Application of Localized Diffusion Folders to Speech/Singing Discrimination

Yan Michalevsky

Technion - Israel Institute of Technology

June 22, 2010
Outline

1 Introduction

2 Algorithm
   - Bottom Level Hierarchy
   - Adaptive Gaussian Kernel
   - Partitioning into Local Neighborhoods
   - Voronoi Diagrams Fusion
   - Upper Level Construction

3 Application
   - Speech/Singing Discrimination by Pitch
   - Using the Adaptive Gaussian Kernel with Diffusion Maps
The task of discrimination between Speech and Singing samples from a given corpus could be interpreted as a clustering problem:

**Problem Definition**

The feature vector of each audio sample represents a point in $\mathbb{R}^n$ where $n$ is the number of features. We want to assign each point to one of two clusters which correspond to Speech and Singing.
Dimensionality Reduction Techniques

Maybe with LDF we can do even better...

[Application of LDF to Speech/Singing Discrimination]
Localized Diffusion Folders

**LDF**

Localized Diffusion Folders is a hierarchical clustering method for multi-dimensional datasets. Diffusion Folders stands for the multi-level partitioning into local neighborhoods.

The LDF method preserves local neighborhoods while eliminating invalid connections between areas that should not relate, created by noisy samples.
For $t = 1$ (number of diffusion steps) the affinity matrix reflects direct connections between points in the dataset. Incrementing $t$ is equivalent to performing several diffusion steps. The problem with the global approach is that increasing $t$ results in noise in the affinity matrix, and connections between unrelated points are created.
Advantages of LDF

- Consideration of local geometry.
- Adaptivity to local sample density.
- Avoiding invalid connections with per-step affinity matrix.
- De-noising (“shake-n-bake”).
Input

The input is a matrix $m \times n$ where $m$ is the number of data points and $n$ is the number of features.

Normalization

Since each data point is a vector of heterogeneous features we normalize them with a $\log$ function to bring them to a common scale.

We obtain the normalized matrix $A$. 

Yan Michalevsky [Application of LDF to Speech/Singing Discrimination]
We construct the similarity matrix $\tilde{A}$ using an Euclidean metric. If $a_i$ denotes the row $i$ of the matrix $A$ then

$$\tilde{a}_{ij} = \sqrt{(a_i - a_j) \cdot (a_i - a_j)^T}$$

It possible to use other metrics as well:

- Weighted Euclidean metric.
- Cosine distance metric.
- Mahabolis distance metric, with covariance matrix or feature matrix as $\Sigma$. 

Yan Michalevsky

[Application of LDF to Speech/Singing Discrimination]
Let’s take a look at the Gaussian kernel

\[ K_{ij} = e^{-\frac{\tilde{a}_{ij}}{\epsilon}} \]

Already far points are pushed away, while close points are pulled together.
The scale control \( \epsilon \) is fixed for all entries in \( \tilde{A} \).

Obviously not optimal since it doesn’t take into account local geometry and densities in different areas.
Adaptive Gaussian Kernel $K$

Key Idea

Automatic determination of the scale control for each point.

Data points in dense areas have a large weight and data points in sparse areas have a small weight. The per-point scale is given by

$$\omega_i^\epsilon = \int_A e^{-\frac{\tilde{a}_{ij}}{\epsilon}} d\mu(a_j)$$

where $A$ is the set of all points and $\epsilon$ is an initial scale.
Adaptive Gaussian Kernel $K$

Since we construct a pairwise-affinity matrix we define the pairwise weight function as the geometric average of the weights

$$\Omega_{ij}^\epsilon = \sqrt{\omega_i^\epsilon \omega_j^\epsilon}$$

Now we construct the Adaptive Gaussian Kernel

$$K_{ij} = e^{-\frac{\tilde{a}_{ij}}{\Omega_{ij}^\epsilon}}$$

The Gaussian Kernel is then normalized into a Markov transition matrix

$$P_{ij}^t = \frac{K_{ij}}{\sqrt{\sum_{q=1}^{m} K_{iq}} \cdot \sqrt{\sum_{q=1}^{m} K_{jq}}}$$
Using the affinity matrix $P^t$ we perform an initial partitioning of the points into non-overlapping sets:
Local Neighborhoods

Using the affinity matrix $P^t$ we perform an initial partitioning of the points into non-overlapping sets:

1. Choose a random point $a_i$ from the dataset.
Local Neighborhoods

Using the affinity matrix $P^t$ we perform an initial partitioning of the points into non-overlapping sets:

1. Choose a random point $a_i$ from the dataset.
2. Denote by $N_\epsilon (a_i) \triangleq \left\{ a_j : P^t_{ij} > \epsilon, i \neq j \right\}$ the neighborhood of $a_i$. 

Yan Michalevsky  
[Application of LDF to Speech/Singing Discrimination]
Local Neighborhoods

Using the affinity matrix $P^t$ we perform an initial partitioning of the points into non-overlapping sets:

1. Choose a random point $a_i$ from the dataset.
2. Denote by $N_\epsilon (a_i) \triangleq \{ a_j : P^t_{ij} > \epsilon, i \neq j \}$ the neighborhood of $a_i$.
3. Repeat steps 1 and 2 for points that haven’t been assigned to a folder yet.
Local Neighborhoods

- Once all points have been assigned we verify that each point is associated with its nearest folder.
- For each folder we calculate its centroid and re-assign each point to the folder with the nearest centroid, i.e. with the maximum average affinity.
- After this stage some folders might end up empty, and therefore they are removed.
- During the reassignment the folders’ content is changed.
- This process is repeated several times until the assignment of points to folders stops changing.

The result is a Voronoi diagram denoted by $D^t$. The diagram is affected by the random selection of the points.
Shake n Bake

The constructed Voronoi diagram was affected by the random selection of the points. Hence, we repeat this process \( r \) times to obtain \( r \) different LDF systems. The set of multiple systems is denoted by

\[
\tilde{D}^t \triangleq \left\{ D^{(t,k)} : k = 1, \ldots, r \right\}
\]

The systems are fused to obtain a cleaner affinity matrix \( \hat{P}^t \).

Fusion Method

Two points are close to each other if they are in the same folder in different systems.
The following metric is applied to the points

\[
d(a_i, a_j) = \begin{cases} 
0 & a_i = a_j \\
\frac{1}{2} & a_i \neq a_j, a_j \in D_q^t(a_i) \\
1 & a_i \neq a_j, a_j \notin D_q^t(a_i)
\end{cases}
\]

where \( D_q^t(a_i) \) is the folder that contains the point \( a_i \).
To fuse the multiple diagrams we define the distance \( d_\mu (a_i, a_j) \) as the average distance of all systems. The localized affinity matrix is defined as

\[
\hat{P}^t_{ij} \triangleq 1 - d_\mu (a_i, a_j)
\]

It is again normalized into a Markov transition matrix as previously described, and once again partitioned into folders. These folders and the localized affinity matrix are the input to the next stage.
Selection of the sub-matrix $\hat{P}_{kl}^{t-1}$ of the affinities between the points in $\hat{S}_k^{t-1} \cup \hat{S}_l^{t-1}$.
Upper Level Construction

The localized affinity sub-matrix is raised by the power of $2^t$

$$Q_{kl}^t \triangleq \hat{P}_{kl} (t-1)^2$$

Generating the sub-matrix $\hat{Q}_{kl}^t$
Upper Level Construction

The affinity $P_{kl}^t$ between the folders $\hat{S}_{k}^{t-1}$ and $\hat{S}_{l}^{t-1}$ is defined by one of the following metrics:

- Fastest random runner.
- Slowest random runner.
- Average random runner.

The affinity matrix $P^t$ is normalized into a Markov transition matrix and the whole process is repeated with $P^t$ as input, when each folder is taken as a point.
Example

[Application of LDF to Speech/Singing Discrimination]
Speech/Singing Discrimination by Pitch

Real labels (left) and clustering with LDF (right)
The Adaptive Gaussian Kernel presented in the context of LDF is a possible replacement to the simple Gaussian kernel we used with Diffusion Maps. Slightly better results were achieved when using this kernel with Diffusion Maps than with the traditional Gaussian kernel.