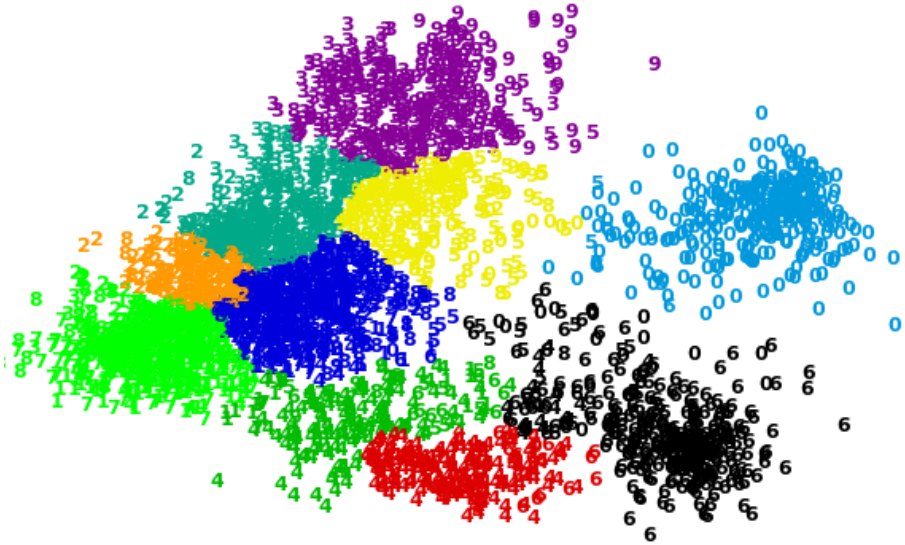
Restricted Boltzmann machines supporting clustering data

André Pacheco
Federal University of Espirito Santo

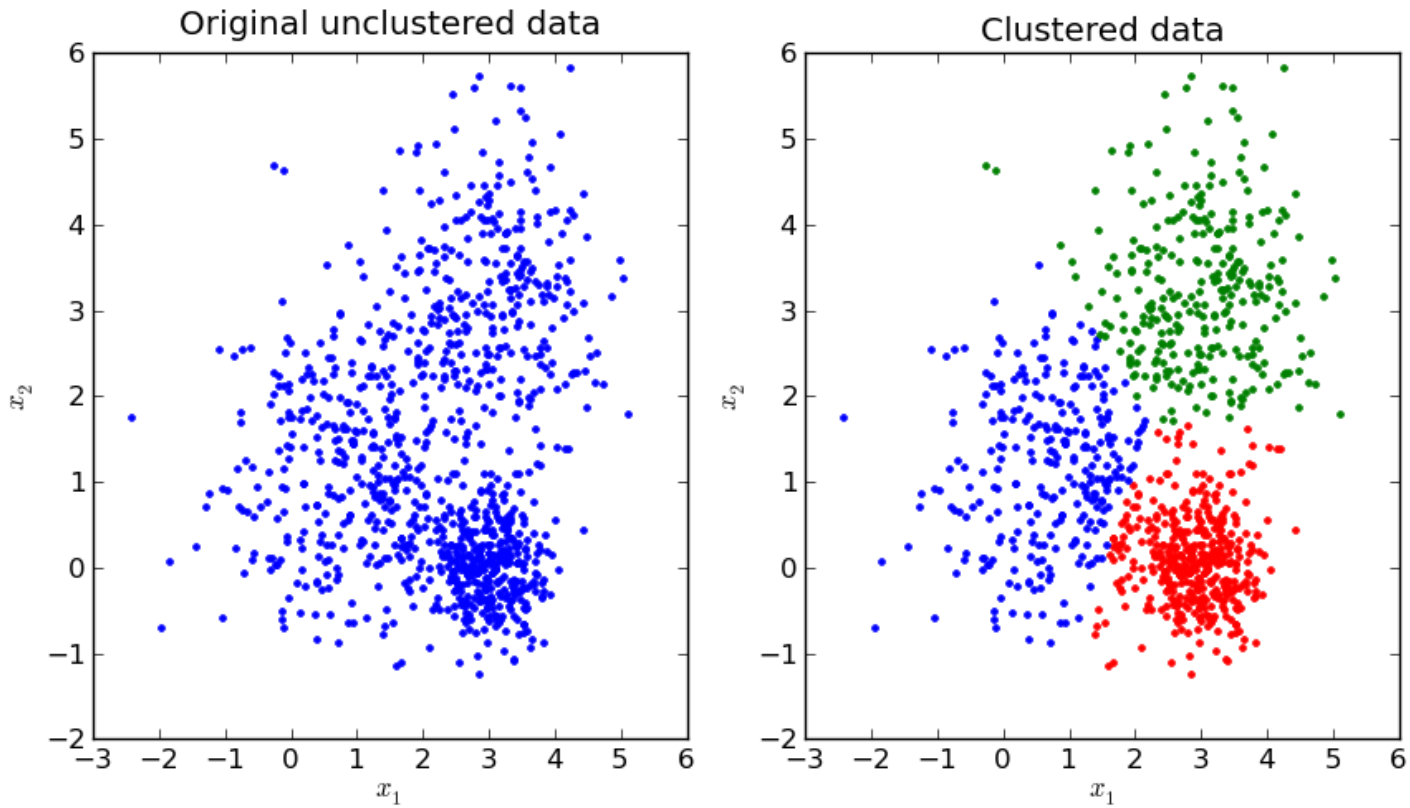


Outubro de 2017

Summary

INTRODUCTION

- Clustering: an **unsupervised** learning where the objects are grouped according to some **similarity** inherent among them



- Aplications: image segmentation, information retrieval, data reduction, time series forecasting etc.

INTRODUCTION

- Different kind of algorithms:
  - Hierarchical
  - Hard and soft clustering
  - Distance based
  - Probabilistic based
- Similarity is a central factor to the clustering process
- Do not confuse clustering and classification
- The main clustering challenges today:
  - The large amount of data (Big data)
  - Online clustering
  - **Data dimensionality**

INTRODUCTION

- The principal component analysis (PCA) is the most popular methodology to tackling the data dimensionality problem
  - Drawback: it is linear!
  - Nonlinear correlations between original variables cannot be preserved
- An alternative is the deep belief network (DBN)
  - **Stack of restricted Boltzmann machines**

K-MEANS

- The K-means aims to partition N observations into K sets $\mathbf{S} = \{S_1, \ldots, S_K\}$ (MacQueen, 1967)
  - Hard cluster
- It is based on minimization of the following objective function:

$$J = \sum_{i=1}^{N} \sum_{j=1}^{K} \left\| \mathbf{x}_i - \mathbf{c}_j \right\| \tag{1}$$

where:

- $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ is the set of samples
- $\mathbf{c}_j$ is the centroid j, i.e., the average of the elements in S_j
- $\|*\|$ is the similarity metric chosen

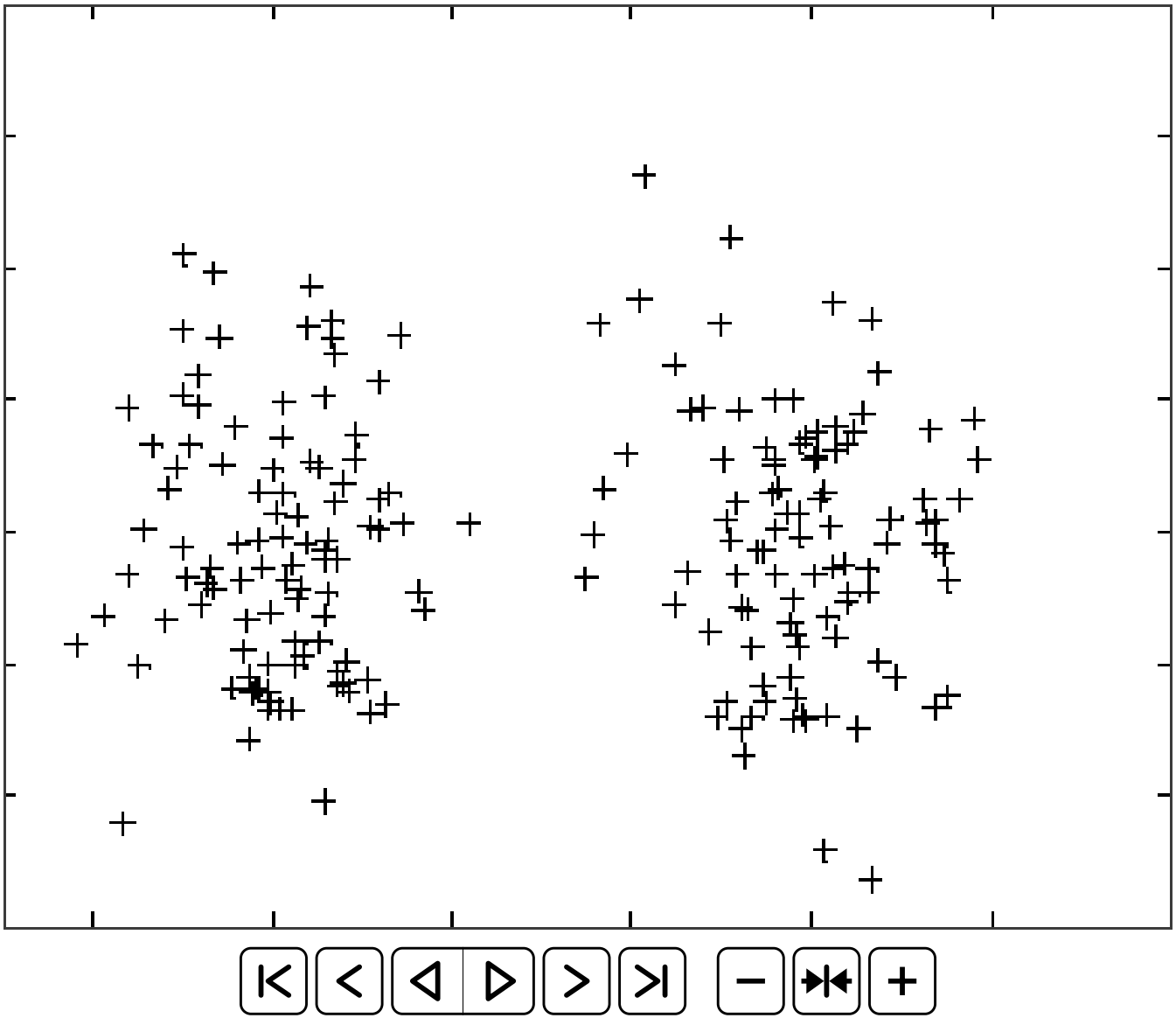
K-MEANS

Algorithm 1: K-means

Input : A set of samples $\mathbf{X}$
The number os clusters K

Set the initial value of each $\mathbf{c} \in \mathbf{S}$ (the centroids)
while $\mathbf{S}$ ***not*** *converged* **do**
  for *each sample* $\mathbf{x} \in \mathbf{X}$ **do**
    Compute $\|\mathbf{x} - \mathbf{S}\|$
    Relate $\mathbf{x}$ to the closest $\mathbf{c} \in \mathbf{S}$
  end
  for *each cluster* $\mathbf{c} \in \mathbf{S}$ **do**
    Update the $\mathbf{c}$ value by computing the mean of the samples relate to it
  end
end
Return: S

K-MEANS OPERATION



FUZZY C-MEANS

- Fuzzy C-means (FCM) is a method of clustering which allows one piece of data to belong to two or more clusters (Dunn, 1973)
  - Soft cluster
- It is based on minimization of the following objective function:

$$J_m = \sum_{i=1}^{N} \sum_{j=1}^{C} \mu_{ij}^m \left\| \mathbf{x}_i - \mathbf{c}_j \right\| \tag{2}$$

where:

- μ_{ij}^m is the degree of membership of $\mathbf{x}_i$ in the cluster $\mathbf{c}_j$ defined as:

$$\mu_{ij}^m = \frac{1}{\sum_{k=1}^{C} \left(\frac{\left\| \mathbf{x}_i - \mathbf{c}_j \right\|}{\left\| \mathbf{x}_i - \mathbf{c}_k \right\|} \right)^{\frac{2}{m-1}}} \quad 0 < \mu_{ij}^m < 1 \tag{3}$$

FUZZY C-MEANS

- m is the level of fuzziness defined in $1 < m < \infty$
  - A large m results in smaller membership values, i.e, fuzzier clusters
  - If $m = 1$ the FCM becomes the K-means
- $\mathbf{x}_i$, $\mathbf{c}_j$ and $\|*\|$ are defined as in K-means
- A cluster $\mathbf{c}_j$ is computed as:

$$\mathbf{c}_j = \frac{\sum_{i=1}^{N} \mu_{ij}^m \times \mathbf{x}_i}{\sum_{i=1}^{N} \mu_{ij}^m} \tag{4}$$

FUZZY C-MEANS

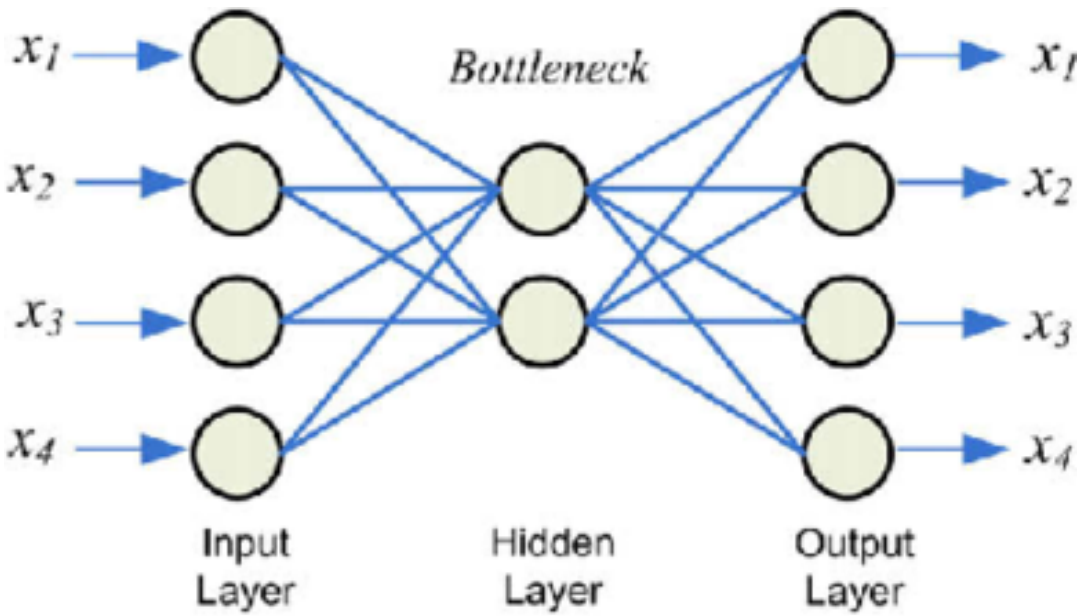
Algorithm 2: Fuzzy C-means

Input : A set of samples **X**
The number os clusters C
The level of fuziness m

Define a matrix $\mathbf{U} = [u_{ij}]$ and initialize it randomly
while U *not* *converged* do
    Compute $\mathbf{c}_j$ according to equation 4
    Update **U** according to equation 3
end
Return: **U** and **S**

AUTOENCODERS

- An unsupervised learning algorithm network that is trained to attempt to copy its input into its output
  - Aplications: feature extraction, dimensionality reduction, denoise etc.

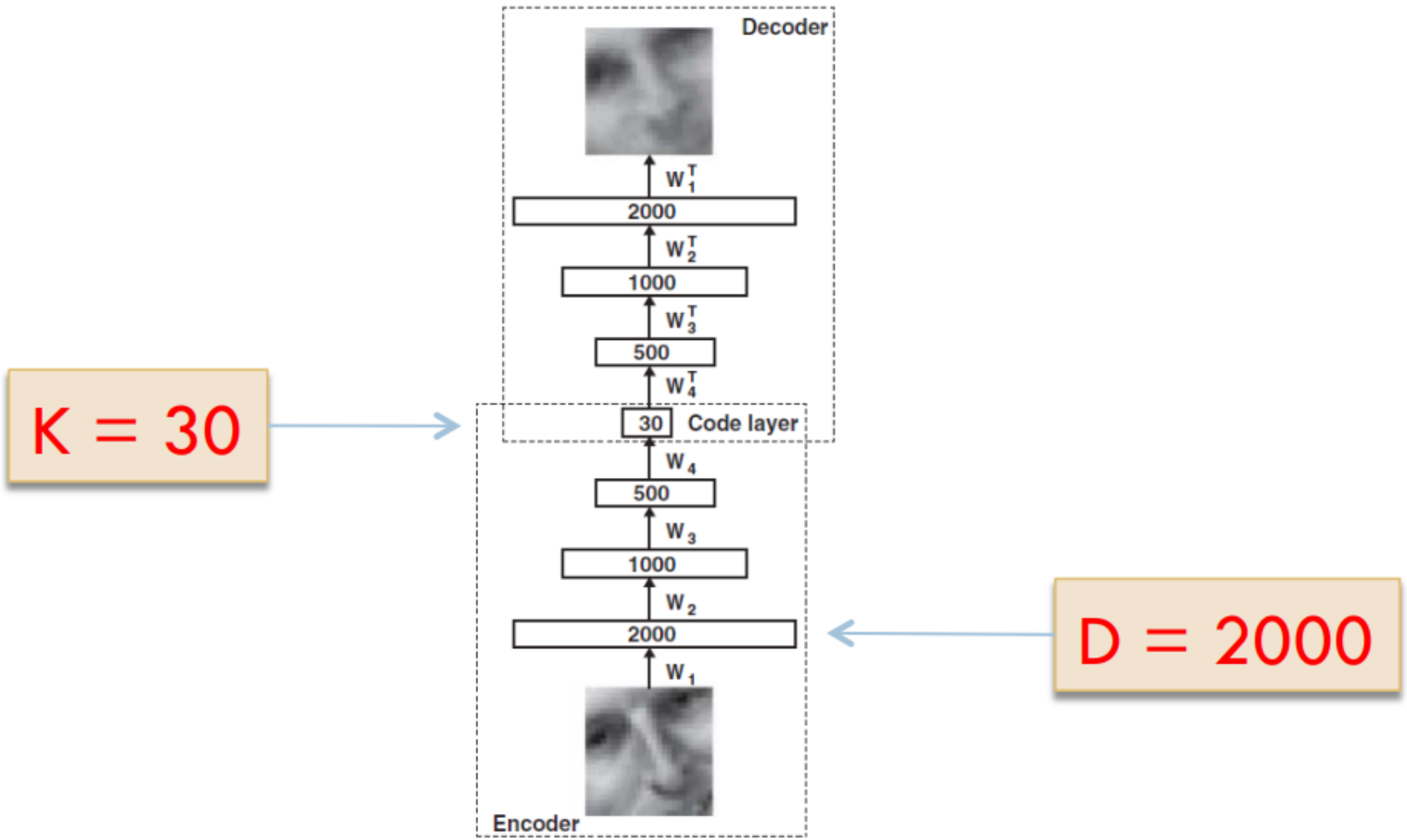


AUTOENCODERS

- The network may be viewed as consisting of two parts:
  - An encoder function $h = f(x)$
  - A decoder that produces a reconstructi on $r = g(h)$
- It is not designed to be unable to learn to copy perfectly
  - The model is forced to prioritize which aspects of the input should be copied
  - It often learns useful properties of the data
- The learning process is described simply as minimizing a loss function $L(x, g(f(x))$
  - The standard approach uses the backpropagation setting the target values to be equal to the inputs, i.e., $y^i = x^i$

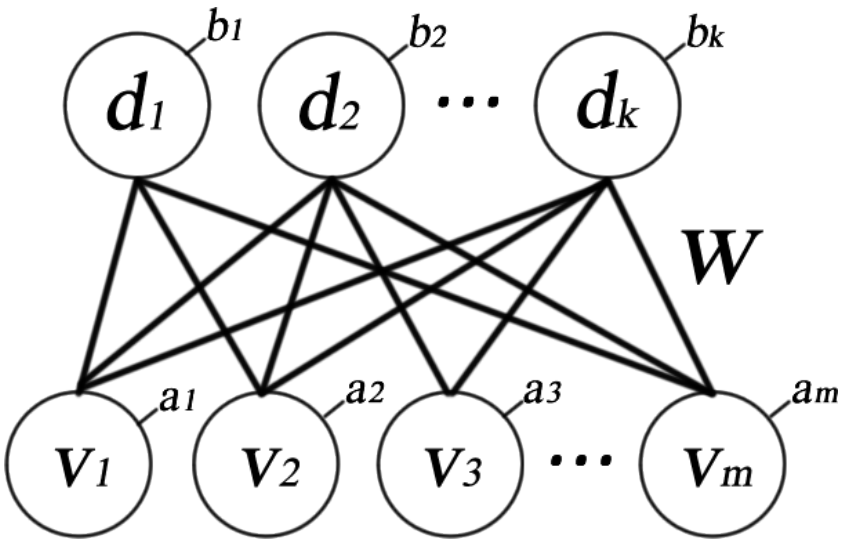
AUTOENCODER

- From (Hinton & Salakhutdinov, 2006):



RESTRICTED BOLTZMANN MACHINE

- The RBM is a stochastic network composed of a visible layer (**v**) and a hidden layer (**h)** (Hinton, 2002)
  - It learns the probability distribution over the input data training
  - Aplications: feature extraction, pattern recognition, dimensionality reduction, denoise etc.



RESTRICTED BOLTZMANN MACHINE

- Each configuration (**v**,**h**) has an associated energy value defined by:

$$E(\mathbf{v}, \mathbf{h}; \boldsymbol{\theta}) = -\sum_{i=1}^{m} \frac{(v_i - a_i)^2}{2\sigma_i^2} - \sum_{j=1}^{k} b_j h_j - \sum_{i,j=1}^{m,k} \frac{v_i}{\sigma^2} h_j w_{ij} \quad (5)$$

- From the network energy, one computes the conditional probabilities of **v** and **h**:

$$p(v_i = v|\mathbf{h}; \boldsymbol{\theta}) = N(v|a_i + \sum_{j=1}^{k} h_j w_{ij}, \sigma^2) \quad (6)$$

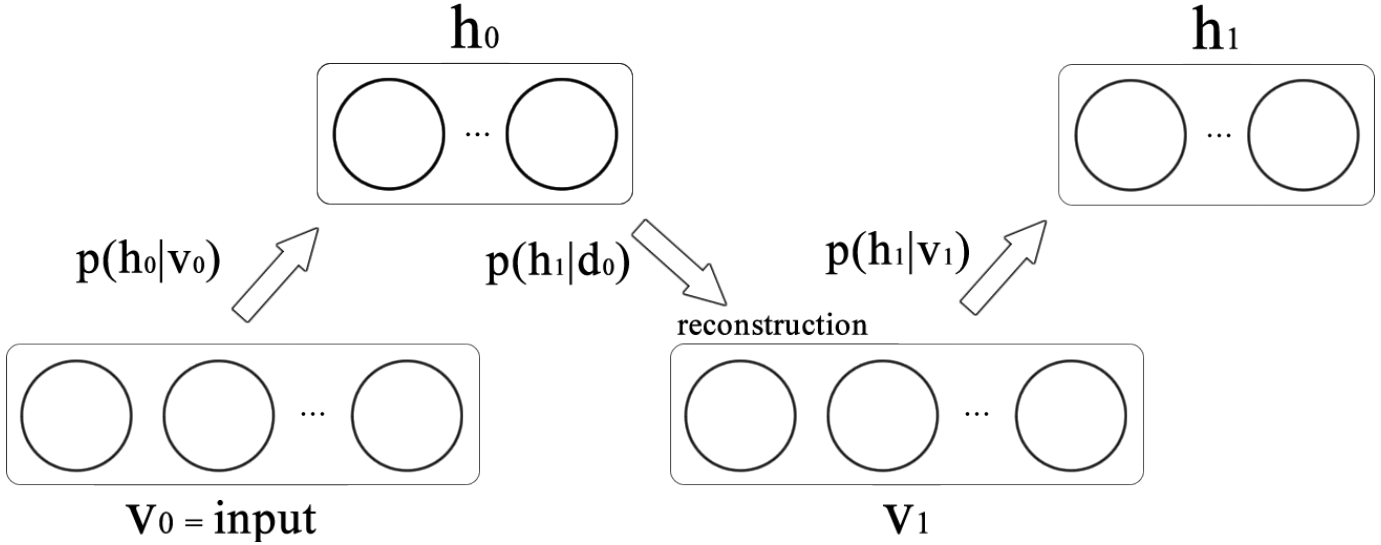
$$p(d_j = 1|\mathbf{v}; \boldsymbol{\theta}) = \phi(b_j + \sum_{i=1}^{m} v_i w_{ij}) \quad (7)$$

where:

- $\boldsymbol{\theta} = (\mathbf{W}, \mathbf{a}, \mathbf{b})$
- $\phi(x) = \frac{1}{1+e^{-x}}$, the logistic function

RESTRICTED BOLTZMANN MACHINE

- The RBM is trained by performing the contrastive divergence algorithm (Hinton, 2002)



RESTRICTED BOLTZMANN MACHINE

- The update rules for $\boldsymbol{\theta}$ are defined as:

$$\mathbf{W^{t+1}} = \mathbf{W^t} + \Delta\mathbf{W^t} \rightarrow \Delta\mathbf{W^t} = \eta(\mathbf{v_0 h_0^T} - \mathbf{v_1 h_1^T}) - \rho\mathbf{W^t} + \alpha\Delta\mathbf{W^{t-1}} \tag{8}$$

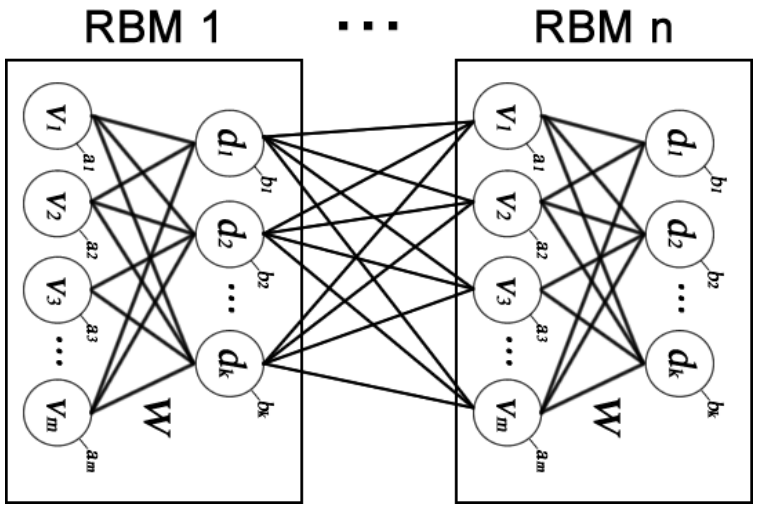
$$\mathbf{a^{t+1}} = \mathbf{a^t} + \Delta\mathbf{a^t} \rightarrow \Delta\mathbf{a^t} = \eta(\mathbf{v_0} - \mathbf{v_1}) + \alpha\Delta\mathbf{a^{t-1}} \tag{9}$$

$$\mathbf{b^{t+1}} = \mathbf{b^t} + \Delta\mathbf{b^t} \rightarrow \Delta\mathbf{b^t} = \eta(\mathbf{h_0} - \mathbf{h_1}) + \alpha\Delta\mathbf{b^{t-1}} \tag{10}$$

- η, ρ and α are known as learning rate, weight decay and momentum, respectively.

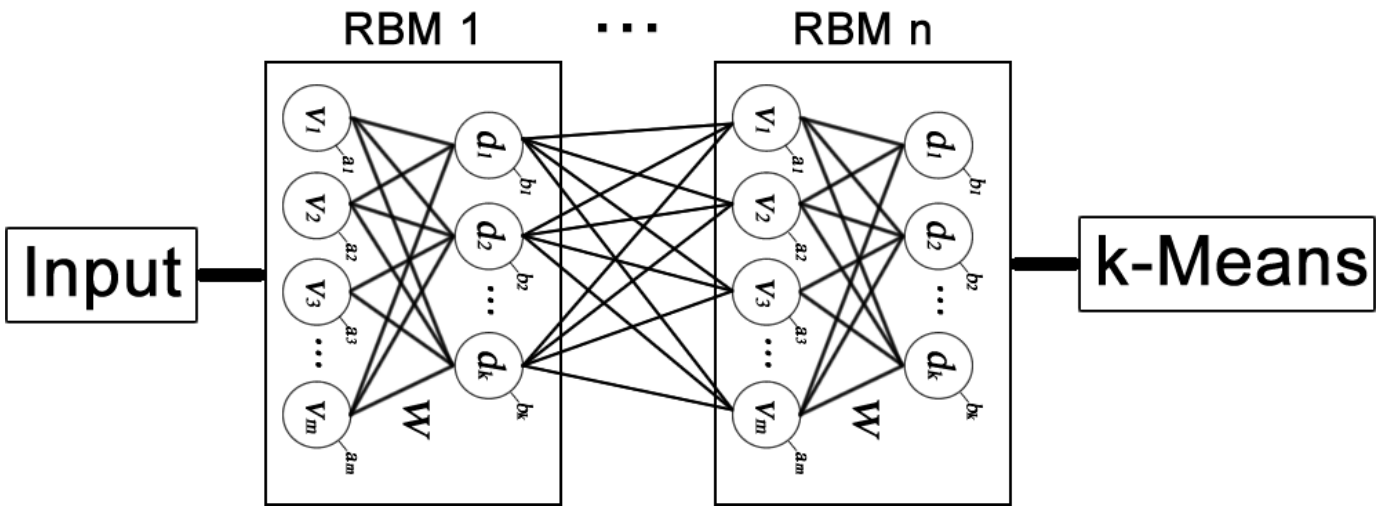
DEEP BELIEF NETWORK

- A DBN is a stack of RBMs or autoencoders or both
- Hinton et al. (2006) proposed a new way to train this kind of network
  - Greedy learning algorithm
  - Hierarchical feature extraction
  - A milestone in the history of deep learning



DBN AND K-MEANS

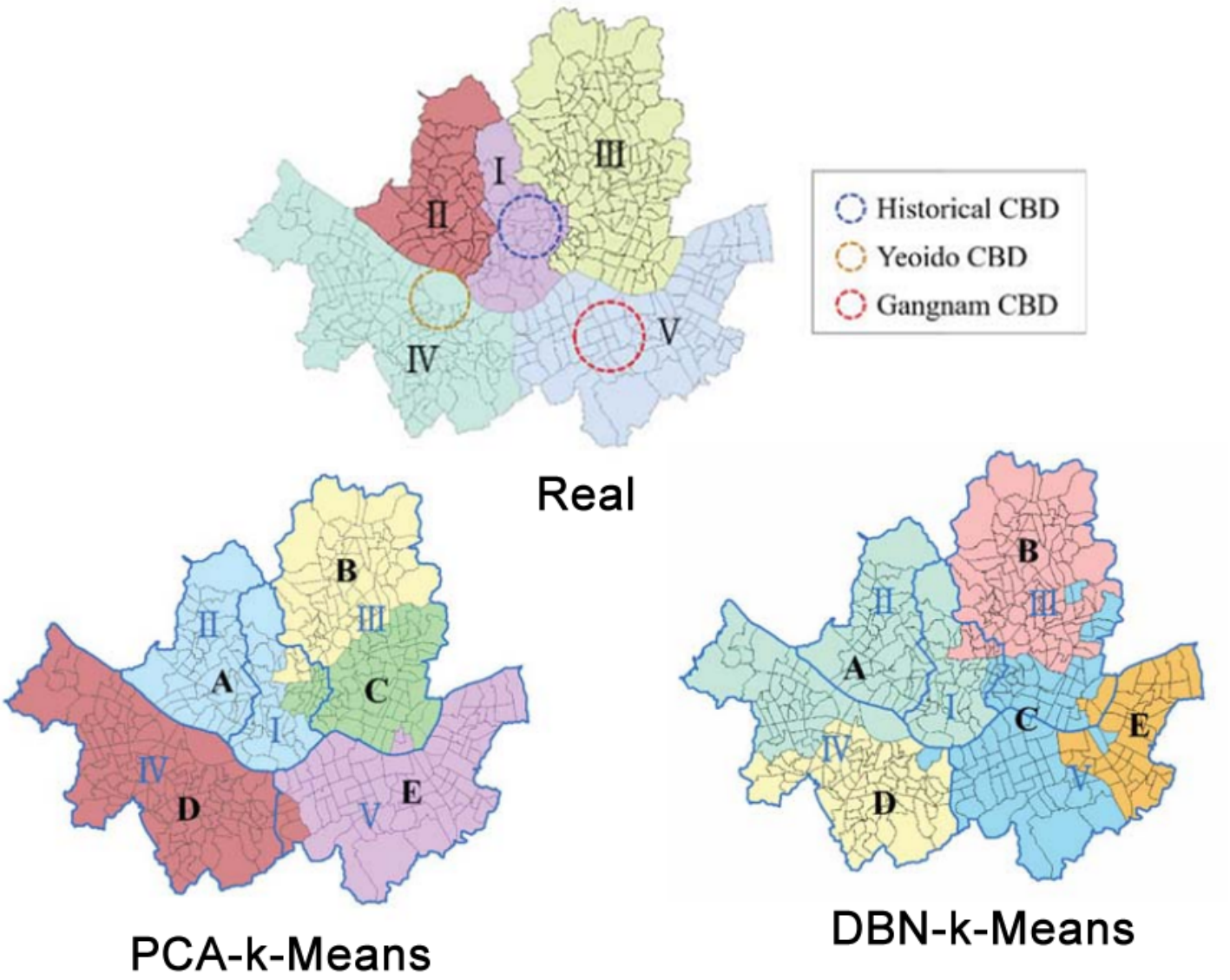
- A stack of RBMs is used to pre-train the data for the K-means algorithm
  - Dimension reduction
  - Feature extraction
  - Nonlinear interactions



DBN AND K-MEANS: EXPERIMENT

- Han & Sohn (2016) used this approach to cluter the seoul metropolitan area acoording to travel patterns
  - They collected data from smart-cards and stored the passengers flow
  - They used a 3-layers DBN with 132-32, 32-32 and 32-32 neuron connections
  - The PCA - K-means was compared with their approach

DBN AND K-MEANS: EXPERIMENT



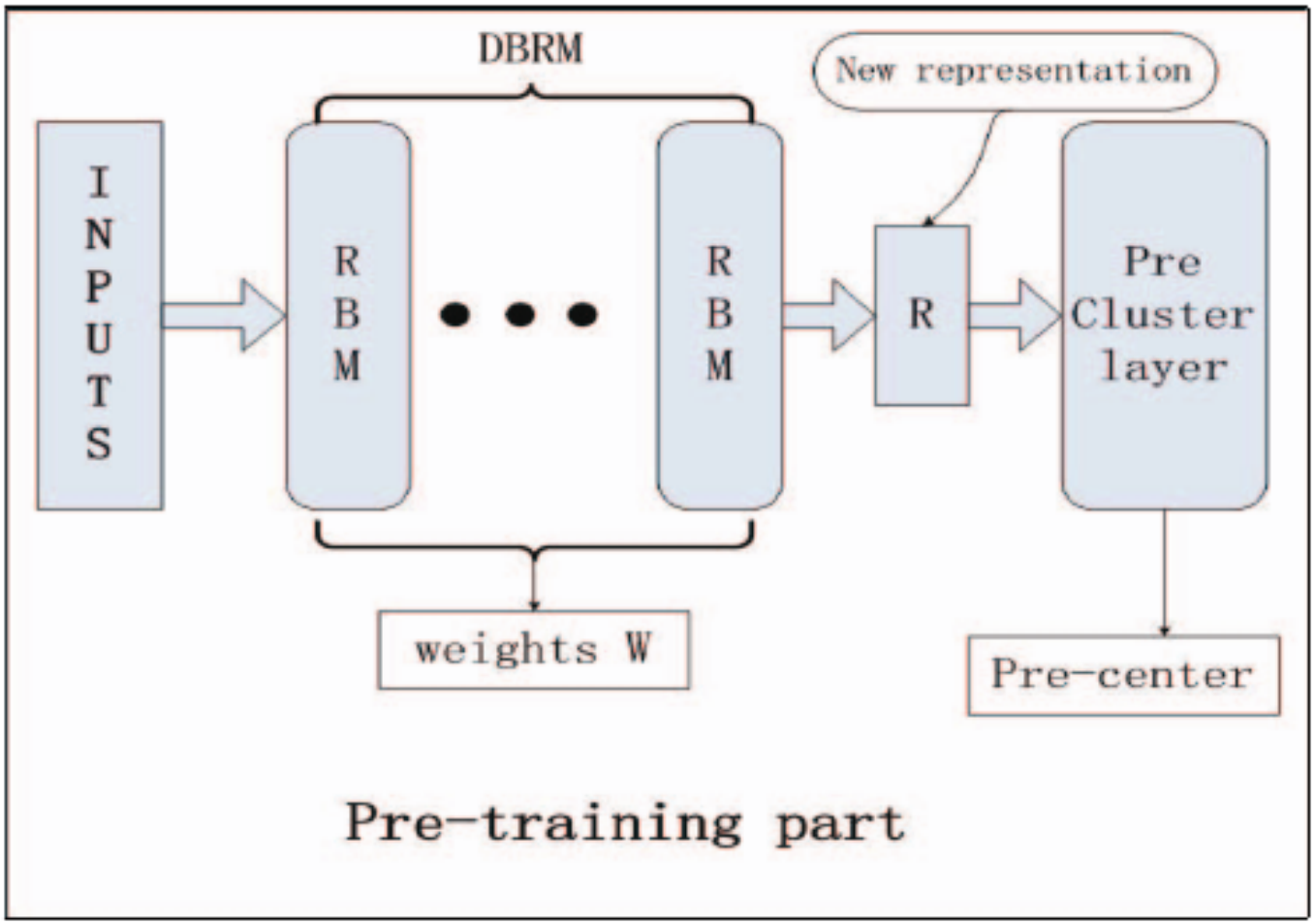
DBN AND K-MEANS: ISSUES

- The approach is straightforward, however, it needs more investigation
  - There is only one study case
- The RBM's setup is not discussed
- Regarding the study case:
  - It is not clear how to handle with the data
  - It is not conclusive

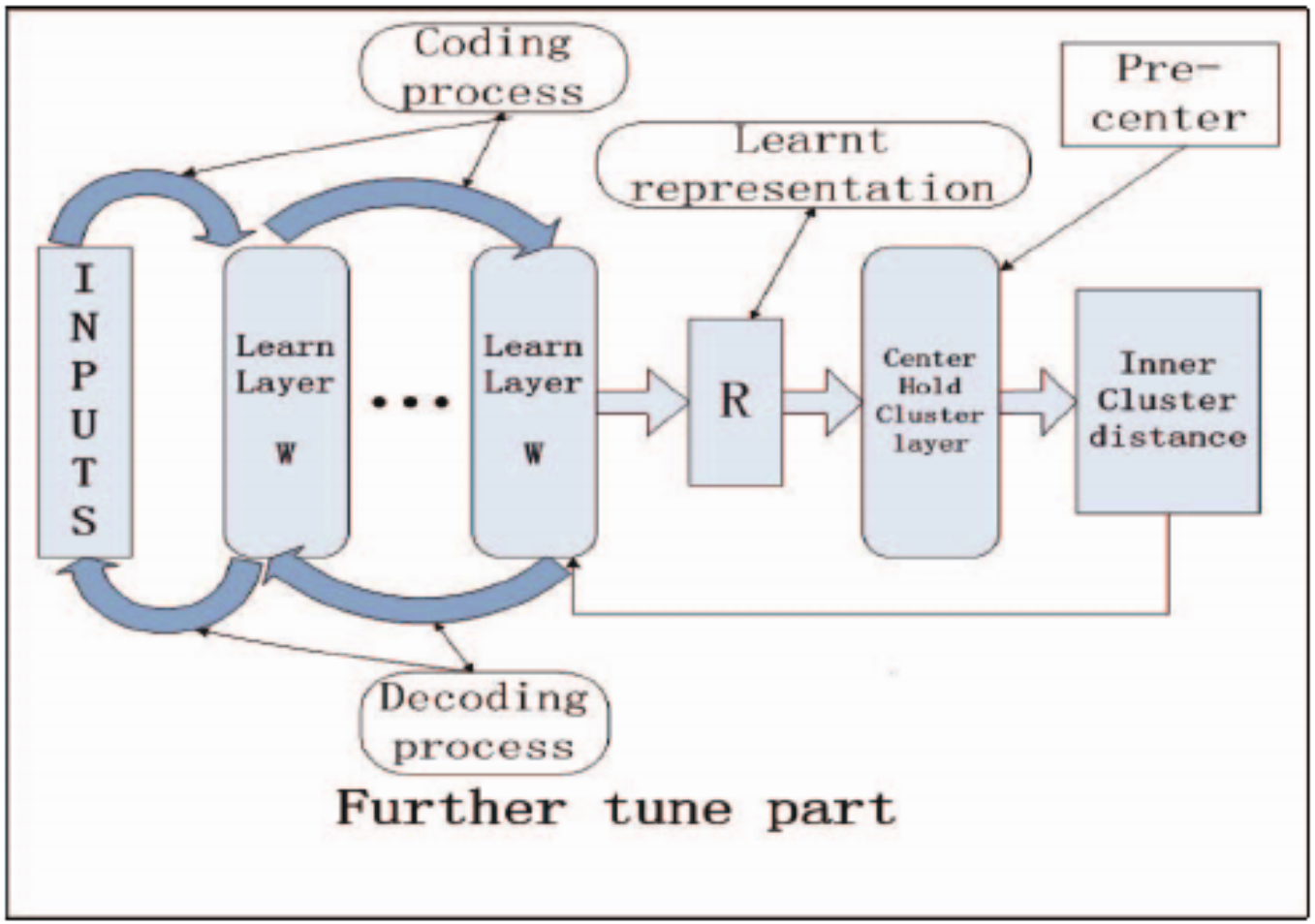
DBN AND FUZZY-C-MEANS

- Yang et al. (2015) proposed a more elaborate algorithm to cluster data by using DBN
- Their algorithm combines DBN and FCM using two steps:
  - First, it uses a stack of RBMs to produce the initial cluster centroid
  - Second, it uses a fine-tuning step by means of deep auto-encoder to optimize the cluster centroid and the membership produced by the FCM
  - They call this algorithm by DBNOC

DBN AND FUZZY-C-MEANS



DBN AND FUZZY-C-MEANS



DBN AND FUZZY-C-MEANS

- In the fining-tune stage, the loss function is achieved by the negative log-likelihood of the reconstruction plus the sumation of inner cluster distance metric:

$$J = e\left(-\sum_i x_i \log \hat{x} - \sum_i (1 - x_i)\log(1 - \hat{x})\right) + \frac{1}{2(1-e)}\left(\sum_{i=1}^{N}\sum_{j=1}^{C}(\hat{x} - c_j)\right) \tag{11}$$

  where $\hat{x}$ is the reconstruction of the inputs x_i
- The backpropagation is used to minimize the loss function
  - The network weights are updated by the rule:

$$W^{k+1} = W^k - \alpha\frac{\partial J}{\partial W} \tag{12}$$

DBN AND FUZZY-C-MEANS: EXPERIMENTS

- Experiments were carried out considering low and high dimensional datasets:

(a) low dimensional datasets

Dataset	Instances	Feature	classes
cred	690	15	2
heart_s	270	13	2
hepa	155	19	2
krvs	3196	36	2
sonar	208	60	2
spec	267	22	2
wine	178	13	3

(b) high dimensional datasets

Dataset	Instances	Feature	classes
isolet	6238	617	26
LSVT	126	256	2
madlon	2000	500	2
musk-clean2	6598	166	2
urbanland_test	507	147	9

DBN AND FUZZY-C-MEANS: EXPERIMENTS

- **First stage:**
  - 3 RBMs stacked with N, 100 and 300 neurons
  - Each RBM iterates for 30 times
- **Second stage:**
  - Iteration number is 50 (high) and 30 (low)
  - $\alpha = 0.1$ (high) and $\alpha = 0.05$ (low)
  - e is described in the next tables

DBN AND FUZZY-C-MEANS: EXPERIMENTS

(a) low dimensional dataset

dataset \ algorithm	k-means		FCM		DBNOC			
					0.2		0.6	
	max	general	max	general	max	general	max	general
cred	0.5623	0.5623	0.5609	0.5609	0.6464	0.5609	0.5609	0.5609
heart_s	0.5926	0.5926	0.5926	0.5926	0.6185	0.6148	0.6444	0.6074
hepa	0.5935	0.5935	0.5935	0.5935	0.5871	0.5613	0.6129	0.6000
krvs	0.5038	0.5038	0.5357	0.5257	0.5514	0.5354	0.5474	0.5407
sonar	0.5385	0.5385	0.5529	0.5529	0.6010	0.5865	0.5817	0.5721
spec	0.4345	0.4345	0.6030	0.6030	0.7266	0.7004	0.7004	0.6629
wine	0.5337	0.5337	0.6854	0.6854	0.6966	0.6910	0.6910	0.6910

DBN AND FUZZY-C-MEANS: EXPERIMENTS

(*b*) high dimensional dataset

dataset \ algorithm	k-means		FCM		DBNOC			
					0.3		0.5	
	max	general	max	general	max	general	max	general
isolet	0.5216	0.5216	0.0774	0.0774	0.0385	0.0385	0.0385	0.0385
LSVT	0.5079	0.5079	0.5635	0.5635	0.6587	0.6587	0.6587	0.6587
madlon	0.5005	0.5005	0.5780	0.5170	0.5205	0.508	0.5355	0.5315
musk-clean2	0.7436	0.7436	0.5288	0.5288	0.8459	0.8459	0.8459	0.8459
urbanland-test	0.2465	0.2465	0.2170	0.2110	0.2604	0.2544	0.2525	0.2406

DBN AND FUZZY-C-MEANS: ISSUES

- The authors did not present the setup of the parameters
- There is no statistical test of the results
  - They did not present the standard deviation of the accuracy
- There is no discussion about the computational time
  - The approach seems to be time-consuming.
- The comparison is not fair enough

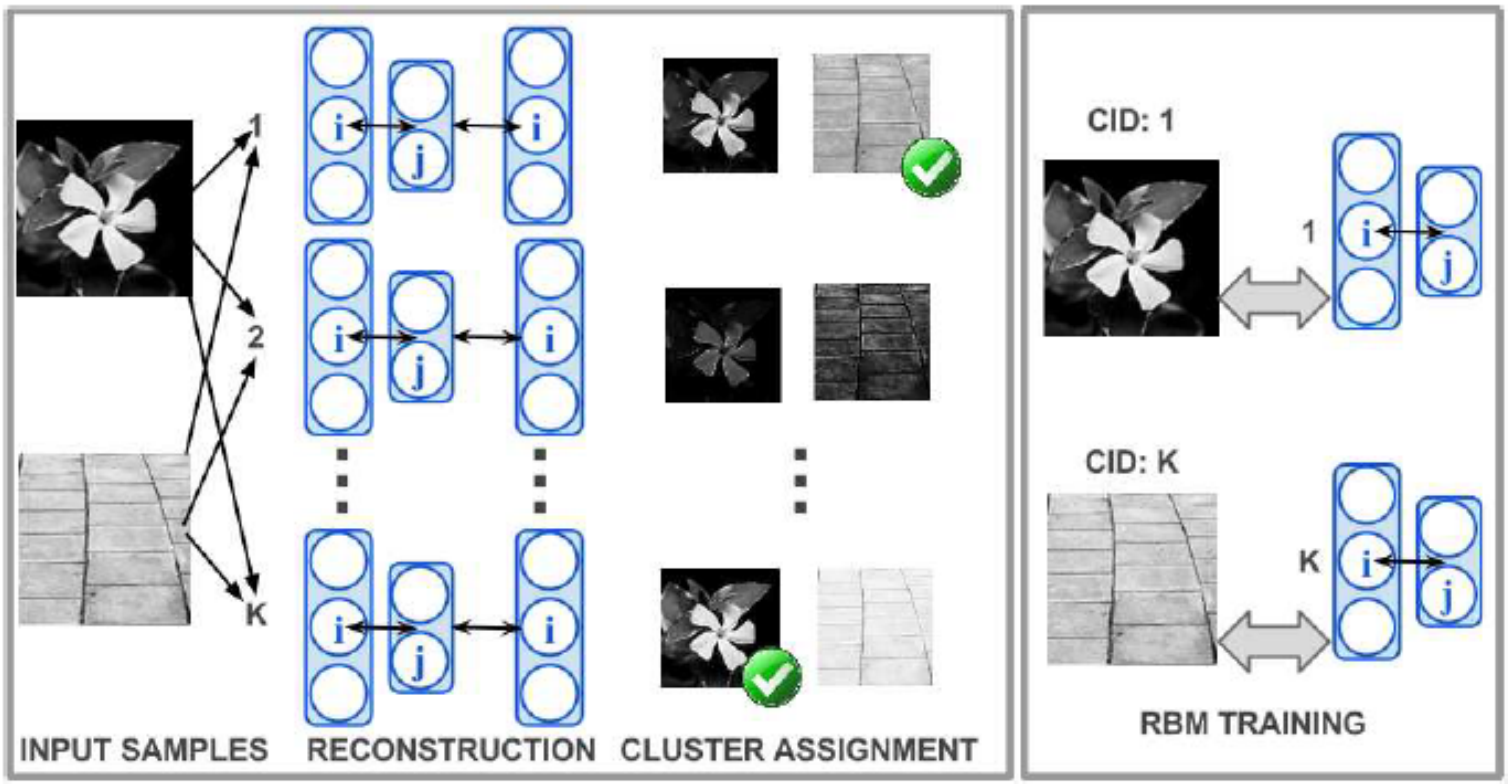
K-RBMs

- Chandra et al. (2013) proposed a framework to cluster data using K-RBMs
- The framework learns K RBMs simultaneously
  - Each RBM learns one non-linear subspace
  - All RBMs have the same architecture
  - The n^{th} data point is associated with the k^{th} RBM (cluster)
    - It is done using the reconstruction error:

$$e = \sum_{i=1}^{m} (v_0 - v_1)^2 \tag{13}$$

  - The RBMs weight's are learnt in a batch update using the associated points

K-RBMs


1
2
K
i
j
INPUT SAMPLES
RECONSTRUCTION
CLUSTER ASSIGNMENT
CID: 1
CID: K
RBM TRAINING

K-RBMS: EXPERIMENTS

- The authors carried out an experiment with 2 synthetic datasets:
  - D_1: 500 samples $\in \mathbb{R}^{144}$, 5 clusters (100×5), Gaussian noise, it has orthogonal basis vectors
  - D_2: 500 samples $\in \mathbb{R}^{144}$, 5 clusters (100×5), it does not has orthogonal basis vectors
- They used 5 RBMs with 144 and 36 visible and hidden layers, respectively [1]
- The K-RBMs were compared with PCA-K-means, RBM-K-means, K-means, Random Sample Consensus (RANSAC) [2], Sparse Subspace Clustering (SSC) [3] and t-SNE [4]

[1] The authors do not present the remaining RBM's parameters
[2] (Fischler & Bolles, 1981)
[3] (Elhamifarm & Vidal, 2009)
[4] (Van der Maaten & Hinton, 2008)

K-RBMs: EXPERIMENTS

- The results in terms of clustering accuracy and computational-time

METHOD	DATASET D1		DATASET D2	
	TIME(S)	ERROR	TIME(S)	ERROR
K-MEANS	0.68	27.4%	2.76	29.6%
PCA	0.37	27.4%	0.42	29.8%
T-SNE	11.68	11.3%	11.93	23.6%
RBM	3.29	26.6%	3.89	28.2%
RANSAC	134.80	66.6%	474.72	69.6%
SSC	365.29	0%	760.48	0%
K-RBM	0.46	0%	3.62	0%

- The authors reported that the clusters converged far before that the RBM training
  - The time-consuming is too low
  - The PCA did not improve the K-means

K-RBMs: ISSUES

- The approach description is not clear
  - The RBMs training phase should have a pseudocode
- The experiments is not enough to draw a conclusion
  - Only two datasets
  - There is no statistical test
  - The time-consuming presented is suspicious
- There is no discussion about the RBMs parameters
  - Neither about the value of K
- The algorithm is strongly sensible to the RBM weights' initialization
- Hinton (2010) stated the reconstruction error is not enough to know if a RBM is learning in its training phase

CONCLUSION

- It was presented three methods to clustering data using RBMs
  - DBN-K-means
  - DBNOC
  - K-RBMs
- The DBN-K-means and DBNOC need to another clustering algorithm to accomplish the task
  - The DBN is used as a fetuare extraction and dimensionality reduction
- The K-RBMs does not need to another clustering algorithm
  - Interesting idea
  - Weak description
- Some ideas can be used in the **Helmholtz machine**

REFERÊNCIAS

1. MacQueen, J. Some methods for classification and analysis of multivariate observations. Proceedings of the fifth Berkeley symposium on mathematical statistics and probability. Vol. 1. No. 14. 1967.
2. Dunn, J. C. A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters, Journal of Cybernetics, v. 3, 32-57, 1973.
3. Hinton, G.E. Training products of experts by minimizing contrastive divergence. Neural Computation 14, 1771–1800, 2002.
4. Hinton G.E., Osindero S., & Teh Y.W. A fast learning algorithm for deep belief nets. Neural computation, v. 18 (7), 1527-54, 2006.
5. Han, G., & Sohn, K. Clustering the seoul metropolitan area by travel patterns based on a deep belief network. In Big Data and Smart City, 3rd MEC International Conference on IEEE, p. 1-6, 2016.
6. Hinton, G.E. and Salakhutdinov, R.R. Reducing the dimensionality of data with neural networks. science, v. 313 (5786), 504-507, 2006.
7. Yang, Q., Wang, H., Li, T. and Yang, Y. Deep Belief Networks Oriented Clustering. In Intelligent Systems and Knowledge Engineering (ISKE), 10th International Conference on IEEE, 58-65, 2015.
8. Chandra, S., Kumar, S., & Jawahar, C. V. Learning multiple non-linear sub-spaces using k-rbms. In the IEEE Conference on Computer Vision and Pattern Recognition, 2778-2785, 2013.
9. Fischler, M. A., & Bolles, R. C. Random sample consensus. ACM, 1981.
10. Elhamifarm E. & Vidal, R. Sparse subspace clustering. In CVPR, 2009.
11. Van der Maaten, L., & Hinton, G. Visualizing Data using t-SNE. In JMLR, 2008.
12. Hinton, G. A practical guide to training restricted Boltzmann machines. Momentum, 9(1), 926., 2010.