Sample-efficient Learning of Mixture Models

Abbas Mehrabian
McGill University, Montreal, Canada

Joint work with Hassan Ashtiani and Shai Ben-David (Waterloo)

(Slides created by Hassan)

July 13, 2017

Overview

- Density estimation (a.k.a. distribution learning)
  - Given an iid sample generated from an unknown density g^*, find density $\hat{g}$ that is "close" to g.

Overview

- Density estimation (a.k.a. distribution learning)
  - Given an iid sample generated from an unknown density g^*, find density $\hat{g}$ that is "close" to g.
- Close in what sense?

Overview

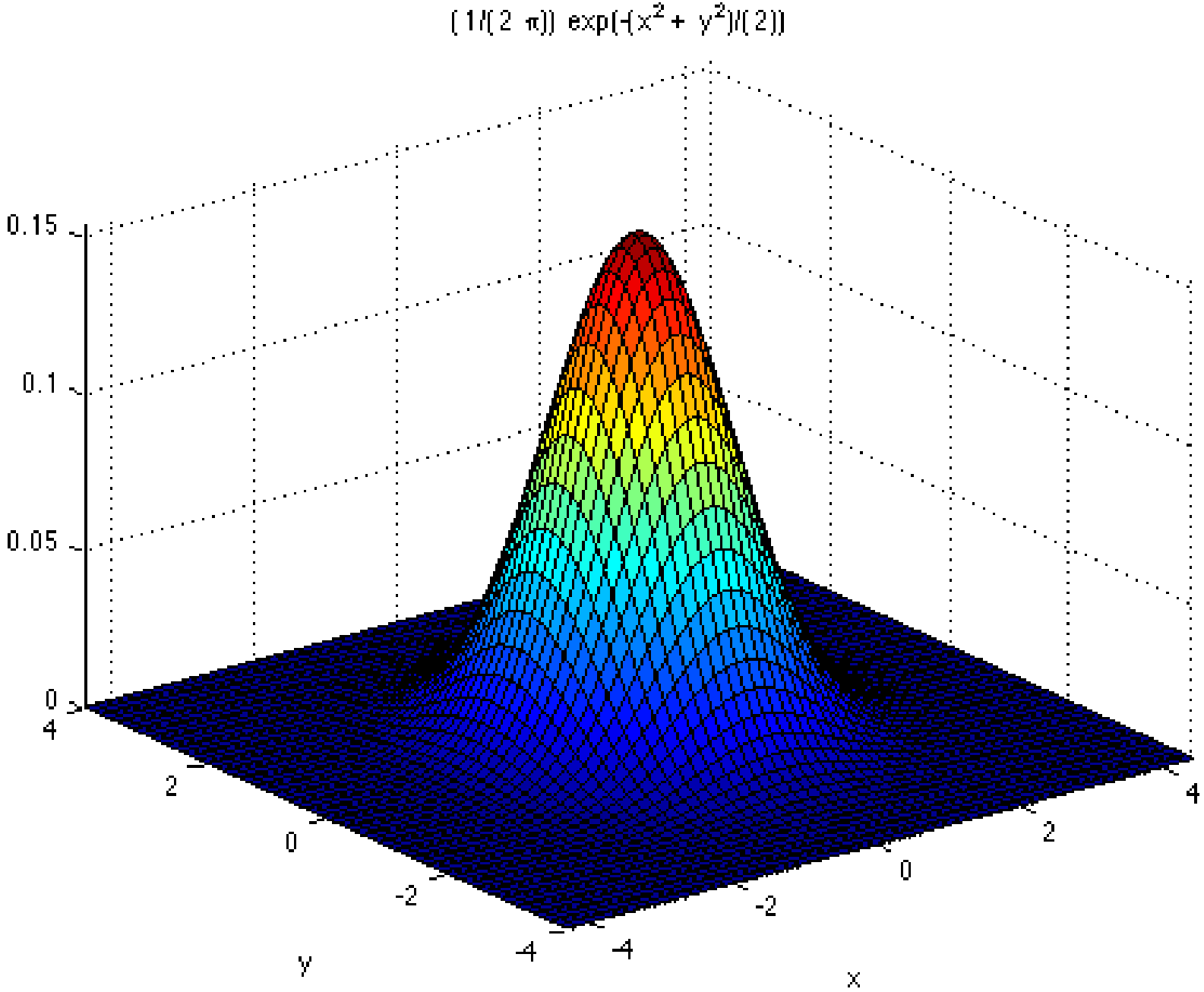
- Density estimation (a.k.a. distribution learning)
  - Given an iid sample generated from an unknown density g^*, find density $\hat{g}$ that is "close" to g.
- Close in what sense?
- Learning Mixtures: g^* belongs to (or can be approximated by) a "mixture class"
  - E.g., learning a mixture of k Gaussians

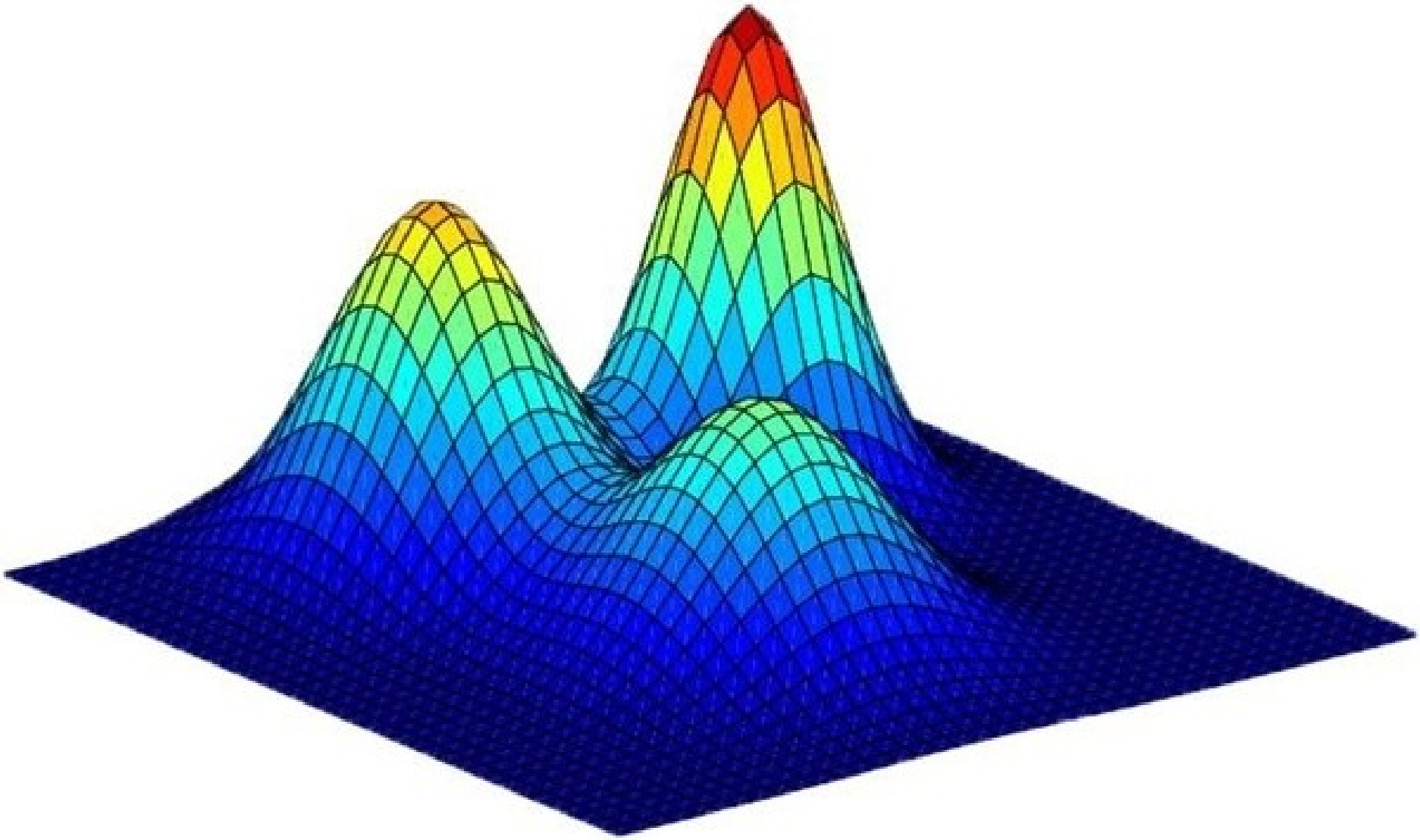
Overview

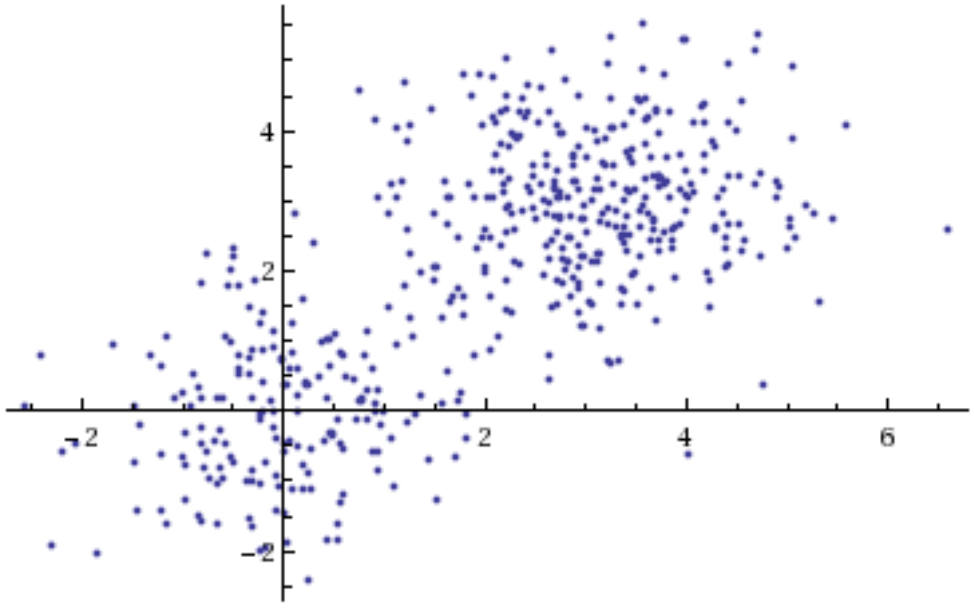
- Density estimation (a.k.a. distribution learning)
  - Given an iid sample generated from an unknown density g^*, find density $\hat{g}$ that is "close" to g.
- Close in what sense?
- Learning Mixtures: g^* belongs to (or can be approximated by) a "mixture class"
  - E.g., learning a mixture of k Gaussians
- How can we efficiently "learn" mixture models?
  - Statistical efficiency
  - Computational efficiency

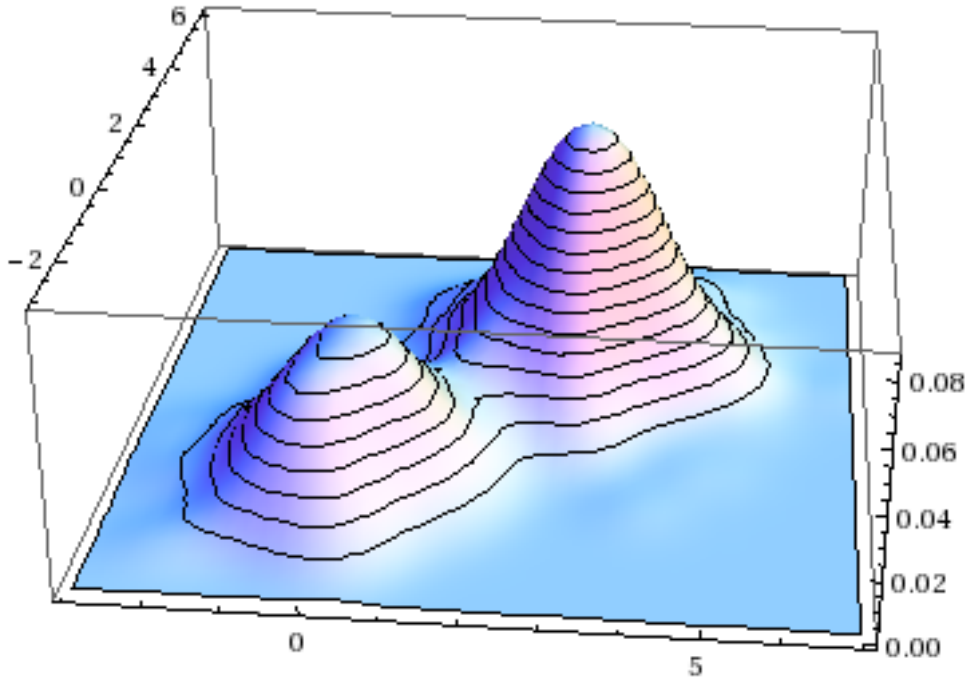
Overview

- Density estimation (a.k.a. distribution learning)
  - Given an iid sample generated from an unknown density g^*, find density $\hat{g}$ that is "close" to g.
- Close in what sense?
- Learning Mixtures: g^* belongs to (or can be approximated by) a "mixture class"
  - E.g., learning a mixture of k Gaussians
- How can we efficiently "learn" mixture models?
  - Statistical efficiency
  - Computational efficiency
- Our focus is on statistical efficiency
- We improve the state-of-the-art for learning mixture models in terms of sample complexity.


(1/(2 π)) exp(-(x^2 + y^2)/(2))
0.15
0.1
0.05
0
4
2
0
-2
-4
y
-4
-2
0
2
4
x




4
2
−2
2
4
6
−2


6
4
2
0
−2
0
5
0.08
0.06
0.04
0.02
0.00

Mixture Classes

- $\mathcal{F}$ is a class of probability density functions.
  - E.g., Multivariate Gaussian pdfs over $\mathbb{R}^d$
- Δ_k is the k-dimensional simplex.
  - $\Delta_k = \{(w_1, \ldots, w_k) : w_i \geq 0, \sum w_i = 1\}$
- The class of k-mixtures of $\mathcal{F}$ is defined by
  - $\mathcal{F}^k := \{\sum_{i=1}^k w_i f_i : (w_1, \ldots, w_k) \in \Delta_k, f_1, \ldots, f_k \in \mathcal{F}\}$
  - E.g., Gaussian Mixture Models (GMMs) with k components.
- Lots of applications, e.g., in modelling multi-modal distributions

Density Estimation vs Parameter Estimation/Identification

- Parameter Learning/Identification
  - E.g., approximate μ_i, Σ_i, w_i for GMMs
  - Is in general harder, exponential in k samples needed.

Density Estimation vs Parameter Estimation/Identification

- Parameter Learning/Identification
  - E.g., approximate μ_i, Σ_i, w_i for GMMs
  - Is in general harder, exponential in k samples needed.
- Density Estimation
  - The output of the algorithm is a pdf, $\hat{g}$
  - The goal is to have $d(g, \hat{g}) < \epsilon$ in some metric

Density Estimation vs Parameter Estimation/Identification

- Parameter Learning/Identification
  - E.g., approximate μ_i, Σ_i, w_i for GMMs
  - Is in general harder, exponential in k samples needed.
- Density Estimation
  - The output of the algorithm is a pdf, $\hat{g}$
  - The goal is to have $d(g, \hat{g}) < \epsilon$ in some metric
  - Our focus: density estimation w.r.t. the total variation distance
    - $\|g - \hat{g}\|_{TV} := \sup_{A \subset \mathbb{R}^d} |g(A) - \hat{g}(A)|$

Density Estimation vs Parameter Estimation/Identification

- Parameter Learning/Identification
  - E.g., approximate μ_i, Σ_i, w_i for GMMs
  - Is in general harder, exponential in k samples needed.
- Density Estimation
  - The output of the algorithm is a pdf, $\hat{g}$
  - The goal is to have $d(g, \hat{g}) < \epsilon$ in some metric
  - Our focus: density estimation w.r.t. the total variation distance
    - $\|g - \hat{g}\|_{TV} := \sup_{A \subset \mathbb{R}^d} |g(A) - \hat{g}(A)|$
    - $\|g - \hat{g}\|_{TV} = \frac{1}{2}\|g - \hat{g}\|_1 = \frac{1}{2}\int_z |g - \hat{g}| dz$

PAC-Learning Distributions

PAC-Learning of Distributions (Realizable Setting)

A distribution learning method is called a PAC-learner for $\mathcal{F}$ with sample complexity $m_{\mathcal{F}}(\epsilon, \delta)$ if for all distribution $g \in \mathcal{F}$ and all $\epsilon, \delta > 0$, given ϵ, δ, and a sample of size $m_{\mathcal{F}}(\epsilon, \delta)$, with probability at least $1 - \delta$ outputs an ϵ-approximation of g.

Results

Main Theorem

Assume that $\mathcal{F}$ can be learned in the realizable setting with sample complexity $m_{\mathcal{F}}(\epsilon, \delta)$ Then the class $\mathcal{F}^k$ can be learned with

$$m_{\mathcal{F}^k}(\epsilon, \delta) = O\left(\frac{k \log k \cdot m_{\mathcal{F}}(\epsilon, \delta/3k)}{\epsilon^2}\right)$$

samples.

So by going from one component to a mixture of k components, the sample complexity increases at most by a factor of $(k \log k)/\epsilon^2$.

Applications

Learning Axis-aligned Gaussians

The class of d-dimensional axis-aligned Gaussians is PAC-learnable in the realizable setting with $m_{\mathcal{F}}(\epsilon, \delta) = O((d + \log(1/\delta))/\epsilon^2)$

Applications

Learning Axis-aligned Gaussians

The class of d-dimensional axis-aligned Gaussians is PAC-learnable in the realizable setting with $m_{\mathcal{F}}(\epsilon, \delta) = O((d + \log(1/\delta))/\epsilon^2)$

Learning Mixture of Axis-aligned Gaussians

The class of mixtures of k axis-aligned Gaussians in $\mathbb{R}^d$ is PAC-learnable with sample complexity $m_{\mathcal{F}^k}(\epsilon, \delta) = O(kd \log k \log(k/\delta)/\epsilon^4) = \widetilde{O}(kd/\epsilon^4)$.

Applications

Learning Axis-aligned Gaussians

The class of d-dimensional axis-aligned Gaussians is PAC-learnable in the realizable setting with $m_{\mathcal{F}}(\epsilon, \delta) = O((d + \log(1/\delta))/\epsilon^2)$

Learning Mixture of Axis-aligned Gaussians

The class of mixtures of k axis-aligned Gaussians in $\mathbb{R}^d$ is PAC-learnable with sample complexity $m_{\mathcal{F}^k}(\epsilon, \delta) = O(kd \log k \log(k/\delta)/\epsilon^4) = \widetilde{O}(kd/\epsilon^4)$.

- (Somewhat) Improvement over previous known upper bounds
  - $\widetilde{O}(dk^9/\epsilon^4)$ (Suresh, Orlitsky, Acharya, Jafarpour'14)
  - $\widetilde{O}((d^2k^4 + d^3k^3)/\epsilon^2)$ (Karpinski and Macintyre (1997))

Applications

Learning Axis-aligned Gaussians

The class of d-dimensional axis-aligned Gaussians is PAC-learnable in the realizable setting with $m_{\mathcal{F}}(\epsilon, \delta) = O((d + \log(1/\delta))/\epsilon^2)$

Learning Mixture of Axis-aligned Gaussians

The class of mixtures of k axis-aligned Gaussians in $\mathbb{R}^d$ is PAC-learnable with sample complexity $m_{\mathcal{F}^k}(\epsilon, \delta) = O(kd \log k \log(k/\delta)/\epsilon^4) = \widetilde{O}(kd/\epsilon^4)$.

- (Somewhat) Improvement over previous known upper bounds
  - $\widetilde{O}(dk^9/\epsilon^4)$ (Suresh, Orlitsky, Acharya, Jafarpour'14)
  - $\widetilde{O}((d^2k^4 + d^3k^3)/\epsilon^2)$ (Karpinski and Macintyre (1997))

Lower Bound (Suresh, Orlitsky, Acharya, Jafarpour'14)

The sample complexity of learning mixtures of k axis-aligned Gaussians in $\mathbb{R}^d$ in the realizable setting is $m_{\mathcal{F}^k}(\epsilon, 1/2) = \Omega(dk/\epsilon^2)$.

Applications

Learning Gaussians

The class of d-dimensional Gaussians is PAC-learnable in the realizable setting with $m_{\mathcal{F}}(\epsilon, \delta) = O((d^2 + \log(1/\delta))/\epsilon^2)$

Learning Mixture of Gaussians

The class $\mathcal{F}^k$ of mixtures of k Gaussians in $\mathbb{R}^d$ is PAC-learnable with sample complexity $m_{\mathcal{F}^k}(\epsilon, \delta) = \widetilde{O}(kd^2/\epsilon^4)$.

Applications

Learning Gaussians

The class of d-dimensional Gaussians is PAC-learnable in the realizable setting with $m_{\mathcal{F}}(\epsilon, \delta) = O((d^2 + \log(1/\delta))/\epsilon^2)$

Learning Mixture of Gaussians

The class $\mathcal{F}^k$ of mixtures of k Gaussians in $\mathbb{R}^d$ is PAC-learnable with sample complexity $m_{\mathcal{F}^k}(\epsilon, \delta) = \widetilde{O}(kd^2/\epsilon^4)$.

- (Somewhat) Improvement over previous known upper bounds
  - $\widetilde{O}(d^2k^3/\epsilon^4)$ (Diakonikolas, Kane, Stewart'17)
  - $\widetilde{O}(d^4k^4/\epsilon^2)$ (Karpinski and Macintyre'97)

High level overview of the algorithm

- Create a finite set of "candidate" pdfs based on the sample
- Choose the best one based on a fresh new sample

High level overview of the algorithm

- Create a finite set of "candidate" pdfs based on the sample
- Choose the best one based on a fresh new sample

Learning Finite Classes (Devroye and Lugosi (2001))

For a finite class $\mathcal{F}$ of distributions, there is an algorithm than learns $\mathcal{F}$ with $O(\log(M)/\epsilon^2)$ new samples.

Learning means choosing the closest candidate from $\mathcal{F}$ to the target.

High level overview of the algorithm

- Create a finite set of "candidate" pdfs based on the sample
- Choose the best one based on a fresh new sample

Learning Finite Classes (Devroye and Lugosi (2001))

For a finite class $\mathcal{F}$ of distributions, there is an algorithm than learns $\mathcal{F}$ with $O(\log(M)/\epsilon^2)$ new samples.

Learning means choosing the closest candidate from $\mathcal{F}$ to the target.

Note that we know how to learn a single Gaussian using d^2/ε^2 samples
But in the case of mixtures, we don't know which sample point came from which component of mixture,
but we can try "all" possible cases (exhaustive search) to generate the candidates.

Algorithm

Recall, target has form $\{\sum_{i=1}^{k} w_i f_i : w_i \geq 0, \sum w_i = 1, f_1, \ldots, f_k \in \mathcal{G}_d\}$
In words: try all possible ways of partitioning data into components, and all possible mixture weights

Input: k, ϵ, δ and an iid sample S of size kd^2/ε^2

0. Let $\widehat{W}$ be an (ϵ/k)-cover for Δ_k in ℓ_∞ distance.
1. $\mathcal{C} = \emptyset$. (set of candidate distributions)
2. For each $(\widehat{w}_1, \ldots, \widehat{w}_k) \in \widehat{W}$ do:
   3. For each possible partition of S into $A_1, A_2, ..., A_{k+1}$:
      4. Provide A_i to the $\mathcal{F}$-learner, and let $\widehat{G}_i$ be its output.
      5. Add the candidate distribution $\sum_{i \in [k]} \widehat{w}_i \widehat{G}_i$ to $\mathcal{C}$.
6. Apply the algorithm for finite classes to $\mathcal{C}$ and output its result.

Algorithm

Recall, target has form $\{\sum_{i=1}^{k} w_i f_i : w_i \geq 0, \sum w_i = 1, f_1, \ldots, f_k \in \mathcal{G}_d\}$
In words: try all possible ways of partitioning data into components, and all possible mixture weights

Input: k, ϵ, δ and an iid sample S of size kd^2/ε^2

0. Let $\widehat{W}$ be an (ϵ/k)-cover for Δ_k in ℓ_∞ distance.
1. $\mathcal{C} = \emptyset$. (set of candidate distributions)
2. For each $(\widehat{w}_1, \ldots, \widehat{w}_k) \in \widehat{W}$ do:
   3. For each possible partition of S into $A_1, A_2, ..., A_{k+1}$:
      4. Provide A_i to the $\mathcal{F}$-learner, and let $\widehat{G}_i$ be its output.
      5. Add the candidate distribution $\sum_{i \in [k]} \widehat{w}_i \widehat{G}_i$ to $\mathcal{C}$.
6. Apply the algorithm for finite classes to $\mathcal{C}$ and output its result.

- $|C| \leq k^{kd^2/\varepsilon^2} \times (1/\varepsilon)^k$ so $\log |C| \leq (kd^2/\varepsilon^2) \log k + k \log(1/\varepsilon)$.
- Remains to prove that there is an ϵ-close candidate in $\mathcal{C}$.

Open Questions

- For learning mixture of k Gaussians in $\mathbb{R}^d$,

$$\frac{kd}{\epsilon^2} \leq \text{statistical complexity} \leq \frac{kd^2}{\epsilon^4}$$

within log factors. Close the gap!
Guess: correct answer is $\frac{kd^2}{\epsilon^2}$

Open Questions

- For learning mixture of k Gaussians in $\mathbb{R}^d$,

$$\frac{kd}{\epsilon^2} \leq \text{statistical complexity} \leq \frac{kd^2}{\epsilon^4}$$

  within log factors. Close the gap!
  Guess: correct answer is $\frac{kd^2}{\epsilon^2}$
- Polynomial algorithm?

Open Questions

- For learning mixture of k Gaussians in $\mathbb{R}^d$,

$$\frac{kd}{\epsilon^2} \leq \text{statistical complexity} \leq \frac{kd^2}{\epsilon^4}$$

  within log factors. Close the gap!
  Guess: correct answer is $\frac{kd^2}{\epsilon^2}$
- Polynomial algorithm?

Thank You!