ONLINE KERNEL LEARNING FOR INTERACTIVE RETRIEVAL IN DYNAMIC IMAGE DATABASES

Philippe-Henri Gosselin

ONLINE KERNEL LEARNING FOR INTERACTIVE RETRIEVAL IN DYNAMIC IMAGE DATABASES

Philippe-Henri Gosselin

ETIS Laboratory, CNRS, ENSEA, Univ. Cergy-Pontoise
BP44 F95014 Cergy-Pontoise, France

ABSTRACT

In this paper, we propose a system for interactive image retrieval in dynamic databases, where images are regularly added or removed. In order to handle this, we propose a method that tunes itself according to user labels. The framework we propose is based on visual dictionaries, with the specificity that the dictionaries are built online, during retrieval sessions. In other words, each user has its own visual dictionary, as opposed to usual approaches where all users share the same visual dictionary. In order to create theses dictionaries, we propose a method based on kernel functions. This method iteratively selects base kernels from a large base kernel pool, where each base kernel is related to a low-level descriptor such as color or texture. This learning process is performed in real time, and the classification of the database is faster than usual techniques since only relevant features for the current query are used. Experiments are carried out on a generalist database, and show the ability of the method to build effective kernels with few labels.

Index Terms— Image databases, Interactive systems, Machine learning algorithms, Boosting

1. INTRODUCTION

During the last decade, many retrieval methods have been proposed for interactive retrieval in multimedia database. In order to use these methods with effectiveness, a tuning step is usually required. In the case of static databases, this is easily handled. However, in the case of dynamic databases, where images are regularly added, a steady tuning of the retrieval method is necessary.

In this paper, we propose to focus on dynamic databases with retrieval methods which require minimal tuning, and in the ideal case no tuning at all. In order to reach this goal, we propose to work with kernel-based methods, a framework that has shown its effectiveness for many tasks including multimedia retrieval. In this context, a usual approach is to first extract low-level features and build a visual dictionary using for instance K-means, Gaussian Mixtures[1] or reconstruction[2]. Visual dictionaries are then used to build a vector for each image in the database, like histograms, bag of words, densities, etc. Then a kernel function on these vectors is selected and tuned to the current database. Once these steps done, retrieval sessions can begin, usually using SVM for database classification and active learning for interaction with users [3].

As we can see, these methods have two main offline steps : dictionary/vectors building and kernel tuning. In this paper, we propose to perform these two steps online, during the retrieval. The idea is to use labels provided by the user to perform the tunings. A simple approach would be to run semi-supervised versions of dictionary methods, but this is intractable from a computational point of view. Instead, we propose to learn both the dictionary and the kernel in a single step. More specifically, we propose to learn a linear combination of base kernel, where each base kernel is related to a low-level feature or a codeword.

2. INTERACTIVE LEARNING

In this paper, we propose a method based on kernel functions, Support Vector Machines (SVM) and active learning to interact with the user [4, 5]. A retrieval session is performed in the following way:

Initialization. A retrieval session is initialized from one image brought by the user. The kernel function is used to compute the similarity between this query image and all images in the database. The images with the highest similarities are then presented to user. Note that other initializations could be used, for instance with keywords. If the user is not satisfied by this initial result, he labels images as relevant or irrelevant.

Classification. The images the user labeled are used to train a SVM classifier. This classifier is based on a kernel function, usually a static kernel function, like a Gaussian with a L^2 or a χ^2 distance. In this paper, we present a dynamic kernel function, that is learned from the current user labels. Relevance of images in the database are then evaluated using the hyperplane computed by the SVM trainer, and images with the highest relevance are presented to the user.

Active learning. In the case where the user is not satisfied with the current result, an active learner is called to select (unlabeled) images the user should label. In the following experiments, we use the method proposed in [3]. Once this labeling is performed, the classification step is repeated, and so on.

3. PROPOSED METHOD

The method we proposed is based on Multiple Kernel Learning (MKL), a strategy which has a great success these last years [6, 7, 8]. The aim of our method is to find a linear combination $K = \sum_{t=1} \beta_t k_t$ of base kernels k_t from a pool $\mathcal{K}$. More specifically, we aim at finding a sparse combination (where most of β_t equal zero), within a very large base kernel pool. The motivation of this choice is to be as generic as possible. Since we consider a very large base kernel pool, we can build many different final kernels. Furthermore, we consider "weak" base kernels, which focus on low-level features with very few parameters. For instance, if each base kernel is related to a visual codeword (a color, a keypoint, etc.), then we are able to consider a very large set of final kernels, each of these kernel possibilities being related to a different visual dictionary.

In the usual case where the base kernel pool is small, one has to work on "strong" base kernels, which must be built to fit the current database in order to be effective. When working on static databases and with some prior knowledge, this problem can be solved with

Algorithm 1 Fast Iterative Selection.

Given: kernel K, minor kernel k, centering matrix H, target kernel L.
Compute inner products:
$a = tr(HKHK)$; $b = tr(HkHk)$; $c = tr(HKHk)$
$\Delta = ab - c^2$
if $|\Delta| < \epsilon$ **then** return "no solution"
Compute weights which maximize $\mathcal{A}_L^H(\beta_K K + \beta_k k)$:
$w_K = tr(HKHL)$; $w_k = tr(HkHL)$
$\beta_K = \frac{bw_K - cw_k}{\Delta}$; $\beta_k = \frac{aw_k - cw_K}{\Delta}$
if $|\beta_K| < \epsilon$ **then** return "no solution"
Compute weight $\beta = \frac{\beta_k}{\beta_K}$
if $\beta < \epsilon$ **then** return "no solution"
Compute alignment $A = \frac{w_K + \beta w_k}{\sqrt{tr(HLHL)(a + 2c\beta + b\beta^2)}}$

Return weight β and alignment A

cross-validation techniques. However, when working on dynamic databases, where images regularly change, "strong" base kernels must be re-tuned to keep the effectiveness of the system.

3.1. Kernel-Target Alignment

In order to find the linear combination, we use the Kernel-Target Alignment :

$$\mathcal{A}_L(K) = \frac{\langle K, L \rangle_F}{||K||_F ||L||_F} \tag{1}$$

where K is the $n \times n$ kernel matrix to evaluate, L is the $n \times n$ target kernel matrix, $\langle K, L \rangle_F$ the Frobenius inner product, and $||K||_F$ the Frobenius norm.

A first target kernel is $\mathbf{y}\mathbf{y}^\top$, with $y_i \in \{-1, 1\}$ the label of document i. This choice is relevant for balanced training sets, where there is the same number of positive and negative labels. In other cases, such a kernel target focuses on the largest class. In the case of interactive retrieval, where the negative class is much larger than the positive one, optimization algorithms using this target kernel will learn more the negative class – the class the user is not interested in.

In order to deal with the unbalance of training data, one can use the target kernel $\tilde{\mathbf{y}}\tilde{\mathbf{y}}^\top$, with $\tilde{y}_i = 1/n^+$ if $y_i > 0$ and $\tilde{y}_i = -1/n^-$ if $y_i < 0$, where n^+ and n^- are the sizes of the positive and negative classes, respectively. An even more interesting approach we use in this paper is the centered alignment [9]:

$$\mathcal{A}_L^H(K) = \frac{\langle HKH, HLH \rangle_F}{||HKH||_F ||HLH||_F} \tag{2}$$

with $L_{ij} = 1$ if $y_i = y_j$, 0 otherwise, and $H = I - \frac{1}{n}\mathbf{1}\mathbf{1}^\top$, with I the identity matrix of size n, and $\mathbf{1}$ the column vector with all elements 1. Centered alignment deals with unbalanced data in the same way than target kernel $\tilde{\mathbf{y}}\tilde{\mathbf{y}}^\top$, but is also invariant to scale ($\mathcal{A}_L^H(\lambda K) = \mathcal{A}_L^H(K)$) and shift ($\mathcal{A}_L^H(K + \lambda \mathbf{1}\mathbf{1}^\top) = \mathcal{A}_L^H(K)$).

3.2. Fast Iterative Selection

In order to deal with computational constraints of interactive retrieval, we propose a fast algorithm to find a good linear combination of base kernels. For each feedback step in the retrieval process, we start with kernel $K_0 = 0$, and then select the base kernel $k_1^\star$ with weight $\beta_1^\star = 1$ that maximizes centered kernel alignment with current user labels. Then, we iteratively select and add new base kernels until no solution is found:

$$K_t^\star = K_{t-1} + \beta_t^\star k_t^\star \tag{3}$$

where

$$\beta_t^\star, k_t^\star = \underset{\beta > 0, k \in \mathcal{K}}{\operatorname{argmax}} \mathcal{A}_L^H(K_{t-1} + \beta k) \tag{4}$$

The optimization problem of Eq. (4) can be solved using linear algebra (cf. [9] for proof). Implementation details are presented in Algorithm 1.

This method can be seen as an approximation of boosting-based kernel learning methods such as [10]. The approximation we made in these processes is the removal of the training of base kernels (weak learner in boosting framework), because this is the most computational part of the learning process. This approximation changes the boosting assumption which states that, for any iteration of selection, it exists a weak learner (base kernel in our case) that enhances the optimization criterion. In our case, this assumption holds if the base kernel pool is infinite – which is obviously impossible in real applications. However, we can reach good performances if the pool is well made.

3.3. Base kernels

We propose to build base kernels that compare two images based on a low-level descriptor like color or texture.

Before any retrieval session, we extract and quantize the descriptors within an image using K-Means with Euclidean distance d. Let us note that quantization is performed independently on each image, as a trick to reduce the number of descriptors in each image. Then, we build for each image i a set P_i of features $\mathbf{p}_{ri} = (\mathbf{f}_{ri}, h_{ri}, \theta_{ri})$, where $\mathbf{f}_{ri}$ is the center of cluster r computed by K-Means, h_{ri} the number of vectors in cluster r, and θ_{ri} the distance of $\mathbf{f}_{ri}$ to the closest cluster center $r' \neq r$. These image feature sets $P_i = \{\mathbf{p}_{ri}\}_i$ are computed off-line.

Then, for each on-line retrieval session, we first build a new feature pool $\hat{\mathcal{P}} = \{\mathbf{p} \in P_i | y_i > 0\}$ using features contained in positive labelled images. Then, we define the corresponding base kernel pool $\mathcal{K} = \{k_{\hat{\mathbf{p}}} | \hat{\mathbf{p}} \in \hat{\mathcal{P}}\}$. With such a definition, there is a base kernel $k_{\hat{\mathbf{p}}}$ for each descriptor $\hat{\mathbf{p}}$, and a feature pool for each base kernel pool, and vise versa. In the following, we mix the two concepts.

Base kernels are defined as:

$$k_{\hat{\mathbf{p}}}(P_i, P_j) = \delta_{\chi^1}(e_{\hat{\mathbf{p}}}(P_i), e_{\hat{\mathbf{p}}}(P_j)) \tag{5}$$

with

$$\delta_{\chi^1}(x, y) = \begin{cases} 1 - \frac{|x-y|}{x+y} & \text{if } x > 0 \text{ and } y > 0 \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

$$e_{\hat{\mathbf{p}}}(P_i) = \begin{cases} h_{r^\star i} & \text{if } d(\hat{\mathbf{f}}, \mathbf{f}_{r^\star i}) \leq \hat{\theta} \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

$$r^\star = \underset{r}{\operatorname{argmin}}\, d(\hat{\mathbf{f}}, \mathbf{f}_{ri}) \tag{8}$$

$$\hat{\mathbf{p}} = (\hat{\mathbf{f}}, \hat{h}, \hat{\theta}) \tag{9}$$

Each of these base kernels can be seen as a triangle kernel with χ^1 distance computed on a single bin of a histogram. We first compute the descriptor $\mathbf{f}_{r^\star i}$ of image P_i the closest to the base kernel descriptor $\hat{\mathbf{f}}$ (Eq. (8)). Then, if this descriptor is in the hypersphere of center $\hat{\mathbf{f}}$ and radius $\hat{\theta}$, we return the number of pixels $h_{r^\star i}$ in image P_i (Eq. (7)). The same computation is performed on image P_j, and finally numbers of pixels in each image are compared (Eq. (6)).

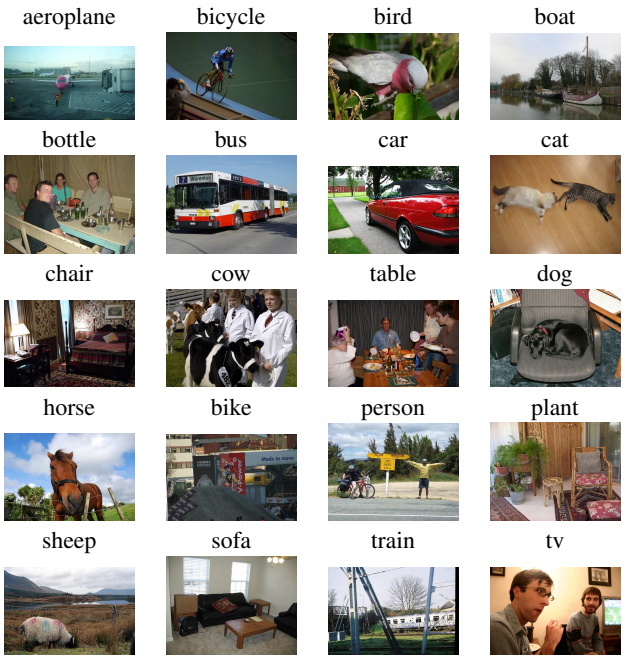

Fig. 1. Images from VOC 2007 database.

Category	rgSIFT/VLAD	Prop./1x1	Prop./1x1+1x3
aeroplane	25,6	24,8	**35,3**
bicycle	10,8	11,9	**13,6**
bird	12,1	11,1	**12,5**
boat	16,8	23,5	**25,9**
bottle	6,3	6,3	**8,2**
bus	9,0	9,6	**15,2**
car	28,8	34,3	**39,6**
cat	14,8	13,6	**15,1**
chair	**24,7**	15,5	19,0
cow	3,6	**5,9**	5,7
diningtable	12,6	9,9	**14,6**
dog	11,4	13,5	**13,8**
horse	20,7	25,0	**29,2**
motorbike	11,0	20,9	**23,6**
person	57,6	58,5	**61,4**
pottedplant	7,6	8,2	**10,5**
sheep	4,1	**5,7**	4,7
sofa	8,3	8,0	**10,0**
train	15,9	21,7	**31,2**
tvmonitor	9,3	12,5	**17,4**
all	15,6	17,0	**20,3**

Fig. 2. Average Precision(%) on the test set of VOC2007 database, for each category. Initialization with 1 image, 10 feedbacks, 5 labels per feedback.

4. EXPERIMENTS

We carried out experiments on VOC 2007 database which contains images belonging to 20 categories. It is split into 5,011 development images (train+val) and 4,952 test images. Images from this database are shown in Fig. 1. Let us note that we do not follow any of the PASCAL evaluation protocols since we are interested in interactive image retrieval.

4.1. Evaluation Protocol

We set up an evaluation protocol to estimate the average performance one can expect when starting an interactive retrieval session with a random image. This is performed by the simulation of retrieval sessions for each category, where the user labels the images selected by the active learning technique. A Precision/Recall curve is computed at each feedback step, and then averaged over all retrieval sessions for the same category. Then, the average quality of the ranking is computed with the usual criterion of Average Precision, which is used for example in TRECVID evaluation campaign. At last, in order to have a global quality measure of our system, we compute the Mean Average Precision (MAP) on all categories. Labelling is only performed on the development set, and performance is computed on the test set.

4.2. Results

Average Precision per category are displayed in Fig. 2 for 51 labels, and Mean Average Precision according to feedback steps in Fig. 3. In all the following experiments, we use a SVM classifier for the classification of the database, and the active learning method is the one of [3].

Baseline. We first show baseline results using a usual learning strategy with static dictionaries that is effective and robust to tuning. In order to find such a method, we tested many techniques proposed in the literature. Most of them use large dictionaries of descriptors (more than 1000) and then compute a vector representation of images using the dictionaries. For instance, Bags of Words computes a dictionary of 4000 keypoint descriptors, and then creates an histogram of visual words occurrences for each image. Our experiments showed that this method leads to very unstable results in an interactive retrieval context. For instance, running another time the dictionary computation can lead to very different results, even with advanced K-Means algorithms. Let us note that is specific to the few number of labels (about 50) we have in interactive retrieval context, and this high instability does not appear with very large training sets. With more recent methods, such as the ones based on locally linear coding [2], results are better but still unstable.

Among all methods we tested, we found an effective one in our interactive context which is based on small visual dictionaries. This method is called Vector of Locally Aggregated Descriptors (VLAD) [11], and is derived from the method based on bag of features and Fisher Kernels proposed in [1]. VLAD approach aggregates keypoint descriptors into a single vector in a more efficient way than bag of features. We integrated this aggregation scheme using keypoints detected with Harris-Laplace and rgSIFT descriptors. Note that we tested the same descriptors as the ones experimented in [12], and rgSIFT turned to be the most effective. We tested several dictionary sizes (from 8 to 64), and it appears that a dictionary of 18 visual words provides the best results. Let us note that we experimented several indexing and learning techniques from the literature, and select this method for its robustness to tuning. For instance, with the worst dictionary size, we get a MAP of 14.7%, which is close to the best result (15.6%). Furthermore, since we need few codewords, the computation of the dictionary is always good, on the contrary to other dictionary base techniques where two K-Means can lead to two different dictionaries with very different performances. In other words, this method is a good candidate for interactive retrieval in dynamic database since performances seem to be quite stable, whatever

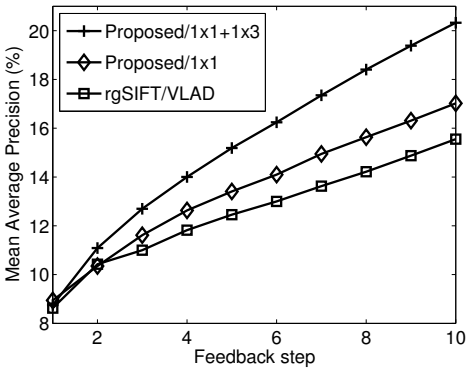

Fig. 3. Mean Average Precision(%) on the test set of VOC2007 database, according to each feedback step.

the changes made in the database.

About computational time, with an iCore7 processor and no shared cache between retrieval sessions, training requires less than 1ms, and classification of 5,000 images from 30ms (8 codewords) to 200ms (64 codewords).

Proposed method. We also show results with the proposed method using global descriptors (denoted as "Proposed/1x1" in Fig. 2 and 3). We used $L^\star a^\star b^\star$ colors (lab) and Quaternionic Wavelets coefficients (qw) for texture. For each image i, we extract lab and qw descriptors for each pixel, and then use K-Means to reduce their number to get sets P_i of R descriptors. We tune the value of R as a compromise between performance and computational time, knowing that fewer descriptors lead to lower performances (16.4% with $R = 32$ lab and qw), and more descriptors lead to higher performances (18.3% with $R = 256$ lab and qw). Using our method with these features, we get a MAP of 17.0% with 51 labels. This result is comparable to the baseline method (VLAD), and thus shows the ability of the method to learn effective kernels only using a few labels.

About computational time, with an iCore7 processor and no shared cache between retrieval sessions, training requires about 250ms, and classification of 5,000 images about 25ms. When compared to the baseline method, the training is much slower, but does not depends on the size of the database. However, the classification of images is always faster, which means that the method we propose is more interesting for large databases.

We also show results with the proposed method using features extracted in 3 horizontal parts of images (denoted as "Proposed/1x1+1x3" in Fig. 2 and 3). In that case, each part of images is used to build a different feature/base kernel pool. We extracted 21 lab and 21 qw descriptors in each part of image, and then get 8 different feature/base kernel pool (64 lab and 64 qw in whole image, 21 lab and 21 qw in upper part, 21 lab and 21 qw in middle part, and 21 lab and 21 qw in bottom part). With 25 positive labels (the usual case at the end of retrieval sessions), base kernel pools with 64 descriptors per image have 1600 elements, and base kernel pools with 21 descriptors per image have 525 elements. Furthermore, at the end of retrieval sessions, the method used to select from 27 to 38 base kernels from each base kernel pool. With this setup, we get a MAP of 20.3%, a good improvement that shows the ability of the method to benefit from more features.

5. CONCLUSION

In this paper, we introduced a method to learn a linear combination of base kernels in an effective and efficient way. The main advantage of this method is its high robustness to tuning, and thus its ability to handle interactive retrieval in dynamic databases. This method can be deployed and easily maintained on real image databases since no steady tuning is required to fit the current needs of users. Moreover, the classification of images is faster than comparable methods, since only relevant feature for the current labels are used for classification.

6. REFERENCES

[1] F. Perronnin and C. R. Dance, "Fisher kernels on visual vocabularies for image categorization," in *IEEE International Conference on Computer Vision and Pattern Recognition*, June 2007, pp. 1–8.

[2] Jingjun Wang, Jianchao Yang, Kai Yu, Fengjun Lv, Thomas Huang, and Yihong Gong, "Locality-constrained linear coding for image classification," in *IEEE International Conference on Computer Vision and Pattern Recognition*, 2010, pp. 3360–3367.

[3] P.H. Gosselin and M. Cord, "Precision-oriented active selection for interactive image retrieval," in *International Conference on Image Processing*, Atlanta, GA, USA, October 2006, pp. 3197–3200.

[4] S. Tong and D. Koller, "Support vector machine active learning with application to text classification," *International Journal on Machine Learning Research*, vol. 2, pp. 45–66, November 2001.

[5] P.H. Gosselin and M. Cord, "Active learning methods for interactive image retrieval," *IEEE Transactions on Image Processing*, vol. 17, no. 7, pp. 1200–1211, 2008.

[6] Francis R. Bach and Gert R. G. Lanckriet, "Multiple kernel learning, conic duality, and the smo algorithm," in *International Conference on Machine Learning*, 2004.

[7] A. Vedaldi, V. Gulshan, M. Varma, and A. Zisserman, "Multiple kernels for object detection," in *Proceedings of the International Conference on Computer Vision (ICCV)*, 2009.

[8] T. Joutou and K. Yanai, "A food image recognition system with multiple kernel learning," in *International Conference on Image Processing*, Cairo, Egypt, September 2009.

[9] M. Kawanabe, S. Nakajima, and A. Binder, "A procedure of adaptive kernel combination with kernel-target alignment for object classification," in *ACM International Conference on Image and Video Retrieval*, 2009.

[10] K. Crammer, J. Keshet, and Y. Singer, "Kernel design using boosting," in *Advances in Neural Information Processing Systems*. 2003, pp. 537–544, MIT Press.

[11] H. Jégou, M. Douze, C. Schmid, and P. Pérez, "Aggregating local descriptors into a compact image representation," in *IEEE International Conference on Computer Vision and Pattern Recognition*, June 2010, pp. 3304–3311.

[12] K. E. A. van de Sande, T. Gevers, and C. G. M. Snoek, "Evaluating color descriptors for object and scene recognition," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2010.