Foundational AI: a mathematician's take

Benjamin Guedj
Inria and University College London
`https://bguedj.github.io`

CogX
June 8, 2020

Prelude: towards Artificial General Intelligence (AGI)

Prelude: towards Artificial General Intelligence (AGI)

Artificial entity capable of interacting and coexisting with its environment, especially humans:

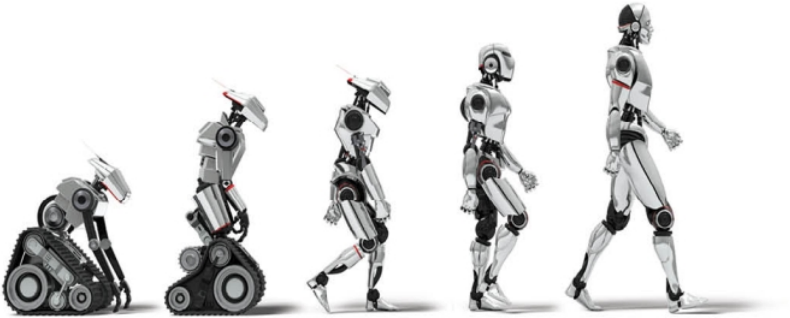
Prelude: towards Artificial General Intelligence (AGI)

Artificial entity capable of interacting and coexisting with its environment, especially humans:

- Comply to oral / written / visual instructions
- Initiate new decisions depending on environment
- Must be able to explain its actions (based on a rationale)
- Compliance to an overarching set of rules (morals, law, time/institution/task-dependent, etc.) likely to evolve
- Acknowledge its environment through "senses" (captors, . . .) and ability to preserve it (especially living creatures such as humans!)
- . . .

AGI could be embedded in physical agents (such as robots, vehicules)

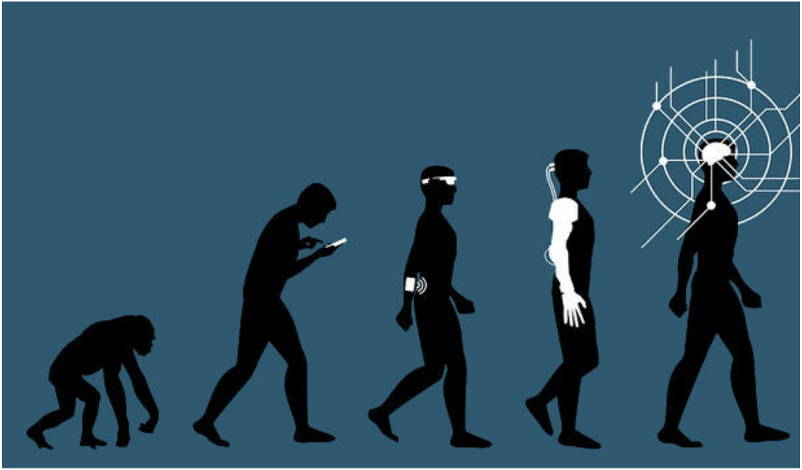
AGI could be embedded in physical agents (such as robots, vehicules)



Credits: blu.digital

AGI could be embedded in physical agents (such as robots, vehicules) or available through digital interfaces (computers).

AGI could be embedded in physical agents (such as robots, vehicules) or available through digital interfaces (computers).



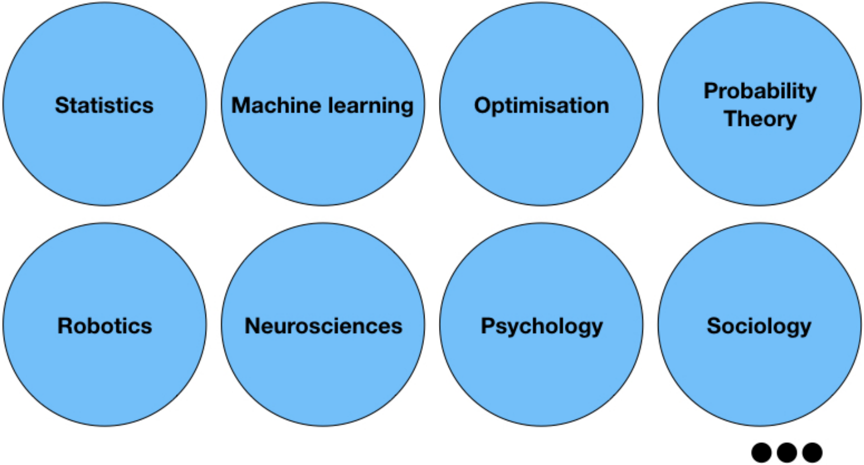
Credits: Gerd Leonhard

Can't be hard-programmed! Must be able to learn from previous sample tasks / data / situations / . . . and adapt its behaviour.

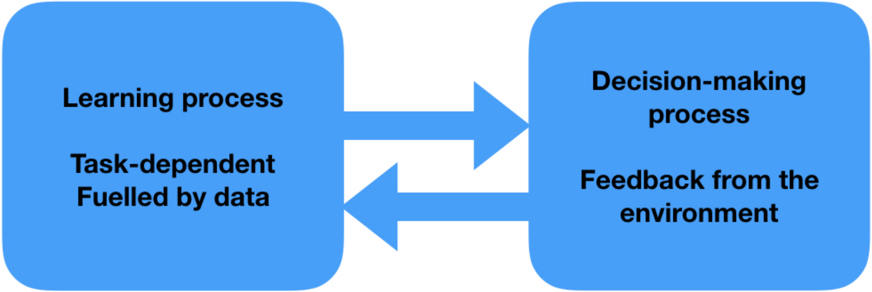
Can't be hard-programmed! Must be able to learn from previous sample tasks / data / situations / . . . and adapt its behaviour.

Must involve multi-disciplinary research efforts!

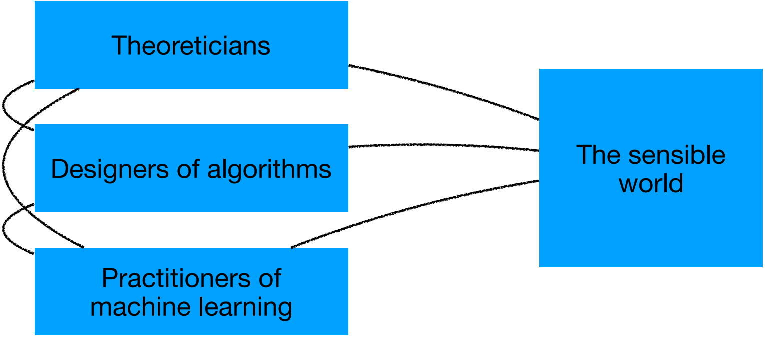
Some of the many fields involved in AGI



Among the many tasks needed to solve AGI, mostly interested in the learning + decision-making module.



It’s all connected



The big picture

The big picture

Solving AGI requires outstanding coordinated multi-disciplinary research efforts.

The big picture

Solving AGI requires outstanding coordinated multi-disciplinary research efforts.

Where do we mathematicians and computer scientists fit in?

Contribute to understanding and designing AGI systems machine learning, probability theory, optimisation, deep learning, computational statistics, reinforcement learning, ...

The big picture

Solving AGI requires outstanding coordinated multi-disciplinary research efforts.

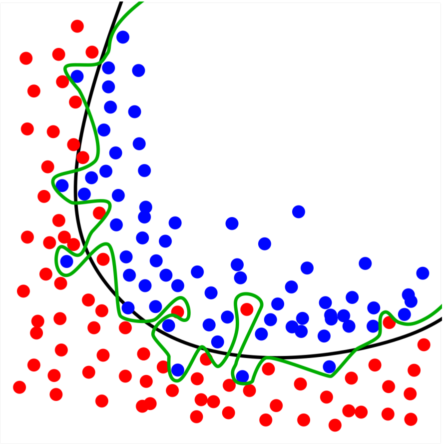
Where do we mathematicians and computer scientists fit in?

Contribute to understanding and designing AGI systems machine learning, probability theory, optimisation, deep learning, computational statistics, reinforcement learning, ...

What about me?
Personal research obsession: rethinking generalisation!

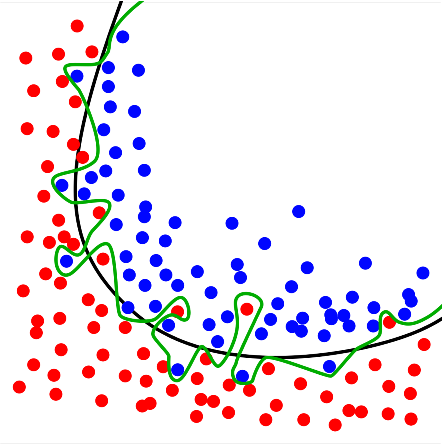
Learning is to be able to generalise

Learning is to be able to generalise



[Credits: Wikipedia]

Learning is to be able to generalise



[Credits: Wikipedia]

From examples, what can a system learn about the underlying phenomenon?

Memorising the already seen data is usually bad $\longrightarrow$ overfitting

Generalisation is the ability to 'perform' well on unseen data.

A few of those slides are inspired by our ICML 2019 tutorial, "A Primer on PAC-Bayesian Learning", Guedj and Shawe-Taylor

`https://bguedj.github.io/icml2019/index.html`

The simplest setting

Learning algorithm $\mathsf{A} : \mathcal{Z}^m \to \mathcal{H}$

- $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$
- $\mathcal{H}$ = hypothesis class

Training set (aka sample): $S_m = ((X_1, Y_1), \ldots, (X_m, Y_m))$
a finite sequence of input-output examples.

- Data-generating distribution $\mathbb{P}$ over $\mathcal{Z}$.
- Learner doesn't know $\mathbb{P}$, only sees the training set.
- The training set examples are *i.i.d.* from $\mathbb{P}$: $S_m \sim \mathbb{P}^m$

Generalisation

Loss function $\ell(h(X), Y)$ to measure the discrepancy between a predicted output $h(X)$ and the true output Y.

Empirical risk: (in-sample) $R_{\text{in}}(h) = \frac{1}{m} \sum_{i=1}^{m} \ell(h(X_i), Y_i)$

Theoretical risk: (out-of-sample) $R_{\text{out}}(h) = \mathbb{E}\left[\ell(h(X), Y)\right]$

Generalisation

Loss function $\ell(h(X), Y)$ to measure the discrepancy between a predicted output $h(X)$ and the true output Y.

Empirical risk: $R_{\text{in}}(h) = \frac{1}{m}\sum_{i=1}^{m} \ell(h(X_i), Y_i)$
(in-sample)

Theoretical risk: $R_{\text{out}}(h) = \mathbb{E}\big[\ell(h(X), Y)\big]$
(out-of-sample)

If predictor h does well on the in-sample (X, Y) pairs...

...will it still do well on out-of-sample pairs?

Generalisation gap: $\Delta(h) = R_{\text{out}}(h) - R_{\text{in}}(h)$

Upper bounds: with high probability $\Delta(h) \leqslant \epsilon(m, \delta)$

▶ $R_{\text{out}}(h) \leqslant R_{\text{in}}(h) + \epsilon(m, \delta)$

Flavours:

- distribution-free
- algorithm-free
- distribution-dependent
- algorithm-dependent

The PAC framework

PAC stands for Probably Approximately Correct.

Roughly translated: with high probability, the error of an hypothesis h is at most something we can control and even compute. For any $\delta > 0$,

$$\mathbb{P}\left[R_{\mathrm{out}}(h) \leqslant R_{\mathrm{in}}(h) + \epsilon(m, \delta) \right] \geqslant 1 - \delta.$$

Think of $\epsilon(m, \delta)$ as $\mathrm{Complexity} \times \frac{\log \frac{1}{\delta}}{\sqrt{m}}$.

The PAC framework

PAC stands for Probably Approximately Correct.

Roughly translated: with high probability, the error of an hypothesis h is at most something we can control and even compute. For any $\delta > 0$,

$$\mathbb{P}\left[R_{\mathrm{out}}(h) \leqslant R_{\mathrm{in}}(h) + \epsilon(m, \delta)\right] \geqslant 1 - \delta.$$

Think of $\epsilon(m, \delta)$ as $\mathrm{Complexity} \times \frac{\log \frac{1}{\delta}}{\sqrt{m}}$.

Rich literature on PAC generalisation bounds, for many machine learning algorithms in a variety of settings.

See Guedj (2019) for a recent survey on PAC-Bayes

Generalisation bounds are a safety check: they give a theoretical guarantee on the performance of a learning algorithm on any unseen data.

Generalisation bounds:

- provide a computable control on the error on any unseen data with prespecified confidence
- explain why specific learning algorithms actually work
- and even lead to designing new algorithm which scale to more complex settings

Is deep learning breaking statistical learning theory?

Is deep learning breaking statistical learning theory?

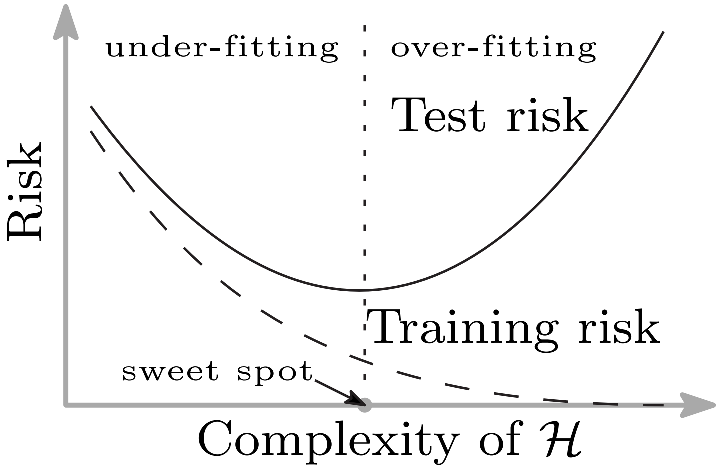
Neural networks architectures trained on massive datasets achieve zero training error which does not bode well for their performance: this strongly suggests overfitting...

Is deep learning breaking statistical learning theory?

Neural networks architectures trained on massive datasets achieve zero training error which does not bode well for their performance: this strongly suggests overfitting...

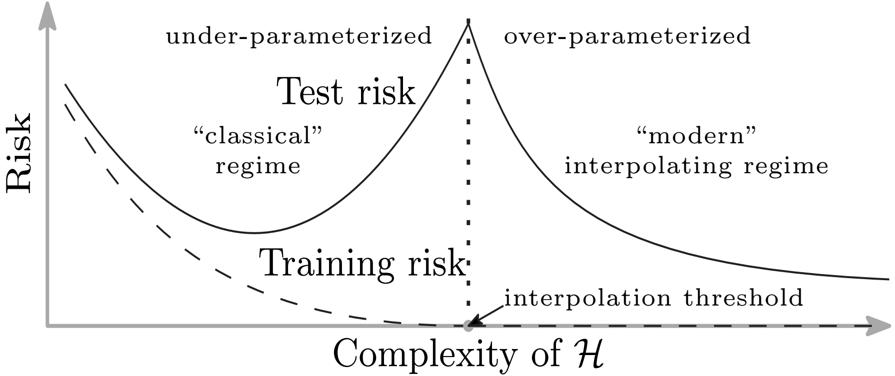
... yet they also achieve remarkably low errors on test sets!

A famous plot...



Belkin et al. (2019)

... which might just be half of the picture



Belkin et al. (2019)

The jigsaw problem

... a.k.a. representations matter.

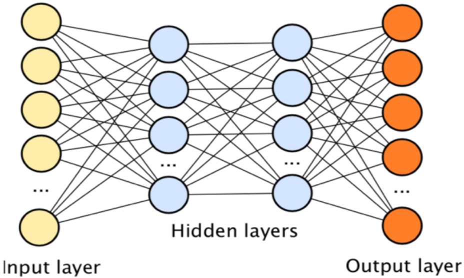
Fig. 1: What image representations do we learn by solving puzzles? Left: The image from which the tiles (marked with green lines) are extracted. Middle: A puzzle obtained by shuffling the tiles. Some tiles might be directly identifiable as object parts, but their identification is much more reliable once the correct ordering is found and the global figure emerges (Right).

Credits: Noroozi and Favaro (2016)

A tale of two learners – 1

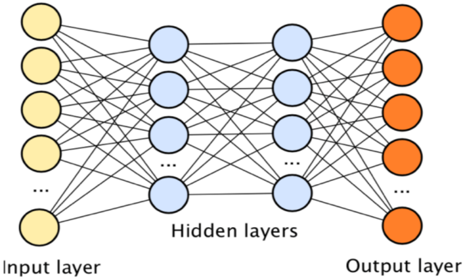
A tale of two learners – 1

Deep neural network



A tale of two learners – 1

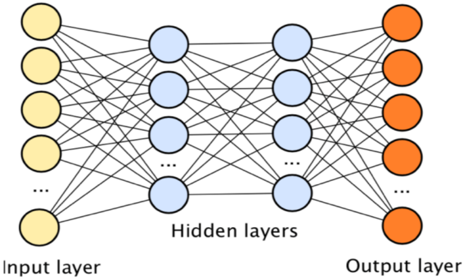
Deep neural network



Typically identifies a specific item (say, a horse) in an image with accuracy $> 99\%$.

A tale of two learners – 1

Deep neural network



Typically identifies a specific item (say, a horse) in an image with accuracy $>$ 99%.
Training samples: millions of annotated images of horses – GPU-expensive training.

A tale of two learners – 2

D. (2.5 yo)

A tale of two learners – 2

D. (2.5 yo)

Identifies horses with 100% accuracy.

A tale of two learners – 2

D. (2.5 yo)

Identifies horses with 100% accuracy.

Training samples: a handful of children books, bedtime stories and (poorly executed) drawings.

A tale of two learners – 2

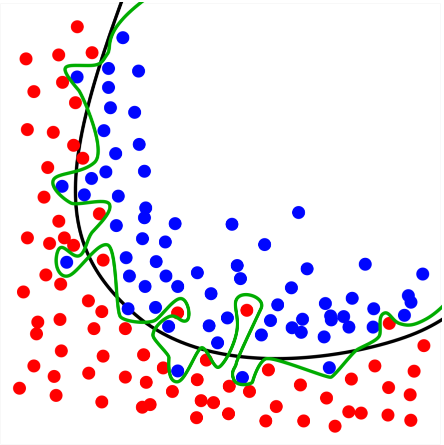
D. (2.5 yo)

Identifies horses with 100% accuracy.

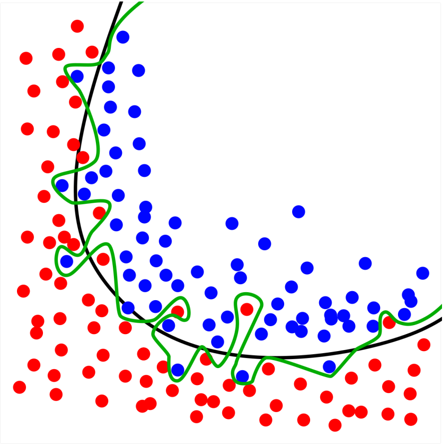
Training samples: a handful of children books, bedtime stories and (poorly executed) drawings.

Also expensive training.

Learning is to be able to generalise...

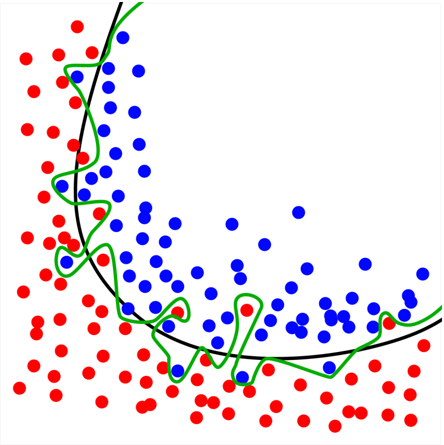


Learning is to be able to generalise...



... but not from scratch! AGI will not be solved by tackling each learning task as a fresh draw – must not be blind to context.

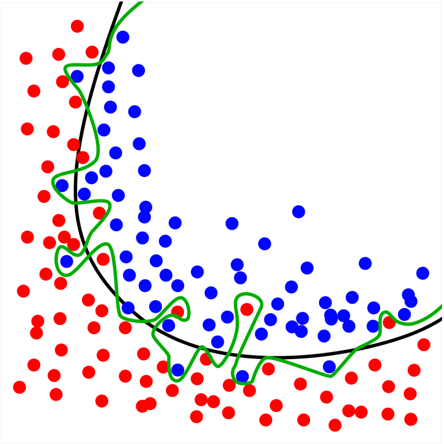
Learning is to be able to generalise...



... but not from scratch! AGI will not be solved by tackling each learning task as a fresh draw – must not be blind to context.

Need to incorporate structure / semantic information / implicit representations of the ”sensible” world.

Learning is to be able to generalise...

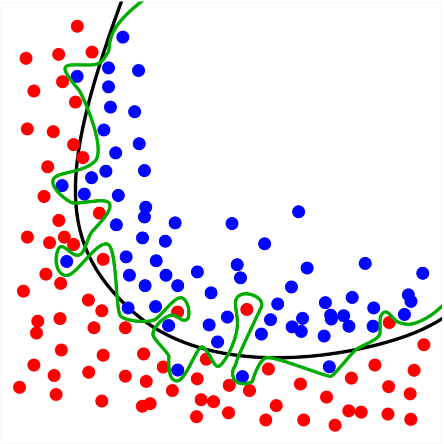


... but not from scratch! AGI will not be solved by tackling each learning task as a fresh draw – must not be blind to context.

Need to incorporate structure / semantic information / implicit representations of the ”sensible” world.

Potential game-changer for algorithms design (more ”intelligent”, resources-efficient, etc.) and practitioners.

Learning is to be able to generalise...



... but not from scratch! AGI will not be solved by tackling each learning task as a fresh draw – must not be blind to context.

Need to incorporate structure / semantic information / implicit representations of the ”sensible” world.

Potential game-changer for algorithms design (more ”intelligent”, resources-efficient, etc.) and practitioners.

Very exciting research avenue for theoreticians for the next decade(s)!

Going further

B. Guedj and J. Shawe-Taylor. ”A Primer on PAC-Bayesian Learning”, ICML 2019 tutorial, `https://bguedj.github.io/icml2019/index.html`

An excellent book: Valiant (2013), *Probably Approximately Correct: Nature's Algorithms for Learning and Prospering in a Complex World*.

Connect with the UCL Centre for Artificial Intelligence (home to our UKRI Centre for Doctoral Training in Foundational Artificial Intelligence)
`https://www.ucl.ac.uk/ai-centre/`

Thanks!

Feel free to reach out!

https://bguedj.github.io

b.guedj@ucl.ac.uk

@bguedj

Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal. Reconciling modern machine learning practice and the bias-variance trade-off. *Proceedings of the National Academy of Sciences*, 116(32):15849–15854, 2019.

P. Germain, A. Lacasse, F. Laviolette, and M. Marchand. PAC-Bayesian learning of linear classifiers. In *Proceedings of the 26th Annual International Conference on Machine Learning, ICML*, 2009.

Benjamin Guedj. A primer on PAC-Bayesian learning. *arXiv preprint arXiv:1901.05353*, 2019.

Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. *Lecture Notes in Computer Science*, pages 69–84, 2016. doi: 10.1007/978-3-319-46466-4_5. URL http://dx.doi.org/10.1007/978-3-319-46466-4_5.

Leslie Valiant. *Probably Approximately Correct: Nature's Algorithms for Learning and Prospering in a Complex World*. Basic Books, Inc., USA, 2013. ISBN 0465032710.

Case study:
Generalisation bounds for deep neural networks

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

https://arxiv.org/abs/1905.10259

Context

Context

- **PAC-Bayes** has been successfully used to analyse and understand generalisation abilities of machine learning algorithms.

Context

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

Context

- **PAC-Bayes** has been successfully used to analyse and understand generalisation abilities of machine learning algorithms.
  - Guedj (2019), "A Primer on PAC-Bayesian Learning", Proceedings of the French Mathematical Society, `https://arxiv.org/abs/1901.05353`
  - Guedj & Shawe-Taylor (2019), "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

- **PAC-Bayes** has been successfully used to analyse and understand generalisation abilities of machine learning algorithms.
  - Guedj (2019), "A Primer on PAC-Bayesian Learning", Proceedings of the French Mathematical Society, `https://arxiv.org/abs/1901.05353`
  - Guedj & Shawe-Taylor (2019), "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. Guedj, F. Laviolette. *Dichotomize and Generalize: PAC-Bayesian Binary Activated Deep Neural Networks*, NeurIPS 2019

https://arxiv.org/abs/1905.10259

This spotlight

G. Letarte, P. Germain, B. Guedj, F. Laviolette. *Dichotomize and Generalize: PAC-Bayesian Binary Activated Deep Neural Networks*, 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. Guedj, F. Laviolette. *Dichotomize and Generalize: PAC-Bayesian Binary Activated Deep Neural Networks*, 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. Guedj, F. Laviolette. *Dichotomize and Generalize: PAC-Bayesian Binary Activated Deep Neural Networks*, 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. Guedj, F. Laviolette. *Dichotomize and Generalize: PAC-Bayesian Binary Activated Deep Neural Networks*, 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. Guedj, F. Laviolette. *Dichotomize and Generalize: PAC-Bayesian Binary Activated Deep Neural Networks*, 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. Guedj, F. Laviolette. *Dichotomize and Generalize: PAC-Bayesian Binary Activated Deep Neural Networks*, 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. Guedj, F. Laviolette. *Dichotomize and Generalize: PAC-Bayesian Binary Activated Deep Neural Networks*, 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

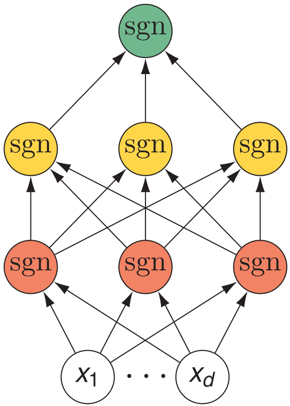
- $\mathbf{x} \in \mathbb{R}^{d_0}$, $y \in \{-1, 1\}$

Architecture:

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

Parameters:

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



Prediction

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

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$_{\text{pre}}$					
– pretrain	linear loss (20 epochs)	50%	-	-	random init
– final	PAC-Bayes bound	50%	-	PAC-Bayes bound	pretrain

Dataset	MLP–tanh		PBGNet$_\ell$		PBGNet			PBGNet$_{\text{pre}}$		
	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