Classification of images based on Hidden Markov Models
Marc Mouret, Christine Solnon, Christian Wolf

To cite this version:
Marc Mouret, Christine Solnon, Christian Wolf. Classification of images based on Hidden Markov Models. IEEE Workshop on Content Based Multimedia Indexing, Jun 2009, Chania, Crète, Greece. pp.169-174, 10.1109/CBMI.2009.22 . hal-01437635

HAL Id: hal-01437635
https://hal.archives-ouvertes.fr/hal-01437635
Submitted on 24 Mar 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Abstract

We propose to use hidden Markov models (HMMs) to classify images. Images are modeled by extracting symbols corresponding to $3 \times 3$ binary neighborhoods of interest points, and by ordering these symbols by decreasing saliency order, thus obtaining strings of symbols. HMMs are learned from sets of strings modeling classes of images. The method has been tested on the SIMPLicity database and shows an improvement over competing approaches based on interest points. We also evaluate these approaches for classifying thumbnail images, i.e., low resolution images.

1. Introduction

Image classification, indexing and retrieval traditionally resort to features of different types, namely color [8], texture [24] and shape [1, 16] extracted globally or locally on interest points [13], regions or contour segments [6]. While object detection algorithms have become quite sophisticated integrating accurate shape models [9, 23, 7], the large intra-class variation in image classification problems prevents precise and detailed shape modeling. Not surprisingly, the most successful methods are based on bags of visual words [12, 18], i.e., representations which build a vocabulary of visual words and which do not take into account the spatial distribution of the features in the image. Introducing spatial dependencies has been attempted several times, however, the task is difficult: the dependencies need to be strong enough to improve classification, but not too strong in order to allow for the large intra-class variance of the dataset.

Hidden Markov models (HMMs) have already been used for object detection, for instance in [21]. However, mostly the very accurate spatial modeling of the object’s appearance does not allow to generalize across large shape variations, which is necessary for image classification. Their application in image classification is mostly restricted to regularizing segmentation algorithms, where the classification itself is handled by a different method [11], or in restricted domains as in document image classification, where blocks can be represented as nodes of a hidden Markov tree [4].

The work closest to our method might be [14], which classifies images using a part based conditional random field whose tree shape is calculated with a minimum spanning tree on the SIFT keypoint positions, and similar work presented in [5].

Our solution is based on the representation of images as strings of symbols: symbols correspond to $3 \times 3$ binary neighborhoods of interest points; they are ordered by decreasing saliency order. By integrating saliency order but ignoring spatial distribution of interest points, we enrich the discriminative power while keeping its variance. A hidden Markov model captures differences between classes and intends to model the large variations between image classes.

The paper is outlined as follows. Section 2 motivates the ordering of visual keywords into strings. Section 3 introduces the Markov model. Section 4 experimentally evaluates our approach on classical, i.e. rather high resolution images, and compares it with other approaches based on interest points. Section 5 gives experimental results on thumbnail images, i.e. rather low resolution images. Section 6 finally concludes.

2. From bags of visual words to strings of salient points

Bags of visual words have been first introduced in [12, 18] and different variants have been proposed in other works. The basic idea can be described as follows: the image is considered as a kind of visual document and treated as such, i.e. as a collection of “words”, which allows to apply many successful algorithms of the text retrieval community. The
vocabulary of visual words can be obtained by different means, for instance by clustering local features extracted from a database of training images. On the test image, the same features are extracted and for each feature the nearest codeword is found, giving rise to a set of words.

In spite of its simplicity, the approach is one of the most successful ones in the image classification community. However, a major drawback remains, namely the lack of interaction between the local features.

Ros et al. have proposed to model images by strings of discrete symbols [17]. Starting from a binarized image (see Fig. 1b), they extract interest points with the detector proposed by Bres and Jolion [2]. The location of the key points corresponds to high contrast at several levels of resolution, extracted from a contrast pyramid. This contrast measure is called saliency in the rest of this paper\(^1\). Fig. 1c shows an example of detected locations.

Each of these points is associated with a symbol which corresponds to its local \(3 \times 3\) binary neighborhood so that a number between 0 and \(2^9 - 1 = 511\) is assigned to each symbol (see Fig. 1d). Finally, these symbols are ordered into a string by decreasing contrast energy.

The main advantage of this method is also its main drawback: the ordering of the symbols by saliency introduces interactions between the features, which are not necessarily representative at a local level. For this reason, classical string edit distances as the one of Levenstein [10] are not relevant and give very disappointing classification results.

In [19] we have proposed and compared histogram-based distances for such strings of symbols. The first distance, called \(d_H\), does not consider the order of symbols in strings: it is defined as the sum, for every symbol \(s \in [0; 511]\), of the absolute difference of the number of occurrences of \(s\) in the two strings. The second distance, called \(d_{H_{\omega}}\) takes into account the order the symbols appear in strings by associating a weight \(\omega_k\) with every position \(k\) in strings. This weight is defined by \(\omega_k = l - k\), where \(l\) is the length of the string, so that symbols at the beginning of the string have higher weights than those at the end of the string. This weight is integrated in \(d_{H_{\omega}}\) by considering a weighted sum instead of the number of occurrence of each symbol \(s\). Experiments showed us that \(d_{H_{\omega}}\) significantly improves classification rates with respect to \(d_H\), thus bringing to the fore the interest of integrating saliency order. Indeed, saliency order avoids fixing the class models too much in terms of geometry and shape, allowing the discrimination-invariance trade-off to slightly favor invariance —needed in image classification problems as opposed to object detection problems.

3. Modeling images with HMMs

In order to exploit saliency ordering when classifying images, we propose to statistically model the symbol changes indirectly by considering them as observations associated to a chain of unknown (hidden) states, which can be described as a HMM [15]. In other words, the ordering of the symbols

\(^1\)Saliency is a term not rigorously defined. It is commonly associated to high and hopefully “relevant” local changes of the image content.
In our approach, each image class is characterized by its own behavior, therefore each class is described by its own model. The difference between two classes is thus of statistical nature: not the interaction between the observations themselves is described, but the interaction between a set of yet to learn hidden states and the production of the observations from these hidden states. Note that, as the hidden states are a number of discrete values only, where the state space size \( T \) is a user-defined parameter, their meaning is defined through their interactions, i.e. the values of the transition matrix \( A = a_{kl} \). The HMM learning algorithm tries to find the optimal description in terms of hidden states and state transition probabilities producing the observed symbols. The meaning of the hidden states themselves can stay unknown to the algorithm, as in similar work [5, 14].

The parameters for each class \( c \) —two matrices and a vector — can be denoted as a parameter vector \( \theta^c = [\pi_k^c, a_{kl}^c, b_k^c]^T \). The problem is thus twofold:

- in the training phase, learn the parameter vector \( \theta^c \) for each class \( c \);
- in the test phase, determine for each image modeled by a string of symbols \( o_t \), the most probable class \( c \), i.e.,

\[
\hat{c} = \arg \max_c p(o | \theta^c)
\]

We use the forward algorithm in order to determine the probability \( p(o | \theta^c) \) as well as the classical Baum-Welch variant of the Expectation-Maximization algorithm in order to learn the parameter vectors [15]. From the multiple variants learning the parameters from multiple observation sequences [3] we chose averaging over the parameters learned from single observation sequences.

Our general training method is supervised, since the class label for each image is considered as known and used during the training phase. However, for each image we resort to the training algorithm Baum-Welch which is considered as unsupervised, since the values of the different hidden states of the training image are not known. Indeed, as mentioned above, during the training phase we try to find the best state behavior for each class, i.e. the parameter vector \( \theta^c \) which best explains the observed symbols \( o_t \) extracted from the training images. This makes it possible to calculate the optimal state sequence \( s_t, i = 1..N \) for each training image using the Viterbi algorithm [20, 15]. It could be interesting to study these sequences in order to find out what kind of “hidden behavior” has been found by the training algorithms. However, the state sequences are not needed for classification.

4. Experimental results

We have performed experiments on the SIMPLIcity database [22] which contains 1000 images extracted from...
Table 1. Average classification rates when considering the 500, 1000, 2000 and 4000 most salient points. HMM(T) denotes HMMs with T states. KNN(d) denotes the k nearest neighbor approach with respect to distance d. GM(d) denotes the generalized median strings with respect to distance d.

<table>
<thead>
<tr>
<th></th>
<th>500</th>
<th>1000</th>
<th>2000</th>
<th>4000</th>
</tr>
</thead>
<tbody>
<tr>
<td>HMM(1)</td>
<td>63.1</td>
<td>63.2</td>
<td>62.7</td>
<td>62.8</td>
</tr>
<tr>
<td>HMM(2)</td>
<td>63.5</td>
<td>64.4</td>
<td><strong>68.1</strong></td>
<td>67.3</td>
</tr>
<tr>
<td>HMM(5)</td>
<td>63.4</td>
<td>64.9</td>
<td>67.1</td>
<td>70.0</td>
</tr>
<tr>
<td>HMM(10)</td>
<td><strong>64.5</strong></td>
<td>65.2</td>
<td>67.1</td>
<td>70.2</td>
</tr>
<tr>
<td>HMM(20)</td>
<td>62.6</td>
<td>65.0</td>
<td>66.6</td>
<td>70.1</td>
</tr>
<tr>
<td>HMM(50)</td>
<td>58.4</td>
<td>63.9</td>
<td>67.2</td>
<td><strong>70.6</strong></td>
</tr>
<tr>
<td>HMM(100)</td>
<td>51.8</td>
<td>60.4</td>
<td>66.1</td>
<td>70.3</td>
</tr>
<tr>
<td>KNN(dH)</td>
<td>63.2</td>
<td>64.3</td>
<td>64.0</td>
<td>62.8</td>
</tr>
<tr>
<td>KNN(dL)</td>
<td>63.0</td>
<td>66.3</td>
<td>67.6</td>
<td>66.2</td>
</tr>
<tr>
<td>GM(dH)</td>
<td>63.4</td>
<td>65.9</td>
<td>58.0</td>
<td>50.8</td>
</tr>
<tr>
<td>GM(dL)</td>
<td>61.6</td>
<td><strong>66.4</strong></td>
<td>65.9</td>
<td>60.8</td>
</tr>
</tbody>
</table>

Table 1. Average classification rates when considering the 500, 1000, 2000 and 4000 most salient points. HMM(T) denotes HMMs with T states. KNN(d) denotes the k nearest neighbor approach with respect to distance d. GM(d) denotes the generalized median strings with respect to distance d.

the well known commercial COREL database\(^2\). The database contains ten clusters representing semantic meaningful categories, i.e., “African people, “beaches”, “buildings”, “buses”, “dinosaurs”, “elephants”, “flowers”, “food”, “horses” and “mountains”. Needless to say that the categories are extremely heterogeneous in terms of signal contents, as illustrated in Fig. 3. There are 100 images per cluster.

Each image contains 384 × 256 pixels and is modeled by a string of symbols extracted as explained in Section 2. The original strings have 4000 symbols\(^3\). To study the influence of the length of the strings, we report experimental results obtained when limiting the number of symbols to different lengths ranging between 500 and 4000.

Table 1 compares classification rates obtained with three different approaches.

- **HMM(T)** is the approach based on hidden Markov models described in section 3 where T is the parameter which determines the number of states.

- **KNN(d)** is the k nearest neighbors approach: to classify an image, we compute the distance between this image and every image of a database, the classes of the k closest images determine the class. We set k to 5, which gives the best average results.

- **GM(d)** gives results obtained when characterizing each image class by the generalized median, i.e., the string which minimizes its distance to every string in the class, and classifying images with respect to the class of its closest generalized median.

Both for KNN and GM, we report results obtained with the two histogram-based distances d_H and d_L introduced in [19] and recalled in Section 2. The training set and the test set were kept separated by following a strict “leave-one-out” principle: the string which is classified is removed from its class before computing the HMM parameters or the generalized median of this class. So each string is tested only on parameters to whose learning it did not contribute.

Let us first compare results obtained with different values of the T parameter which determines the size of the state space of HMMs. Indeed, when T = 1, HMMs only learn the probability b_{it} = p(O_t = t | S_i = 1) of observing symbol t (with t ∈ [0; 511]) as all hidden states are always equal to 0. In this case, the symbol order is not taken into account at all. Increasing T allows to integrate information brought by the ordering by saliency. Table 1 shows that for shorter strings, HMMs often obtain better results with smaller numbers of hidden states whereas for strings of 4000 symbols, the best results are obtained with larger numbers of hidden states.

Note that linearly increasing the size of the state space T will quadratically increase the number of parameters in the matrix A, which causes the well-known associated problems: very quickly the requirements in terms of training data will be too high in order to reliably learn the parameters.

The result also shows us that the 500 first symbols are more or less all equally important so that the information related to the ordering should not be too much emphasized. However, classification rates are significantly improved when integrating information provided by other symbols, after the 500 first ones, as classification rates are improved when considering longer strings, provided that the order of symbols is actually taken into account.

The interest of integrating saliency ordering is also brought to the fore when comparing results obtained with d_H and d_L, both for KNN and GM. Indeed, for strings of 500 symbols, d_H obtains better results than d_L whereas, for strings of more than 500 symbols, d_L is significantly better than d_H.

The fact that HMM obtains better results than KNN and GM (except for strings of length 1000) shows that saliency order is better exploited with HMMs. Hence, HMMs are able to classify correctly more than 70% of the 1000 images of the simplicity database. Table 2 gives the corresponding confusion matrix. The best results are obtained on dinosaurs (100%), buses (92%) and flowers (88%), which contain rather specific images, whereas the worst results are

\(^2\)The SIMPLCity database can be downloaded on the James Z. Wang web site at http://wang.ist.psu.edu/~jwang/test1.tar.

\(^3\)Images with 384 × 256 often have less than 4000 salient points so that it is not reasonable to extract more than 4000 points. When there are less than 4000 salient points, strings are padded with a new extra symbol.
Figure 3. Some examples from the SIMPLIcity image database used in the experiments.

Table 2. Confusion matrix: line X column Y gives the number of images of class X that have been classified in class Y by HMM(50) when using 4000 points, where A=African people, B=buses, C=horses, D=dinosaurs, E=elephants, F=flowers, G=mountains, H=buildings, I=beaches, J=food.

Table 3. Average classification rates for thumbnail images respectively modeled by 500 and 1000 symbols.

5 Classification of thumbnail images

In this section, we evaluate our approach and compare it with those of [19] on thumbnail images, i.e., low resolution images such as those returned by search engines on the Web. Indeed, approaches designed for high resolution images, such as methods based on segmentation and/or local features with large support, can hardly be used to process those thumbnail images. Our goal here is to evaluate relevancy of approaches based on interest points, including our new HMM-based approach, for classifying these low resolution images.

To this aim, we have reduced all images of the SIMPLIcity database from their original size of $384 \times 256$ pixels to $128 \times 86$ pixels. For these thumbnail images, we have only extracted the 1000 first salient points.

Table 3 shows us that, for these thumbnail images, it is not really relevant to integrate saliency ordering. Indeed,
better results are obtained with smaller number of states for HMM, and with \( d_H \) for KNN and GM. However, we are still able to correctly classify 61.6% of the thumbnail images with HMM(1).

6 Conclusion

We have proposed to use hidden Markov models to classify images modeled by strings of interest points ordered with respect to saliency. Experimental results have shown that this approach allows one to improve classification rates by better integrating saliency ordering. Experimental results have also shown that approaches based on interest points may be used to classify thumbnail images, that have rather low resolution. However, in this case, saliency ordering is not really relevant.

References


