A Language Modeling Approach to Image Classification

Pierre Tirilly[1], Vincent Claveau[2], Patrick Gros[3]

[1]CNRS/IRISA, Rennes, France; [2]CNRS/IRISA, Rennes, France; [3]INRIA/IRISA, Rennes, France

E-mail: [1]ptirilly@irisa.fr, [2]vclaveau@irisa.fr, [3]pgros@inria.fr

Abstract: **Due to the recent and fast diffusion of new digital devices (digital cameras, camera cell phones, internet), the number and size of image databases is dramatically increasing. Managing such databases is an important issue, for professional databases (*e.g.* from photo agencies) as well as for personal collections. Image classification and retrieval are therefore becoming more and more challenging. Discriminant image descriptors and robust classifiers are needed to handle these tasks. Nowadays approaches generally rely on describing images as a set of elementary and independent image patches called visual words [15], then using a classical classifier such as Support Vector Machines [6]. In this paper, we propose a more precise description of images, called *visual sentences*, that includes simple spatial information between visual words. We then propose a classification technique based on language modeling. This classifier can exploit the spatial information of the visual sentences. Experiments on two classical datasets show that our classification method clearly outperforms the state-of-the-art SVM classifier.**

Keywords: image classification, language modeling, bag-of-visual words, support vector machines

1 INTRODUCTION

Image databases are getting bigger and more complex since numerical technologies (internet, digital cameras…) have become widespread. We therefore need new algorithms to manage these databases, and especially to perform the tasks of image classification and retrieval. These algorithms must be efficient, in order to handle huge databases with an acceptable computation time, as well as effective, in order to perform well even in the case of images with a varied content.

The aim of image classification, is, when given a set of categories (*e.g.* people, cars, horses…) and an image whose content is unknown, to tell automatically which category the image belongs to. Supervised machine learning techniques are usually used for this task: using sample images of each category (called the training data), such algorithms assign to an unknown image the category whose samples are the most similar to it. The standard scheme of supervised image classification systems is shown in Figure 1. Except from the need for training data, the critical parts of these systems are:

- image description: the descriptors must be reliable enough to differentiate images with different contents, and robust enough to allow the matching of images with similar content, in spite of changes in position, luminosity, rotation or scale.
- classifier: the classifier must generalize well from training data to unknown images and properly take into account the information available in the descriptor.

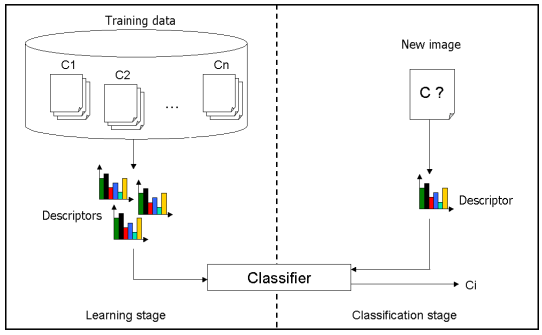

Figure 1: n-class image classification

1.1 Image descriptors

We usually classify image descriptors in two categories, global descriptors and local descriptors:

- global descriptors: they represent some aspects of the whole image, basically color (*e.g.* color histograms) or texture (*e.g.* Gabor filter banks). They can be computed very quickly but they only weakly describe images. First, since they are computed on the whole images, the description lacks precision and may change sharply due to changes in the background. Moreover, the visual features (color, texture) they rely on clearly seem more suited for the classification of some content (typically, natural objects: trees, tigers…) than others (basically, non-natural objects: a car can have any color). These drawbacks can be partially overcome by computing global descriptors on images patches.
- local descriptors: they represent local properties of images, such as angles. These descriptors are computed in two stages: first interest regions are detected (using detectors like Harris-affine or Hessian-affine), then they are described as a numerical vector (SIFT descriptor is the most commonly used). Such descriptors are very effective in image matching, because they describe very precise parts of the image

Corresponding author: Pierre Tirilly, CNRS/IRISA, Campus de Beaulieu, 35042 Rennes, France, +33299847233, ptirilly@irisa.fr

content. However, the high number of regions detected per image (a few hundred) and the high dimensional aspect of the descriptors (128 dimensions for a single SIFT vector) make the matching between images computationally very expensive.

Recent work deals with the limitations of these descriptors using an approach called *bag-of-visual words* or *bag-of-features*. The idea is to construct a global descriptor by counting the occurrences of local features in the image. Here is the process of bag-of-visual words image description, illustrated in Figure 2.

1) Detection and description of interest regions in a set of images.
2) Clustering of these descriptors. Each cluster represents a typical region of images, called a visual word. Figure 3 shows samples of such visual words.
3) Images can then be described as sets of visual words. By counting the occurrences of each word in an image, we obtain a high-dimensional sparse vector similar to the vector-space representation of textual documents traditionally used in text retrieval.

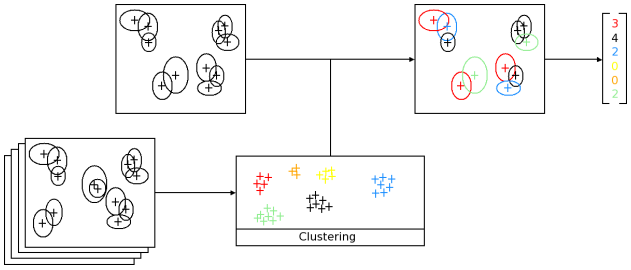

Figure 2: The bag-of-visual words construction process

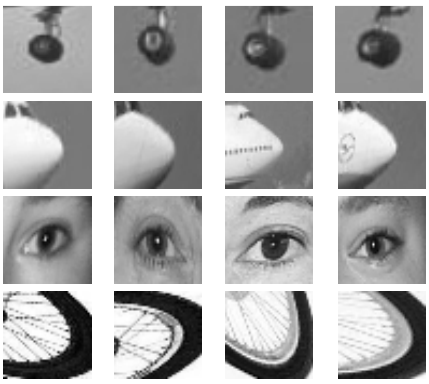

Figure 3: Examples of visual words. Patches on a given line are occurrences of the same visual word.

Since this descriptor is very similar to the one used in text retrieval, text retrieval techniques [14] can be adapted to bag-of-visual words. In particular, term weighting schemes (such as *tf.idf*) can improve the descriptor quality [15] and inverted file index allow to match documents faster in the case of retrieval tasks [15]. One can notice that, in this model, words are supposed to occur independently from each other. Whereas this hypothesis is acceptable in the case of text retrieval (generally, a single word represents one concept), it seems damaging in the case of image retrieval, since an object is described by a set of visual words with a given layout, not by a unique visual word.

1.2 Image classifiers

Image descriptors usually take the form of numerical vectors (color histograms, bag-of-visual words), so that classical machine learning tools can be used to perform supervised classification of these descriptors: Support Vector Machines (SVM) [6], random forests [2], Bayes classifiers [8]… To our knowledge, the use of customized SVM with a bag-of-words representation yields the best results for image classification [17].

In this work, we first propose a new image descriptor relying on visual words: visual sentences. They describe images as sequences of visual words, based on the layout of the words in the image. Therefore they allow to consider spatial information between words that were lost with the bag-of-visual words description. We then perform image classification using language models. Language modeling is commonly used in the field of text classification and retrieval. It models not only independent words occurring in texts, but also sequences of words. We can therefore use it to build a classifier which takes into account the spatial information contained in the visual sentences. In next section we show how to build visual sentences. Section 3 presents language modeling and its use in classification. Then in Section 4, we test our approach and compare it to a state-of-the-art classification method.

2 DESCRIBING PICTURES AS VISUAL SENTENCES

In this section, we introduce *visual sentences* as a new representation of images. Given a bag-of-words representation, our goal is to describe an image as a sequence of ordered symbols, which is a way to consider very simple spatial relations between words and use text-related techniques in the case of image retrieval and classification.

The visual sentence construction process is the following, as illustrated in Figure 4:

1) Construction of a visual vocabulary and representation of the pictures as sets of visual words.
2) Definition of an axis consistent with the position of the object in the picture.
3) Projection of the visual words from the image to the axis to obtain a sequence of visual words: a visual sentence.

The remaining of this section precisely describes each of stages 2 and 3.

2.1 Choosing an axis

Given several pictures of an object, we want to build visual sentences ordering the words in the same way from one image to another. Therefore, the axis we choose to project words on must have the two following properties:

- an orientation fitting the orientation of the object in the image, so that visual words are projected in the same order independently of the rotation or translation of the object in the image;
- a direction fitting the direction of the object, so the words can be read in the same order, whether the object is oriented from left to right or reversely.

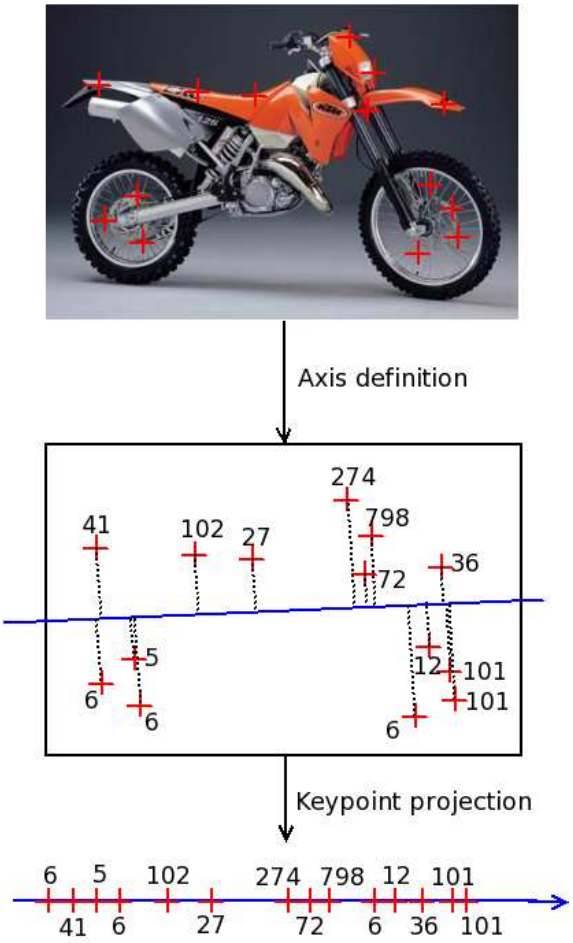

Figure 4: The visual sentence construction process

Since the regions detected using most detectors (in our case, the Hessian-affine detector) have certain repeatability and invariance properties [11], we can rely on the spatial distribution of the interest-points in the image to compute the axis. Principal Component Analysis (PCA) [9] gives us a solution, as, for a given set of points, it finds the direction vectors of the axes that best explain the distribution of the points. In our two-dimensional case, PCA gives us two axes. We choose to keep only the one with the most important contribution, *i.e.* the axis that best explains the distribution of the points. Then, given an object (a set of points), we can find an axis whose orientation and direction fit the ones of the object, whatever the position, orientation or direction changes of the object from one image to another. Moreover, the axis computation remains fast because PCA can be performed very efficiently for a limited set of points (about 1,000 per image) in a few dimensions (2 for interest-point coordinates). Figure 5 shows a few examples of axes obtained using PCA. We can note that this technique seems well suited for images containing one object. If there are two or more objects in the picture, results of the PCA can be biased by the relative positions of the objects. This point is discussed in the conclusion.

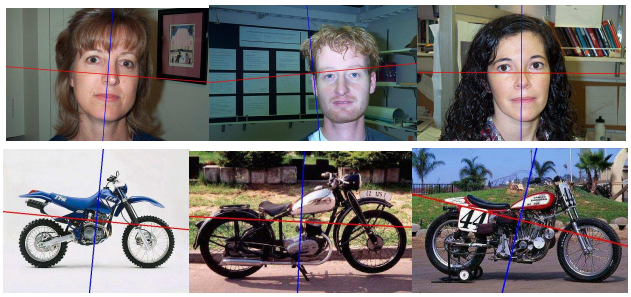

Figure 5: Examples of axes obtained by performing a PCA on the coordinates of the interest-points. The main axis is red, the second is blue.

We can also explore the possibility to use several axes, and thereby to produce several sentences per image, as it might give additional spatial information compared to a unique axis. In Section 4.2, we test several axis configurations:

- using the main axis given by the PCA.
- using the x-axis, since, in the datasets we use, objects are mostly aligned with this axis.
- using the two axes given by the PCA. They are orthogonal so they may bring complementary information about the spatial relationships between interest-points.
- using a set of axes generated by rotating the initial axis from 0 to 90 degrees. Rotations of more than 90 degrees should not be considered, since it would result in axes with contradictory reading directions.
- using a random axis, to compare with specific axes (PCA axis and x-axis).

2.2 Word projection

The interest-point detectors often detect redundant regions with similar shape, position and orientation. We eliminate them using a geometrical matching of the regions, as they add inconsistent spatial information (see [16] for details). We then simply project the remaining words on the axis (or axes) computed at the previous step, using orthogonal projection. We obtain one sentence for each axis we used.

3 CLASSIFICATION USING LANGUAGE MODELS

First used in the field of speech recognition, language modeling has become very popular in text classification [1, 3] and text retrieval [13] as it can model sequences of n words instead of independent words. A sequence of n words $w_1 w_2 \ldots w_n$ is called a n-gram, and a language model dealing with this kind of sequences is called a n-gram model.

A n-gram model is a probabilistic model that estimates for any word w_n the probability $Pr(w_n | w_1 \ldots w_{n-1})$ that w_n should occur in the language given the $n-1$ preceding words. Thus, it is able to model not only occurrences of

independent words (in the case of a unigram model ($n = 1$)), but also to deal with the fact that several words often occur together. This ability is very interesting in text analysis because words used together (*e.g.* White House) can have a different meaning from the same words used independently.

The probabilities are estimated in a statistical way, by counting the n-gram occurrences in a set of training documents. So, for a given language model Λ computed from a training set T, the probability that a n-gram $w_1w_2...\ w_n$ occurs is basically:

$$\Pr_\Lambda(w_n \mid w_1 w_2 ... w_{n-1}) = \frac{C(w_1 w_2 ... w_n)}{\sum_{w_i \in T} C(w_1 w_2 ... w_i)}$$

where $C(w_1w_2...\ w_n)$ is the number of occurrences of $w_1w_2...\ w_n$ in T.

Given a language model Λ, the probability of generating a document $d = w_1...w_k$ is:

$$\Pr_\Lambda(d) = \prod_{i=1}^{k} \Pr_\Lambda(w_i \mid w_1 ... w_{i-1})$$

It is approximated, using the n-grams, as:

$$\Pr_\Lambda(d) \approx \prod_{i=1}^{k} \Pr_\Lambda(w_i \mid w_{i-n+1} ... w_{i-1})$$

With this model, a n-gram that did not occur in the training data is assigned a null probability, as well as any new document containing it. Smoothing techniques overcome this problem: they assign a non-zero probability to unknown n-grams by reducing the probability mass assigned to known n-grams. More details about smoothing techniques are given in [4].

The use of LM in classification is quite simple: given a set of classes C and a training set of labelled documents T, we build, for each class $c \in C$, a language model Λ_c computed on the training subset $T_c = \{d \mid d \in T \wedge d \in c\}$. Then given an unknown document d_{unk}, its class $c(d_{unk})$ can be predicted as:

$$c(d_{unk}) = \arg\max_{c \in C}(\Pr_{\Lambda_c}(d_{unk}))$$

In the case of image classification, LM can be used exactly in the same way as in the text case, using an suitable image representation like the visual sentences we described in Section 2.

4 EXPERIMENTS

4.1 Global experimental settings

4.1.1 Datasets

We performed these experiments on two datasets, so we obtained results on datasets of different scale and difficulty. These datasets contain images that are widely used in image classification papers.

6 Caltech categories: we use 6 Caltech categories (available at *http://www.robots.ox.ac.uk/~vgg/data/data-cats.html*): car rears (1155 images), airplanes (1074), backgrounds (900), motorbikes (826), faces (450) and guitars (1030). The first five categories are widely used in image classification experiments, and we chose the last instead of the car side data commonly used, since it was not available anymore.

Caltech-101: to obtain more general results, we also test our method on a larger dataset, the Caltech-101 dataset [7], containing 8,697 images divided into 101 categories, each category containing from 31 to about 800 images.

4.1.2 Visual vocabulary

We use a Hessian-affine interest-point detector and a SIFT descriptor, as they offer good performances [10, 11] and are commonly used in bag-of-visual words approaches. We build our visual vocabulary using the hierarchical k-means algorithm proposed by Nister and Stewenius [12]. This approximate algorithm performs very fast clustering with an acceptable accuracy. The word number is set to 6,556 for the 6-category dataset, and to 61,687 for the Caltech-101 dataset. We chose these values as they provide optimal results in terms of recall and precision in an image retrieval context.

4.1.3 Baseline : Support Vector Machines

We chose SVM as a baseline because they provide state-of-the-art results and SVM software is easily available. We tested several kernels and weighting schemes, best results are obtained with linear kernel and *tf.idf* [14] weighting, like the results in [6]. We used two versions of the vectors, normalized or not, since the two versions give different results on some image categories. Non-normalized vectors are referred as SVM and normalized vectors are referred as SVM-N.

We carried out the SVM experiments ourselves instead of comparing them with available results as we must use the same parameters (detector, clustering algorithm, word number) to make a consistent comparison. To perform SVM experiments we use Joachims' multi-class SVM software (available at *http://svmlight.joachims.org*).

4.1.4 Performance measure

In the following experiments, we classically measured the performance of the system as the number of test images whose category is correctly predicted. The performance score S_c of classifier c is computed as a percentage:

$$S_c = \frac{|\{test\ images\ well\ classified\ by\ c\}|}{|\{test\ images\}|}$$

4.1.5 Language Models

For these experiments we used the *Carnegie Mellon University Statistical Language Modeling toolkit* [5], developed by Philip Clarkson and Ronald Rosenfeld (available at: *http://www.speech.cs.cmu.edu/SLM/toolkit.html*). We use the linear smoothing technique, as it performed better than the other smoothing techniques available in this software. We used n values from 2 to 4. Experiments showed that the use of greater values results in overfitting [16].

4.2 Choice of axis

4.2.1 Experimental settings

For this experiment, we used the 6-category dataset, divided into a training set (1200 images, 200 per category) and a test set (4215 images), and a 3-gram model. We tested several axis configurations to know which one is the most beneficial to image classification. We used:

- one axis obtained by PCA as explained in Section 2.1;
- two orthogonal axes obtained by PCA;
- ten axes obtained by successive rotations of 10 degrees of the main axis given by PCA, from 0 to 90 degrees;
- the x-axis;
- one random axis (the same for all images).

4.2.2 Results

Table 1: Classification performance against the number and nature of axis used for the visual sentence construction

	Training set	Test Set
PCA axis	66.75	66.90
2 axes	100	41.23
10 axes	100	38.20
x-axis	83.67	68.68
Random axis	66.75	65.67

Results are shown in Table 1. It presents the classification performance for all images in each set considered. The approach using the x-axis performs better than the PCA approach. Since the objects are all aligned with the x-axis in this dataset, the x-axis can be considered as the best axis: it is always well aligned with the objects, independently of the detected regions. The PCA axis is less robust because it is biased by the background clutter in some images. To be effective, the PCA approach would require to eliminate better background visual words. Although the random axis approach benefits from the fact that, in this dataset, objects have a similar position in the images, it yields slightly worse results than the PCA axis. It shows that the use of PCA is still a promising way to choose the axis: the PCA axis is better suited than a random axis to take account of spatial relations between visual words and it is adapted to varied positions of the objects in the images. The use of several axes results in overfitting: the more axes we use, the better the classification is on the training set and the worse it is on the testing set.

4.3 Classification performance

4.3.1 Experimental settings

We compared the classification performance of our technique with SVM performance on two datasets: our dataset containing 6 categories of Caltech images, and the Caltech-101 dataset. For the LM experiments, we always used the x-axis as it yields the best results and is a good reference axis for both datasets.

4.3.2 Results

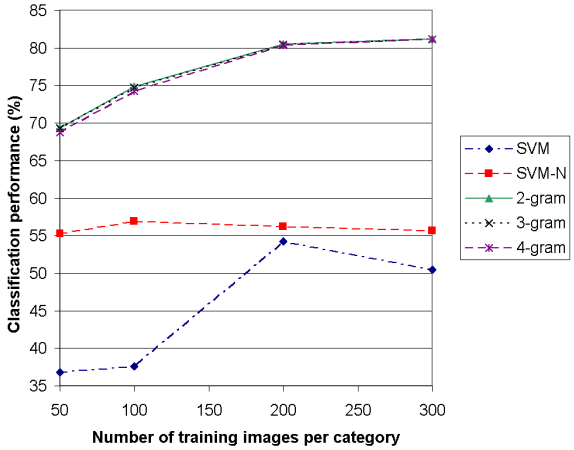

Figure 6: Classification performance on 6-category dataset

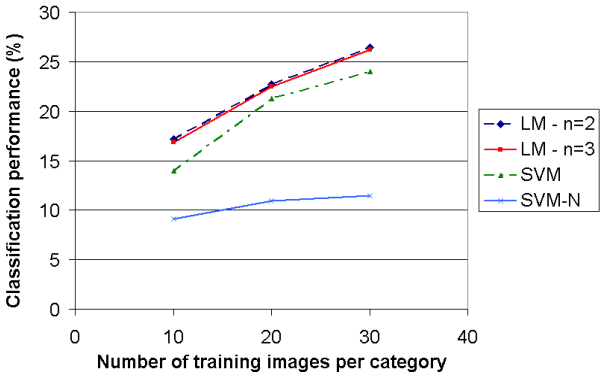

Figure 7: Classification performance on Caltech-101 dataset

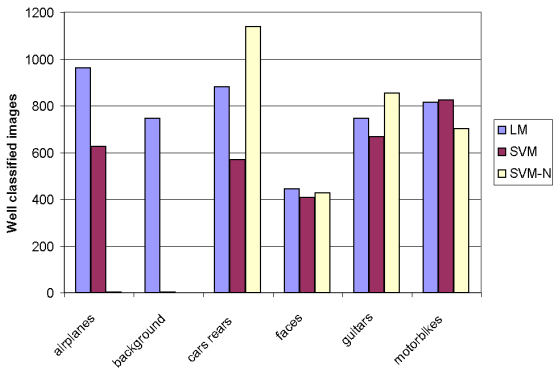

Figure 8: Classification performance on 6-category dataset with 200 training images per category

Results on the 6-category dataset are shown in Figure 6. Our LM approach clearly outperforms the SVM classifiers in terms of average classification performance. Figure 8 compares the performance of the best LM ($n = 2$) and SVM when using 200 training images per category. SVM approaches, and especially SVM-N, have very variable results from one category to another. SVM-N has very good results on car rears, but very negative results on backgrounds and airplanes. SVM is also very bad on backgrounds. One explanation is that much less interest-points are detected on backgrounds, with more variable visual words. Thus SVM cannot generalize well on this

category. On the contrary, LM classification gives steadier results, and best performs on 4 of the 6 categories. It yields much better results on backgrounds than SVM classifiers do. This can be explained by the ability of LM to take into account unseen words, and the fact that less words, so less n-grams, characterize backgrounds: since background images do not fit any n-grams of the other category models, this category is taken as default. LM shows the same behavior when using too long n-grams: most of the pictures are classified in one category. In particular, when $n = 9$ and $n = 10$, all pictures are classified as backgrounds.

On the Caltech-101 dataset, which presents a more difficult classification problem, LM-based classification yields also better results than any SVM-based classification (Figure 7).

Table 2: Mean execution time (in seconds) over 10 runs on the 6-category dataset, on a Linux PC with a 3 GHz Intel Xeon CPU and 8 GB RAM

	Training time	Classification time
LM $n = 1$	0.542	5.629
LM $n = 3$	0.973	8.019
SVM	7.293	17.94
SVM-N	5.728	18.18

Table 2 shows the execution time for each classifier. LM learning and classification are clearly faster than SVM ones. The greater n is, the more computation time is required because more n-grams have to be computed. The difference between LM and SVM is mostly due to I/O, since SVM deals with very high dimensional sparse vectors, whereas LM only uses sentences containing a few hundred words.

5 CONCLUSION

In this paper, we presented a new image classification scheme, based on the joint use of visual words and language models. The advantages of this technique are:

- The inclusion of spatial information between visual words, yielding better classification performance than the state-of-the-art SVM;
- Reduced learning and classification times.

However, our technique presents a few drawbacks:

- Finding an axis for any image is difficult, even if the x-axis might be suited in many cases;
- Finding the optimal parameters of the initial bag-of-words (detector, clustering algorithm, number of words…) is generally an awkward issue.

Future work will include improvements to our method. We will try to make the PCA more robust so that we can get a good axis for any image. We will also investigate the case of images containing several objects: in this case finding axes is difficult since PCA might get biased by the relative positions of the objects. We may overcome this limit by grouping the interest-points and get several axes per image. The groups can be based on geometrical properties of the interest-points (*e.g.* density-based) or on the cooccurrences of visual words in the whole collection (*e.g.* using Latent Semantic Analysis). We will also try to improve the standard language models, so we can take into account more complete spatial information.

Moreover, our work can be easily adapted to other tasks such as image retrieval or image annotation. With the current growth of image databases, such tasks are particularly interesting, in an amateur context (personal collections) as well as in a professional context (news photos, museums…). Finally, we will also use this method in the case of another rising media: video, as bag-of-visual words approaches have already shown promising results in this context [15].

References

[1] J. Bai and J.-Y. Nie. *Using language models for text classification.* In Proceedings of the Asia Information Retrieval Symposium, Beijing, China, Oct 2004.

[2] A. Bosch, A. Zisserman, and X. Munoz. *Image classification using random forests and ferns.* In Proceedings of ICCV, 2007.

[3] W. B. Cavnar and J. M. Trenkle. *N-gram-based text categorization.* In Proceedings of the Symposium on Document Analysis and Information Retrieval, pages 161–175, Las Vegas, US, 1994.

[4] S. F. Chen and J. Goodman. *An empirical study of smoothing techniques for language modeling.* Technical report, Cambridge, MA, August 1998.

[5] P. Clarkson and R. Rosenfeld. *Statistical language modeling using the CMU–Cambridge toolkit.* In Proceedings of the Eurospeech Conference, pages 2707–2710, Rhodes, Greece, 1997.

[6] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray. *Visual categorization with bags of keypoints.* In ECCV: Workshop on Statistical Learning in Computer Vision, Prague, Czech Republic, May 2004.

[7] L. Fei-Fei, R. Fergus, and P. Perona. *Learning generative visual models from few training examples: an incremental bayesian approach tested on 101 object categories.* In Proceedings of CVPR: Workshop on Generative-Model Based Vision, June-July 2004.

[8] D. Gokalp and S. Aksoy. *Scene classification using bag-of-regions representations.* In Proceedings of CVPR, pages 1–8, 2007.

[9] I. Jolliffe. *Principal Component Analysis.* Springer, 2002.

[10] K. Mikolajczyk and C. Schmid. *A performance evaluation of local descriptors.* IEEE Transactions on Pattern Analysis and Machine Intelligence, 27(10):1615–1630, 2005.

[11] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool. *A comparison of affine region detectors.* International Journal of Computer Vision, 65(1-2):43–72, 2005.

[12] D. Nister and H. Stewenius. *Scalable recognition with a vocabulary tree.* In Proceedings of CVPR, pages 2161–2168, Washington, DC, USA, 2006.

[13] J. M. Ponte and W. B. Croft. *A language modelling approach to information retrieval.* In ACM SIGIR Conference on Research and Development in Information Retrieval, pages 275–281, 1998.

[14] G. Salton and M. McGill. *Introduction to modern information retrieval.* McGraw-Hill, 1983.

[15] J. Sivic and A. Zisserman. *Video Google: A text retrieval approach to object matching in videos.* In Proceedings of ICCV, volume 2, pages 1470–1477, Nice, France, 2003.

[16] P. Tirilly, V. Claveau, and P. Gros. *A language modeling approach to bag-of-visual words image categorization.* In Proceedings of CIVR, Niagara Falls, Canada, 2008.

[17] J. Zhang, M. Marszalek, S. Lazebnik, and C. Schmid. *Local features and kernels for classification of texture and object categories: A comprehensive study.* International Journal of Computer Vision, 2007.