMORE_4	24.1	29.5	36.2
UNIMORE_5	31.5	31.9	45.6
UNIMORE_6	31.1	32.0	44.1

runs UNIMORE_6 and UNIMORE_5 obtained applying the complete textual information processing pipeline.

6 Conclusions

In this paper we presented the approach developed to participate at ImageCLEF 2013 for the Scalable Concept Image Annotation task. Our proposal focus on the definition of a new image descriptor that encodes local features, densely extracted from a region, as a Multivariate Gaussian Distribution. A new textual information processing strategy is also presented to cope with the high level of noise of the training data. To deal with the large-scale nature of this task, we use an online linear SVM classifier based on the Stochastic Gradient Descent algorithm. Experimental results show that both visual and textual information processing are necessary to build a competitive system.

References

1. Caputo, B., Muller, H., Thomee, B., Villegas, M., Paredes, R., Zellhofer, D., Goeau, H., Joly, A., Bonnet, P., Martinez Gomez, J., Garcia Varea, I., Cazorla, M.: Imageclef 2013: the vision, the data and the open challenges. In: Proc CLEF. (2013)

2. Villegas, M., Paredes, R., Thomee, B.: Overview of the imageclef 2013 scalable concept image annotation subtask. In: CLEF 2013 working notes, Valencia, Spain. (2013)
3. Tuzel, O., Porikli, F., Meer, P.: Pedestrian Detection via Classification on Riemannian Manifolds. IEEE T. Pattern Anal. **30**(10) (2008) 1713–1727
4. Borghesani, D., Grana, C., Cucchiara, R.: Miniature illustrations retrieval and innovative interaction for digital illuminated manuscripts. Multimedia Systems (2013)
5. Martelli, S., Tosato, D., Farenzena, M., Cristani, M., Murino, V.: An FPGA-based Classification Architecture on Riemannian Manifolds. In: DEXA Workshops. (2010)
6. Perronnin, F., Sánchez, J., Mensink, T.: Improving the fisher kernel for large-scale image classification. In: ECCV. (2010)
7. Grana, C., Serra, G., Manfredi, M., Cucchiara, R.: Image classification with multivariate gaussian descriptors. In: ICIAP. (2013)
8. Baeza-Yates, R.A., Ribeiro-Neto, B.: Modern Information Retrieval. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA (1999)
9. Mandreoli, F., Martoglia, R.: Knowledge-Based Sense Disambiguation (Almost) For All Structures. Information Systems (Information) **36**(2) (2011) 406–430
10. Mandreoli, F., Martoglia, R., Ronchetti, E.: Versatile Structural Disambiguation for Semantic-aware Applications. In: ACM International Conference on Information Knowledge and Management. (2005)
11. Lin, Y., Lv, F., Zhu, S., Yang, M., Cour, T., Yu, K.: Large-scale image classification: Fast feature extraction and svm training. In: CVPR. (2011)
12. Vedaldi, A., Fulkerson, B.: VLFeat: An open and portable library of computer vision algorithms. http://www.vlfeat.org/ (2008)