Unsupervised Clustering of Images using their Joint Segmentation

Yevgeny Seldin Sonia Starik Michael Werman

School of Computer Science and Engineering
The Hebrew University of Jerusalem, Israel
E-mail: {seldin,starik,werman}@cs.huji.ac.il

Abstract

We present a method for unsupervised content based classification of images. The idea is to first segment the images using centroid models common to all the images in the set and then, through drawing an analogy between models/images and words/documents, to apply algorithms from the field of unsupervised document classification to cluster the images. The first step may be regarded as unsupervised feature selection while the second may be regarded as unsupervised classification of images based on the selected features.

We regard our image set as a mixture of textures. The centroid models of the mixture representing the textures are based on histograms of marginal distributions of wavelet coefficients calculated on image subwindows. The models are used in our algorithm (which is analogous to the work of Hofmann, Puzicha and Buhmann [HPB98]) to jointly *segment all the images in the input set. Such joint segmentation enables us to link between multiple appearances of the same texture in different images. We finally use the sequential Information Bottleneck algorithm of Slonim, Friedman and Tishby [SFT02] to cluster the images based on the result of the segmentation. In general, due to the modularity of the approach each of the three components of the presented method (local image modeling, segmentation and classification) can be substituted by alternative algorithms satisfying mild conditions.*

The method is applied to nature views classification and painting categorization by painter's drawing style. The method is shown to be superior to image

classification algorithms that regard each image as a single model. We see our current work as opening a new perspective on high level unsupervised data analysis.

1. Introduction

Clustering a set of images into meaningful classes has many applications in the organization of image data bases and the design of their human interface. [RHC99, MK01, WWFW97, SMM, GGG02] are only a few reviews and some of the recent works in the field. Image clustering can also be used in order to segment a movie, for example [GFT98], and to facilitate image data mining, for example searching for interesting partitions of medical images [THI+00]. [BDF02] do unsupervised clustering using extra information.

In this paper we treat the problem of unsupervised clustering of an image set into clusters of similar images, where we are only given the images. We treat the images as (soft) mixtures of textures. Since the same texture may be present in different images we build a *common* mixture model for the whole set. This is done by *joint* segmentation of the images. The components of the mixture (*centroid models*) are then regarded as a "dictionary" of image segments (segments with similar textures are associated with the same component). Co-occurences of the centroid models and images are finally used in order to cluster the images (analogously to using word and document co-occurences in documents clustering). See Fig. 1 for a schematic illustration of our algorithm.

As a common approach for texture modeling we choose our centroid model to be a weighted set of histograms of wavelet subband coefficients of image sub-windows. We have also tried using color histograms of the image sub-windows, but texture approach based on wavelet statistics appear to be more powerful.

We use the deterministic annealing (DA) framework (see [Ros98]) to get a top-down hierarchy of segmentations at increasing levels of resolution. We also present a modification of the DA framework we call *forced hierarchy* to decrease the computation time.

In the last step we treat centroids as words and images as documents and use the sequential Information Bottleneck algorithm [SFT02] to obtain the classification.

It should be noted that while the idea of drawing a parallel between word counts in a document and model (feature) probability integral over the image already appeared in supervised classification literature [Ker], we see our current work as a new point of view on unsupervised data classification. Namely we do

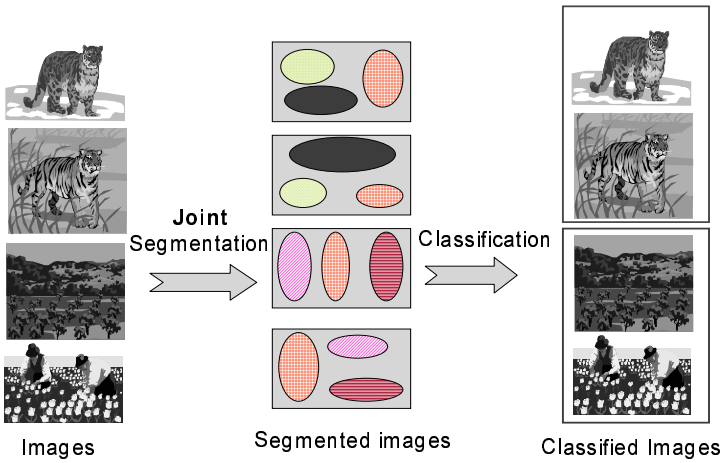


Figure 1: **General scheme of our algorithm.** The algorithm is built up of two steps. The first one is joint segmentation of the input images. As a result of the segmentation we get a "class map" representation for each input image - the "Segmented images" on the illustration. In this representation each image segment is labeled with a corresponding label, while the labeling set is common to all the images. On the illustration we express this idea by marking the image segments belonging to one cluster with the same color and filling texture. (The segmentation is soft - hard partition is shown for illustrative purposes only.) Then the second step is to classify the images using co-occurrence statistics of the images and the segment classes composing them.

an *unsupervised feature selection* and then perform unsupervised classification of images based on the features found.

Two papers should be mentioned in the context of our work. [HPB98] give a very similar image segmentation algorithm. The major difference which is important to us is that we segment all the images *jointly* while [HPB98] deal with segmentation of a single image.

[GGG02] suggest to cluster the images by first segmenting each image *separately* into Gaussian mixture of pixel color and location values and use agglomerative Information Bottleneck to cluster the mixtures. Comparing our and their approaches, our *joint* segmentation provides a much more general view on the data. Also, wavelet based models used for segmentation are much more powerful and location independent compared to Gaussian mixtures.

The approach suggested by us is very general and may be applied not only

to image classification, but also in unsupervised analysis of audio signals, protein sequences, spike trains and many other types of data, while using appropriate algorithms and data structures for their segmentation.

The paper is organized as follows: in Sec. 2 we describe how to build parametric models for image sub-windows using the wavelet coefficients statistics or color features, and how to build the centroid models. In Sec. 3 we segment the images using those models and obtain a (soft) segmentation of the images into a small number of centroid models, common to all the images in the set. Finally, in Sec. 4 we use the obtained segmentation for image classification analogously to classification of documents based on the statistics of their words appearances. Experimental results in Sec. 5.3 and discussion in Sec. 6 summarize our work.

2. Texture Based Image Modeling

We start with a description of our parametric probabilistic model for image sites. Being the basic building-block of the algorithm, the correct choice of the model is of crucial importance for the final success of the whole process. In the current work we choose our image sites as square subimages - *windows* - of a predefined size, and we use a *texture approach* to parametrically model them - we model each window as if it was a homogeneous texture sampled i.i.d. from a single distribution and an image as a collection of those textures.

To model a single window, we use a common approach that texture can be identified by the energy distribution in its frequency domain, by modeling the marginal densities of its wavelet subband coefficients. Such an approach was successfully used by [DV00] in image retrieval applications. In our algorithm, we characterize a texture as a set of marginal histograms of its wavelet subband coefficients. The number of bins in a histogram was chosen to be the square root of the total number of coefficients at the corresponding subband, as an optimal compromise between distribution resolution and statistical significance of the empirical counts. In order to assign approximately the same number of samples to each bin of the histogram, we make a coarse estimation of the distribution of coefficients in this subband as a Gaussian function, by fitting the parameters on the whole data set, and use the inverse Gaussian distribution to construct the histogram bins. Although the distribution is more likely to be a Generalized Gaussian density (as was shown by [DV00]), this coarse estimation by simple Gaussian is sufficient. The conventional pyramid wavelet decomposition is then performed with L levels (usually $L = 3$) with one of the known wavelet filters (we used Daubechies, reverse biorthogonal and symmetric wavelets), and one histogram for each subband

is computed. We normalize the histograms to obtain probability distributions and take the resulting set of the distributions to be our parametric model for the window. By moving to the space of wavelet histograms, the number of parameters characterizing the texture image is reduced to about the square root of the image area, and we also profit from the statistical nature of such model.

As a similarity measure between distributions p and q we use Kullback-Leibler divergence, which is a natural measure of distance between probability distributions and is defined as

$$D_{KL}(p||q) = \sum_x p(x) \log \frac{p(x)}{q(x)}$$

(see [CT91]). To compute the distance from a window model H to a centroid model M we compute pairwise $D_{KL}(H_l||M_l)$ for each subband l and take a weighted sum as the total distance. Since the number of coefficients decreases by a factor of 4 from level to level and due to the fact that there is more variability at higher resolutions, we give an accordingly decreasing weight to the distances at lower resolutions.

To build a centroid for a weighted set of window models, we build an average model as a weighted sum:

$$M_l^{average} = \frac{1}{\sum_k w_k} \sum_k w_k \cdot H_{k,l}$$

where $H_{k,l}$ is a histogram of subband l of window model k and w_k is the weight of the window. The average model computed this way minimizes the weighted sum of distances of the windows to the centroid: $\min_M \sum_k D_{KL}(H_k||M)$ (see [HPB98]). The centroid model has exactly the same parametric structure as the window models.

In the same manner we can use other features, such as image color, or a weighted combination of two or more types of wavelet filters and color histograms.

In our experiments, when we used color model, we took a histogram of the hue component of image color space.

3. Joint Image Segmentation Algorithm

Our primary goal is to represent each image in the input set as a *soft* mixture of a small number of textures *common to all the images in the set*. Such a representation will help us later to link between appearences of the same textures in different

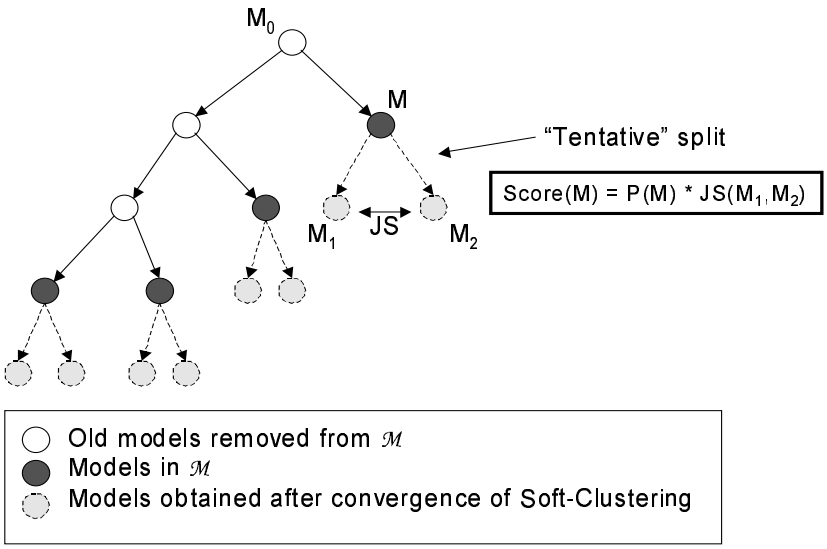


Figure 2: **Forced hierarchy framework illustration.** We start from the all-data-average model M_0. We then recursively choose the model having maximal score from $\mathcal{M}$ and replace it with the two models obtained at its tentative split. The models obtained at the tentative split are also used to calculate the score of their parent model.

images. Segmentation of images is done in a top-down hierarchical manner, while we work with all the input images simultaneously. The algorithm integrates the ideas developed in [SBT01] for sequence segmentation with texture segmentation algorithm of [HPB98]. We also present a novel approach to computations in deterministic annealing (DA) framework (see [Ros98]) we call *forced hierarchy*.

Our segmentation works on a shifted grid. An image is divided by overlaping grid windows into small patches which determine the resolution of our segmentation. The dependence between the overlapping grid windows $\{z_t\}$ is introduced through the definition of a distance measure between a window z_t and a model M_j, which is defined to be a weighted average of the D_{KL} distances over all the windows overlapping with z_t. The weight is taken to be proportional to the area of the overlap.

The goal of the segmentation process is to find K segment models $\{M_j\}_{j=1}^{K} = \mathcal{M}$, so that the average in-segment distortion

$$\langle d \rangle = \frac{1}{N} \sum_i \sum_j P(M_j|z_i) \cdot d(M_j, z_i)$$

Forced Hierarchy Segmentation($\{H(z_i)\}_i$, β_0, β_{step})

Initialize:

$M_0 = \frac{1}{N}\sum_i H(z_i)$, $\mathcal{M} = \{M_0\}$, $\beta_{M_0} = \beta_0$, $P(M_0) = 1$, $P(M_0|z_i) = 1$
Find-Score(M_0, β_{M_0})

While the desired resolution of the segmentation is not reached:

1. Choose M having maximal score out of $\mathcal{M}$
2. For $\{M_1, M_2\}$ - used for the calculation of the score of M:
   Find-Score(M_1, β_{M_1}), Find-Score(M_2, β_{M_2}).
3. Replace M in $\mathcal{M}$ with M_1 and M_2.

Find-Score(M, β)

1. $\{M_1, M_2\} = \text{Split}(M)$, $P(M_1) = \frac{1}{2}P(M)$, $P(M_2) = \frac{1}{2}P(M)$
2. Soft-Clustering(M_1, M_2, $P(M_1)$, $P(M_2)$, β, $P(M|z_i)$)
3. if $JS(M_1, M_2) < \frac{1}{\beta}$ then
   $\beta = \beta \cdot \beta_{step}$, return to 1.
   else
   $\beta_{M_1} = \beta$, $\beta_{M_2} = \beta$, Score$(M) = P(M) \cdot JS(M_1, M_2)$, exit function.

Figure 3: **Forced Hierarchy image segmentation algorithm pseudo code.** See text for explanation.

will be minimal (N is the total number of windows in all the images). We approach this problem in the DA framework (see [Ros98]) which essentially uses EM to estimate the segmentation at increasing levels of resolution. This framework allows us to avoid many local minima as well as to get a hierarchy of segmentation solutions. We apply the forced hierarchy modification of the framework to achieve better computational performance.

The algorithm starts from $\mathcal{M}$ consisting of a single average model M_0 ($M_0 =$

$\frac{1}{N}\sum H(z_i)$, where $H(z_i)$ stays for the set of histograms of wavelet coefficients of a window z_i). We then repeatedly choose the largest and the most heterogeneous model out of $\mathcal{M}$ and divide the corresponding segment (which is spread over the input images) into two. We continue to refine our partition until a maximal predefined number of models is reached.

To estimate the heterogeneity of a cluster corresponding to a centroid model M we perform a *tentative* split. We create two copies of the cluster centroid M, M_1 and M_2 and add small antisymmetric perturbations to each copy. We then run a Soft-Clustering procedure to *softly* divide the data of the cluster into two new segments in a "good" way. The divergence between the centroids of the two segments we get after the division is used for the estimation of the heterogeneity of the parent cluster (segment).

The Soft-Clustering procedure is much like the EM algorithm with the only difference being that *resolution* of the partition is introduced through the Lagrange multiplier β. This multiplier emerges from the solution of an optimization problem when we are trying to find the assignment probabilities $P(M_j|z_i)$ and the models $\{M_j\}$ that will minimize the *mutual information*

$$I(Z,\mathcal{M}) = \frac{1}{N}\sum_i \sum_j P(M_j|z_i)\cdot \log\frac{P(M_j|z_i)}{P(M_j)}$$

with $P(M_j) = \frac{1}{N}\sum_i P(M_j|z_i)$, while allowing for a limited permitted level of distortion $\langle d\rangle \leq D$ (see [CT91]). Note that the assignment probabilities $P(M_j|z_i)$ determine a *soft* partition of the data among M_1 and M_2. The reason for minimizing the mutual information $I(Z,M)$ is that under given constraints (distortion in our case) the assignment minimizing it is the most probable one (see [CT91, Ros98]).

An iterative solution to the described optimization problem:

$$R(D) = \min_{\{P(M_j|z_i)\}_{i,j},\{M_j\}_j:\langle d\rangle\leq D,\sum_j P(M_j|z_i)=1} I(Z,\mathcal{M}) \qquad (1)$$

is given in the Soft-Clustering procedure (see Fig. 4). The procedure starts from a pair of models M_1, M_2, a prior distribution $P(M_*)$ over them, current resolution β and a (normalization) vector $\bar{P}(M|z_i)$ that stores for each window z_i the probability it belonged to the parent model M (thus we segment only the parent segment and mask out the rest of the data). Soft-Clustering then iterates between data partition (steps 1 and 2) and model reestimation (step 3) until convergence to a local minimum.

Soft-Clustering(M_1**,** M_2**,** $P(M_1)$**,** $P(M_2)$**,** β**,** $\bar{P}(M|z_i)$**)**

Repeat until convergence:

1. $P(M_j|z_i) = \frac{P(M_j)e^{-\beta d(z_i,M_j)}}{\sum_{k=1}^{K} P(M_k)e^{-\beta d(z_i,M_k)}} \cdot P(M|z_i)$
2. $P(M_j) = \frac{1}{n}\sum_{i=1}^{n} P(M_j|z_i)$
3. $M_j = \sum_t H(z_t) \cdot P(M_j|z_t)$

Figure 4: **Soft-Clustering procedure**

The Soft-Clustering procedure has the following important property. When working at low levels of resolution β (this corresponds to high levels of permitted distortion D) only one model suffices to describe the data. In this case, after the convergence of Soft-Clustering, the two models M_1 and M_2 either become almost identical or one of them vanishes ($P(M_j)$ is close to zero). Using this fact we start from low resolution parameter β and return on Soft-Clustering multiple times, each time increasing β by a multiplicative constant until we reach a critical value β_c. At this point two distinct non-trivial models are required to describe the data at the corresponding resolution. (When returning on Soft-Clustering we discard the result of the previous run and perform a new random split of M.)

The important point in this approach is that the lower the resolution is, the less local optima our optimization functional (1) has. Thus by working at gradually increasing levels of resolution we get to the point where our segmentation (Soft-Clustering) procedure has the greatest potential to succeed (minimal chances to fall into a local minima).

To determine whether we got a trivial or an interesting partition of the data we calculate the Jensen-Shannon divergence between M_1 and M_2 after the convergence of Soft-Clustering:

$$JS(M_1, M_2) = \frac{P(M_1)}{P(M)} D_{KL}(M_1||M) + \frac{P(M_2)}{P(M)} D_{KL}(M_2||M)$$

This measure of distance between M_1 and M_2 answers the question of how probable it is that samples corresponding to M_1 and those corresponding to M_2 are actually coming from the same statistical source (see [Slo02, Sec. 1.2.5] for a

relevant discussion). We compare $JS(M_1, M_2)$ to $\frac{1}{\beta}$. If $JS(M_1, M_2) > \frac{1}{\beta}$ we consider the result of the partition as interesting. We further define $Score(M) = P(M) \cdot JS(M_1, M_2)$. Note that $Score(M)$ equals the decrease in the distortion $\langle d \rangle$ we will obtain if we replace M with M_1 and M_2 in the partition.

To summarize the algorithm: We start from $\mathcal{M}$ consisting of a single average model M_0. We then repeatedly choose the model having the maximal score from $\mathcal{M}$ and replace it with the two child models used to calculate its score. This way we always advance in the direction of maximal decrease in the distortion $\langle d \rangle$. To calculate the score of a model M we divide the corresponding segment into two and use the obtained models to estimate the heterogeneity of the parent segment. While segmenting the segment corresponding to M, we gradually increase the resolution parameter β in order to solve the problem at the optimal resolution, where we have minimal chances to get trapped in a local minimum. For each model M we remember the resolution parameter β at which it was created and when its turn comes to be further segmented, we start the search for the optimal resolution for this segment from that value. Initial resolution value, as well as the multiplicative step for its increase are manually chosen, while "the rules of thumb" are: (1) at the initial resolution the data should be best represented with a single model and (2) no more than a single increment in the number of models in $\mathcal{M}$ must occur between two subsequent increments in β. See Fig. 2 for forced hierarchy framework schematic illustration and Fig. 3 for the pseudo code.

Compared to the "classical" DA framework (as, e.g. in [Ros98, Sel01]), in which the models produced by split "see" and compete on all the data, our new framework, which we call *forced hierarchy DA* is more limited. This is because each time child models "see" only the fraction of the data that was captured by their parent. But due to this limitation our algorithm is much faster, since at every moment we work with only two models. Thus we get the same segmentation resolution in a much shorter time. The fact that we split each model at its optimal resolution helps us to avoid many local minima. Therefore the algorithm almost does not suffer from the hierarchical limitation on the partitions.

Choosing "the correct" number of segments.

While multiple segmentation solutions may emerge at different levels of resolution, some of them are more stable while others are less. This may be best demonstrated by an example - see Fig. 5. To search for such stable solutions we look at the distortion-rate function $D(R)$ (which is the inverse of the $R(D)$ function defined by (1)). Each time a model out of $\mathcal{M}$ is split we obtain a new point on this

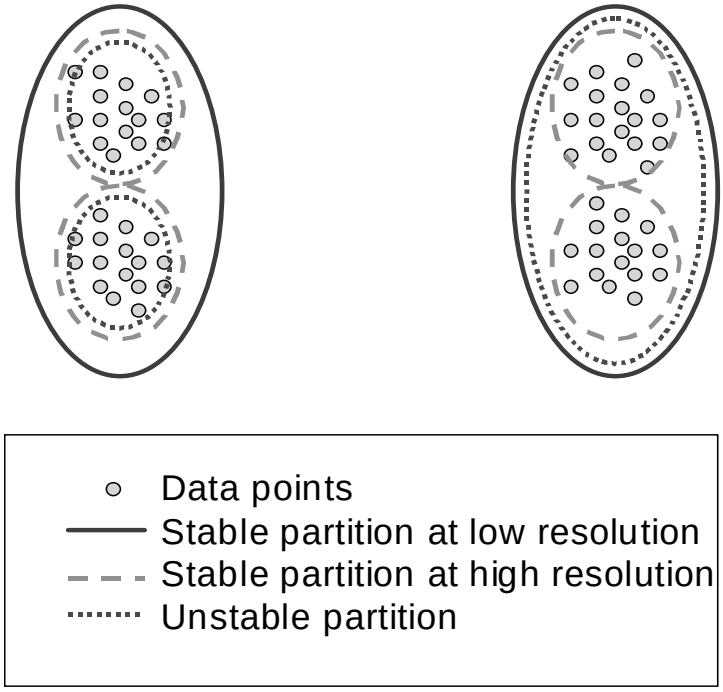


Figure 5: **Stable clusters illustration example.** This illustration example comes to show the difference between stable and unstable clusters. Suppose that your input is four "clouds" of points as shown here. Then the segmentation into two clusters shown by the solid line is stable at low resolution and the segmentation into four clusters shown by dashed line is stable at high resolution. At the same time, the segmentation into three clusters shown by the dotted line is unstable since small variations in the input may easily cause the algorithm first to split the right pair of clouds and not the left. See [Ros98] for a deeper discussion on clusters stability.

curve. The rate of the decrease in the distortion $\langle d \rangle$ with the increase in the mutual information $I(Z, M)$ changes while the number of clusters increases. As argued in [Slo02, Sec. 6.2.2], points where this rate slows down (the points of minima in the second derivative of $D(R)$) correspond to stable segments in the partition. Thus we used the second derivative of $D(R)$ to choose the appropriate number of segments for the subsequent classification step.

4. Images Classification According to Segmentation Map

Now we come to the final and major algorithmic point of our paper. We use the segmentation result we got in the previous section for unsupervised classification of the input set of images. The idea is to regard each image I_i as a document and each model M_j as a word, defining $P(M_j|I_i) = \frac{1}{N(I_i)} \sum_{z_t \in I} P(M_j|z_t)$ analogous to the normalized count of the number of appearances of M_j in I ($N(I_i)$ is the number of windows in I_i). With this analogy we use the sequential Information Bottleneck (sIB) clustering algorithm [SFT02] to cluster by images the co-occurrence matrix $P(I_i, M_j)$.

The sIB clustering algorithm is based on the Information Bottleneck (IB) clustering framework proposed in [TPB99]. Its goal is to represent input images $\{I_i\}$ with a small number of clusters $\{C_k\}$ so that the distribution of models (features) inside each cluster $P(M_j|C_k) = \frac{1}{|C_k|} \sum_{I_i \in C_k} P(M_j|I_i)$ will be as close as possible to the original distributions $P(M_j|I_i)$ of images constituting the clusters. In [TPB99] it is argued that the correct measure for quality of the representation is the mutual information fraction $\frac{I(C;M)}{I(I;M)}$.

In [SFT02] it was shown that the sIB algorithm has a superior performance over a range of other clustering methods (agglomerative, K-means and more), typically by a significant margin, and even comparable with standard Naive Bayes (supervised) classifier on problems of document classification. The idea of the sIB algorithm is to start from a random partition of the data into K clusters and sequentially take a random sample I_i from its cluster and reassign it to a new cluster C_*, so that the mutual information $I(C;M)$ (and thus the quality of the representation $\frac{I(C;M)}{I(I;M)}$) will be maximized. Due to the monotonic growth of $I(C;M)$ the procedure is guaranteed to converge to a local optimum.

5. Results

5.1 Texture Segmentation Example

As a basic test for the segmentation step of our algorithm, we took synthetic compositions of gray-scaled textures from [HPB98]. Each of the compositions consists of five different textures. We applied our segmentation algorithm with windows of 32x32 pixels size, grid shift of 8 pixels, $L = 3$ wavelet decomposition levels with $rbio\_3.3$ and $db4$ wavelet filters. Both filters provided us with similar

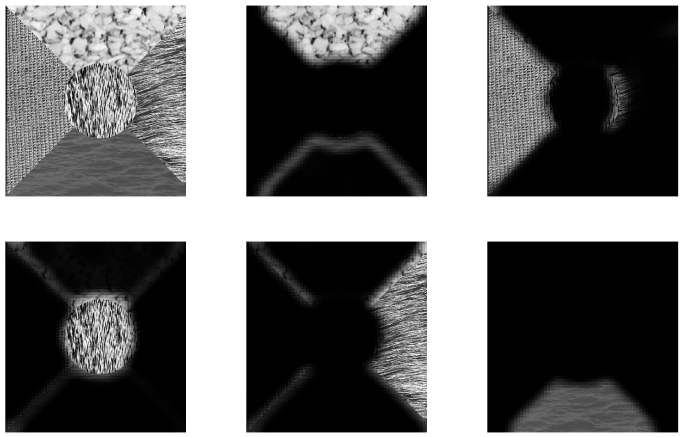


Figure 6: **Texture Segmentation Example.** Top left - the original image consisting of 5 regions, each of different texture. The rest five images are the obtained segmentation of the original image into 5 segments.

results. See an example of such segmentation in Fig. 6. The image is divided into 5 models, corresponding to each of the 5 textures.

5.2 Classification of Natural Views

In this section we demonstrate the classification ability of our algorithm on two examples of natural views image sets[1].

Sea Views

As a first example, we took 29 color pictures (640x480 pixels) of natural sea views and ran the algorithm with window size of 64x64, grid shift of 32, $L = 3$ wavelet decomposition levels, with $rbio3.3$ and $db4$ wavelet filters. Both filters provided us with similar results. The partition into 5 clusters is shown in Fig. 7. The first obtained cluster consists of panoramic views of a sandy shore. In the second the shore is more rocky. The third cluster contains close-up photos of water-plants, the fourth consists of the sea with a beach of plowed sand, and the fifth are close up photos of the sea waves.

[1]The results of sections 5.2 and 5.3 may be also viewed in a better quality at *http://www.cs.huji.ac.il/∼seldin/SCTV2003*

The Hebrew University of Jerusalem Campus Views

The second set consists of 55 color pictures (640x480 pixels) taken on the Hebrew University campus. We divided the images into 2,3,4,5 and 6 clusters (see the partitions into 4 and 6 clusters in Fig. 8 and Fig. 9 respectively). It is encouraging to see almost hierarchical splitting of the clusters as their number increases.

The division into 4 clusters is as following: The first cluster contains perspective views including mixtures of trees, trails, sky etc., the second consists of indoor scenes (which can be easily identified by their smooth textures), the third are tree branches on the sky background (detailed texture on a smooth background), and the last is close up views of flowers and bushes (very detailed textures).

In further divisions, the first cluster splits into two, separating out flowers with a large portion of asphalt background, and cluster 5 selects images, containing very few details on a smooth background.

In Fig. 10 we show some examples of the segmentation of the images into 3 segments, which was used for the classification. One representative image from each cluster is taken.

As we may see, the algorithm was successful in classifying input images of nature views by their content. The obtained results were stable (as was explained in Fig. 5). That is, we converged to the same classification results in different algorithm runs and with different wavelet filters.

5.3 Classification of Painting styles

To try our algorithm on another possible application, we took 35 pictures (about 600x800 pixels size) drawn by 5 different painters, trying to identify the painters by their drawing style. In the experiments we have carried out, each of the painters was classified correctly, and the obtained clusters were stable. We used windows of 128x128 and 64x64, overlapping and not overlapping, with different wavelet filters, and always converged to the same result, shown in Fig. 11.

The obtained clusters differentiate between impressionism style of Van Gogh, characterized by a large amount of bold brush-strokes, classic style of Rembrandt, with its soft lines and smooth textures, landscape reproductions of Shishkin, rich of small scrupulous details, cubism of Picasso and marine landscapes of Aivazovsky.

It should be noted that in this example the correct classification was obtained only when we took sufficiently fine segmentation of the input images (into about 20-25 segments). In order to choose a segmentation with an appropriate number of

segments for the subsequent classification step, we looked at the inverse rate distortion function $D(R)$, as described in Sec. 3. See a graph of this function and its two derivatives on the painters classification example in Fig. 12. The points where the decrease of the distortion slows down relative to the increase in the mutual information correspond to stable segmentation solutions. The vertical grid lines in the graph correspond to the increments of the number of segments in the partition by one. By looking at the minima of the second derivative of the function, we identify stabilization points in the segmentation solution. In the example shown, one of the evident (actually most evident) local minima corresponds to segmentation into 24 segments, which is actually the point where the classification results become correct and stabilize. When taking segmentations with a greater number of segments we always converged to the same correct classification of the input images. At the same time, when classifying images using segmentation into less than 24 segments, a small number of misclassifications were present.

When using large (256x256 and more) windows, at all the segmentation resolutions in subsequent classification some of the pictures were misclassified (for example, some of the Picasso paintings were mixed up with those of Aivazovsky). As well, we got poor results when used one big window of the whole image size, which is equivalent to classifying the images without segmenting them first. These results justify the advantage of image segmentation as a preprocessing step for classification.

6. Discussion and Future Work

The presented work combines several ideas from different areas of machine learning and image processing into one coherent procedure for unsupervised content based image classification. The major contribution of our work is the idea of using *joint* segmentation of a set of images for its clustering using image/document, segment/word analogy. Our framework may be viewed as two-levels unsupervised approach to data analysis - on the first level we do unsupervised feature extraction from the data and on the second we perform an unsupervised classification of the data based on the extracted features.

In our experiments the suggested method is successfully applied to two types of visual data: natural scenes classification and classification of painters by their drawing styles. As we show there, the classification based on segmented images gives better results compared to classifying unsegmented images. This confirms the superiority of our algorithm.

The suggested approach of two-levels unsupervised data analysis is general

and may be applied to essentially any type of data, including but not limited to biomolecular sequences, audio signals, spike trains and handwriting data analysis. Of course, in each of those fields appropriate models for local data modeling and algorithms for segmentation must be used.

Various directions for further research may be suggested. We may naturally categorize them into four categories. One is applying our approach to other types of data. For example, it would be interesting to combine it with our previous work [BSMT01] on feature extraction in proteins (there we called them signatures). Another is improving the results we got in this paper by using better algorithms for image modeling and segmentation. Here we thought about using beamlets [DH01] instead of wavelets for local image modeling. This may provide results which are closer to human perception of visual data. A third direction focuses on the classification step of our algorithm. Here, by using more sophisticated algorithms for classification we may achieve such interesting things as classification of the input set by multiple parameters (as in the work of [CT02], where they cluster the input set of human faces once by the person on the image and once by the illumination of the person - persons illuminated from the left, from the right, homogeneously, etc.). Finally, the most attractive direction seems to find a way of "getting feedback" from the classification step during the segmentation to develop a new algorithm for unsupervised *discriminative* feature selection.

Acknowledgments

We thank Gal Chechik for providing us his implementation of the sequential Information Bottleneck algorithm and Efim Belman for helpful discussions.

Yevgeny Seldin was partially supported by the Leibniz Center scholarship. Sonia Starik was partially supported by the ISF grant number 591/00.

References

[BDF02] K. Barnard, P. Duygulu, and D. Forsyth. Modeling the statistics of image features and associated text. In *SPIE Electronic Imaging 2002, Document Recognition and Retrieval IX*, 2002.

[BSMT01] Gill Bejerano, Yevgeny Seldin, Hanah Margalit, and Naftali Tishby. Markovian domain fingerprinting: statistical segmentation of protein sequences. *Bioinformatics*, 17(10):927–934, 2001.

[CT91] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. Wiley Series in Telecommunications. John Wiley & Sons, New York, NY, 1991.

[CT02] Gal Chechik and Naftali Tishby. Extracting relevant structures with side information. In *Advances in Neural Information Processing Systems, NIPS-15.*, Vancouver. Canada., 2002.

[DH01] David Donoho and Xiaoming Huo. Beamlets and multiscale image analysis. *Lecture Notes in Computational Science and Engineering: Multiscale and Multiresolution Methods. Springer.*, 2001.

[DV00] Minh N. Do and Martin Vetterli. Texture similarity measurement using kullback-leibler distance on wavelet subbands. In *In proceedings of IEEE International Conference on Image Processing, ICIP-2000*, September 2000.

[GFT98] B. Gunsel, A. Ferman, and A. Tekalp. Temporal video segmentation using unsupervised clustering and semantic object tracking. *Journal of Electronic Imaging*, 7(3):592–604, July 1998. SPIE IS&T.

[GGG02] Jacob Goldberger, Hayit Greenspan, and Shiri Gordon. Unsupervised image clustering using the information bottleneck method. In *DAGM-Symposium*, pages 158–165, 2002.

[HPB98] Thomas Hofmann, Jan Puzicha, and Joachim M. Buhmann. Unsupervised texture segmentation in a deterministic annealing framework. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(8):803–818, 1998.

[Ker] Daniel Keren. Recognizing image style and activities in video using local features and naive bayes.

[MK01] Swarup Medasani and Raghu Krishnapuram. Categorization of image databases for efficient retrieval using robust mixture decomposition. *Computer Vision and Image Understanding: CVIU*, 83(3):216–235, 2001.

[RHC99] Y. Rui, T. Huang, and S. Chang. Image retrieval: current techniques, promising directions and open issues. *Journal of Visual Communication and Image Representation*, 10(4):39–62, April 1999.

[Ros98] Kenneth Rose. Deterministic annealing for clustering, compression, classification, regression and related optimization problems. *IEEE Trans. Info. Theory*, 80:2210–2239, November 1998.

[SBT01] Yevgeny Seldin, Gill Bejerano, and Naftali Tishby. Unsupervised sequence segmentation by a mixture of switching variable memory Markov sources. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001)*, pages 513–520, 2001.

[Sel01] Yevgeny Seldin. On unsupervised learning of mixtures of Markovian sources. Master's thesis, The Hebrew University of Jerusalem, 2001.

[SFT02] Noam Slonim, Nir Friedman, and Naftali Tishby. Unsupervised document classification using sequential information maximization. In *Proceedings of the* 25^{th} *Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR)*, 2002.

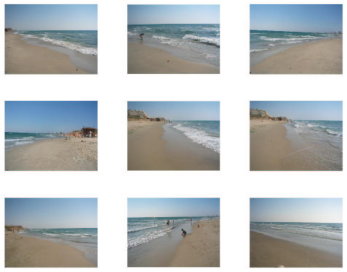
[Slo02] Noam Slonim. *The Information Bottleneck: Theory and Applications*. PhD thesis, The Hebrew University of Jerusalem, 2002.

[SMM] S.Krishnamachari and M.Abdel-Mottaleb. Hierarchical clustering algorithm for fast image retrieval. In *IS&T/SPIE Conference on Storage and Retrieval for Image and Video databases VII*.

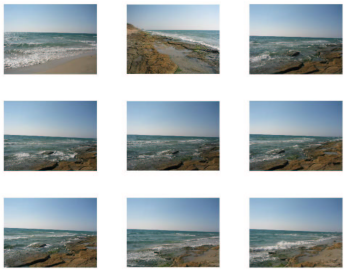
[THI+00] L. Tang, R. Hanka, H.. Ip, K. Cheung, and R. Lam. Semantic query processing and annotation generation for content-based retrieval of histological images. In *SPIE EMedical Imagingg 2000, Document Recognition and Retrieval IX*, 2000.

[TPB99] Naftali Tishby, Fernando Pereira, and William Bialek. The information bottleneck method. In *Allerton Conference on Communication, Control and Computation*, volume 37, pages 368–379. 1999.

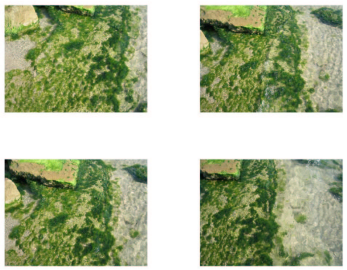
[WWFW97] James Ze Wang, Gio Wiederhold, Oscar Firschein, and Sha Xin Wei. Content-based image indexing and searching using daubechies' wavelets. *Int. J. on Digital Libraries*, 1(4):311–328, 1997.



(a) Cluster 1 of 5: Sandy shore.



(b) Cluster 2 of 5: Rocky shore.



(c) Cluster 3 of 5: Water-plants.

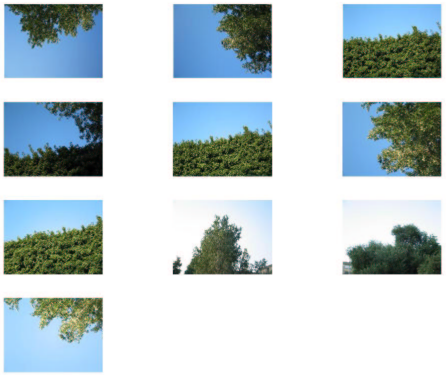
(d) Cluster 4 of 5: Plowed sand and sea.

(e) Cluster 5 of 5: Close view of sea waves.

Figure 7: **Sea views classification.**

(a) Cluster 1 of 4: Perspective views.

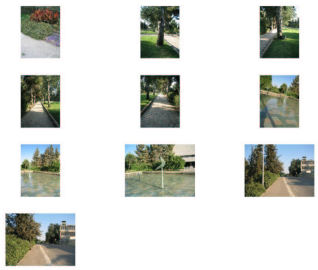
(b) Cluster 2 of 4: Indoor scenes.



(c) Cluster 3 of 4: Tree branches on the sky background.

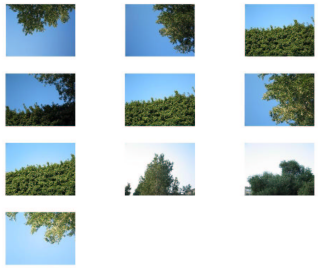
(d) Cluster 4 of 4: Bushes and flowers, close view.

Figure 8: **The Hebrew University Campus views classification into four clusters.**



(a) Cluster 1 of 6: Perspective views.

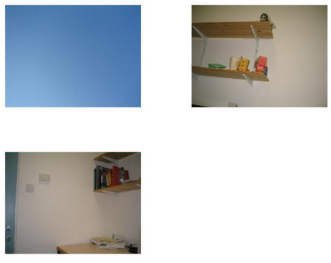
(b) Cluster 2 of 6: Indoor scenes.



(c) Cluster 3 of 6: Tree branches on the sky background.

(d) Cluster 4 of 6: Bushes and flowers, close view.

(e) Cluster 5 of 6: Flowers with asphalt.

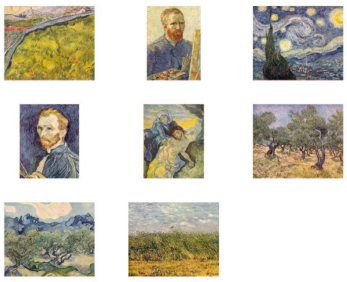


(f) Cluster 6 of 6: Smooth background with few details.

Figure 9: **The Hebrew University Campus views classification into 6 clusters.**

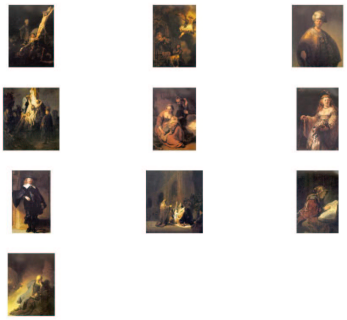
(a) Original Image (b) Segment 1 (c) Segment 2 (d) Segment 3

Figure 10: **The Hebrew University Campus views segmentation into 3 segments.** The above are examples of image segmentations into 3 segments before their classification. Each row contains an image from a different cluster (see Fig. 9): The original image - column (a), and its segmentation into the 3 segments - columns (b)-(d). Segment 1 corresponds to smooth textures like sky and walls in all the input images. Segment 3 corresponds to very detailed textures like bushes and tree branches. Segment 2 corresponds to textures with middle level of detail like asphalt roads and trails.

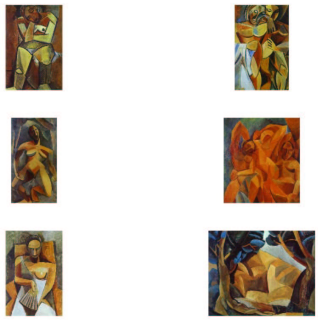


(a) Van Gogh, impressionism.

(b) Aivazovsky, marine landscapes.



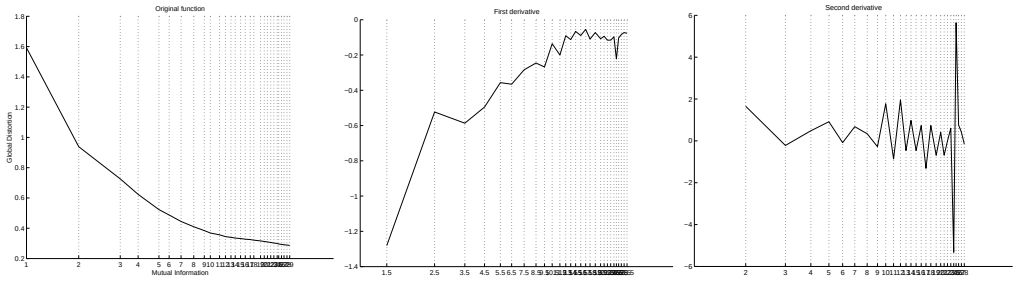
(c) Rembrandt, classic style.



(d) Picasso, cubism.

(e) Shishkin, landscape reproductions.

Figure 11: **Classification of drawings by painting style.**



(a) $D(R)$ function

(b) The first derivative of $D(R)$

(c) The second derivative of $D(R)$

Figure 12: **Inverse Rate Distortion Function Graph for Painters Classification Example.** The numbers below the R ($R = I(Z, M)$) axis and vertical grid lines correspond to the number of segments in the segmentation at which the point on the graph was measured. We see a clear minimum in the second derivative of the $D(R)$ function at the solution with 24 segments. This solution is a stable segmentation. (The numbers below the axis are not clearly seen. Please, refer our web site for higher resolution graphs.)