Lecture 5: t-SNE and Consensus Clustering, November 21, 2017

Lecturer: Prof. Ron Shamir *Scribe: Tomer Wolfson*

5.1 t-SNE

5.1.1 Introduction

t-distributed stochastic neighbor embedding (t-SNE), by van der Maaten and Hinton [12] is a machine learning algorithm for dimensionality reduction. It is a nonlinear dimensionality reduction technique that embeds high-dimensional datapoints into a space of two or three dimensions, which can then be visualized in a scatter plot. Each datapoint is mapped to a *map-point*, where the mapping is designed such that similar datapoints are modeled by nearby map-points and dissimilar datapoints are modeled by distant map-points.

The t-SNE algorithm is comprised of two main stages. First, t-SNE constructs a probability over pairs of the datapoints such that similar points have a high probability of being picked by each other, while dissimilar datapoints have an extremely low probability of being picked. Second, t-SNE defines a similar probability distribution over the map-points. It then fits the locations of the points in the map to minimize the *Kullback-Leibler* (KL) divergence between the two distributions.

5.1.2 Stochastic Neighbor Embedding

Stochastic Neighbor Embedding (SNE), by Hinton and Roweis [13], starts by converting the high dimensional Euclidean distances between data-points into conditional probabilities that represent similarities. Given a set of high-dimensional data-points x_1,...,x_n, the similarity of data-point x_j to x_i is the conditional probability $p_{j|i}$, that x_i would pick x_j as its neighbor. We assume that neighbors are picked in proportion to their probability density under a Gaussian centered at x_i with variance σ_i. For nearby points, $p_{j|i}$ is relatively high, whereas for far away points it will be close to zero (for reasonable values of σ_i). Mathematically, the conditional probability is given by

$$p_{j|i} = \frac{exp(-\|x_i - x_j\|^2/2\sigma_i^2)}{\sum_{k \neq i} exp(-\|x_i - x_k\|^2/2\sigma_i^2)}$$

Because we are only interested in modeling pairwise similarity $p_{i|i}$ is set to zero.
For the low dimensional counterparts y_i and y_j of x_i and x_j, we similarly compute a conditional probability $q_{j|i}$. For the Gaussian used to compute $q_{j|i}$ we set the variance to $\frac{1}{\sqrt{2}}$. We

now model the similarity of map point y_j to y_i by

$$q_{j|i} = \frac{exp(-\|y_i - y_j\|^2)}{\sum_{k \neq i} exp(-\|y_i - y_k\|^2)}$$

Once again we set $q_{i|i}$ to zero.
If map points y_i and y_j correctly model the similarity between the high dimensional data-points, the conditional probabilities $p_{j|i}$ and $q_{j|i}$ will be equal. Motivated by this observation, SNE aims to find a low-dimensional data representation that will minimize the mismatch between $p_{j|i}$ and $q_{j|i}$. To measure the faithfulness with which $q_{j|i}$ models $p_{j|i}$ we shall use the *Kullback-Leibler divergence*.

Definition *Kullback-Leibler divergence*
Given two discrete probability distributions P and Q the KL divergence from Q to P is defined to be

$$KL(P||Q) = -\sum_i P_i \log \frac{P_i}{Q_i}$$

We assume P_i and Q_i are both greater than zero for all i. The KL divergence is in fact the expectation of the logarithmic difference between probabilities P and Q, where the expectation is taken using the probabilities P.

SNE minimizes the sum of KL divergences over all data-points using a *gradient descent* method. The cost function C is given by

$$C = \sum_i KL(P_i||Q_i) = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}$$

where P_i represents the conditional probability distribution over all given data-points given x_i, and Q_i represents the conditional probability distribution over all map points given y_i. Because KL divergence is not symmetric, different errors in the pairwise distances in the low-dimensional map are not weighted equally. In particular, there is a large cost for using far map points to represent data-points that are close (i.e, using a small $q_{j|i}$ to model a large $p_{j|i}$). In contrast, there is a small cost for using nearby map points to represent widely separated data-points. The small cost comes from wasting some of the probability mass in the relevant Q distributions. In other words, the SNE cost function focuses on retaining the local structure of the data in the map.
We now show how SNE selects the variance σ_i of the Gaussian that is centered over each x_i. The value of σ_i induces a probability distribution P_i over all the other data-points. SNE performs a binary search for the value of σ_i that produces a P_i with a fixed *perplexity*, specified by the user. The perplexity is defined as

$$Perp(P_i) = 2^{H(P_i)},$$

where $H(P_i)$ is the *Shannon entropy* of P_i measured in bits

$$H(P_i) = -\sum_j p_{j|i} \log_2 p_{j|i}.$$

The perplexity is a smooth measure of the effective number of neighbors. The performance of SNE was tested and shown to be robust to changes in the perplexity in the range 5 - 50. We turn back to the cost function C that we wish to minimize by finding the appropriate parameters.

$$C = \sum_i KL(P_i||Q_i) = \sum_i \sum_j p_{j|i} \log \frac{p_{j|i}}{q_{j|i}}$$

The minimization is performed using a *gradient descent* method. We optimize the function by taking steps proportional to the negative of the function gradient. The gradient of C w.r.t. y_i,

$$\frac{\partial C}{\partial y_i} = 2 \sum_j (p_{j|i} - q_{j|i} + p_{i|j} - q_{i|j})(y_i - y_j).$$

Since gradient descent does not guarantee convergence to the optimal solution (it might get stuck at a local optimum), it is common to run the optimization several times in order to find the appropriate parameters.

5.1.3 t-Distributed Stochastic Neighbor Embedding

SNE constructs reasonably good visualizations but is hampered by a cost function that is difficult to optimize and by the *crowding problem* discussed later. The *t-Distributed Stochastic Neighbor Embedding* (t-SNE) is a technique that aims to alleviate the problems of the original SNE. The cost function of t-SNE differs from that of SNE in two ways: (1) it uses a symmetric version of the SNE cost with simpler gradients and (2) it uses a *Student-t* distribution rather than Gaussian to compute the similarity between map-points. t-SNE employs a heavy-tailed distribution in the low-dimensional space to alleviate both the crowding and optimization problems of SNE. We will first introduce the symmetric version of SNE followed by presenting the crowding problem and conclude with presenting the t-SNE algorithm.

5.1.4 Symmetric SNE

An alternative to minimizing the sum of the KL divergences between $p_{j|i}$ and $q_{j|i}$ (conditional probabilities) is to minimize a single KL divergence between a *joint* probability distribution P, in the high-dimensional space and a joint distribution Q, in the low-dimensional space:

$$C = KL(P||Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

where we set p_{ii} and q_{ii} to zero. We refer to this as *symmetric SNE*, because it has the property that $p_{ij} = p_{ji}$ and $q_{ij} = q_{ji}$ for all i, j. In symmetric SNE the pairwise similarities in the low-dimensional map q_{ij} are given by

$$q_{ij} = \frac{exp(-\|y_i - y_j\|^2)}{\sum_{k \neq l} exp(-\|y_k - y_l\|^2)}$$

An intuitive but *problematic* solution for high-dimensional similarity would be to define p_{ij} as

$$p_{ij} = \frac{exp(-\|x_i - x_j\|^2/2\sigma^2)}{\sum_{k \neq l} exp(-\|x_k - x_l\|^2/2\sigma^2)}$$

This definition causes problems when a data-point x_i is an outlier (i.e., all pairwise distances $\|x_i - x_j\|^2$ are large for x_i). For such an outlier, the values of p_{ij} are extremely small for all j, so the location of its map point y_i has little effect on the cost function. This results in map point positions not being well determined by other map points positions. The problem is addressed by defining p_{ij} to be the symmetric conditional probability,

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$$

This ensures $\sum_j p_{ij} > \frac{1}{2n}$ for all data-points x_i, so that each data-point makes a significant contribution to the cost function. The advantage of symmetric SNE is the simpler form of its gradient, which is faster to compute. The gradient of symmetric SNE is given by

$$\frac{\partial C}{\partial y_i} = 4 \sum_j (p_{ij} - q_{ij})(y_i - y_j).$$

Thus symmetric SNE presents an easier optimization for the cost function.

5.1.5 The Crowding Problem

Consider a set of data-points that lie on a two-dimensional curved manifold which is approximately linear on a small scale, and which is embedded within a higher-dimensional space. It is possible to model small pairwise distances between data-points fairly well in a 2-dimensional map. Suppose that the manifold has 10-intrinsic dimensions (for example images of the handwritten digits 0-9) and is embedded within a space of much higher dimensionality. The pairwise distances in a 2-dimensional map will not be able to faithfully model the distances between points on the 10-dimensional manifold. For instance, in 10 dimensions, it is possible to have 11 data-points that are mutually equidistant but there is no way to

model this faithfully in a 2-dimensional map. Another problem is the different distribution of pairwise distances in the two spaces. The volume of a sphere centered on data-point i scales as r^m, where r is the radius and m the dimensionality of the sphere. If the data-points are uniformly distributed in the region around i in the 10-dimensional manifold, and we try to model distances from i to the other data-points in the 2-dimensional map, we get the following *"crowding problem"*: the area of the 2-dimensional map available to accommodate moderately-distant data-points will not be large enough compared with the area available to accommodate nearby data-points. If we want to model the small distances accurately in the map, most of the points that are at a moderate distance from data-point i will have to be placed much too far away in the 2-dimensional map. In SNE the connection of data-point i to each of these too-distant map points will exert a very small attractive force. Though small in force, the very large number of these attractive forces crushes together the points in the center of the map, preventing gaps from forming between natural clusters. Thus harming the clustering process.

5.1.6 Student t-distribution

We addresses the *crowding problem* by using the fact that symmetric SNE is actually matching the joint probabilities of pairs of data-points in the high and low dimensional spaces rather than their distances. In the high-dimensional space we convert distances into probabilities using a Gaussian distribution. In the low-dimensional map, we use the Student t probability distribution, which has much heavier tails than a Gaussian (Figure 5.1) to convert distances into probabilities. This allows moderate distances in the high-dimensional space to be faithfully modeled by a much larger distance in the map, thus eliminating the unwanted attractive forces between map points that represent moderately distant data-points.

Using this distribution, the joint probabilities q_{ij} are defined as

$$q_{ij} = \frac{(1 + \|y_i - y_j\|^2)^{-1}}{\sum_{k \neq l} (1 + \|y_k - y_l\|^2)^{-1}}$$

We use a Student t-distribution with a single degree of freedom because it has the property that $(1+\|y_i-y_j\|^2)^{-1}$ approaches an inverse square law for large pairwise distances $\|y_i-y_j\|$ in the low-dimensional map. This makes the map's representation of joint probabilities almost invariant to changes in the scale of the map for map-points that are far apart. It also means that large clusters of points that are far apart interact in the same way as individual points, so the optimization will operate in the same way at all scales.

The gradient of the Kullback-Liebler divergence between P and the Student-t based joint

probability distribution Q (computed from q_{ij}) is given by

$$\frac{\partial C}{\partial y_i} = 4 \sum_j (p_{ij} - q_{ij})(y_i - y_j)(1 + \|y_i - y_j\|^2)^{-1}.$$

For a derivation of the gradient, see [12].

Figure 5.2 shows the gradients between two low-dimensional data-points y_i and y_j as a function of $\|x_i - x_j\|$ and $\|y_i - y_j\|$ for SNE and for t-SNE. In the figures, positive values of the gradient represent an attraction between the y_i and y_j, whereas negative values represent a repulsion between the two points. We observe two advantages of the t-SNE gradient over the gradient of SNE. First, the t-SNE gradient strongly repels dissimilar data-points that are modeled by a small pairwise distance in the low-dimensional representation. SNE has such a repulsion as well, but its effect is minimal compared to the strong attractions elsewhere in the gradient. Second, although t-SNE introduces strong repulsions between dissimilar data-points that are modeled by small pairwise distances, these repulsions do not go to infinity.

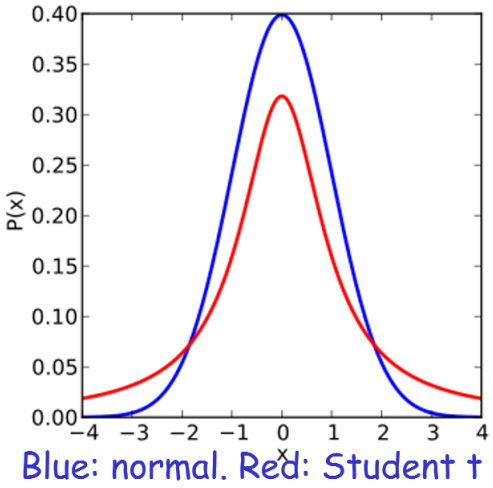
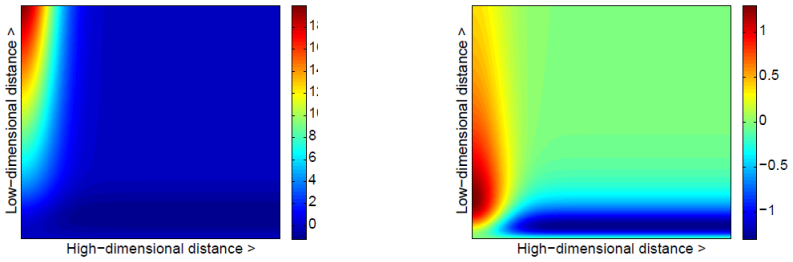


Figure 5.1: Comparing the Student t-distribution (red) with the normal distribution (blue). Note the heavier tail of the Student-t distribution.



(a) Gradient of SNE. (c) Gradient of t-SNE.

Figure 5.2: Gradients of SNE and t-SNE as a function of the pairwise Euclidean distance between two points in the high-dimensional and the pairwise distance between the points in the low-dimensional data representation. Note the t-SNE gradient displays strong repulsion for data-points mapped to short distances.

5.1.7 Algorithm

Algorithm: **t-Distributed Stochastic Neighbor Embedding**

- **Data:** data set $X = \{x_1, ..., x_n\}$
  cost function parameters: perplexity $Perp$,
  optimization parameters: number of iterations T, learning rate η, momentum $\alpha(t)$.
- **Results:** low-dimensional data representation $Y^{(T)} = \{y_1, ..., y_n\}$.
- **begin**
  1. compute pairwise affinities $p_{j|i}$ with perplexity $Perp$
  2. set $p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$
  3. sample initial solution $Y^{(0)} = \{y_1, ..., y_n\}$ from $N(0, 10^{-4}I)$
  4. **for** $t = 1$ **to** T **do**
     (a) compute low-dimensional affinities q_{ij}
     (b) compute gradient $\frac{\partial C}{\partial Y}$
     (c) set $Y^{(t)} = Y^{(t-1)} + \eta \frac{\partial C}{\partial Y} + \alpha(t)(Y^{(t-1)} - Y^{(t-2)})$

The time complexity of t-SNE is $O(n^2)$ per iteration, and the space complexity is $O(n^2)$.

Figure 5.3 shows 2-dimensional images created by t-SNE and three other popular dimension reduction algorithms on a real dataset.

5.1.8 t-SNE for single-cell data

New technologies can measure dozens of parameters in individual cells. Interpreting such data requires clear methods of visualization. The paper of Amir et al. (2013) [14] presents *viSNE*, a visualization tool that uses t-SNE to map high-dimensional cytomery data onto two dimension. Amir et al. applied the method to measurements of 13 cell markers over several thousands of single cells. They used t-SNE to map the high-dimensional data points to a 2-dimensional space. Results in Figure 5.4 show that the distinct "clouds" in the map indeed correspond to distinct cell types.

The paper also used viSNE to map healthy and cancerous bone marrow samples (Figure 5.5). The mapping distinguishes between cells of the two cancer (ALL) samples, and between the normal and cancer samples. The two normal samples are not well separated, which is expected since they reflect the same phenotype.

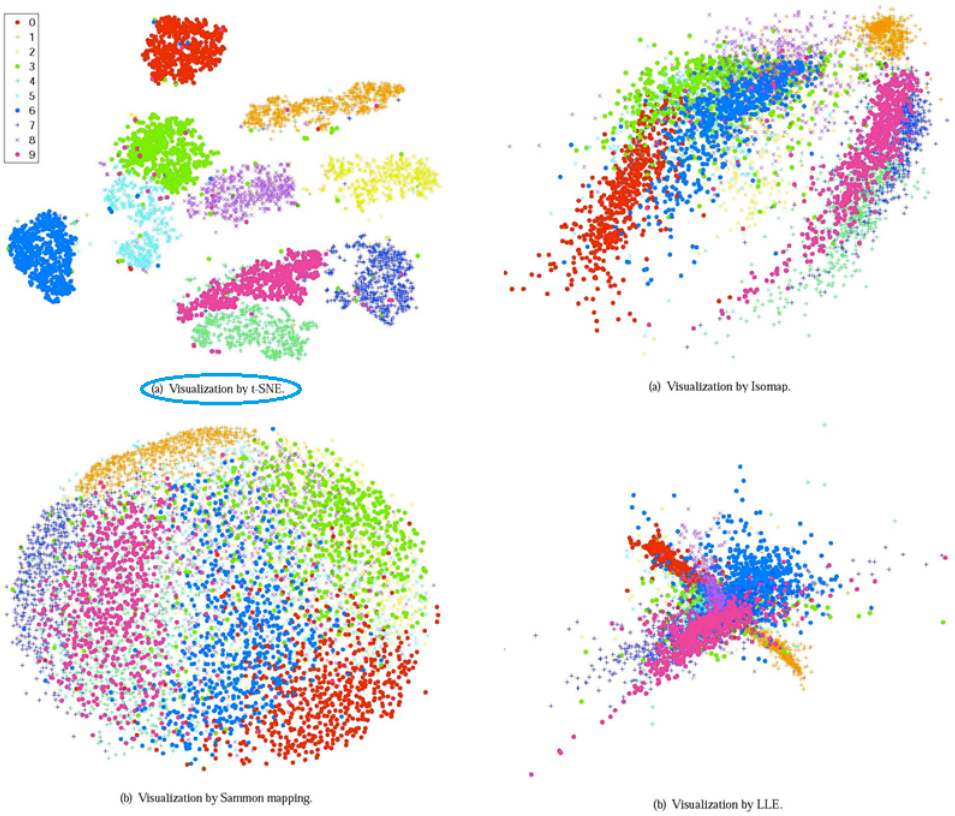


Figure 5.3: Visualization of 6,000 handwritten digits from the MNIST data set. Each digit is represented as 28×28 pixels. Each color representing a different digit. Comparing 4 different methods. t-SNE visualization (circled) provides clear separation between datapoints of different digits.

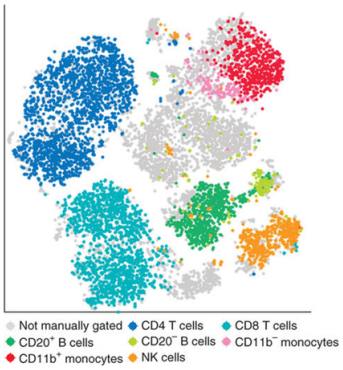


Figure 5.4: Application of viSNE to a healthy human bone marrow sample, stained with 13 markers. The algorithm automatically separates cells into spatially distinct subsets based on the combination of markers that they express. Each point in the viSNE map represents an individual cell and its color represents its immune cell subset. Gray points were not classified. The axes are in arbitrary units.

Figure 5.5: Bone marrow samples from two healthy donors (Marrow 5, 6) and two samples from leukemia patients (ALL A, B) were mapped using viSNE. Samples are color-coded as indicated in key. Samples A and B from tumor patients occupy completely separate regions within the viSNE map, and each forms a distinct population separate from the other ALL sample.

5.2 Consensus Clustering

5.2.1 Introduction

Clustering is in broad use in functional genomics and gene expression (GE) data analysis, due to its use for the discovery of distinct, non-overlapping sub-populations within a larger population. The members of each discovered cluster share some common features and properties that are deemed relevant to the GE domain.

The issues to be addressed when clustering data include: (i) how to determine K, the number of clusters; (ii) how to assign confidence to the selected clusters. The latter issue is particularly important in GE data analysis, where a relatively small number of samples have very high dimensionality, making clustering results especially sensitive to noise and susceptible to over-fitting.

Consensus Clustering (Monti et al., 2003 [15]) is a model-independent resampling based method for clustering validation, determining the number of clusters, and also visualization, which is tailored to the task of GE analysis. The information provided by the analysis is graphically visualized and can be incorporated in decisions about clusters' number and membership. The visualization helps to gain additional insight into the recommendations returned by the algorithm.

5.2.2 Context

We briefly survey some of the previously studied clustering algorithms and their comparison to consensus clustering.

In functional genomics *hierarchical clustering* (HC) has been widely adopted due to its intuitive appeal and visualization properties. By not committing to a specific K (number of clusters), HC provides for a multi resolution view of the data which is extremely useful when exploring new unknown data. However, by not specifying K HC does not provide an "objective" criterion to establish the number of clusters and is prone to the biases and preconceptions of its analyst user. Furthermore, the resulting trees can lock on to accidental features due to the deterministic nature of HC's agglomeration rule and its bottom-up direction.

Other clustering methods such as SOM and K-means clustering both produce well defined clusters and cluster boundaries, unlike HC. They however, lack the visual appeal of HC and also require the number of clusters as input.

5.2.3 Methodology

We assume the data represents a sample of items drawn from distinct sub-populations, and if we were to observe a different sample, drawn from the same sub-populations, the cluster

composition and number induced by the algorithm should be roughly the same. The more the attained clusters are robust to sampling variability, the more confident we can be that these clusters represent real structure. To achieve this, perturbations of the original data are simulated by resampling techniques. A clustering algorithm of choice can then be applied to each of the perturbed data sets, and the agreement (*consensus*) among multiple runs can be used to create a final clustering.

Here is the pseudo-code for the algorithm.

Algorithm: **Consensus Clustering**

- **Input:** a set of items $D = \{e_1, ..., e_N\}$
  a clustering algorithm $Cluster$,
  a resampling scheme $Resample$,
  number of resampling iterations H,
  set of cluster numbers to try $\mathcal{K} = \{K_1, ..., K_{max}\}$
- **for** $K \in \mathcal{K}$ **do**
  1. $\mathbf{M} \leftarrow \emptyset$ {set of connectivity matrices, initially empty}
  2. **for** $h = 1$ **to** H **do**
     (a) $D^{(h)} \leftarrow Resample(D)$ {generate perturbed version of D}
     (b) $M^{(h)} \leftarrow Cluster(D^{(h)}, K)$ {cluster $D^{(h)}$ into K clusters}
     (c) $\mathbf{M} \leftarrow \mathbf{M} \cup M^{(h)}$
  3. $\mathcal{M}^{(K)} \leftarrow$ compute consensus matrix from $\mathbf{M} = \{M^{(1)}, ..., M^{(H)}\}$
- $\hat{K} \leftarrow$ best $K \in \mathcal{K}$ based on consensus distribution of $\mathcal{M}^{(K)}$
- $P \leftarrow$ Partition D into $\hat{K}$ clusters based on $\mathcal{M}^{(K)}$
- **return** P and $\{\mathcal{M}^{(K)} : K \in \mathcal{K}\}$.

5.2.4 Measuring Consensus

Assuming a resampling scheme and a clustering algorithm have been chosen, we devise a method for representing agreement among clustering runs over the perturbed data sets. To this end we define both the *connectivity matrix* and the *consensus matrix*. Let N be the number of items that are clustered.

Definition *Connectivity matrix*
For perturbed data set $D^{(h)}$ we define its corresponding $N \times N$ connectivity matrix as follows,

$$M^{(h)}(i,j) = \begin{cases} 1, & \text{if items i and j belong to the same cluster} \\ 0, & \text{otherwise} \end{cases}$$

Definition *Consensus matrix*
A consensus matrix is an $N \times N$ matrix that stores, for each pair of items, the proportion of clustering runs in which two items are clustered together. Let $I^{(h)}(i,j) = 1$ if i and j were both included in sample h and 0 otherwise.

$$\mathcal{M}(i,j) = \frac{\sum_h M^{(h)}(i,j)}{\sum_h I^{(h)}(i,j)}$$

Note that the consensus matrix is symmetric, for all i, j: $\mathcal{M}(i,j) = \mathcal{M}(j,i)$.
We will refer to $\mathcal{M}(i,j)$ as the *consensus index* of i and j.

As defined, each entry in $\mathcal{M}$ is a real number between 0 and 1, and *perfect consensus* corresponds to a consensus matrix $\mathcal{M}$ where all entries are either 0 or 1. If items in $\mathcal{M}$ are arranged so that items belonging to the same true cluster are adjacent to each other, perfect consensus would translate into a block-diagonal matrix with non-overlapping blocks of 1's along the diagonal, each block corresponding to a different cluster, surrounded by 0's.
Another property of the consensus matrix is that it provides for a similarity measure that can be used in the HC algorithm to yield a tree (dendogram) of item adjacencies. That is $1 - \mathcal{M}$ defines a new *distance matrix* that can be used in place of the usual measures (Euclidean distance, KL divergence, etc).

The consensus matrix can be easily used as a visualization tool to help assess the clusters' composition and number. We associate a color gradient to the 0-1 range of real numbers so that white corresponds to 0 and red to 1. If we assume the matrix is arranged so that items belonging to the same cluster are adjacent to each other (with the same item order used to index both rows and columns of the matrix), then a matrix corresponding to a perfect consensus will be displayed as a color-coded *heat map* with red blocks along the diagonal on a white background. Figure 5.6 shows the heat maps obtained by applying consensus clustering to two simulated data sets. Data set $Uniform1$ is generated by sampling from a uniform distribution over a 600-dimensional hypercube. Data set $Gaussian3$ represents the union of three Gaussian distributions in a 600-dimensional space, where in each distribution a disjoint set of 200 dimensions has higher values. The heat maps represent the consensus matrix computed over 500 iterations. It is evident from the figure that the map corresponding to $Gaussian3$ displays a 3-block structure, while the heat map of $Uniform1$ shows no such structure.

5.2.5 Determining the number of clusters

The properties of the consensus matrix suggest a method for finding the number of clusters that best fits the data. Given that perfect consensus translates into a consensus matrix with

all entries set to either 1 or 0, deviation from this optimal case indicates lower stability of the solution. The idea is to construct a consensus matrix $\mathcal{M}^{(K)}$ for each of a series of cluster numbers ($K = 2, 3, ..., K_{max}$), to compare the resulting consensus matrices and to select the cluster number corresponding to the "cleanest" matrix. We will introduce a measure of consensus based on the matrix $\mathcal{M}$, called *consensus distribution*, based on an assessment of how the entries of $\mathcal{M}$ are distributed within the 0-1 range. The extent to which this distribution is skewed toward 0 and 1 indicates good clustering.

When plotting a histogram of the consensus matrix entries (histogram of the $N(N-1)/2$ entries $\mathcal{M}(i,j)$ for $i < j$), perfect consensus translates to two bins centered at 0 and 1. A histogram corresponding to structureless data would translate in the limit into a single bin centered at some fractional value in $(0, 1)$, since any two items have equal probability of being clustered together.

We return to the two simulated datasets, $Uniform1$ and $Gaussian3$. Figure 5.7 shows the corresponding histograms of consensus indices. The multi-clustered structure of $Gaussian3$ results in the bimodal nature of its histogram around 0 and 1 (pairs of items either always cluster together or never do). The lack of multi-clustered structure in $Uniform1$ results in a largely unimodal histogram (the long tail is due to properties of HC elaborated in [15]).

The following definitions will aid us in the analysis of the clustering visualization. Recall that k denotes the number of clusters.

Definition *Empirical cumulative distribution (CDF)*
For a given histogram of consensus indices we define the corresponding CDF defined over the range $[0, 1]$ as:

$$CDF(c) = \frac{\sum_{i<j} 1\{\mathcal{M}(i,j) \leq c\}}{N(N-1)/2}$$

where $1\{...\}$ denotes the indicator function, $\mathcal{M}(i,j)$ denotes entry (i,j) of the consensus matrix and N is the number of rows (and columns) of $\mathcal{M}$.

Definition *Area under CDF*
The area under the CDF corresponding to $\mathcal{M}^{(K)}$ is computed by the following formula:

$$A(K) = \sum_{i=2}^{m} (x_i - x_{i-1}) CDF(x_i),$$

where the set $\{x_1, ..., x_m\}$ is the sorted set of $\mathcal{M}^{(K)}$ entries with $m = N(N-1)/2$.

Definition *Proportion increase in CDF*
The *proportion increase in the CDF area* as K increase is computed as follows:

$$\Delta(K) = \begin{cases} A(K), & \text{if } K = 2 \\ \frac{A(K+1) - A(K)}{A(K)}, & \text{if } K > 2 \end{cases}$$

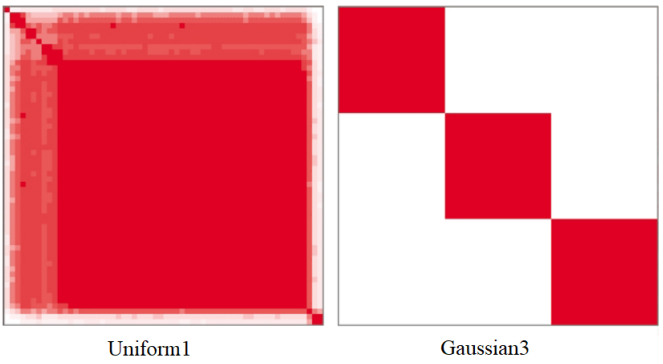


Figure 5.6: Color-coded heat maps corresponding to the consensus matrices for Uniform1 and Gaussian3. Clustering was done using SOM with $K = 3$.

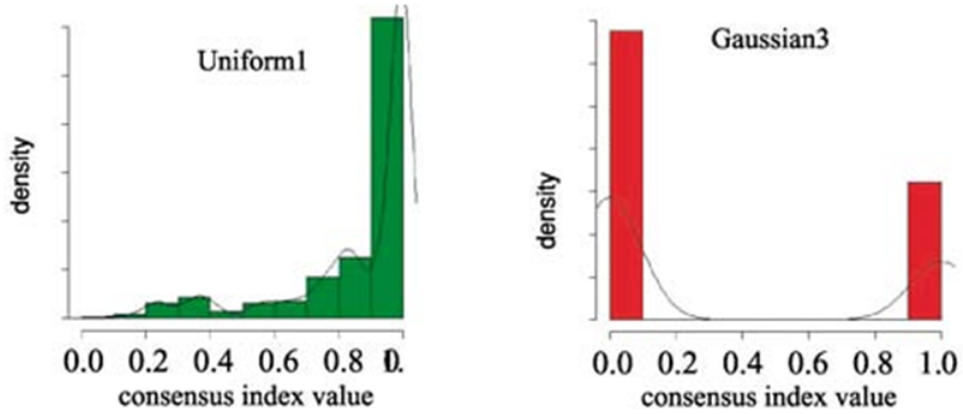


Figure 5.7: Histograms of the entries of consensus matrices $\mathcal{M}(3)$ for Uniform1 and Gaussian3.

The special case of $K = 2$ is due to the fact that $A(1) = 0$ (for $K = 1$ the consensus matrix contains only 1's), thus not allowing for the standard computation of relative increase in area for $K = 2$.

Figures 5.8 and 5.9 (of $Gaussian3$ and $Uniform1$ respectively) present plots of the empirical CDFs corresponding to the entries of $\mathcal{M}^{(K)}$ and of the proportion increase $\Delta(K)$ in CDF as a function of K. For the right plot of $Gaussian3$ the predominance of 0's and 1's affects the shape of the corresponding CDFs, with a step around 0 (the proportion of 0's in $\mathcal{M}^{(K)}$), a flat line across (0,1) and a second step around 1. For $K \leq 3$ the shape of the curve approaches the ideal step function. The CDF for $Uniform1$, on the other hand, displays a different shape, reflecting the lack of stability in cluster membership.
The figures on the right plot the value of $\Delta(K)$ for different numbers of clusters. For $Gaussian3$ the $\Delta(K)$ is significantly larger than 0 only up to $K = 3$. Conversely, for $Uniform1$, as K increases the corresponding $\Delta(K)$ remains large.

The CDFs and the corresponding $\Delta(K)$'s try to quantify the concentration of the consensus distribution along the different values of K. The goal is to choose a K that maximizes this concentration. The selection of the "right" K is often done by inspection of the CDF's shape and progression as K increases. Inspection of the CDF progression helps to select the largest K that induces a large enough increase in the area under the corresponding CDF.

5.2.6 The Adjusted Rand Index

Another measure used by consensus clustering to evaluate the quality of the chosen cluster partition uses the *adjusted Rand index* (Hubert and Arabie, 1985 [16]). The adjusted Rand index (ARI) is a measure of agreement between alternative data partitions that can be used when considering partitions with different numbers of clusters. When ARI equals 1, it corresponds to perfect agreement between the two partitions. It can be shown that ARI has an expected value of 0 for two random partitions.

We start by presenting the *Rand index* [17]. We wish to compare two partitions P and Q of set S where $|S| = N$. The elements of each partition are disjoint subsets of S whose union is equal to S. Rand index measures agreement between P and Q according to how object pairs are classified between the two partitions. We denote,

- A: the number of pairs that are in the same subset both in P and in Q.
- B: the number of pairs that are together in P but not in Q.
- C: the number of pairs that are together in Q but not in P.
- D: the number of pairs that are not together both in P and in Q.

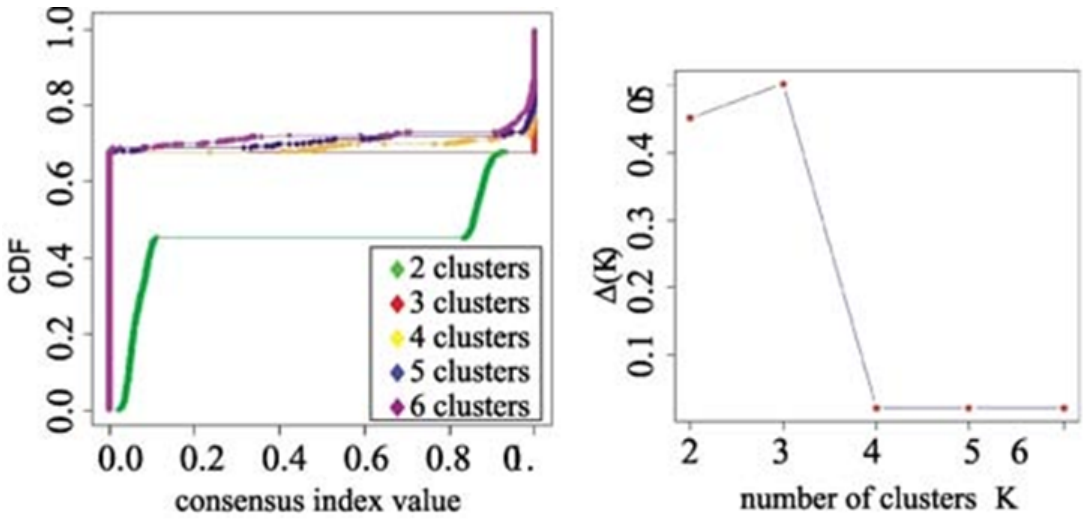


Figure 5.8: Measuring consensus for different K's on *Gaussian*3; Left: the empirical CDFs corresponding to the entries of $\mathcal{M}^{(K)}$ for $K = 2, 3, ..., 6$; Right: proportion increase $\Delta(K)$ in the area under the CDF.

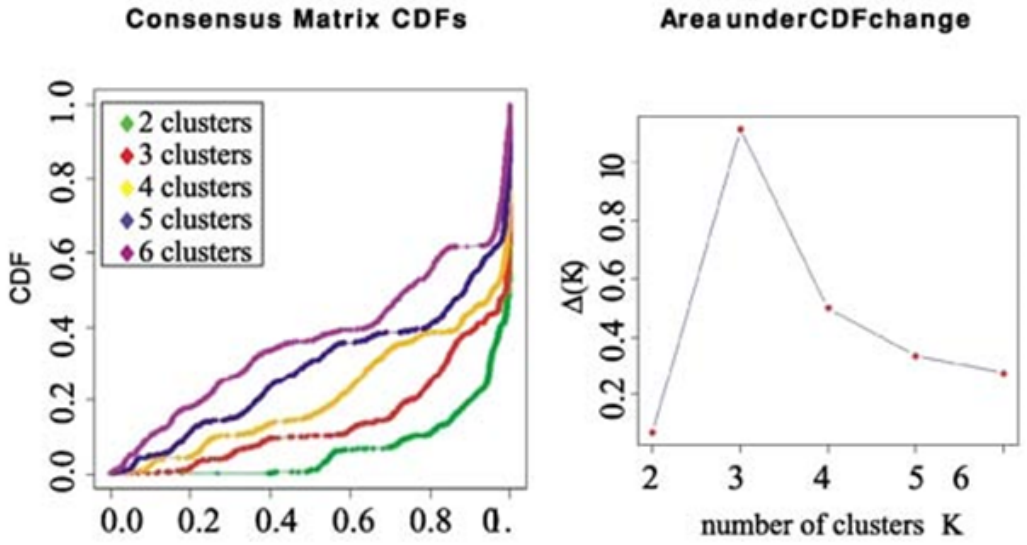


Figure 5.9: Measuring consensus for different K's on $Uniform1$; Left: the empirical CDFs corresponding to the entries of $\mathcal{M}^{(K)}$ for $K = 2, 3, ..., 6$; Right: proportion increase $\Delta(K)$ in the area under the CDF.

The Rand index is defined as,

$$r = \frac{A + D}{A + B + C + D}$$

Hence $0 \leq r \leq 1$, and r reflects the fraction of pairs on which both partitions agree. In order to correct the index for chance behavior, we adjust r as follows,

$$\frac{r - E[r]}{r_{max} - E[r]}$$

Where r_{max} is the maximum index.

We now formally define ARI. Let $\mathbf{P}_a = \{P_{a1}, P_{a2}, ..., P_{aK_a}\}$ and $\mathbf{P}_b = \{P_{b1}, P_{b2}, ..., P_{bK_b}\}$ be two partitions of dataset D, with K_a and K_b not necessarily equal. Let N_{ij} be the number of items of D that are both members of cluster P_{ai} and of cluster P_{bj}. These N_{ij} entries basically define a confusion matrix (with rows indexed by $\mathbf{P}_a$ and columns indexed by $\mathbf{P}_b$) relating the cluster assignments in $\mathbf{P}_a$ with the cluster assignments in $\mathbf{P}_b$. Accordingly, let N_{i*} denote column sums (i.e., the number of items members of cluster P_{ai} irrespective of their membership in $\mathbf{P}_b$), and let N_{*j} denote row sums (i.e., the number of items members of cluster P_{bj} irrespective of their membership in $\mathbf{P}_a$). The assumption of the model is that $\mathbf{P}_a$ and $\mathbf{P}_b$ are picked at random, subject to having the original number of classes and objects in each. $E[r]$ can be computed using this model, and the ARI is computed as follows:

$$ARI = \frac{\sum_{ij} \binom{N_{ij}}{2} - [\sum_i \binom{N_{i*}}{2} \sum_j \binom{N_{*j}}{2}]/\binom{N}{2}}{\frac{1}{2}[\sum_i \binom{N_{i*}}{2} + \sum_i \binom{N_{*j}}{2}] - [\sum_i \binom{N_{i*}}{2} \sum_j \binom{N_{*j}}{2}]/\binom{N}{2}}$$

5.2.7 Results

The datasets used in the experiments are listed in Figure 5.10. The first six datasets represent simulated data, while the last six represent gene-expression microarray data. Figures 5.11 and 5.12 summarize the results of the evaluation on simulated data, while Figures 5.13 and 5.14 summarize the results on real gene expression data.

When applying consensus clustering with HC to a leukemia dataset, if we take into account only the consensus distribution (and the proportion increase in the area under the CDF), the suggested number of clusters is 5. However, if we look at the consensus matrices for K between 5 and 9, shown in Figure 5.15, a 6-cluster structure is clearly visible, with cleaner boundaries than for $K = 5$. This illustrates the advantage of being able to visually inspect the CDF plots and the consensus matrices, allowing us to select the cluster number based on the consensus distribution.

Table 2. Description of the simulated and real dataset used in the experimental evaluation.

Dataset	No. of classes	No. of samples	No. of features
Uniform1	1	60	600
Gaussian1	1	60	600
Gaussian3	3	60	600
Gaussian4	4	400	2
Gaussian5	5	500	2
Simulated6	6	60	600
Leukemia (Golub et al., 1999)	3	38	999
Novartis multi-tissue (Su et al., 2002)	4	103	1000
St. Jude leukemia (Yeoh et al., 2002)	6	248	985
Lung cancer (Bhattacharjee et al., 2001)	4+	197	1000
CNS tumors (Pomeroy et al., 2002)	5	48	1000
Normal tissues (Ramaswamy et al., 2001)	13	99	1277

Figure 5.10: Description of the simulated and real datasets used in the experimental evaluation.

Dataset	K_{true}	CC_{HC}	CC_{SOM}
Uniform1	1	1	1
Gaussian1	1	1	1
Gaussian3	3	3	3
Gaussian4	4	4	4
Gaussian5 ($\lambda = 3$)	5	5	5
Gaussian5 ($\lambda = 2$)	5	4–5	4
Simulated6	6–7	7	6
Simulated4	4	4	4

Figure 5.11: Estimated number of clusters by consensus clustering (CC), in combination with hierarchical clustering (HC) and self-organizing map (SOM). Application to simulated data.

Dataset	NB	HC	CC_{HC}	CC_{SOM}
Uniform1	–	–	–	–
Gaussian1	–	–	–	–
Gaussian3	1.000	1.000	1.000	1.000
Gaussian4	0.896	0.768	0.915	0.908
Gaussian5 ($\lambda = 3$)	0.951	0.932	0.932	0.941
Gaussian5 ($\lambda = 2$)	0.667	0.522	0.589	0.592
Simulated6	0.906	0.986	0.986	0.986
Simulated4	1.000	1.000	1.000	1.000

Figure 5.12: Adjusted Rand index for naive-Bayes (NB), hierarchical clustering (HC), consensus clustering with hierarchical clustering (CC_{HC}), and consensus clustering with SOM (CC_{SOM}).

Dataset	K_{true}	CC_{HC}	CC_{SOM}
Leukemia	3	5	4
Novartis	4	4	4
St. Jude	6	5 (6)	5/7 (6)
Lung cancer	4+	5	5 (7)
CNS tumors	5	5	5/6
Normal tissues	13	7	4/5

Figure 5.13: Estimated number of clusters by consensus clustering (CC), in combination with hierarchical clustering (HC) and self-organizing map (SOM). In parentheses is the estimated number of clusters based on visual inspection of the consensus matrices (when it differs from the one based on the consensus distribution).

Dataset	NB	HC	CC_{HC}	CC_{SOM}
Leukemia	1.00	0.648 (0.46)	0.648 (1.0)	0.721 (0.6)
Novartis-tissue	0.946	0.83	0.921	0.897
St. Jude	0.971	0.949	0.948	0.825
Lung cancer	0.904	0.307 (0.28)	0.310 (0.28)	0.233 (0.22)
CNS tumors	0.632	0.628	0.549	0.429
Normal tissues	0.655	0.457 (0.572)	0.457 (0.572)	0.214 (0.487)

Figure 5.14: Rand index for naive-Bayes (NB), hierarchical clustering (HC), consensus clustering with hierarchical clustering (CC_{HC}), and consensus clustering with SOM (CC_{SOM}). In parentheses is the Rand index corresponding to the partition into K_{true} classes (when it differs from the estimated K).

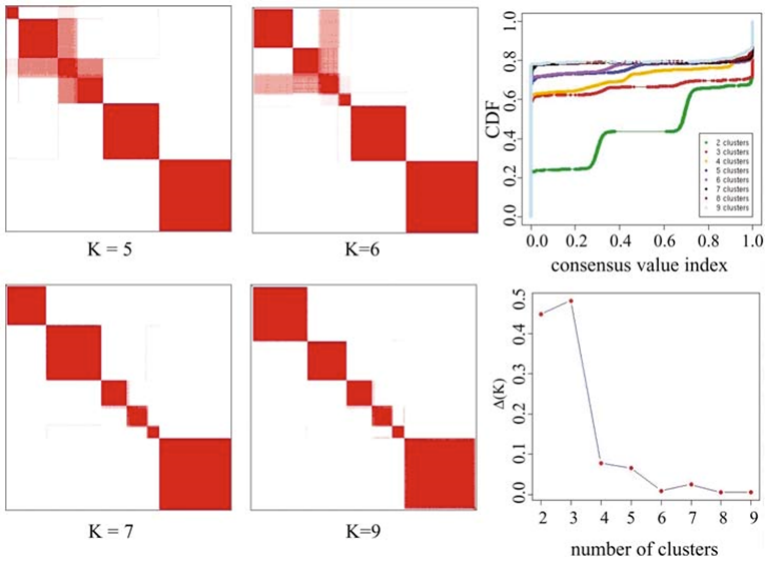


Figure 5.15: Consensus clustering applied to the leukemia data: consensus matrices for $K = 5, 6, 7, 9$.

5.2.8 Conclusion

We have shown a new method to determine K based on *consensus*. In addition we have shown the importance of visualizations in determining the optimal number of clusters. We have also shown resampling (*subsampling*) in consensus clustering proves quite useful in increasing our clustering robustness.

Bibliography

[1] RJ. Cho et al. A genome-wide transcriptional analysis of the mitotic cell cycle. *Mol Cell*, 2:65–73, 1998.

[2] MB. Eisen, PT. Spellman, PO. Brown, and D. Botstein. Cluster analysis and display of genome-wide expression patterns. *Proceedings of the National Academy of Science USA*, 95:14863–14868, 1998.

[3] F. Geraci, M. Leoncini, M. Montangero, M. Pellegrini, and M. E. Renda. Fpf-sb: a scalable algorithm for microarray gene expression data clustering. *Lect. Notes Comput. Sci*, 4561:606?–615, 2007.

[4] F. Geraci, M. Leoncini, M. Montangero, M. Pellegrini, and M. E. Renda. K-boost: A scalable algorithm for high-quality clustering of microarray gene expression data. *Journal of Computational Biology*, 16:859?–873, 2009.

[5] F. Gibbons and F. Roth. Judging the quality of gene expression-based clustering methods using gene annotation. *Genome Res*, 12:1574?–1581, 2000.

[6] T. F. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical Comput. Sci.*, 38:293–306, 1985.

[7] H. Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24:417–441, 1933.

[8] K. Nishimura, K. Abe, S. Ishikawa, S. Tsutsumi, K. Hirota, H. Aburatani, and M. Hirose. Analysis of a complex of statistical variables into principal components. *Genome Informatics*, 14:346?–347, 2003.

[9] K. Pearson. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, 12:559?–572, 1901.

[10] PT. Spellman et al. Comprehensive identification of cell cycle regulated genes of the yeast Saccharomyces cerevisiae by microarray hybridization. *Mol Biol Cell*, 9:3273–3297, 1998.

[11] R. Tibshirani, G. Walther, and D. Botstein et al. Cluster validation by prediction strength. *Journal of Computational and Graphical Statistics*, 14:511?–528, 2005.

[12] L. Maaten, and G. Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

[13] G. Hinton, and S. Roweis. Stochastic neighbor embedding. *Advances in neural information processing systems*, :857–864, 2003.

[14] A. El-ad-David, K. Davis, M. Tadmor, E. Simonds, J. Levine, S. Bendall, D. Shenfeld, S. Krishnaswamy, G. Nolan, and D. Pe'er. viSNE enables visualization of high dimensional single-cell data and reveals phenotypic heterogeneity of leukemia. *Nature biotechnology*, 31:545–552, 2013.

[15] S. Monti, P. Tamayo, J. Mesirov, and G. Todd. Consensus clustering: a resampling-based method for class discovery and visualization of gene expression microarray data. *Machine learning*, 52:91–118, 2003.

[16] L. Hubert, and P. Arabie. Comparing partitions. *Journal of classification*, 2:193–218, 1985.

[17] W. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66:846–850, 1971.