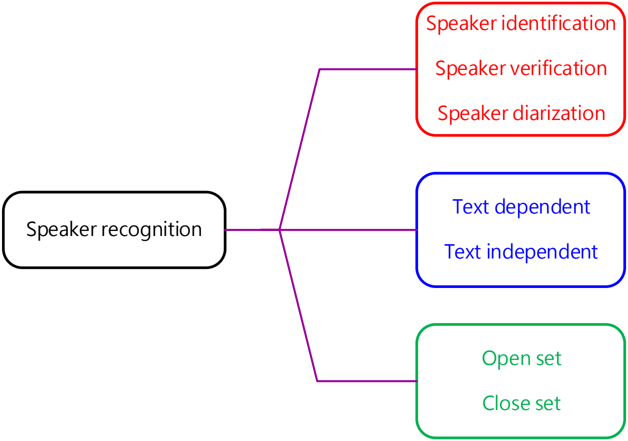
Outline

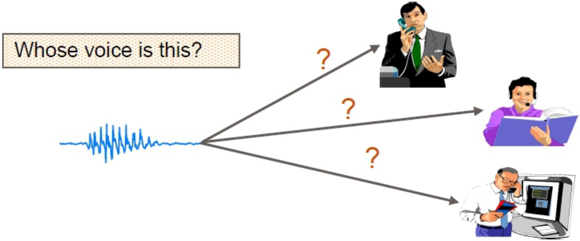
Fundamentals of speaker recognition

- Speaker recognition is a technique to recognize the identity of a speaker from a speech utterance.



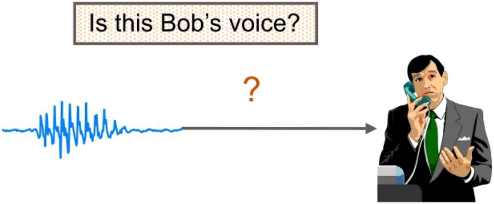
Speaker identification

- Determine whether unknown speaker matches one of a set known speakers
- One-to-many mapping
- Often assumed that unknown voice must come from a set of known speakers – referred to as close-set identification
- Adding "none of the above" option to closed-set identification gives open-set identification



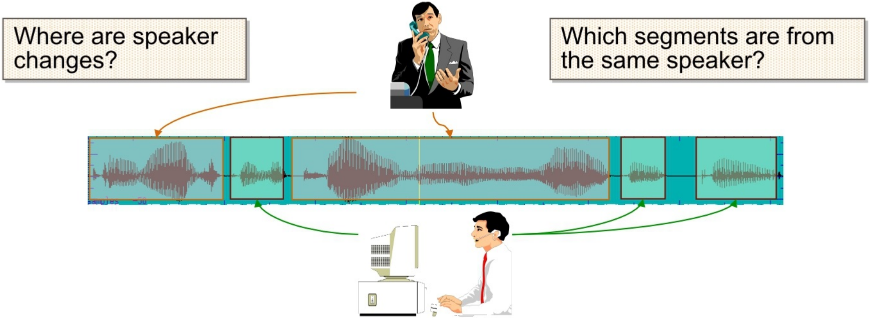
Speaker verification

- Determine whether unknown speaker matches a specific speaker
- One-to-one mapping
- Close-set verification: The population of clients is fixed
- Open-set verification: New clients can be added without having to redesign the system.



Speaker diarization

- Determine when a speaker change has occurred in speech signal (segmentation)
- Group together speech segments corresponding to the same speaker (clustering)
- Prior speaker information may or may not be available

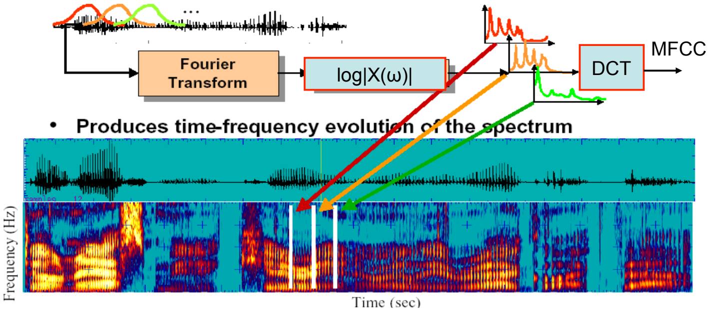


Input mode

- Text-dependent
  - Recognition system knows text spoken by persons
  - Fixed phrases or prompted phrases
  - Used for applications with strong control over user input, e.g., biometric authentication
  - Speech recognition can be used for checking spoken text to improve system performance
  - Sentences typically very short
- Text-independent
  - No restriction on the text, typically conversational speech
  - Used for applications with less control over user input, e.g., forensic speaker ID
  - More flexible but recognition is more difficult
  - Speech recognition can be used for extracting high-level features to boost performance
  - Sentences typically very long

Outline

- Speech is a time-varying signal conveying multiple layers of information
  - Words
  - Speaker
  - Language
  - Emotion
- Information in speech is observed in the time and frequency domains



Feature extraction from speech

- Feature extraction consists in transforming the speech signal to a set of feature vectors. Most of the feature extraction used in speaker recognition systems relies on a cepstral representation of speech.

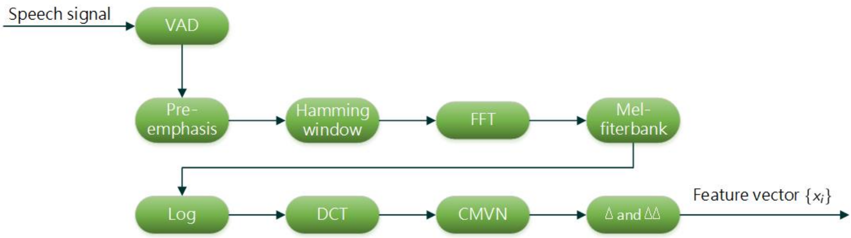
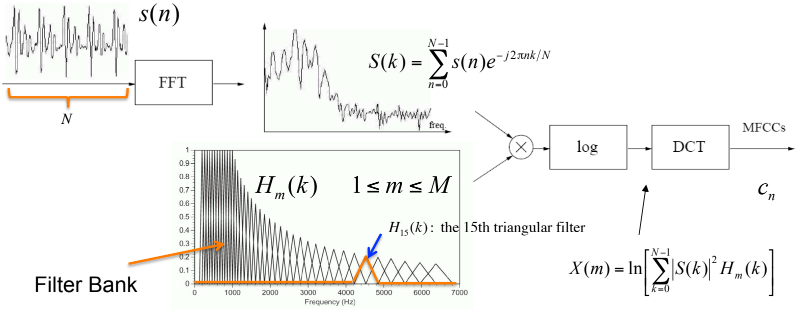


Figure: Modular representation of MFCC feature extractor



$$\mathrm{MFCC}_n = c_n = \sum_{m=1}^{M} \cos\left[n\left(m-\frac{1}{2}\right)\frac{\pi}{M}\right]X(m) = \mathbf{d}_n^T \vec{X}, \qquad 0 \le n \le P$$

$$\Rightarrow \text{MFCC vector} = \begin{bmatrix} c_0 & c_1 & \cdots & c_P \end{bmatrix}^T = \mathbf{D}\vec{X}$$

$\mathbf{D}$: DCT Transformation matrix $[P \times M]$

M : No. of triangular filters in the filter bank, typically 20 ~ 30

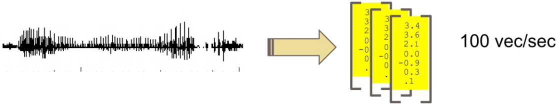
P : No. of cepstral coefficients, typically 12

c_0 : Logarithm of energy of the current frame

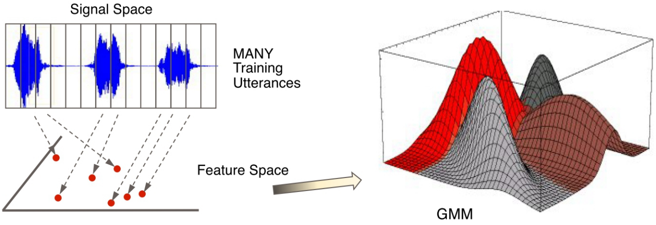
Figure: Computing MFCC from one frame of speech

Modeling sequence of features

- For most recognition tasks, we need to model the distribution of feature vector sequences



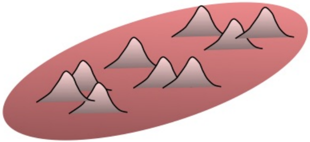
- In practice, we often use the Gaussian mixture models (GMMs)



GMM–UBM speaker verification

- A Gaussian mixture model, namely the universal background model (UBM), is trained to represent the speech of the general population.

$$p(\mathbf{x}|\text{UBM}) = p(\mathbf{x}|\Lambda^{\text{ubm}}) = \sum_{c=1}^{C} \pi_c^{\text{ubm}} \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_c^{\text{ubm}}, \boldsymbol{\Sigma}_c^{\text{ubm}})$$



- The UBM parameters $\Lambda^{\text{ubm}} = \left\{\pi_c^{\text{ubm}}, \boldsymbol{\mu}_c^{\text{ubm}}, \boldsymbol{\Sigma}_c^{\text{ubm}}\right\}_{c=1}^{C}$ are estimated by the expectation-maximization algorithm using the speech of many speakers.

Expectation maximization (EM)

- Denote the acoustic vectors from a large population as $\mathcal{X} = \{\mathbf{x}_t; t = 1, \ldots, T\}$
- **E**xpectation step:
  - Conditional distribution of mixture component c:

$$\gamma_t(c) = p(c|\mathbf{x}_t) = \frac{\pi_c \mathcal{N}(\mathbf{x}_t|\boldsymbol{\mu}_c^{\text{ubm}}, \boldsymbol{\Sigma}_c^{\text{ubm}})}{\sum_{c=1}^{C} \pi_c^{\text{ubm}} \mathcal{N}(\mathbf{x}_t|\boldsymbol{\mu}_c^{\text{ubm}}, \boldsymbol{\Sigma}_c^{\text{ubm}})}$$

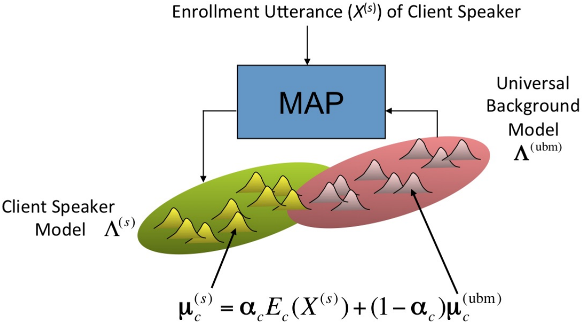
- **M**aximization step:
  - Mixture weights: $\pi_c^{\text{ubm}} = \frac{1}{T}\sum_{t=1}^{T} \gamma_t(c)$
  - Mean vectors: $\boldsymbol{\mu}_c^{\text{ubm}} = \frac{\sum_{t=1}^{T} \gamma_t(c)\mathbf{x}_t}{\sum_{t=1}^{T} \gamma_t(c)}$
  - Covariance matrices: $\boldsymbol{\Sigma}_c^{\text{ubm}} = \frac{\sum_{t=1}^{T} \gamma_t(c)\mathbf{x}_t\mathbf{x}_t^T}{\sum_{t=1}^{T} \gamma_t(c)} - \boldsymbol{\mu}_c^{\text{ubm}}(\boldsymbol{\mu}_c^{\text{ubm}})^{\mathsf{T}}$

Target-speakers' GMMs

- Each target speaker is represented by a Gaussian mixture model:

$$p(\mathbf{x}|\text{Spk } s) = p(\mathbf{x}|\Lambda^{(s)}) = \sum_{c=1}^{C} \pi_c^{(s)} \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_c^{(s)}, \boldsymbol{\Sigma}_c^{(s)})$$

where $\Lambda^{(s)} = \left\{\pi_c^{(s)}, \boldsymbol{\mu}_c^{(s)}, \boldsymbol{\Sigma}_c^{(s)}\right\}_{c=1}^{C}$ are learned by using maximum *a posteriori* (MAP) adaptation [Reynolds et al., 2000].



Maximum *a posteriori* (MAP)

- The MAP algorithm finds the parameters of target-speaker's GMM given UBM parameters $\Lambda^{\text{ubm}} = \left\{\pi_c^{\text{ubm}}, \boldsymbol{\mu}_c^{\text{ubm}}, \boldsymbol{\Sigma}_c^{\text{ubm}}\right\}_{c=1}^{C}$
- First step is the same as EM. Given T_s acoustic vectors $\mathcal{X}^{(s)} = \{\mathbf{x}_1, \ldots, \mathbf{x}_{T_s}\}$ from speaker s, we compute the statistics:

$$n_c = \sum_{t=1}^{T_s} \gamma_t(c) \quad \text{and} \quad E_c(\mathcal{X}^{(s)}) = \frac{1}{n_c} \sum_{t=1}^{T_s} \gamma_t(c)\mathbf{x}_t$$

- Adapt UBM parameters by

$$\boldsymbol{\mu}_c^{(s)} = \alpha_c E_c(\mathcal{X}^{(s)}) + (1 - \alpha_c)\boldsymbol{\mu}_c^{\text{ubm}}$$

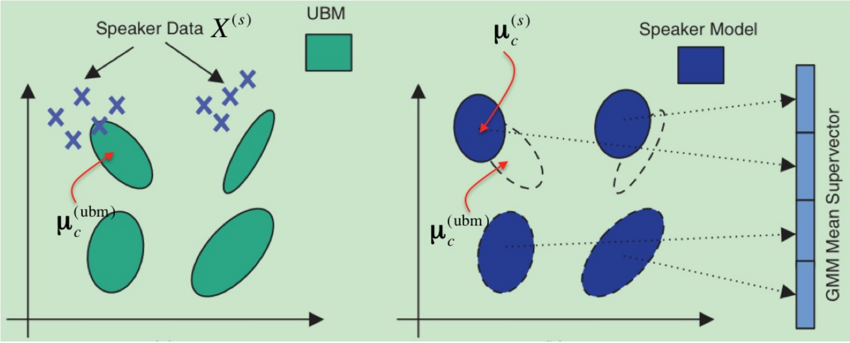
where

$$\alpha_c = \frac{n_c}{n_c + r}$$

and r is called the relevance factor. Typically, $r = 16$.

MAP adaptation

- Adapt the UBM model to each speaker using the MAP algorithm:[1]



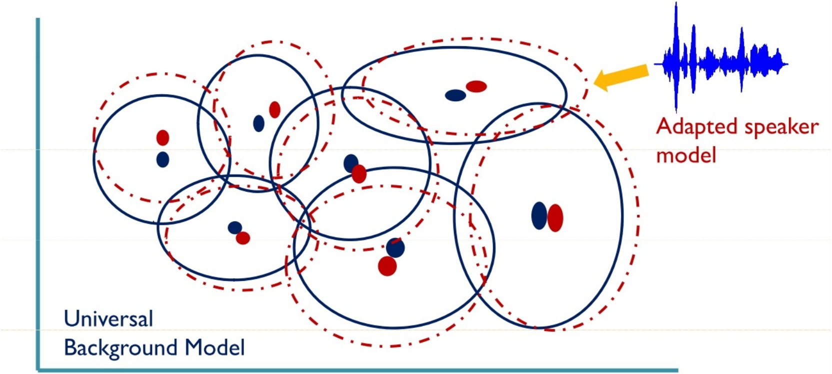
$$\boldsymbol{\mu}_c^{(s)} = \alpha_c E_c(\mathcal{X}^{(s)}) + (1 - \alpha_c)\boldsymbol{\mu}_c^{\mathrm{ubm}}$$

- $\alpha_c \to 1$ when $\mathcal{X}^{(s)}$ comprises lots of data and $\alpha_c \to 0$ otherwise.

[1]Source: J. H. L. Hansen and T. Hasan, *IEEE Signal Processing Magazine*, 2015.

MAP adaptation

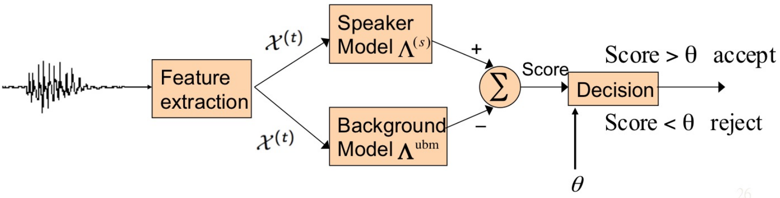
- In practice, only the mean vectors will be adapted:



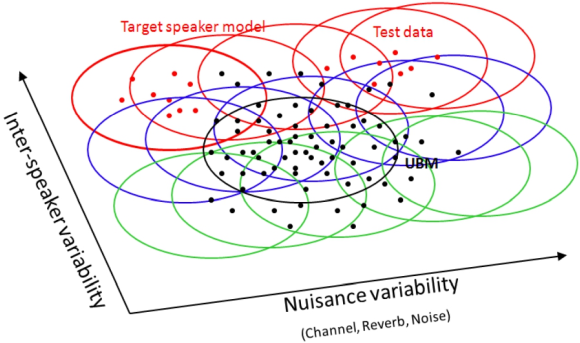
GMM-UBM scoring

- Given the acoustic vectors $\mathcal{X}^{(t)}$ from a test speaker and a claimed identity s, speaker verification can be formulated as a 2-class hypothesis problem:
  - H_0: $\mathcal{X}^{(t)}$ comes from the true speaker s
  - H_1: $\mathcal{X}^{(t)}$ comes from an impostor
- Verification score is a log-likelihood ratio:

$$S_{\text{LR}}(\mathcal{X}^{(t)}|\Lambda^{(s)}, \Lambda^{\text{ubm}}) = \log p(\mathcal{X}^{(t)}|\Lambda^{(s)}) - \log p(\mathcal{X}^{(t)}|\Lambda^{\text{ubm}})$$



Sources of variability



How to account for variability

- **GMM-SVM** [Campbell et al., 2006]:
  - Create supervectors from target-speaker GMMs.
  - Then, project the supervectors to a subspace in which inter-speaker variability is maximized and nuisance variability is minimized.
  - Perform SVM classification on the projected subspace.
- **Joint Factor Analysis**:
  - Speaker and session variabilities are represented by latent variables (speaker factors and channel factors) in a factor analysis model.
  - During scoring, session variabilities are accounted for by integrating over the latent variables, e.g., the channel factors as in [Kenny et al., 2007a].
- **I-Vector + PLDA**:[2]
  - Utterances are represented by the posterior means of latent factors, called the i-vectors [Dehak et al., 2011].
  - I-vectors capture both speaker and channel information.
  - During scoring, the unwanted channel variability is removed by LDA projection or by integrating out the latent factors in the PLDA model.

[2] For the relationship between JFA and I-vectors and their derivations, see http://www.eie.polyu.edu.hk/∼mwmak/papers/FA-Ivector.pdf

Performance measures

- For speaker identification

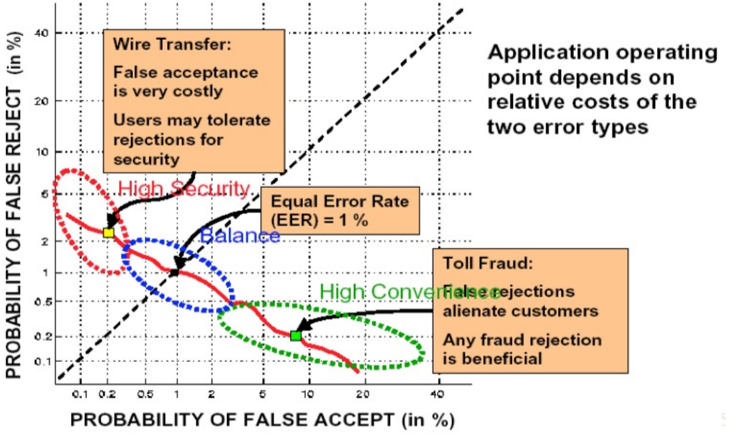
$$\text{Recogntion Rate} = \frac{\text{No. of correct recognitions}}{\text{Total no. of trials}}$$

- For speaker verification

$$\begin{aligned}\text{False Rejection Rate (FRR)} &= \text{Miss probability}\\ &= \frac{\text{No. of true-speakers rejected}}{\text{Total no. of true-speaker trials}}\\ \text{False Acceptance Rate (FAR)} &= \text{False alarm probability}\\ &= \frac{\text{No. of impostors accepted}}{\text{Total no. of impostor attempts}}\end{aligned}$$

- Equal error rate (EER) corresponds to the operating point at which FAR = FRR

- Detection error tradeoff (DET) curves are similar to receiver operating characteristic curves but with nonlinear x- and y-axis.



Detection cost functions

- Detection cost function (DCF) is a weighted sum of the FRR ($P_{\text{Miss}|\text{Target}}$) and FAR ($P_{\text{FalseAlarm}|\text{Nontarget}}$):

$$C_{\text{Det}}(\theta) = C_{\text{Miss}} \times P_{\text{Miss}|\text{Target}}(\theta) \times P_{\text{Target}} + C_{\text{FalseAlarm}} \times P_{\text{FalseAlarm}|\text{Nontarget}}(\theta) \times (1 - P_{\text{Target}})$$

  where θ is a decision threshold.

- Normalized cost:

$$C_{\text{Norm}} = C_{\text{Det}}(\theta)/C_{\text{Default}}$$

  where

$$C_{\text{Default}} = \min \begin{cases} C_{\text{Miss}} \times P_{\text{Target}} \\ C_{\text{FalseAlarm}} \times (1 - P_{\text{Target}}) \end{cases}$$

  - NIST 2008 SRE and earlier:

$$C_{\text{Miss}} = 10; \quad C_{\text{FalseAlarm}} = 1; \quad P_{\text{Target}} = 0.01$$

  - NIST 2010 SRE:

$$C_{\text{Miss}} = 1; \quad C_{\text{FalseAlarm}} = 1; \quad P_{\text{Target}} = 0.001$$

Detection cost functions

- Dectection cost function for NIST 2012 SRE:

$$
\begin{aligned}
C_{\text{Det}}(\theta) = C_{\text{Miss}} \times P_{\text{Miss}|\text{Target}}(\theta) \times P_{\text{Target}} \; + \\
C_{\text{FalseAlarm}} \times (1 - P_{\text{Target}}) \; \times \\
[P_{\text{FalseAlarm}|\text{KnownNontarget}}(\theta) \times P_{\text{Known}} \; + \\
P_{\text{FalseAlarm}|\text{UnKnownNontarget}} \times (1 - P_{\text{Known}}])
\end{aligned}
$$

$$C_{\text{Norm}}(\theta) = C_{\text{Det}}(\theta)/(C_{\text{Miss}} \times P_{\text{Target}})$$

  - Parameters for core test conditions

    $C_{\text{Miss}} = 1;\; C_{\text{FalseAlarm}} = 1;\; P_{\text{Target1}} = 0.01;\; P_{\text{Target2}} = 0.001;$
    $P_{\text{Known}} = 0.5$

  - $P_{\text{Target1}} \rightarrow C_{\text{Norm1}}(\theta_1)$ and $P_{\text{Target2}} \rightarrow C_{\text{Norm2}}(\theta_2)$

- Primary cost:

$$C_{\text{Primary}} = \frac{C_{\text{Norm1}}(\theta_1) + C_{\text{Norm2}}(\theta_2)}{2}$$

Detection cost functions

- Detection cost function for NIST 2016 SRE:

$$C_{\text{Det}}(\theta) = C_{\text{Miss}} \times P_{\text{Miss}|\text{Target}}(\theta) \times P_{\text{Target}} + C_{\text{FalseAlarm}} \times P_{\text{FalseAlarm}|\text{Nontarget}}(\theta) \times (1 - P_{\text{Target}})$$

$$C_{\text{Norm}}(\theta) = C_{\text{Det}}(\theta)/(C_{\text{Miss}} \times P_{\text{Target}})$$

  - Parameters for core test conditions

$$C_{\text{Miss}} = 1;\ C_{\text{FalseAlarm}} = 1;\ P_{\text{Target1}} = 0.01;\ P_{\text{Target2}} = 0.005$$

  - $P_{\text{Target1}} \rightarrow C_{\text{Norm1}}(\theta_1)$ and $P_{\text{Target2}} \rightarrow C_{\text{Norm2}}(\theta_2)$

- Primary cost:

$$C_{\text{Primary}} = \frac{C_{\text{Norm1}}(\theta_1) + C_{\text{Norm2}}(\theta_2)}{2}$$

Outline

- Main approach
  - Nonparametric model
    - Support vector machine
  - Parametric model
    - Gaussian mixture model
    - Probabilistic linear discriminant analysis
    - Joint factor analysis
    - i-vector
  - Artificial neural network
    - Auto-encoder
    - Deep neural network
    - Restricted Boltzmann machine