USING IMAGE SEGMENTATION AS A BASIS FOR CATEGORIZATION

Janez Brank
Department of Intelligent Systems
Jozef Stefan Institute
Jamova 39, 1000 Ljubljana, Slovenia
e-mail: janez.brank @ijs.si

ABSTRACT

Image categorization is the problem of classifying images into one or more of several possible categories or classes, which are defined in advance. Classifiers can be trained using machine learning algorithms, but existing machine learning algorithms cannot work with images directly. We consider a representation based on texture segmentation and a similarity measure which has been used successfully in the related area of image retrieval. A generalized kernel for use with the support vector machine (SVM) algorithm can be built from such a similarity measure. We compare this approach with a more straightforward representation based on auto-correlograms.

1 INTRODUCTION

Besides textual and relational data, people increasingly have to deal with pictorial data, or data in the form of images. Large *pictorial databases* are being produced as archives digitize their collections, and additionally the World Wide Web contains a huge number of images. Apart from purely technical problems of storing and processing such large amounts of data, the emergence of large collections of images opens the problems of enabling the users to make sense of this data and find what they need. *Image categorization* deals with one aspect of this problem: given a set of images and a set of predefined categories or classes, we assume that each image should belong to one or possibly several of these categories. For a large collection it would be impractical to have a human observer categorize all the images, so we want to be able to classify the images automatically after a small number of images has been classified manually to be used for training the automatical classifiers.

However, this view of image categorization as a machine learning task immediately opens up a new problem: existing machine learning algorithms generally cannot work with images directly. Instead, they often assume they will be dealing with instances described by vectors or tuples. We need to be able to represent images using structures of this kind to make use of existing machine learning algorithms.

We can build on existing work in *image retrieval*, which is a related area where the problem of representation has already been encountered. In image retrieval, the user poses a query to the system and the system should find images that are somehow relevant to the query. Thus a way of representing the query, a way of representing images, and a way of comparing a query and an image are needed. If textual descriptions of images can be easily obtained, querying by keywords is both technically feasible and often useful enough in practice. However, when such external information about images is not available, one needs to rely solely on what can be automatically extracted from the images themselves. The user's query is then often simply a request to look for images similar to a given query image (this approach is known as "querying by content").

In image categorization, if a new image is similar to training images from a particular category, it should probably itself belong to that category; in image retrieval, if an image from the database is similar to the query image, it should probably be shown to the user. Thus we see that both areas need a way of representing images and assessing similarity between them. Many image representations and similarity measures have been proposed in image retrieval, and we would like to assess some of them from the point of view of image categorization as well.

One popular class of image representations is based on simplifying the image by approximating the color of each pixel by the nearest color from a predefined and fixed color palette; this can also be seen as partitioning (or *quantizing*) the space of all possible colors. Some information is then recorded about the presence of each color on the image. When simply the proportion of the image covered by (the pixels of) that color is stored, the resulting description is called a *histogram* [SB91]. However, this disregards all spatial information (how the color is distributed around the image), and improved versions have been proposed. For example, *autocorrelo-grams* [HKM97] record probabilities that a randomly chosen pixel in the vicinity of a randomly chosen pixel of a given color will itself be of the same color. This retains information about the amount of a color present on the image, but also records something about the spatial ar-

rangement of each color. Still, all "global" representations of this type can be seen as somewhat rigid as they record a strictly fixed amount of data for each image.

Another, more sophisticated, class of image representations is based on *segmentation*, or dividing an image into a set of *regions* such that each region is roughly homogeneous in color and/or texture. Each image is then represented by a set of regions; each region is typically described by a short vector that is a by-product of the segmentation procedure (containing e.g. the average color of the region, information about texture, and so on), as well as by its location on the image (i.e. which parts of the image are covered by that region). In general, regions might overlap, and each region might itself be composed of several disjoint parts; this is not inherently problematic as they need not be shown to the user. Various segmentation algorithms have been proposed [NRS99], as well as measures of similarity between segmented images (such as IRM [LWW00]). Representations based on segmentation can adapt well to differences in complexity between images, and have been used successfully in image retrieval [NRS99, WLW00].

To use these representations for image categorization, one could use global representations (e.g. autocorrelograms) in combination with any of several machine learning algorithms (such as support vector machines); or use a segmentation-based similarity measure with an algorithm that allows an arbitrary similarity measure to be plugged into it (e.g. the nearest-neighbor method). However, it is less obvious how to use segmentation in combination with support vector machines, because the latter assume all instances to be described by vectors with the same number of components, while in the case of segmentation the description of each image has more structure than that, and the number of regions can also vary from image to image.

2 SUPPORT VECTOR MACHINES

Support Vector Machines (SVMs) [CV95] are a relatively recent family of machine learning algorithms that have been used successfully in many application domains. In the most elementary form of this method, we assume that each training example is a vector from some d-dimensional real space, and that there are exactly two classes, called positive and negative. (Several extensions to multiclass problems are possible, e.g. by training one classifier for each pair of classes [HL01].) We want to separate the positive vectors from the negative ones using a hyperplane such that all the training vectors lie on the correct side of the hyperplane and are as distant from it as possible. Maximizing this distance (known as the *margin*) from the plane to the nearest training example can be cast as an optimization problem, and it is also possible to look for tradeoffs in allowing some training instances to be misclassified if this leads to a wider margin on the other training instances.

It turns out that the optimization problem on which the SVM algorithm is based, as well as the resulting hyperplane, can be expressed so that the training vectors need never be accessed directly, as long as we are able to compute the dot product of any two vectors. Now suppose we used some mapping φ to map our original instances x_i into some other (possibly higher-dimensional) vector space F. Let $K(x_i, x_j) := \langle \varphi(x_i), \varphi(x_j) \rangle_F$ be a function that, given two instances x_i and x_j, computes the scalar product (in F) of their images under the mapping φ. It follows from the above that we could traing a hyperplane in F without ever working with the mapped vectors $\varphi(x_i)$ explicitly, as long as we are able to compute $K(x_i, x_j)$ for any two instances x_i and x_j. The function K defined in this way is known as a *kernel*. The importance of kernels arises from the fact that a hyperplane in F could correspond to some highly nonlinear separation surface in the original space; thus, kernels allow the SVM algorithm to induce nonlinear models.

A kernel corresponds to a scalar product in some vector space and can therefore in some sense be seen as a sort of similarity measure: the doc product of two vectors (if we fix their length) is greatest when they point in the same direction, and then decreases as the angle between them increases. However, the converse is not true, as not every similarity measure corresponds to a scalar product in some vector space. If we used a non-kernel similarity measure as if it were an actual kernel, we would no longer have guarantees that the SVM training algorithm would converge, and even if it converged there would be no theoretical grounds to expect the resulting classifier to have good performance.

3 GENERALIZED KERNELS

Generalized SVMs have been proposed by Mangasarian [Man00] to allow an arbitrary similarity function to be used in a way analogous to a kernel. Given a set of training examples x_i with their labels y_i ($y_i = +1$ for positive and -1 for negative examples), the original SVM algorithm would produce a classifier of the form $prediction(x) = \mathrm{sgn}[\langle w, \phi(x) \rangle_F + b]$, where w is the normal vector to the separating hyperplane (in the F-space), and b is a real number that indicates the position of the hyperplane. In addition, w would be of the form $\Sigma_i \ \alpha_i \ y_i \ \phi(x_i)$, where α_i would be scalars obtained during training to maximize the margin of the resulting hyperplane. Thus the classifier can also be expressed as $prediction(x) = \mathrm{sgn}[b + \Sigma_i \alpha_i y_i \langle x_i, x \rangle_F]$.

Now if some arbitrary function K were used instead of a proper kernel function, giving us a classifier of the form $\mathrm{sgn}[b + \Sigma_i \alpha_i y_i K(x_i, x)]$, this might still be a perfectly reasonable and useful classifier, but it wouldn't necessarily correspond to some hyperplane in some vector space F to which the instances x_i and x might have been mapped. Thus we couldn't obtain the α_i values using the criterion of maximizing the margin, because there wouldn't even be a hyperplane whose margin to maximize. Instead, [Man00]

proposes to minimize the value $\alpha^T H\alpha$ (subject to the same constraints as before, i.e. that our training instances should lie on the correct side of the separation surface) for some positive definite matrix H. In the simplest case, we would take $H = I$ and minimize $\Sigma_i \alpha_i^2$. This can be interpreted intuitively as looking for a separation surface that can be expressed in the simplest possible way, possibly with many α_i equal to 0 (i.e. without using the training example x_i in the description of the separating surface).

It can be shown that the formulation for $H = I$ is equivalent to mapping each instance x into the vector $(K(x, x_1), \ldots, K(x, x_n))$ of its similarities (as measured by K) to all the training instances $x_1, \ldots, x_n$, and then using an ordinary linear support vector machine over this new representation. For the problem of image categorization, this amounts to the intuitively appealing suggestion that two images should be treated as similar if they exhibit a similar pattern of similarities to known training images.

4 REGION CLUSTERING

In this section we consider another approach to using segmentation-based representations for image categorization. Each image has its own set of regions and regions belonging to different sets are in a sense quite independent of each other. This leads to the need for special similarity measures that compare two images by considering all pairs of regions, and aggregating the similarities of regions into a measure of similarity between the images. As an alternative, we propose to bring the region-based representations of images to a "common denominator" by clustering the descriptions of all the regions of all the training images. An image would then be described by recording, for each cluster of regions, what proportion of the area of this image is covered by regions of this cluster. If there are d region clusters, each image would now be represented by a d-dimensional real vector (with possibly many zero-value components). With all images represented in this same d-dimensional space, we can then use the ordinary linear support vector machine to train classifiers.

5 EXPERIMENTAL EVALUATION

To compare the approaches described in the previous sections, we conducted experiments on the *misc* database, which is publicly available (http://www-db.stanford.edu/IMAGE/) and has already been used in image retrieval literature [WWFW97, NRS99], as well as in our earlier work on image categorization [Bra01]. We selected 1172 images from the database and manually assigned each of them to one of 14 categories (butterflies, US flag, sunsets, autumn, flowers, planets, satellite images of Earth, cars, mountains, clouds, sea, surfboards, sailboats, prairie animals). Categories vary in size and difficulty (some have characteristic and easily recognizable color distributions, while some categories are quite similar in this respect and would therefore be more difficult to distinguish, e.g. sea and clouds with lots of blue and white pixels). To train the SVM classifiers, we used the LibSvm [CL01] program, which has the advantage of natively supporting multiclass problems (it uses the all-pairs approach to convert a multiclass problem to several two-class problems).

We compared the following approaches to image categorization:

1. Images are represented in the HSV (hue, saturation, value) color space, which is quantized into 256 colors (the H axis is split into 16 and the S and V axes into 4 ranges). Each image is then described by an autocorrelogram in the resulting quantized color space. The autocorrelograms are 1024-dimensional vectors and are used as input for linear SVM.

2. Images are segmented into regions using the segmentation algorithm from WALRUS [NRS99]. The IRM similarity metric [LWW00] is then used to construct a generalized kernel as described in Section 3 above. In other words, each image is represented by a vector of its IRM similarities to all the training images; these vectors are then used as input for linear SVM.

3. Images are segmented as in the previous paragraph. Each region is described by a short (12-dimensional) vector, which is a by-product of the segmentation algorithm. The vectors resulting from all the regions of all the training images are then clustered (here we use the same algorithm, BIRCH [ZRL96], that is also used by WALRUS during segmentation). An image is then described by a sparse vector specifying what proportion of the area of the image is covered by regions from each region cluster. Depending on the parameters of the segmentation, the average number of regions per image might vary from less than ten to more than a hundred; then, depending on the parameters of the clustering, the number of region clusters (and hence the dimensionality of the space in which our images are now represented) is usually on the order of a few hundred. Once images are represented in this way, linear SVM can be used to train classifiers for them.

For the sake of comparison, we also report the performance of the nearest neighbor method with the IRM similarity metric (that is, each image is predicted to belong to the same class as the most similar training image). All performance values reported here are averages (and standard errors) based on tenfold stratified cross-validation.

Method	Classification accuracy
Autocorrelograms	80.2 % ± 1.3 %
Generalized kernels	79.0 % ± 1.3 %
Region clustering	70.0 % ± 1.6 %
Nearest neighbors + IRM	69.1 % ± 1.3 %

As expected, the nearest-neighbor method is in general less successful than the approaches based on SVM. However, it turns out that the two segmentation-based approaches do not outperform the representation based on autocorrelograms. The performance of the generalized

kernel method is not significantly different (using a paired t-test) from that of autocorrelograms, and the generalized kernel method has the additional disadvantage of much greater computational.

In addition, the performance of the region clustering approach is remarkably poor. A closer examination suggests that the partitioning of regions into region clusters is problematic and unstable. For example, if the centroid of each cluster is recorded and then all regions are distributed to the cluster with the nearest centroid, most of the regions will tend to move to a different cluster than they were originally attached to. This means that two otherwise similar regions might fall into different clusters by pure chance, and the similarity between their images would thus go unnoticed. The authors of the BIRCH clustering algorithm were aware of the possibility of such problems, and proposed several redistribution passes where the regions are redistributed to the nearest centroids, but in our experiments this did not lead to a really stable partition even after five or ten such passes.

An alternative way of making use of the region clustering approach might be to include the test images in the region clustering phase. This really amounts to a form of transduction, i.e. using test data as if it was simply additional unlabeled training data. It ensures that both the training images and the test images are really being represented in a space that treats both groups of images equally. In this setting, the performance of the region clustering increases considerably, and it achieves an accuracy of 86.4 % ± 1.0 %. However, for the comparison with other methods to be fair, transduction should also be included in the SVM learning process. Since LibSvm does not support tranduction, we used the SvmLight program [Joa99a] for these experiments; it implements Joachims' transduction SVM algorithm [Joa99b]. With transduction SVM, region clustering achieves an average accuracy of 91.9 % ± 1.0 %, while autocorrelograms achieve an accuracy of 90.7 % ± 1.1 %. Although this difference is not really significant from a practical point of view (a t-test shows that it is statistically significant at a confidence level of 0.945, slightly below the usual 0.95), it suggests that the region clustering approach does have at least some potential to be useful.

6 CONCLUSIONS AND FUTURE WORK

Our experiments show that it is difficult to use segmentation-based image representation methods in image categorization. Relatively complex ways of using information obtained from segmentation, such as the generalized kernel approach and (to a lesser extent) the region clustering approach, have been found able to compete with a simpler and more straightforward approach such as autocorrelograms but not to significantly outperform it.

We nonetheless believe that there must be ways of using segmentation profitably for image categorization, just as it is used in image retrieval, and that this is still an interesting topic for future work. In particular, it would be interesting to further explore the influence of the clustering algorithm used in the region clustering approach, and to look for more stable clustering algorithms that would allow the region clustering approach to perform better in the inductive in additional to the transductive setting. In addition, as segmentation is a relatively complex task, and segmentation algorithms usually depend on several parameters, it would be interesting to explore the influence of these various parameters on the segmentation (and consequently on image categorization) in a more systematic way.

Additionally, the methods considered here should be tested on other datasets, as (given that widely different methods achieve highly similar accuracy values on the present dataset) it is perhaps simply unrealistic to expect better performance on the present dataset, as the categories have an essentially "semantic" motivation that the current image representation methods simply cannot capture.

References

[Bra01] J. Brank: *Machine learning on images* (in Slovenian). Proc. IS 2001, Ljubljana, 2001, pp. 152–55.

[CL01] C.-C. Chang, C.-J. Lin: *LibSVM: a library for support vector machines* (version 2.3). Dept. of Comp. Sci. and Inf. Eng., Nat'l. Taiwan University, April 2001.

[CV95] C. Cortes, V. Vapnik: *Support-vector networks.* Machine Learning, 20(3):273–297, September 1995.

[HKM97] J. Huang, S. R. Kumar, M. Mitra: *Combining supervised learning with color correlograms for content-based image retrieval*. Proc. 5th ACM Multimedia Conf., Seattle, USA, 1997, pp. 325–334.

[HL01] C.-W. Hsu, C.-J. Lin: *A comparison of methods for multi-class support vector machines*. Dept. of Comp. Sci. and Inf. Eng., Nat'l Taiwan University, April 2001.

[Joa99a] T. Joachims: *Making large-scale SVM learning practical.* In: B. Schölkopf et al. (eds.), Advances in Kernel Methods. MIT Press, 1999, pp. 169–184.

[Joa99b] T. Joachims: *Transductive inference for text classification using support vector machines.* Proc. 16th ICML, Bled, Slovenia, 1999, pp. 200–209.

[LWW00] J. Li, J. Z. Wang, G. Wiederhold: *IRM: Integrated region matching for image retrieval.* Proc. 8th ACM Multimedia Conf., Los Angeles, 2000, pp. 147–156.

[Man00] O. L. Mangasarian: *Generalized support vector machines.* In: A. J. Smola et al. (eds.), Advances in Large Margin Classifiers, MIT Press, 2000, pp. 135–146.

[NRS99] A. Natsev, R. Rastogi, K. Shim: *WALRUS: a similarity retrieval algorithm for large databases*. Proc. ACM SIGMOD, 1999, pp. 395–406.

[SB91] M. J. Swain, D. H. Ballard: *Color indexing*. Int. Journal of Computer Vision, 7(1):11–32, Nov. 1991

[WLW00] J. Z. Wang, J. Li, G. Wiederhold: *SIMPLIcity: Semantics-sensitive integrated matching for picture libraries.* Advances in Visual Inf. Systems, 4th Int. Conf., 2000, pp. 360–371.

[WWFW97] J. Z. Wang, G. Wiederhold, O. Firschein, S. X. Wei: *Content-based image indexing using Daubechies' wavelets.* Int. Journal of Dig. Lib., 1(4):311–328, December 1997.

[ZRL96] T. Zhang, R. Ramakrishnan, M. Livny: *BIRCH: An efficient data clustering method for very large databases*. Proc. ACM SIGMOD, 1996. pp. 103–114.