treeKL: A distance between high dimension empirical distributions

Riwal Lefort [(1)], François Fleuret [(1,2)]

(1) Idiap research institute, Switzerland, riwal.lefort@idiap.ch
(2) EPFL, Switzerland, francois.fleuret@idiap.ch

Abstract

This paper offers a methodological contribution for computing the distance between two empirical distributions in an Euclidean space of very large dimension.

We propose to use decision trees instead of relying on standard quantification of the feature space. Our contribution is two-fold: We first define a new distance between empirical distributions, based on the Kullback-Leibler (KL) divergence between the distributions over the leaves of decision trees built for the two empirical distributions. Then, we propose a new procedure to build these unsupervised trees efficiently.

The performance of this new metric is illustrated on image clustering and neuron classification. Results show that the tree-based method outperforms standard methods based on standard bag-of-features procedures.

Keywords: Kulback-Leibler distance, unsupervised trees, distribution modeling.

1. Introduction

This paper tackles the problem of computing distance between two sets of points in an Euclidean space of large dimension.

The most straight-forward methods to address this problem consists of quantifying the space into bins, and computing a distance between the resulting empirical distributions [3]. However, such approaches are useless when the dimension of the feature space dimension gets very large. In many application domains, a popular approach consists of fitting a mixture of Gaussians

[4], and estimating the Kullback-Leibler (KL) divergence between the distributions from these models. The first drawback of these methods based on Gaussian kernels is that a model (number of clusters, regularization of the covariance matrices, etc) must be chosen for the probability density function. The second drawback is that such a distance, or related methods for instance based on Parzen windows [7] or the Mahalanobis distance [6], are computationally intensive when the number of points is high.

In many application fields, in particular in computer vision, very efficient techniques rely on the idea of bag-of-features (bof), which model the empirical distributions with distribution over clusters computed adaptively from data [9, 11, 22, 10].

In this paper, we propose a new tree-based method for computing the distance between two sets of points. The core idea of our method is to build a fully developed unsupervised tree from each family of points, and to compute the KL divergence between the empirical distribution over leaves estimated from each family of points. The distribution associated to the family used to build the tree will be uniform, but the distribution associated to other families may be more deterministic, reflecting the distance between them.

In § 2, we present this KL-based distance between dissimilar unsupervised trees, and introduce a new fast method for learning the unsupervised trees. Our efforts have been focused on trees because they offer three clear benefits: they tolerate high dimensional data-sets, they do not require to choose an empirical density model for the distribution of the points, and they are flexible to data in the sense that they can mix categorical features with continuous features [21, 8]. The main advantage of our proposed distance is that there is no tuning parameters and low computational cost.

After describing in § 3 how the distance is used for both unsupervised and supervised learning, we provide experimental results in § 4 that show how our method outperforms bag-of-features on average.

2. Tree-based Kullback-Leibler divergence

We describe here the proposed method which consists of measuring the distance between two sets of points. The global distance is presented in § 2.1, and the method for building trees is presented in § 2.2. Then, in sections § 2.3 and § 2.4, we discuss respectively the dimension of the feature space, and the computational cost.

2.1. Distance between dissimilar trees

Let $\mathbf{X}_i = \{X_{i,1} \dots X_{i,K_i}\}$ be a matrix that represents one object to classify, where $X_{i,k} \in \mathbb{R}^F$ and F is the number of features. Any object i is then associated to K_i points in a feature space. For instance, in computer vision $X_{i,k}$ would stand for the kth SIFT point in the image i [9, 11]. In biology, for the classification of neurons in videos (see § 4.3), $X_{i,k}$ would denote the parameters of the neuron i in the kth frame. Let Q_i be the probability density function of the points in $\mathbf{X}_i$. Having the objective to classify the object i, we will define a distance between the distributions $\{Q_i\}_i$.

The distributions $\{Q_i\}_i$ are modeled by using unsupervised trees (§ 2.2). Let $\mathcal{T}_{X_i}$ be the unsupervised tree associated to the object i, and let M_i be the number of leaves in $\mathcal{T}_{X_i}$. For computing the distance $\Delta(Q_i, Q_j)$ between the object i and the object j, we propose to pass $\mathbf{X}_i$ through the tree $\mathcal{T}_{X_j}$ and to pass $\mathbf{X}_j$ through the tree $\mathcal{T}_{X_i}$. Let the vector $\mathcal{T}_{X_i}(\mathbf{X}_j) \in \mathbb{R}^{M_i}$ be the distribution of $\mathbf{X}_j$ over the leafs of the tree $\mathcal{T}_{X_i}$. Then, we define the distance between the object i and the object j as follows:

$$\Delta(Q_i, Q_j) = \frac{1}{2}\left[d_{KL}(\mathcal{T}_{X_i}(\mathbf{X}_j), \mathcal{T}_{X_i}(\mathbf{X}_i)) + d_{KL}(\mathcal{T}_{X_j}(\mathbf{X}_i), \mathcal{T}_{X_j}(\mathbf{X}_j))\right] \qquad (1)$$

where $d_{KL}(\mathcal{T}_{X_i}(\mathbf{X}_j), \mathcal{T}_{X_i}(\mathbf{X}_i))$ denotes the KL divergence between vector $\mathcal{T}_{X_i}(\mathbf{X}_j)$ and vector $\mathcal{T}_{X_i}(\mathbf{X}_i)$:

$$d_{KL}(\mathcal{T}_{X_i}(\mathbf{X}_j), \mathcal{T}_{X_i}(\mathbf{X}_i)) = \sum_{m=1}^{M_i} \mathcal{T}^m_{X_i}(\mathbf{X}_j) \log \frac{\mathcal{T}^m_{X_i}(\mathbf{X}_j)}{\mathcal{T}^m_{X_i}(\mathbf{X}_i)} \qquad (2)$$

where $\mathcal{T}^m_{X_i}(\mathbf{X}_j)$ is the mth component of the vector $\mathcal{T}_{X_i}(\mathbf{X}_j)$. In other words, $\mathcal{T}^m_{X_i}(\mathbf{X}_j)$ is related to the number of points $X_{j,k}$ that reach the mth leaf of the tree $\mathcal{T}_{X_i}$. Note that the vector $\mathcal{T}_{X_i}(\mathbf{X}_i) \in \mathbb{R}^{M_i}$ is uniform with the components equal to $1/M_i$.

The distance (1) can be interpreted as follows: If the distribution of the points of one family is uniform over the leaves of the tree built with the other family, Q_i are Q_j similar. In other words, if Q_i and Q_j are identical, the points $\mathbf{X}_i$ should fill all the leafs of the tree $\mathcal{T}_{X_j}$ and the points $\mathbf{X}_j$ should fill all the leafs of the tree $\mathcal{T}_{X_i}$. In this case, we should find that $\mathcal{T}_{X_i}(\mathbf{X}_j) = \mathcal{T}_{X_i}(\mathbf{X}_i)$ and $\mathcal{T}_{X_j}(\mathbf{X}_i) = \mathcal{T}_{X_j}(\mathbf{X}_j)$, and then, the distance $\Delta(Q_i, Q_j)$ must reach its minimum value: $\Delta(Q_i, Q_j) = 0$.

If the distribution of the points in the leaves of the tree build with the other family is deterministic, i.e. they all fall in the same leaf, then Q_i and

Q_j are as dissimilar as possible. In other words, if Q_i and Q_j are widely separated in the feature space, the points $\mathbf{X}_i$ should fill only one leaf of the tree $\mathcal{T}_{X_j}$ and the points $\mathbf{X}_j$ should fill only one leaf of the tree $\mathcal{T}_{X_i}$. In this case, $\mathcal{T}_{X_i}(\mathbf{X}_j)$ and $\mathcal{T}_{X_j}(\mathbf{X}_i)$ have a binary form, i.e. only one component equals one and the others equal zero, and then, the distance $\Delta(Q_i, Q_j)$ must reach its maximum value: $\Delta(Q_i, Q_j) = \frac{1}{2}[\log M_i + \log M_j]$. The process is illustrated in Figure 1.

The proposed method offers has three nice practical properties. First, the quantity (2) is always numerically defined since $\mathcal{T}^m_{X_i}(\mathbf{X}_i) = \frac{1}{M_i}$ is always greater than zero. Second, since $\mathcal{T}_{X_i}(\mathbf{X}_i)$ is always uniform, it is not necessary to pass $\mathbf{X}_i$ through $\mathcal{T}_{X_i}$. Third, the distance can be used with dissimilar trees, i.e. trees that do not have the same number of leaves.

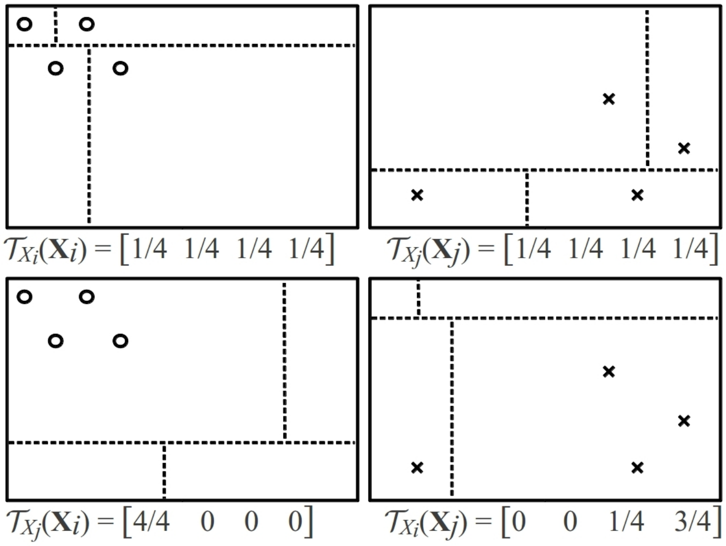

$\mathcal{T}_{Xi}(\mathbf{X}i) = [1/4 \ \ 1/4 \ \ 1/4 \ \ 1/4]$
$\mathcal{T}_{Xj}(\mathbf{X}j) = [1/4 \ \ 1/4 \ \ 1/4 \ \ 1/4]$
$\mathcal{T}_{Xj}(\mathbf{X}i) = [4/4 \ \ 0 \ \ 0 \ \ 0]$
$\mathcal{T}_{Xi}(\mathbf{X}j) = [0 \ \ 0 \ \ 1/4 \ \ 3/4]$

Figure 1: Top-left: the circles ($\mathbf{X}_i = \{X_{i,1} \dots X_{i,4}\}$) and the boundaries (dashed) of the associated decision tree ($\mathcal{T}_{X_i}$) such that the distribution $\mathcal{T}_{X_i}(\mathbf{X}_i)$ is uniform. Top-right: the crosses ($\mathbf{X}_j = \{X_{j,1} \dots X_{j,4}\}$) and the boundaries (dashed) associated decision tree ($\mathcal{T}_{X_j}$) such that the distribution $\mathcal{T}_{X_j}(\mathbf{X}_j)$ is uniform. Bottom-left: the distribution $\mathcal{T}_{X_j}(\mathbf{X}_i)$ is computed by listing the circles ($\mathbf{X}_i$) in the leafs of the tree $\mathcal{T}_{X_j}$. Bottom-right: the distribution $\mathcal{T}_{X_i}(\mathbf{X}_j)$ is computed by listing the crosses ($\mathbf{X}_j$) in the leafs of the tree $\mathcal{T}_{X_i}$. In this example, $\Delta(Q_i, Q_j) = \frac{1}{2}\left[\frac{3}{4}\log 3 + \log 4\right] = 1.1$. $\Delta(Q_i, Q_j) \neq 0$ which means that Q_i and Q_j are not equal.

2.2. Unsupervised tree learning

Unsupervised trees are usually used as tools for classification and clustering. In this content, they are an alternative to k-means and are used for grouping together similar data [20, 24, 25]. For instance, a tree can be used for creating clusters in the feature space [24]. Then, bof are built by passing the key points in the tree.

Alternatively, unsupervised trees can be used for modeling probability density functions. In this case, a set of unsupervised trees $\{\mathcal{T}_{X_i}\}_i$ is generated, where each tree is used for modeling the corresponding distribution Q_i (please refer to § 2.1 for the notations). An unsupervised tree can then be viewed as a histogram with bins of different sizes. Breiman [8] proposed to generate such trees by simulating synthetic classes and to use supervised decision trees for separating the data of interest from the synthetic data [19]. Other authors proposed unsupervised tree techniques based on a specific criterion but for only one dimension [26]. The main drawback of these methods is the impossibility to process high dimensional data-set. Herein, we propose a fast and very simple way for constructing unsupervised trees. The major advantages of our method are that there are no tuning parameters or other criterion optimization, and it can be used with high-dimensional data.

Formally, given the points $\mathbf{X}_i = \{X_{i,1} \dots X_{i,K_i}\}$ that follow the distribution Q_i, a M_i-leaf tree $\mathcal{T}_{X_i}$ is built such that each component of the vector $\mathcal{T}_{X_i}(\mathbf{X}_i)$ equals $\frac{1}{K_i} = \frac{1}{M_i}$, i.e. the distribution $\mathcal{T}_{X_i}(\mathbf{X}_i)$ is uniform (please refer to § 2.1 for the notations and see the Figure 1 for illustration). Intuitively, a tree is built such that each final node contains only one training instance, i.e. the probability for the training instance to reach a leaf always equals $\frac{1}{K_i} = \frac{1}{M_i}$. This is illustrated in Figure 1: each leaf of the trees contains only one point.

The speed of the process is a significant issue. For this reason, oblique unsupervised trees are considered. This means that each node of the tree is associated to a hyperplane separator that considers the whole feature space. The learning step of the unsupervised tree involves the computation of the hyperplanes coefficients. Efficiency, hyperplane coefficients are derived from the bisection of two random points that are sampled among the data of the considered node. In Figure 1, hyperplanes are represented by both horizontal and vertical dashed lines.

2.3. Discussion about the dimensionality

Decision trees are naturally suitable for distribution modeling. In comparison to histograms, they can capture all aspects of a distribution, and they focus on their particularities. For instance, a 3D histogram partitions the feature space in homogeneous subspaces, regardless of the distribution of observations. In comparison, decision trees partition the feature space according to the distribution.

In addition, decision trees are naturally suitable for dimensionality reduction. Let F be the number of features and B the number of bins for each feature of histogram. Computing histograms in high dimensional space is impossible with histograms, because the total number of bins is B^F. In comparison, considering decision trees, the number of bins always equals the number of leaves.

In data analysis, classification tasks are often preceded by dimensionality reduction. By using unsupervised tree, our method considers in a unique step both dimensionality reduction and classification task.

The high-dimensional tolerance is illustrated in the experiments (§ 4), by using data-sets that consider $F = 2$ and $F = 1000$ (§ 4.1), $F = 128$ (§ 4.2) and $F = 95$ (§ 4.3).

2.4. Computational cost

For Parzen-windows [7], for Gaussian kernel [17], and for mixture of Gaussian [2], the complexity for computing distance between two sets of points is $\mathcal{O}(N_i N_j)$ where N_i and N_j denote the number of points for instance i and j respectively. This may be computationally difficult if N_i and N_j are high or if the number of feature is high.

In comparison, the complexities for building the two trees are $\mathcal{O}(N_i log(N_i))$ and $\mathcal{O}(N_j log(N_j))$ respectively. The complexities for passing the samples $\mathbf{X_i}$ in the opposite tree $\mathcal{T}_{X_j}$ is $\mathcal{O}(N_i log(N_j))$ and the complexities for passing the samples $\mathbf{X_j}$ in the opposite tree $\mathcal{T}_{X_i}$ is $\mathcal{O}(N_j log(N_i))$. Then, the final complexity is $\mathcal{O}((N_i + N_j) log(N_i + N_j))$ which is less than $\mathcal{O}(N_i N_j)$ asymptotically.

3. Unsupervised classification and supervised classification

In this section, we present how the proposed divergence (1) can be used for both unsupervised learning (§ 3.1) and supervised learning (§ 3.2).

3.1. Unsupervised learning

Recall that Q_i denotes the distribution of the points $\mathbf{X}_i = \{X_{i,1} \dots X_{i,K_i}\}$. Using a clustering method, similar Q_i are grouped together. Once a distance between Q_i and Q_j is defined, any clustering method can be used.

In this paper, we consider k-means. The k-means algorithm groups together similar objects by alternating the two following stpdf. First, the labels are updated according to the distance between the examples and the class centroids. Second, the class centroids have to be re-assessed.

When working with distributions of points, the centroids of the distributions have to be defined. Among the methods that we have tried, the best performance of the clustering has been achieved when the two k-means stpdf are fused. Instead of computing the distance between each distribution and the centroid distribution, the mean distance is computed. The distance $\Delta(Q_i, \bar{Q})$ between any distribution Q_i and the centroid $\bar{Q}$ can then be directly computed without centroid assessment:

$$\Delta(Q_i, \bar{Q}) = \frac{1}{\sum_{j=1}^{N} \delta(Q_j)} \sum_{j=1}^{N} \Delta(Q_i, Q_j)\delta(Q_j) \tag{3}$$

where $\delta(Q_j) = 0$ if Q_j belong to the considered class and $\delta(Q_j) = 1$ otherwise, and N is the number of objects. The distance $\Delta(Q_i, Q_j)$ is computed as in equation (1).

3.2. Supervised learning

Given a distance, the k-nearest-neighborhood classifier (k-nn) can be used to classify data, but it usually does not model properly the class boundary. Using the distance as a kernel, one can also use SVM, which are usually more efficient in such case [9, 11, 22, 10].

Let us consider the special case of two classes. Let $h(x) = \sum_n \alpha_n y_n K(x, x_n)$ be the classification function of the example x, where $y_n \in \{+1, -1\}$ refers to the classes associated to the training example x_n, coefficients $\{\alpha_n\}$ are assessed in the training step, and $K(x_1, x_2)$ is a kernel function. The Gaussian kernel is chosen as $K(x_1, x_2) = exp(-d(x_1, x_2)^2/\sigma)$ where σ is a scale parameter and $d(x_1, x_2)$ is the distance between examples x_1 and x_2.

Depending on the application, $d(x_1, x_2)$ can refer to the Euclidean distance [9] or to the χ^2 distance [22, 10], etc. We use the distance proposed in equation (1): $d(\mathbf{X}_i, \mathbf{X}_j) = \Delta(Q_i, Q_j)$.

The generalization of the method to multi-class classification is straightforward. Multi-class k-nn requires no particular tools, as well as kernel-based classifiers for which the kernel matrix contains all the distances between all the examples from each classes.

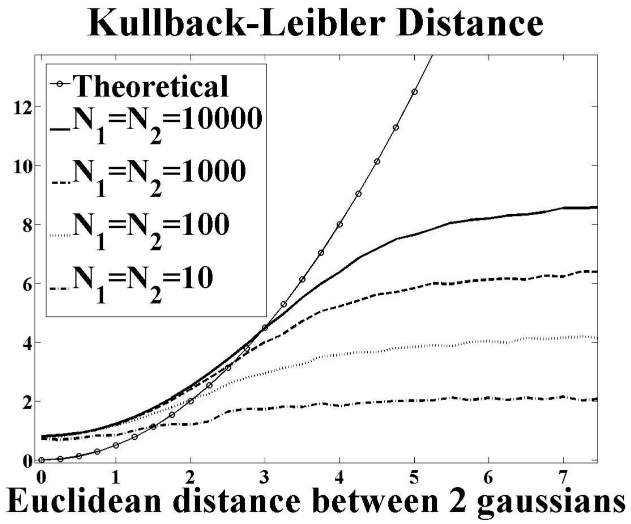


Figure 2: Considering two Gaussians, the theoretical KL distance (§ 4.1) and the proposed KL distance (1) are compared.

4. Experiments

4.1. Comparison to the Kullback-Leibler divergence

We look in this section at the behavior of the proposed distance (1), compared to the exact KL divergence by considering two Gaussian distributions Q_1 and Q_2 with the mean μ_1 and μ_2 respectively, and diagonal covariance matrices. The numbers of generated points are N_1 and N_2 respectively. Then, the theoretical expression of the KL-divergence is:

$$d_{KL}^{th}(Q_1, Q_2) = \frac{1}{2}(\mu_1 - \mu_2)^T(\mu_1 - \mu_2) \tag{4}$$

Figure 2 shows the values of the KL divergence (4) and the corresponding values of the proposed distance (1). Distances are reported as regards to the Euclidean distance between μ_1 and μ_2. In this figure, results are reported for

$F = 2$ features. We obtained exactly the same results for $F = 1000$ features. As expected, the closer μ_1 and μ_2, the lower the distance. Note that the tree-based KL distance (1) is closer to the theoretical KL distance (4) when the number of points increases.

For an intuitive understanding, we discuss the extreme values that are reached. If $\mu_1 = \mu_2$, we observe that our tree-based distance $\Delta(Q_1, Q_2) \neq 0$ when $d^{th}_{KL}(Q_1, Q_2) = 0$. This is due to the fact that different subsequent realizations of a given Gaussian distribution are not exactly the same. If the Euclidean distance between μ_1 and μ_2 tends towards infinity, the tree-based distance (1) never tends towards infinity. This is due to the finite number of points (N_1 and N_2) for each realization.

For instance, considering that only one leaf of the tree is reached and the Euclidean distance between μ_1 and μ_2 tends to infinity, we can easily show that the distance (1) equals $\Delta(Q_1, Q_2) = \log(M)$ where $M = N_1 = N_2$ is the number of leaves. Thus, if $N_1 = N_2 = 10$, $\Delta(Q_1, Q_2) = 2.3$, if $N_1 = N_2 = 100$, $\Delta(Q_1, Q_2) = 4.6$, if $N_1 = N_2 = 1,000$, $\Delta(Q_1, Q_2) = 6.9$, and if $N_1 = N_2 = 10,000$, $\Delta(Q_1, Q_2) = 9.2$. The extreme values in Figure 2 are then correct.

Table 1: The mean error rate and the standard deviation are reported for the three data-sets (§ 4.2). Two clustering methods are considered: the proposed clustering method (§ 3.1) that uses the distance (1) between unsupervised trees, and the bag of features that considers typical k-means with Euclidean distance between histograms of clusters.

Data	treeKL	Bag of Features
CBCL	**0.27±0.06**	0.28±0.12
ALOI	**0.06±0.04**	0.19±0.13
HPID	0.24±0.08	**0.18±0.08**
Mean	**0.19±0.06**	0.21±0.11

4.2. Unsupervised classification of images

For assessing the reliability of the proposed divergence (1), we test it on the unsupervised classification of images (§ 3.1) using three data-sets. The CBCL face and car data-set[1] contains 3 classes of images: no-face, face, and car. The Amsterdam Library of Object Images [12] (ALOI) contains $1,000$ classes of images with 24 images per class. In this paper, we only take the first

[1] http://cbcl.mit.edu/projects/cbcl/software-data-sets/

30 classes for improving the time of the experiment. The Head Pose Image Database [18] (HPID) contains 15 classes, i.e. one class for one person, each class containing 93 images.

For each image i, a set of SIFT points $\mathbf{X}_i = \{X_{i,1} \dots X_{i,K_i}\}$ is extracted [23]. A histogram with $F = 128$ features is then associated to each image. Clustering of the images is now equivalent to clustering of the distributions $\{Q_i\}_i$ of the key points bu using the proposed method.

Note that the experiments only deal with two-class classification. All pairs of classes are considered and the reported results refer to the mean error rate. A 50-iteration cross validation is used to extract the mean error rate and the standard deviation. At each iteration, for each class, 20 objects are sampled from the database. The error e is defined as a function of pairwise error: $e = 1 - \frac{TP+TN}{TP+FP+FN+TN}$ where TP denotes a true positive decision, TN denotes a true negative decision, FP denotes a false positive decision, and FN denotes a false negative decision.

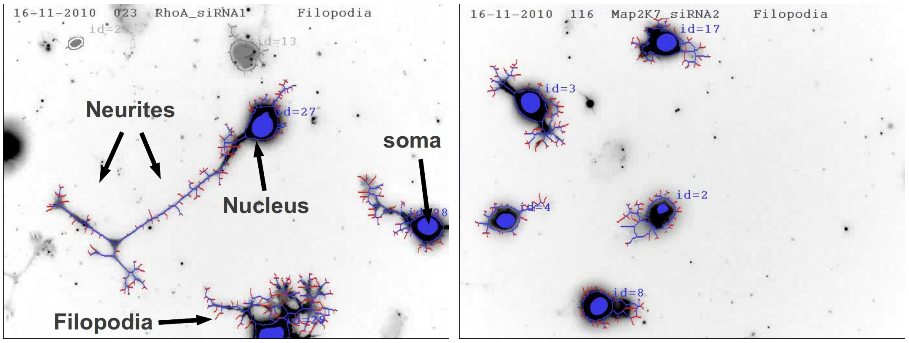
Classification performances are reported in Table 1. "Unsupervised Tree" is the proposed tree-based clustering method. "Bag of Features" is the usual bof that uses the Euclidean distance between the histograms of the clusters. For bof, we have found that the optimal number of clusters is 30.

The proposed distance is better for two out of three data-sets and the standard deviation is better on average. Results show that our proposed method can outperform the baseline, i.e. the bof. Finding one data-set for which the proposed method has the better results is sufficient to say that the method is interesting. Also, in comparison to bof that consider a sensitive parameters, i.e. the number of clusters, the proposed distance does not consider parameter and the results obtained in Table 1 are steady.

Note that the classification performance could have been improved by extracting other features from the images. We have chosen the SIFT features which correspond perfectly to our application: a set of points in a feature space.

4.3. Classification of neurons in videos

Understanding cell morphologies and cell dynamics remains a difficult challenge in biology [14]. In the field of neurobiology, researchers have observed links between the static image of neurons and their genotypes [13]. Recent works in oncology have shown that studying the cell dynamics provides important information about its genotype [15, 16]. In the same line of thinking, we propose to study if the neuron morpho-dynamics in videos depend on their genotype characteristics. For instance, Figure 3-(a) shows one video of neurons for which the gene RhoA has been knocked-down which leads to longer neurites. Figure 3-(b) shows one video of neurons for which the gene Map2K7 has been knocked-down which leads to shorter neurites and faster protrusion and retraction process.


16-11-2010 023 RhoA_siRNA1 Filopodia
id=2
id=13
Neurites
id=27
soma
Nucleus
Filopodia
16-11-2010 116 Map2K7_siRNA2 Filopodia
id=17
id=3
id=2
id=4
id=8

(a) (b)

Figure 3: (a) Video of neurons for which the gene RhoA has been knocked down. (b) Video of neurons for which the gene Map2K7 has been knocked down.

Automated tools are needed for helping and assisting biologists in the analysis of the videos. One method for pointing out the difference between genotypes is to measure the ability for classifying the neurons. Each neuron is tracked by first detecting soma and nucleus in each frame of the video, and then neurons in different frames are fused together [5]. Then, four types of features are associated to each neuron:

- Three video features: the entropy of the frame intensities, the intensity divergence between frames, and the pixel-based intensity divergence between frames. Let $\mathbf{X}_{vn}^1 \in \mathbb{R}^{(F_v \times 3)}$ denote these features associated to the neuron n in the video v. F_v denotes the number of frames in the video v.

- 30 global neuron features such as the nucleus time expanding, the nucleus time contracting, the neurites time expending, etc. Let $\mathbf{X}_{vn}^2 \in \mathbb{R}^{30}$ denote these features associated to the neuron n in the video v.

- 37 "by frame" neuron features such as the total cable length of neurites in each frame, the total number of filopodia in each frame, the soma eccentricity in each frame, etc. Let $\mathbf{X}_{vn}^3 \in \{\mathbb{R}^{37}, \mathbb{R}^{37}, \ldots\}$ denote these features associated to each frame of the neuron n in the video v.

- 25 "by frame" neurite features such as the number of branches in each neurite in each frame, the number of filopodia in each neurite in each frame, the length of each neurite in each frame, etc. Let $\mathbf{X}_{vn}^4 \in \{\mathbb{R}^{(N_{vn1} \times 25)}, \mathbb{R}^{(N_{vn2} \times 25)}, \ldots\}$ denotes these features associated to the neuron n in the video v such that N_{vnf} denotes the neurite number of the neuron n in the frame f in the video v.

Supervised classification can be applied to investigate if the morphodynamics of the neurons depend on the neuron genotypes. The objects to classify are the neurons in the videos. 100 experiments were performed for computing the average and standard deviation of a classification rate. At each iteration, data are separated into training data-set (7 videos for each class) and test data-set (3 videos for each class). The classifiers are built using the training data-set and the mean classification rate is estimated on the test data-set.

Six classes are considered. The class "Control" contains neurons which are not genetically modified and the other classes ("RhoA", "SrGap2", "Net", "Map2K7", and "Trio") corresponds to categories of neurons whose genotype have been modified. An experiment consists of doing comparisons between the class "Control", and one with modified genotype.

Four classifiers are used for classifying the neurons:

- **RF**: The bof are computed from $\{\mathbf{X}_{vn}^1\}_{vn}$, $\{\mathbf{X}_{vn}^2\}_{vn}$, $\{\mathbf{X}_{vn}^3\}_{vn}$, and $\{\mathbf{X}_{vn}^4\}_{vn}$, independently. The bof vectors $X_{vn} \in \mathbb{R}^{(3+30+37+25)}$ are obtained by concatenation. Based on X_{vn}, random forests are used for classifying the neuron n in the video v.

- **linSVM**: The bof vectors X_{vn} are built as previously. Based on X_{vn}, a linear SVM is used for classifying the neuron n in the video v.

- **rbfSVM**: The bof vectors X_{vn} are built as previously. Based on X_{vn}, a Gaussian SVM is used for classifying the neuron n in the video v.

- **treeKL**: Distance between the neuron i and the neuron j is computed by using the tree-based KL distance (1) and by combining the four information levels as follows: $d_{ij} = \Delta(Q_i^1, Q_j^1) + \Delta(Q_i^2, Q_j^2) + \Delta(Q_i^3, Q_j^3) + \Delta(Q_i^4, Q_j^4)$ where Q_i^n denotes the probability density function of the

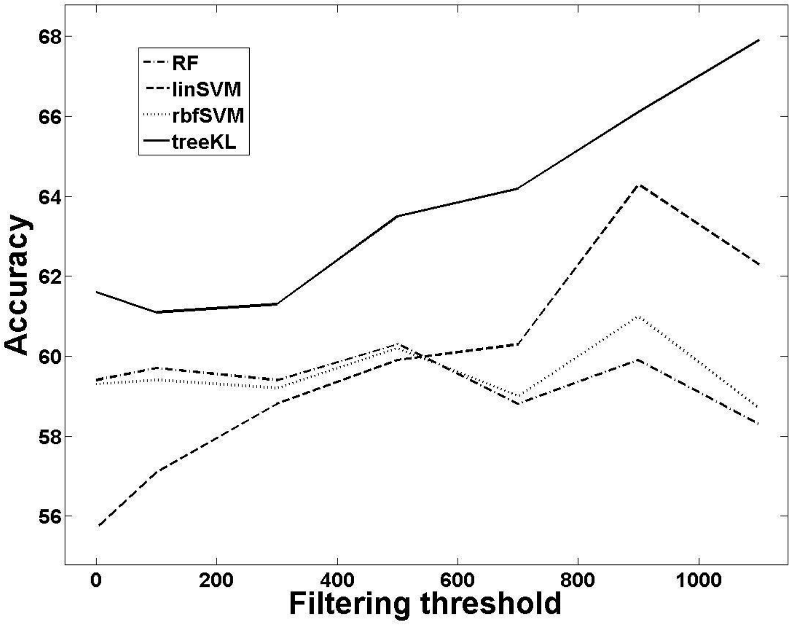


Figure 4: The classification accuracy is reported as a function of the filtering threshold. The accuracy is reported for bag of feature (bof) associated to Random Forest (RF), for bof associated to linear SVM (linSVM), for bof associated to rbf SVM (rbfSVM), and for our proposed method (treeKL).

points $\mathbf{X}_i^n$. Then, as presented in § 3.2, we use SVM with the following kernel: $K_{ij} = exp(-d_{ij}^2/\sigma)$.

To reduce the noise induced by neuron death, we filter neurons which have short neurites. Figure 4 shows the classification accuracy as a function of the filtering threshold. The provided values refer to average classification over all experiments. The results confirm the good classification performances of the proposed distance, in compliance with the performances of the unsupervised classification in § 4.2. On average, the proposed method outperforms the bof approaches. Also, the tree-based classifier (treeKL) outperforms the other classifiers 44.7% of the time. RF outperforms 23.4% of the time, linSVM outperforms 14.9% of the time, and rbfSVM outperforms 17% of the time.

We conclude that the genotype of the neurons do modify the morphodynamic of the neurons. This is proved by the fact that the groups of neurons that have been genetically modified (“RhoA”, “SrGap2”, “Net”, “Map2K7”, and “Trio”) can be discriminated from to the “Control” neurons.

Table 2: The classification accuracy is reported for each class of genes. The accuracy is reported for bag of feature associated to Random Forest (RF), for bof associated to linear SVM (linSVM), for bof associated to rbf SVM (rbfSVM), and for our proposed method (treeKL).

	RF	linSVM	rbfSVM	treeKL
RhoA siRNA 1	0.50±26	0.52±21	0.53±24	**0.56±22**
RhoA siRNA 2	**0.74±21**	0.66±22	0.69±21	0.72±20
RhoA siRNA 3	**0.69±23**	0.62±22	0.62±21	0.65±22
Map2K7 siRNA 1	**0.68±20**	0.60±23	0.61±21	0.62±23
Map2K7 siRNA 2	0.49±27	0.53±24	0.51±25	**0.55±23**
Map2K7 siRNA 3	0.61±26	0.56±25	0.55±26	**0.65±26**
Net	0.64±24	0.64±22	0.63±20	**0.68±22**
SrGap2 siRNA 3	0.58±23	0.58±22	0.57±22	**0.61±23**
SrGap2 siRNA 2	**0.56±23**	0.55±21	0.55±19	0.55±22
SrGap2 siRNA 3	0.61±22	0.61±22	0.58±19	**0.62±24**
Trio siRNA 1	0.49±25	0.57±23	0.57±21	**0.57±21**
Trio siRNA 2	0.53±23	0.51±23	0.52±20	**0.57±24**
average	0.59±24	0.58±23	0.58±21	**0.61±23**

For knocking down a gene, biologists use siRNAs [13]. This method is not as accurate as waited, such that several genes can be knocked down at the same time. For illustrating this purpose, in table 2 we show the accuracy between control and each other class. First, we note that our proposed distance (treeKL) outperforms the other distances in average. Second, we note that performances are very sensitive to the corresponding siRNA. For instance, the accuracy can vary from nearly 50% in average (RhoA siRNA 1) to nearly 70% in average (RhoA siRNA 2). This result provides a tool for biologists who want to evaluate the accuracy of siRNAs. For instance, regarding the previous example, the conclusion is that “siRNA 1” is not a good drug for RhoA.

5. Conclusion

We have proposed a new similarity measure between sets of points in a large-dimension space. This measure relies on the KL-divergence between empirical distributions over the leaves of trees build for each set of points independently. It avoids the usual fine tuning of density model parameters, and leverages the very good behavior of decision trees in high dimension. Synthetic experiments show that in small dimension, this distance is monotonic with the KL divergence of the underlying densities.

We have demonstrated experimentally how it can be applied to both unsupervised and supervised learning. Both on image clustering and neuron dynamic classification in videos, it outperforms baselines using bag-of-features.

Acknowledgments

This work was supported by the Swiss National Science Foundation under grant CRSII3 127456 - Understanding Brain Morphogenesis, and by the European Community's 7th Framework Programme under grant agreement 247022 - MASH.

References

[1] Zhang G., and Wang Y., Hierarchical and discriminative bag of features for face profile and ear based gender classification, IJCB, 2011

[2] Goldberger, J. and Gordon, S. and Greenspan, H., An Efficient Image Similarity Measure Based on Approximations of KL-Divergence Between Two Gaussian Mixtures, ICCV, 2003

[3] Pele, O. and Werman, M., The quadratic-Chi Histogram distance family, European Conference on Computer Vision, 2010

[4] Dempster, A. and Laird, N. and Rubin, D., Maximum likelihood from incomplete data via the EM algorithm, Journal of the Royal Statistic Siciety, 39, 1-38, 1977

[5] Gonzáles, G. and all., Steerable features for statistical 3D dentrite detection, International Conference on Medical Image Computing and Computed Assisted Intervenction, 2009

[6] Mahalanobis, P.C., On the generalised distance in statistics, Proceedings of the National Institute of Sciences of India, 2, 1, 49-55, 1936

[7] Parzen, E., On estimation of a probability density function and mode, Annals of Mathematical Statistics, 33, 1065-1076, 1962

[8] Breiman, L., Random forest, Machine Learning, 45, 5-32, 2001

[9] Csurka, G. and Dance, C. and Fan, L., Visual categorization with bags of keypoints, ECCV Workshop Statistical Learning in Computer Vision, 59-74, 2004

[10] Duan, L. and Xu, D. and Tsang, I.W. and Luo, J., Visual event recognition in videos by learning from web data, Conference on Computer Vision and Pattern Recognition, 2010

[11] Fei-Fei, L. and Fergus, R. and Torralba, A., Recognizing and learning object categories, Conference on Computer Vision and Pattern Recognition, 2007

[12] Geusebroek, J.M. and Burghouts, G.J. and Smeulders, A.W.M., The Amsterdam Library of object images, International Journal of Computer Vision, 61, 1, 103-112, 2005

[13] Pertz, O. and all., Spatial mapping of the neurite and soma proteomes reveals a functional Cdc42/Rac regulatory network, The National Academy of Sciences of the USA, 105, 1931-1936, 2008

[14] Bakal, C. and Aach, J. and Church, G. and Perrimon, N., Quantitative morphological signatures define local signaling networks regulating cell morphology, Science, 316, 5832, 1753-1756, 2007

[15] Held, M. and all., CellCognition: time-resolved phenotype annotation in high-throughput live cell imaging, Nature Methods, 7, 747-754, 2010

[16] Neumann, B. and all., Phenotypic profiling of the human genome by time-lapse micorscopy reveals cell division genes, Nature, 464, 721-727, 2010

[17] Schölkopf, B. and Smola, A., Learning with kernel, The MIT Press, 2002

[18] Gourier, N. and Hall, D. and Crowley, J.L., Estimating face orientation from robust detection of salient facial features, International Workshop on Visual Observation of Deictic Gestures, 2004

[19] Hooker, G., Diagnosing extrapolation: tree-based density estimation, Association for Computing Machinery - SIGKDD Conference, 569-574, 2004

[20] Karakos, D. and all., Unsupervised classification via decision trees: an information-theoretic perspective, International Conference on Acoustics, Speech, and Signal Processing, 5, 1081-1084, 2005

[21] Quinlan, J., C4.5: Programs for machine learning, Morgan Kaufmann Publisher, 1993

[22] Laptev, I. and Marszalek, M. and Schmid, C. and Rozenfeld, B., Learning realistic human actions from movies, Conference on Computer Vision and Pattern Recognition, 2008

[23] Lowe, D. G., Object recognition from local scale-invariant features, International Conference on Computer Vision, 2, 1150-1157, 1999

[24] Moosman, F. and Nowak, E. and Jurie, F., Randomized clustering forests for image classification, Transaction on Pattern Analysis and Machine Intellingence, 30, 9, 2008

[25] Reichart, R. and Rappoport, A., Unsupervised induction of labeled parse trees by clustering with syntactic feature, International Conference on Computational Linguistics, 721-728, 2008

[26] Schmidberger, G. and Frank, E., Unsupervised discretization using tree-based density estimation, Conference on Principles and Practice of Knowledge Discovery in Databases, 2005