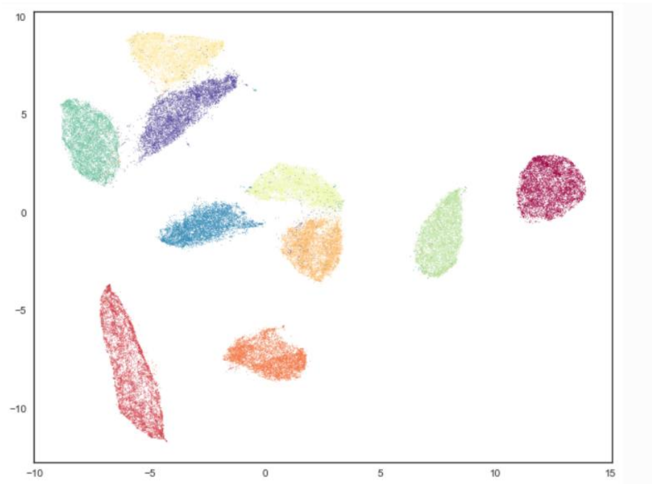
Unsupervised learning (UL)

SUMMARY

1. Unsupervised learning (UL)

- unlike supervised learning models which are *predictive* ones, the **unsupervised learning** models are *descriptive models*
- DM
- determine how the data are organized
- the learner receives only unlabeled examples
    - there is no feedback received about the correct output
- UL - data analysis before applying a supervised learning model
- most important UL models
    - *clustering*
    - *self-organizing maps*
    - dimensionality reduction techniques
        - may be viewed as unsupervised learning
        - PCA
        - *autoencoders* (the encoding part)
        - t-SNE
        - UMAP *Uniform Manifold Approximation and Projection for Dimension Reduction*)
            - constructed from a theoretical framework based on Riemannian geometry and algebraic topology
            - the UMAP algorithm is competitive with t-SNE for visualization quality
            - preserves more of the global structure with superior run time performance
            - **data visualization** (before or after clustering)
                - reduce the data using UMAP
                    - e.g., MNIST data reduced (28x28 pixel grayscale images of handwritten digits – 0..9) to 2 dimensions using UMAP

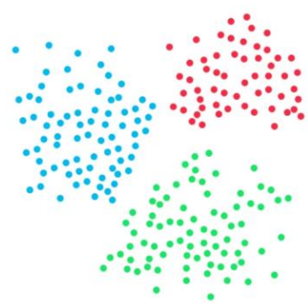


- outlier detection
- text embeddings
- etc

- other UL models
  - UL in RNNs
  - UL in Hidden Markov Models (HMMs)
  - Association Rule (AR) mining
  - Hebbian learning
  - ….

2. Clustering

- is considered the most important *unsupervised learning* problem
- is **unsupervised classification**
- *clustering* is the division of the data in groups of similar objects, with respect to a set of relevant attributes (features) of the analyzed objects



- data modeling puts *clustering* in a historical perspective rooted in mathematics, statistics, and numerical analysis
- from a *machine learning* perspective, *clusters* correspond to **hidden patterns,** the search for clusters is **unsupervised learning**, and the resulting system represents a data concept
- from a practical perspective, clustering plays an important role in data mining applications such as:
  - scientific data exploration
  - information retrieval and text mining
  - spatial databases applications

    - Web analysis
    - CRM
    - marketing
    - medical diagnostics
    - computational biology
    - bioinformatics (gene expression clustering)
    - pattern recognition
    - image processing
    - economic science (market research)
    - WWW
        - document classification
        - cluster weblog data to discover groups of similar access patterns
    - etc
- examples of clustering applications
    - *Marketing*
        - E.g., help marketers discover groups in their customer bases, and then use this knowledge to develop targeted marketing programs.
    - *Land use*
        - E.g., identification of areas of similar land use in an earth observation database.
    - *Insurance*
        - E.g., identifying groups of insurance policy holders with a high average claim cost.
    - *City planning*
        - E.g., identifying groups of houses according to their house type, value, and geographical location.
    - *Earthquake studies*
        - E.g., observed earthquake epicenters should be clustered along continent faults.
- **Formalization**
    - $X=\{x_1, x_2,\ldots, x_n\}$ is a data set consisting of data **instances** (objects)
    - $A=\{a_1, a_2,\ldots, a_m\}$ is a set of relevant attributes (features, characteristics) characterizing the instances from X

      $$\Rightarrow x_i=(x_{i1}, x_{i2},\ldots, x_{im}) \quad i=1,\ldots n$$

        - x_{ij} is the value of attribute a_j for the input instance x_i
        - x_{ij} can be **numerical** or nominal **categorical**
            - *categorical* variables
                - describe categories
                - *ordinal* variables
                    - e.g., (low, medium, high).
                    - ordering between categories
                - *nominal* variables
                    - describe categories that do not have a specific order to them (e.g., *gender*, *ethnicity*)
                - encoding
                    - label encoding
                    - one-hot encoding
                    - count or frequency encoding

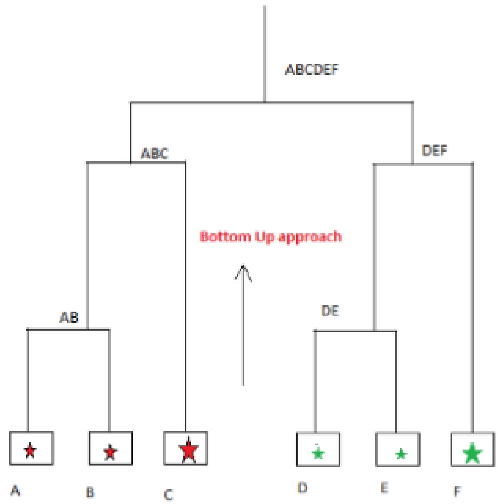
            - replace the categories by the count of the observations that show that category in the dataset
        - hash encoding
            - hashing function
        - binary encoding
            - combination of Hash encoding and one-hot encoding
                - the categorical feature is first converted into numerical using an ordinal encoder
                - then the numbers are transformed into a binary number
        - others
    - a *metric* or *semi-metric* distance function $d: X \times \underline{X} \rightarrow \Re^+$ is defined between two instances from the input space X
        - *d* expresses the dissimilarity between the objects
        - example of distance functions
            (1). ***metrics***: Euclidian, Minkovski, Manhattan, Levestein (for strings), Hamming, Mahalanobis (measures the distance between a multidimensional data point and a distribution)
            (2). ***semi-metrics***: Cosine (used for texts), Pearson/Spearman (correlation)
    - **<u>Goals of clustering</u>**
        - assign instances from X to a finite system of *k* subsets $C_1, C_2, \ldots, C_k$ (clusters) such that $C_1 \cup C_2 \cup \ldots C_k = X$
        - objects within a cluster to have a high similarity with each other and low similarity with instances in other clusters
            - if the subsets $C_1, C_2, \ldots, C_k$ are all disjoint (i.e. an instance belongs to a single cluster) $\Rightarrow$ **<u>hard clustering</u>**
            - otherwise, if an instance may belong to multiple clusters $\Rightarrow$ **soft/<u>fuzzy clustering</u>**
                - an instance belongs to a cluster with a certain membership degree
        - <u>good clustering</u>
            - ***high intra-cluster similarity***
            - ***low inter-cluster similarity***
    - **Online clustering**
    - **<u>Deep clustering</u>**
        - learn clustering representations using DNNs

- **<u>Clustering algorithms</u>** [7]
    - **hierarchical methods**
        - minimize the inter-cluster similarity
            - a **linkage metric** expressing the distance between two clusters is used $D(X, Y)$
                - *single link*

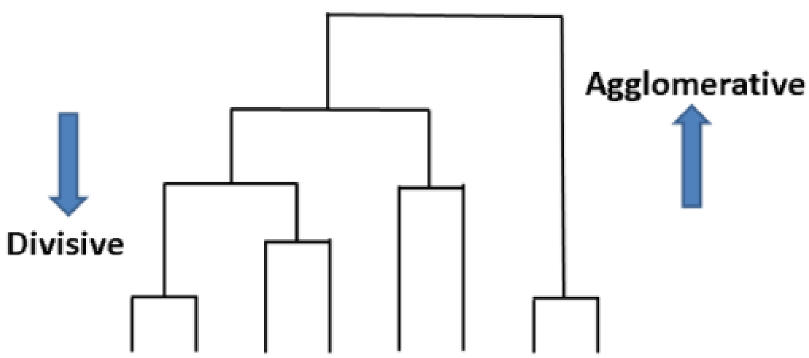
$$D(X,Y) = \min_{x \in X, y \in Y} d(x,y)$$

                    - elongated clusters
                    - sensitive to noise
                - *complete link*

        - *average link*
        - *Ward*
  - are computationally costly
    - $O(n^3)$ time, $O(n^2)$ space
  - agglomerative algorithms
        - CURE (Clustering Using Representatives)



  - divisive algorithms
    - a top-down approach



        - DIANA (DIvisive ANAlysis)
        - useful in linguistics, information retrieval
            - applications: information filtering, search engines
- **partitioning methods**
  - maximize the intra-cluster similarity
  - the number *k* of clusters is given
    - the best *k* may be computed from data
        - grid-search
        - internal evaluation measures
  - *k-means*, *k-medois* (PAM – Partitioning Around Medoids)
  - *k-means*
    - iterative process
        - an instance is assigned to the cluster with the nearest mean (centroid)

        - the objective of *k-means* clustering is to minimize total intra-cluster variance, or, the squared error function
        - the algorithm does not guarantee convergence to the global optimum

number of clusters number of cases case i centroid for cluster j

$$\text{objective function} \leftarrow J = \sum_{j=1}^{k}\sum_{i=1}^{n} \underbrace{\left\| x_i^{(j)} - c_j \right\|^2}_{\text{Distance function}}$$

    - *graph-based methods*
    - *constraint-based clustering*
    - *grid-based methods*
    - *evolutionary clustering*
    - ….
- **Evaluation measures**
    - **external evaluation**
        - when the correct partition (ground truth, gold standard partition) is known
        - *homogeneity*, *compactness*, *v-measure*
    - **internal evaluation** – validity indices
        - Dunn index

$$DI_m = \frac{\min\limits_{1 \leqslant i < j \leqslant m} \delta(C_i, C_j)}{\max\limits_{1 \leqslant k \leqslant m} \Delta_k}.$$

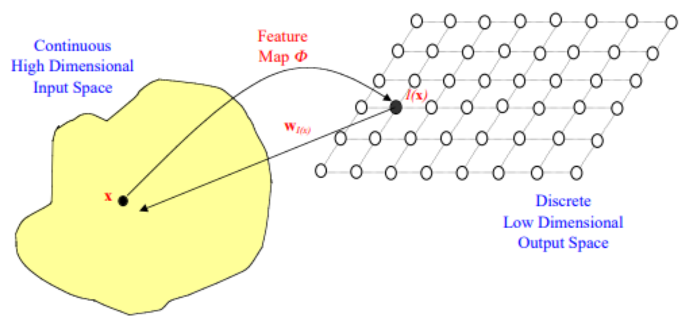
where $\Delta_i = \max\limits_{x,y \in C_i} d(x,y)$

            - **maximized**
        - DB index
        - Silhouette coefficient
        - Calinski-Harabasz index
        - ….

- **Issues** in clustering
    - determine the optimal number of clusters in data?
        - e.g., heuristics, validity indices
    - select relevant features
    - the curse of dimensionality problem
    - outliers, isolated data
    - …

3. Self-organizing maps (SOMs)

- learn to classify data without external supervision
    - training a SOM requires no target vector
- Kohonen network
- is a type of neural network that is trained using UL to produce a low dimensional (typically 2 dimensional) representation of the input space of training samples, called a *map*.
    - **training** builds the *map* using the input patterns
    - after the map was unsupervisedly built, a new input vector can be automatically classified using the **mapping** phase
- **applications**
    - speech recognition
    - robot control
    - web sites classification
    - solving combinatorial optimization problems (e.g. TSP)
- **characteristics** of a SOM
    - is based on **competitive learning**, in which the output neurons compete amongst themselves to be activated
        - only one neuron is activated at one time
        - the activated neuron is called the ***winning neuron***
    - **topological mapping**
        - preserves the neighborhood relations within the input data
            - i.e., if two instances are close to each other in the input space, they will be also close to each other on the *map* (output space)
    - SOMs may be viewed as a non-linear generalization of Principal Component Analysis (PCA)

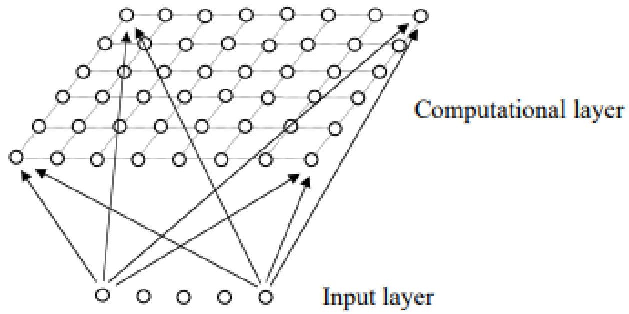


Each point I in the output space will map to a corresponding point $\mathbf{w}(I)$ in the input space.

[9]

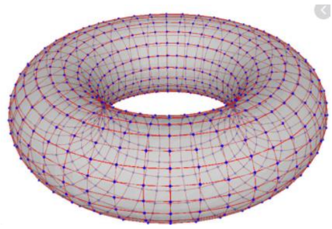
    - SOMs are useful tools for visualizing high-dimensional data
- **Kohonen network**
    - a type of SOM having a feed-forward structure
        - a single computational layer arranged in rows and columns
        - each neuron is fully connected to all the nodes from the input layer
            - if the input space is D dimensional, there are D input units
        - the connections are weighted

- if j is a neuron in the computational layer, the weight vector **is** $w_j = (w_{j1}, w_{j2}, \dots, w_{jD})$



[9]

- topologies:
  - lattice
  - **torus**



- 3D shape generated by revolving a circle along a line that is coplanar with the circle
- provides better neighborhood

- **Self-organization process** [9]

(1). *Initialization*

- the connection weights are initialized with small random values

(2). *Competition*

- for each input pattern, the neurons compute the values of a **discriminant function** which provides the basis for competition
- e.g., the squared Euclidian distance between the input vector $x = (x_1, x_2, \dots, x_D)$

  and the weight vector $w_j = (w_{j1}, w_{j2}, \dots, w_{jD})$ for neuron j

$$d_j(\mathbf{x}) = \sum_{i=1}^{D} (x_i - w_{ji})^2$$

  - other distances (e.g. cosine)
- the neuron with the smallest value of the discriminant function is declared the **winner** – the best matching unit (BMU)
  - the neuron whose weight vector comes closest to the input vector (i.e. is most similar to it) is declared the winner

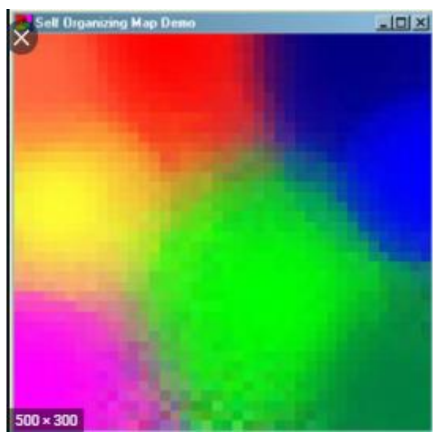
(3). *Cooperation*

- the winning neuron determines the spatial location of a topological neighbourhood of excited neurons, thereby providing the basis for cooperation among neighbouring neurons

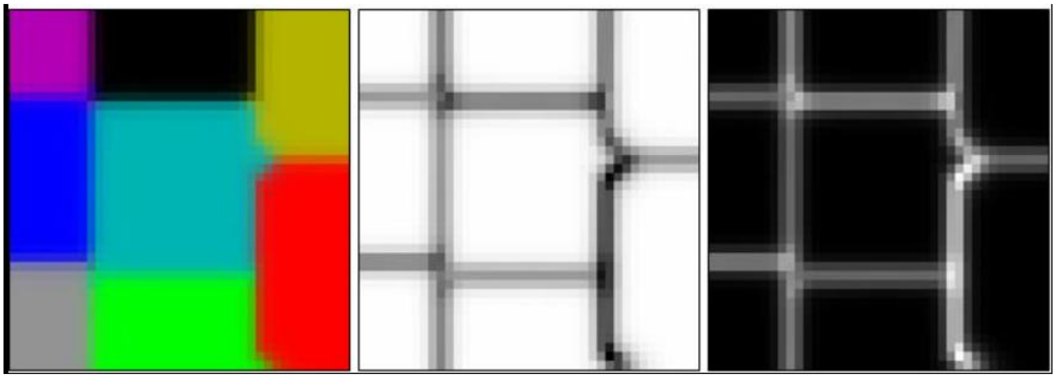
(4). *Adaptation*

- the weights of the winning neuron and of its neighbors will be updated
- the excited neurons decrease their individual values of the discriminant function in relation to the input pattern by adjusting the associated connection weights
- after many presentations, neighboring neurons have learned vectors similar to each other

- **SOM visualization**: **U-Matrix (Unified Distance Matrix)** [10]
  - visualizing the cluster structure of the SOM
  - a matrix of distances between the weight vectors of adjacent neurons on the map
    - the matrix is then represented using a color scale
    - e.g. the lighter/darker the color between two map units, the smaller is the relative distance between their weight vector
      - in this case, dark/white areas on the map identify boundaries between the clusters in the underlying data
  - e.g. Color SOM
      - input vectors in $\Re^3$ RGB color codes



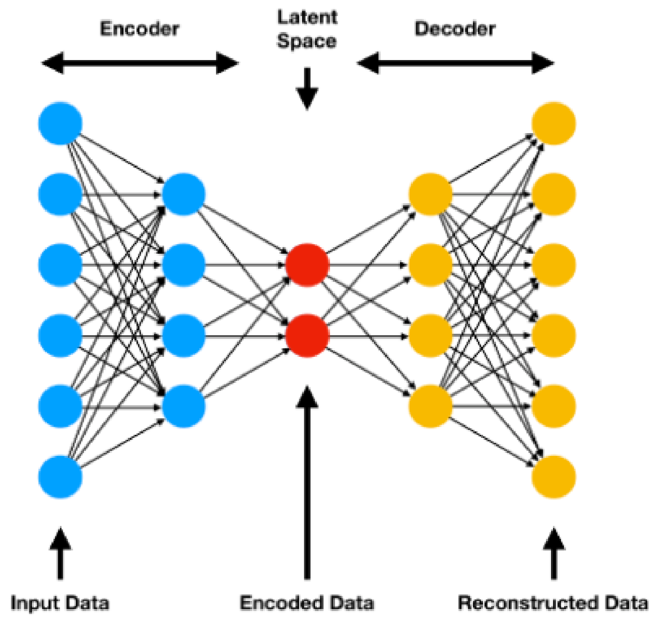
U-matrix visualization



- **Evaluation measure**
  - as an **internal evaluation** measure, for measuring the quality of the SOM, the **Average Quantization Error (AQE)** is used
    - AQE - the average distance between each input vector x and its BMU
- **Deep SOMs**
  - Self-organizing convolutional map (SOCOM)
  - SOMs + CNN

4. Autoencoders (AEs)

- feed-forward neural networks
- input instances are points in the high dimensional space, i.e. $X=\Re^n$
- are **self-supervised learning** techniques
- an AE is composed of two components which are stacked together
  - an *encoder*
  - a *decoder*

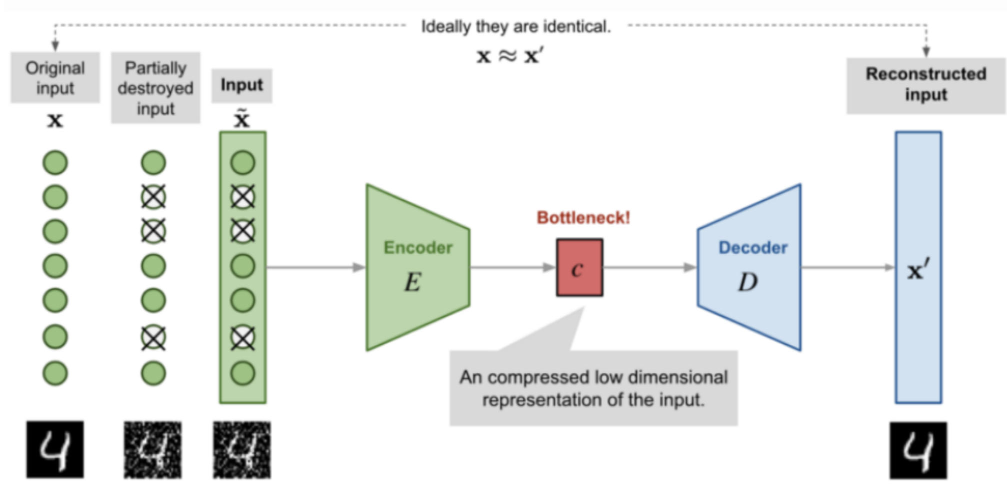


- the goal of an AE is to model two functions f and g, such that $f(g(x)) \approx x$
  - *encoder* $g: \Re^n \rightarrow \Re^m$, $g(x)=h$

        - h – hidden representation of x
    - *decoder* f: $\Re^m \rightarrow \Re^n$, $f(h) = \hat{x}$
    - **minimizing** a loss function
        - $$L(\widehat{x}, x) = \frac{1}{n}\sum_{i=1}^{n}(\widehat{x}_i - x_i)^2$$
        - …..
    - **optimization algorithm**: backpropagation
    - goal
        - useful representation of data in the hidden state
        - tasks: information retrieval, data representation, etc
    - risk of *overfitting*
- **characteristics**
    - if $m<n$, the AE is called *undercomplete*
    - AEs are better than PCA
        - AEs are not restricted to perform linear mappings.
    - An AE with a single layer and linear activation function can be viewed as equivalent to a PCA mapping
    - types of AEs
        - **sparse**
            - help the model avoid the simple copying of the input to the output by introducing a sparsing penalty
                - usually the sparsing penalty is the L1 regularization on the encoded state
                - the penalty term is scaled using a small real number λ.

$$L(\widehat{x}, x) = \frac{1}{n}\sum_{i=1}^{n}(\widehat{x}_i - x_i)^2 + \lambda \sum_{i=1}^{m}|h_i|$$

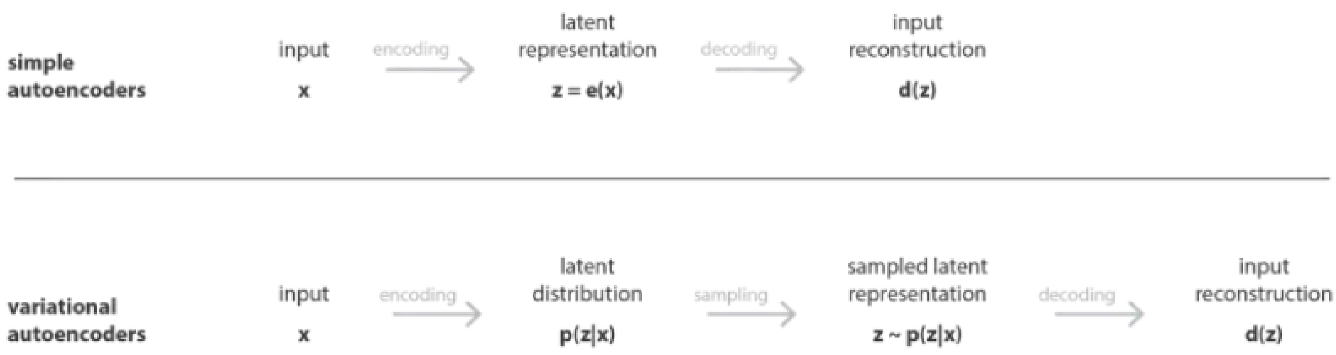
        - **denoising** (DAEs)
            - another technique to avoid the copying of the input data to the output layer, forcing the hidden layers to learn the most robust features of the input.
            - decrease the risk of overfitting
            - a denoising AE is fed stochastically corrupted input data and tries to reconstruct the original (uncorrupted) input data.
            - training example ($\widetilde{x}$, x)



Denoising AE architecture by Lilian Weng

- **convolutional** (CAEs)
  - CNN architectures
  - consider convolutional layers for an image autoencoder
  - applications: image compression and denoising
- **variational**
- other: **contractive** (regularizing AEs), etc…

- **variational** (VAEs)
  - deep generative models (like GANs)
  - probabilistic
  - VAEs view the hidden representation as a latent variable with its own prior distribution
    - Bayesian approach
    - VAEs are generative models with properly defined prior and posterior data distributions.
    - instead of encoding an input as a single point, it is encoded as a distribution over the latent space.
  - AE whose training is regularised to avoid overfitting and to ensure that the latent space has good properties that enable generative process.



Difference between autoencoder (deterministic) and variational autoencoder (probabilistic).

- **convolutional** VAEs (CVAEs)
- **Wasserstein** AEs
- **applications**
  - data augmentation
  - *bioinformatics*
    - molecular design
    - protein/DNA/gene expression analysis
    - protein design
  - NLP
    - text generation
  - image generation
  - etc

- **applications**
  - feature extraction – using only the encoding part
  - image analysis, speech processing,…
  - anomaly detection, one class classification (OCC)

5. Other UL related research topics

- **clustering**
  - adaptive clustering
    - clustering + Reinforcement Learning (RL)
    - adapt the distance function
  - **Bayesian** clustering
  - **Biclustering**
    - simultaneously cluster rows and columns of a matrix
    - used for biological data
  - use of SVM for clustering
  - clustering for selecting prototypes in RBFNs

  - online clustering
- **SOMs**
  - Fuzzy SOMs
  - Bayesian SOM
  - SOM for Reinforcement Learning (RL)
  - Deep SOMs
- **AEs**
  - one class classification and anomaly detection
  - Temporal difference VAE (TD-VAE)
    - TD approach from RL
  - Dynamical VAEs
    - temporal model

[SLIDES]

– Unsupervised learning [6]
– Clustering [8]
– Self-organizing maps (J. Bullinaria) [9], SOMs (B. Silva) [10]

[READING]

– Unsupervised learning (N. Nilsson) [5]
– Clustering [7]
– Deep Autoencoders (Deng and Yu) [3]
– Autoencoders (Goodfellow et al.) [2]

Bibliography

[1] Mitchell, T., *Machine Learning*, McGraw Hill, 1997 (available at www.cs.ubbcluj.ro/~gabis/ml/ml-books)

[2] Ian Goodfellow, Yoshua Bengio, Aaron Courville, *Deep Learning*, MIT Press, 2016 (online edition at http://www.deeplearningbook.org/)

[3] Li Deng and Dong Yu, *Deep Learning. Methods and Applications*, Foundations and Trends® in Signal Processing, Volume 7 Issues 3-4, 2014 (https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/DeepLearning-NowPublishing-Vol7-SIG-039.pdf)

[5] Nillson, N., *Introduction to Machine Learning*, Stanford University, 1996 (available at www.cs.ubbcluj.ro/~gabis/ml/ml-books)