

50.007 Machine Learning

Generative Models, Naïve Bayes, Gaussian Mixture Models & EM Algorithm

Berrak Sisman

Assistant Professor, ISTD Pillar, SUTD

berrak_sisman@sutd.edu.sg

Introduction & Content

- **Generative Models, Naïve Bayes, Gaussian Mixture Models (week 6)**

Instructor: Prof. Berrak Sisman

Email: berrak_sisman@sutd.edu.sg

Feel free to contact me!



Generative vs. Discriminative Models

Let's assume we want to classify the data x into label y .

- A generative model learns the joint probability distribution $p(x, y)$.
- A discriminative model learns the conditional probability distribution $p(y|x)$.

Let's assume we have the following data (x, y) : $\{(1,0), (1,0), (2,0), (2,1)\}$

$$p(x, y)$$

	$y = 0$	$y = 1$
$x = 1$	$\frac{1}{2}$	0
$x = 2$	$\frac{1}{4}$	$\frac{1}{4}$

$$p(y|x)$$

	$y = 0$	$y = 1$
$x = 1$	1	0
$x = 2$	$\frac{1}{2}$	$\frac{1}{2}$

Generative vs. Discriminative Models

A **generative model** would have the goal of understanding what dogs look like and what cats look like. You might literally ask such a model to ‘generate’, i.e. draw, a dog.

A **discriminative model**, by contrast, is only trying to learn to distinguish the classes (perhaps without learning much about them)

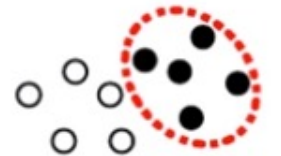


Generative vs. Discriminative Models

Both can be used for classification! The way they do classification is very different...

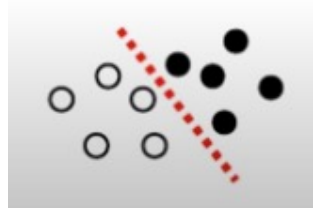
Generative models build a probabilistic model for each class (most of the time). You can see it as some characterization of entire population of one cluster.

→ To perform classification, it looks for a boundary in space where one model becomes more likely than others. They can work with both labeled and unlabeled data.



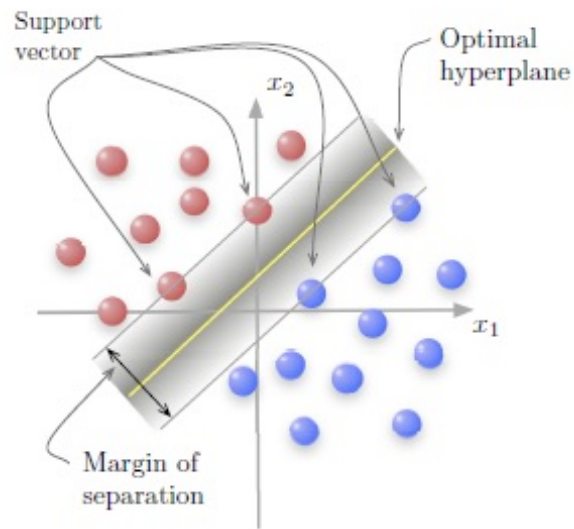
Discriminative models focus on the decision boundary. They focus on the modeling of boundary (it can be linear, or any other form).

→ They generally used with labeled data. They do not care about modeling the points out there, they care about the boundary that separates them.

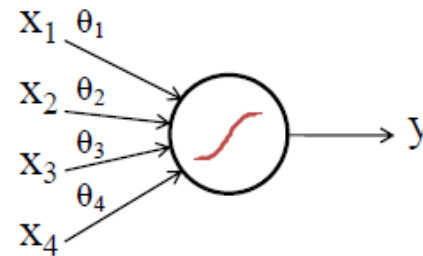


Discriminative Models

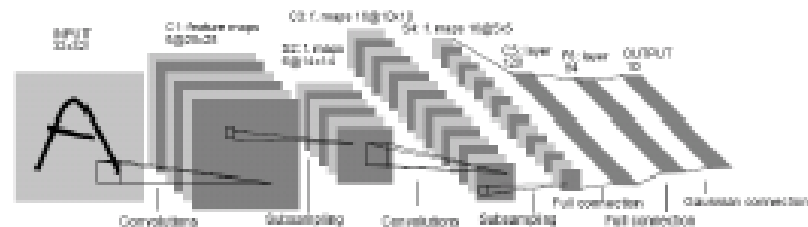
Support Vector Machines



Logistic Regression

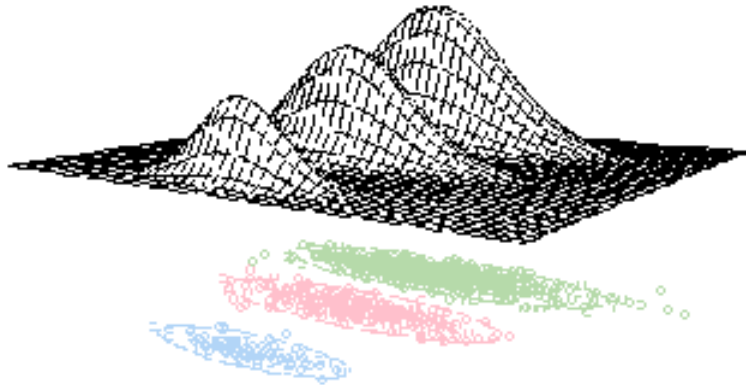


Neural Networks

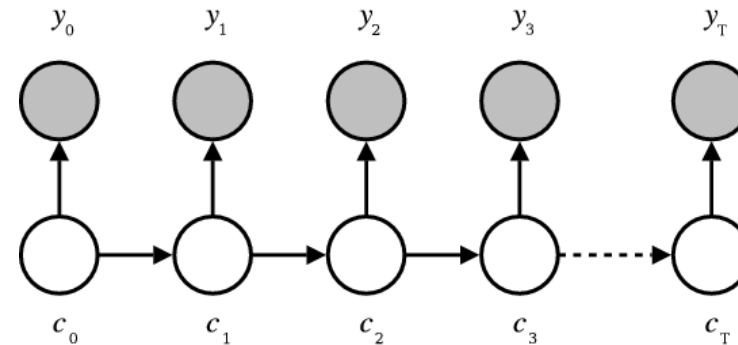


Generative Models

Gaussian Mixture Model, Naïve Bayes (*We'll study together 😊*)



Hidden Markov Model (*Prof. Roy*)



Naïve Bayes

Naïve Bayes Generative Model

- The Naive Bayes algorithm comes from a generative model, and generally used for classification.
- There is an important distinction between generative and discriminative models.
- In all cases, we want to predict the label y , given x , that is, we want

$$P(Y = y|X = x)$$

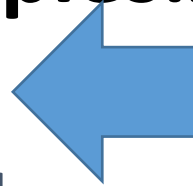
- A generative model (e.g., naive Bayes) explicitly models the **joint probability distribution** and then uses the Bayes rule to compute $P(Y = y|X = x)$.

Naïve Bayes Generative Model	Estimate $P(X = x Y = y)$ and $P(Y = y)$ and use Bayes rule to get $P(Y = y X = x)$
Discriminative Model	Directly estimate $P(Y = y X = x)$

Let's derive the mathematical formulation of Naïve Bayes through some examples...

1/ Basic formulation

2/ formulation through example (document classification, spam filtering)

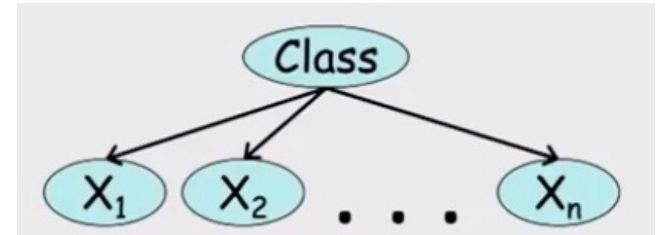


Naïve Bayes – Basic Formulation

- Here, we observe bunch of features.
- Our goal is to find each class C the particular feature belongs.
- Assumption is that every pair of features are **conditionally**

independent given the class:

- One gives no information about other once the class label is known.
- X_i and X_j are independent given the class C . We can write the joint probability as:



Naïve Bayes Model

$$P(C, X_1, X_2, \dots, X_n) = P(C) \prod_{i=1}^n P(X_i | C)$$

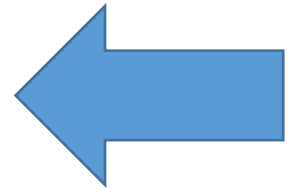
Joint distribution

We will use this joint distribution to formulate the Naïve Bayes in the later slides.

Let's derive the mathematical formulation of Naïve Bayes through an example...

1/ Basic formulation

2/ Formulation through 2 examples (document classification, spam filtering)



Example 1

Document Classification

Naïve Bayes Formulation – Document Classification

Bayes' rule:

$$P(C|D) = \frac{P(D|C)P(C)}{P(D)}$$

BIG
PICTURE

Where D is document and C is class labels.

Naïve Bayes Classifier:

THE BEST CLASS $\rightarrow C_{\text{MAP}} = \underset{C}{\operatorname{argmax}} P(C|D)$ Out of all classes, we are looking for the class whose probability is the greatest (given document)...

MAP is "maximum a posteriori" = most likely class

$$C_{\text{MAP}} = \underset{C}{\operatorname{argmax}} \frac{P(D|C)P(C)}{P(D)} \quad \text{BAYES' RULE}$$

Prior of the class

$$C_{\text{MAP}} = \underset{C}{\operatorname{argmax}} P(D|C)P(C) \quad (\text{We can drop the denominator})$$

"Probability of observing a document given class, also called likelihood". We will study easier ways to write this "conditional independence".

Example 2

Spam Filter

Constructing a spam filter - Naïve Bayes

- Each example is an email, each dimension “ j ” of vector x represents the presence of a word.
- This x represents an email containing the words “a” and “buy”, but not “aardvark” or “zyxt”. The size of the vocabulary could be ~50,000 words, so we are in a 50,000 dimensional space.
- Naive Bayes makes the assumption that the $x(j)$ ’s are **conditionally independent** given y (label).

$$\mathbf{x} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{bmatrix} \begin{matrix} \text{a} \\ \text{aardvark} \\ \text{aardwolf} \\ \vdots \\ \text{buy} \\ \vdots \\ \text{zyxt} \end{matrix}$$

What is conditionally independent?

Constructing a spam filter - Naïve Bayes

Conditionally Independent

- Naive Bayes makes the assumption that the $x(j)$'s are conditionally independent given y .
- Assume that $y = 1$ means spam email, word 2,087 is “buy”, and word 39,831 is “price.”
 - Naive Bayes assumes that if $y = 1$ (it's spam), then knowing $x(2,087) = 1$ (email contains “buy”) won't effect your belief about $x(39,831)$ (email contains “price”).
- This does not mean $x(2,087)$ and $x(39,831)$ are independent! It only means they are **conditionally independent** given y .

Constructing a spam filter - Naïve Bayes

$$P(X(1) = x(1), \dots, X(n) = x(n) | Y = y) =$$

The conditional independence assumption gives us:

$$= \prod_{j=1}^n P(X(j) = x(j) | Y = y) \quad (1)$$

Constructing a spam filter - Naïve Bayes

$$P(X(1) = x(1), \dots, X(n) = x(n) | Y = y) =$$

The conditional independence assumption gives us:

$$= \prod_{j=1}^n P(X(j) = x(j) | Y = y)$$

(1)

According to Bayes rule:

$$P(Y = y | X(1) = x(1), \dots, X(n) = x(n)) =$$

EQUAL TO

$$= \frac{P(Y = y) P(X(1) = x(1), \dots, X(n) = x(n) | Y = y)}{P(X(1) = x(1), \dots, X(n) = x(n))}$$

Constructing a spam filter - Naïve Bayes

so plugging in (1), we have

$$\begin{aligned} P(Y = y \mid X(1) = x(1), \dots, X(n) = x(n)) &= \\ &= \frac{P(Y = y) \prod_{j=1}^n P(X(j) = x(j) \mid Y = y)}{P(X(1) = x(1), \dots, X(n) = x(n))} \end{aligned}$$

For a new test instance, called x_{test} , we want to choose the most probable value of y .

Constructing a spam filter - Naïve Bayes

To choose the most probable value of y , the simplest version of Naïve Bayes can be written as:

$$\underset{y}{\operatorname{argmax}} P(Y = y) \prod_{j=1}^n P(X(j) = x(j) | Y = y)$$

**we want to choose the
most probable value of y**

Potential problem: most of the conditional probabilities are 0. Why?

If no training examples from class “spam” have the word “tomato,” we’d never classify a test example containing the word “tomato” as spam!

Home study...

Naïve Bayes is not necessarily the best algorithm, but is a good first thing to try, and performs surprisingly well given its simplicity!



Gaussian Mixture Model

Intuition...

What is a Mixture Model?

A **Mixture Model** is the weighted sum of a number of probability density functions (pdfs) where the weights are determined by a distribution, π .

$$p(x) = \pi_1 f_1(x) + \pi_2 f_2(x) + \cdots + \pi_K f_K(x)$$

where $\sum_{i=1}^K \pi_i = 1$

π_i : proportion of each class

$$p(x) = \sum_{i=1}^K \pi_i f_i(x)$$

What is Gaussian Mixture Model?

GMM: the weighted sum of a number of Gaussians where the weights are determined by a distribution, π .

$$p(x) = \pi_1 N(x|\mu_1, \Sigma_1) + \pi_2 N(x|\mu_2, \Sigma_2) + \cdots + \pi_K N(x|\mu_K, \Sigma_K)$$

where $\sum_{i=1}^K \pi_i = 1$

$$p(x) = \sum_{i=1}^K \pi_i N(x|\mu_i, \Sigma_i)$$



Carl Friedrich Gauß (1777–1855),
painted by Christian Albrecht
Jensen

Gaussian Distribution in 1D

Recall

- How to calculate mean:

$$\mu = \frac{\text{Summation of all data point}}{\text{Number of data points}}$$

- How to calculate standard deviation:

$$\sigma = \sqrt{\frac{1}{N} \sum_{i=1}^N (x_i - \mu)^2}$$

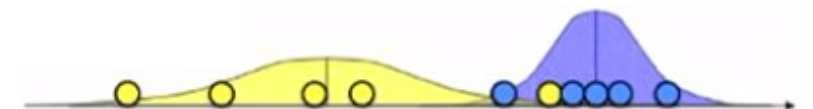
- How to write Gaussian distribution PDF:

$$N(x|\mu, \sigma^2) = \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

Variance



Carl Friedrich Gauß
(1777–1855), painted by
Christian Albrecht
Jensen



Motivation: Mixture Models in 1D

Observations $x_1, x_2, x_3, \dots, x_N$

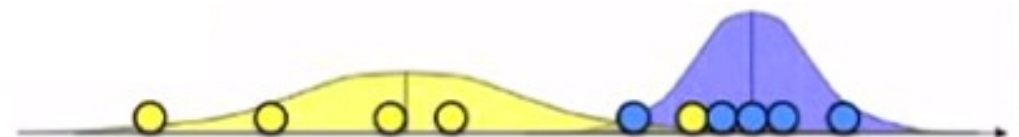
We know the sources (labels) of these observations

We also know there are 2 clusters: 2 Gaussians (with unknown mean and variance)



Can we estimate the mean and variance of these 2 Gaussian models?

- Yes, very easy. Because we know which data belong to which class, we can directly calculate the mean and variance 😊

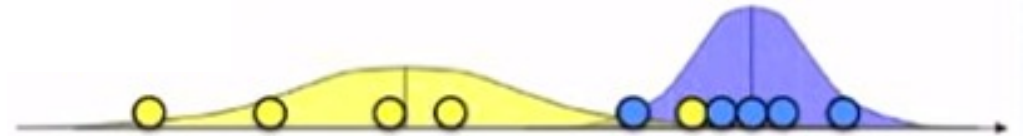


Motivation: Mixture Models in 1D

- What if we don't know the source (label) of the points? We don't know which point comes from blue or yellow. Can we easily fit the Gaussians?

K different Gaussians

If we know the parameters of the Gaussians... We could actually figure out which data point is likely from which Gaussian. “We need to check if a point is more likely to be from cluster blue, or cluster yellow”.



“Chicken and Egg Problem”

If somebody tells you which point is from which cluster, then you can calculate mean/variance (nobody tells!).

If you have the parameters of Gaussian distributions, you can figure out which point comes from which distribution... (but you don't have...)

EM Algorithm (intuition)

“Chicken and Egg Problem”

EM Algorithm can solve this problem.

- It starts with randomly placed Gaussians (random mean/variance) – similar to k-means algorithm
- E** - With these current parameters, it calculates: for each point, does it look like it comes from yellow or blue Gaussian?
 - It is going to assign them to yellow or blue, but unlike k-means, it is not going to do hard assignment. EM computes the probability that it goes to blue or yellow. It never sets it to 0 or 1, it doesn't assign a point, it keeps the probability → soft assignment/clustering
- M** - It is going to use these probabilities (that we find in E step) to re-estimate the mean and variance of Gaussian distributions.
- KEEP ITERATE UNTIL CONVERGES

GMM

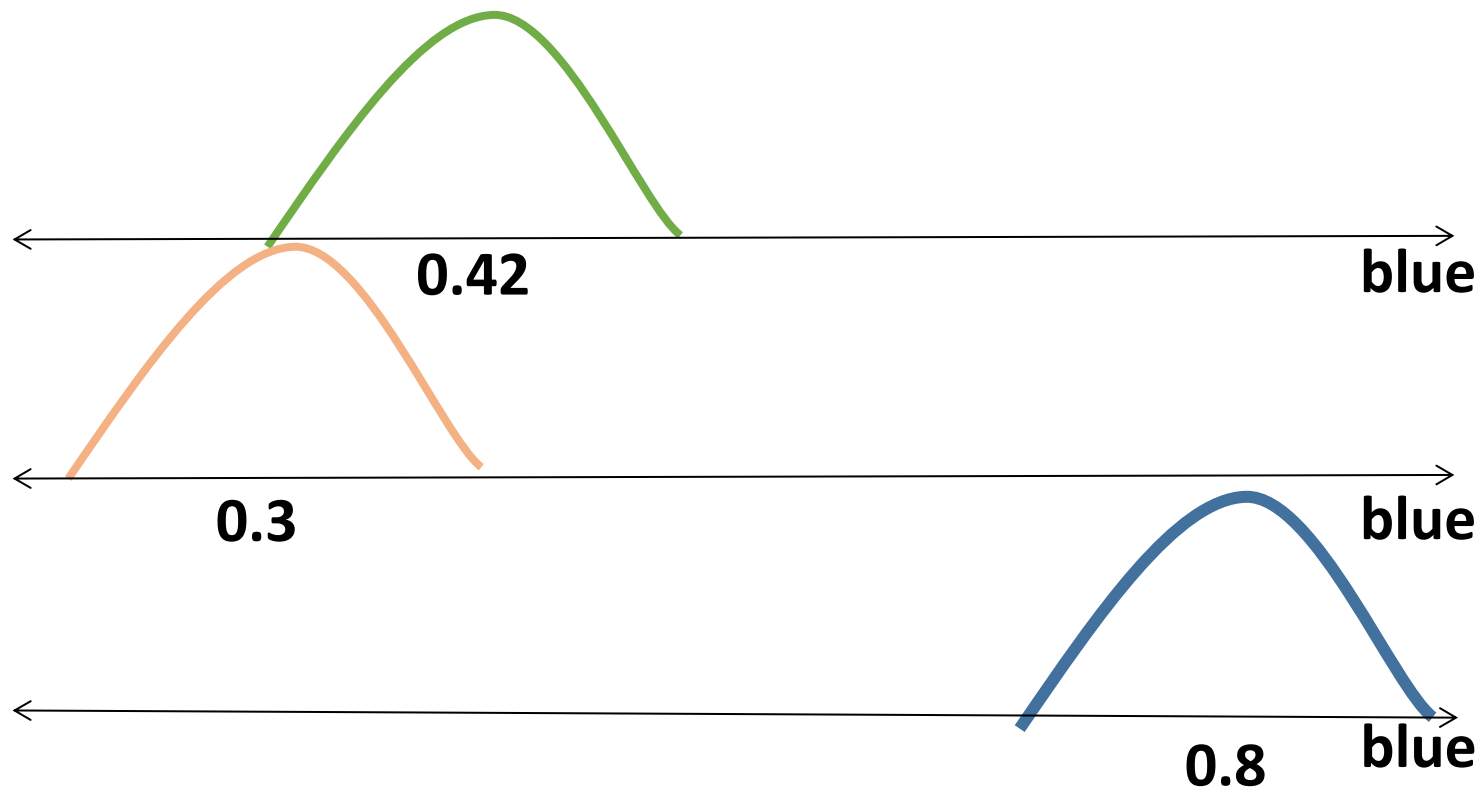
Computer Vision Perspective

Computer Vision - GMM

Model as Gaussian per category

Now that we provide a background with Gaussian distribution, let's look at the following example. We use Gaussian distribution to model the classes of different images.

- Assumption: we have the labels



forest



sunset



clouds

Computer Vision - GMM

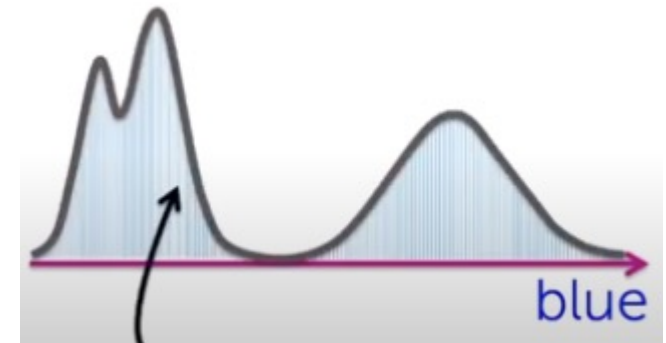
Real life: unlabeled data

In real life, what we have:



Photos from forest, sunset and clouds with NO labels...
We cannot separate or analyze each classes separately.
So how can we model and/or classify this data?

Our histogram may look like this:



How do we model this distribution?

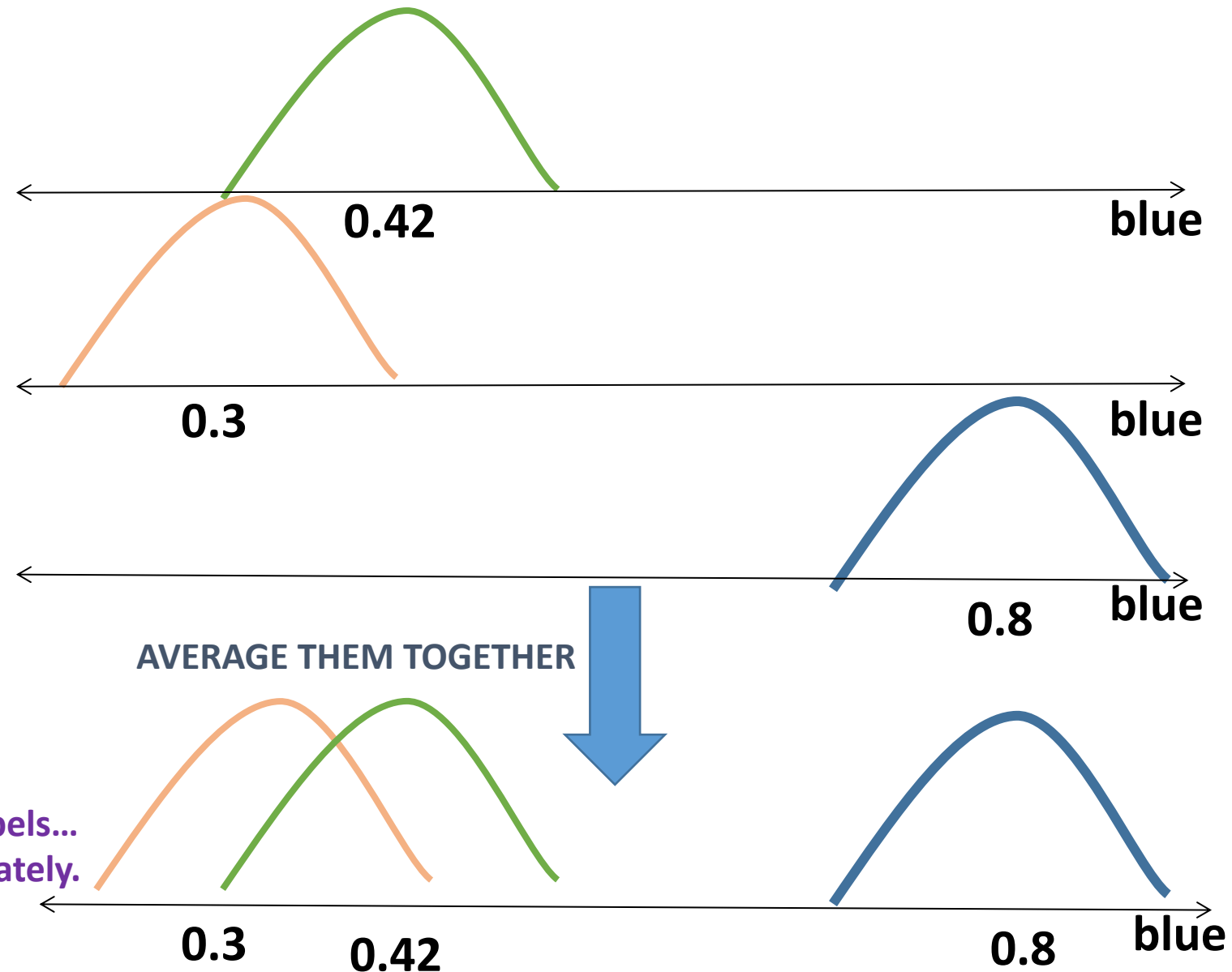
Computer Vision - GMM

Real life: unlabeled data

In real life, what we have:

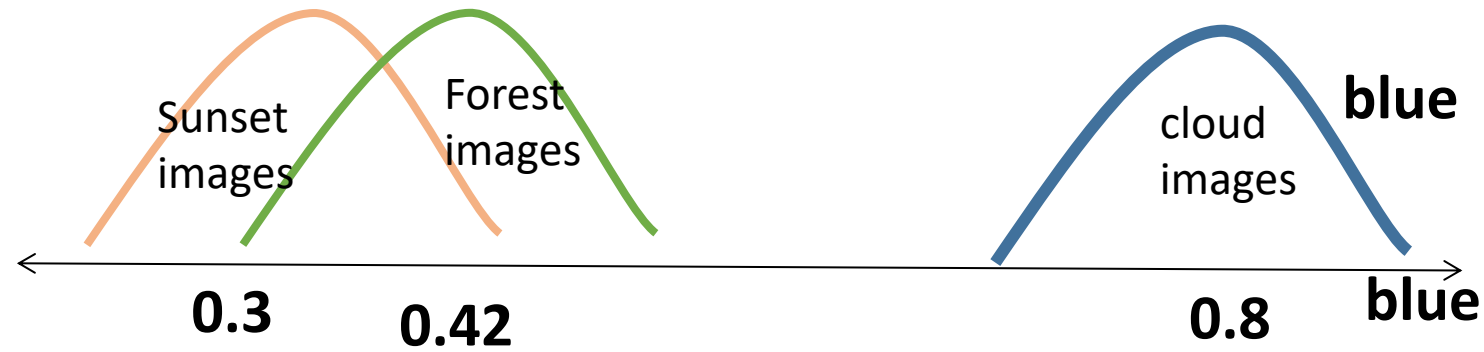


Photos from forest, sunset and clouds with NO labels...
We cannot separate or analyze each classes separately.
So how can we model and/or classify this data?

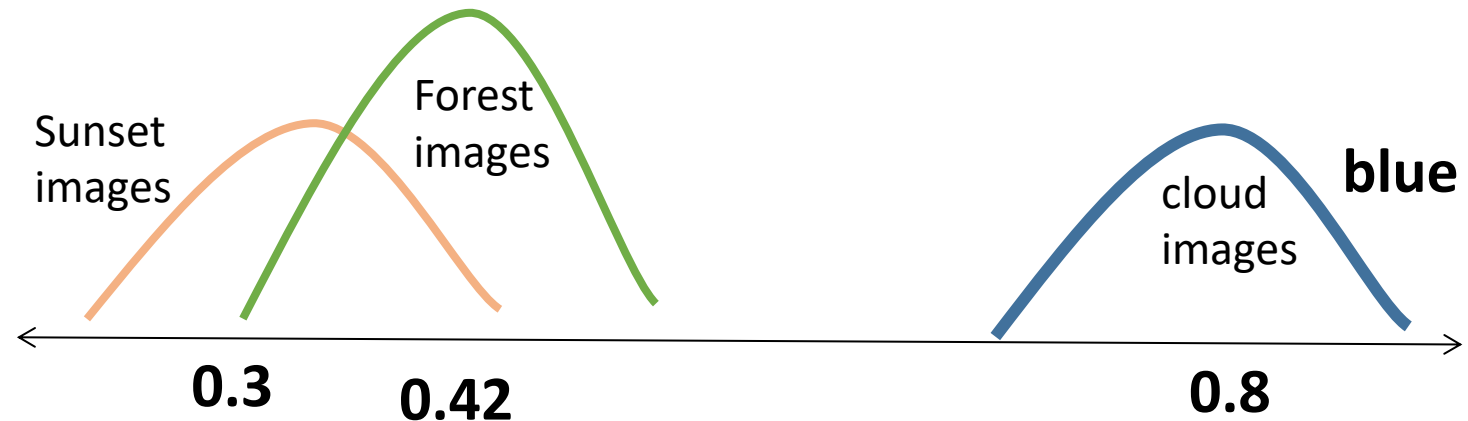


Computer Vision - GMM

This simple averaging assumes that each categories occurs in same proportion.



But what if we have many more forest images than sunset images or cloud images?



→ we'll do a weighted average, where forest distribution gets bigger weight.

We'll associated a weight π_i to each Gaussian (each cluster)

$$0 < \pi_i < 1$$

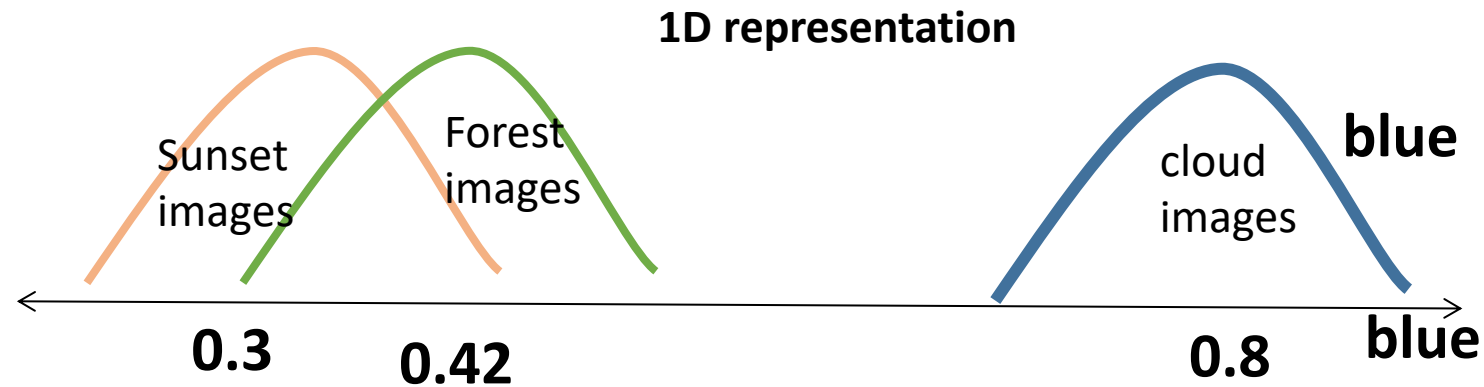
$$\sum_{i=1}^K \pi_i = 1$$

1D vs 2D

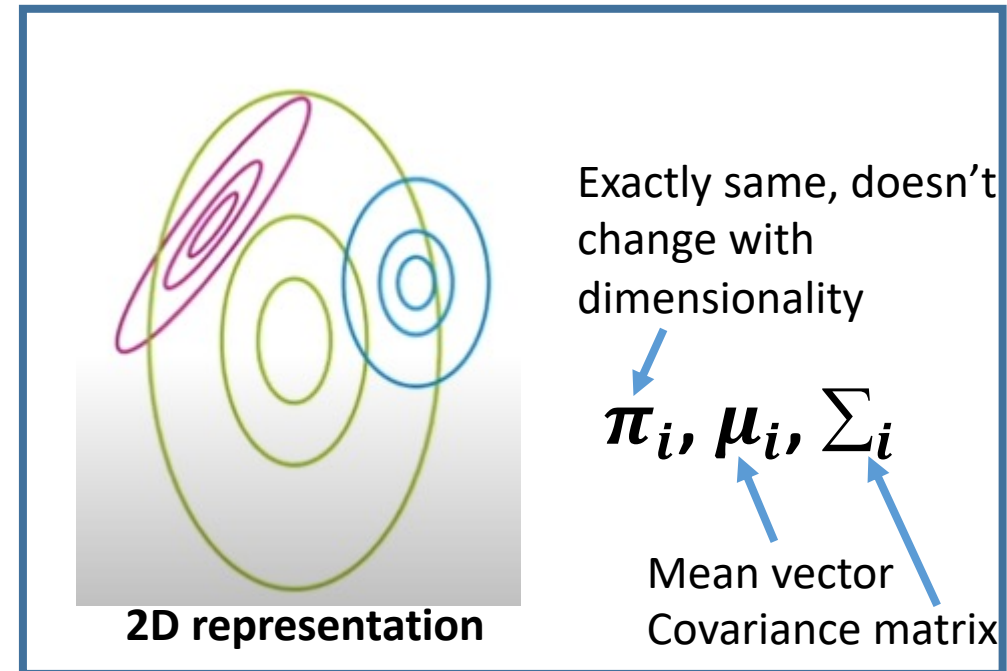
1D representation

For each cluster, we have:

- cluster specific mixture weight π_i ,
- mean μ_i (specifies location)
- variance σ_i^2 (specifies spread).



We'll associated a weight π_i to each Gaussian (each cluster)



Generative Model: Gaussian Mixture Model

Mathematical Representation...

What is Gaussian Mixture Model?

The weighted sum of a number of Gaussians where the weights are determined by a distribution, π .

$$p(x) = \pi_1 N(x|\mu_1, \Sigma_1) + \pi_2 N(x|\mu_2, \Sigma_2) + \cdots + \pi_K N(x|\mu_K, \Sigma_K)$$

where $\sum_{i=1}^K \pi_i = 1$

$$p(x) = \sum_{i=1}^K \pi_i N(x|\mu_i, \Sigma_i)$$



Gaussian Mixture Models

We will fit a set of K Gaussians to the **unlabeled** data.

Parameters



We don't know the
labels & Gaussian
parameters...

We'll find the **Maximum Likelihood estimate (MLE)** of parameters over a mixture model.

→ HOW? Expectation Maximization algorithm...

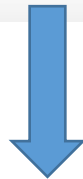
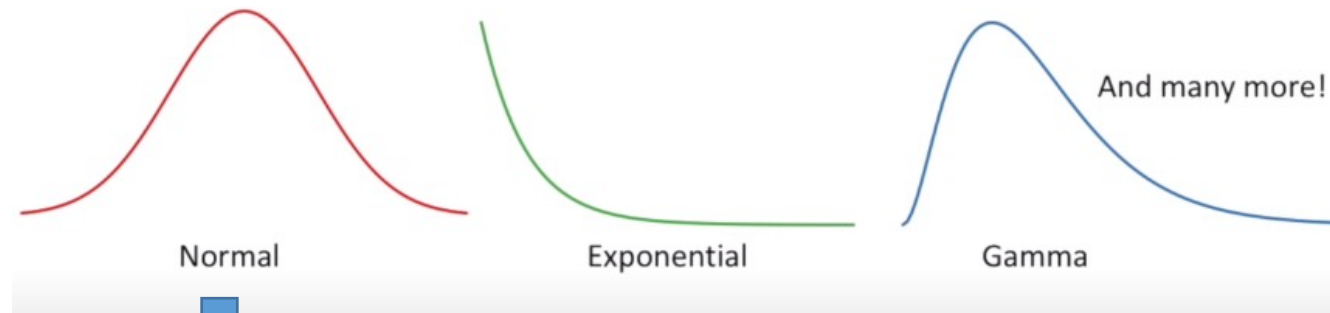
Do you remember MLE?



What is Maximum Likelihood? (recall)

Aim: To find an optimal way to fit a distribution to the data.

There are lots of distributions for different types of data



We expect most measurements to be close to the mean

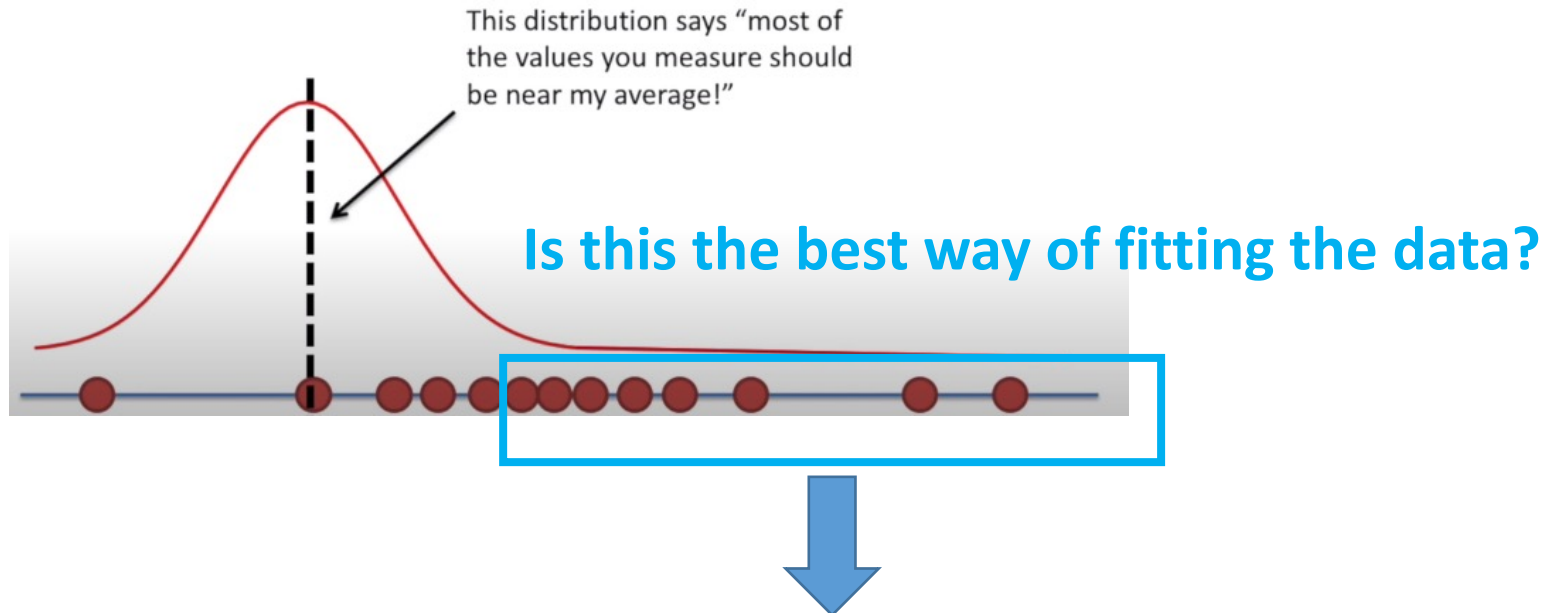
We expect most measurements to be relatively symmetrical around the mean

They can be skinny, medium or large-boned...



What is Maximum Likelihood? (recall)

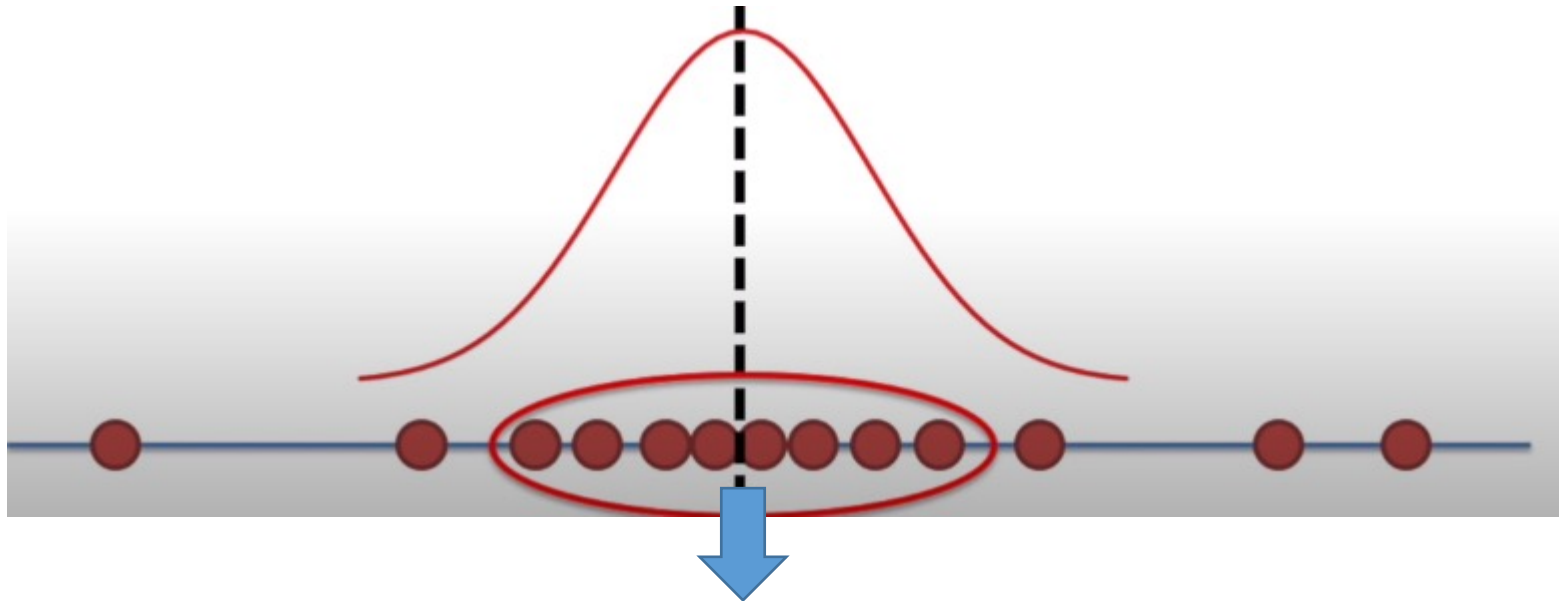
Let's pick a distribution, and see how it fits the data..



Most of the values are far from the mean of this distribution...
According to a normal distribution with this mean value, the **likelihood of observing these points is low.**

What is Maximum Likelihood? (recall)

- Let's shift the mean value...



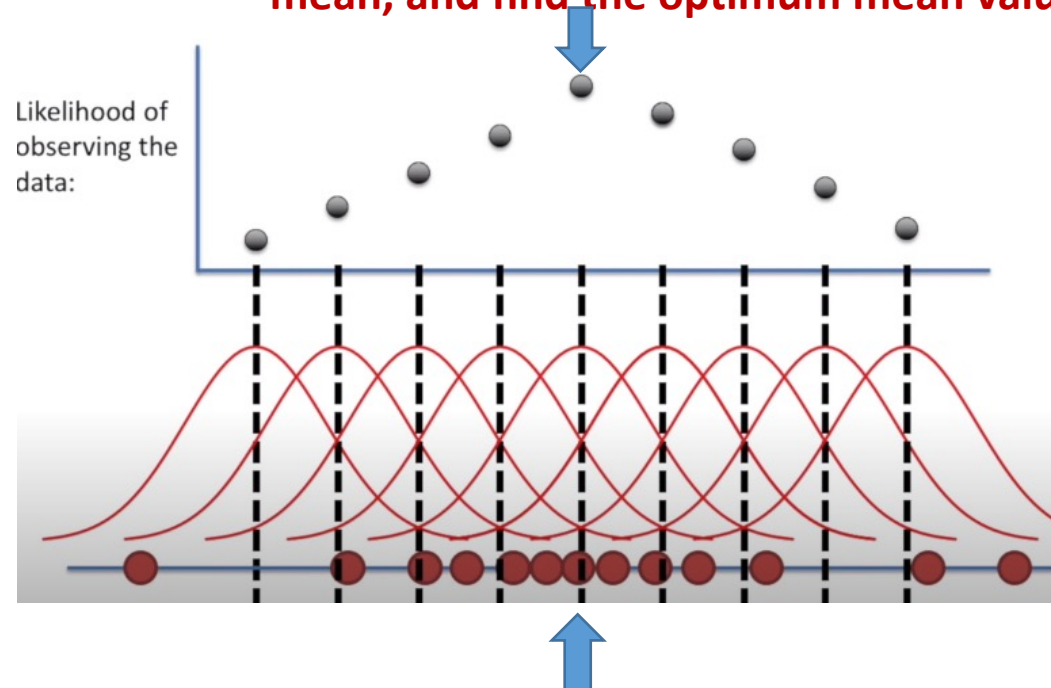
The likelihood of observing these points is relatively high now!



What is Maximum Likelihood? (recall)

- We can plot the likelihood of observing the data over the location of the mean of the distribution.

The location that maximizes the likelihood of observing the data! If we have the likelihood function, we can take the derivative with respect to mean, and find the optimum mean value.



This location for the mean maximizes the likelihood.

**IT IS THE MAXIMUM
LIKELIHOOD FOR THE MEAN**

What is Maximum Likelihood? (recall)

"I found the maximum likelihood estimate of mean & standard deviation!"



means

"I found the value of mean and standard deviation that maximizes the likelihood of observing our data."



How to estimate the GMM parameters?
Let's go back to EM Algorithm...



The EM Algorithm

- **Expectation maximization (EM)** is a method for finding maximum likelihood estimate of parameter(s) in statistical model.
- EM is an iterative method which alternates between performing an **Expectation (E)** step and a **Maximization (M)** step.



The EM Algorithm

- **Expectation maximization (EM)** is a method for finding maximum likelihood estimate of parameter(s) in statistical model.
- EM is an iterative method which alternates between performing an **Expectation (E)** step and a **Maximization (M)** step.

Mathematical perspective

Notations for EM Algorithm

- We have **unlabeled** data x_1, x_2, \dots, x_N .
- There are **K** Gaussian components. The i th component is called w_i . Component w_i has an associated mean vector μ_i and covariance matrix Σ_i .
- Unknown parameters of component k : $\mu_k, \Sigma_k, p(w_k)$. Please note that each Gaussian component can have a different probability.

z_k^n : the probability of which component k the data n is sampled from.

Notations for EM Algorithm

- We have unlabeled data x_1, x_2, \dots, x_N .
- There are **K** Gaussian components. The i th component is called w_i . Component w_i has an associated mean vector μ_i and covariance matrix Σ_i .
- Unknown parameters of component k : $\mu_k, \Sigma_k, p(w_k)$. Please note that each Gaussian component can have a different probability.

M STEP

z_k^n : the probability of which component k the data n is sampled from.

E STEP

Notations for EM Algorithm

*Each Gaussian component can
have a different probability*

$$p(w_i) = \pi_i, \quad \sum_{i=1}^K \pi_i = 1$$

$$z_i = p(w_i|x) = \frac{p(x|w_i) \cdot p(w_i)}{\sum_{j=1}^K p(x|w_j) \cdot p(w_j)}$$

*From Bayes rule
Soft clustering intuition*

Remember Bayes theorem:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

$z_k^n = p(w_k|x_n)$
the probability
of which
component k
the data n is
sampled from

Interesting examples:

<https://www.mathsisfun.com/data/bayes-theorem.html>

Compute Likelihood

$$p(w_i) = \pi_i, \quad \sum_{i=1}^K \pi_i = 1$$

$$z_i = p(w_i|x) = \frac{p(x|w_i) \cdot p(w_i)}{\sum_{j=1}^K p(x|w_j) \cdot p(w_j)}$$

Identify a log-likelihood function:

$$p(x_1, \dots, x_N | \pi, \mu, \Sigma) = \prod_{n=1}^N p(x_n | \pi, \mu, \Sigma) \quad x_n \text{'s were drawn independently}$$
$$p(x_1, \dots, x_N | \pi, \mu, \Sigma) = \prod_{n=1}^N \sum_{k=1}^K p(x_n | w_k, \mu_k, \Sigma_k) p(w_k)$$

Imagine it like “trying to find the best Gaussian and its parameters to fit the data”

Compute Likelihood

$$p(w_i) = \pi_i, \quad \sum_{i=1}^K \pi_i = 1$$

$$z_i = p(w_i|x) = \frac{p(x|w_i) \cdot p(w_i)}{\sum_{j=1}^K p(x|w_j) \cdot p(w_j)}$$

Identify a log-likelihood function:

$$\ln p(x_1, \dots, x_N | \pi, \mu, \Sigma) = \sum_{n=1}^N \ln \sum_{k=1}^K p(x_n | w_k, \mu_k, \Sigma_k) p(w_k)$$

Imagine it like “trying to find the best Gaussian and its parameters to fit the data”


Take the partial derivatives of the log-likelihood!

$$\ln p(x_1, \dots, x_N | \pi, \mu, \Sigma) = \sum_{n=1}^N \ln \sum_{k=1}^K p(x_n | w_k, \mu_k, \Sigma_k) p(w_k)$$

Derivation with respect to μ_k

$$\frac{\partial \ln p(x_1, \dots, x_N | \pi, \mu, \Sigma)}{\partial \mu_k} = 0$$

Normal (Gaussian)
Distribution



Similarly, after computing the log likelihood and taking partials derivatives to 0, we have

$$\mu_k = \frac{\sum_{n=1}^N z_k^n x_n}{\sum_{n=1}^N z_k^n}$$

$z_k^n = p(w_k | x_n)$
the probability of
which component k
the data n is
sampled from

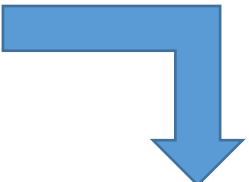
$$\Sigma_k = \frac{\sum_{n=1}^N z_k^n (x_n - \mu_k)(x_n - \mu_k)^T}{\sum_{n=1}^N z_k^n}$$

$$\pi_k = p(w_k) = \frac{\sum_{n=1}^N z_k^n}{N}$$

These are our unknown parameters, and will be used during M Step of EM Algorithm!

E-step

We will compute “expected” classes of all data points for each class...

$$z_k^n = \frac{\pi_k N(x_n | \mu_k, \Sigma_k)}{\sum_{j=1}^K \pi_j N(x_n | \mu_j, \Sigma_j)}$$


COMES FROM

where $\pi_k = p(w_k)$

$$z_i = p(w_i | x) = \frac{p(x | w_i) \cdot p(w_i)}{\sum_{j=1}^K p(x | w_j) \cdot p(w_j)}$$

z_k^n : gives us the probability of which component k the data n is sampled from

M-Step

We'll re-estimate parameters...

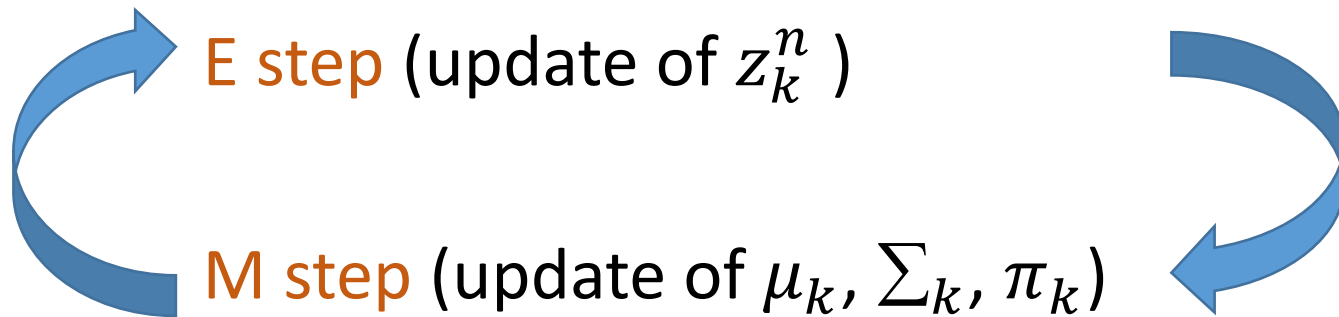
$$\mu_k^{new} = \frac{\sum_{n=1}^N z_k^n x_n}{\sum_{n=1}^N z_k^n}$$

$$\Sigma_k^{new} = \frac{\sum_{n=1}^N z_k^n (x_n - \mu_k^{new})(x_n - \mu_k^{new})^T}{\sum_{n=1}^N z_k^n}$$

$$\pi_k^{new} = p(w_k)^{new} = \frac{\sum_{n=1}^N z_k^n}{N}$$

Latent variables
become
constants here.

EM Algorithm (summary)



Unknown parameters of components: $\mu_k, \Sigma_k, p(w_k) = \pi_k$

z_k^n : gives us the probability of which component k the data n is sampled from

Summary: EM for GMMs

- Initialize the parameters
 - Evaluate the log likelihood
- Expectation step: Compute the “expected” classes of all data points
- Maximization step: Re estimate Parameters
 - Check for convergence (iterate till likelihood converges.)

EM Algorithm (intuition)

“Chicken and Egg Problem”

EM Algorithm can solve this problem.

- It starts with randomly placed Gaussians (random mean/variance) – similar to k-means algorithm
- E** - With these current parameters, it calculates: for each point, does it look like it comes from yellow or blue Gaussian?
 - It is going to assign them to yellow or blue, but unlike k-means, it is not going to do hard assignment. EM computes the probability that it goes to blue or yellow. It never sets it to 0 or 1, it doesn't assign a point, it keeps the probability → soft assignment/clustering
- M** - It is going to use these probabilities (that we find in E step) to re-estimate the mean and variance of Gaussian distributions.
- KEEP ITERATE UNTIL CONVERGES

Conclusion

- Generative models
- Naïve Bayes
- What is a mixture model?
- Gaussian Mixture Models (with MLE recall)
- EM Algorithm

MUST: Please study the lecture notes.

Suggestion: Pattern Recognition and Machine Learning. Springer, 2006”
(recommended text book).

In the next pages, you'll see some examples and applications. Please go through them at home. These could be helpful for those who are interested to learn more 😊

Naïve Bayes - Example

- Naïve Bayes is not necessarily the best algorithm, but is a good first thing to try, and performs surprisingly well given its simplicity!

Let's solve some questions together...

Real-life example

- We receive **8 normal messages** from friends & family.



- We receive **4 spam (unwanted) messages** from other people or companies.



We want to filter out the spam messages...

Real-life example

- We receive **8 normal messages** from friends & family.



- We receive **4 spam (unwanted) messages** from other people or companies.



We want to filter out the spam messages...

First step is to make a histogram of all words that occur in normal messages, and spam messages. This is to calculate the probabilities of seeing each word given the label (normal or spam).

Real-life example

- We receive **8 normal messages** from friends & family, **17 words in total**.



All the words occur in spam messages with the conditional probabilities

$$P(\text{Dear} | \text{normal}) \\ = 8/17 = 0.47$$

$$P(\text{Friend} | \text{normal}) \\ = 5/17 = 0.29$$

$$P(\text{Lunch} | \text{normal}) \\ = 3/17 = 0.18$$

$$P(\text{money} | \text{normal}) \\ = 1/17 = 0.06$$

"Dear": 8 times

"Friend": 5 times

"Lunch": 3 times

"Money:" only once

Real-life example

- We receive **8 normal messages** from friends & family, **17 words in total**.



All the words occur in spam messages with the conditional probabilities

$$P(\text{Dear} | \text{normal}) = 8/17 = 0.47$$

$$P(\text{Friend} | \text{normal}) = 5/17 = 0.29$$

$$P(\text{Lunch} | \text{normal}) = 3/17 = 0.18$$

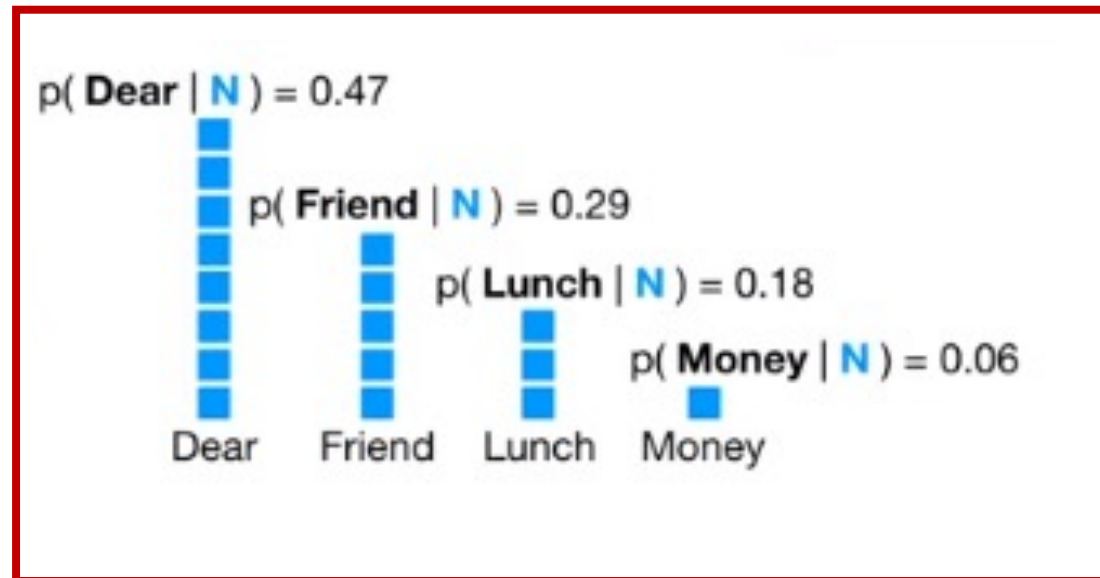
$$P(\text{money} | \text{normal}) = 1/17 = 0.06$$

"Dear": 8 times

"Friend": 5 times

"Lunch": 3 times

"Money:" only once



Histogram of all the words occurring in normal messages

Real-life example

- We receive **4 spam (unwanted) messages** from other people or companies, **7 words in total**.



All the words occur in spam messages with the conditional probabilities

$$P(\text{Dear} | \text{spam}) \\ = 2/7 = 0.29$$

$$P(\text{Friend} | \text{spam}) \\ = 1/7 = 0.14$$

$$P(\text{Lunch} | \text{spam}) \\ = 0/7 = 0$$

$$P(\text{money} | \text{spam}) \\ = 4/7 = 0.57$$

"Dear": 2 times

"Friend": only once

"Lunch": -

"Money": 4 times

Home study:
Make a histogram of all the words occurring in spam messages

Real-life example

- We receive 8 normal messages from friends & family, 17 words in total.



All the words occur in spam messages with the conditional probabilities

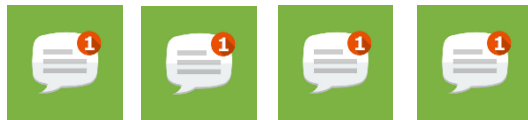
$$P(\text{Dear} | \text{normal}) \\ = 8/17 = 0.47$$

$$P(\text{Friend} | \text{normal}) \\ = 5/17 = 0.29$$

$$P(\text{Lunch} | \text{normal}) \\ = 3/17 = 0.18$$

$$P(\text{money} | \text{normal}) \\ = 1/17 = 0.06$$

- We receive 4 spam (unwanted) messages from other people or companies, 7 words in total.



All the words occur in spam messages with the conditional probabilities

$$P(\text{Dear} | \text{spam}) \\ = 2/7 = 0.29$$

$$P(\text{Friend} | \text{spam}) \\ = 1/7 = 0.14$$

$$P(\text{Lunch} | \text{spam}) \\ = 0/7 = 0$$

$$P(\text{money} | \text{spam}) \\ = 4/7 = 0.57$$

Probability of seeing the word “dear” given it is a normal message: **$P(\text{Dear} | \text{normal})$**

Real-life example

- Let's imagine we have a new message that says "*Dear friend*"

Is it a normal message or spam?

Real-life example

- Let's imagine we have a new message that says "*Dear friend*"

Is it a normal message or spam?

Let's assume it is a normal message (regardless of what it says):

$$p(\text{normal}) = 8/12 = 0.67$$

$$p(\text{normal}) * p(\text{Dear}|\text{normal}) * p(\text{Friend}|\text{normal}) = 0.67 * 0.47 * 0.29$$

= **0.09** (it is the score "*Dear friend*" gets if it is a normal message and this score is proportional to $p(\text{normal}|\text{Dear Friend})$)

Real-life example

- Let's imagine we have a new message that says "*Dear friend*"

Is it a normal message or spam?

Let's assume it is a normal message (regardless of what it says):

$$p(\text{normal}) = 8/12 = 0.67$$

$$p(\text{normal}) * p(\text{Dear}|\text{normal}) * p(\text{Friend}|\text{normal}) = 0.67 * 0.47 * 0.29$$

= **0.09** (it is the score "*Dear friend*" gets if it is a normal message and this score is proportional to $p(\text{normal}|\text{Dear Friend})$)

Let's assume it is a spam message (regardless of what it says):

$$p(\text{spam}) = 4/12 = 0.33$$

$$p(\text{spam}) * p(\text{Dear}|\text{spam}) * p(\text{Friend}|\text{spam}) = 0.33 * 0.29 * 0.14$$

= **0.01** (it is the score "*Dear friend*" gets if it is a spam message and this score is proportional to $p(\text{spam}|\text{Dear Friend})$)

Real-life example

- Let's imagine we have a new message that says "*Dear friend*"

Is it a normal message or spam?

Let's assume it is a normal message (regardless of what it says):

$$p(\text{normal}) = 8/12 = 0.67$$

$$p(\text{normal}) * p(\text{Dear}|\text{normal}) * p(\text{Friend}|\text{normal}) = 0.67 * 0.47 * 0.29$$

= **0.09** (it is the score "*Dear friend*" gets if it is a normal message and this score is proportional to $p(\text{normal}|\text{Dear Friend})$)

Let's assume it is a spam message (regardless of what it says):

$$p(\text{spam}) = 4/12 = 0.33$$

$$p(\text{spam}) * p(\text{Dear}|\text{spam}) * p(\text{Friend}|\text{spam}) = 0.33 * 0.29 * 0.14$$

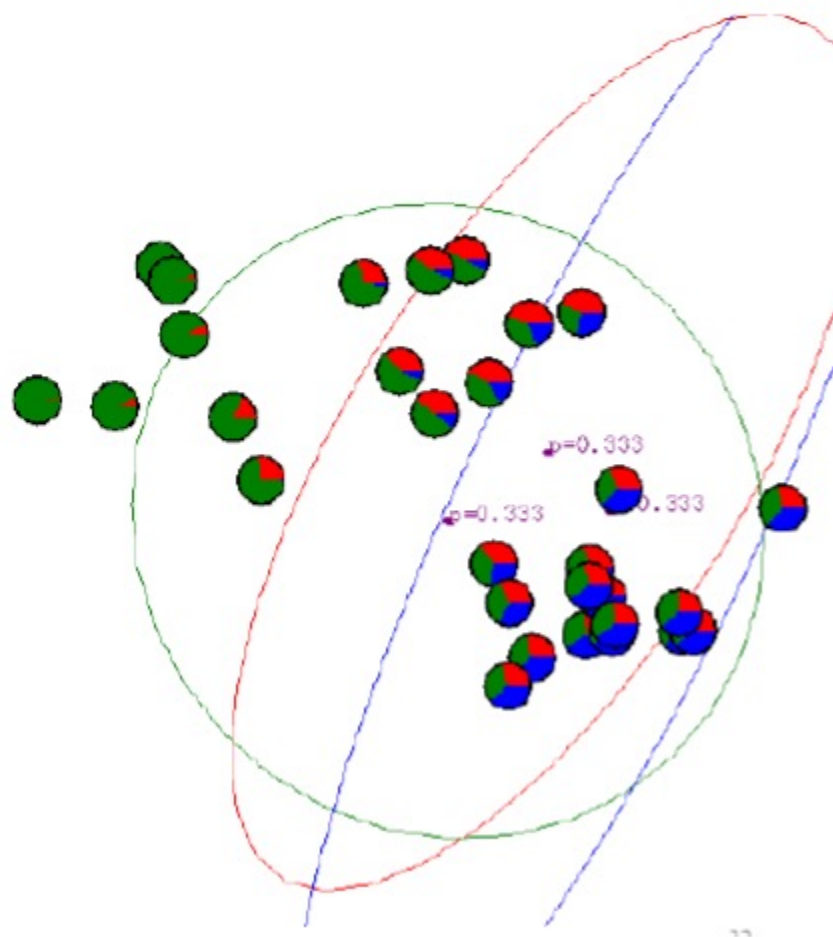
= **0.01** (it is the score "*Dear friend*" gets if it is a spam message and this score is proportional to $p(\text{spam}|\text{Dear Friend})$)

"*Dear friend*" is a normal message because $0.09 > 0.01$

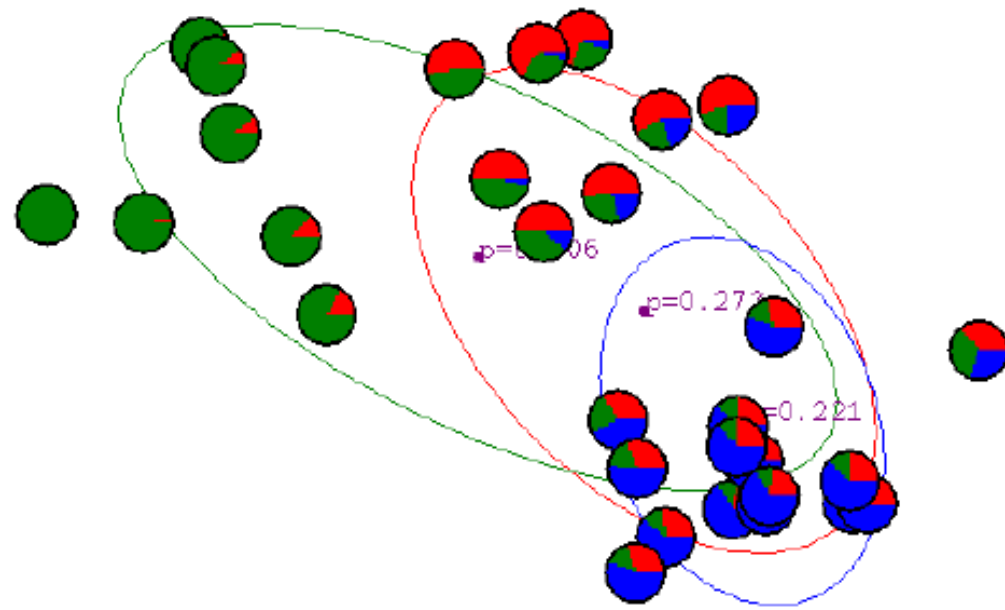


GMM Example

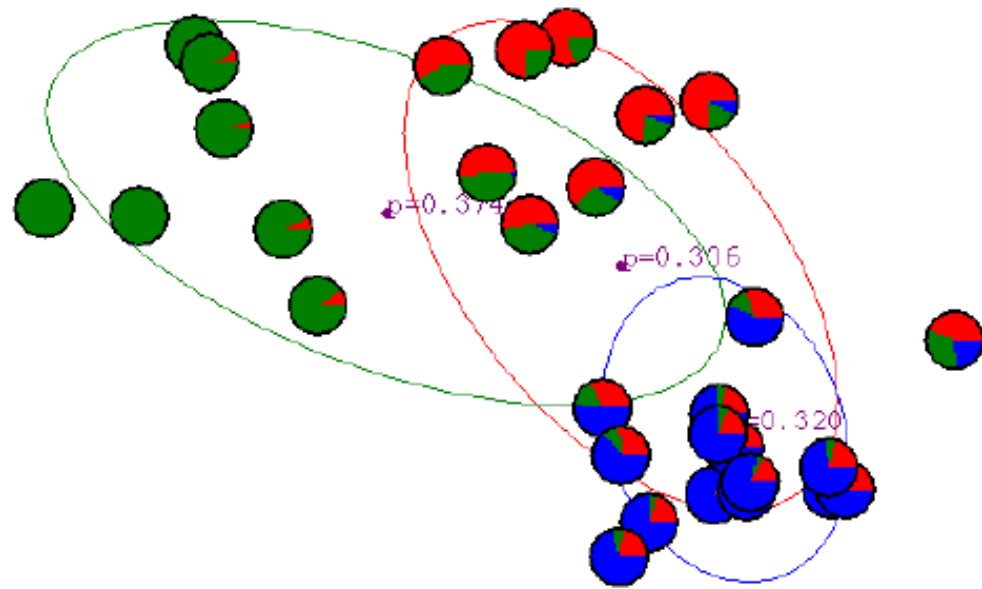
Start



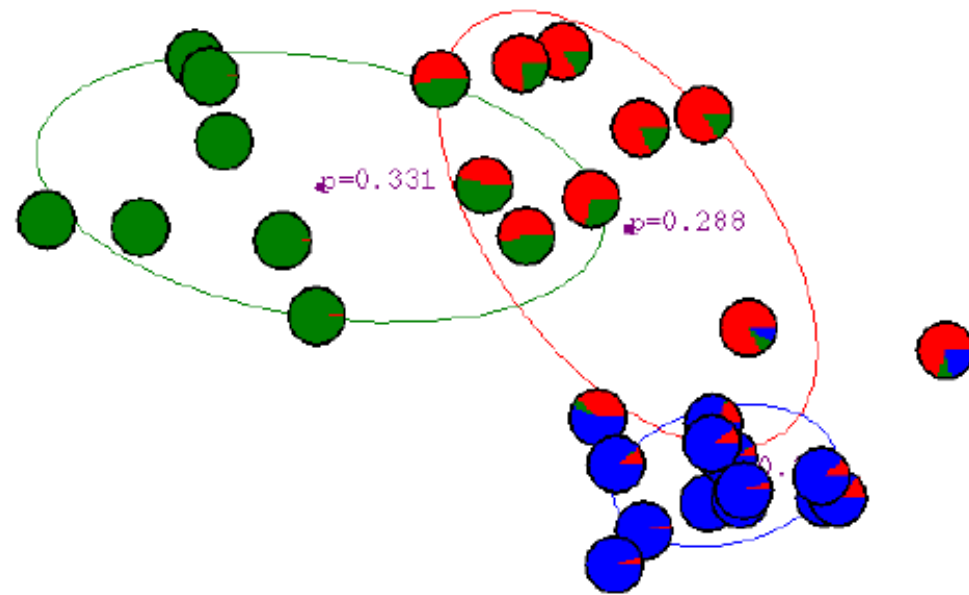
After 1st iteration



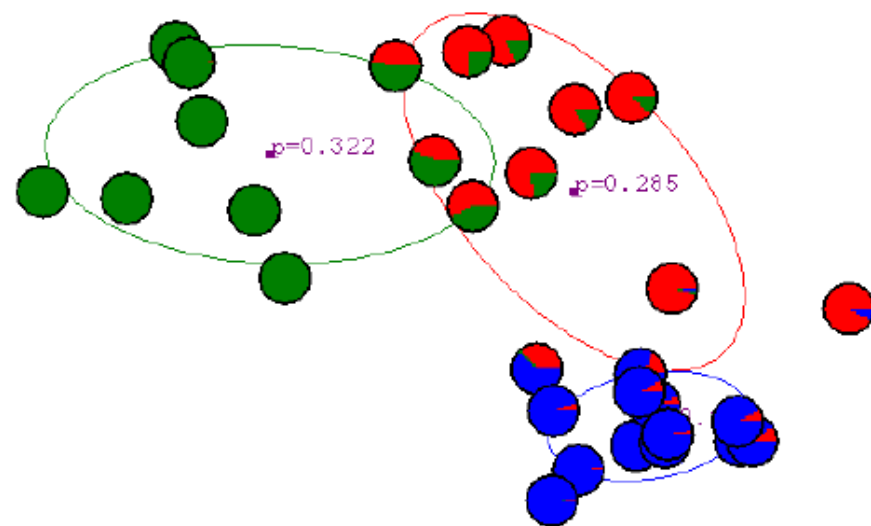
After 2nd iteration:



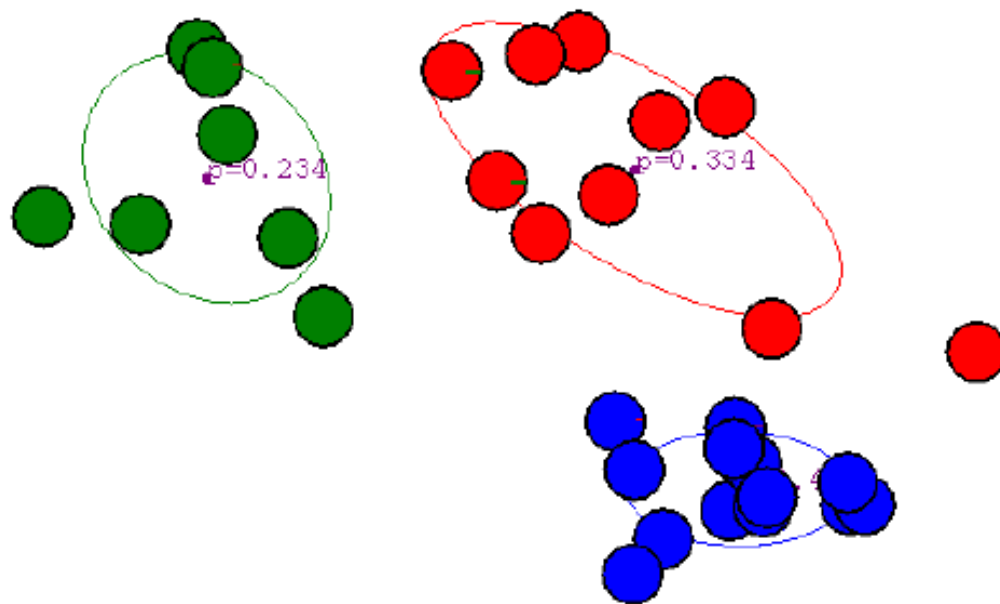
After 3rd iteration:



After 5th iteration:

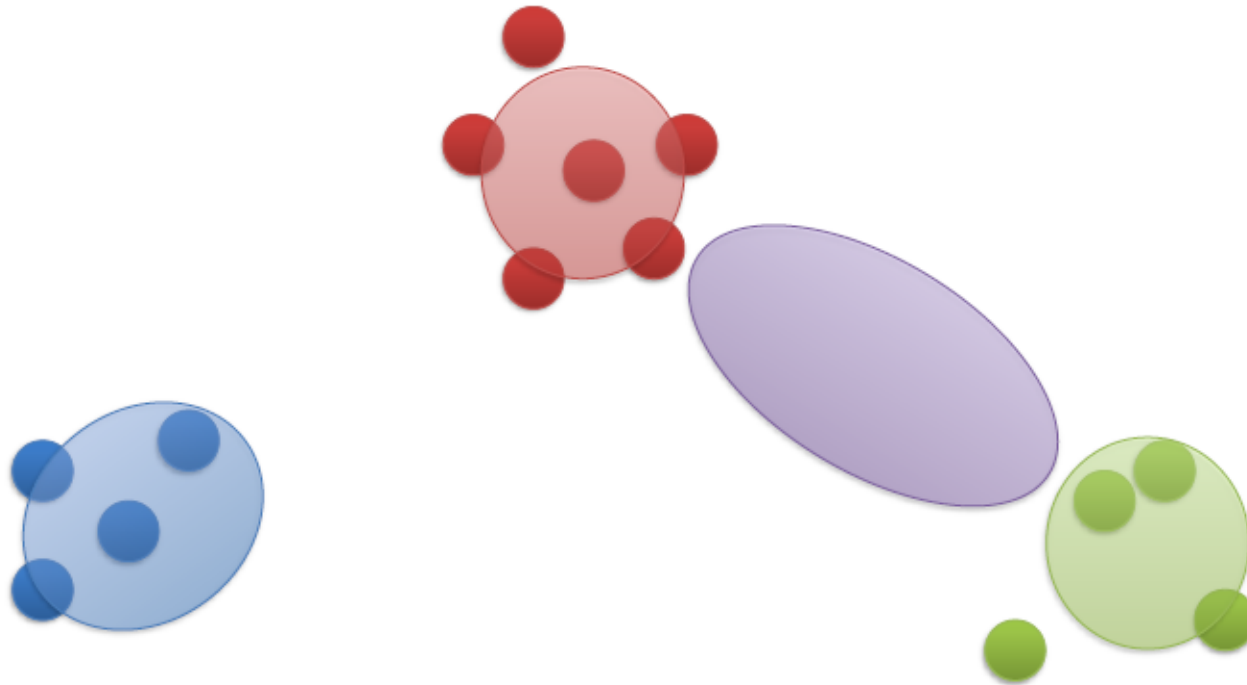


After 20 iteration:



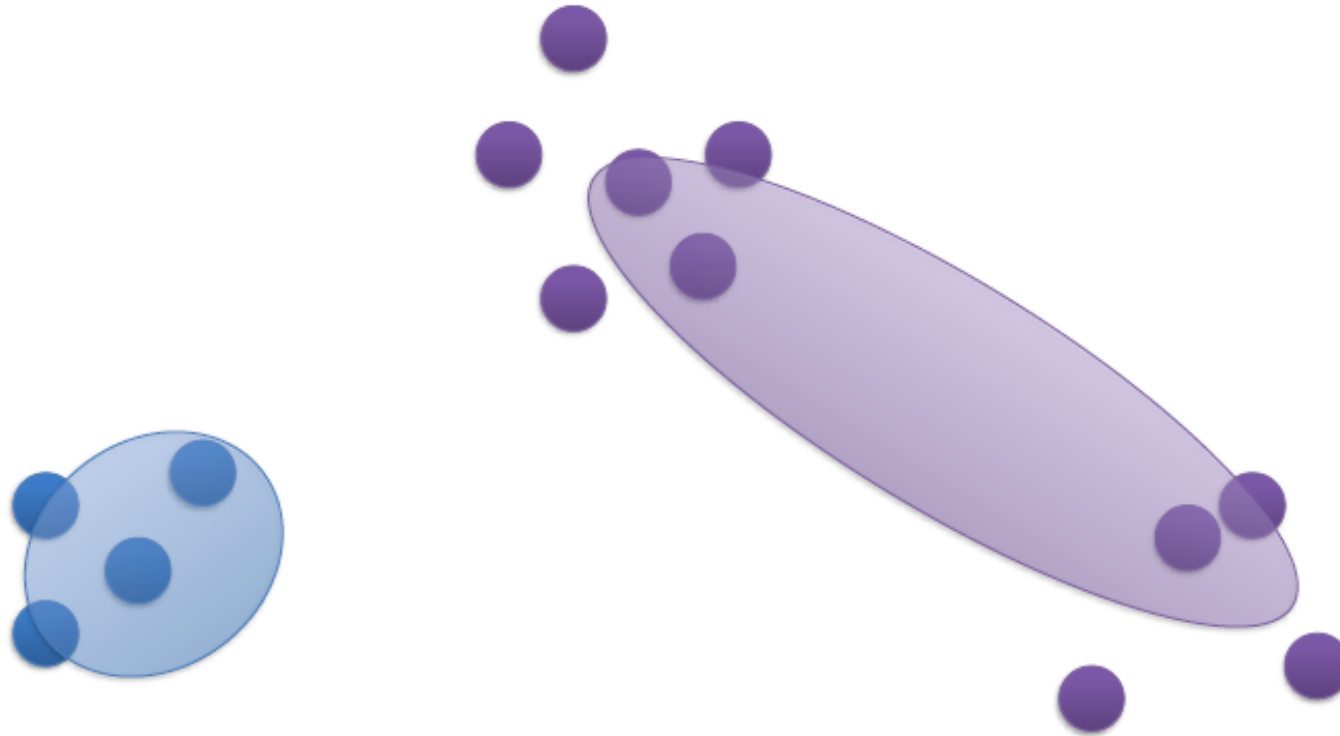
Potential Problem: Incorrect number of Gaussians

Example 1



Potential Problem: Incorrect number of Gaussians

Example 2

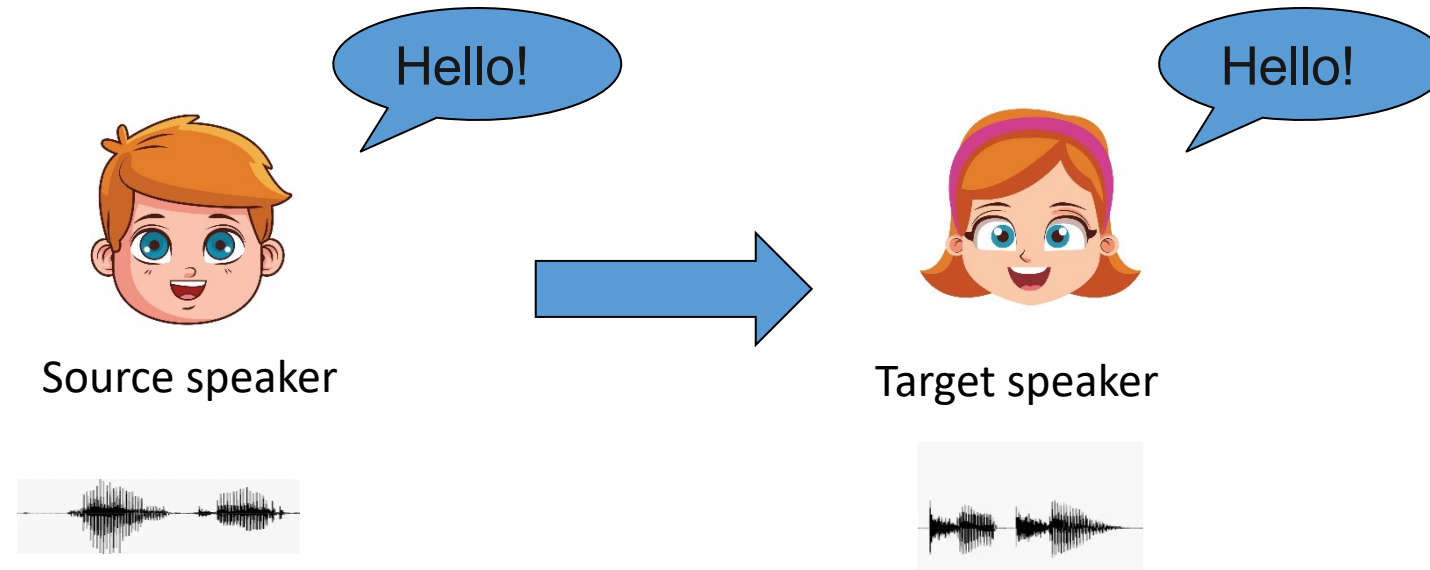


GMM Real-life Applications

Can we convert someone's identity?

Voice Cloning/Conversion

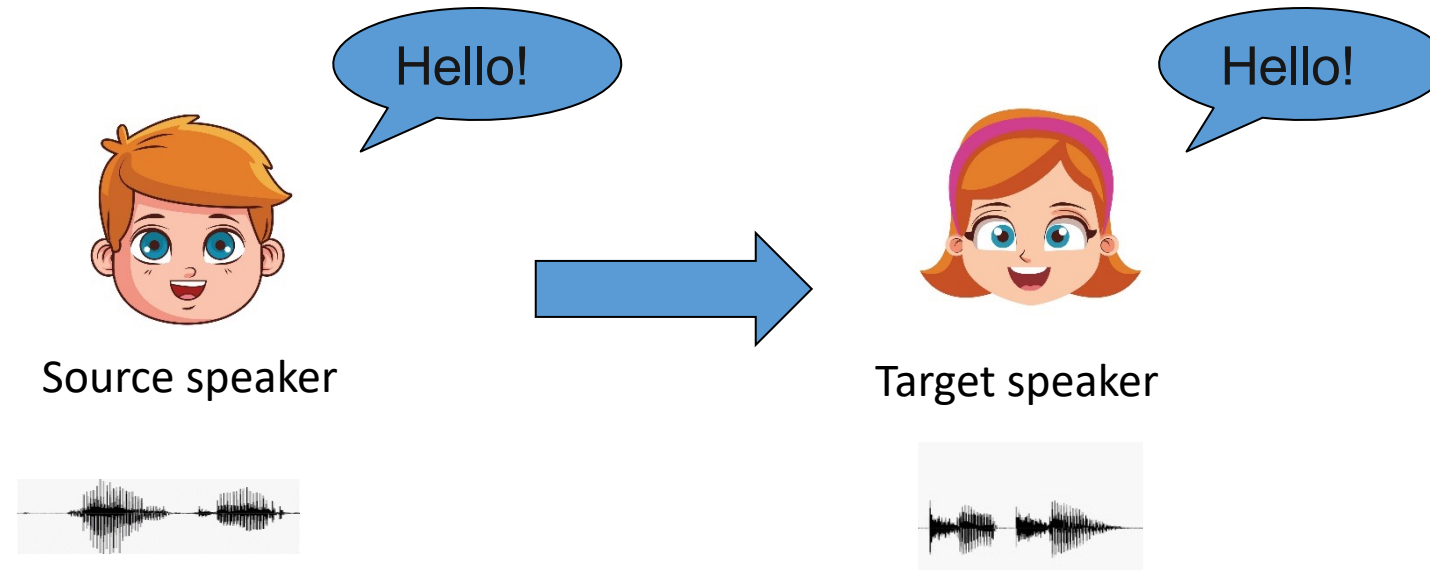
Voice conversion (VC) is a technique to convert a speaker identity of a source speaker into that of a target speaker.



A traditional VC system based on a Gaussian mixture model (GMM)
<https://github.com/k2kobayashi/sprocket>

Voice Cloning/Conversion

Voice conversion (VC) is a technique to convert a speaker identity of a source speaker into that of a target speaker.



Things to consider:

Data limitation, language, accent, speaking style, emotional states, gender,

GMM Examples – Human Voice Generation and Modification

IEEE Xplore® Browse ▾ My Settings ▾ Help ▾ Institutional Sign In

All ▾ ADVANCED SEARCH

Conferences > 2001 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2001)

Voice conversion algorithm based on Gaussian mixture model with dynamic frequency warping of STRAIGHT spectrum

Publisher: IEEE Cite This PDF

T. Toda; H. Saruwatari; K. Shikano All Authors

65 Paper Citations 4 Patent Citations 828 Full Text Views

Abstract

Document Sections

1. INTRODUCTION

Abstract:
In the voice conversion algorithm based on the Gaussian Mixture Model (GMM) applied to STRAIGHT, quality of converted speech is degraded because the converted spectrum is exceedingly smooth. We propose the GMM-based algorithm with dynamic frequency warping to avoid the over-smoothing. We also propose an addition of the weighted residual spectrum, which is the difference between the GMM-based

ICASSP 2001

IEEE Xplore® Browse ▾ My Settings ▾ Help ▾ Institutional Sign In

All ▾ ADVANCED SEARCH

Conferences > 2018 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2018)

Statistical Voice Conversion Based on Wavenet

Publisher: IEEE Cite This PDF

Jumpei Niwa; Takenori Yoshimura; Kei Hashimoto; Keiichi Oura; Yoshihiko Nankaku; Keiichi Tokuda All Authors

6 Paper Citations 1 Patent Citation 406 Full Text Views

Abstract

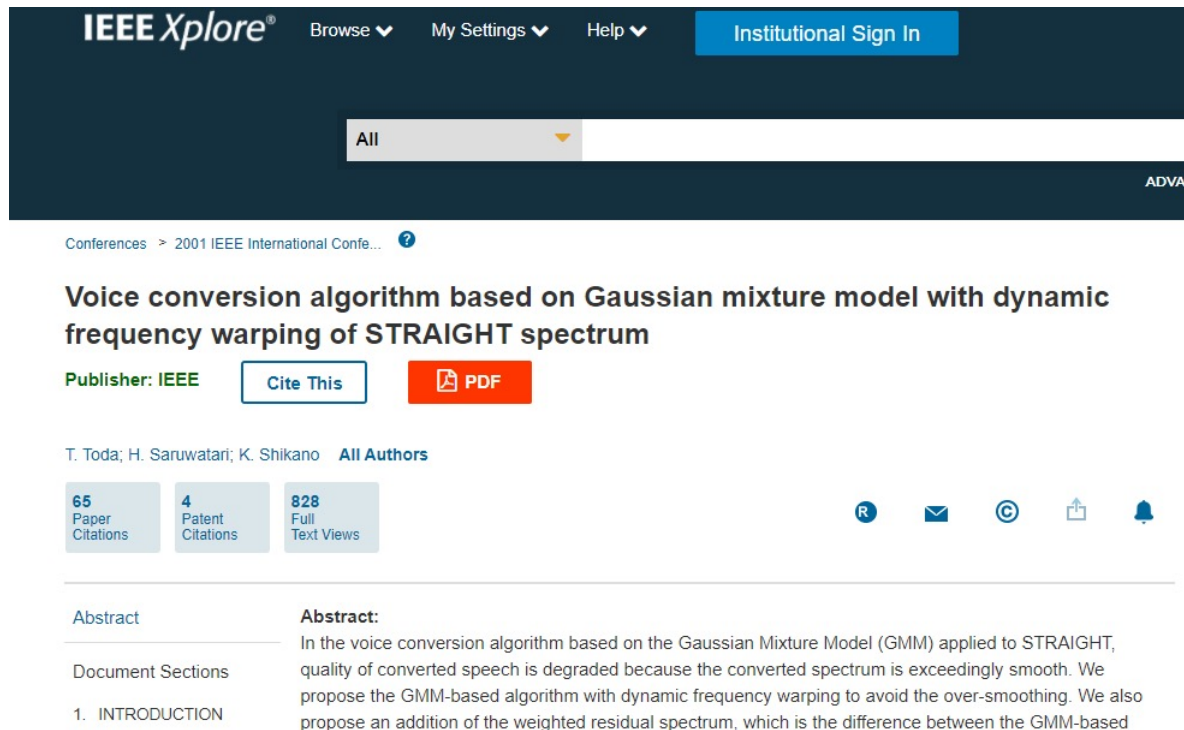
Document Sections

1. Introduction
2. Wavenet

Abstract:
This paper proposes a voice conversion technique based on WaveNet to directly generate target audio waveforms from acoustic features of a source speaker. In voice conversion based on statistical models, the relation between acoustic features, such as spectral parameters, extracted from source and target audio waveforms is generally modeled using statistical models, such as Gaussian mixture models and neural networks. Although modeling the relation between acoustic features is reasonable and efficient, these models are not optimized for predicting target audio waveforms because the vocoder parameters are used

ICASSP 2018

GMM Examples – Human Voice Generation and Modification



IEEE Xplore® Browse ▾ My Settings ▾ Help ▾ Institutional Sign In

All ▾ ADVANCED SEARCH

Conferences > 2001 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2001)

Voice conversion algorithm based on Gaussian mixture model with dynamic frequency warping of STRAIGHT spectrum

Publisher: IEEE Cite This PDF

T. Toda; H. Saruwatari; K. Shikano All Authors

65 Paper Citations 4 Patent Citations 828 Full Text Views

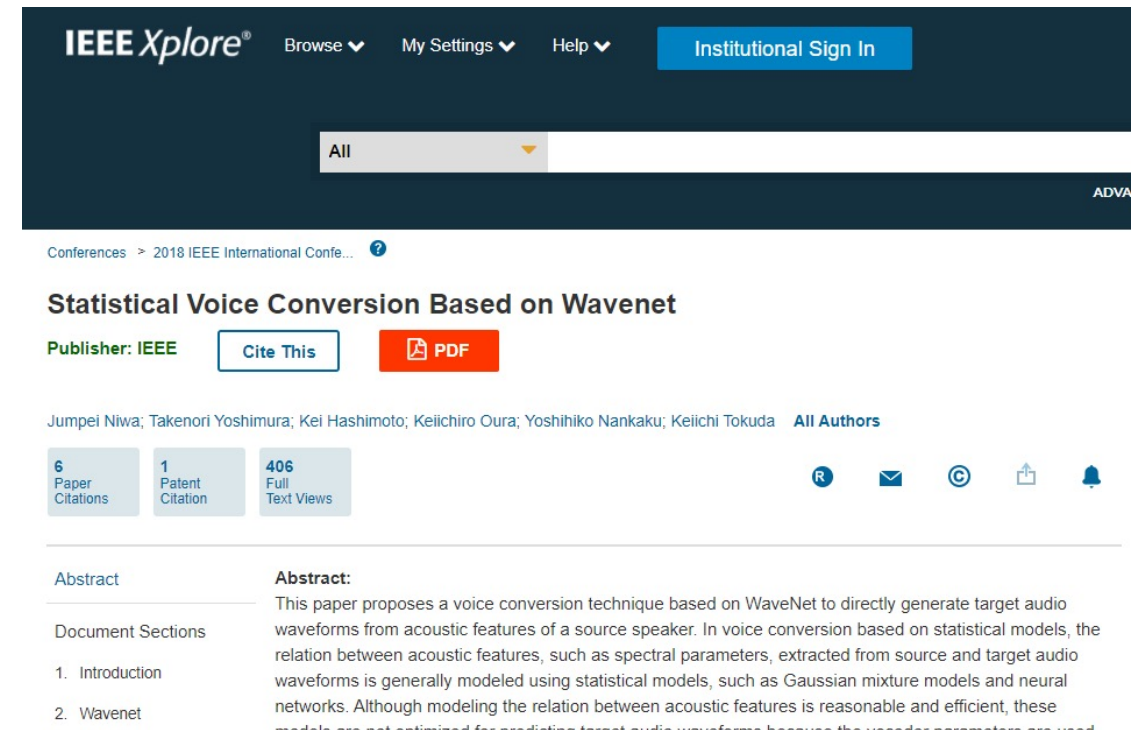
Abstract

Document Sections

1. INTRODUCTION

Abstract:
In the voice conversion algorithm based on the Gaussian Mixture Model (GMM) applied to STRAIGHT, quality of converted speech is degraded because the converted spectrum is exceedingly smooth. We propose the GMM-based algorithm with dynamic frequency warping to avoid the over-smoothing. We also propose an addition of the weighted residual spectrum, which is the difference between the GMM-based

ICASSP 2001



IEEE Xplore® Browse ▾ My Settings ▾ Help ▾ Institutional Sign In

All ▾ ADVANCED SEARCH

Conferences > 2018 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2018)

Statistical Voice Conversion Based on Wavenet

Publisher: IEEE Cite This PDF

Jumpei Niwa; Takenori Yoshimura; Kei Hashimoto; Keiichi Oura; Yoshihiko Nankaku; Keiichi Tokuda All Authors

6 Paper Citations 1 Patent Citation 406 Full Text Views

Abstract

Document Sections

1. Introduction
2. Wavenet

Abstract:
This paper proposes a voice conversion technique based on WaveNet to directly generate target audio waveforms from acoustic features of a source speaker. In voice conversion based on statistical models, the relation between acoustic features, such as spectral parameters, extracted from source and target audio waveforms is generally modeled using statistical models, such as Gaussian mixture models and neural networks. Although modeling the relation between acoustic features is reasonable and efficient, these models are not optimized for predicting target audio waveforms because the vocoder parameters are used

ICASSP 2018

Please note that we can use advance deep learning techniques for voice cloning :)

GMM Examples – Face Editing



This CVPR 2020 paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Interpreting the Latent Space of GANs for Semantic Face Editing

Yujun Shen¹, Jinjin Gu², Xiaoou Tang¹, Bolei Zhou¹

¹The Chinese University of Hong Kong ²The Chinese University of Hong Kong, Shenzhen

{sy116, xtang, bzhou}@ie.cuhk.edu.hk, jinjingu@link.cuhk.edu.cn

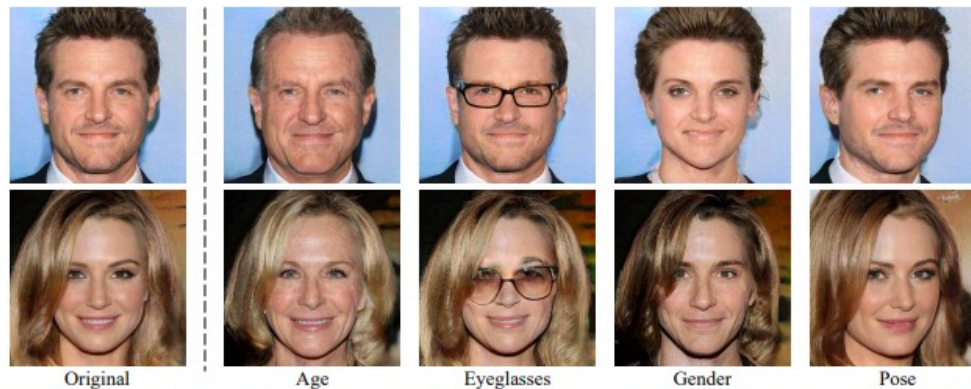


Figure 1: Manipulating various facial attributes through varying the latent codes of a well-trained GAN model. The first column shows the original synthesis from PGGAN [19], while each of the other columns shows the results of manipulating a specific attribute.

AND MANY MORE!

CVPR 2020

(latent space modeling with many Gaussian distributions)