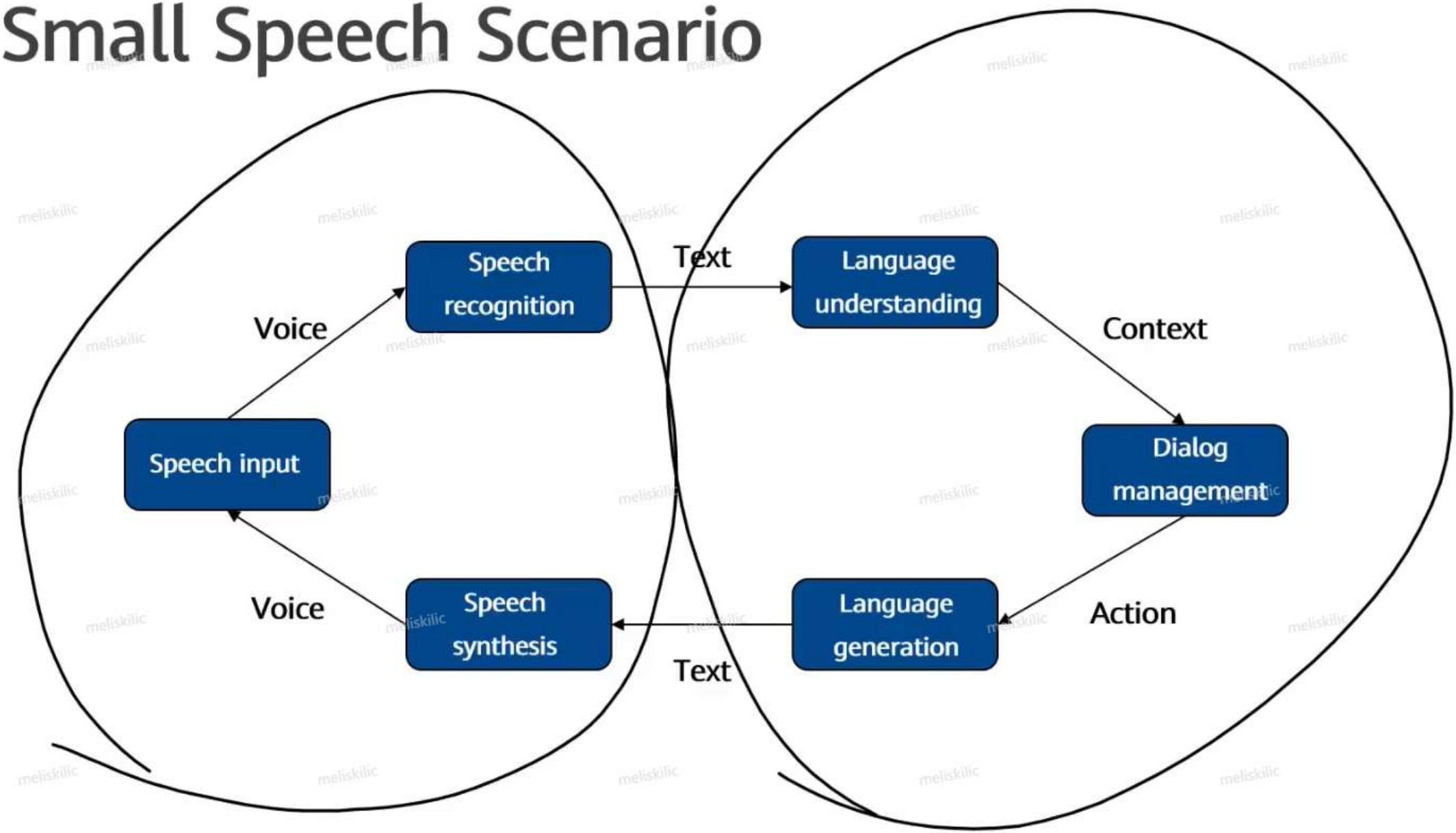
Small Speech Scenario



Main Application Scenarios of Speech Processing

Technology

- Speech preprocessing
- Speech recognition
- Speaker recognition
- Speech translation
- Speech synthesis
- Voiceprint recognition
- Speech coding

Scenario

- Man-machine interaction
- Security protection
- Smart home
- Smart city
- Elderly care
- Education
- Customer service

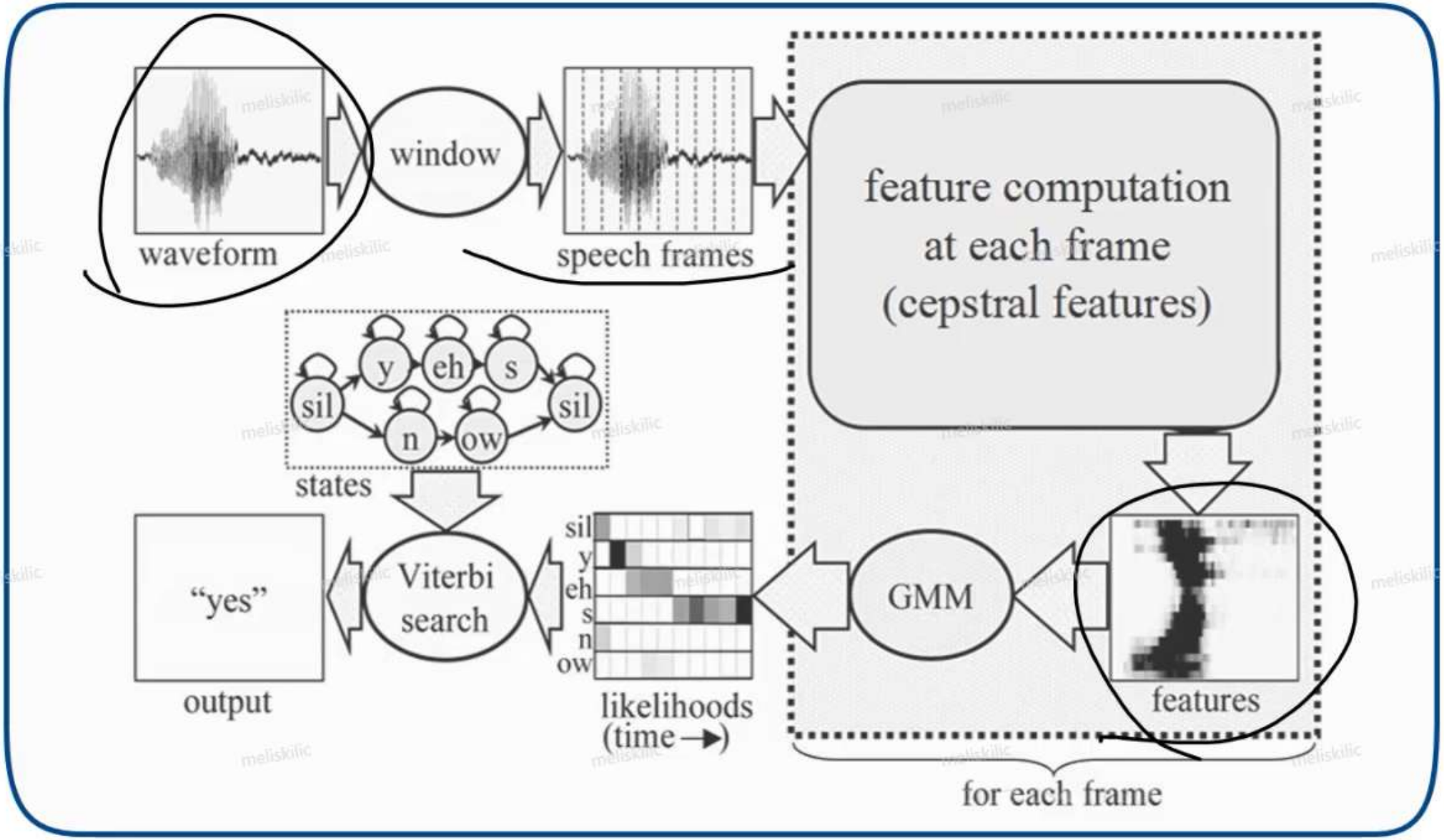
Main Speech Recognition Technologies

- GMM
- HMM
- EM
- Baum-Welch
- Viterbi
- N-gram
- DNN
- RNNs
- CNN

Acoustic

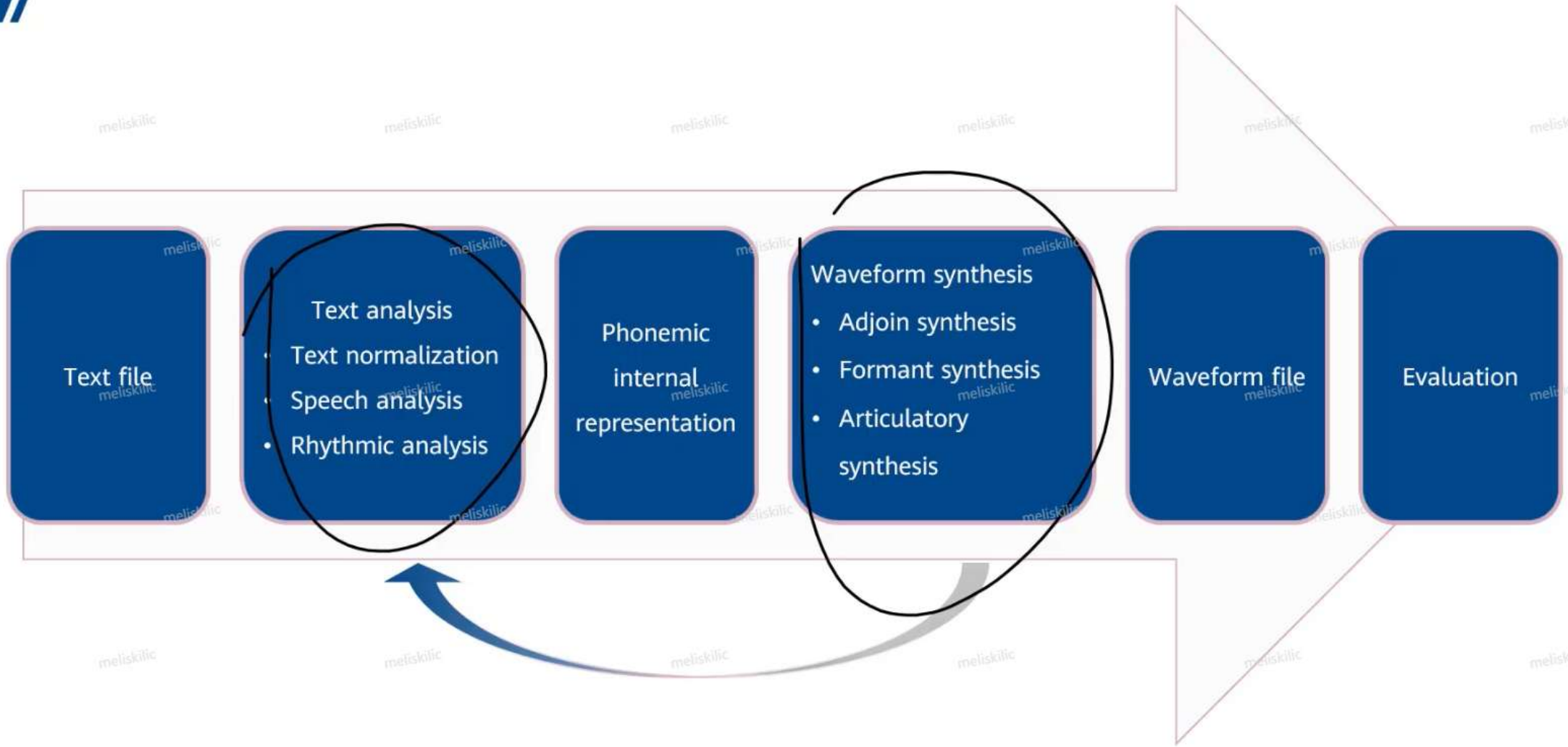
- AM
- LM
- BP
- BPTT
- Cross Entropy
- MFCC
- DBN
- RBM
- SGD

Speech Recognition Flowchart



(Source: From Encyclopedia of Information System, 2002)

Speech Synthesis Flow



Speech Synthesis Method

Text Analysis:

- Text normalization
- Speech analysis
- Rhythmic analysis

Speech Synthesis :

- Parameter synthesis
- Waveform concatenation

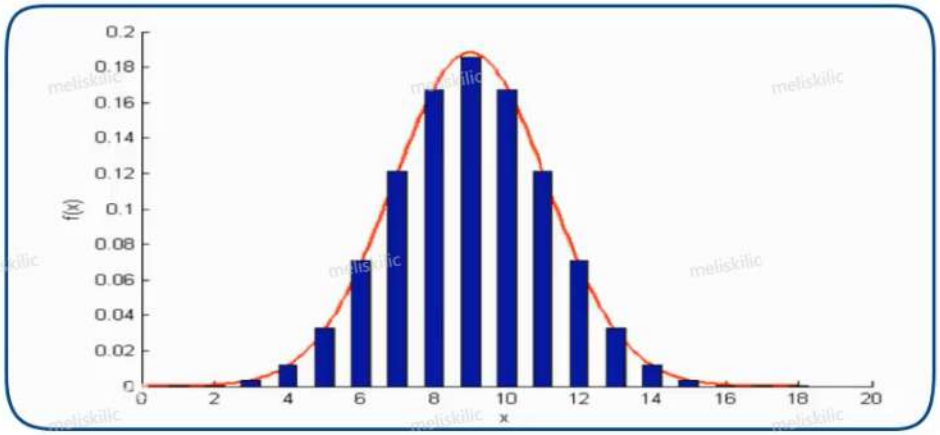
Gaussian Distribution

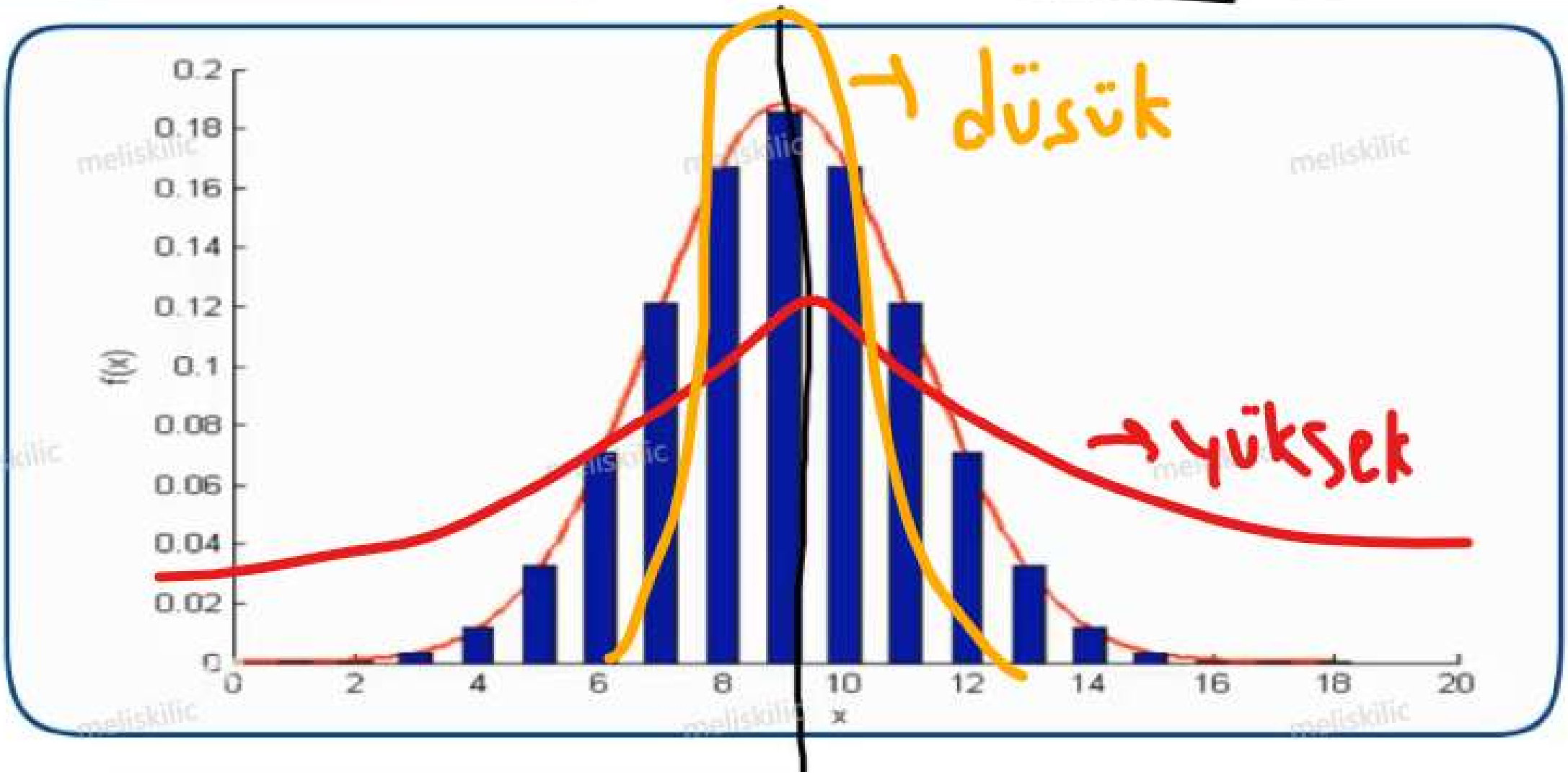
Also called SGM. If the random variable X follows a mathematical expectation μ, the variance is the normal distribution of σ^2 and is recorded as $N(\mu, \sigma^2)$. The probability density function (PDF) of the random variable is a normal distribution and the expectation value μ decides the position of the variable and the standard variance σ decides the distribution amplitude. When μ is 0 and σ is 1, the normal distribution is a standard normal distribution.

The formula is as follows:

one-dimensional: $$f(x) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

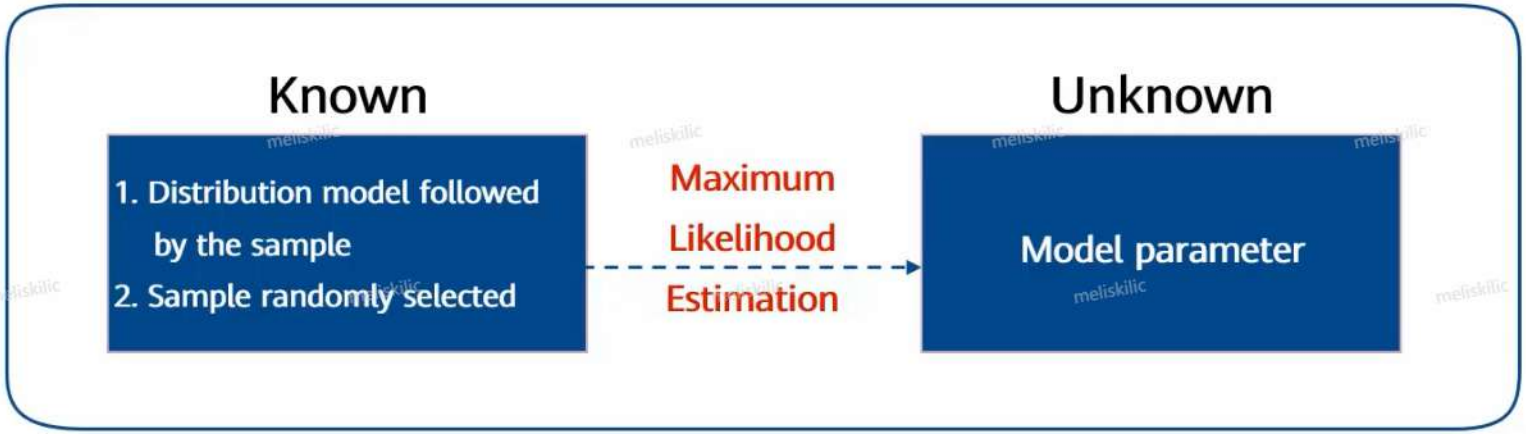
multi-dimensional: $$P(x|\theta) = \frac{1}{(2\pi)^{\frac{D}{2}}} e^{-\frac{(x-\pi)^T \sum (x-\mu)^{-1}}{2}}$$




düşük
yüksek
f(x)
x
0.2
0.18
0.16
0.14
0.12
0.1
0.08
0.06
0.04
0.02
0
0
2
4
6
8
10
12
14
16
18
20

Introduction to Maximum Likelihood

Maximum likelihood (ML), also called maximum likelihood estimation, is a theoretical point estimation method. The maximum likelihood estimation is a statistical method. It is used to solve the parameter of the related PDF for a sample set.



Maximum Likelihood

Assume N independent data points follow a distribution $Pr(x;\theta)$. We want to find a group of parameters θ to make the maximum probability of generating the data points. The probability is:

$$\prod_{i=1}^{N} Pr(x_i;\theta)$$

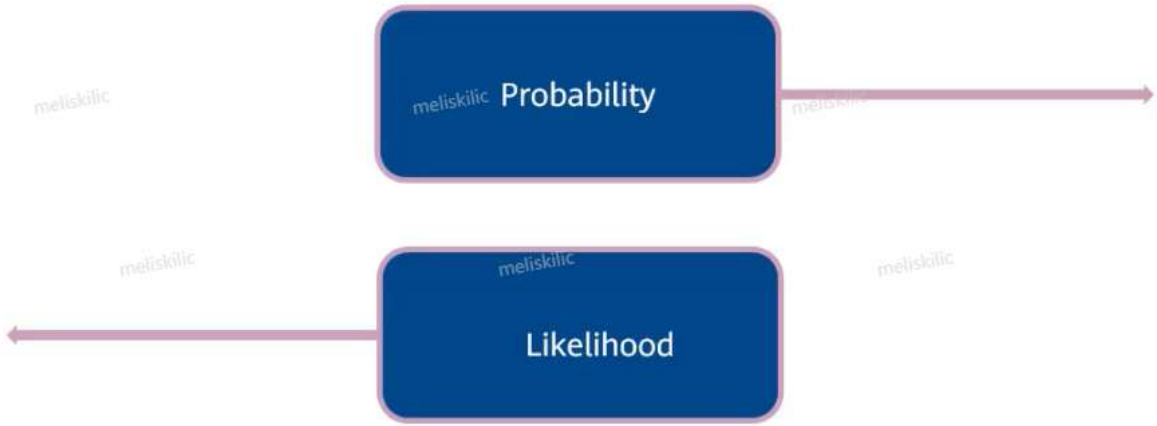
It is called a likelihood function. Generally, the probability of one point is lower. After continuous multiplication, data becomes smaller, which may cause floating point underflow. Therefore, the logarithm of data is used. As a result, the probability becomes:

$$\sum_{i=1}^{N} logPr(x_i;\theta)$$

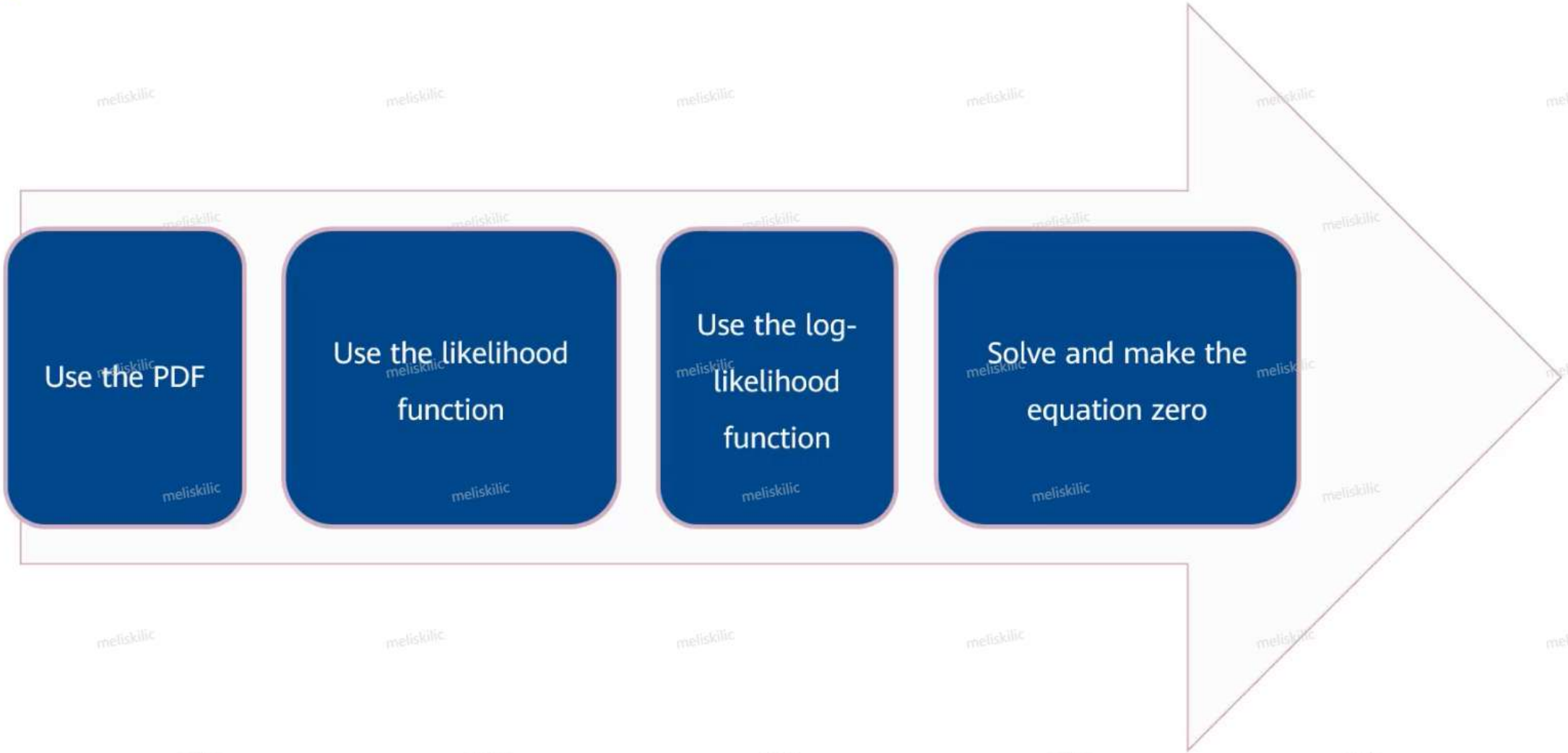
It is called a log-likelihood function. Then derivation can be performed to find the parameter θ that makes the value of the preceding formula the greatest. We think the possibility of obtaining the observed values is the lowest but the parameter θ makes this happen at the maximum likelihood.

Probability and Likelihood

Probability and likelihood are similar in concepts but they are also different. For example, if the random variable X follows a distribution (for example, Gaussian distribution), the probability refers to the possibility that X equals x under the condition of the given parameter and likelihood refers to the authenticity of a group of parameters under the condition that X equals x.

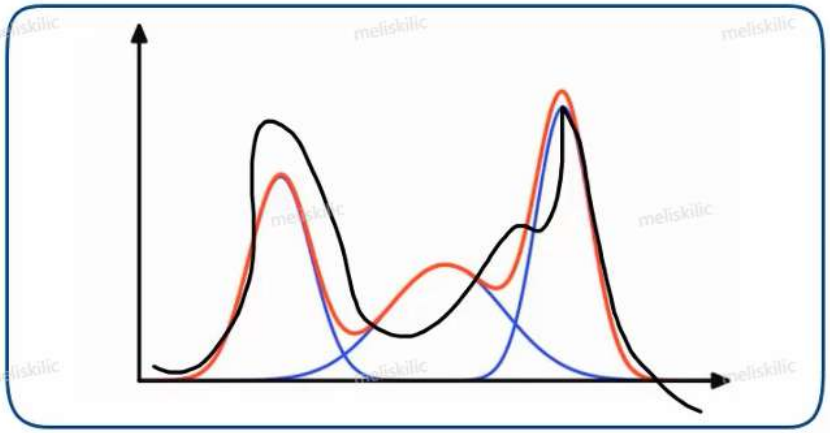


Maximum Likelihood Estimation Process



Concept of Mixture Model

A mixture model is a probabilistic model that can be used to indicate K sub-distributions of the general distribution. In other words, the mixture model indicates the probabilistic distribution of observational data in general data. It is a mixed distribution consisting of K sub-distributions.

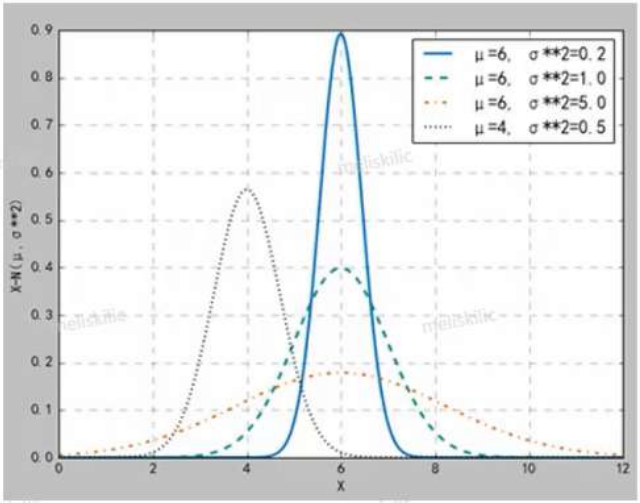


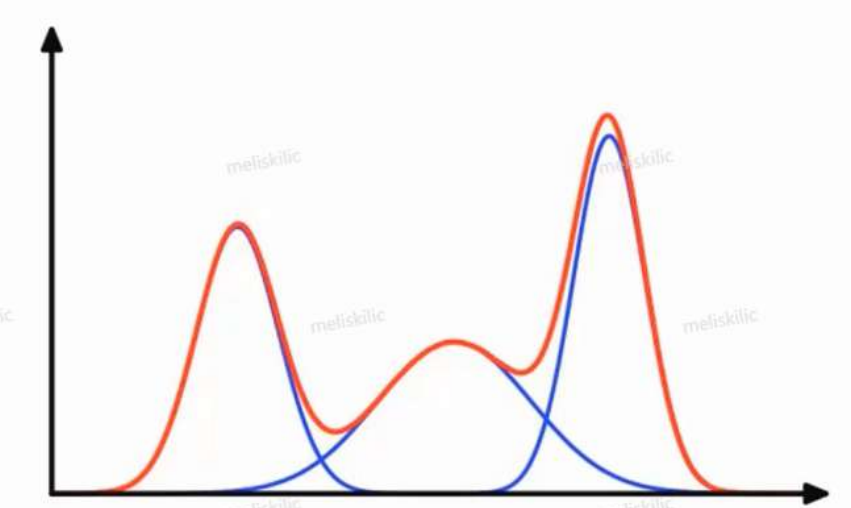
Definition of a mixture model:

$$y(i,t) = \alpha + X(i,t)\beta + \varepsilon(i,t).(i = 1,2,3,\dots,N; t = 1,2,3,\dots T)$$

GMM Introduction

GMM can also be abbreviated as Mixture of Gaussian (MOG). A GMM is used to accurately quantify a thing using the Gaussian density function (normal distribution curve) and break down one thing into several models formed based on the Gaussian probability density function (PDF).





GMM

Probability distribution of the GMM:

$$P(x|\theta) = \sum_{k=1}^{K} \alpha_k \phi(x|\theta_k)$$

For the GMM, the parameter $\theta = (\widetilde{\mu_k}, \widetilde{\sigma_k}, \widetilde{\alpha_k})$ indicates the occurrence probability of the expectation and variance (or covariance) of each sub-model in the GMM.

Parameter description:

x_j indicates the jth observed data. $j = 1,2,3, \ldots, N$. K indicates the number of Gaussian models in a GMM; α_k indicates the probability that observed data belongs to the k_{th} sub-model. $\alpha_k \geq 0$, $\sum_{k=1}^{K} \alpha_k = 1$. $\emptyset(x|\theta_k)$ indicates the Gaussian PDF of the k_{th} sub-model: $\theta_k = (\mu_k, \sigma^2{}_k)$.

Parameter Learning of GMM

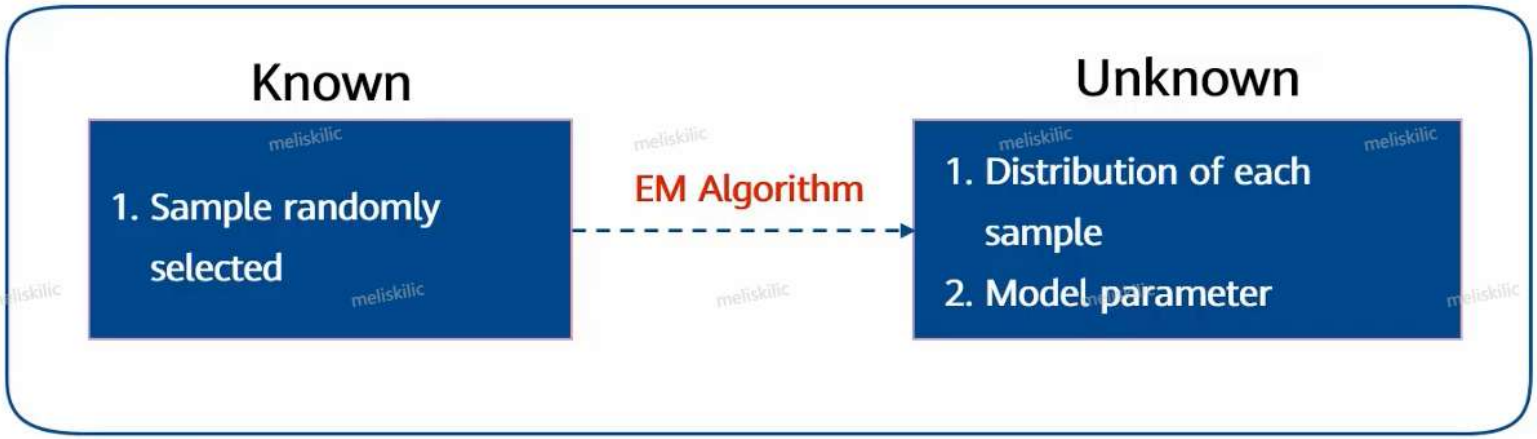
For a GMM, the log-likelihood function is:

$$\log L(\theta) = \sum_{j=1}^{N} logP(x_j|\theta) = \sum_{j=1}^{N} \log(\sum_{k=1}^{N} \alpha_k \emptyset(x|\theta_k))$$

How to calculate parameters of a GMM? We cannot use maximum likelihood to derive and calculate the parameter that makes likelihood the maximum like a Gaussian model because we do not know which sub-distribution (hidden variable) an observed data point belongs to. Therefore, summation also exists in the log. The sum of K Gaussian models is not a Gaussian model. For each sub-model, α_k, μ_k, and σ_k are unknown and cannot be calculated through direct derivation. The parameters must be solved through an iterative approach.

EM Algorithm

The expectation maximization (EM) algorithm is an iterative algorithm. It is used for the maximum likelihood estimation(MLE) or maximum a posterior estimation(MAP) of probability parameter models that contain hidden variables.



Solving of the EM Algorithm

Initialize parameters:

E-step: According to current parameters, calculate the possibility that each data j comes from the sub-model k.

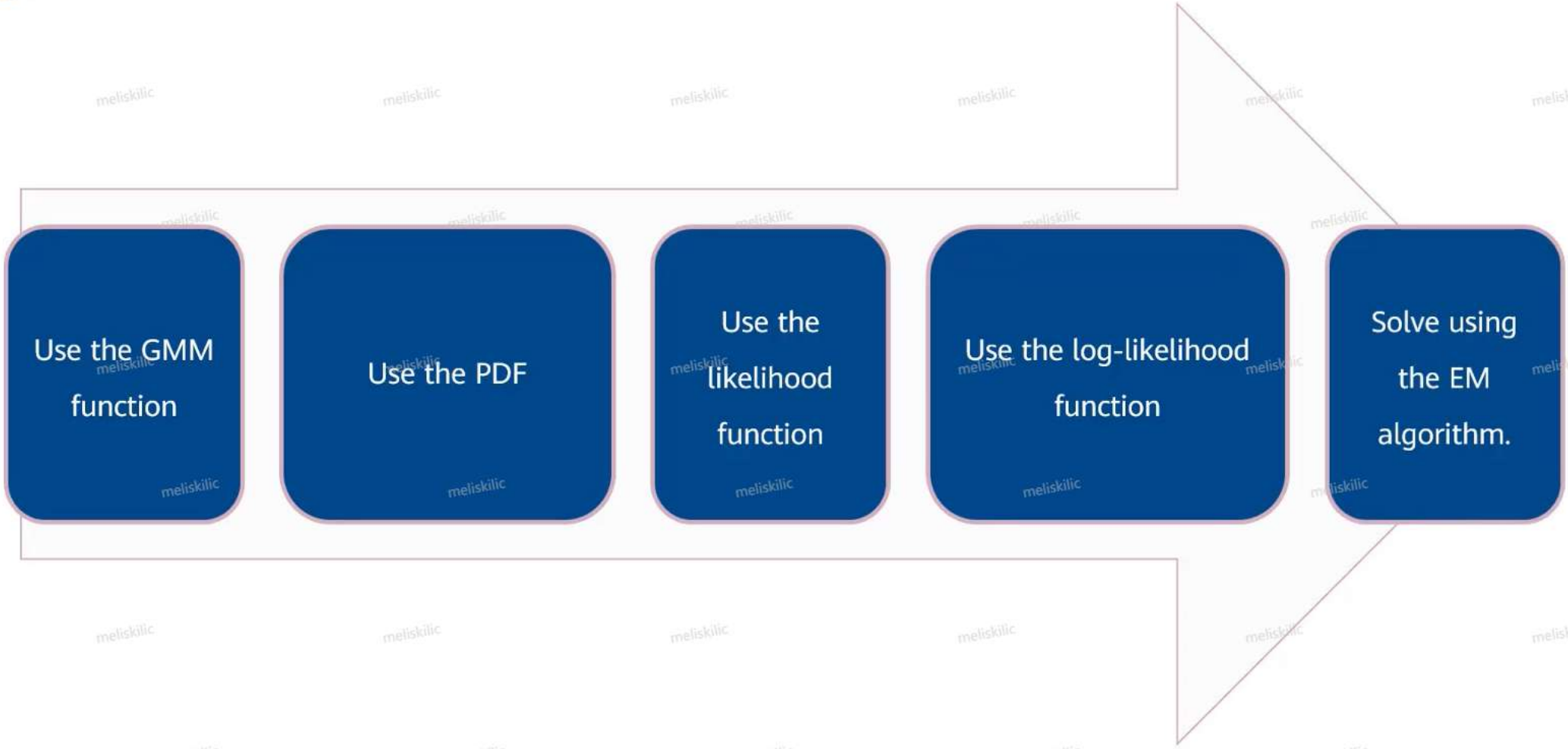
$$\gamma_{\mathrm{jk}} = \frac{\alpha_k \emptyset(x_j|\theta_k)}{\sum_{k=1}^{K} \alpha_k \emptyset(x_j|\theta_k)}$$

Where: $j = 1,2,3,\ldots,N;\ k = 1,2,3,\ldots,K$

M-step: Calculate the model parameter in a new round of iteration.

$$\mu_k = \frac{\sum_j^N (\gamma_{jk} x_j)}{\sum_j^N \gamma_{jk}}, \text{ where: } k = 1,2,3,\ldots,N$$

GMM Learning Steps



Disadvantages of GMM

Advantages

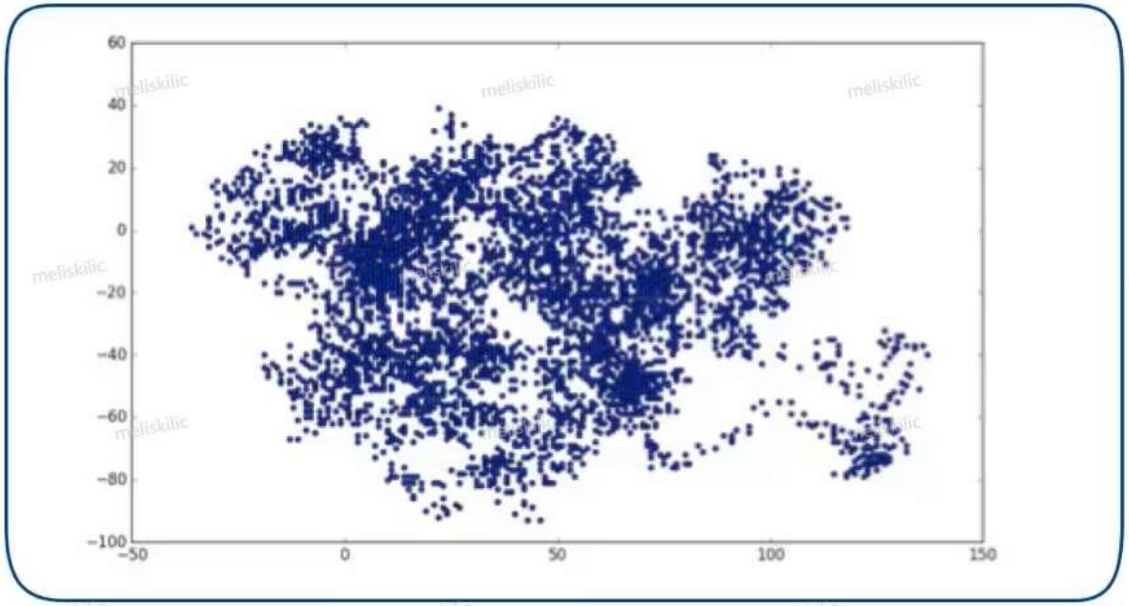
- Strong fitting capability
- Maximum probability of speech feature matching

Disadvantages

- The sequence factor cannot be processed.
- Linear or approximate linear data cannot be processed.

Random Walk

Random walk is a mathematical statistical model that consists of a series of tracks. Each track is random. Random walk can be used to indicate an irregular form of change.



Cases of Markov Chain (1)

The publicity for commodities A, B, and C under a category is different. The probabilities for customers to attempt to select and buy commodities A, B, and C under the advertising effect are respectively 0.2, 0.4, and 0.4. The following table describes purchase predisposition of customers. Find the probabilities for customers to buy the commodities the fourth time.

		Second Purchase		
		A	B	C
First Purchase	A	0.8	0.1	0.1
	B	0.5	0.1	0.4
	C	0.5	0.3	0.2

Cases of Markov Chain (2)

Three elements:

- Initial probability: $\pi = (0.2, 0.4, 0.4)$
- Transition probability: $p_{AA} = 0.8, p_{AB} = 0.1, p_{AC} = 0.1, p_{BA} = 0.5, \ldots$
- Transition probability matrix: $A = \begin{pmatrix} 0.8 & 0.1 & 0.1 \\ 0.5 & 0.1 & 0.4 \\ 0.5 & 0.3 & 0.2 \end{pmatrix}$

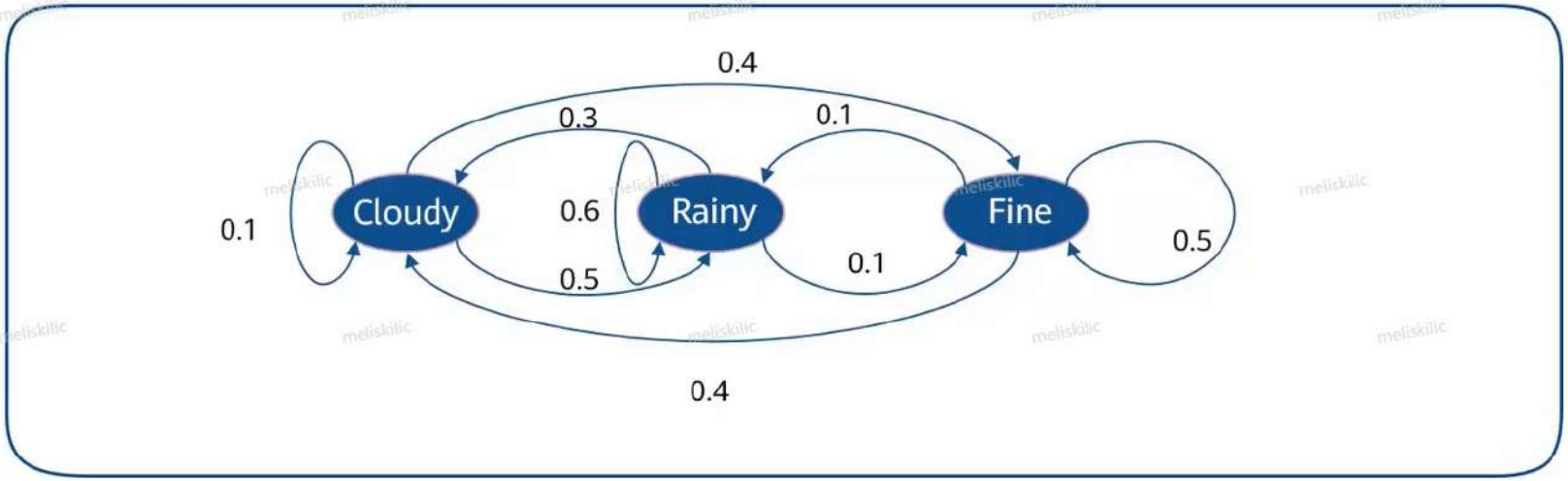
Solving:

- Model of Markov chain:
- $\mathrm{P}(X_{n+1} = x | X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n) = \mathrm{P}(X_{n+1} = x | X_n = x_n)$
- Third probability: $P_{AAA} = ?$

$0.2 \times 0.8 \times 0.8 = 0.128$

Markov Chain

Markov chain refers to a random discrete event process with Markov properties in mathematics. In the process, when current knowledge or information is provided, future prediction is unrelated to the past and is related to only the current state.



Observable Markov Model

For one question, we have initial distribution π and transition probability matrix A. In any given time t, we have a state Q_t. When one state transits to another with the change of time, an observation sequence is obtained, that is, state sequence $O = [q_1, q_2, q_3, q_4, \dots, q_n]$. In the whole question, there are n observation states in total.

The probability of such a sequence is:

$$P(O|A,\pi) = P(q_1)\prod_{t=2}^{m} P(q_t|q_{t-1})$$

Therefore, an observable Markov model has a triplet description (A, π, n), which can be abbreviated as (A, π).

Method of Exhaustion

If we list all observation sequences...

$$\pi_i = \frac{Number\ of\ sequences\ starting\ with\ state\ i}{Total\ number\ of\ sequences}$$

$$p_{ij} = \frac{Number\ of\ sequences\ transiting\ to\ state\ j\ from\ state\ i}{Total\ number\ of\ sequences\ from\ state\ i}$$

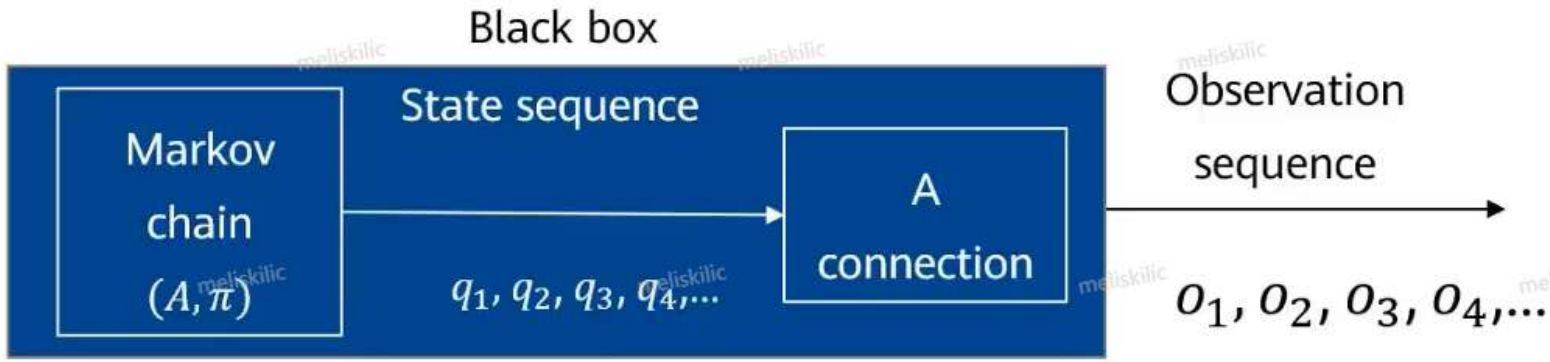
[Red, red, red], [red, red, blue], [red, blue, red], [blue, red, red]

$$\pi = \{0.75, 0.25\}$$

$$p_{red,blue} = \frac{1}{3},\ p_{red,red} = \frac{2}{3}, p_{blue,blue} = 0,\ p_{blue,red} = 1$$

Hidden Markov Model

The hidden Markov model (HMM) is a type of Markov chain.



HMM Description

The HMM can be described using five elements:

- Observation set: $R = \{R_1, R_2, R_{3,\ldots,} R_m\}$
- Observation sequence: O= $\{o_1, o_2, o_{3,\ldots,} o_l\}$
- State set: S= $\{S_1, S_2, S_{3,\ldots,} S_n\}$
- State sequence: Q= $\{q_1, q_2, q_{3,\ldots,} q_l\}$
- Observation probability: $P\{o_j = R_k | q_t = S_j\} = b_j(i), B = [b_j(i)]$

Therefore, the HMM can be described using a quintet $\lambda = (A, B, \pi, R, S)$, which can be abbreviated as $\lambda = (A, B, \pi)$.

Three Issues of the HMM

Evaluation:

- Forward algorithm
- Backward algorithm

Decoding

- Dynamic planning algorithm
- Viterbi algorithm

Learning

- Supervised algorithm
- Unsupervised Baum-Welch algorithm

HMM Evaluation - Forward Algorithm

Evaluation means to calculate the likelihood of a specific observation sequence in the HMM, specifically, an HMM, a parameter $\lambda = (A, B)$, and an observation sequence $O = o_1 o_2 \ldots o_T$ are provided to calculate the likelihood of the observation sequence $P(O|\lambda)$.

Steps of the forward algorithm:

- Initialization: $\alpha_1(j) = \alpha_{0j} \times b_j(o_1)\ (1 \leq j \leq N)$
- Recursion: $\alpha_t(j) = \sum_{i=1}^{N} a_{t-1}(i) \times a_{ij} \times b_j(o_t)\ (1 \leq j \leq N, 1 \leq t \leq \mathrm{T})$
- Termination: $P(O|\lambda) = \alpha_T(qF) = \sum_{i=1}^{N} \alpha_T(i) \times \alpha_i F$

HMM Evaluation - Backward Algorithm

The backward algorithm is similar to the forward algorithm. It defines a concept called backward probability: a Markov model λ is given and the observation sequence $o_{t+1}, o_{t+2}, \ldots, o_T$ from $t+1$ to T is defined as the backward probability under the conditions where the time is t and the state is q_i, which is recorded as:

$$\beta_t(i) = P(o_{t+1}, o_{t+2}, \ldots, o_T | i_t = q_i, \lambda)$$

Similarly, the backward probability $\beta_t(i)$ and observation probability $P(O|\lambda)$ can be obtained using a recursive method.

HMM Learning - Supervised

When the observation sequence and state sequence are given, obtain the HMM.

By using the law of large numbers, estimate three probabilities of HMM based on frequency.

- Initial probability: $\widehat{\pi_i} = \frac{|q_i|}{\sum_i |q_i|}$
- Transition probability: $\widehat{a_{ij}} = \frac{|q_{ij}|}{\sum_{j=1}^{N} |q_{ij}|}$
- Observation probability: $\widehat{b_{ij}} = \frac{|S_{ik}|}{\sum_{k=1}^{M} |q_{ij}|}$

HMM Learning - Unsupervised Bauw-Welch

When an observation sequence is provided but no state sequence is provided, obtain the HMM.

In fact, the algorithm is essentially the EM algorithm because it solves the following question: When the observed value X is available and the observed value has a hidden variable Z, obtain the joint probability under the HMM parameter λ $P(X, Z|\lambda)$.

Solving steps

- Determine the log-likelihood function of complete data.
- E-step of the EM algorithm: Solve the Q function $Q(\lambda, \hat{\lambda})$.
- M-step of the EM algorithm: Obtain parameters of the maximum Q function.

HMM Decoding - Viterbi

Viterbi algorithm is a special dynamic planning algorithm that is the most widely used. It is proposed for the shortest path of the directed graph of the fence network. Questions described using an HMM can be decoded using the Viterbi algorithm, including current digital communication, speech recognition, and machine translation.

Steps

- Initialization
- Recursion
- Termination
- Optimal path backtracking

Role of GMM - HMM (1)

Role of GMM

- A GMM is used to obtain the probability of a factor.
- A GMM consists of three to five superimposed Gaussian models.
- A Gaussian model is a normal distribution and represents the probability density of signals.
- In speech recognition, one word consists of multiple phonemes. One phoneme is consist of many states and one state is associated with a GMM.
- Each GMM has K model parameters.
- (Here, the corresponding model is a GMM, that is, K Gaussian weights corresponding to each state, each Gaussian average vector, and variance matrix.)

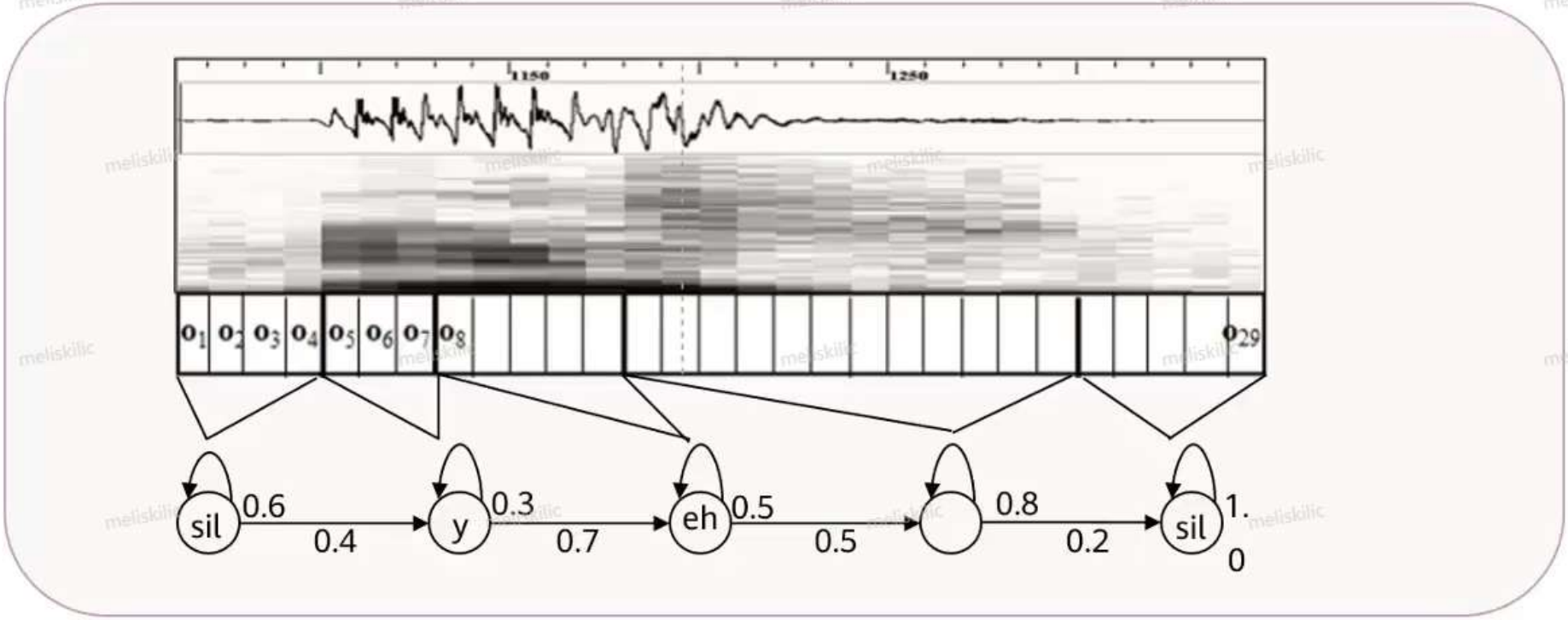
Role of GMM - HMM (2)

An HMM performs speech modeling

- An HMM is created for each state. Training samples of the word are used. The training samples are labeled in advance, that is, each sample corresponds to a section of audio and the audio contains only the pronunciation of the word.
- After multiple training samples of the word are available, the samples, together with the Baum-Welch algorithm and the EM algorithm, are used to train out all parameters of GMM-HMM. These parameters include the probability vector of the initial state, inter-state transition matrix, and observation matrix corresponding to each state.

GMM - HMM Speech Recognition

A wav file is obtained. The following shows the process of recognizing a word:

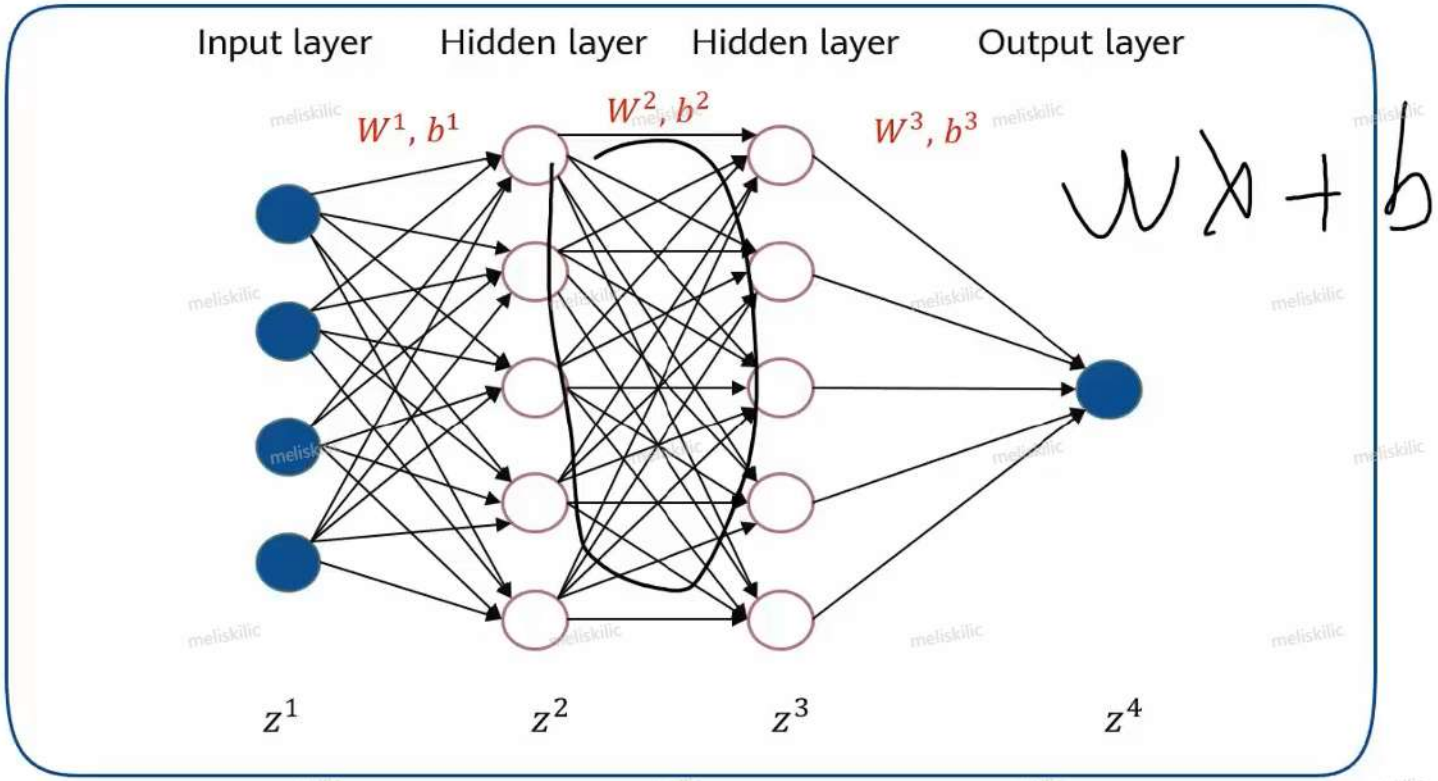


$$b_{sil}(o_1) \cdot 0.6 \cdot b_{sil}(o_2) \cdot 0.6 \cdot b_{sil}(o_3) \cdot 0.6 \cdot b_{sil}(o_4) \cdot 0.4 \cdot b_y(o_5) \cdot 0.3 \cdot b_y(o_6) \cdot \ 0.3 \cdot b_y(o_7) \cdot \ 0.7 \cdots$$

s

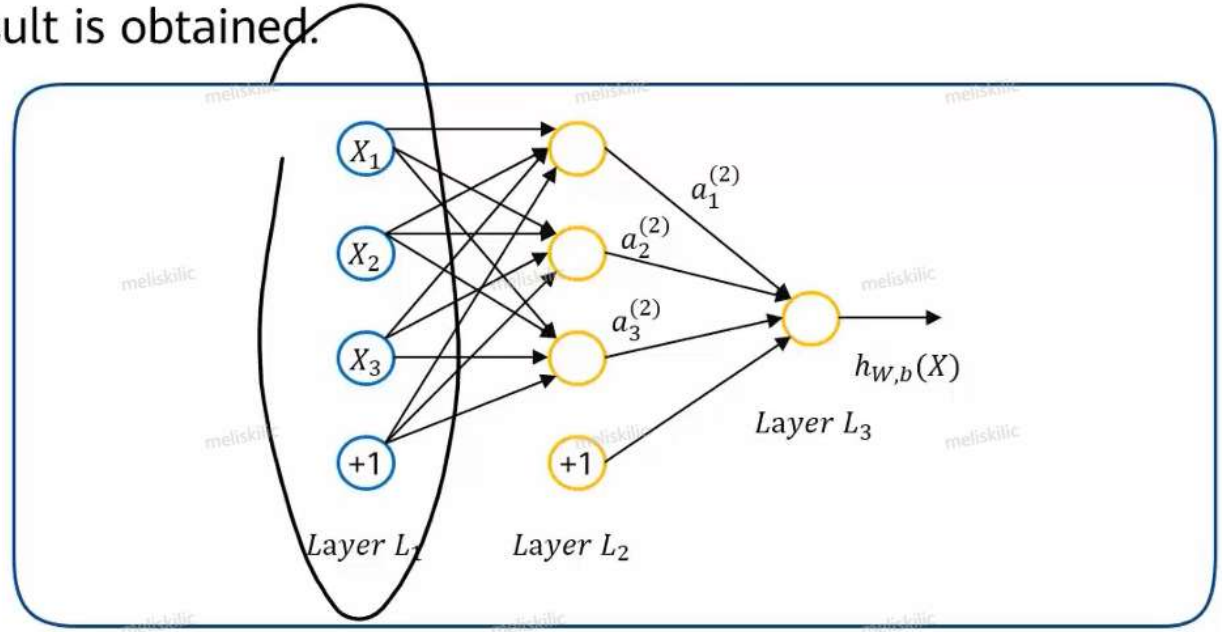
Deep Neural Network

The deep neural network (DNN) is a traditional multilayer perception (MLP) with multiple (more than two) hidden layers.



Forward Propagation Principle of the DNN

Based on the weight coefficient matrix W, offset b, and input value x, perform a series of linear operations and activation function operations by layers from the input layer to the output layer until the output result is obtained.



Backward Propagation Algorithm

The back propagation (BP) algorithm is a learning algorithm applicable to a network with multiple layers of nerve cells. It is based on the gradient descent method. The information processing capability of the BP network derives from multiple recombinations of simple and non-linear functions. Therefore, the BP network provides a strong function reproduction capability.

In the BP algorithm, two aspects (incentive propagation and weight update) iterate repeatedly until the response provided by the network reaches the predetermined target range.

- Incentive communication
- Weight update

Data Preprocessing

Before features are extracted, a series of preprocessing is performed for the original sequence to eliminate the impact on speech signal quality due to factors such as aliasing, high-order harmonic distortion, and high frequencies brought by human vocal organs and speech signal collection equipment. Try to ensure that signals obtained after subsequent speech processing are steady and smooth to provide good reference for signal parameter extraction and improve speech processing quality.

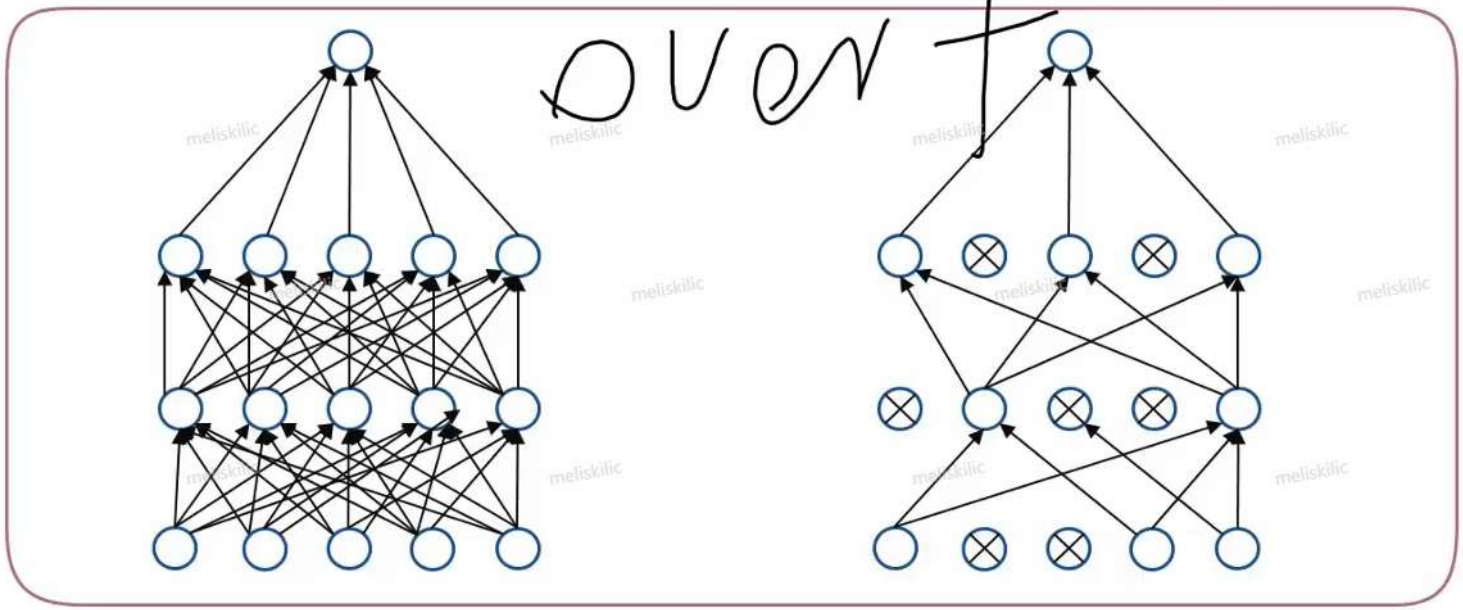
Common methods include feature normalization, standardization, endpoint detection, pre-emphasis, noise reduction, and framing.

Model Initialization

Because the DNN is a highly non-linear model and training guidelines are non-convex functions relative to parameters, model initialization has a great impact on the final effect.

Dropout

In addition to weight decay, dropout is another popular method for preventing over-fitting. Its basic idea is to randomly discard a certain proportion of nerve cells at each hidden layer during training. This means the remaining nerve cells still need to provide good performance in each random combination even if some nerve cells are discarded during training.



it has more parameters, which, however, may cause over-fitting

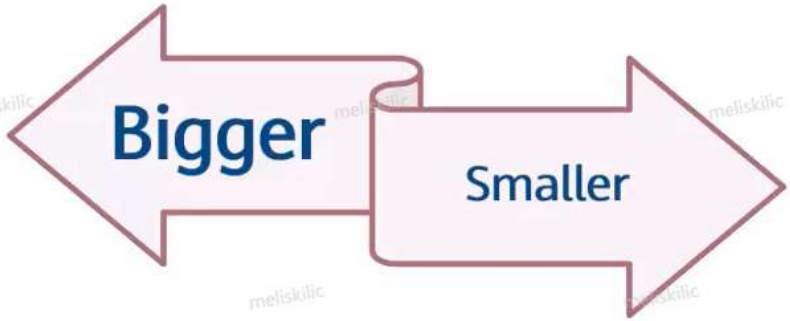
Batch Size Selection

During training, gradient calculation is required in a batch collection of training samples. The selection of the batch block size also affects the convergence rate and model training result.

Batch block selection involves the following common circumstances:

Learning Rate

From the perspective of the gradient descent algorithm, the gradient descent algorithm can obtain better performance after a proper learning rate is selected. The learning rate refers to the speed at which optimal values of parameters are obtained. When the learning rate is higher, that is, the descent is fast, parameters may bypass optimal values in a step. When the learning rate is lower, convergence cannot be completed within a long time. Therefore, the learning rate directly decides the performance of the learning algorithm.



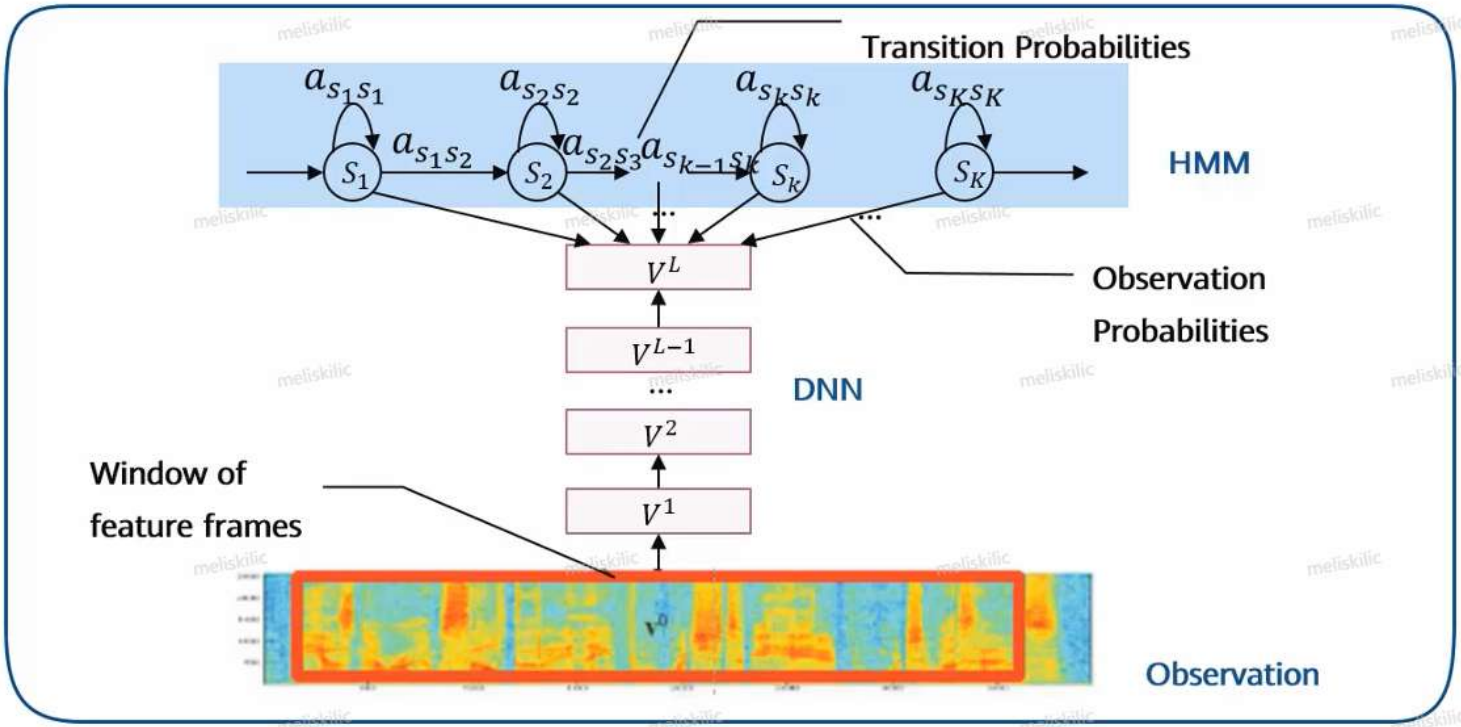
Network Structure

Each hidden layer of the DNN can be considered as the feature extractor of the previous layer and the number of nodes at each layer should be large enough to obtain the essential mode.

Wide and deep models may cause over-fitting, and narrow and shallow models may cause under-fitting. If a layer is small, the model performance degrades greatly. Relative to a narrow and shallow model, it is easier for a wide and deep model to find good configuration.

DNN - HMM Speech Recognition

Acoustic signals use the HMM framework for modeling. The generation probability of each state uses the DNN instead of the original GMM for estimate. The output of each unit on the DNN indicates the posterior probability of the state.



DNN Superior to GMM

The DNN is a discrimination model. It can better discriminate annotation categories.

The DNN provides outstanding performance with big data. As the data volume increases continuously, the performance of a GMM becomes saturated within about 2000 hours. However, the DNN model can still improve performance when the data volume increases to more than 10,000 hours.

The DNN model provides stronger robustness for ambient noise. Through noisy training, the recognition performance of the DNN model can even surpass that of a GMM using the speech enhanced algorithm.

Composition of the CD - DNN - HMM

The CD-DNN-HMM is composed of three parts

- A DNN
- An HMM
- A state prior probability distribution

Because the CD-DNN-HMM and GMM-HMM share the factor binding structure, the first step for training the CD-DNN-HMM is to use training data to train a GMM-HMM. The standard results of decoding using Viterbi are used for the DNN.

Recognition Result Integration Technology

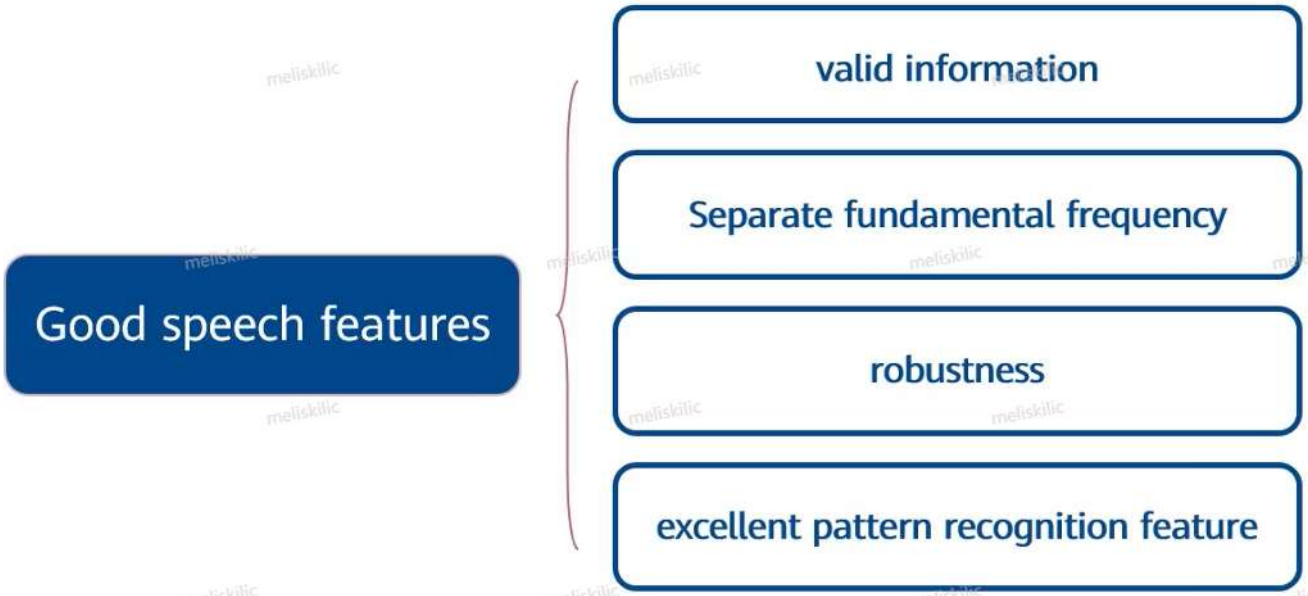
Recognition errors of a traditional GMM-HMM are often different from those of a DNN-HMM. The global performance can be improved by integrating the results of GMM-HMM and DNN-HMM.

System integration technologies that are most widely used include

- Recognizer output voting error reduction (ROVER)
- Segment conditional random field (SCARF)
- MBR-based lattice combination

Speech Features

Speech features are core information describing speech and play an important role in speech modeling.



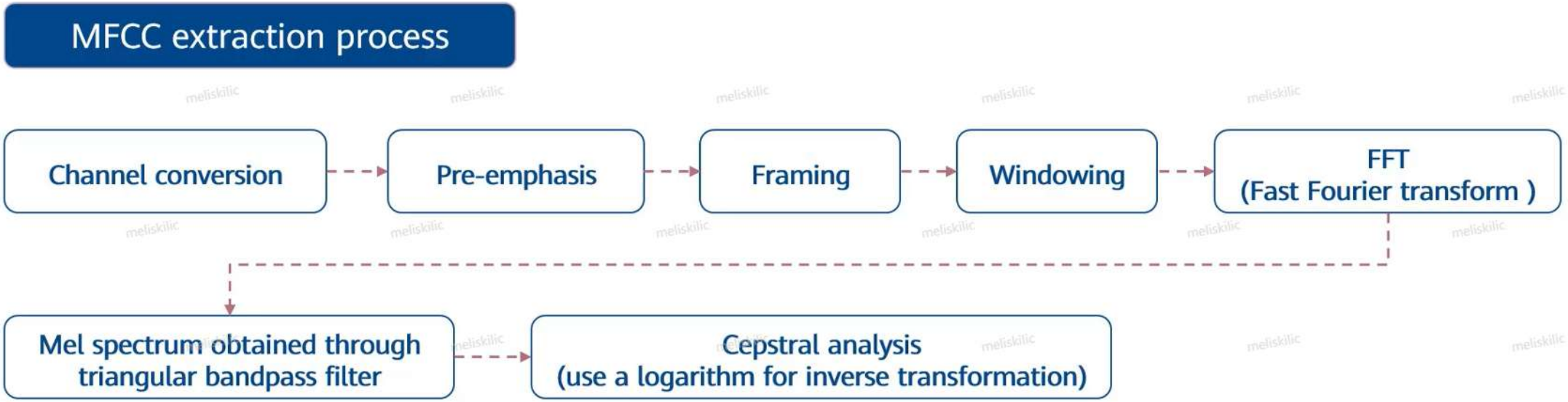
Feature Extraction Methods

Feature extraction methods

- LPC(Linear prediction coefficient)
- LPCC(LPC cepstral coefficient)
- LSP(Line spectral pair)
- first three formants
- Short-time spectrum
- MFCC(Mel frequency cepstrum coefficient)
- PLP(Perceptual linear prediction)

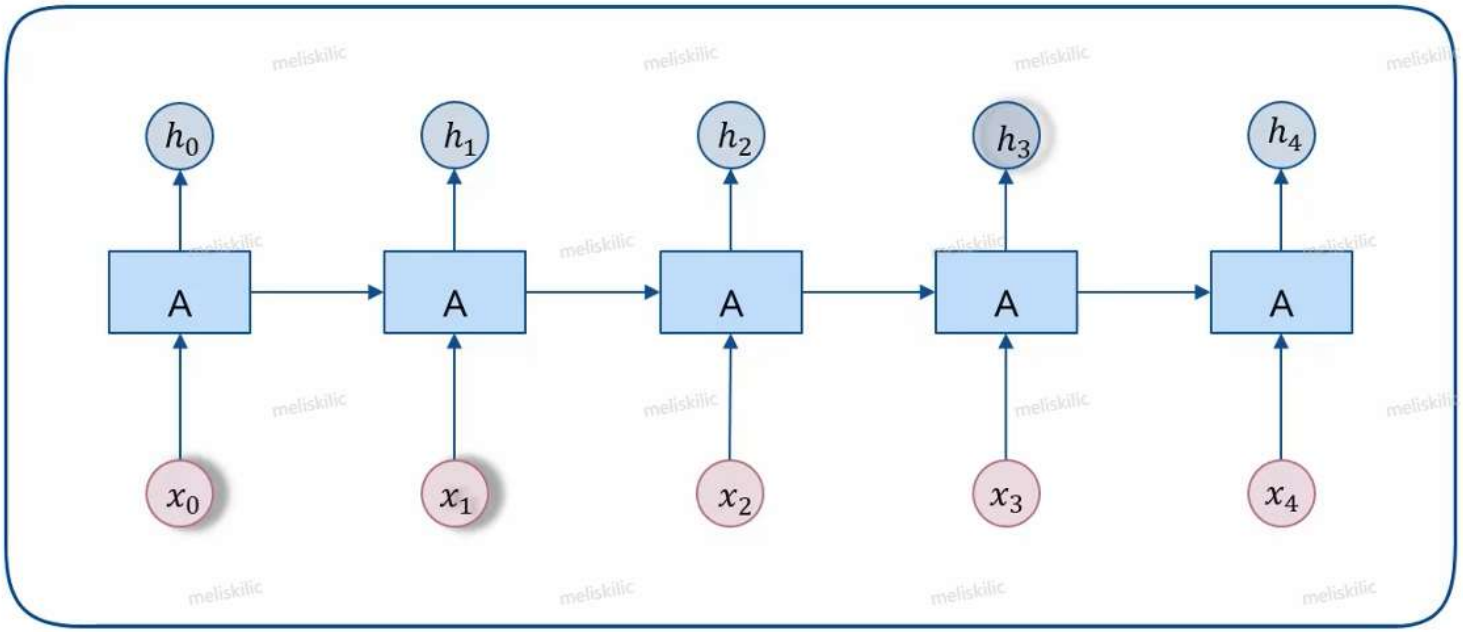
MFCC

The most common speech feature is MFCC in speech recognition and speaker recognition.

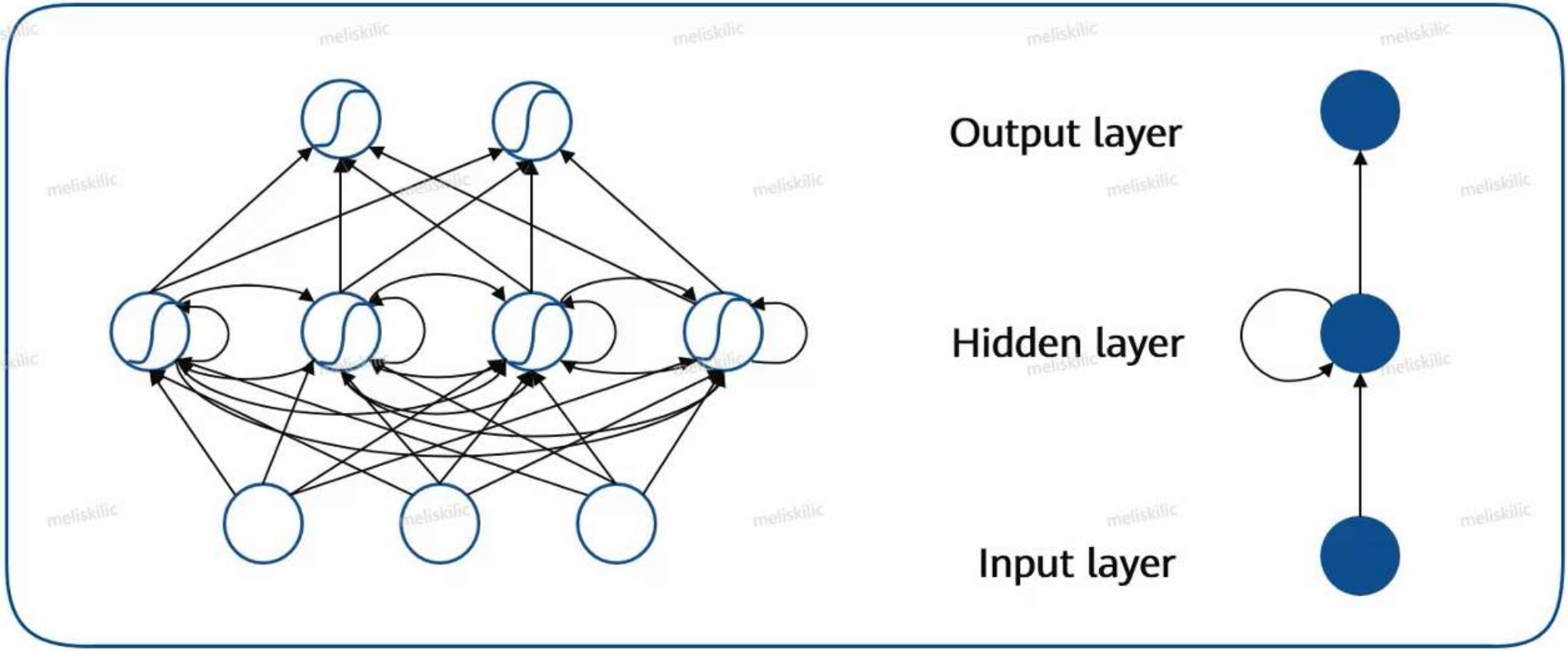


Recurrent Neural Network

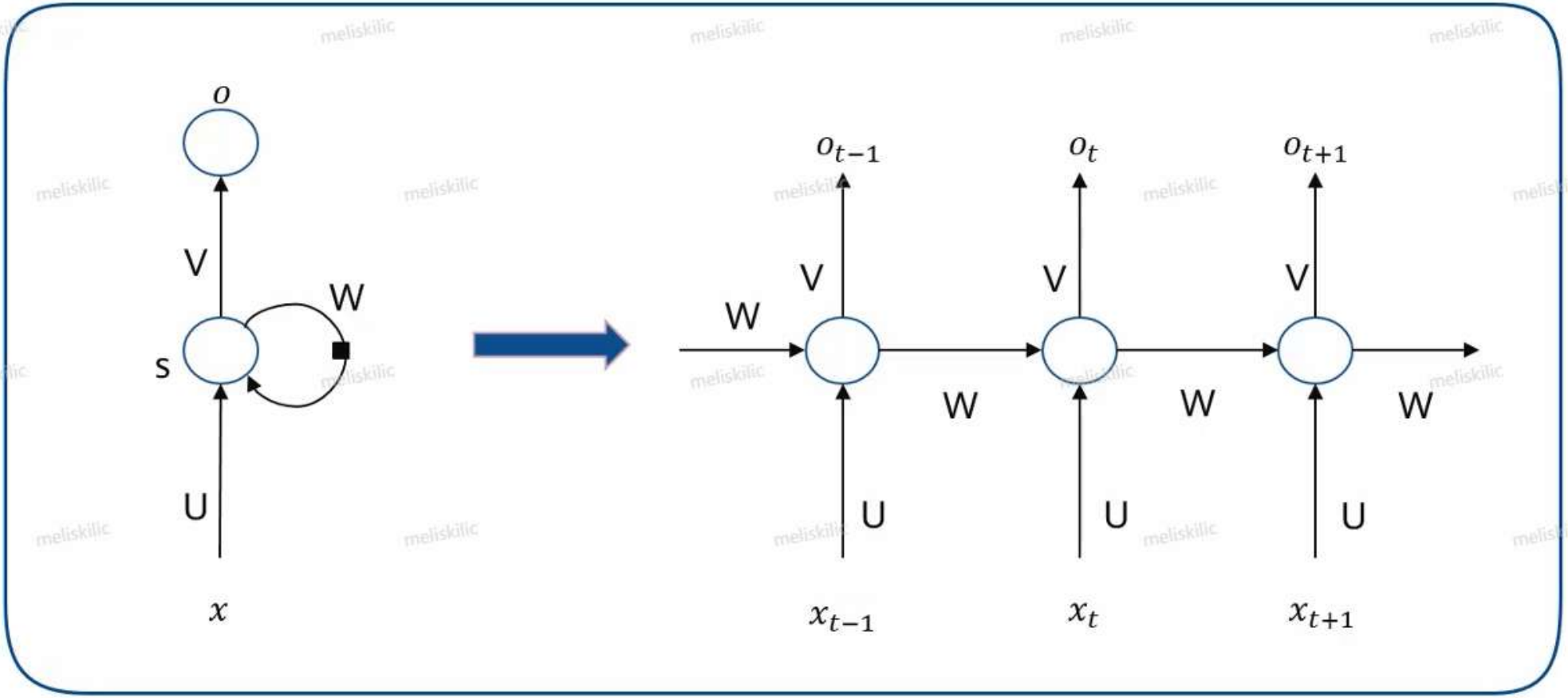
A recurrent neural network (RNN) is a neural network that captures dynamic information from serialized data through **periodic connection** of hidden layer nodes. It can classify **sequence data**.



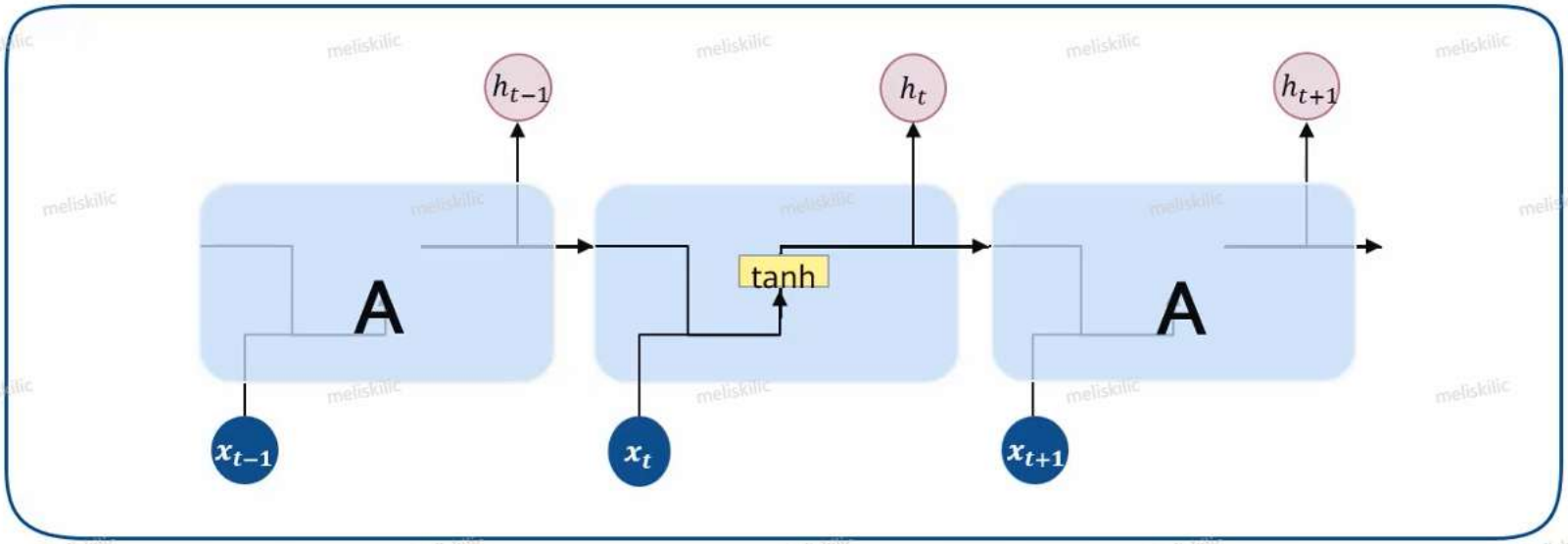
RNN Structure



Unfolding the RNN Structure



Standard RNN



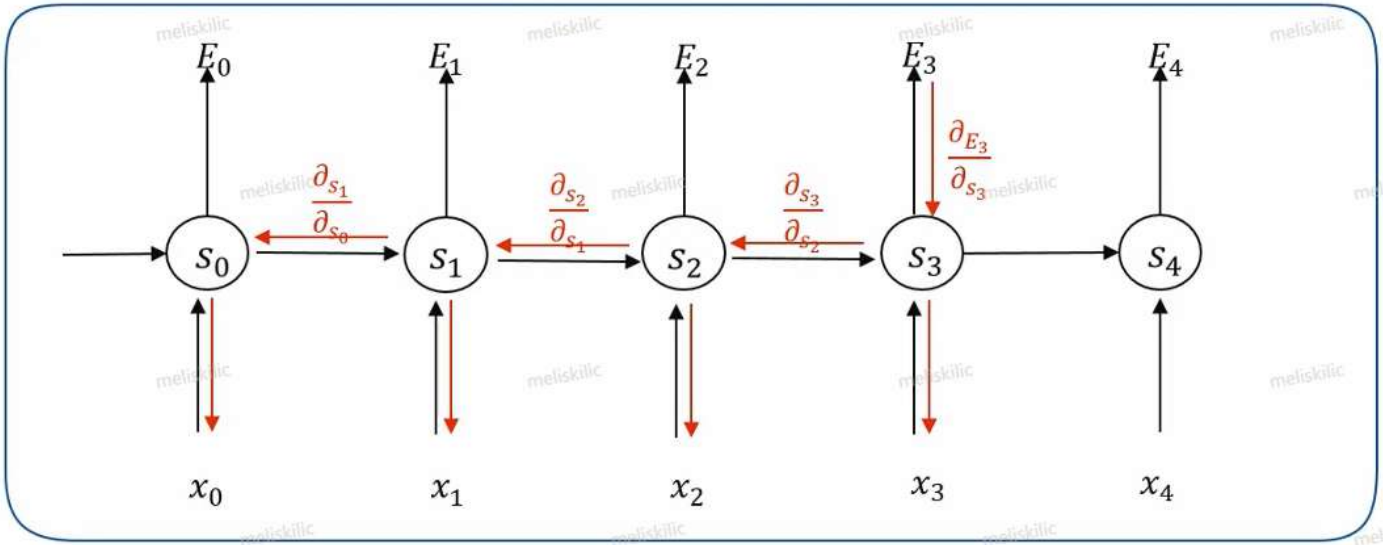
All RNNs have a chain formed by a repeated neural network module.

In a standard RNN, the repeated module has a simple structure, for example, the module has a tanh layer.

BPTT

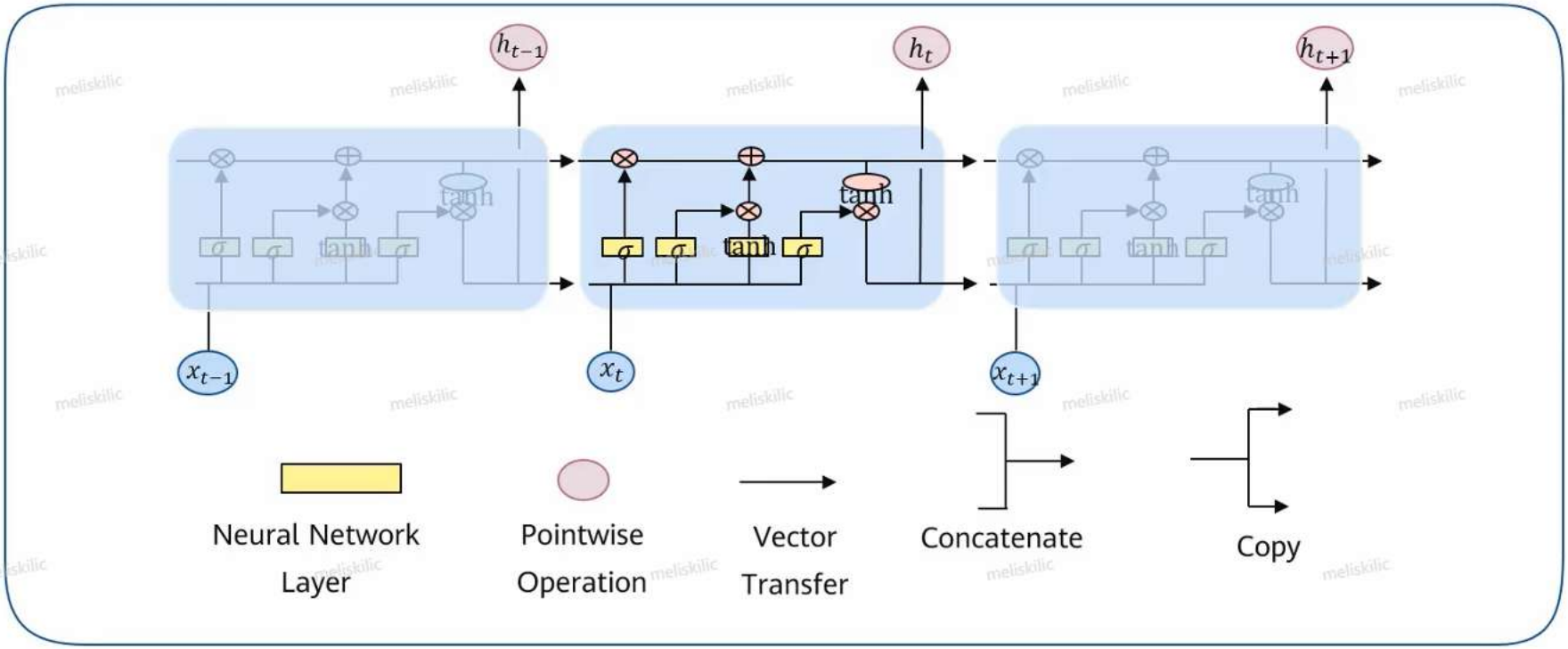
After an RNN is unfolded, forward propagation means to calculate once according to the time sequence and back propagation through time (BPTT) means to transfer accumulated residual errors back from the last time point. This is essentially similar to training of a common neural network. The main difference of BPTT is that we add the gradient at each moment.

Our goal is to calculate the error of gradients about parameters U, V, and W and learn good parameters using the gradient descent method. To calculate the gradients, we need to use the differential chain rule.

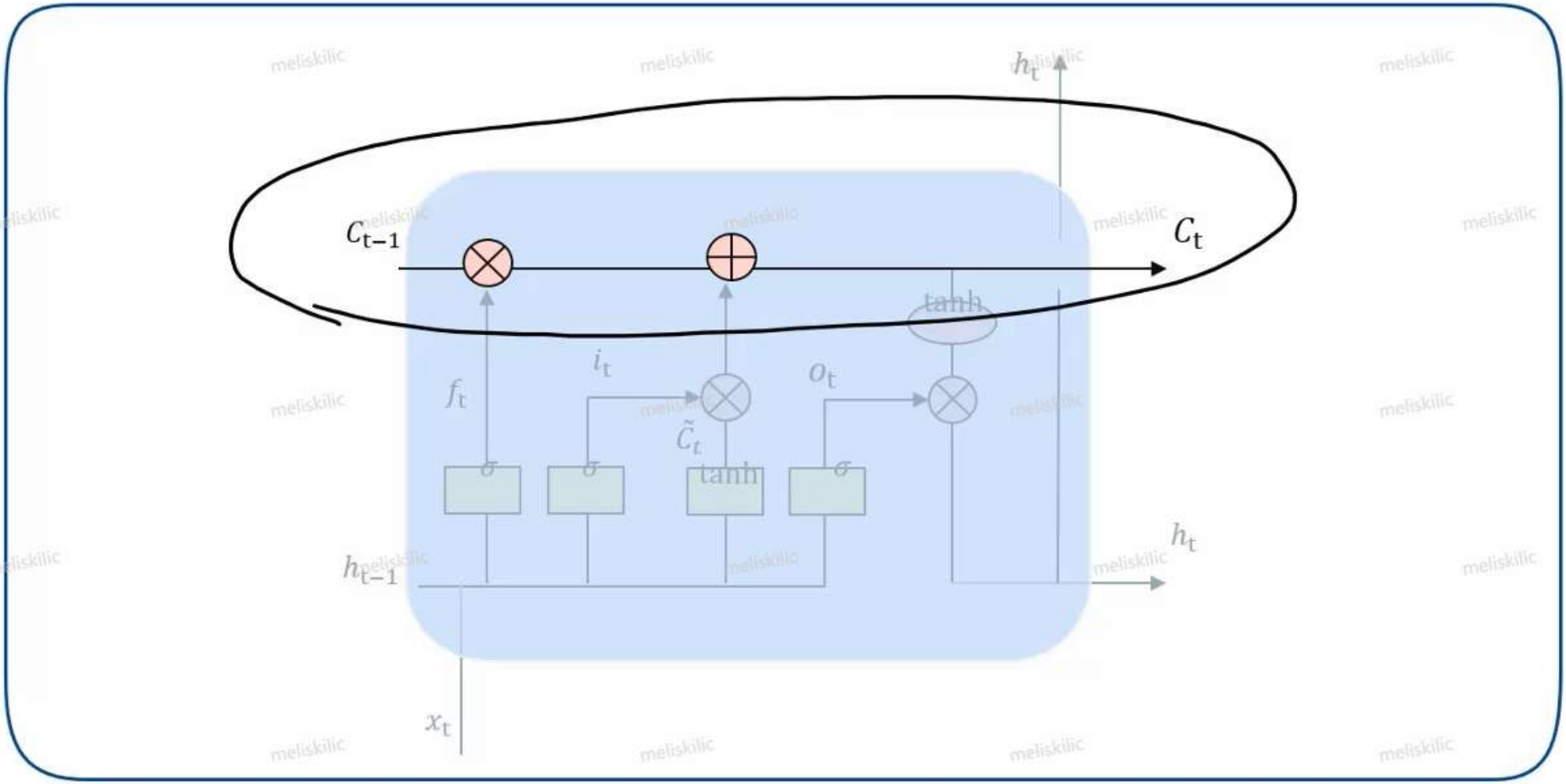


Long Short - Term Memory

A long short-term memory (LSTM) is a time recursive neural network. It is applicable to process and predict important events with long interval and delay in the time sequence.

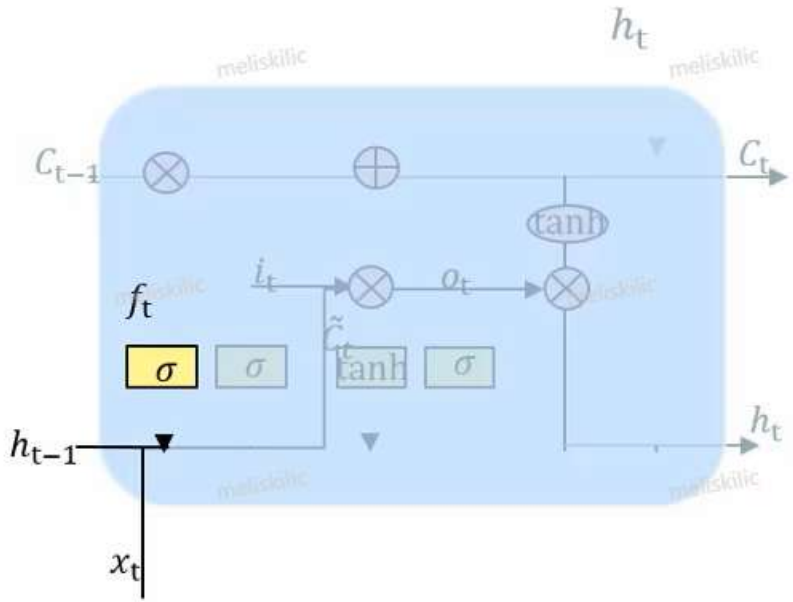


Core Knowledge Points of LSTM: Cell State



This is cell state. The cell state is similar to a conveyor

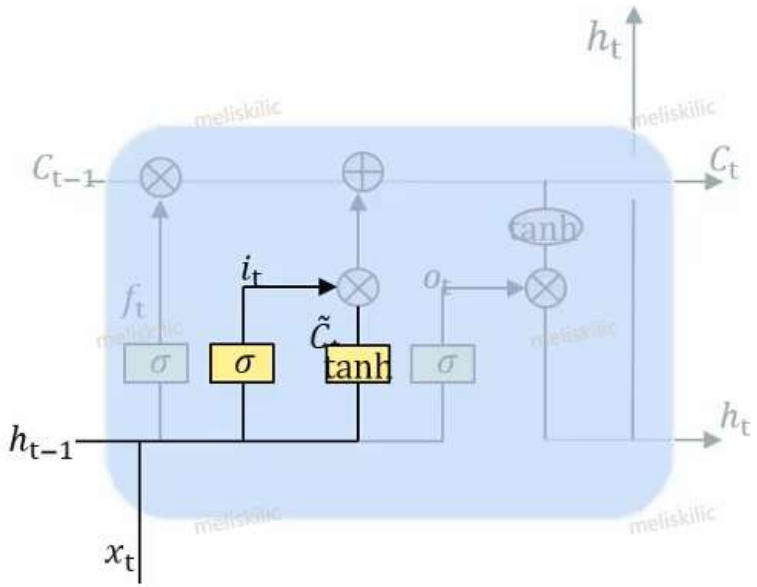
Core Knowledge Points of LSTM: Forget Gate



$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

The decision is completed through a function

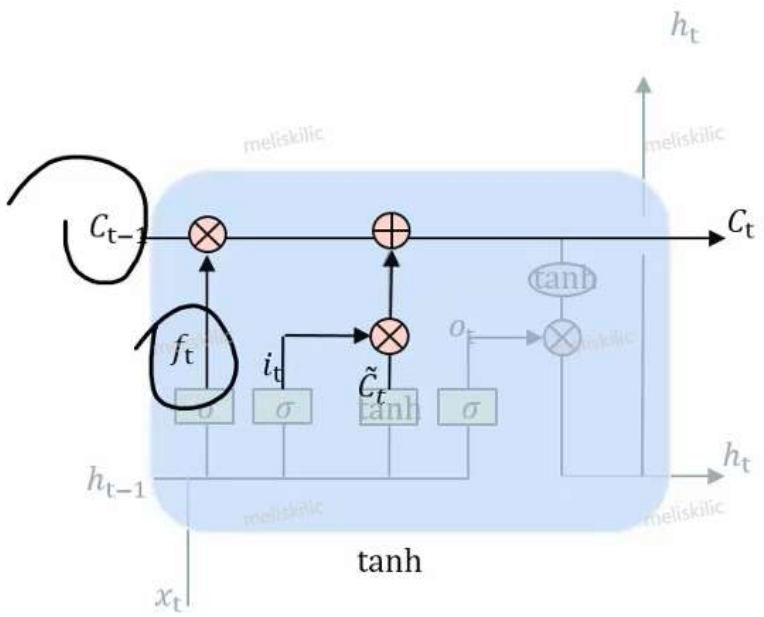
Core Knowledge Points of LSTM: Input Gate



$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

$$\tilde{C}_t = \sigma(W_C \cdot [h_{t-1}, x_t] + b_C)$$

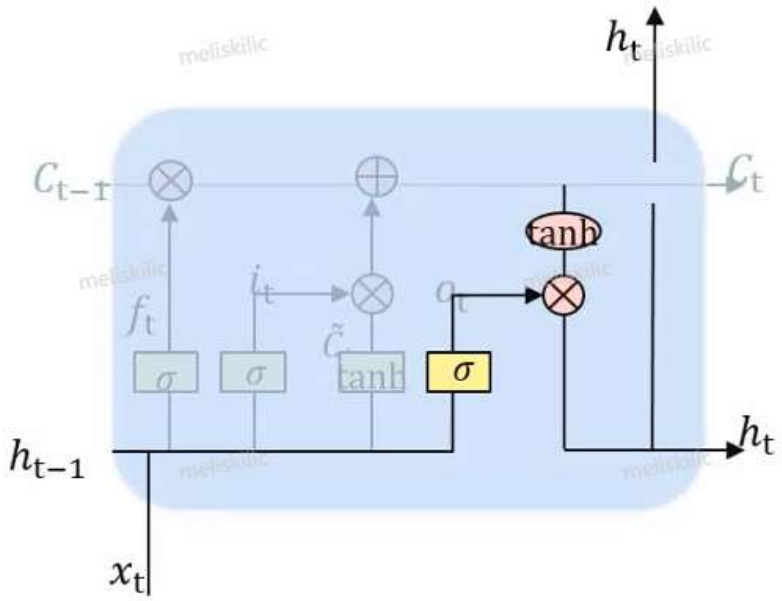
Core Knowledge Points of LSTM: Information Update



$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

In fact, here is to delete useless information

Core Knowledge Points of LSTM: Output Gate



$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o)$$

$$h_t = o_t * \tanh(C_t)$$

Application of the LSTM

At present, it has been proved that LSTM is an effective technology that solves the issue of long sequence dependence. In addition, the technology is universal, bringing many possible changes. Based on LSTM, researchers propose their variable versions. This makes LSTM able to process changing vertical issues.

LSTM currently has multiple applications in the technological field. Systems based on LSTM can learn and translate languages, control robots, analyze images, summarize documents, provide speech recognition, image recognition, and handwriting recognition, control chat robots, predict diseases, click-through rate and stocks, and synthesize music.

Technical Frontier

Neural networks

- RNN, LSTM, BiLSTM, FNN, DFSMN, LCBLSTM, and LFR-LCBLSTM

Adaptive technologies

- i-vector, AEC, etc.

Language models

- N-gram, word2vec, etc.

Difficulties in Speech Recognition

- Far-field microphone recognition
- Speech recognition in high-noise scenarios
- Multiplayer speech recognition
- Speech recognition in the background of conversation
- Non-standard speech recognition (variable speed and emotional)

Future Outlook

- Better algorithms and models
- More advanced microphone array technologies
- More advanced acoustic models and paradigms
- More powerful tools
- More in-depth characteristic methods and preprocessing methods
- Multi-disciplinary integration