Learning Gaussian Mixtures and Multiplayer Online Learning

Abbas Mehrabian

McGill University
IVADO Fellow

8 April 2019

Past and current research

Past research: Randomized algorithms with applications to network science and distributed algorithms

Current interests: mathematical foundations of ML

Active research areas:

1. Distribution learning
2. Online learning

Outline

1. Distribution learning

2. Online learning

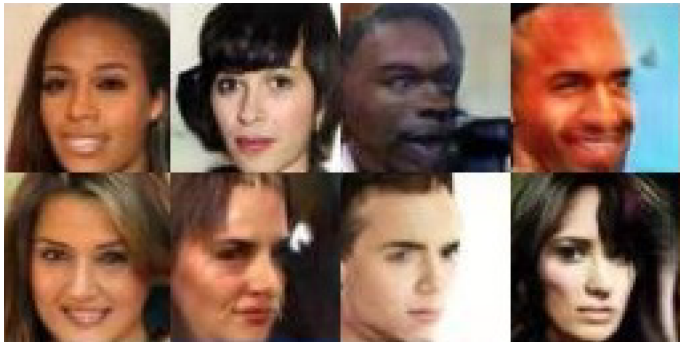
What is distribution learning?

- ✓ Unsupervised learning: find structure in data
  1. Clustering
  2. Anomaly detection
  3. Principal component analysis
  4. Generative models

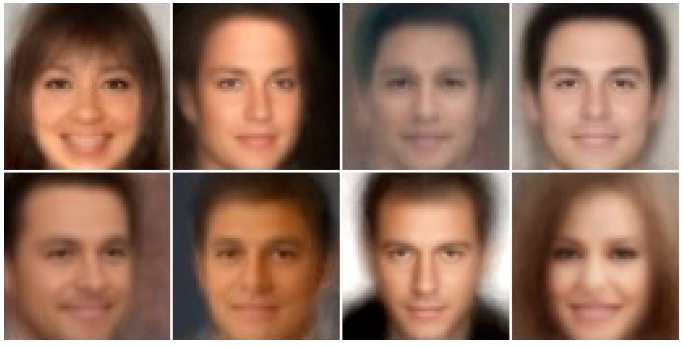
- ✓ Distribution learning: explicitly learn the data distribution

An example of distribution learning

Generating random faces



neural networks



Gaussian mixtures

[Richardson and Weiss 2018]

1. Training data consists of actual faces.
2. A probability density function is learned.
3. Generate new faces using the learned distribution.

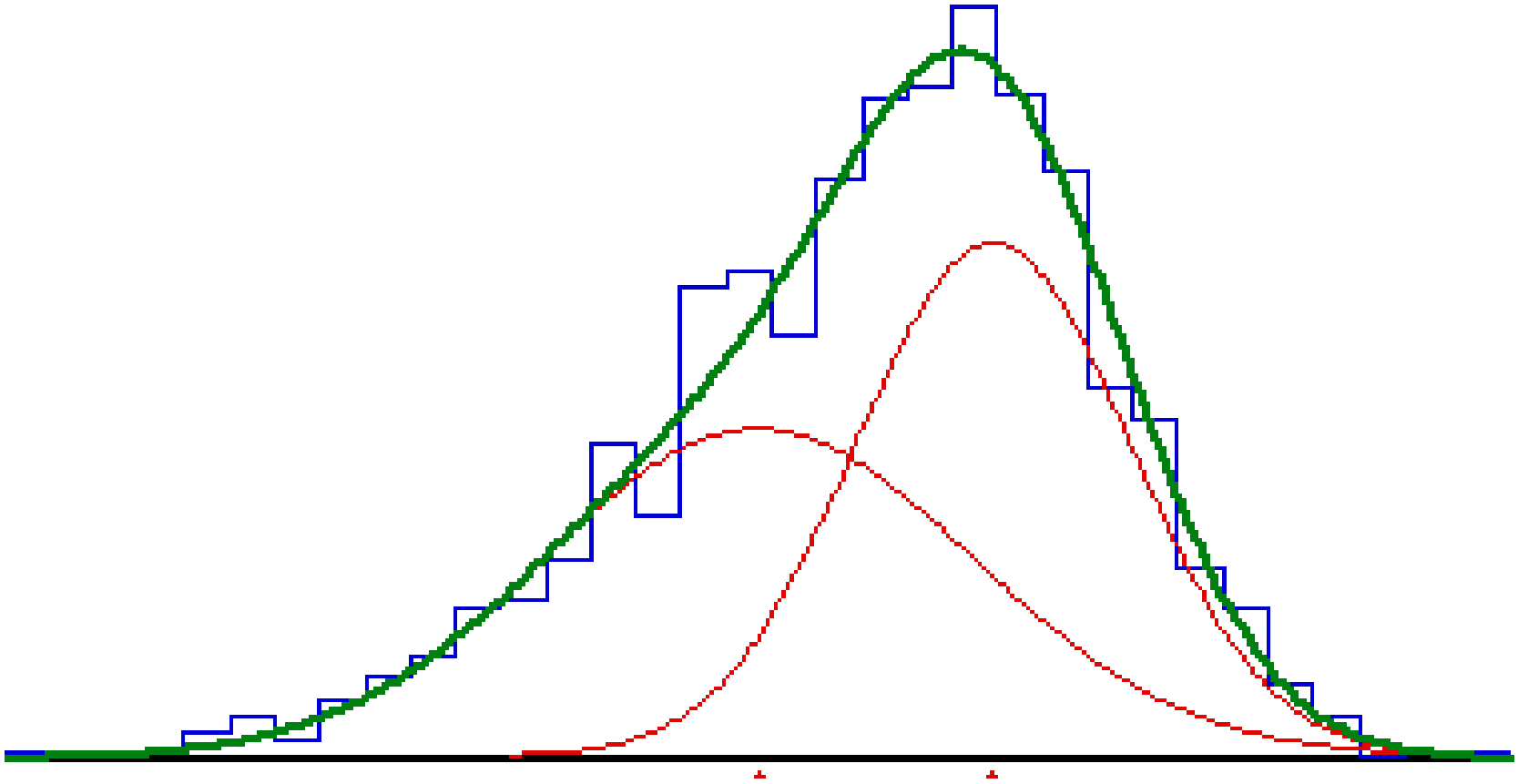
Modelling data with mixtures of Gaussians

Widely used in practice: [Tarassenko, Hayton, Cerneaz, Brady 1996], [Gao, Tai-hua, Gao 2010], [Doroshin, Tkachenko, Lubimov, Kotov 2013], [Stepanek, Franc, Kus 2015], [Khanmohammadi, Chou 2016]

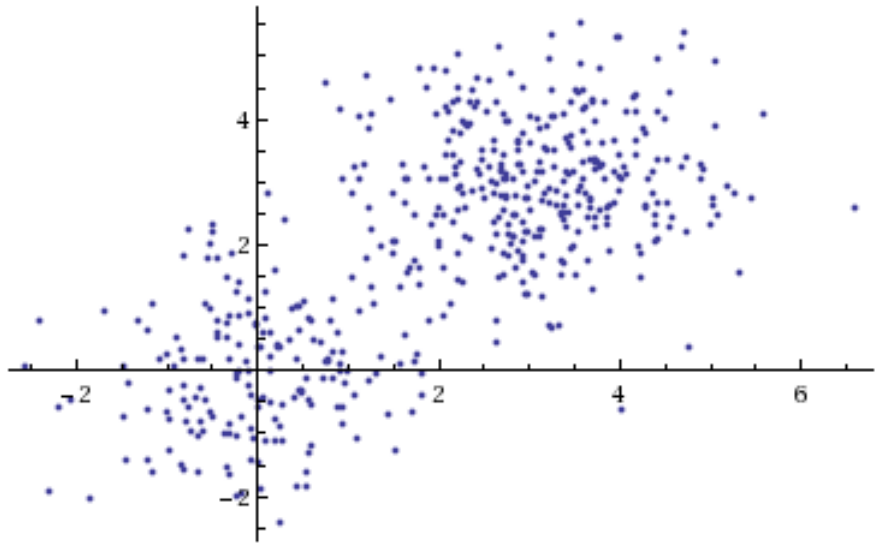
Why mixtures of Gaussians?

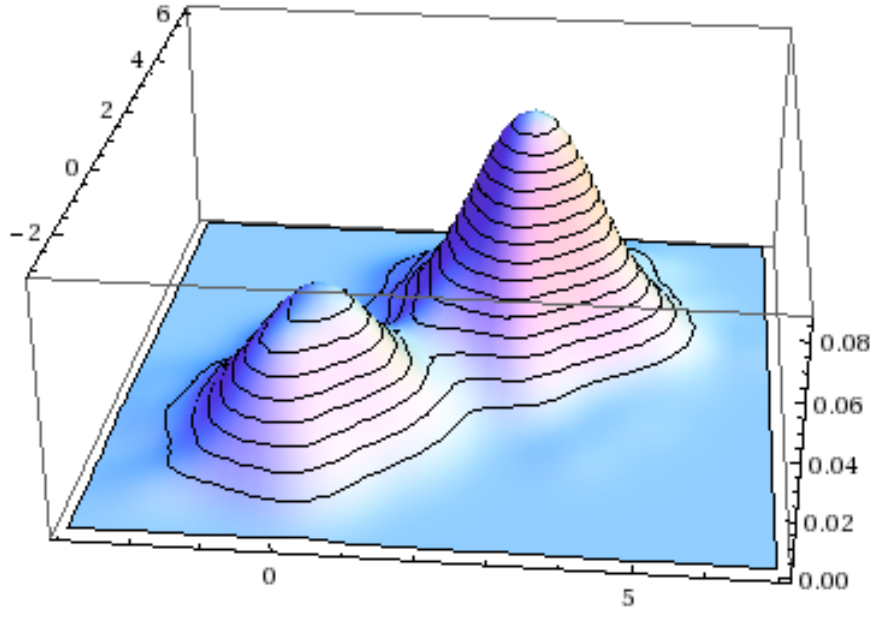
- ✓ fit some natural data well
- ✓ universal approximators
- ✓ clustering

Fitting crabs forehead lengths by a Gaussian mixture



Data collected by Raphael Weldon
Mixture parameters estimated by Karl Pearson (1894)
Plot by Peter D. M. Macdonald, McMaster University


4
2
−2
−2
2
4
6


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

Problem formulation

Goal: learn a Gaussian mixture given a polynomial-size sample.

1. Maximum likelihood estimation.
2. Parameter estimation. Learn the model parameters.
3. Density estimation. Learn the density.

Problem formulation

Goal: learn a Gaussian mixture given a polynomial-size sample.

1. Maximum likelihood estimation. EM algorithm.
   Does not always converge to a local maximum
   [Redner and Walker 1984].
2. Parameter estimation. Learn the model parameters.
3. Density estimation. Learn the density.

Problem formulation

Goal: learn a Gaussian mixture given a polynomial-size sample.

1. Maximum likelihood estimation. EM algorithm.
   Does not always converge to a local maximum
   [Redner and Walker 1984].
2. Parameter estimation. Learn the model parameters.
   Exponential sample complexity
   [Moitra, Valiant 2010].
3. Density estimation. Learn the density.

Problem formulation

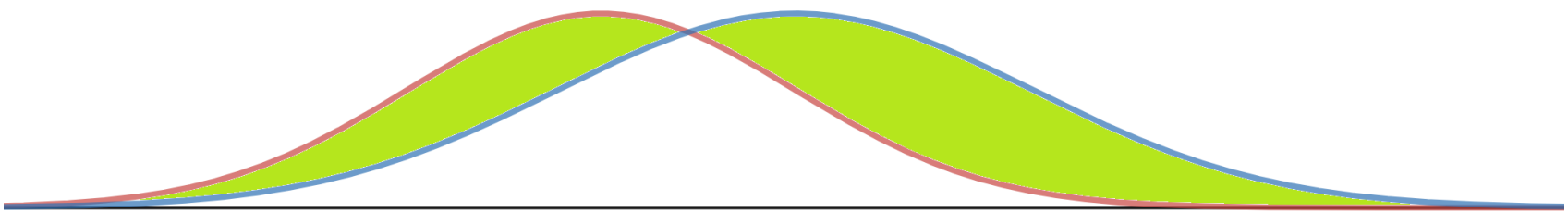
Goal: learn a Gaussian mixture given a polynomial-size sample.

1. Maximum likelihood estimation. EM algorithm.
   Does not always converge to a local maximum
   [Redner and Walker 1984].
2. Parameter estimation. Learn the model parameters.
   Exponential sample complexity
   [Moitra, Valiant 2010].
3. Density estimation. Learn the density.
   Polynomial sample complexity.

Density estimation

Given samples from an unknown density f from some known class $\mathcal{C}$ of densities, output a density $\widehat{f}$ that is close to f.

$$\left\| f - \widehat{f} \right\|_1 = \int \left| f(x) - \widehat{f}(x) \right| \mathrm{d}x$$



Precise question we study today

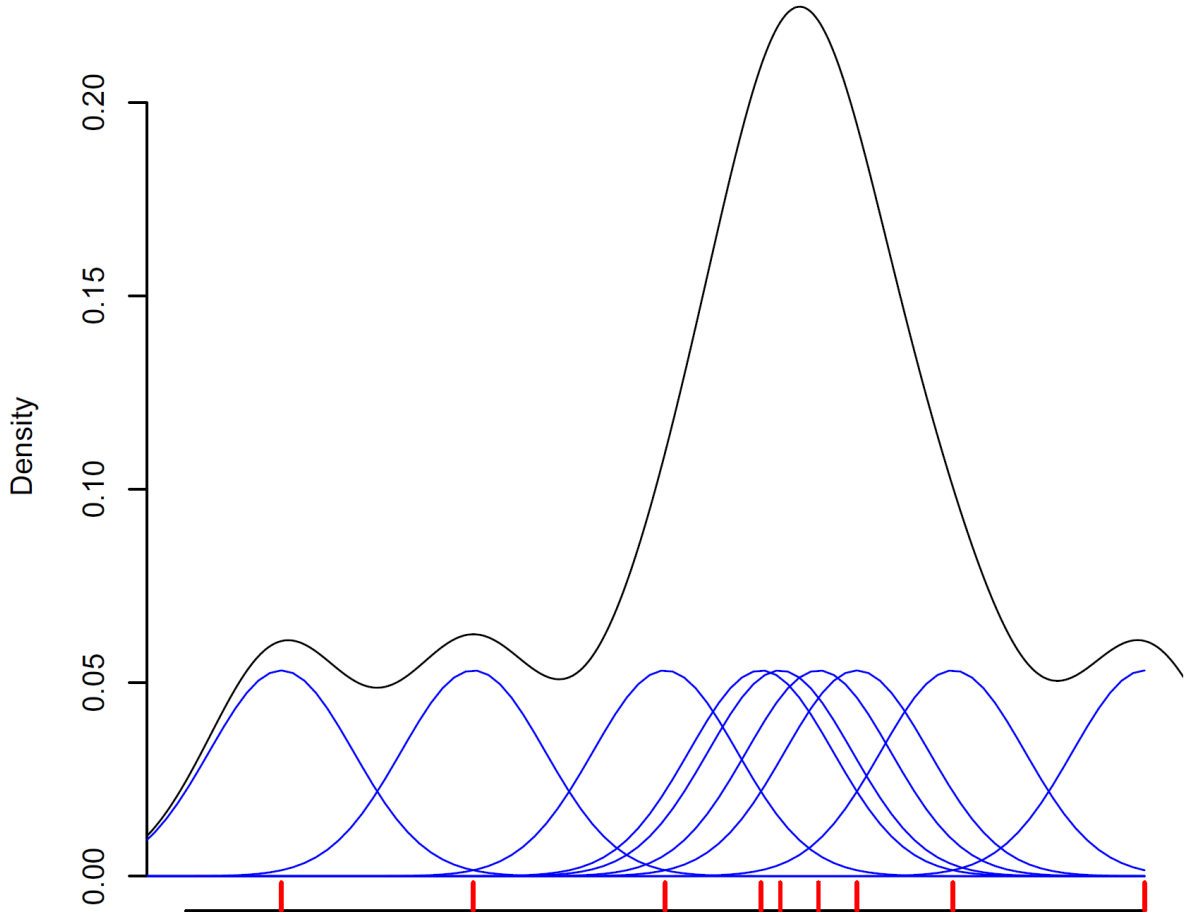
Question

Let f be an unknown mixture of k Gaussians in $\mathbb{R}^d$.
How many i.i.d. samples from f is needed to produce $\widehat{f}$ satisfying $\|f - \widehat{f}\|_1 \leq \varepsilon$?

Remarks:

1. Success probability 99%
2. Algorithm knows k
3. Focus is on sample complexity
4. Unbounded for $L_{p>1}$ or KL

Kernel density estimation



Plot: Patrick Breheny, University of Kentucky

Unfortunately, sample complexity is exponential in d.

Prior results

Question

Let f be an unknown mixture of k Gaussians in $\mathbb{R}^d$.
How many i.i.d. samples from f is needed to produce $\widehat{f}$ satisfying $\|f - \widehat{f}\|_1 \leq \varepsilon$?

$k = 1$: sample complexity $\leq O(d^2/\varepsilon^2)$
compute empirical mean and covariance (folklore)

$d = 1$: sample complexity $\leq O(k/\varepsilon^2)$
approximate by piecewise polynomials
[Chan, Diakonikolas, Servedio, Sun 2014]

Conjecture: sample complexity $\leq$ number of parameters $/\,\varepsilon^2$

Our result

Question

Let f be an unknown mixture of k Gaussians in $\mathbb{R}^d$.
How many i.i.d. samples from f is needed to produce $\widehat{f}$ satisfying $\|f - \widehat{f}\|_1 \leq \varepsilon$?

Theorem (Ashtiani, Ben-David, Harvey, Liaw, M, Plan, NeurIPS 2018)

The sample complexity is at most $\widetilde{O}(kd^2/\varepsilon^2)$,
and is at least $\Omega(kd^2/\varepsilon^2)$.

Our result

Question

Let f be an unknown mixture of k Gaussians in $\mathbb{R}^d$.
How many i.i.d. samples from f is needed to produce $\widehat{f}$ satisfying $\|f - \widehat{f}\|_1 \leq \varepsilon$?

Theorem (Ashtiani, Ben-David, Harvey, Liaw, M, Plan, NeurIPS 2018)

The sample complexity is at most $\widetilde{O}(kd^2/\varepsilon^2)$,
and is at least $\Omega(kd^2/\varepsilon^2)$.

Previous bounds:

$\widetilde{O}(kd^2/\varepsilon^4)$	[Ashtiani, Ben-David, M, AAAI 2017]
$O(k^4 d^4/\varepsilon^2)$	[Anthony and Bartlett 1999], [Devroye and Lugosi 2001]
$\Omega(kd/\varepsilon^2)$	[Suresh, Orlitsky, Acharya, and Jafarpour 2014]

Algorithm outline

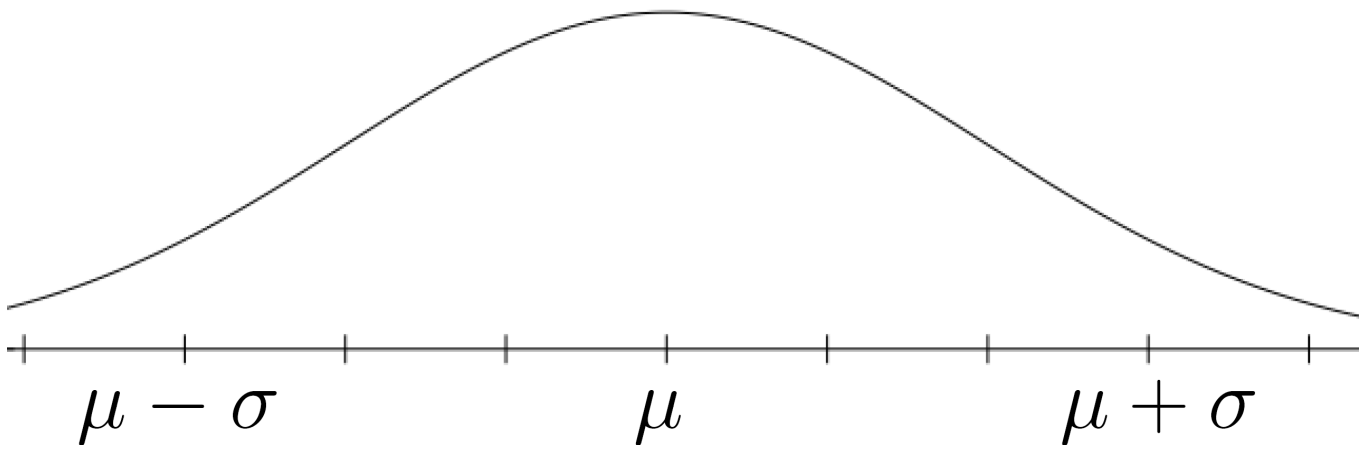
Consider the case $k = d = 1$, fix some $\varepsilon > 0$.

1. Once $m = 99/\varepsilon$ points are generated from f, there are two points approximately determining f.

Algorithm outline

Consider the case $k = d = 1$, fix some $\varepsilon > 0$.

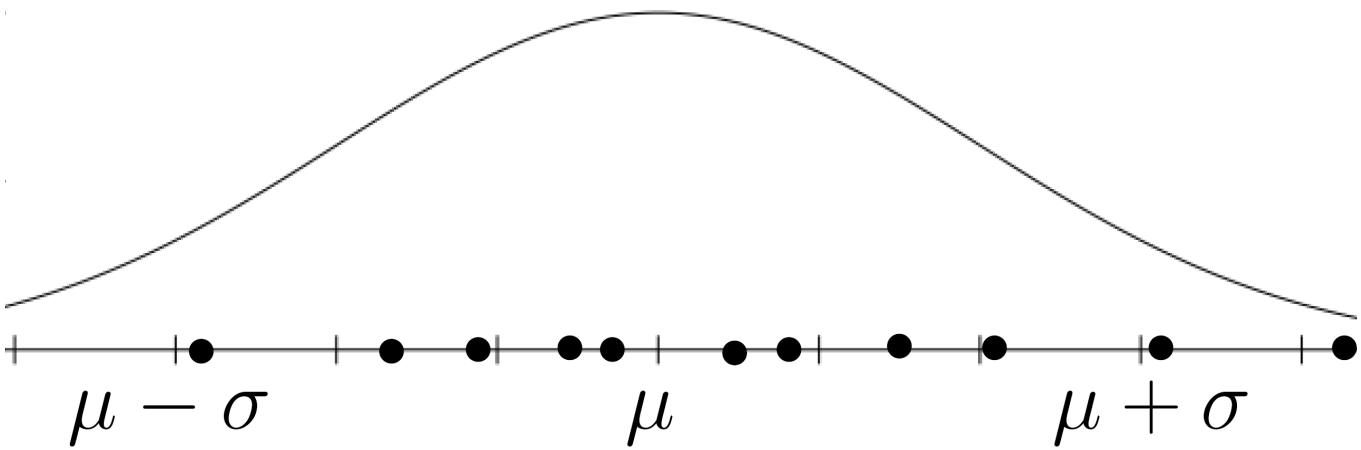
1. Once $m = 99/\varepsilon$ points are generated from f, there are two points approximately determining f.



Algorithm outline

Consider the case $k = d = 1$, fix some $\varepsilon > 0$.

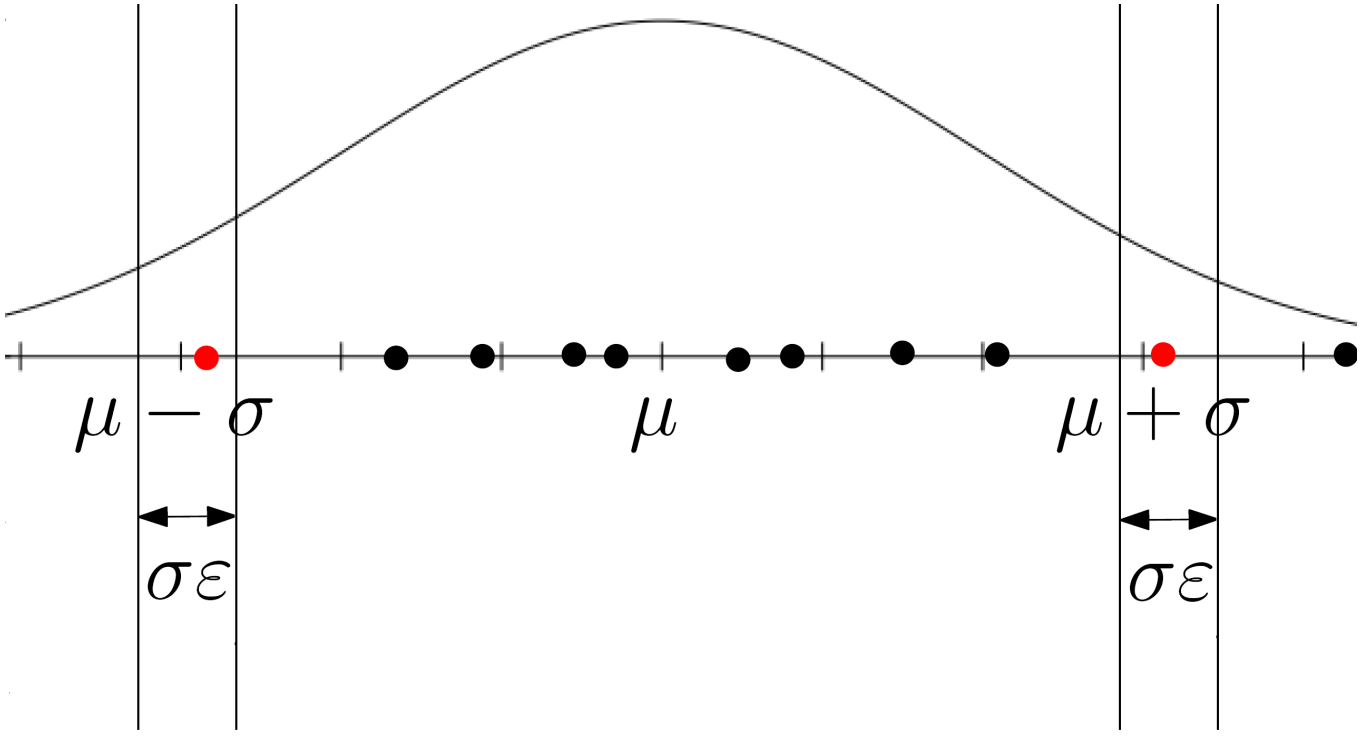
1. Once $m = 99/\varepsilon$ points are generated from f, there are two points approximately determining f.



Algorithm outline

Consider the case $k = d = 1$, fix some $\varepsilon > 0$.

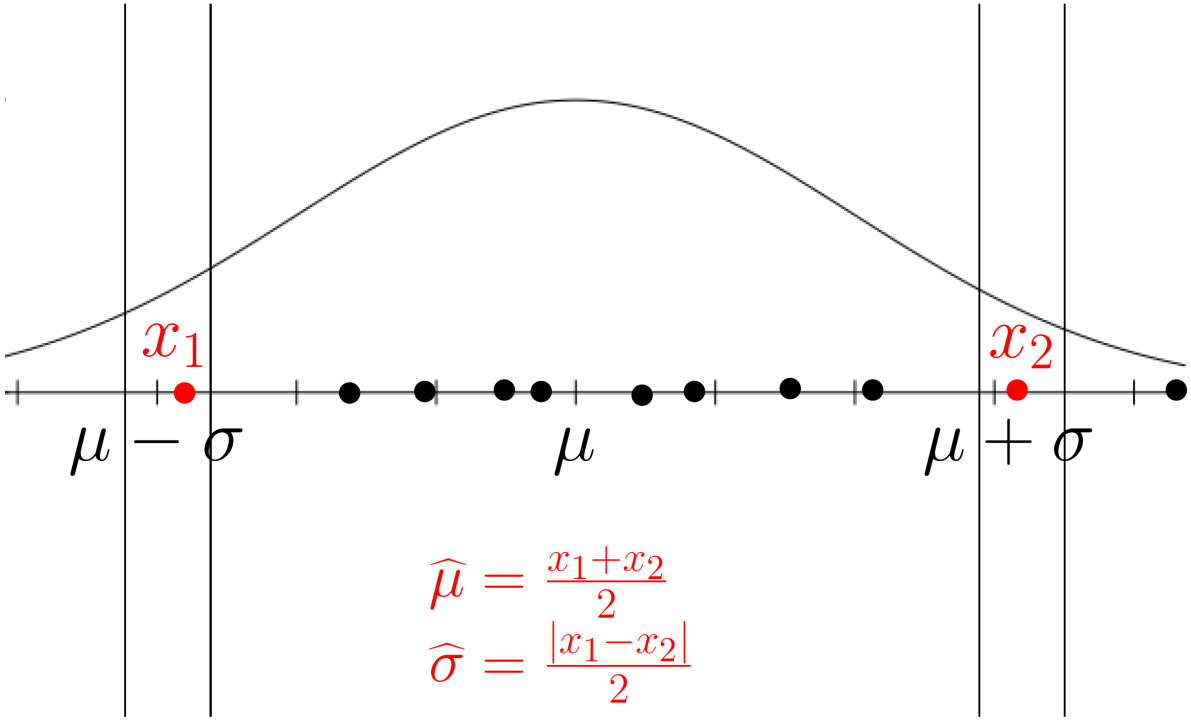
1. Once $m = 99/\varepsilon$ points are generated from f, there are two points approximately determining f.



Algorithm outline

Consider the case $k = d = 1$, fix some $\varepsilon > 0$.

1. Once $m = 99/\varepsilon$ points are generated from f, there are two points approximately determining f.



Algorithm outline

Consider the case $k = d = 1$, fix some $\varepsilon > 0$.

1. Once $m = 99/\varepsilon$ points are generated from f, there are two points approximately determining f.
2. This gives $\binom{m}{2}$ candidate distributions, at least one of them ε-close to f.
3. Take $\log(\binom{m}{2})/\varepsilon^2 = \widetilde{O}(1/\varepsilon^2)$ more samples, and output $\widehat{f}$ = the candidate closest to the empirical distribution.

Algorithm outline

Consider the general case, $k, d \geq 1$, fix some $\varepsilon > 0$.

1. Once $m = 99kd^2 \log^2(d/\varepsilon)$ points are generated, there are $p = kd^2 \log^2(d/\varepsilon)$ points approximately determining f.
2. This gives $\binom{m}{p}$ candidate distributions, at least one of them ε-close to f.
3. Take $\log(\binom{m}{p})/\varepsilon^2 = \widetilde{O}(kd^2/\varepsilon^2)$ more samples, and output $\widehat{f}$ = the candidate closest to the empirical distribution.

Main result

Theorem (Ashtiani, Ben-David, Harvey, Liaw, M, Plan, NeurIPS 2018)

Sample complexity for learning mixtures of k Gaussians in $\mathbb{R}^d$ is bounded by $\widetilde{O}(kd^2/\varepsilon^2)$, and this is tight.

A nearly-tight characterization for the amount of data needed to learn a Gaussian mixture model.

To reach accuracy ε, need kd^2/ε^2 samples

$\Updownarrow$

With n data points, can get accuracy $\sqrt{kd^2/n}$

Our algorithm is robust (agnostic learning).

My research goal in distribution learning

Distribution learning problem

Given data, estimate the underlying distribution.

Fundamental problem in unsupervised machine learning: anomaly detection, generative models, PCA, clustering.

My research goal

Design sample-efficient, time-efficient algorithms with mathematical guarantees that are robust against noise.

My research goal in distribution learning

Distribution learning problem

Given data, estimate the underlying distribution.

Fundamental problem in unsupervised machine learning: anomaly detection, generative models, PCA, clustering.

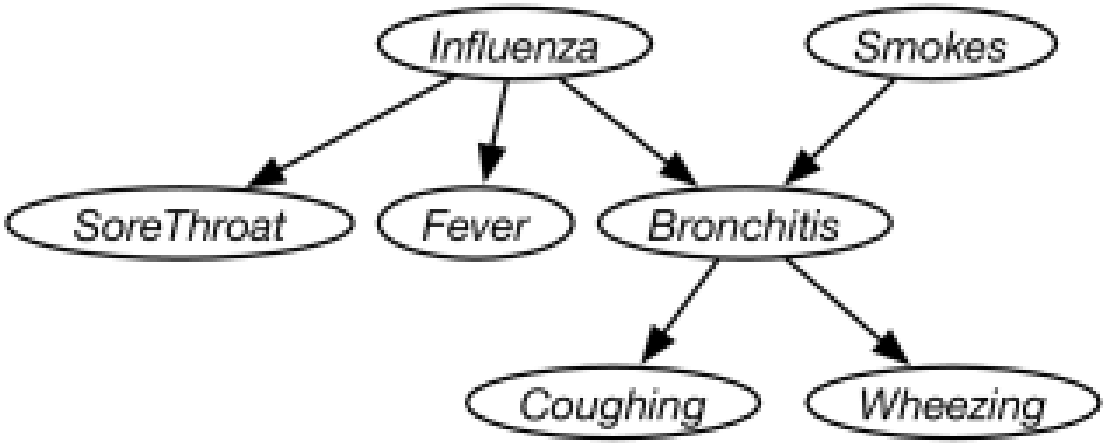
My research goal

Design sample-efficient, time-efficient algorithms with mathematical guarantees that are robust against noise.

Research direction 1

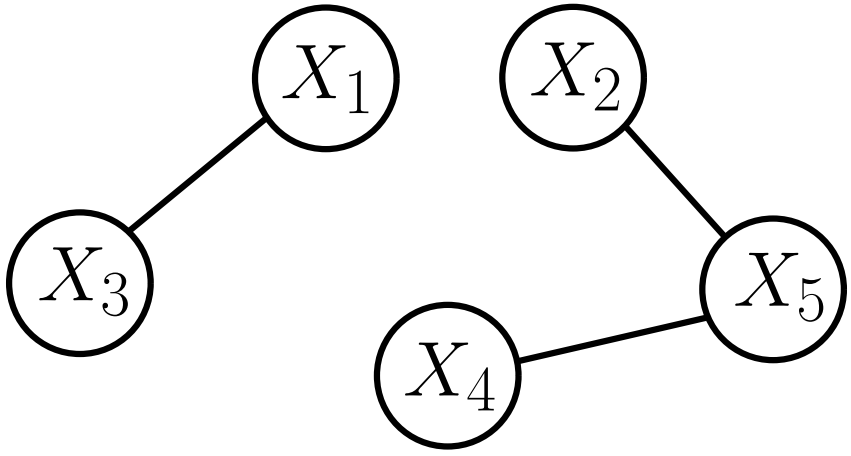
What's the amount of data needed for learning other distribution classes?

Probabilistic graphical models



Poole, Mackworth 2017

Probabilistic graphical models



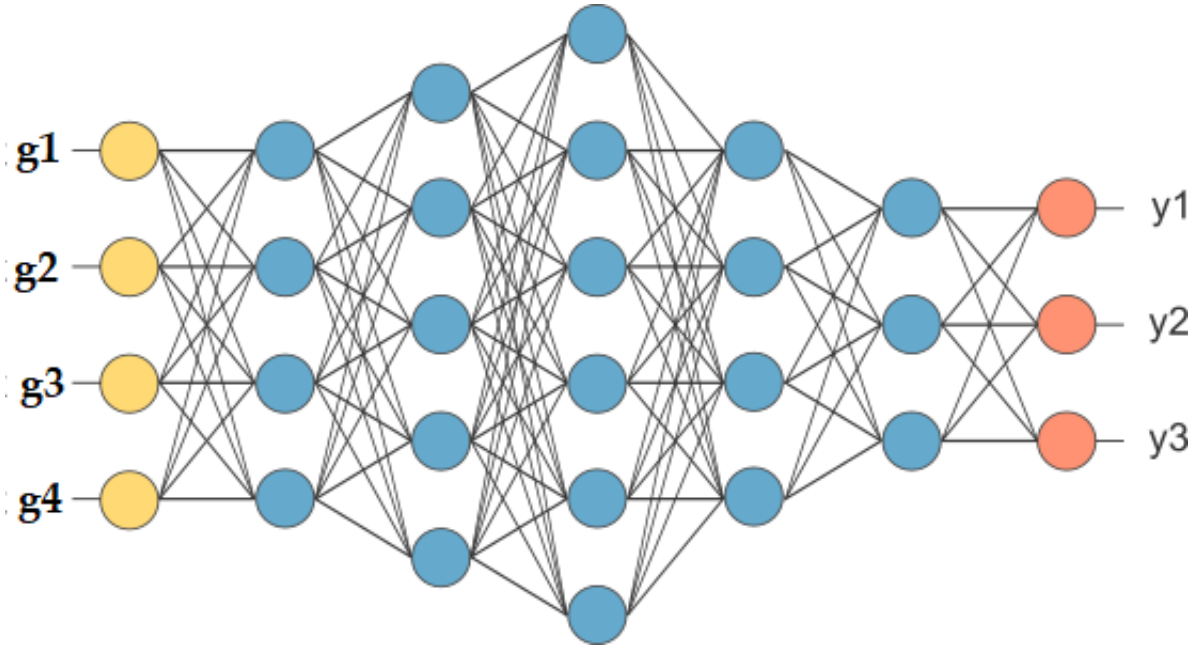
The Ising model. Each $X_i \in \{-1, +1\}$ and

$$\mathbb{P}[X_1 = x_1, \ldots, X_d = x_d] \propto \exp\left(\sum_{ij \in E(G)} w_{i,j} x_i x_j\right)$$

Theorem (Devroye, M, Reddad 2018)

The sample complexity for learning Ising models on a graph G is $O(|E(G)|/\varepsilon^2)$, and this is tight.

Distributions generated by neural networks



[Karras, Aila, Laine, and Lehtinen 2017]

Other research directions

Research direction 2: computational complexity

Which distribution classes are learnable in polynomial time?

E.g. polynomial time algorithm for mixtures of Gaussians?

Research direction 3: model selection

Can we learn the distribution class itself from data?

Outline

1. Distribution learning

2. Online learning

What is online learning?

Online learning framework. Data not revealed at once, but in an online manner.

Example: choosing a restaurant to eat each night

The multi-arm bandit problem

The multi-arm bandit model

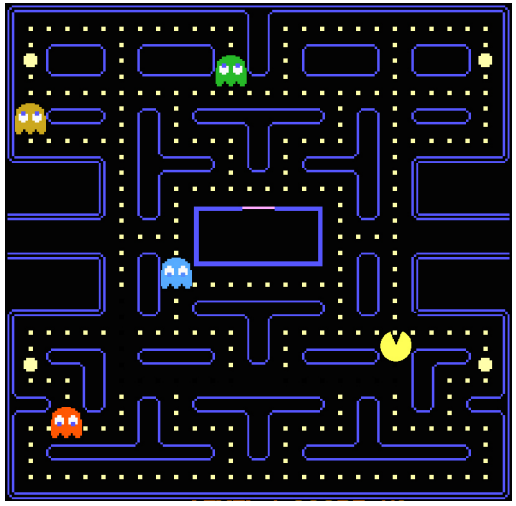
1. A multi-round single player game, a finite set of actions.
2. In each round the player chooses one of the actions and receives a (stochastic) reward.
3. The rewards of each action come from some unknown distribution.

The multi-arm bandit problem

The multi-arm bandit model

1. A multi-round single player game, a finite set of actions.
2. In each round the player chooses one of the actions and receives a (stochastic) reward.
3. The rewards of each action come from some unknown distribution.

Exploration/exploitation trade-off (central in reinforcement learning).



The multi-arm bandit problem

The multi-arm bandit model

1. A multi-round single player game, a finite set of actions.
2. In each round the player chooses one of the actions and receives a (stochastic) reward.
3. The rewards of each action come from some unknown distribution.

Oracle's strategy. In all rounds, choose the action with the largest expected reward.

Regret of a learning algorithm: difference between algorithm's total reward and the oracle's total reward.

The multi-arm bandit problem

known results

$T =$ number of rounds.

Theorem (Lai and Robbins 1985, Auer, Cesa-Bianchi, Fischer 1998)

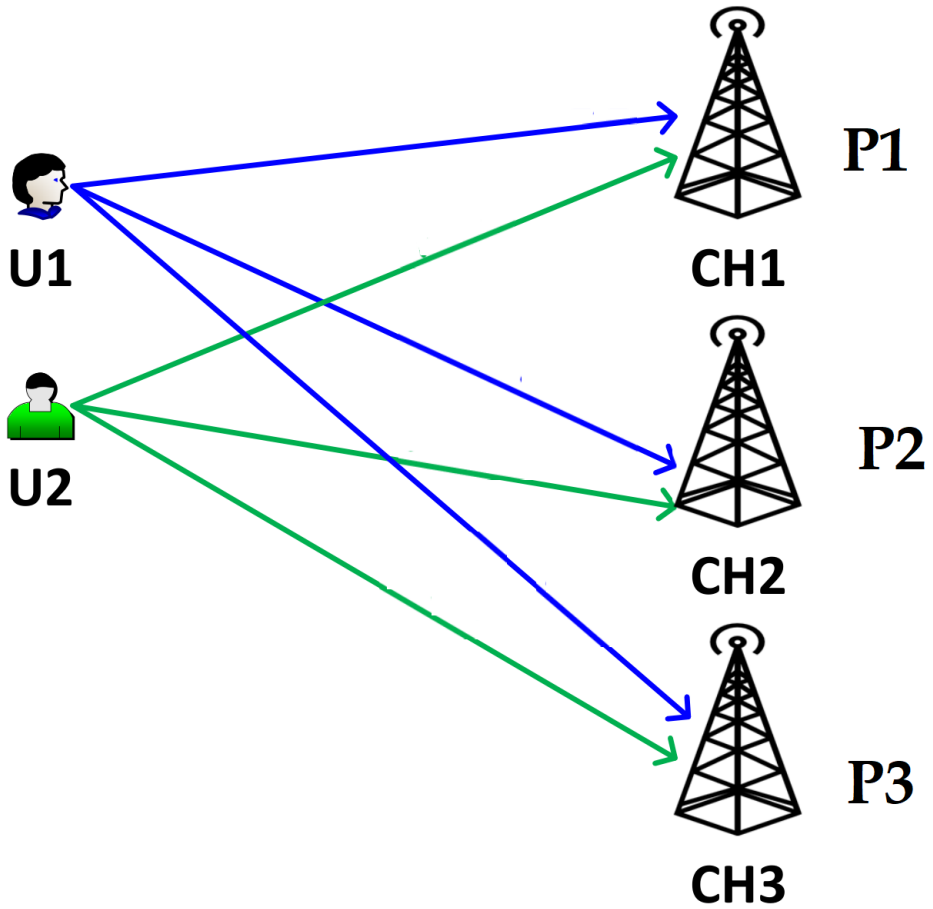
If each single reward $\in [0,1]$, *there is an algorithm with regret* $O(\log T)$, *and this is tight.*

Per round suboptimality $\longrightarrow \frac{\log T}{T}$

Upper confidence bound (UCB) algorithm.

Multiplayer multi-arm bandits

Opportunistic spectrum access in cognitive radios



Two feedback models can be considered

Multiplayer multi-armed bandits

Our results

Harder than single player (players have to avoid collisions)

Question: is $O(\log T)$ system regret still possible?

If players observe collisions, $O(\log T)$ regret is attainable [Liu and Zhao, 2009].

Multiplayer multi-armed bandits

Our results

Harder than single player (players have to avoid collisions)

Question: is $O(\log T)$ system regret still possible?

If players observe collisions, $O(\log T)$ regret is attainable [Liu and Zhao, 2009].

Theorem (Lugosi, M 2018)

In the harder setup that players do not observe collisions, there still exists a polynomial-time algorithm with regret $O(\log T)$.

Multiplayer multi-armed bandits

Algorithm outline

Theorem (Lugosi, M 2018)

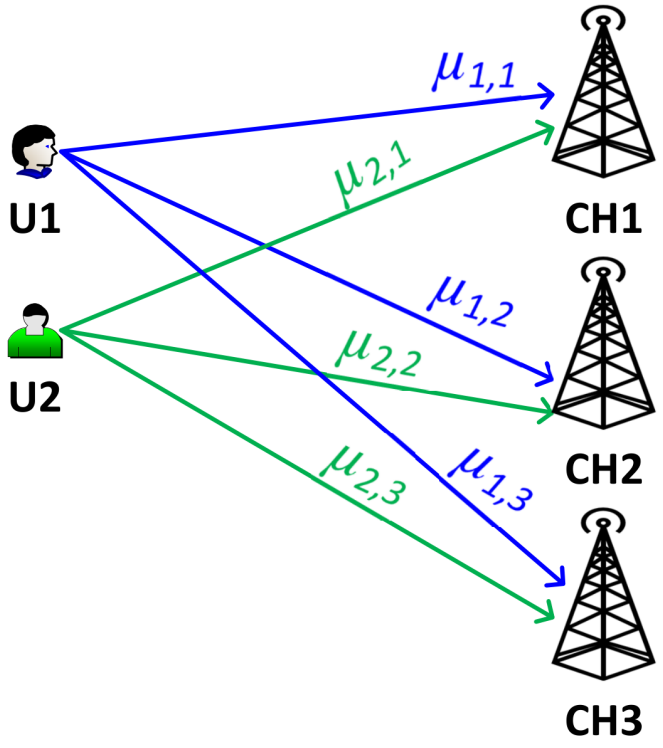
In the harder setup that players do not observe collisions, there exists polynomial-time algorithm with regret $O(\log T)$.

p players, rewards $\in [0, 1]$

1. Exploration phase. Play actions uniformly at random and obtain estimates for the expected rewards.
2. Determine the p best actions.
3. Musical Chairs phase. Play one of the best actions. If positive reward, play that action forever; else try again.

Multiplayer multi-armed bandits

Inhomogeneous variant

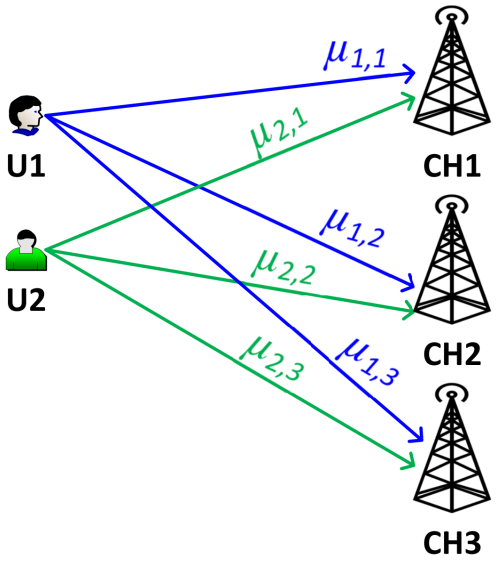


Cooperative game-theoretic situation.

	channel 1	channel 2	channel 3
Player 1	1	0.9	0.2
Player 2	1	0.1	0.3

Multiplayer multi-armed bandits

Inhomogeneous variant



Question: is $O(\log T)$ system regret still possible?

$O(\log^2 T)$ regret is attainable [Bistritz and Leshem 2018]

Theorem (Kaufmann, M 2019)

There is a polynomial-time algorithm with regret $O(\log T)$.

Inhomogeneous variant

Algorithm outline

Theorem (Kaufmann, M 2019)

There is a polynomial-time algorithm with regret $O(\log T)$.

We assume the players can observe collisions. They can use forced collisions for implicit communication.

1. A leader is selected.
2. Each player estimates the expected rewards of each action.
3. The players send their estimations to the leader.
4. Leaders either finds the optimal matching, or asks for finer estimations.

Research agenda in online learning

Online learning problem

Data to the learning problem is not revealed at once, but is received sequentially. Can we compete against batch algorithm that sees all the data at once?

Connections with game theory and optimization.
At the core of reinforcement learning.

My research goal

Design fast algorithms with provably small regret.

Past and current research

- ✓ Network formation (creation) games
  [ACM Transactions on Algorithms 2015]
- ✓ Randomized rumour spreading, or gossip protocols
  [DISC 2014, PODC 2015, APPROX-RANDOM 2017]
- ✓ Load balancing protocols [IPDPS 2017]
- ✓ Theory of deep learning [COLT 2017, Journal of Machine Learning Research 2019]

Current interests: mathematical foundations of ML

Summary of research agenda

Distribution learning

Design sample-efficient, time-efficient algorithms with mathematical guarantees that are robust against noise.

Online learning

Design fast algorithms with provably small regret.

Summary of research agenda

Distribution learning

Design sample-efficient, time-efficient algorithms with mathematical guarantees that are robust against noise.

Online learning

Design fast algorithms with provably small regret.

THANKS FOR LISTENING!!

An application of distribution learning

detecting breast cancer

- ✓ Training data consists of normal (non-cancerous) X-ray images.
- ✓ Probability density function f is learned from the data.
- ✓ When a new input x is presented, a high value for $f(x)$ indicates a normal image, while a low value indicates a novel input, which might be characteristic of an abnormality.

[Tarassenko, Hayton, Cerneaz, Brady: Novelty detection for the identification of masses in mammograms]

High-dimensional Gaussians formula

Multivariate normal distribution:

$$\mathcal{N}_{\mu,\Sigma}(x) = \frac{\exp\left(-\frac{1}{2}(x-\mu)^{\mathsf{T}}\Sigma^{-1}(x-\mu)\right)}{(2\pi)^{k/2}\sqrt{\det(\Sigma)}} \qquad \text{for } x \in \mathbb{R}^d$$

$X \sim \mathcal{N}_{\mu,\Sigma}$: $\mathbb{E}[X] = \mu \in \mathbb{R}^d, \mathbb{E}\left[(X-\mu)(X-\mu)^{\mathsf{T}}\right] = \Sigma \in \mathbb{R}^{d\times d}$

Mixture of k Gaussians in $\mathbb{R}^d$: $\sum_{i=1}^{k} w_i \mathcal{N}_{\mu_i,\Sigma_i}$
w_i are mixture weights (component weights), $w_i \geq 0$ and $\sum w_i = 1$
Parameters of the model: $(w_i, \mu_i, \Sigma_i)_{i=1}^{k}$: $\Theta(kd^2)$ parameters

Our results on VC-dimension of neural networks

Theorem (Bartlett, Harvey, Liaw, M, COLT 2017, JMLR 2019)

Let p be the number of parameters, ℓ the number of layers.
Then VC-dim $\leq O(p\ell \log p)$.
There exist networks with VC-dim $\geq \Omega(p\ell \log(p/\ell))$.

$O(p^2)$	[Goldberg and Jerrum'95]
$O(p\ell^2 + p\ell \log p)$	[Bartlett, Maiorov, Meir'98]
$\Omega(p \log p)$	[Maass'94]
$\Omega(p\ell)$	[Bartlett, Maiorov, Meir'98]

Our results on VC-dimension of neural networks

Theorem (Bartlett, Harvey, Liaw, M, COLT 2017, JMLR 2019)

Let p be the number of parameters, ℓ the number of layers.
Then VC-dim $\leq O(p\ell \log p)$.
There exist networks with VC-dim $\geq \Omega(p\ell \log(p/\ell))$.

Theorem (Vapnik and Chervonenkis 1971, Dudley 1978)

Given a training set of size m, generalization error is $\leq C\sqrt{\text{VC-dim}/m}$.

Upper bound for training data size needed to train a neural network: $\sqrt{p\ell \log p/m} \leq \varepsilon \Rightarrow m \geq p\ell \log p/\varepsilon^2$

Carries over to real-output neural networks as well (regression).