Convergence and Dynamical Behavior of the ADAM Algorithm for Non Convex Stochastic Optimization

Anas Barakat, Pascal Bianchi

LTCI, Télécom Paris, Institut polytechnique de Paris

Machine Learning in the Real World, October 2nd 2019

Optimization in Deep Learning

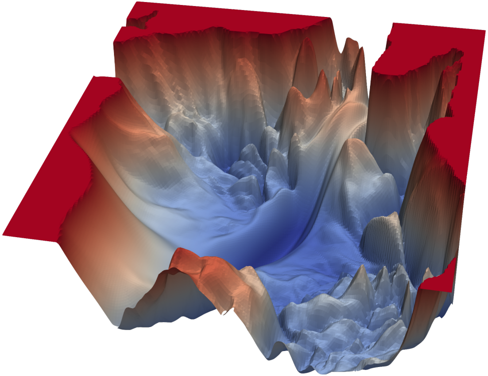


Figure 1: Visualization of a loss landscape (VGG-56 on CIFAR-10)
https://www.cs.umd.edu/ tomg/projects/landscapes/

Li et al., Visualizing the Loss Landscape of Neural Nets, NeurIPS 2018

Problem statement

Problem

$$\min_x F(x) := \mathbb{E}(f(x,\xi)) \quad \text{w.r.t.} \quad x \in \mathbb{R}^d$$

Assumptions

- $f(\,.\,,\xi)$: **nonconvex** differentiable function
- regularity assumptions on f (smoothness, coercivity of F, etc.)
- $(\xi_n : n \geq 1)$: iid copies of r.v ξ revealed online

ADAM : an adaptive algorithm

[Kingma and Ba, 2015]

- Regime : **constant step size** $\gamma > 0$.

Algorithm 1 $\mathrm{ADAM}\ (\gamma, \alpha, \beta, \varepsilon)$

$x_0 \in \mathbb{R}^d, m_0 = 0,\ v_0 = 0,\ \gamma > 0,\ \varepsilon > 0,\ (\alpha, \beta) \in [0, 1)^2.$

for $n \geq 1$ **do**

$\quad m_n = \alpha m_{n-1} + (1 - \alpha)\nabla f(x_{n-1}, \xi_n)$

$\quad v_n = \beta v_{n-1} + (1 - \beta)\nabla f(x_{n-1}, \xi_n)^2$

$\quad \hat{m}_n = \frac{m_n}{1-\alpha^n}$

$\quad \hat{v}_n = \frac{v_n}{1-\beta^n}$

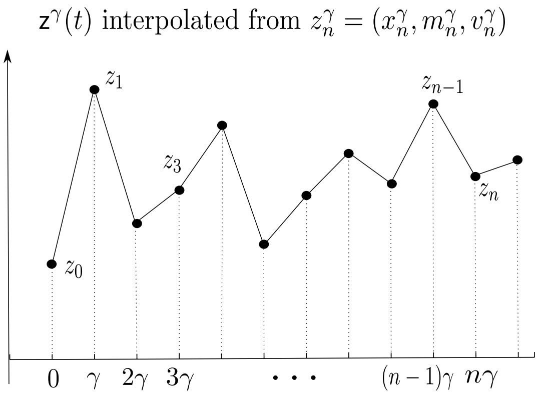
$\quad x_n = x_{n-1} - \frac{\gamma}{\varepsilon + \sqrt{\hat{v}_n}} \hat{m}_n$

$\quad x_n = x_{n-1} - \gamma \nabla f(x_{n-1}, \xi_n)$ (SGD for comparison)

end for

From Discrete to Continuous Time

The ODE Method [Ljung, 1977, Kushner and Yin, 2003]



Continuous Time System

similar approach to [Su, Boyd and Candès, 2016]

Non autonomous ODE

If $z(t) = (x(t), m(t), v(t))$,

$$\dot{z}(t) = h(t, z(t)) \tag{ODE}$$

Theorem (Convergence)

$$\lim_{t\to\infty} \mathrm{d}(x(t), \nabla F^{-1}(\{0\})) = 0\,.$$

$$c_1(t)\,\ddot{x}(t) + c_2(t)\,\dot{x}(t) + \nabla F(x(t)) = 0\,,$$

- 2nd vs 1st order: acceleration (even if oscillations).
- Escaping local traps (saddle points)

Long run convergence of the ADAM iterates

- No a.s convergence : regime $n \to \infty$ then $\gamma \to 0$

Theorem (ergodic convergence of the ADAM iterates)

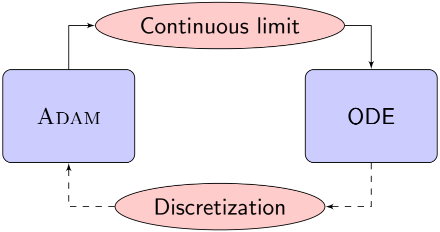
Let $x_0 \in \mathbb{R}^d$, $\gamma > 0$, $(z_n^\gamma : n \in \mathbb{N})$, $z_0^\gamma = (x_0, 0, 0)$. Under the same assumptions and :

- **Stability assumption:** $\sup_{n,\gamma} \mathbb{E}\|z_n^\gamma\| < \infty$.

Then, for all $\delta > 0$,

$$\lim_{\gamma \downarrow 0} \limsup_{n \to \infty} \frac{1}{n} \sum_{k=1}^{n} \mathbb{P}(\mathrm{d}(x_n^\gamma, \nabla F^{-1}(\{0\})) > \delta) = 0 . \tag{1}$$

Thank you for your attention



For more details: submitted article, available on arXiv.

AB, P. Bianchi. *Convergence and Dynamical Behavior of the ADAM Algorithm for Non Convex Stochastic Optimization.*

Utility/Privacy Trade-off through the lens of Optimal Transport

Etienne Boursier [1] Vianney Perchet [2, 3]

[1] ENS Paris-Saclay, CMLA

[2]Criteo AI Lab, Paris

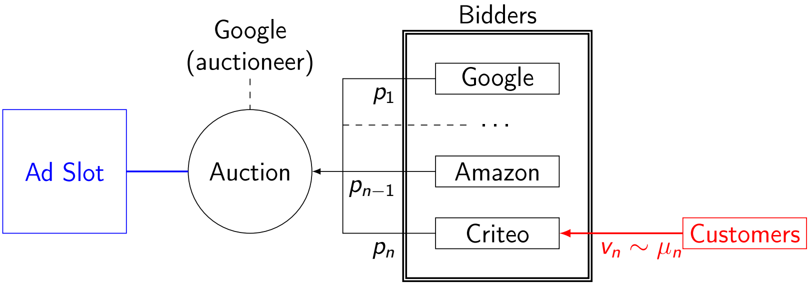
[3]ENSAE Paris

MLITRW '19, Criteo Paris

An economic motivation

Online repeated auctions

Ad slot valued v. Bid $p \implies$ auctioneer infers v.
Auctioneer's revenue $\nearrow$ while bidder's utility $\searrow$ when v public.



Online advertisement auction system

Bidder's goal: short term utility **and** hide value distribution μ_n

Toy example

Player: minimizes utility loss

$$\min_{x \in \mathcal{X} \subset \mathbb{R}^d} x^\top y_k$$

y_k depends on **private type** $k \in \{1, \ldots, K\}$ with prior $p_0 \in \Delta_K$.
Adversary: observes x and infers k

Program in previous literature[1]:

$$\min_{\mu_1, \ldots, \mu_K} \sum_{k=1}^{K} p_0(k)\ \mathbb{E}_{x \sim \mu_k}\left[x^\top y_k\right]$$
$$\text{such that } \mathbb{E}[KL(p_x, p_0)] \leq \varepsilon$$

[1] Eilat, R., Eliaz, K., and Mu, X. (2019). Optimal privacy-constrained mechanisms

General formulation of the problem

Our general program:

$$\inf_{\substack{\gamma \in \mathcal{P}(\mathcal{X}\times\mathcal{Y}) \\ \pi_2 \# \gamma = p_0}} \int_{\mathcal{X}\times\mathcal{Y}} \left(c(x,y) + \lambda\, D(p_x, p_0)\right) \mathrm{d}\gamma(x,y) \quad \text{(P-OPT)}$$

- type $y \sim p_0 \in \mathcal{P}(\mathcal{Y})$
- $\pi_2 \# \gamma(A) = \gamma(\mathcal{X} \times A)$
- $c =$ utility loss ; $D =$ privacy loss (e.g. KL)

Theoretical results

Theorem (Convexity)

If D is an f-divergence, then (P-OPT) *is convex in γ.*

$\rightarrow$ (P-OPT) easy for finite $\mathcal{X}$ and $\mathcal{Y}$.

Theorem (Finite prior support)

If $|\mathrm{supp}(p_0)| = K$, for all $\varepsilon > 0$, we can look for a solution of (P-OPT) *with support of size $K(K+2)$.*

$\rightarrow$ finite dimension ☺ but not jointly convex ☹

Sinkhorn divergence minimization

Definition (Sinkhorn divergence)

$$\mathrm{OT}_{c,\lambda}(\mu,\nu) = \inf_{\gamma \in \Pi(\mu,\nu)} \int c \mathrm{d}\gamma + \lambda \int \log\left(\frac{\mathrm{d}\gamma}{\mathrm{d}\mu \mathrm{d}\nu}\right) \mathrm{d}\gamma$$

- entropic regularization $\implies$ fast OT distances approximation[2]

If D=KL, (P-OPT) equivalent to

$$\inf_{\mu \in \mathcal{P}(\mathcal{X})} \mathrm{OT}_{c,\lambda}(\mu, p_0).$$

[2] Cuturi, M. (2013). Sinkhorn distances: Lightspeed computation of optimal transport

Recap

- utility-privacy trade-off motivated by economic mechanisms
- general regularized problem
- convexity + finiteness under mild assumptions
- benefit from Sinkhorn divergence
- find our simulations in the paper

Slides, code and paper at `eboursier.github.io`

Thank you !

Bayesian computation and machine learning

Nicolas Chopin (ENSAE, IPP)

PAC-Bayes

Uses as an estimator the expectation of pseudo-posterior:

$$p(x|y) \propto p(x) \exp\{-\gamma R(x, y)\}$$

where $R(x, y)$ is the empirical risk, for parameter x and data y.

How to compute this expectation?

1. Fast variational approximation: but can you we obtain the same non-asymptotic bounds? See Alquier, Ridgway and C. (2016, JMLR).
2. Monte Carlo methods: isn't that slow? not if you do it right, e.g. Sequential Monte Carlo (Ridgway et al, NIPS, 2014).

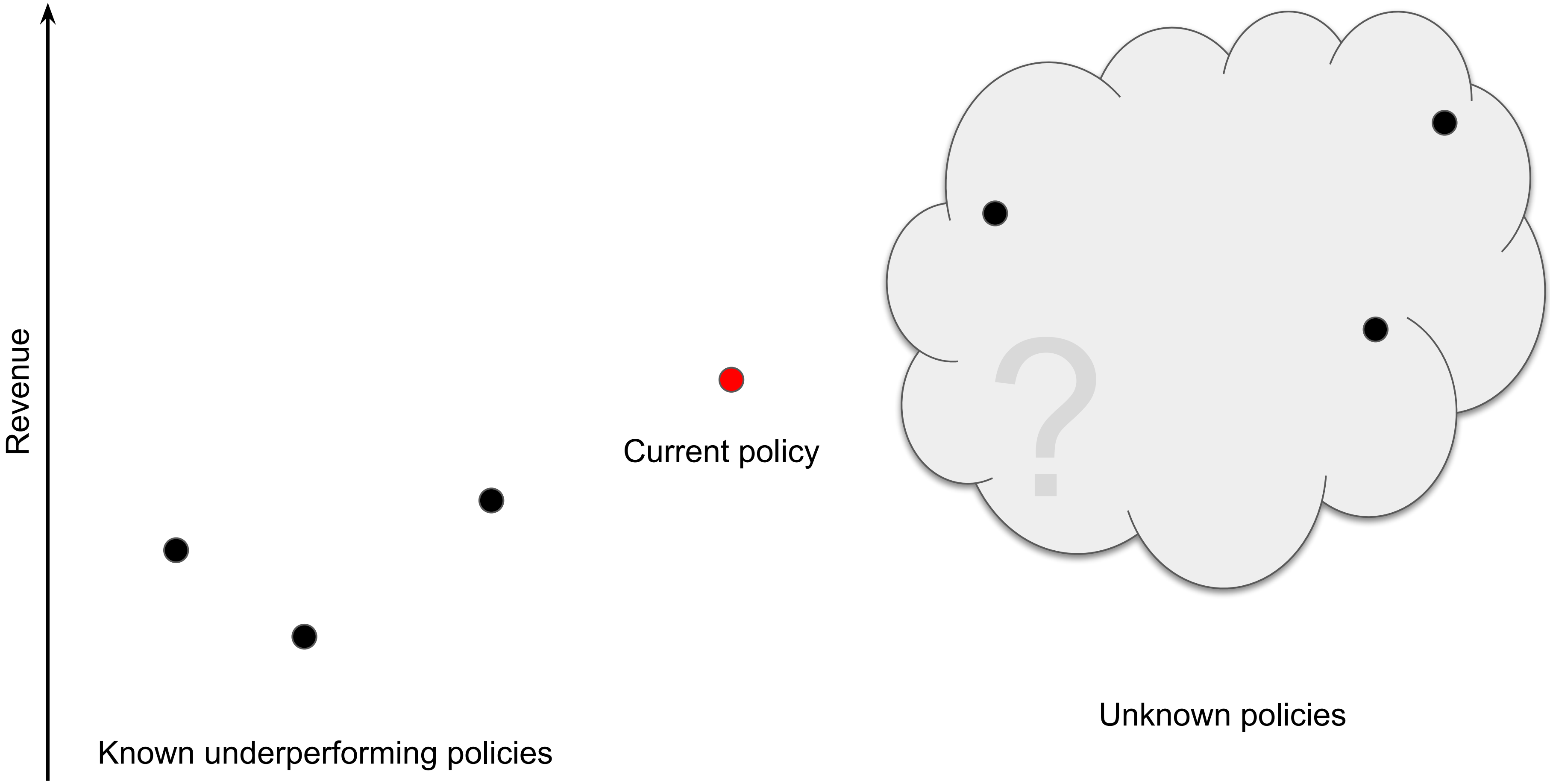
Other applications of Bayesian computation

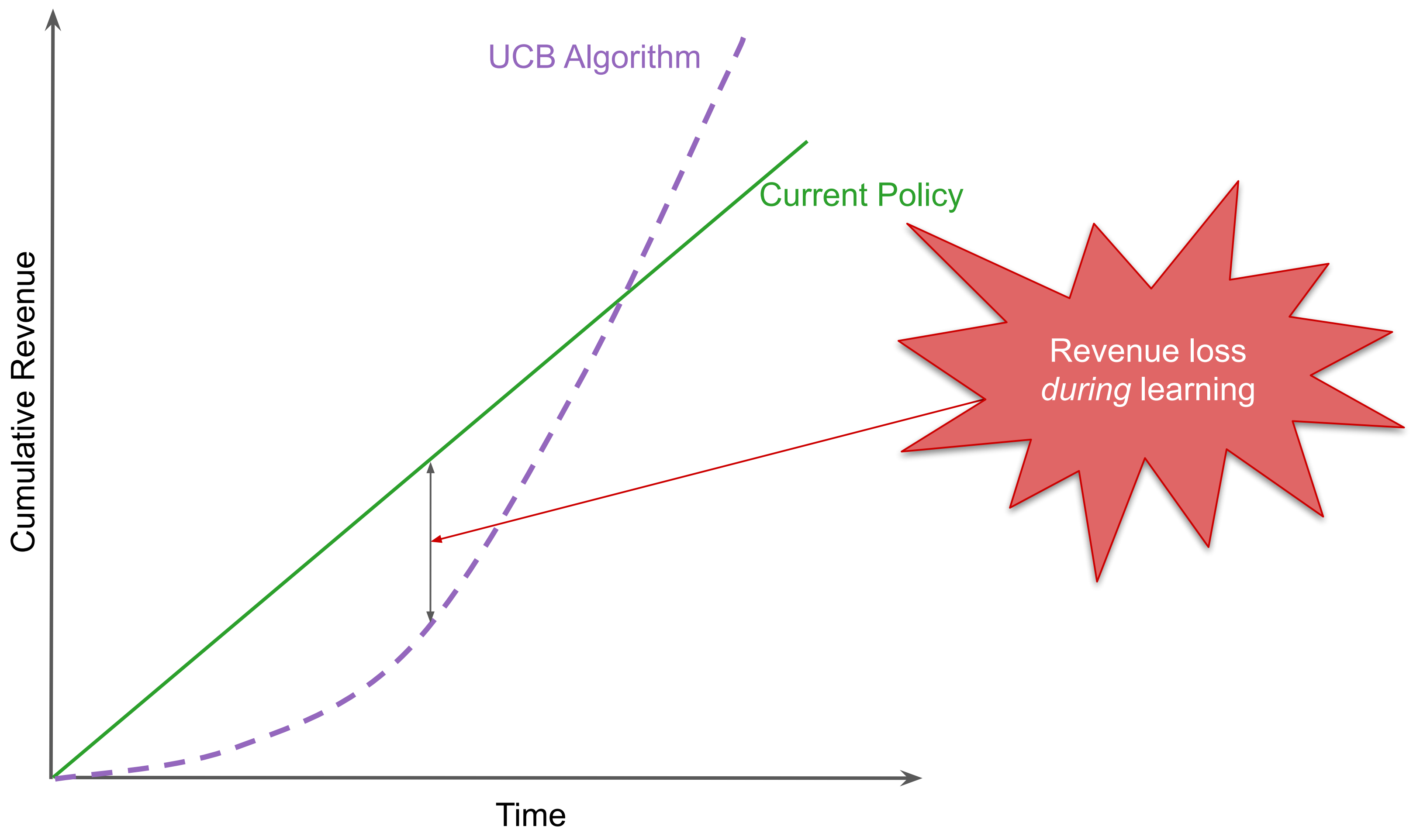
1. Probabilistic machine learning.
2. Sequential learning: use Sequential Monte Carlo?
3. Non-convex optimisation

Improved Algorithms for Conservative Exploration in Bandits

Evrard Garcelon, Mohammad Ghavamzadeh, Alessandro Lazaric and Matteo Pirotta

Facebook AI Research


Revenue
Current policy
Unknown policies
Known underperforming policies


UCB Algorithm
Current Policy
Revenue loss
during learning
Cumulative Revenue
Time

Problem: How to learn an optimal policy without sacrificing much revenue?

(aka: how to perform exploration in a ***conservative*** *way?)*

Conservative Condition

Should hold *uniformly* in time

Mean revenue of current policy

$$\boxed{\forall t > 0},\quad \boxed{\mathbb{E}\left(\sum_{l=1}^{t} r_{l,a_l}\right)} \geq \boxed{(1-\alpha)}\,\boxed{t\ \mu_b}$$

Mean *revenue* of the learning algorithm

Controls maximum revenue lost during learning

Previous Work:

- ❏ Theoretically optimal algorithms for conservative exploration (CUCB) (Wu et al. 2016, Kazerouni et al. 2017)

Contributions:

- ➔ *Improved* empirical performance in multi-armed and linear bandit (CUCB2)
- ➔ Novel *relaxed* conservative condition

CUCB *(previous algorithm)*

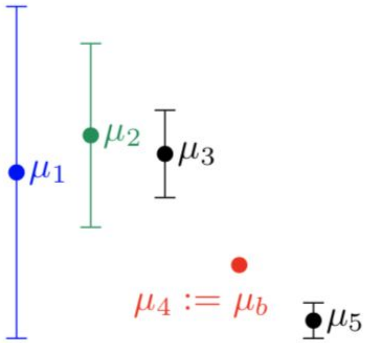
- Two phase algorithm
    a. Computes optimistic arm
    b. Checks a lower bound on the total revenue

=> impacts empirical performance!

CUCB2 *(our algorithm)*

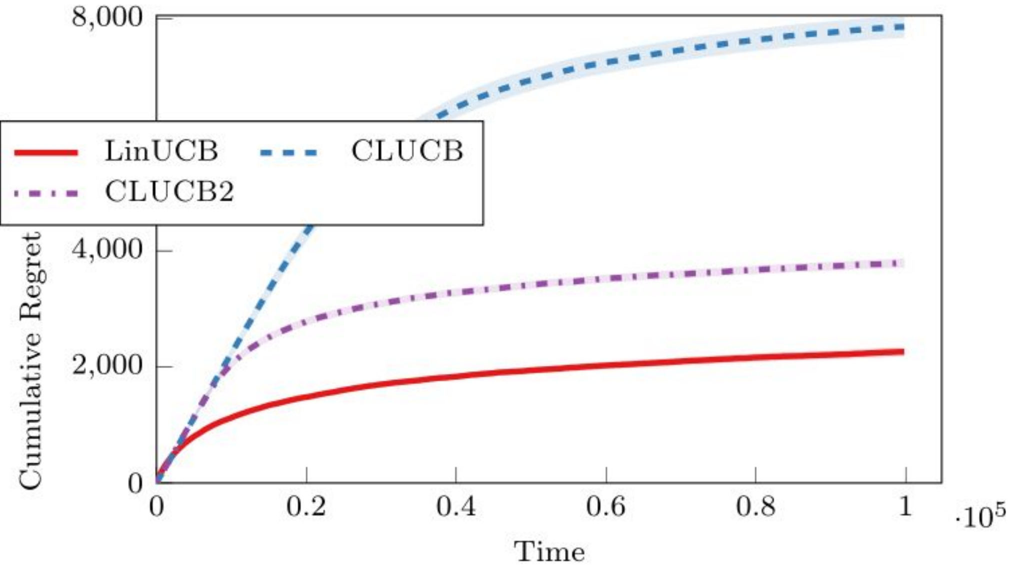
- Computes set of safe arms
- Plays the optimistic arm among safe arms

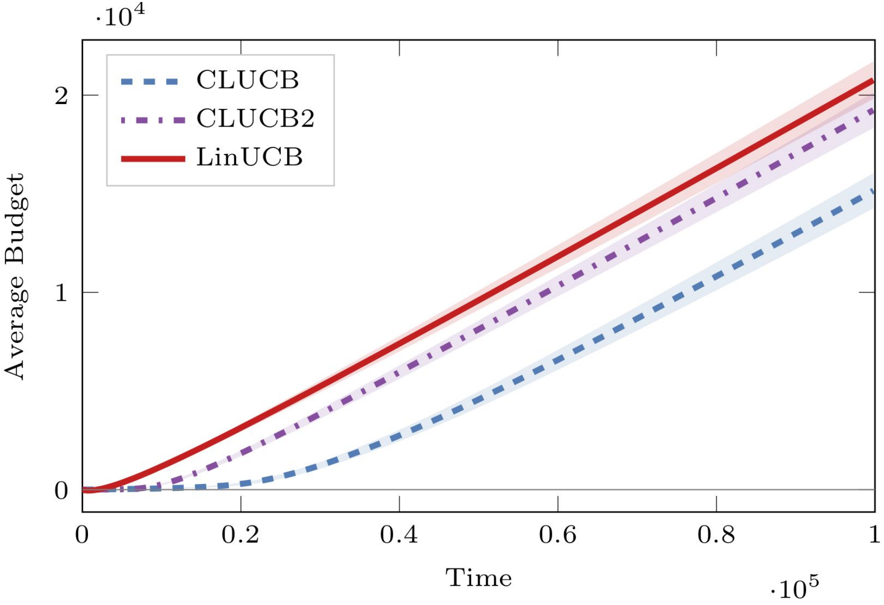
=> same regret but **better** performance!



Example: CUCB approach is suboptimal

Jester Jokes Dataset (Goldberg et al. 2001)





- Cold start problem
- Linear features

A PAC-Bayes perspective on binary-activated deep neural networks

Benjamin Guedj
`https://bguedj.github.io`

MLRW #5, Criteo
October 2, 2019

Context

Context

- Learning is to be able to **generalise**!

Context

- Learning is to be able to **generalise**!
- **PAC-Bayes** has been successfully used to analyse and understand generalisation abilities of machine learning algorithms.

Context

- Learning is to be able to **generalise**!

- **PAC-Bayes** has been successfully used to analyse and understand generalisation abilities of machine learning algorithms.
  - G., "A Primer on PAC-Bayesian Learning", invited for publication in the Proceedings of the French Mathematical Society, `https://arxiv.org/abs/1901.05353`
  - G. & Shawe-Taylor, "A Primer on PAC-Bayesian Learning", **ICML 2019 tutorial** `https://bguedj.github.io/icml2019/index.html`

Context

- Learning is to be able to **generalise**!
- **PAC-Bayes** has been successfully used to analyse and understand generalisation abilities of machine learning algorithms.
  - G., "A Primer on PAC-Bayesian Learning", invited for publication in the Proceedings of the French Mathematical Society, `https://arxiv.org/abs/1901.05353`
  - G. & Shawe-Taylor, "A Primer on PAC-Bayesian Learning", **ICML 2019 tutorial** `https://bguedj.github.io/icml2019/index.html`
- Most PAC-Bayes generalisation bounds are **computable** tight upper bounds on the population error, *i.e.* an estimate of the error on any unseen future data.

Context

- Learning is to be able to **generalise**!

- **PAC-Bayes** has been successfully used to analyse and understand generalisation abilities of machine learning algorithms.
  - G., "A Primer on PAC-Bayesian Learning", invited for publication in the Proceedings of the French Mathematical Society, `https://arxiv.org/abs/1901.05353`
  - G. & Shawe-Taylor, "A Primer on PAC-Bayesian Learning", **ICML 2019 tutorial** `https://bguedj.github.io/icml2019/index.html`

- Most PAC-Bayes generalisation bounds are **computable** tight upper bounds on the population error, *i.e.* an estimate of the error on any unseen future data.

- PAC-Bayes bounds hold for **any distribution on hypotheses.** As such, they are a principled way to invent new learning algorithms.

This spotlight

G. Letarte, P. Germain, B. G., F. Laviolette. *Dichotomize and Generalize: PAC-Bayesian Binary Activated Deep Neural Networks*, to appear in NeurIPS 2019
https://arxiv.org/abs/1905.10259

This spotlight

G. Letarte, P. Germain, B. G., F. Laviolette. *Dichotomize and Generalize: PAC-Bayesian Binary Activated Deep Neural Networks*, to appear in NeurIPS 2019
`https://arxiv.org/abs/1905.10259`

We focused on DNN with a **binary activation function**: surprisingly effective while preserving low computing and memory footprints.

This spotlight

G. Letarte, P. Germain, B. G., F. Laviolette. *Dichotomize and Generalize: PAC-Bayesian Binary Activated Deep Neural Networks*, to appear in NeurIPS 2019

`https://arxiv.org/abs/1905.10259`

We focused on DNN with a **binary activation function**: surprisingly effective while preserving low computing and memory footprints.

- Very few meaningful generalisation bounds for DNN

This spotlight

G. Letarte, P. Germain, B. G., F. Laviolette. *Dichotomize and Generalize: PAC-Bayesian Binary Activated Deep Neural Networks*, to appear in NeurIPS 2019
https://arxiv.org/abs/1905.10259

We focused on DNN with a **binary activation function**: surprisingly effective while preserving low computing and memory footprints.

- Very few meaningful generalisation bounds for DNN
  Breakthrough: SOTA PAC-Bayes generalisation bound

This spotlight

G. Letarte, P. Germain, B. G., F. Laviolette. *Dichotomize and Generalize: PAC-Bayesian Binary Activated Deep Neural Networks*, to appear in NeurIPS 2019
https://arxiv.org/abs/1905.10259

We focused on DNN with a **binary activation function**: surprisingly effective while preserving low computing and memory footprints.

- Very few meaningful generalisation bounds for DNN
  Breakthrough: SOTA PAC-Bayes generalisation bound
- How to train a network with non-differentiable activation function?

This spotlight

G. Letarte, P. Germain, B. G., F. Laviolette. *Dichotomize and Generalize: PAC-Bayesian Binary Activated Deep Neural Networks*, to appear in NeurIPS 2019
https://arxiv.org/abs/1905.10259

We focused on DNN with a **binary activation function**: surprisingly effective while preserving low computing and memory footprints.

- Very few meaningful generalisation bounds for DNN
  Breakthrough: SOTA PAC-Bayes generalisation bound
- How to train a network with non-differentiable activation function?
  Breakthrough: training by minimising the bound (SGD + tricks)

This spotlight

G. Letarte, P. Germain, B. G., F. Laviolette. *Dichotomize and Generalize: PAC-Bayesian Binary Activated Deep Neural Networks*, to appear in NeurIPS 2019
`https://arxiv.org/abs/1905.10259`

We focused on DNN with a **binary activation function**: surprisingly effective while preserving low computing and memory footprints.

- Very few meaningful generalisation bounds for DNN
  Breakthrough: SOTA PAC-Bayes generalisation bound
- How to train a network with non-differentiable activation function?
  Breakthrough: training by minimising the bound (SGD + tricks)
- Who cares? Generalisation bounds are a theoretician's concern!

This spotlight

G. Letarte, P. Germain, B. G., F. Laviolette. *Dichotomize and Generalize: PAC-Bayesian Binary Activated Deep Neural Networks*, to appear in NeurIPS 2019
https://arxiv.org/abs/1905.10259

We focused on DNN with a **binary activation function**: surprisingly effective while preserving low computing and memory footprints.

- Very few meaningful generalisation bounds for DNN
  Breakthrough: SOTA PAC-Bayes generalisation bound
- How to train a network with non-differentiable activation function?
  Breakthrough: training by minimising the bound (SGD + tricks)
- Who cares? Generalisation bounds are a theoretician's concern!
  Breakthrough: Our bound is computable and serves as a safety check to practitioners

Binary Activated Neural Networks

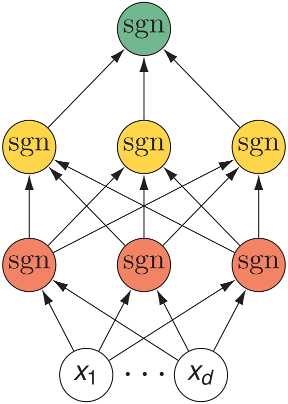
- $\mathbf{x} \in \mathbb{R}^{d_0}$, $y \in \{-1, 1\}$

Architecture:

- *L fully connected* layers
- d_k denotes the number of neurons of the k^{th} layer
- $\operatorname{sgn}(a) = 1$ if $a > 0$ and $\operatorname{sgn}(a) = -1$ otherwise

Parameters:

- $\mathbf{W}_k \in \mathbb{R}^{d_k \times d_{k-1}}$ denotes the weight matrices.
- $\theta = \operatorname{vec}\left(\{\mathbf{W}_k\}_{k=1}^{L}\right) \in \mathbb{R}^D$



Prediction

$$f_\theta(\mathbf{x}) = \operatorname{sgn}\Big(\mathbf{w}_L \operatorname{sgn}\Big(\mathbf{W}_{L-1} \operatorname{sgn}\Big(\ldots \operatorname{sgn}\Big(\mathbf{W}_1 \mathbf{x}\Big)\Big)\Big)\Big),$$

Generalisation bound

Generalisation bound

For an arbitrary number of layers and neurons, with probability at least $1 - \delta$, for any $\theta \in \mathbb{R}^D$

$$R_{\mathrm{out}}(F_\theta) \leqslant$$

$$\inf_{C>0} \left\{ \frac{1}{1-e^{-C}} \left(1 - \exp\left(-C R_{\mathrm{in}}(F_\theta) - \frac{\frac{1}{2}\|\theta - \theta_0\|^2 + \log \frac{2\sqrt{m}}{\delta}}{m} \right) \right) \right\},$$

where

$$R_{\mathrm{in}}(F_\theta) = \mathop{\mathbf{E}}_{\theta' \sim Q_\theta} R_{\mathrm{in}}(f_{\theta'}) = \frac{1}{m} \sum_{i=1}^{m} \left[\frac{1}{2} - \frac{1}{2} y_i F_\theta(\mathbf{x}_i) \right].$$

(A selection of) numerical results

Model name	Cost function	Train split	Valid split	Model selection	Prior
MLP–tanh	linear loss, L2 regularized	80%	20%	valid linear loss	-
PBGNet$_\ell$	linear loss, L2 regularized	80%	20%	valid linear loss	random init
PBGNet	**PAC-Bayes bound**	**100 %**	**-**	**PAC-Bayes bound**	**random init**
PBGNet$_{pre}$					
– pretrain	linear loss (20 epochs)	50%	-	-	random init
– final	PAC-Bayes bound	50%	-	PAC-Bayes bound	pretrain

Dataset	MLP–tanh		PBGNet$_\ell$		PBGNet			PBGNet$_{pre}$		
	E_S	E_T	E_S	E_T	E_S	E_T	Bound	E_S	E_T	Bound
ads	0.021	0.037	0.018	**0.032**	0.024	0.038	0.283	0.034	0.033	0.058
adult	0.128	0.149	0.136	**0.148**	0.158	0.154	0.227	0.153	0.151	0.165
mnist17	0.003	**0.004**	0.008	0.005	0.007	0.009	0.067	0.003	0.005	0.009
mnist49	0.002	**0.013**	0.003	0.018	0.034	0.039	0.153	0.018	0.021	0.030
mnist56	0.002	0.009	0.002	0.009	0.022	0.026	0.103	0.008	**0.008**	0.017
mnistLH	0.004	**0.017**	0.005	0.019	0.071	0.073	0.186	0.026	0.026	0.033

Thanks!

We have several PhD / postdoc / visiting researcher positions available in my group, based in London and affiliated with Inria and UCL.

Feel free to reach out!

`https://bguedj.github.io`

Positive solutions for Large Random Linear Systems

Jamal Najim

najim@univ-mlv.fr
CNRS & Université Paris Est

joint work with Pierre Bizeul

Machine Learning in the real world - Criteo Labs - july 2019

A Large Random Linear System

We are interested in the equation

$$\boxed{\boldsymbol{x} = \mathbf{1} + \frac{A}{\boldsymbol{\alpha}\sqrt{N}}\boldsymbol{x}}$$

where

A Large Random Linear System

We are interested in the equation

$$\boxed{\boldsymbol{x} = \mathbf{1} + \frac{A}{\boldsymbol{\alpha}\sqrt{N}}\boldsymbol{x}}$$

where

- $\boldsymbol{x}$ is a $N \times 1$ unknown vector,
- $\mathbf{1}$ is a $N \times 1$ vector of ones,
- A is a $N \times N$ random matrix with i.i.d. entries $\mathcal{N}(0, 1)$,
- $\boldsymbol{\alpha}$ is a positive scalar parameter to be tuned.

A Large Random Linear System

We are interested in the equation

$$\boxed{\boldsymbol{x} = \mathbf{1} + \frac{A}{\boldsymbol{\alpha}\sqrt{N}}\boldsymbol{x}}$$

where

- $\boldsymbol{x}$ is a $N \times 1$ unknown vector,
- $\mathbf{1}$ is a $N \times 1$ vector of ones,
- A is a $N \times N$ random matrix with i.i.d. entries $\mathcal{N}(0, 1)$,
- $\boldsymbol{\alpha}$ is a positive scalar parameter to be tuned.

Questions

- Does this system admit a solution $\boxed{\boldsymbol{x} = \left(I - \frac{A}{\boldsymbol{\alpha}\sqrt{N}}\right)^{-1}\mathbf{1}}$?

A Large Random Linear System

We are interested in the equation

$$\boxed{\boldsymbol{x} = \mathbf{1} + \frac{A}{\boldsymbol{\alpha}\sqrt{N}}\boldsymbol{x}}$$

where

- $\boldsymbol{x}$ is a $N \times 1$ unknown vector,
- $\mathbf{1}$ is a $N \times 1$ vector of ones,
- A is a $N \times N$ random matrix with i.i.d. entries $\mathcal{N}(0,1)$,
- $\boldsymbol{\alpha}$ is a positive scalar parameter to be tuned.

Questions

- Does this system admit a solution $\boxed{\boldsymbol{x} = \left(I - \frac{A}{\boldsymbol{\alpha}\sqrt{N}}\right)^{-1}\mathbf{1}}$?
- Conditions to get a solution $\boldsymbol{x}$ with positive components?

A Large Random Linear System

We are interested in the equation

$$\boxed{\boldsymbol{x} = \mathbf{1} + \frac{A}{\boldsymbol{\alpha}\sqrt{N}}\boldsymbol{x}}$$

where

- $\boldsymbol{x}$ is a $N \times 1$ unknown vector,
- $\mathbf{1}$ is a $N \times 1$ vector of ones,
- A is a $N \times N$ random matrix with i.i.d. entries $\mathcal{N}(0,1)$,
- $\boldsymbol{\alpha}$ is a positive scalar parameter to be tuned.

Questions

- Does this system admit a solution $\boxed{\boldsymbol{x} = \left(I - \frac{A}{\boldsymbol{\alpha}\sqrt{N}}\right)^{-1}\mathbf{1}}$?
- Conditions to get a solution $\boldsymbol{x}$ with positive components?

Motivation

- Feasibility and stability in ecological networks.

Confinement of the spectrum of $\frac{A}{\sqrt{N}}$

Confinement of the spectrum of $\frac{A}{\sqrt{N}}$

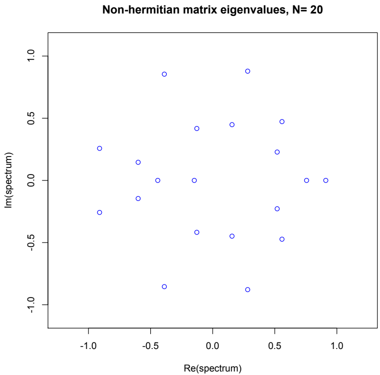


Figure: Distribution of $A_N/\sqrt{N}$'s eigenvalues

Confinement of the spectrum of $\frac{A}{\sqrt{N}}$

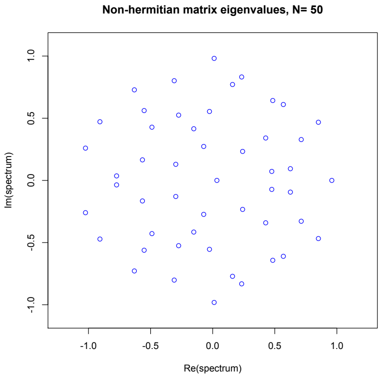


Figure: Distribution of $A_N/\sqrt{N}$'s eigenvalues

Confinement of the spectrum of $\frac{A}{\sqrt{N}}$

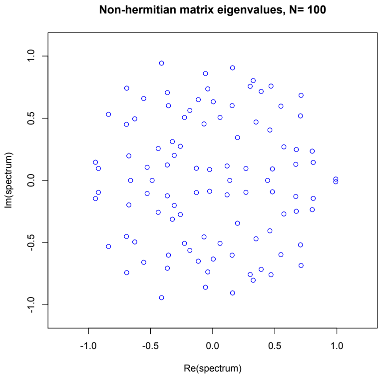


Figure: Distribution of $A_N/\sqrt{N}$'s eigenvalues

Confinement of the spectrum of $\frac{A}{\sqrt{N}}$

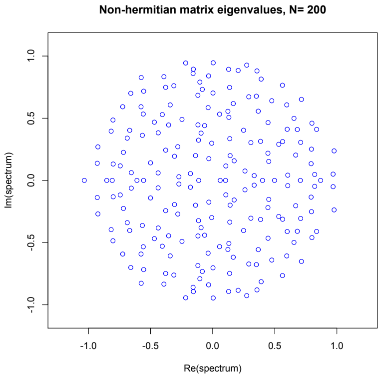


Figure: Distribution of $A_N/\sqrt{N}$'s eigenvalues

Confinement of the spectrum of $\frac{A}{\sqrt{N}}$

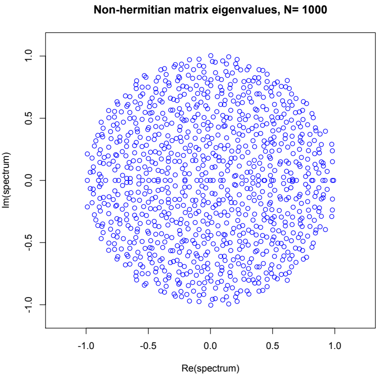


Figure: Distribution of $A_N/\sqrt{N}$'s eigenvalues

Confinement of the spectrum of $\frac{A}{\sqrt{N}}$

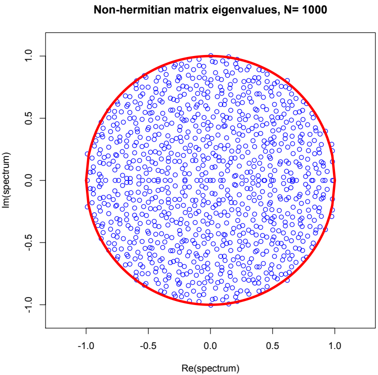


Figure: The circular law (in red)

Theorem: The Circular Law (Ginibre, Metha, Girko, Tao & Vu, etc.)

The spectrum of $\frac{A}{\sqrt{N}}$ converges to **the uniform probability on the disc**

Existence of a solution .. with no positive components

- From the spectrum confinement property,

$$\boldsymbol{x} = \left(I - \frac{A}{\boldsymbol{\alpha}\sqrt{N}}\right)^{-1} \mathbf{1} \quad \text{exists for } \boldsymbol{\alpha} > \mathbf{1}$$

Existence of a solution .. with no positive components

- From the spectrum confinement property,

$$\boldsymbol{x} = \left(I - \frac{A}{\boldsymbol{\alpha}\sqrt{N}} \right)^{-1} \mathbf{1} \quad \text{exists for } \boldsymbol{\alpha} > \mathbf{1}$$

- **but**

$$x_k \ \sim \ \mathcal{N}\left(1, \frac{1}{\boldsymbol{\alpha}^2 - 1}\right) \qquad \text{i.i.d.} \quad \text{as } N \to \infty$$

Existence of a solution .. with no positive components

- From the spectrum confinement property,

$$\boldsymbol{x} = \left(I - \frac{A}{\boldsymbol{\alpha}\sqrt{N}} \right)^{-1} \mathbf{1} \quad \text{exists for } \boldsymbol{\alpha} > \mathbf{1}$$

- **but**

$$x_k \sim \mathcal{N}\left(1, \frac{1}{\boldsymbol{\alpha}^2 - 1}\right) \qquad \text{i.i.d.} \quad \text{as } N \to \infty$$

- As a consequence

$$\mathbb{P}\left\{ \inf_{k \in [N]} x_k > 0 \right\} \quad \sim \quad \mathbb{P}\left\{x_k > 0\right\}^N \quad \xrightarrow[N \to \infty]{} \quad 0\,.$$

$\Rightarrow$ no positive solutions

Positivity of the solution

Consider now the case $\boxed{\boldsymbol{\alpha} = \boldsymbol{\alpha}_N \xrightarrow[N\to\infty]{} \infty}$

Positivity of the solution

Consider now the case $\boxed{\boldsymbol{\alpha} = \boldsymbol{\alpha}_N \xrightarrow[N\to\infty]{} \infty}$

Theorem (phase transition, Bizeul-N. '19)

- If

$$\boldsymbol{\alpha}_N \ \leq_\delta \ \sqrt{2\log(N)} \qquad \Leftrightarrow \qquad \boldsymbol{\alpha}_N \leq (1-\delta)\sqrt{2\log(N)}$$

then

$$\boxed{\mathbb{P}\left\{\inf_{k\in[N]} x_k > 0\right\} \xrightarrow[N\to\infty]{} 0} \qquad \Rightarrow \qquad \text{no positive solutions.}$$

Positivity of the solution

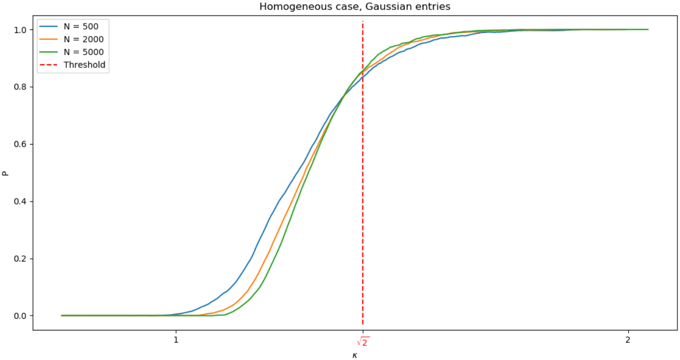
Consider now the case $\boxed{\boldsymbol{\alpha} = \boldsymbol{\alpha}_N \xrightarrow[N\to\infty]{} \infty}$

Theorem (phase transition, Bizeul-N. '19)

- If
$$\boldsymbol{\alpha}_N \ \leq_\delta \ \sqrt{2\log(N)} \qquad \Leftrightarrow \quad \boldsymbol{\alpha}_N \leq (1-\delta)\sqrt{2\log(N)}$$
then
$$\boxed{\mathbb{P}\left\{\inf_{k\in[N]} x_k > 0\right\} \xrightarrow[N\to\infty]{} 0} \qquad \Rightarrow \quad \text{no positive solutions.}$$

- If
$$\boldsymbol{\alpha}_N \ \geq_\delta \ \sqrt{2\log(N)} \qquad \Leftrightarrow \quad \boldsymbol{\alpha}_N \geq (1+\delta)\sqrt{2\log(N)}$$
then
$$\boxed{\mathbb{P}\left\{\inf_{k\in[N]} x_k > 0\right\} \xrightarrow[N\to\infty]{} 1} \qquad \Rightarrow \quad \text{positive solutions.}$$

Phase transition (gaussian case)



- We plot the frequency (over 500 trials) of positive solutions for the linear system

$$\boldsymbol{x} = \mathbf{1} + \frac{1}{\kappa\sqrt{\log(N)}} \frac{A}{\sqrt{N}} \boldsymbol{x}$$

  as a function of the normalization parameter κ.
- As expected, we observe threshold phenomenon around the critical value $\kappa = \sqrt{2}$.

A heuristics for the critical scaling

1. **Unfold the resolvent** and write

$$x_k \quad = \quad \boldsymbol{e}_k^* \left(I - \frac{A}{\boldsymbol{\alpha}\sqrt{N}} \right)^{-1} \mathbf{1}$$

A heuristics for the critical scaling

1. **Unfold the resolvent** and write

$$x_k \quad = \quad \boldsymbol{e}_k^* \left(I - \frac{A}{\boldsymbol{\alpha}\sqrt{N}} \right)^{-1} \mathbf{1} \quad = \quad 1 + \frac{Z_k}{\boldsymbol{\alpha}} + \frac{R_k}{\boldsymbol{\alpha}^2} \text{ (remainder term)}$$

A heuristics for the critical scaling

1. **Unfold the resolvent** and write

$$x_k = \boldsymbol{e}_k^* \left(I - \frac{A}{\boldsymbol{\alpha}\sqrt{N}} \right)^{-1} \mathbf{1} = 1 + \frac{Z_k}{\boldsymbol{\alpha}} + \frac{R_k}{\boldsymbol{\alpha}^2} \text{ (remainder term)}$$

2. **Notice that**

$$Z_k \sim \mathcal{N}(0,1) \quad \text{i.i.d.} \qquad \text{and} \qquad \min_{k \in [N]} Z_k \sim -\sqrt{2\log(N)}$$

by extreme value theory.

A heuristics for the critical scaling

1. **Unfold the resolvent** and write

$$x_k \quad = \quad \boldsymbol{e}_k^* \left(I - \frac{A}{\boldsymbol{\alpha}\sqrt{N}} \right)^{-1} \mathbf{1} \quad = \quad 1 \ + \ \frac{Z_k}{\boldsymbol{\alpha}} \ + \ \frac{R_k}{\boldsymbol{\alpha}^2} \ \text{(remainder term)}$$

2. **Notice that**

$$Z_k \ \sim \ \mathcal{N}(0,1) \quad \text{i.i.d.} \qquad \text{and} \qquad \min_{k \in [N]} Z_k \sim -\sqrt{2\log(N)}$$

   by extreme value theory.

3. **Conclude**

$$\min_{k \in [N]} x_k \quad \approx \quad 1 + \frac{\min_{k \in [N]} Z_k}{\boldsymbol{\alpha}} + \cdots$$

A heuristics for the critical scaling

1. **Unfold the resolvent** and write

$$x_k \quad = \quad \boldsymbol{e}_k^* \left(I - \frac{A}{\boldsymbol{\alpha}\sqrt{N}} \right)^{-1} \mathbf{1} \quad = \quad 1 + \frac{Z_k}{\boldsymbol{\alpha}} + \frac{R_k}{\boldsymbol{\alpha}^2} \text{ (remainder term)}$$

2. **Notice that**

$$Z_k \sim \mathcal{N}(0,1) \quad \text{i.i.d.} \qquad \text{and} \qquad \min_{k \in [N]} Z_k \sim -\sqrt{2\log(N)}$$

   by extreme value theory.

3. **Conclude**

$$\min_{k \in [N]} x_k \quad \approx \quad 1 + \frac{\min_{k\in[N]} Z_k}{\boldsymbol{\alpha}} + \cdots \quad \approx \quad 1 - \frac{\sqrt{2\log(N)}}{\boldsymbol{\alpha}}$$

A heuristics for the critical scaling

1. **Unfold the resolvent** and write

$$x_k \quad = \quad \boldsymbol{e}_k^* \left(I - \frac{A}{\boldsymbol{\alpha}\sqrt{N}} \right)^{-1} \mathbf{1} \quad = \quad 1 \; + \; \frac{Z_k}{\boldsymbol{\alpha}} \; + \; \frac{R_k}{\boldsymbol{\alpha}^2} \; (\text{remainder term})$$

2. **Notice that**

$$Z_k \;\sim\; \mathcal{N}(0,1) \quad \text{i.i.d.} \qquad \text{and} \qquad \min_{k \in [N]} Z_k \sim -\sqrt{2\log(N)}$$

   by extreme value theory.

3. **Conclude**

$$\min_{k \in [N]} x_k \quad \approx \quad 1 + \frac{\min_{k \in [N]} Z_k}{\boldsymbol{\alpha}} + \cdots \quad \approx \quad 1 - \frac{\sqrt{2\log(N)}}{\boldsymbol{\alpha}}$$

4. The key control of the remainder term R_k can be proved via gaussian concentration.

$$\boxed{\frac{\max_{k \in [N]} R_k}{\alpha\sqrt{2\log(N)}} \xrightarrow[N \to \infty]{\mathcal{P}} 0} \qquad \text{and} \qquad \boxed{\frac{\min_{k \in [N]} R_k}{\alpha\sqrt{2\log(N)}} \xrightarrow[N \to \infty]{\mathcal{P}} 0}\,.$$

A heuristics for the critical scaling

1. **Unfold the resolvent** and write

$$x_k \quad = \quad \boldsymbol{e}_k^* \left(I - \frac{A}{\boldsymbol{\alpha}\sqrt{N}} \right)^{-1} \mathbf{1} \quad = \quad 1 + \frac{Z_k}{\boldsymbol{\alpha}} + \frac{R_k}{\boldsymbol{\alpha}^2} \text{ (remainder term)}$$

2. **Notice that**

$$Z_k \sim \mathcal{N}(0,1) \quad \text{i.i.d.} \qquad \text{and} \qquad \min_{k \in [N]} Z_k \sim -\sqrt{2\log(N)}$$

by extreme value theory.

3. **Conclude**

$$\min_{k \in [N]} x_k \quad \approx \quad 1 + \frac{\min_{k \in [N]} Z_k}{\boldsymbol{\alpha}} + \cdots \quad \approx \quad 1 - \frac{\sqrt{2\log(N)}}{\boldsymbol{\alpha}}$$

4. The key control of the remainder term R_k can be proved via gaussian concentration.

$$\boxed{\frac{\max_{k \in [N]} R_k}{\alpha\sqrt{2\log(N)}} \xrightarrow[N \to \infty]{\mathcal{P}} 0} \qquad \text{and} \qquad \boxed{\frac{\min_{k \in [N]} R_k}{\alpha\sqrt{2\log(N)}} \xrightarrow[N \to \infty]{\mathcal{P}} 0}.$$

Thank you for your attention!

LightOn
We bring Light to AI

Bringing light to AI

Iacopo Poli – Lead Machine Learning Engineer
iacopo@lighton.io

INCREASING DEMAND OF COMPUTE

LightOn

Microsoft boss: World needs more computing power

By Joe Miller
BBC News, Davos

Thom Quinn
@tpq__

Is deep learning right for you? Now in 1 easy step:

(Q) Do you have more than 10,000 samples?
> If no -- sorry, you don't have enough samples
> If yes -- sorry, you don't have enough compute

INCREASING DEMAND OF COMPUTE

ECOLOGICAL IMPACT OF AI

Dr Chloé Azencott
@cazencott
Follow

In a single day, I heard both Marc Schoenauer and @katecrawford discuss the ecological impact of AI and we need much more of this conversation.

Andrej Karpathy
@karpathy
Follow

"Hybrid Optical-Electronic Convolutional Neural Networks" computationalimaging.org/publications/h … incredibly interesting work - develops a hybrid optoelectronic CNN with an optical CONV1 layer that operates at zero power consumption (with rest of the forward pass in electronics (for now))

Green AI

Roy Schwartz, Jesse Dodge, Noah A. Smith, Oren Etzioni

(Submitted on 22 Jul 2019 (v1), last revised 13 Aug 2019 (this version, v3))

The role of artificial intelligence in achieving the Sustainable Development Goals

Ricardo Vinuesa, Hossein Azizpour, Iolanda Leite, Madeline Balaam, Virginia Dignum, Sami Domisch, Anna Felländer, Simone Langhans, Max Tegmark, Francesco Fuso Nerini

(Submitted on 30 Apr 2019)

OPTICAL PROCESSING UNIT

$$\mathbf{y} = |\mathbf{R}\mathbf{x}|^2 \qquad R_{ij} \in \mathbb{C}$$

$$Re\{R_{ij}\} \sim \mathcal{N}(0, \sigma^2)$$

$$Im\{R_{ij}\} \sim \mathcal{N}(0, \sigma^2)$$

1M input – 1M output
Speed: 2 kHz
Power:30W

Random Features for Large-Scale Kernel Machines

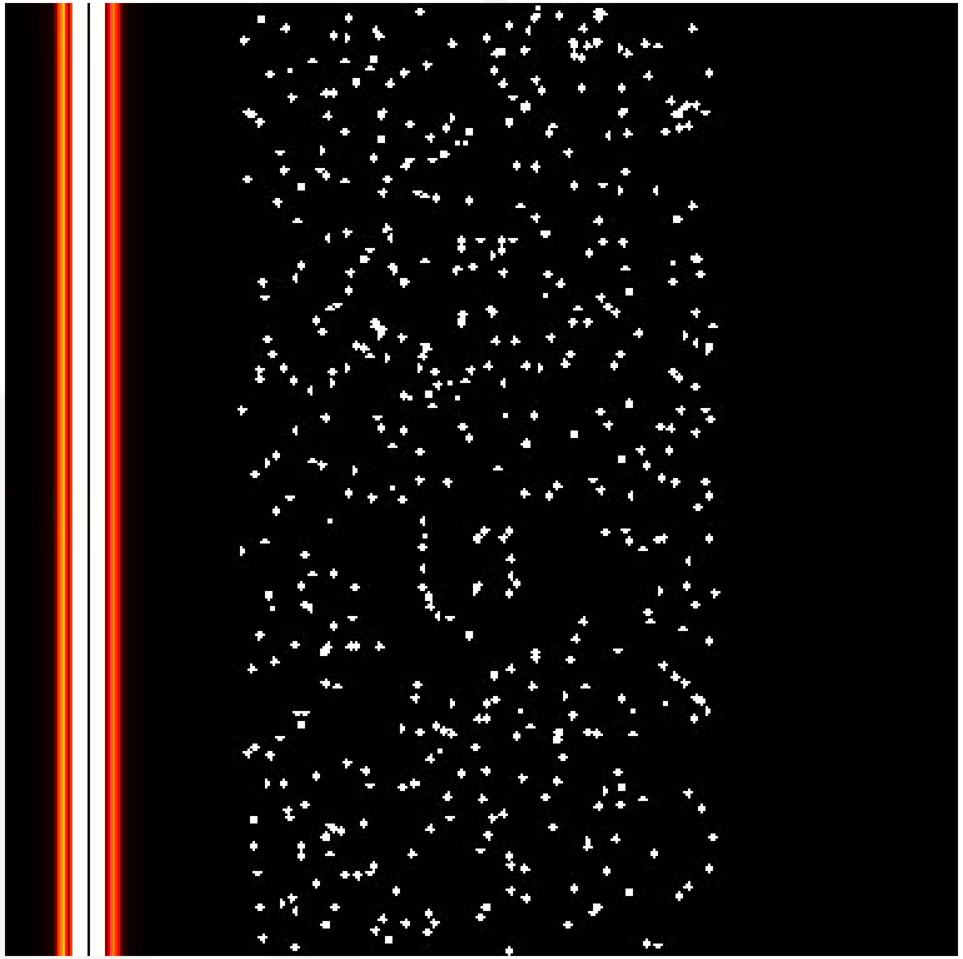
Ali Rahimi
Intel Research Seattle
Seattle, WA 98105
ali.rahimi@intel.com

Benjamin Recht
Caltech IST
Pasadena, CA 91125
brecht@ist.caltech.edu

Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions

Nathan Halko, Per-Gunnar Martinsson, Joel A. Tropp

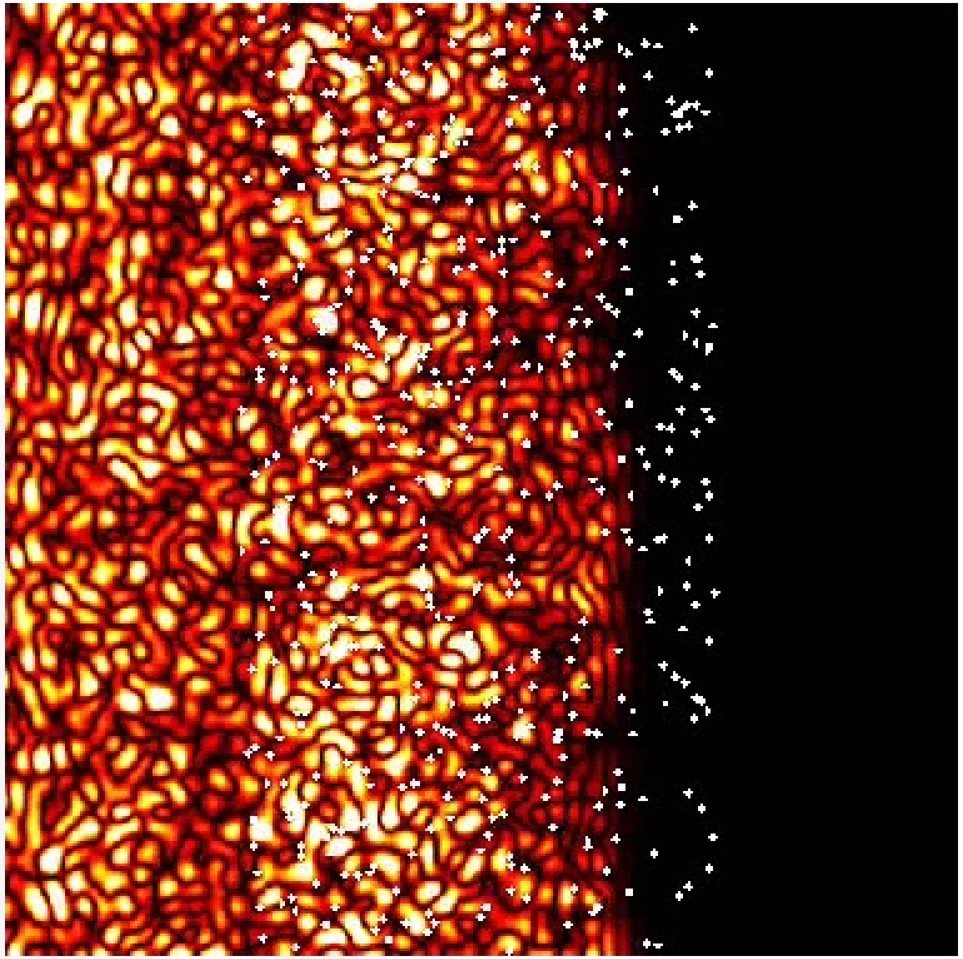
LIGHT SCATTERING



Credit: Emmanuel Bossy- Simsonic Software

Georges de la Tour – Saint Joseph charpentier

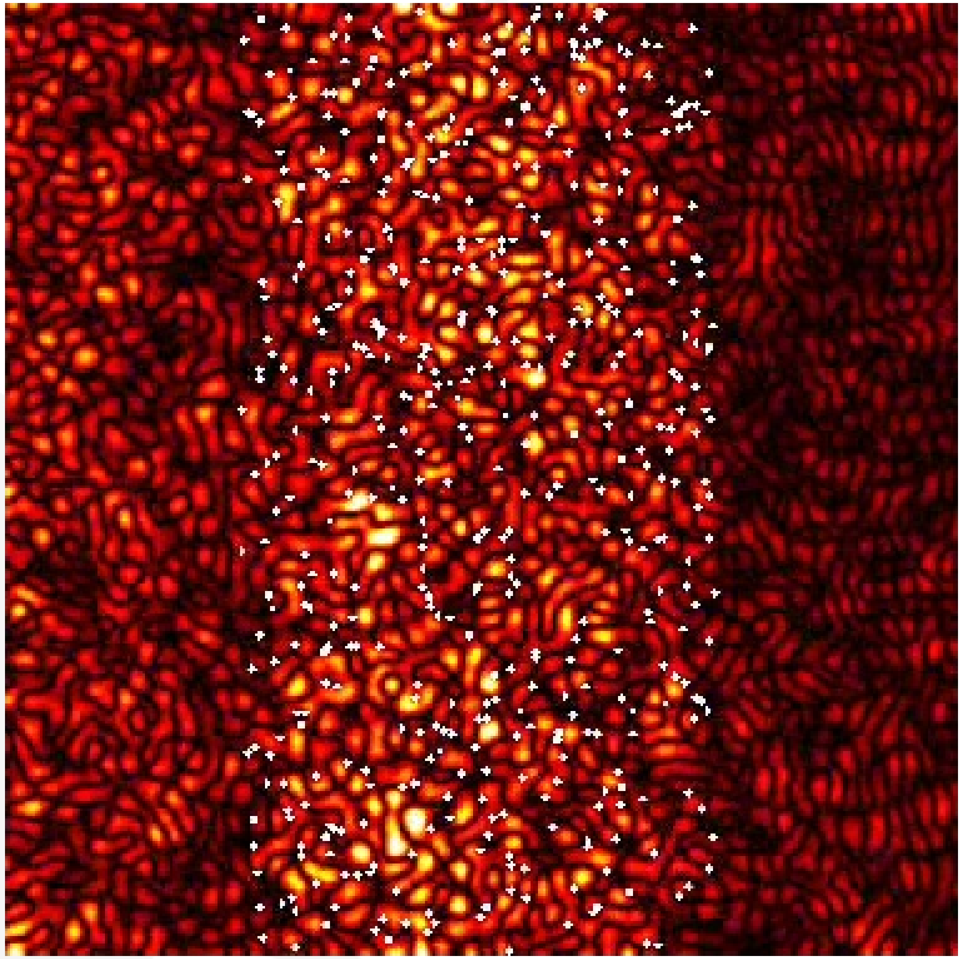
LIGHT SCATTERING



Credit: Emmanuel Bossy- Simsonic Software

Georges de la Tour – Saint Joseph charpentier

LIGHT SCATTERING



Credit: Emmanuel Bossy- Simsonic Software

Georges de la Tour – Saint Joseph charpentier

REINFORCEMENT LEARNING

Model-Free Episodic Control

Charles Blundell
Google DeepMind
cblundell@google.com

Benigno Uria
Google DeepMind
buria@google.com

Alexander Pritzel
Google DeepMind
apritzel@google.com

Yazhe Li
Google DeepMind
yazhe@google.com

Avraham Ruderman
Google DeepMind
aruderman@google.com

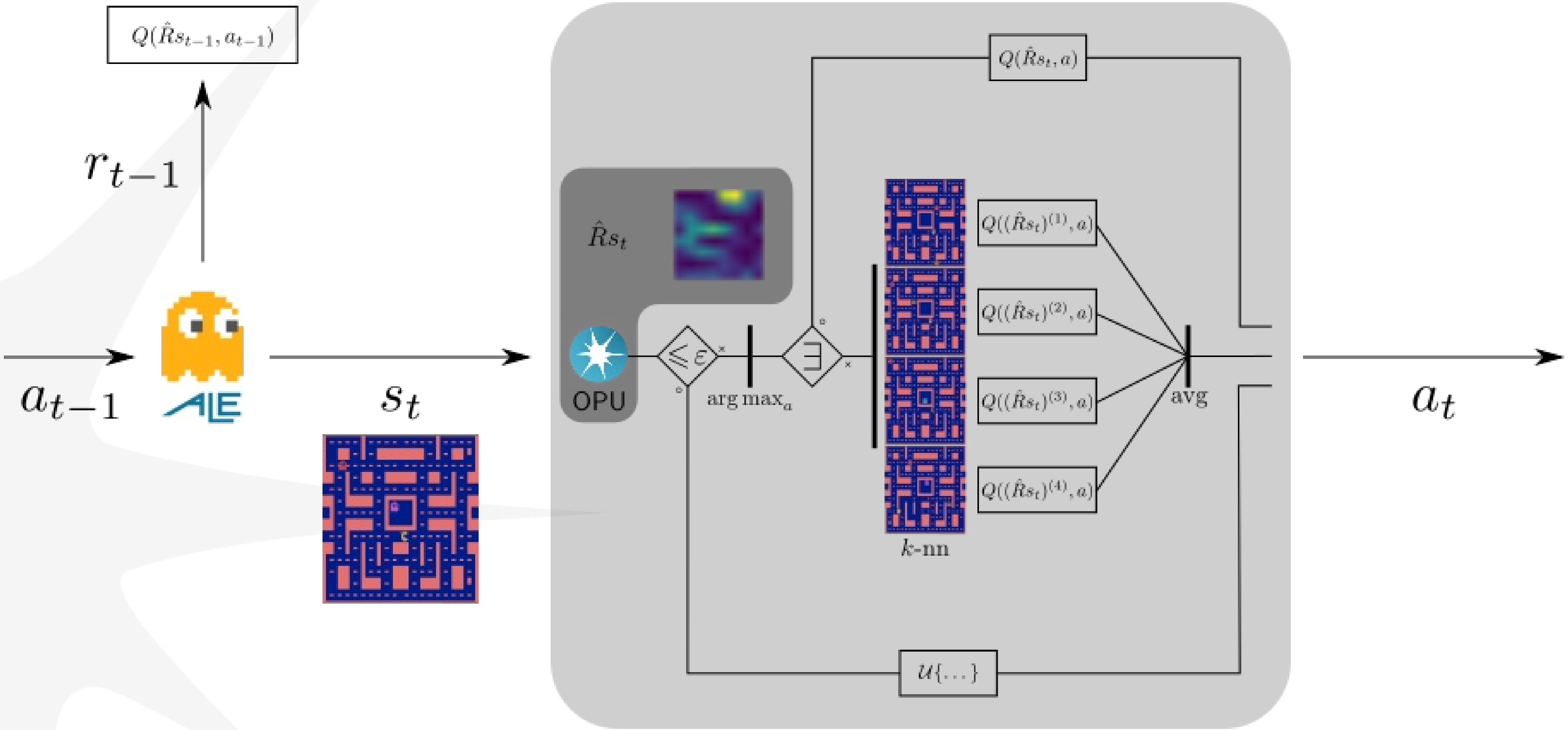
Joel Z Leibo
Google DeepMind
jzl@google.com

Jack Rae
Google DeepMind
jwrae@google.com

Daan Wierstra
Google DeepMind
wierstra@google.com

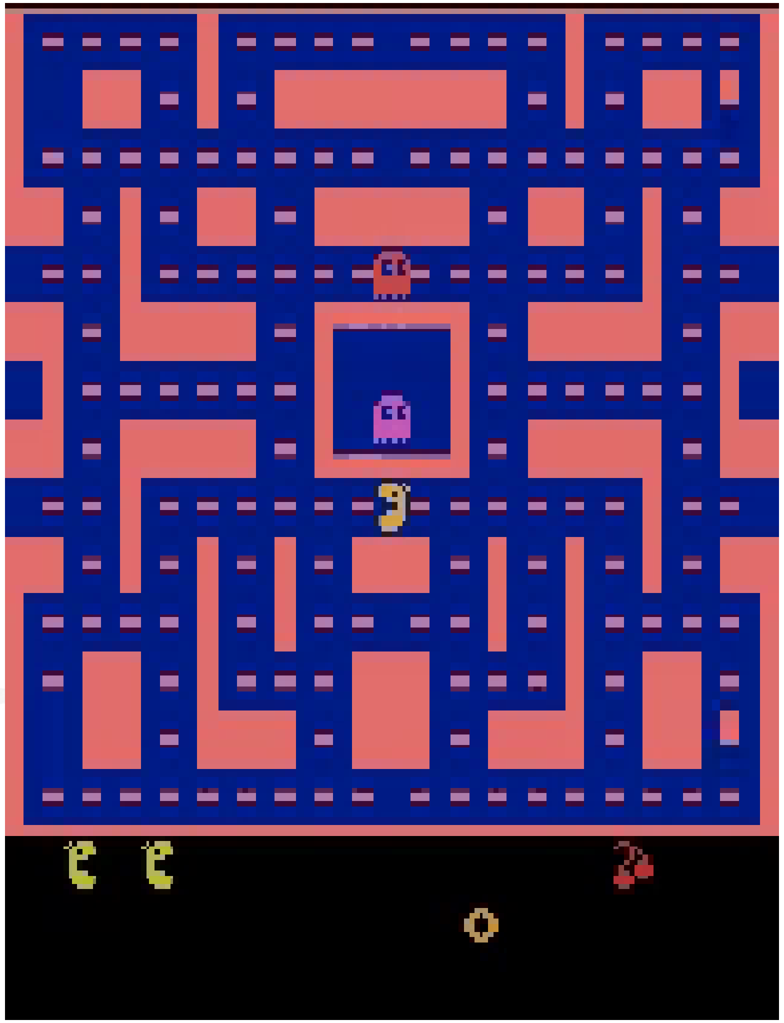
Demis Hassabis
Google DeepMind
demishassabis@google.com

REINFORCEMENT LEARNING



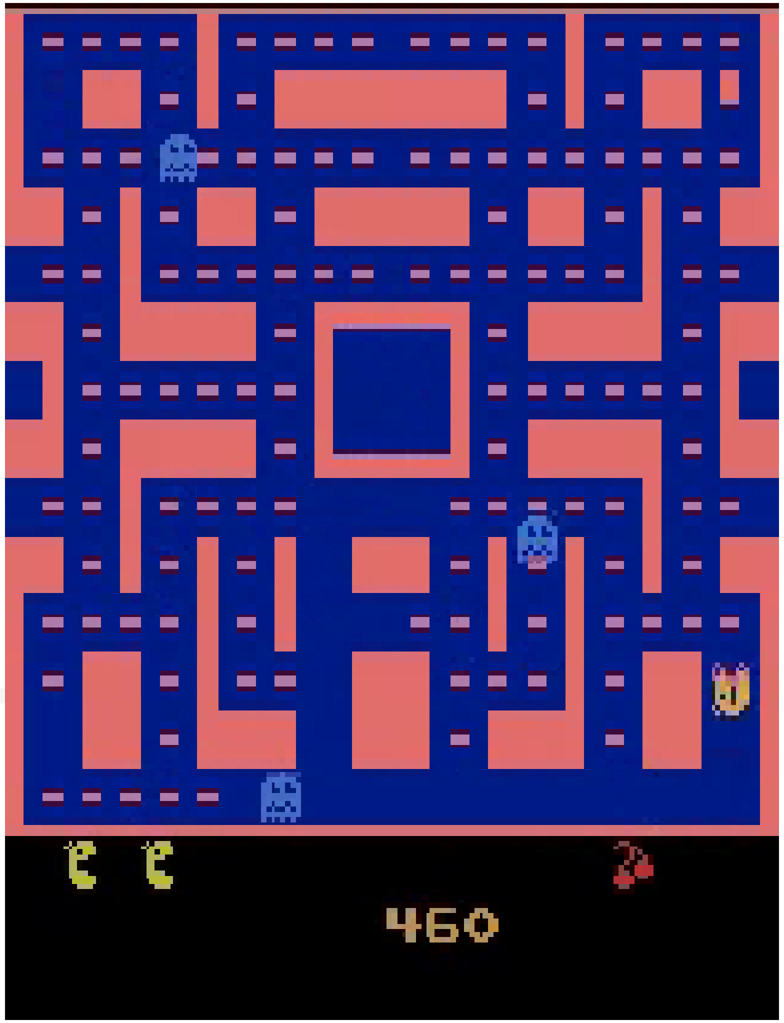
Credit: Martin Graive - Lighton

REINFORCEMENT LEARNING

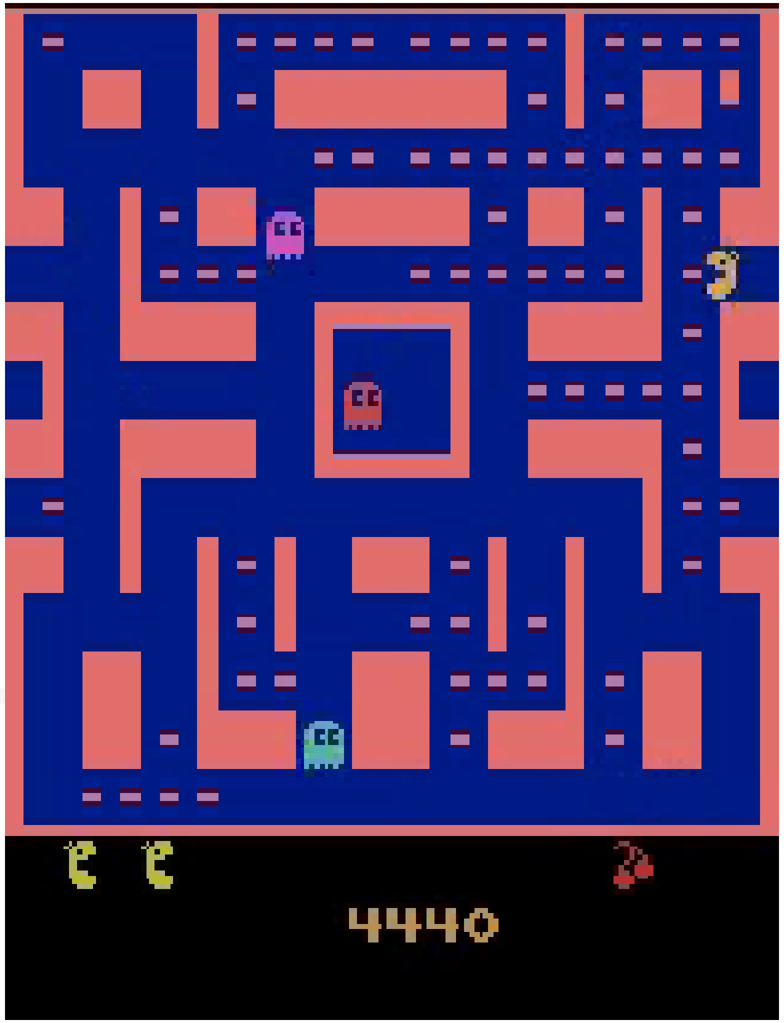


Credit: Martin Graive - Lighton

REINFORCEMENT LEARNING



Credit: Martin Graive - Lighton



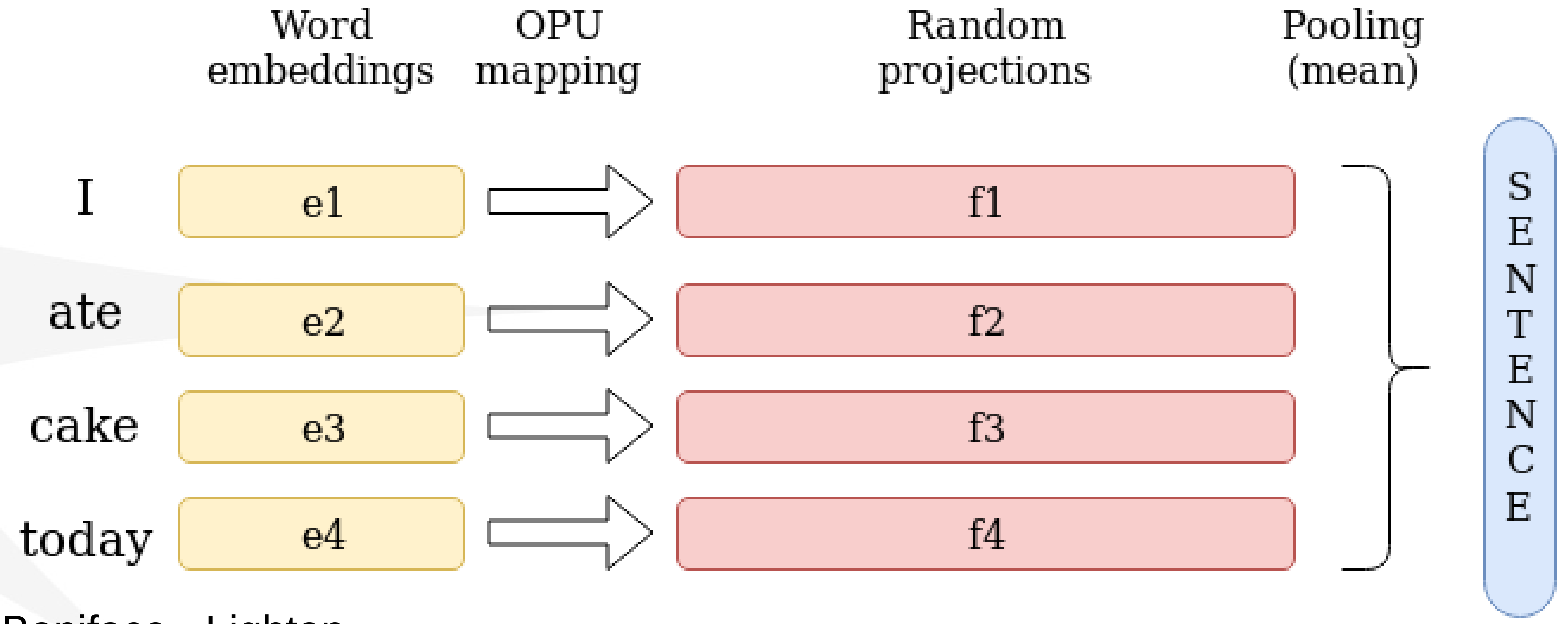
Credit: Martin Graive - Lighton

NOT ONLY IMAGES...

NO TRAINING REQUIRED: EXPLORING RANDOM ENCODERS FOR SENTENCE CLASSIFICATION

John Wieting*
Carnegie Mellon University
jwieting@cs.cmu.edu

Douwe Kiela
Facebook AI Research
dkiela@fb.com



Credit: François Boniface - Lighton

NOT ONLY IMAGES...

LightOn

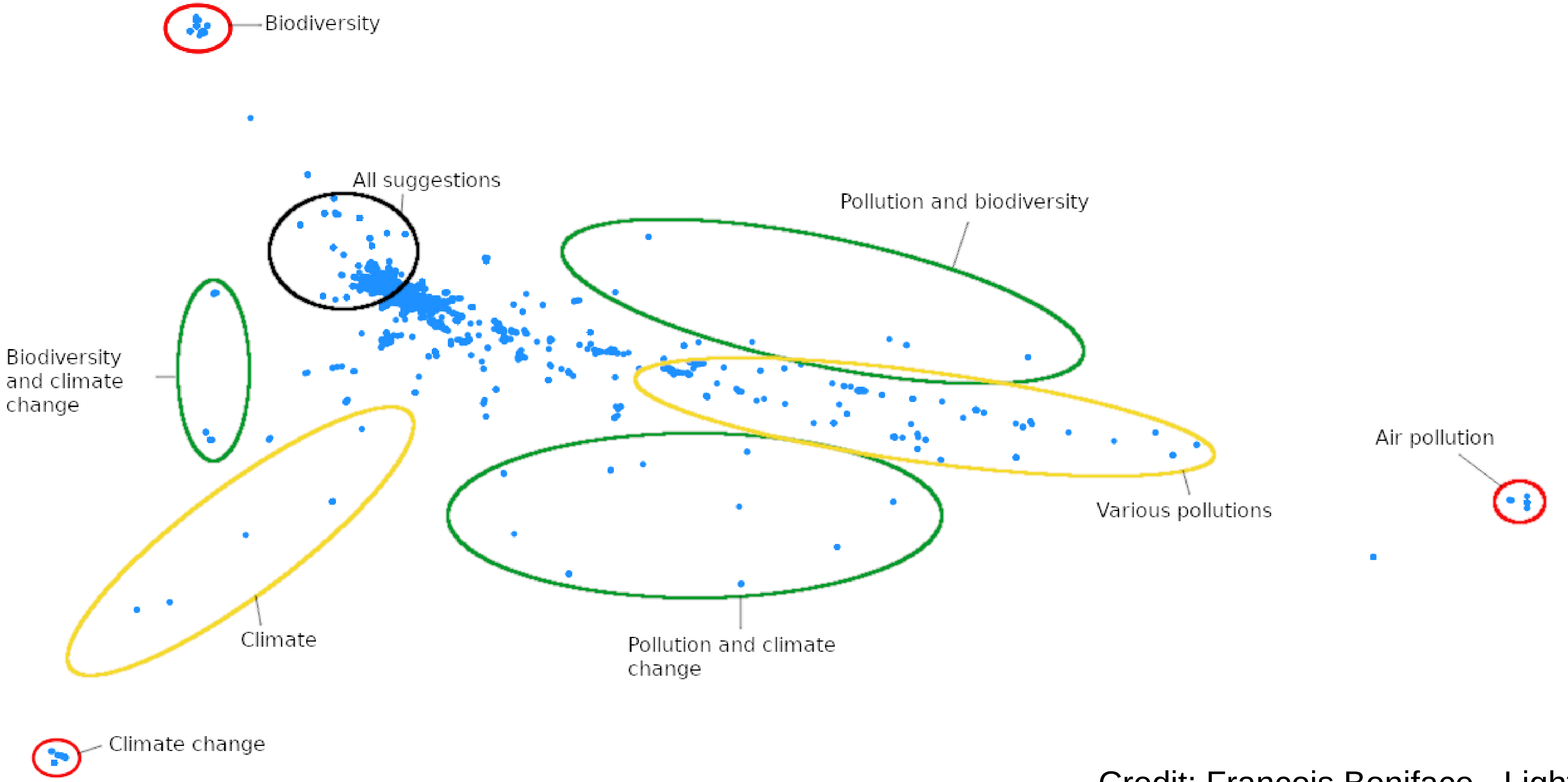
Shedding Light on the “Grand Débat”

LightOn
Apr 11 · 11 min read

Credit: François Boniface - Lighton

NOT ONLY IMAGES…

LightOn



Credit: François Boniface - Lighton

TRY IT OUT !

Thank you !

The RecoGym Challenge

Design a Recommendation Agent that can collect the largest reward in the RecoGym environment!

Motivating Example

Product
view

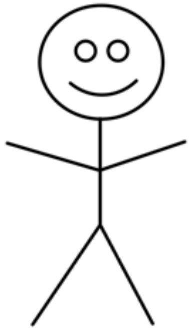
Motivating Example

Product
view

Motivating Example

Product
view

Motivating Example



Time

Product view

Recommend

The RecoGym Challenge 100

@RecoGym

RecoGym is recommendation simulator/game that allows us to:

- Simulate an A/B Test from the comfort of your own home, **allowing evaluation that is currently impossible** using static datasets
- You too can experience the excitement and joy of negative and neutral A/B Tests!
- Prize of 1000 euros (deadline 30 Nov)
- https://github.com/criteo-research/reco-gym

RecoGym Challange

Organized by Criteo - Current server time: Oct. 1, 2019, 8:22 p.m. UTC

Current	Next	End
Development	Final	Competition Ends
Oct. 1, 2019, midnight UTC	Nov. 30, 2019, midnight UTC	Dec. 1, 2019, midnight UTC

Development Final

Phase description

Development phase: create models and submit them or directly submit results on validation and/or test data; feed-back are provided on the validation set only.

Max submissions per day: 5

Max submissions total: 100

Download CSV

Click-Through Rate Results

#	User	Entries	Date of Last Entry	Team Name	CTR (q0.500), %	CTR (q0.025), %	CTR (q0.975), %	Time (seconds)	Detailed Results
1	ihtiihti	2	10/01/19		1.492 (1)	1.431 (1)	1.554 (1)	59.4 (2)	View
2	MartinB	2	10/01/19		1.434 (2)	1.374 (2)	1.495 (2)	55.2 (1)	View
3	Criteo	1	09/30/19	Criteo	1.403 (3)	1.344 (3)	1.464 (3)	375.6 (3)	View

CodaLab

Search Competitions | My Competitions | Help | Sign Up | Sign In

RecoGym Challange

Organized by Criteo - Current server time: Oct. 1, 2019, 8:22 p.m. UTC

▶ Current	Next	End
Development	Final	Competition Ends
Oct. 1, 2019, midnight UTC	Nov. 30, 2019, midnight UTC	Dec. 1, 2019, midnight UTC

Learn the Details | Phases | Participate | Results | Public Submissions | Forums

Development | Final

Phase description

Development phase: create models and submit them or directly submit results on validation and/or test data; feed-back are provided on the validation set only.

Max submissions per day: 5

Max submissions total: 100

Your amazing RecoGym Agent here!

Download CSV

Click-Through Rate Results

#	User	Entries	Date of Last Entry	Team Name	CTR (q0.500), % ▲	CTR (q0.025), % ▲	CTR (q0.975), % ▲	Time (seconds) ▲	Detailed Results
1	ihtiihti	2	10/01/19		1.492 (1)	1.431 (1)	1.554 (1)	59.4 (2)	View
2	MartinB	2	10/01/19		1.434 (2)	1.374 (2)	1.495 (2)	55.2 (1)	View
3	Criteo	1	09/30/19	Criteo	1.403 (3)	1.344 (3)	1.464 (3)	375.6 (3)	View

Okay, what is the challenge?

Within the challenge, there are two tasks:

- **RecoGym Challenge 100:** Learning to recommend with 100 just actions. Prize 1000 euros. Live now!

- **RecoGym Challenge 10 000:** Learning to recommend in a larger action spaces. Prize 2000 euros. Stay Tuned

Important links

Follow us on Twitter: @RecoGym

RecoGym challenge website:
https://sites.google.com/view/recogymchallenge/home

RecoGym repo: https://github.com/criteo-research/reco-gym
(the simulator, along with many tutorials and notebooks)

Difference-of-Convex Algorithm applied to adversarial robustness verification

Ismaila Seck [1,2,3] Gaelle Loosli [3,4]
Stephane Canu [2,3] Yi-Shuai Niu [5]

[1]Normandie Univ, INSA Rouen [2]UNIROUEN, UNIHAVRE, LITIS [3]UCA, LIMOS

[4]PobRun [5]School of Mathematical Sciences, Shanghai Jiao Tong University

October 2, 2019

Adversarial examples

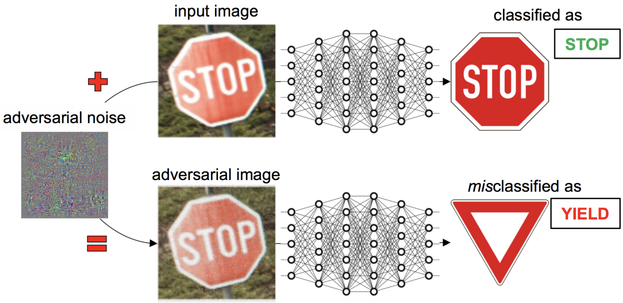


Figure 1: Illustration of the use of adversarial examples.

Formulation as an optimization problem

- $\mathbf{x}, y$: original image and its class
- $\mathbf{x}'$: adversarial image we are looking for
- $f_k(.)$: the k-th output of the network

$$\left\{ \begin{array}{ll} \min & \|\mathbf{x} - \mathbf{x}'\| \\ \text{s.t.} & \underset{k=1,\ldots,c}{\operatorname{argmax}} f_k(\mathbf{x}') \neq y, \\ & \mathbf{x}' \in [0,1]^d. \end{array} \right. \tag{1}$$

Formulation as an optimization problem

- $\mathbf{x}, y$: original image and its class
- $\mathbf{x}'$: adversarial image we are looking for
- $f_k(.)$: the k-th output of the network

$$\left\{ \begin{array}{ll} \min & \|\mathbf{x} - \mathbf{x}'\| \\ \text{s.t.} & \underset{k=1,\ldots,c}{\operatorname{argmax}} f_k(\mathbf{x}') \neq y, \\ & \mathbf{x}' \in [0,1]^d. \end{array} \right. \tag{1}$$

Linearization of the argmax constraint

$$
(1) \iff \left\{
\begin{array}{lll}
\min & \|\mathbf{x}-\mathbf{x}'\| & \\
\text{s.t.} & m \leq f_k(\mathbf{x}') + (1-\mathrm{a}_k)M_m, & k = 1,\dots,c \\
 & m \geq f_k(\mathbf{x}'), & k = 1,\dots,c \\
 & \sum_{k=1}^{c} \mathrm{a}_k = 1, & \\
 & \mathrm{a}_y = 0, & \\
 & m \in \mathbb{R}, & \\
 & \mathrm{a} \in \{0,1\}^c, & \\
 & \mathbf{x}' \in [0,1]^d. &
\end{array}
\right. \tag{2}
$$

Using DC to get rid of the binary variables

$$\begin{cases} \min & \|\mathbf{x} - \mathbf{x}'\| + \sum_{k=1}^{c} \mathrm{a}_k(1 - \mathrm{a}_k) \\ \text{s.t.} & m \leq f_k(\mathbf{x}') + (1 - \mathrm{a}_k)M_m, \quad k = 1, \dots, c \\ & m \geq f_k(\mathbf{x}'), \quad k = 1, \dots, c \\ & \sum_{k=1}^{c} \mathrm{a}_k = 1, \\ & \mathrm{a}_y = 0, \\ & m \in \mathbb{R}, \\ & \mathrm{a} \in [0,1]^c, \\ & \mathbf{x}' \in [0,1]^d. \end{cases} \tag{3}$$

The End

Thanks for your attention !

Functional Isolation Forest

Guillaume Staerman*

Joint work with

Pavlo Mozharovskyi*, Stéphan Clémençon* and Florence D'Alché-Buc*

MLITRW, October 02, 2019

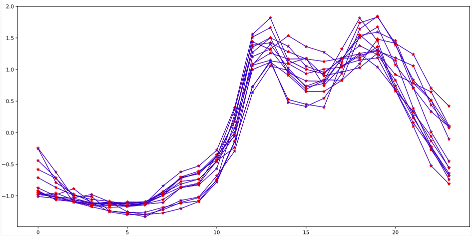
*LTCI, Telecom Paris, Institut Polytechnique de Paris.

Functional Data Framework

- Let $X = \{X(t) \in \mathbb{R}^d,\ t \in [0,1]\}$ be a random variable that takes its values in a (multivariate) functional space.

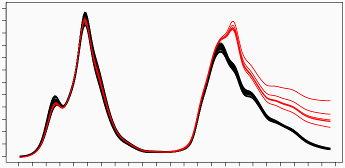
- In practice, we only have access to the realization of X at a finite number of arguments/times, $x = \{x(t_1), \ldots, x(t_p)\}$ such that $0 \leq t_1 < \cdots < t_p \leq 1$.

- The first step: reconstruct functional object from partial observations (time-series) with interpolation or basis decomposition.

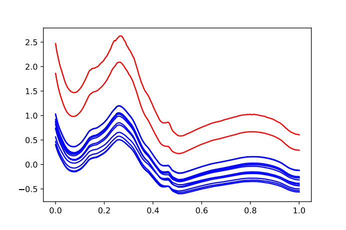


Anomaly detection and functional data

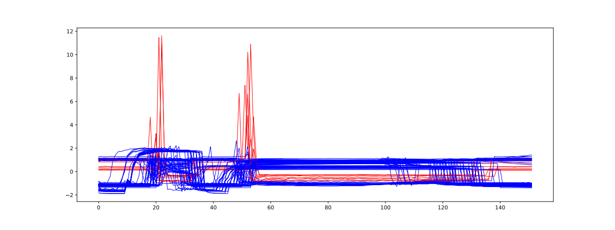
Shape anomalies



Shift anomalies



Isolated anomalies



Functional Isolation Forest

- This ensemble learning algorithm builds a collection of *functional isolation trees*.
- *Functional isolation tree* : binary tree based on a recursive and randomized tree-structured partitioning procedure.

- **General principle**:
  1. Select a function $\mathbf{d}$ into a dictionary $\mathcal{D}$.
  2. Compute the dot products $\langle \cdot, \cdot \rangle$ between $\mathbf{d}$ and the data.
  3. Draw randomly a treshold κ on the real line.
  4. Split the space by a perpendicular hyperplan along $\mathbf{d}$ going through κ.
  5. Repeat this procedure until every data are isolated!!!

- The trick : an anomaly should be isolated faster than normal data.

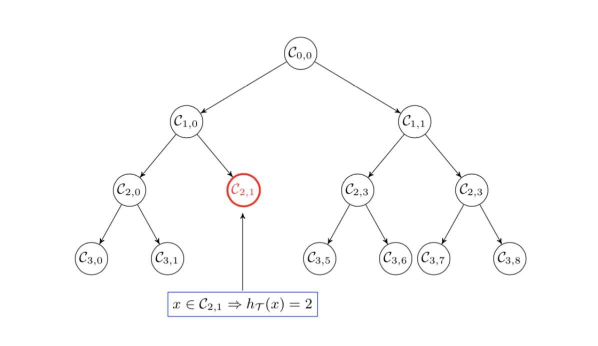
Anomaly score prediction

- One may then define the piecewise constant function $h_\tau : \mathcal{X} \to \mathbb{N}$ by: $\forall x \in \mathcal{X}$,

  $h_\tau(x) = j$ if and only if $x \in \mathcal{C}_{j,k}$ and $\mathcal{C}_{j,k}$ is associated to a terminal node.

- Considering a collection of F-$i$tree $\mathcal{T}_1, \ldots, \mathcal{T}_N$, the scoring function is defined by

$$s_n(x) = 2^{-\frac{1}{Nc(n)} \sum_{l=1}^{N} h_{\tau_l}(x)},$$



Thank you !

All codes are available at `https://github.com/Gstaerman/FIF`