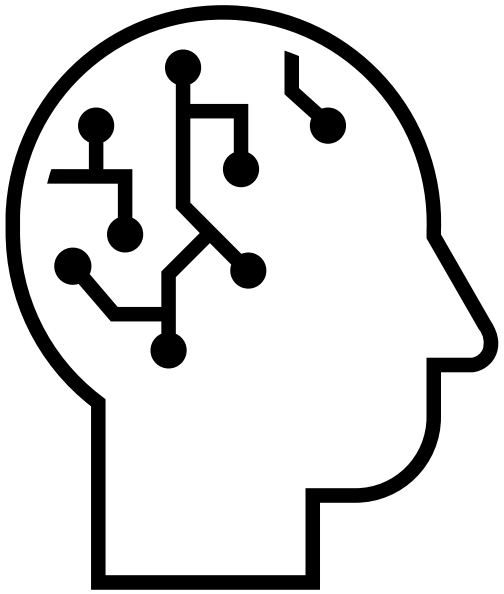
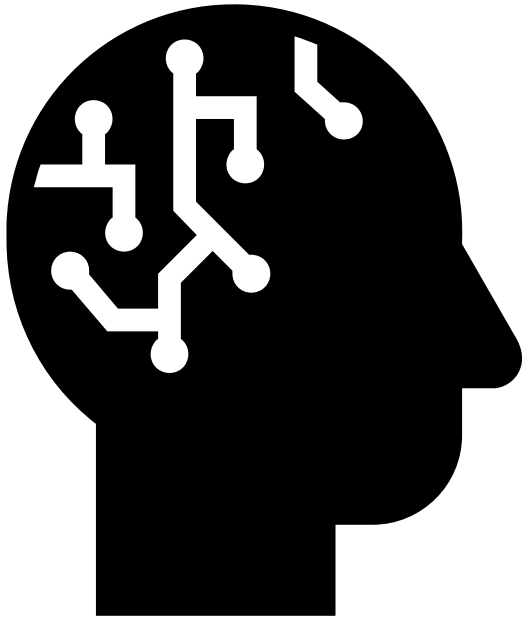
Learnability of Complex Objects in Modern AI

Maria-Florina (Nina) Balcan
Carnegie Mellon University

Machine Learning/AI are Everywhere

Ineluctable, inevitable, inescapable, unstoppable.

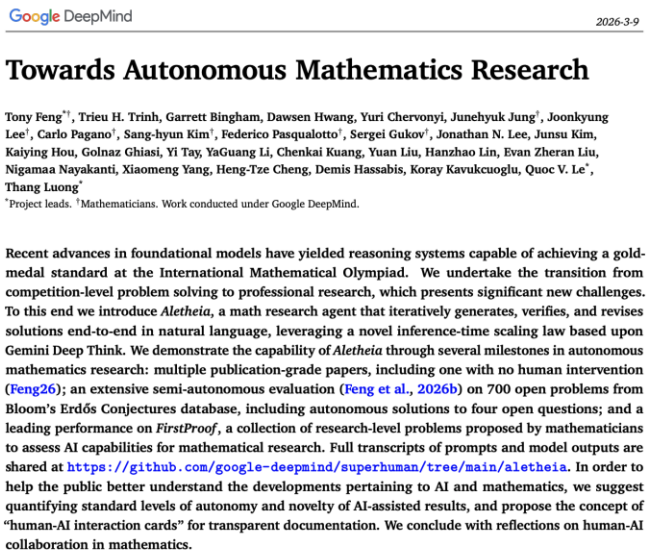




Current AI: More Complex Systems

Used for much **more complex tasks**. E.g.,

- solving math: OpenAI gold medal-level performance IMO, research open questions (Erdos's pbs or **COLT open pbs**).
- algorithms for solving pbs hard in classic (TCS) models.
- agents for planning events.
- research agendas.



Google DeepMind 2026-3-9

Towards Autonomous Mathematics Research

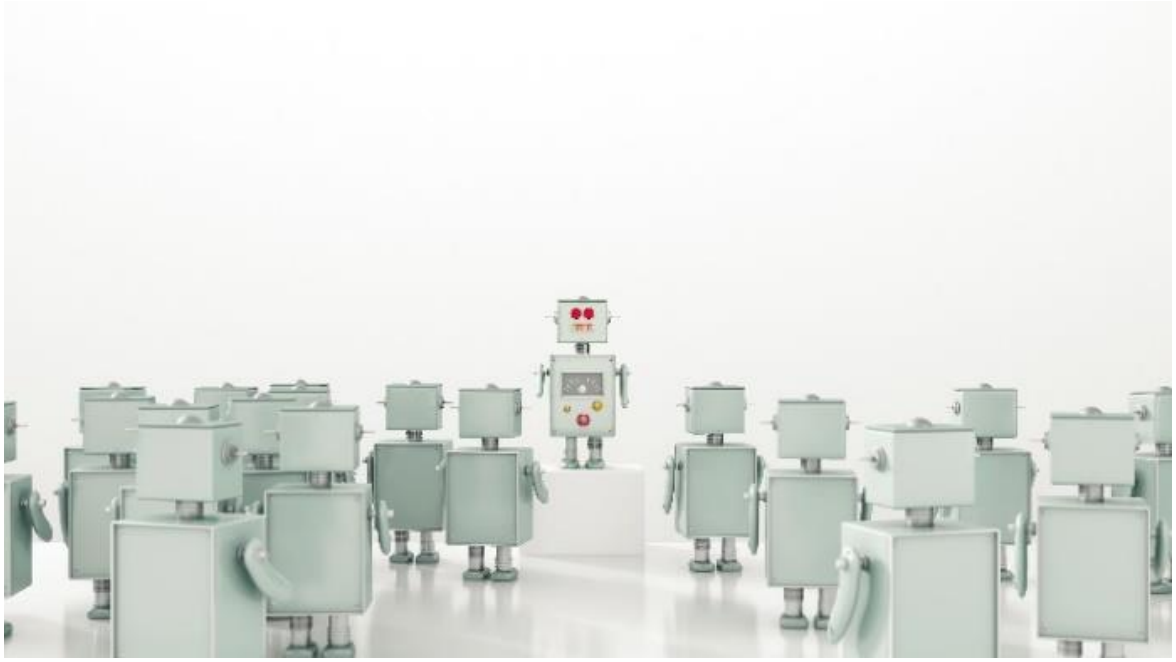
Tony Feng*†, Trieu H. Trinh, Garrett Bingham, Dawsen Hwang, Yuri Chervonyi, Junehyuk Jung†, Joonkyung Lee†, Carlo Pagano†, Sang-hyun Kim†, Federico Pasqualotto†, Sergei Gukov†, Jonathan N. Lee, Junsu Kim, Kaiying Hou, Golnaz Ghiasi, Yi Tay, YaGuang Li, Chenkai Kuang, Yuan Liu, Hanzhao Lin, Evan Zheran Liu, Nigamaa Nayakanti, Xiaomeng Yang, Heng-Tze Cheng, Demis Hassabis, Koray Kavukcuoglu, Quoc V. Le*, Thang Luong*

*Project leads. †Mathematicians. Work conducted under Google DeepMind.

Recent advances in foundational models have yielded reasoning systems capable of achieving a gold-medal standard at the International Mathematical Olympiad. We undertake the transition from competition-level problem solving to professional research, which presents significant new challenges. To this end we introduce *Aletheia*, a math research agent that iteratively generates, verifies, and revises solutions end-to-end in natural language, leveraging a novel inference-time scaling law based upon Gemini Deep Think. We demonstrate the capability of *Aletheia* through several milestones in autonomous mathematics research: multiple publication-grade papers, including one with no human intervention (Feng26); an extensive semi-autonomous evaluation (Feng et al., 2026b) on 700 open problems from Bloom's Erdős Conjectures database, including autonomous solutions to four open questions; and a leading performance on *FirstProof*, a collection of research-level problems proposed by mathematicians to assess AI capabilities for mathematical research. Full transcripts of prompts and model outputs are shared at https://github.com/google-deepmind/superhuman/tree/main/aletheia. In order to help the public better understand the developments pertaining to AI and mathematics, we suggest quantifying standard levels of autonomy and novelty of AI-assisted results, and propose the concept of "human-AI interaction cards" for transparent documentation. We conclude with reflections on human-AI collaboration in mathematics.

Current AI: More Complex Systems

- Used for much **more complex tasks**.
  - E.g. proofs, plans, research agendas, etc.
- **Learning with other agents** with complex utilities.
  - E.g. in security applications.



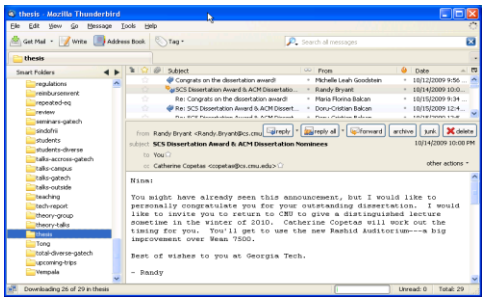
Current AI: More Complex Systems

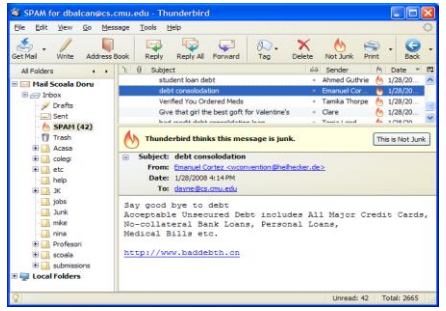
- Used for much **more complex tasks**.
  - E.g. proofs, plans, research agendas, etc.
- **Learning with other agents** with complex utilities.
- Used for **much more high-stakes domains**.
  - Diplomacy (international relations).
  - Medical domains.
  - Banking, etc.

Classic ML Theory

A single learner, simple task (e.g., classification or regression).

E.g., spam detection:



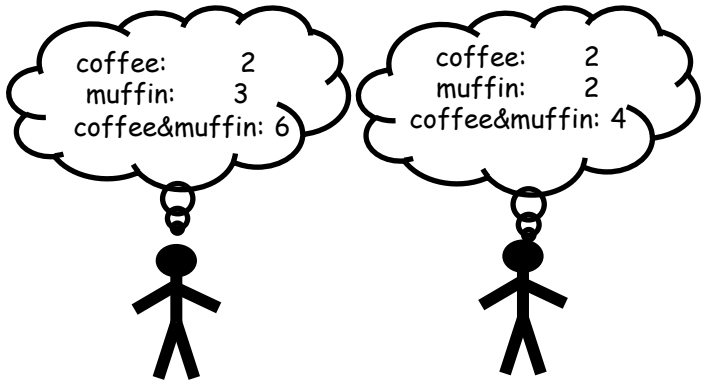


Powered by seminal results in:

- **Optimization/algorithms** on training data.
- **Generalization** on unseen data.

Classic AI Theory

- **Multi-agents systems: game theory lenses.**
  - static solution concepts (equilibria).
  - simple learners (e.g., humans).



- **Algorithms**: hand-designed, often worst-case guarantees.

Current AI: More Complex Systems

- Used for much **more complex tasks**.
- **Learning with other agents** with complex utilities.
- Used for **much more high-stakes domains**.

Multi-faceted approach:

- extend ideas from many fields (learning theory, game theory, algorithms, …) to analyze and better design AI.

Structure of the talk

Learning rich structured objects via dual function classes.

- Machine Learning for Algorithm Design.
  - Learning algorithms for solving pbs that remain hard in classic frameworks of TCS, OR, Economics, etc.
- Hyperparameter tuning of ML.

Learning in the presence of other learners.

- **Strategic settings** (with commitment).
- **Cooperative environments** (verifiers & generators for LLMs).

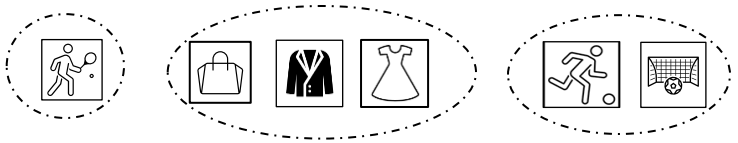
Learning rich structured objects via dual function classes

Machine Learning for Algorithm Design

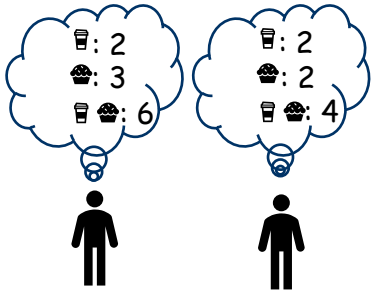
Algorithm: (finite) sequence of precise step-by-step instructions to solve a well specified class of problems.

E.g.: algorithms for solving combinatorial problems.

- clustering, partitioning



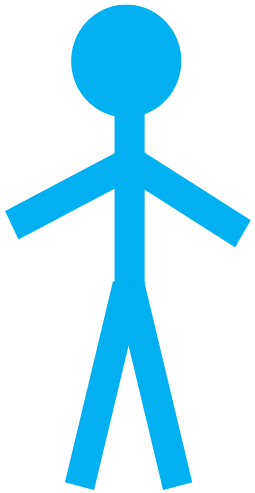
- pricing, auction design



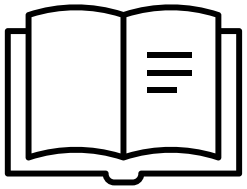
- logistics, scheduling, …

Classic Design and Analysis of Algorithms

- Algorithm **hand-designed**, stroke of genius.



- **Worst-case analysis**:



algorithm must succeed on **worst-case instances** of the problem.

- For many problems, guarantees are weak (e.g., for solution quality, or running time, or other performance measures).

Design and Analysis of Algorithms

Classic Approach

1. Algorithm **hand-designed**, stroke of genius.

2. **Worst-case analysis**: algorithm must succeed on **worst-case instances** of the problem.

Many problems hard in classic frameworks.

Modern Approach

Machine Learning for Algorithm Design.

LLMs for Algorithm Design.

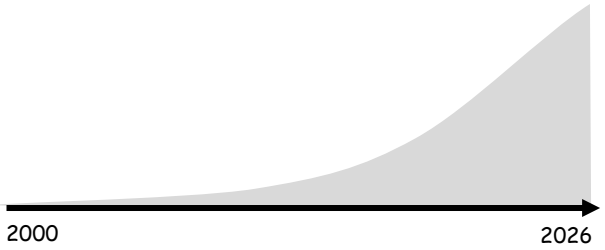
Agents for Algorithm Design.

What provable guarantees can we provide?

Machine Learning for Algorithm Design

ML for algo design: use learning & data for algo design.

- often repeatedly solve instances of the same algo problem.



Early Work:

- **Rich history of empirical work.**

  AI, Computational Biology, Game Theory

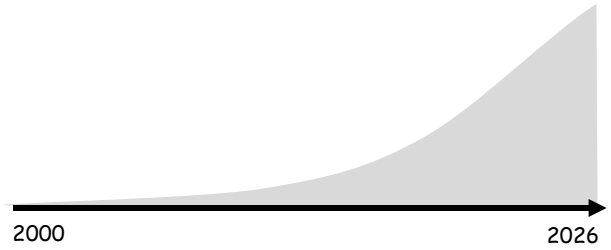
  [Horvitz-Ruan-Gomes-Kautz-Selman-Chickering, 2001] [DeBlasio-Kececioglu, 2018]

  [Xu-Hutter-Hoos-LeytonBrown, 2008] [Likhodedov and Sandholm, 2004]

- **Some case study theory work.**
  - Mechanism Design via ML [Balcan-Blum-Hartline-Mansour, FOCS 2005]
  - Subset Selection Pbs [Gupta-Roughgarden, ITCS'16 &SICOMP'17]

Machine Learning for Algorithm Design

Prior Work: largely empirical

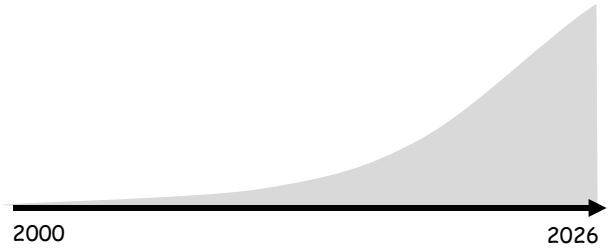


Our Recent Work:

- Systematic learning theoretic analysis.
- Numerous cases studies and general principles.

Machine Learning for Algorithm Design

Prior Work: largely empirical

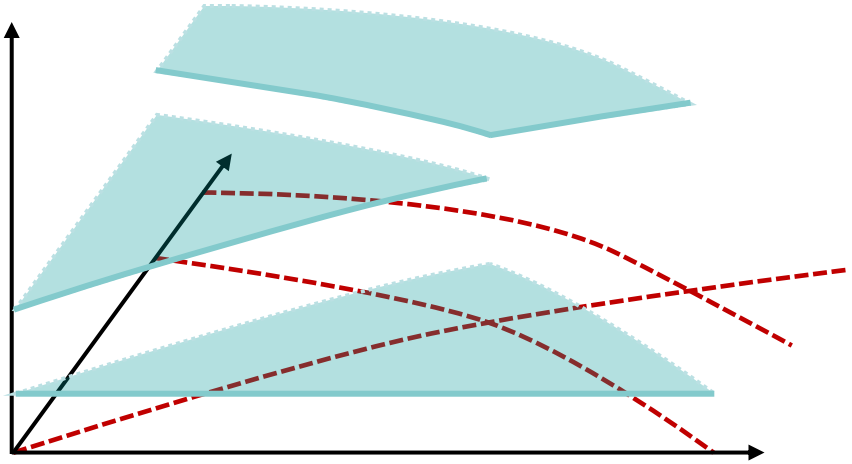


Our Recent Work:

- Systematic learning theoretic analysis.
- Numerous cases studies and general principles (via dual function classes).

Cool learning theory:

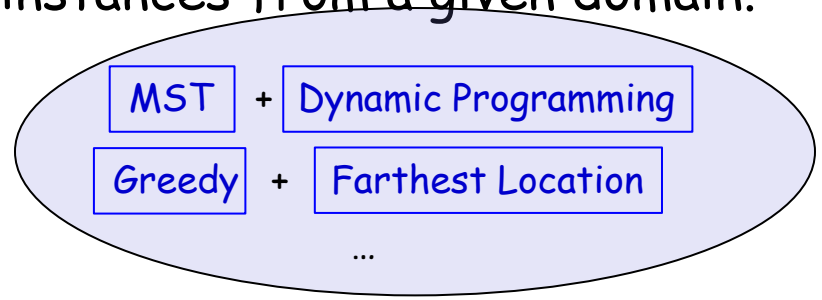
- New connections
  - OR, Optimization, TCS, Econ,….
- New function classes with challenging structure.



Algorithm Design as Distributional Learning

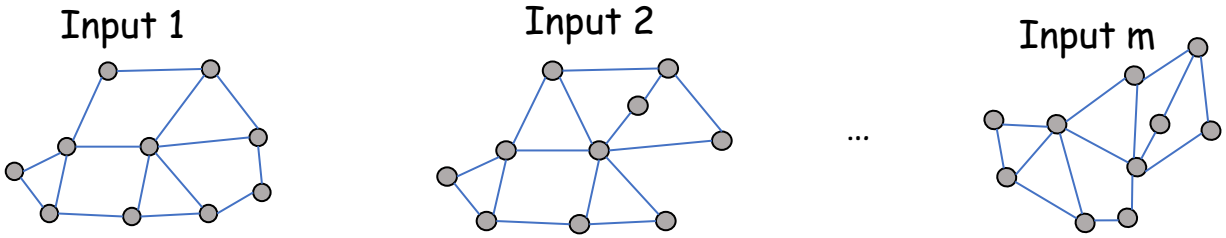
Data-driven algo design: directly learn an algorithm (from a parametric family of algos) that does well on instances from a given domain.

Large family $\mathbf{F}$ of algorithms



Sample of typical inputs

Clustering:



Pricing:

Input 1		
$v_1(C)$	...	$v_n(C)$
$v_1(M)$		$v_n(M)$
$v_1(C\&M)$		$v_n(C\&M)$

...

Input m		
$v_1(C)$	...	$v_n(C)$
$v_1(M)$		$v_n(M)$
$v_1(C\&M)$		$v_n(C\&M)$

Knapsack:

$(v_1 s_1), \ldots, (v_n s_n), C$ ... $(v_1 s_1), \ldots, (v_n s_n), C$

Data-driven algorithm design: Problem Setup

"Data-driven algorithm design", M.F. Balcan, book chapter in "Beyond the Worst-Case Analysis of Algorithms", Cambridge University Press, 2020.

[Gupta-Roughgarden, ITCS'16 &SICOMP'17]

- Fix an algorithmic pb (e.g., subset selection or clustering).
- Let Π be the set of problem instances for this problem.

  Let Alg be a family of algos, parameterized by set $P \subseteq R^d$; A_α the algo in Alg parametrized by $\alpha \in P$.

- Fix a utility function $u: \Pi \times P \to [0, H]$ where $u(I, \alpha)$ measures the performance of algo A_α on problem instance I.
  - $u_\alpha: \Pi \to [0, H]$ induced by A_α, where $u_\alpha(I) = u(I, \alpha)$.

Example: Knapsack Problem

[Gupta-Roughgarden, ITCS'16 &SICOMP'17]

Input: An instance I consists of n items (each item i has a value v_i and a size s_i), and knapsack capacity C.

Output: select most valuable subset of items that fits. Find subset V to maximize $\sum_{i \in V} v_i$ subject to $\sum_{i \in V} s_i \leq C$.

Alg : greedy algos parametrized by $P = R$.

For $\alpha \in P$, algo A_α:

- Set score of item i to be v_i / s_i^α.
- In decreasing order of score, add each item to the knapsack if there is enough capacity left.

$u(I, \alpha)$ = value of items chosen by A_α on I.

Example: Clustering Problems

[Balcan-Nagarajan-Vitercik-White, COLT'17] [Balcan-Dick-Lang, ICLR'20]

Input: I consists of set of objects S, distance fnc d.

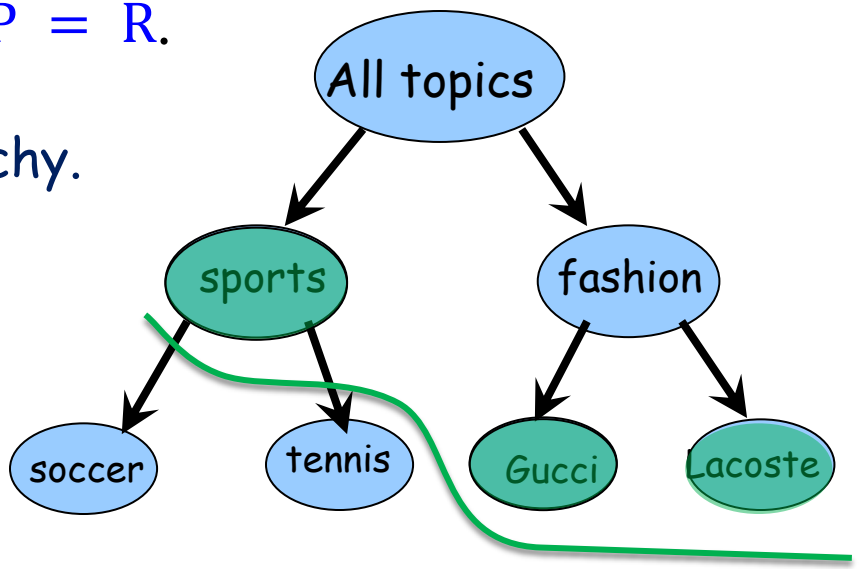
Output: centers $\{c_1, c_2, \ldots, c_k\}$ that min $\sum_p \min_i d^2(p, c_i)$

Alg : family of algos, linkage+DP, $P = R$.

1. Greedy linkage-based, get hierarchy.

   E.g., $\text{dist}_\alpha(A, B) = (1-\alpha)\text{SL} + \alpha \max_{x \in A, x' \in B} d(x, x')$

2. Fixed algo (e.g., DP or last k-merges) to select a good pruning.

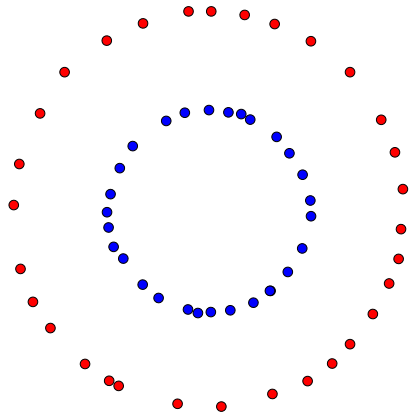


$u(I, \alpha)$ = k-means value of clustering output by A_α on I.

Different Algos Work in Different Settings

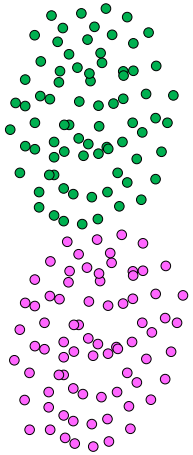
Assume 2 target clusters.

Instance 1



Easy for Single-Linkage
Hard for Complete-Linkage

Instance 2



Hard for Single Linkage
Easy for Complete-Linkage

Data-driven algorithm design: Problem Setup

- Specific domain: unknown input distribution D over Π.
- Learning algo uses m i.i.d. samples $I_1, I_2, \ldots I_m \sim D$ to find an algo $A_\alpha \in Alg$ for future inputs from D.

  (can measure $u_\alpha(I)$ of each algo $A_\alpha \in Alg$ on each input I_i)

- Goal: output an algo of Alg that performs almost as well as the optimal algorithm $A_{\alpha^*} \in Alg$ for D that maximizes

  $E_{I \sim D}[u_\alpha(I)]$ over $A_\alpha \in Alg$.

- Typical approach: pick $\hat{A}$ that does well over the sample.

Sample Complexity: How large should training set be to guarantee that algos that do well over training set do well on new instances?

Uniform Convergence

Uniform convergence: for any algo in Alg, average performance over samples "close" to its expected performance.

- Imply that $\widehat{\mathbf{A}}$ that does best over the sample has high expected performance.

Learning theoretic notion of dimension, e.g., pseudo-dimension

Theorem

$m = O(\dim(\mathbf{F})/\epsilon^2)$ suffices so that for any distribution D over Π, with prob. at least $1-\delta$ over the draw $\{I_1, \ldots, I_m\} \sim D$, for all algos $A_\alpha \in \mathrm{Alg}$,

$$\left| \mathbb{E}_{I \sim D}[u_\alpha(I)] - \frac{1}{m}\sum_{i=1}^{m} u_\alpha(I_i) \right| \leq \epsilon$$

General Sample Complexity via Dual Classes

Theorem (informally) [Balcan-DeBlasio-Kingsford-Dick-Sandholm-Vitercik, STOC 2021&JACM 2024]

Technique for analyzing $\dim(\{u_\alpha(\cdot) : \text{param } \alpha\})$ that takes advantage of the structure of **dual class** $\{u_I(\cdot): \text{instances } I\}$.

Given $u: \Pi \times P \to [0, H]$, $u(I, \alpha)$ = performance of A_α on I.

- $u_\alpha: \Pi \to [0, H]$ induced by A_α, $u_\alpha(I) = u(I, \alpha)$.
- $u_I: P \to [0, H]$ induced by I, $u_I(\alpha) = u(I, \alpha)$.

$\{u_I: \text{instances } I\}$ dual class for of $\{u_\alpha(\cdot) : \text{param } \alpha\}$

General Sample Complexity via Dual Classes

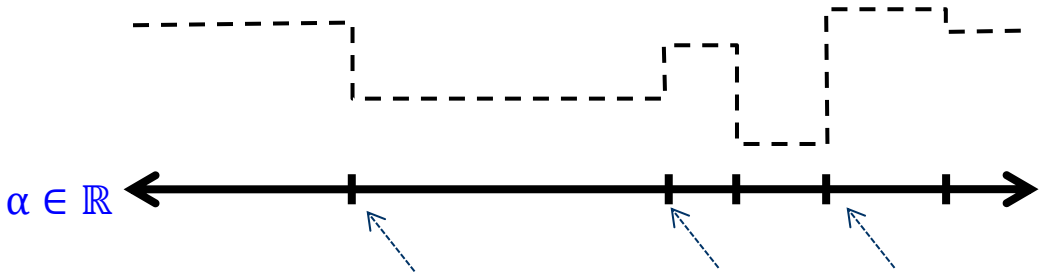
Theorem (informally) [Balcan-DeBlasio-Kingsford-Dick-Sandholm-Vitercik, STOC 2021&JACM 2024]

Technique for analyzing $\dim(\{u_\alpha(\cdot) : \text{param } \alpha\})$ that takes advantage of the structure of **dual class** $\{u_I(\cdot): \text{instances } I\}$.

Key motivation: can often show u_I is structured.

Example: knapsack, greedy family Alg, u_I piece-wise linear:

[Gupta-Roughgarden, ITCS'16 &SICOMP'17]



Critical points of the form $\frac{\log\left(\frac{v_i}{v_j}\right)}{\log\left(\frac{s_i}{s_j}\right)}$, so $O(n^2)$ pieces.

General Sample Complexity via Dual Classes

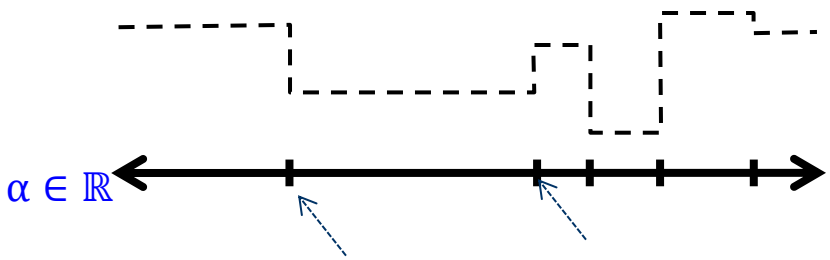
Theorem (informally) [Balcan-DeBlasio-Kingsford-Dick-Sandholm-Vitercik, STOC 2021&JACM 2024]

Technique for analyzing $\dim(\{u_\alpha(\cdot) : \text{param } \alpha\})$ that takes advantage of the structure of **dual class** $\{u_I(\cdot): \text{instances } I\}$.

Example: clustering, parametrized-linkage, u_I piece-wise linear

[Balcan-Nagarajan-Vitercik-White, COLT'17] [Balcan-Dick-Lang, ICLR'20]

1. Greedy linkage-based, get hierarchy. $\text{dist}_\alpha(A, B) = (1-\alpha)\text{SL} + \alpha \max_{x \in A, x' \in B} d(x, x')$
2. Fixed algo to select a good pruning.



$O(n^8)$ pieces.

Roots of linear eqs where we merge one pair vs another pair of clusters.

General Sample Complexity via Dual Classes

Theorem (informally) [Balcan-DeBlasio-Kingsford-Dick-Sandholm-Vitercik, STOC 2021&JACM 2024]

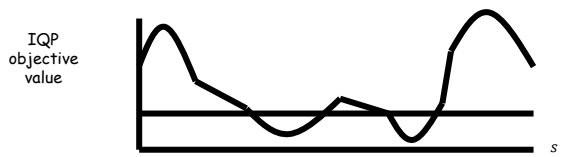
Technique for analyzing $\dim(\{u_\alpha(\cdot) : \text{param } \alpha\})$ that takes advantage of the structure of **dual class** $\{u_I(\cdot): \text{instances } I\}$.

Key motivation: can often show u_I is structured.

Partitioning Pbs via IQPs

SDP + s-linear rounding

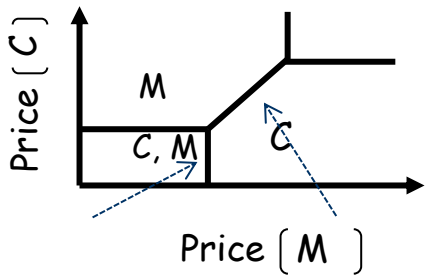
[Balcan-Nagarajan-Vitercik-White, COLT 2017]



$\alpha \in \mathbb{R}$

Posted Pricing, Two-Part Tariffs, Parametrized VCG auctions, etc.

[Balcan-Sandholm-Vitercik, EC'18]



Decision boundary where the buyer prefers one bundle over the other, is a hyperplane.

Pseudo-dimension (for real valued classes)

VC-dim of H is cardinality of the largest set S that can be labeled in all possible ways $2^{|S|}$ by H.

[If arbitrarily large finite sets can be shattered by H, then $\mathrm{VCdim}(H) = \infty$]

The **pseudo-dimension** [Pollard 1984] of F is cardinality of the largest set $S = \{x_1, \ldots, x_m\}$ and thresholds $y_1, \ldots, y_m$ s.t. all 2^m above/below patterns can be achieved by functions $f \in F$.

- $m = 2$, need $f_1 \in F$ s.t. $f_1(x_1) < y_1, f_1(x_2) < y_2$; $f_2 \in F$ s.t. $f_2(x_1) > y_1$, $f_2(x_2) < y_2$; $f_3 \in F$ s.t. $f_3(x_1) < y_1, f_3(x_2) > y_2$,; $f_4 \in F$ s.t. $f_4(x_1) > y_1, f_4(x_2) > y_2$

Pseudo-dimension of F is VC-dim of class of "below-the-graph" indicator functions $\{B_f(x, y) = \mathrm{sgn}(f(x) - y) : f \in F\}$

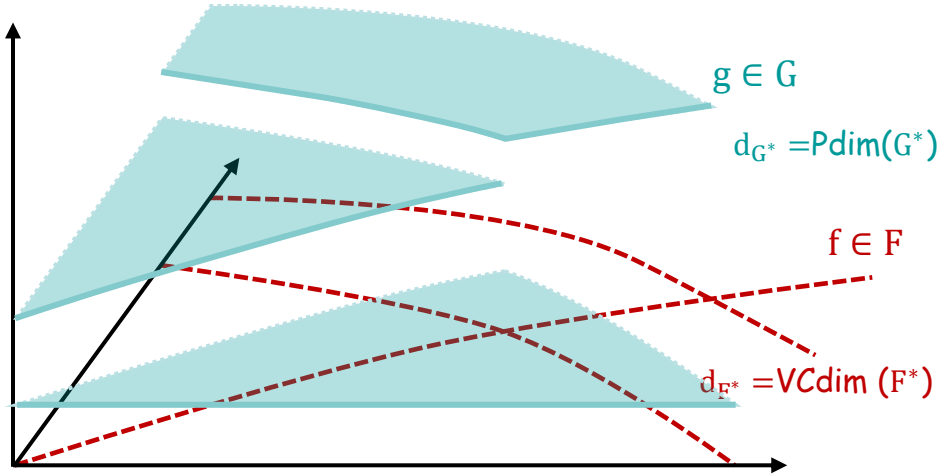
General Sample Complexity via Dual Classes

Theorem [Balcan-DeBlasio-Kingsford-Dick-Sandholm-Vitercik, STOC 2021&JACM 2024]

Suppose for each $u_I(\alpha)$ there are $\leq N$ boundary fns $f_1, f_2, \ldots \in F$ s. t within each region defined by them, $\exists\, g \in G$ s.t. $u_I(\alpha) = g(\alpha)$.

$$\text{Pdim}(\{u_\alpha(I)\}) = \tilde{O}\left((d_{F^*} + d_{G^*}) + d_{F^*} \log N\right)$$

d_{F^*} =VCdim of dual of F, d_{G^*} =Pdim of dual of G.



General Sample Complexity via Dual Classes

Theorem [Balcan-DeBlasio-Kingsford-Dick-Sandholm-Vitercik, STOC 2021&JACM 2024]

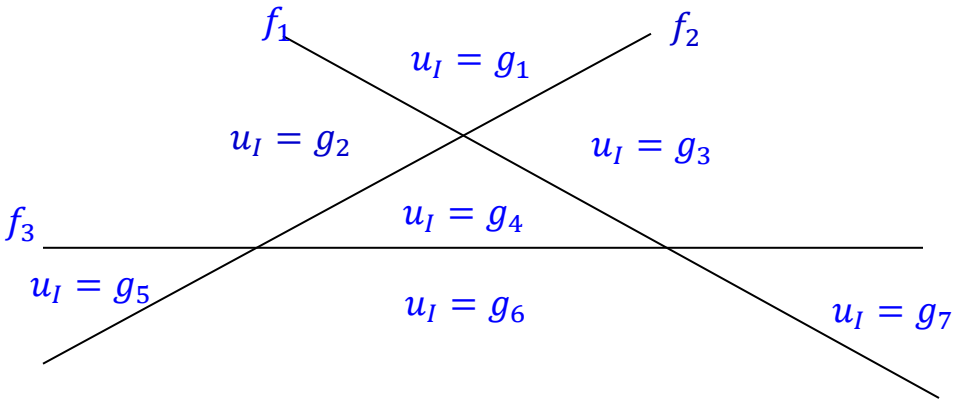
Suppose for each $u_I(\alpha)$ there are $\leq N$ boundary fns $f_1, f_2, \ldots \in F$ s. t within each region defined by them, $\exists\, g \in G$ s.t. $u_I(\alpha) = g(\alpha)$.

$$\text{Pdim}(\{u_\alpha(I)\}) = O\big((d_{F^*} + d_{G^*}) \log(d_{F^*} + d_{G^*}) + d_{F^*} \log N\big)$$

d_{F^*} =VCdim of dual of F, d_{G^*} =Pdim of dual of G.

VCdim(F): fix m pts. Bound # of labelings of these pts by $f \in F$ via Sauer's lemma in terms of VCdim(F).

VCdim(F^*): fix N fns, want to bound # regions induced. There is a direct correspondence between the shattering coefficient of the dual and the number of regions induced by these fns. Can use Sauer's lemma in terms of VCdim(F^*).



General Sample Complexity via Dual Classes

Theorem [Balcan-DeBlasio-Kingsford-Dick-Sandholm-Vitercik, STOC 2021&JACM 2024]

Suppose for each $u_I(\alpha)$ there are $\leq N$ boundary fns $f_1, f_2, \ldots \in F$ s. t within each region defined by them, $\exists\, g \in G$ s.t. $u_I(\alpha) = g(\alpha)$.

$$\text{Pdim}(\{u_\alpha(I)\}) = O\big((d_{F^*} + d_{G^*}) \log(d_{F^*} + d_{G^*}) + d_{F^*} \log N\big)$$

d_{F^*} =VCdim of dual of F, d_{G^*} =Pdim of dual of G.

Proof insights:

- Fix D instances $I_1, \ldots, I_D$ and D thresholds $z_1, \ldots, z_D$. Bound # sign patterns $(u_\alpha(I_1), \ldots, u_\alpha(I_D))$ ranging over all α. Equivalently, $(u_{I_1}(\alpha), \ldots, u_{I_D}(\alpha))$.
- Use VCdim of F^* , bound # of regions induced by $u_{I_1}(\alpha), \ldots, u_{I_D}(\alpha)$: $(eND)^{d_{F^*}}$.
- On a region, bound # of sign patterns in that region by $(eD)^{d_{G^*}}$.
- Combining, total of $(eND)^{d_{F^*}}(eD)^{d_{G^*}}$. Set to 2^D and solve.

General Sample Complexity via Dual Classes

Theorem [Balcan-DeBlasio-Kingsford-Dick-Sandholm-Vitercik, STOC 2021&JACM 2024]

Suppose for each $u_I(\alpha)$ there are $\leq N$ boundary fns $f_1, f_2, \ldots \in F$ s. t within each region defined by them, $\exists\, g \in G$ s.t. $u_I(\alpha) = g(\alpha)$.

$$\text{Pdim}(\{u_\alpha(I)\}) = O\big((d_{F^*} + d_{G^*}) \log(d_{F^*} + d_{G^*}) + d_{F^*} \log N\big)$$

d_{F^*} =VCdim of dual of F, d_{G^*} =Pdim of dual of G.

Proof insights:

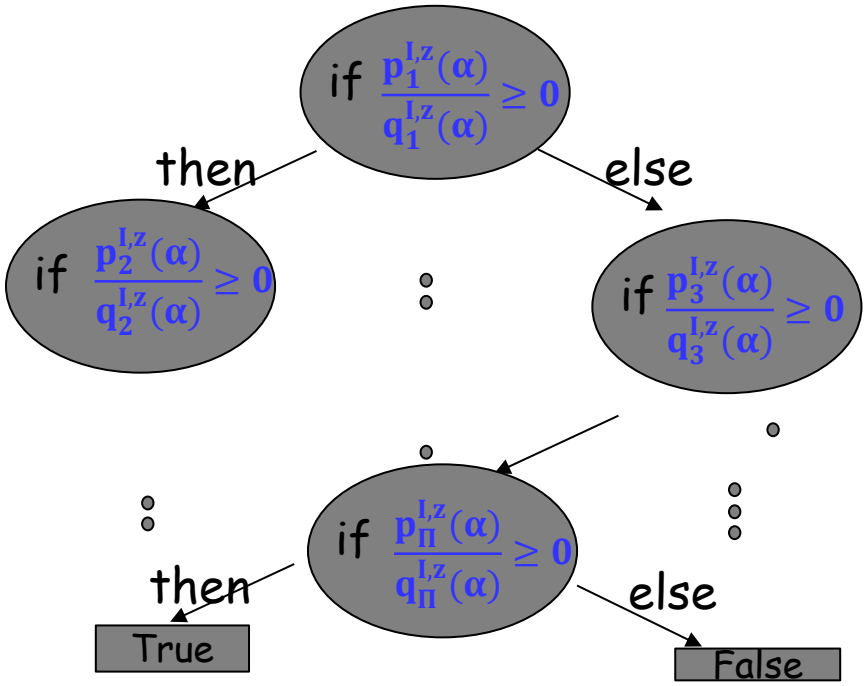
- Fix D instances $I_1, \ldots, I_D$ and D thresholds $z_1, \ldots, z_D$. Bound # sign patterns $(u_\alpha(I_1), \ldots, u_\alpha(I_D))$ ranging over all α. Equivalently, $(u_{I_1}(\alpha), \ldots, u_{I_D}(\alpha))$.
- Use VCdim of F^* , bound # of regions induced by $u_{I_1}(\alpha), \ldots, u_{I_D}(\alpha)$: $(eND)^{d_{F^*}}$.
- On a region, exist $g_{I_1}, \ldots, g_{I_D}$ s.t.,$(u_{I_1}(\alpha), \ldots, u_{I_D}(\alpha)) = (g_{I_1}(\alpha), \ldots, g_{I_D}(\alpha))$, which equals $\big(\alpha(g_{I_1}), \ldots, \alpha(g_{I_D})\big)$. These are fns in dual class of G. Sauer's lemma on G^*, bounds # of sign patterns in that region by $(eD)^{d_{G^*}}$.
- Combining, total of $(eND)^{d_{F^*}}(eD)^{d_{G^*}}$. Set to 2^D and solve.

Refined GJ Frameworks

Theorem [Bartlett, Indyk, Wagner. COLT'22] Assume $\alpha \in R^n$, i.e. each $A_\alpha \in Alg$ has n real param. For any I and z, there is a GJ procedure $\Gamma_{I,z}$ that determines for all α if $u_I(\alpha) \geq z$ by evaluating Π distinct predicates (ratios of polys) with max. degree Δ. Then:

$$\mathbf{Pdim}(\{\mathbf{u_\alpha(I)}\}) = O(n \log(\Delta\Pi))$$

GJ (95) Procedure $\Gamma_{I,z}$



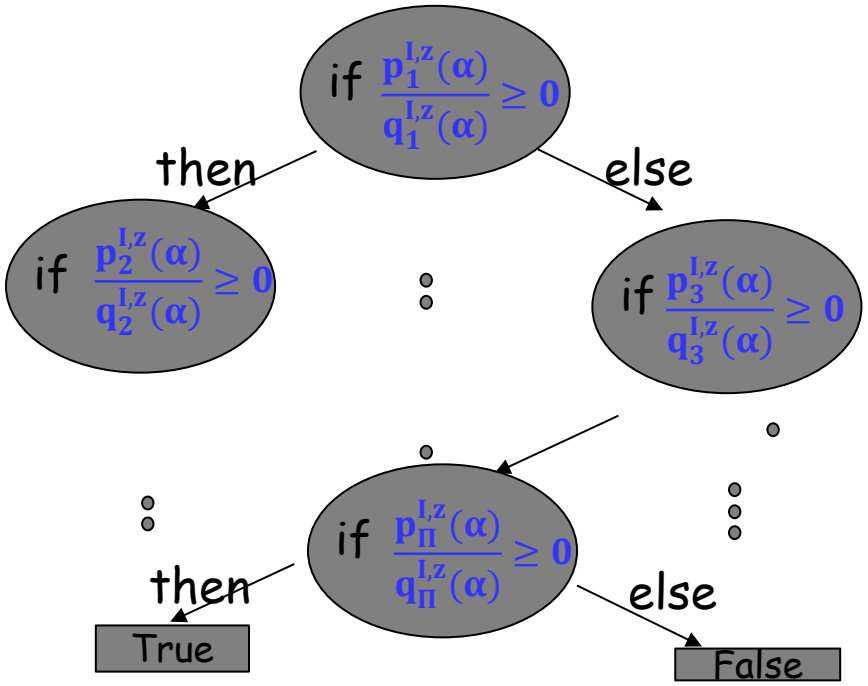
Machine Learning for Algorithm Design

General tools via dual function classes. E.g.,

Theorem [Balcan, Ngyuen, Sharma, TMLR'25] Assume $\alpha \in R^n$, i.e. each $A_\alpha \in Alg$ has n real param. For any instance I and threshold z, there is a Pfaffian GJ algorithm $\Gamma_{I,z}$ that determines for all α if $u_I(\alpha) \geq z$ by evaluating Π distinct Pfaffian predicates with Pfaffian chain length q, degree Δ, and Pfaffian degree M.

$$\mathbf{Pdim}(\{u_\alpha(I)\}) = O\left(n^2q^2 + nq\ln(\Delta + M) + n\ln\Pi\right)$$

GJ (95) Procedure $\Gamma_{I,z}$



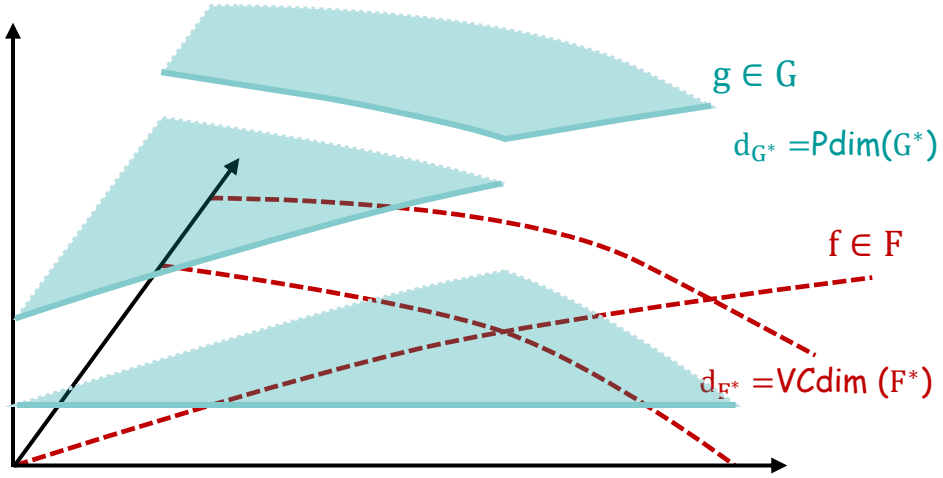
Sample Complexity of ML for Algo Design

General tools via dual function classes.

- Function class whose complexity want to control: $\{\text{cost}_\alpha\text{: parameter } \alpha\}$.
- Sufficient to understand structure of **dual class** $\{\text{cost}_I\text{: instances } I\}$.

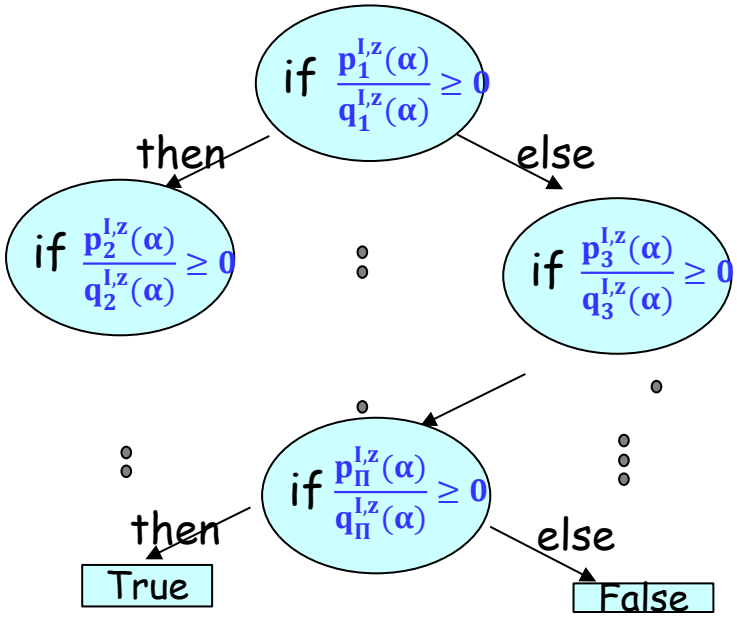
[Balcan-DeBlasio-Kingsford-Dick-Sandholm-Vitercik, STOC 2021 & JACM 2024]

$$\text{Pdim}(\{\text{cost}_\alpha(I)\}) = \tilde{O}\left((d_{F^*} + d_{G^*}) + d_{F^*} \log N\right)$$



[Bartlett, Indyk, Wagner. COLT'22]

[Balcan, Ngyuen, Sharma, TMLR'25]



Case Studies Data-Driven Algorithm Design

Many new implications and technical connections.

Mechanism design

[Balcan-Sandholm-Vitercik, EC'18]

[Balcan-Beyhaghi, TMLR'24]

Operations Research/AI: Branch and Cut

[Balcan-Dick-Sandholm-Vitercik, ICML'18]

[Balcan-Prasad-Sandholm-Vitercik, NeurIPS'21, CP'22, NeurIPS'22]

Data Science: Clustering Algorithms

Initialize Lloyds [Balcan-Dick-White, NeurIPS '18]

Parametrize Linkage + DP

[Balcan-Nagarajan-Vitercik-White, COLT'17]

[Balcan-Dick -Long, ICLR'20]

Comp. Bio. Sequence Alignment

[Balcan-DeBlasio-Kingsford-Dick-Sandholm-Vitercik, STOC 2021&JACM 2024]

Scientific Computing

Linear Systems Solvers

[Khodak-Chow- Balcan-Talwalkar, ICLR'24]

Theory of Computing

Partitioning via IQPs: SDP + Rounding [Balcan-Nagarajan-Vitercik-White, COLT'17]

Subset selection (knapsack, MWIS) [Balcan-Dick-Vitercik, FOCS'18]

Can machine learning fix its own problem?

Also think about designing machine learning algorithms themselves (e.g., hyperparameter tuning of ML algorithms)!

- What are interesting tunable families of algos?

- How do we tune machine learning algorithms, to achieve best desired performance for a given domain, with provable guarantees?

Hyperparameter Tuning in Machine Learning

Build on and expand on insights from data-driven alg design to achieve provable hyperparameter tuning of ML methods.

- **Clustering**: initialize Lloyds and learn how to do local search [Balcan-Dick-White, NeurIPS '18 (spotlight)]
- **Semi-supervised learning** (SSL): learn graph for graph-based SSL [Balcan-Sharma, NeurIPS 2021 (Oral)]
- **Regression** (linear and logistic): learn regularization param.

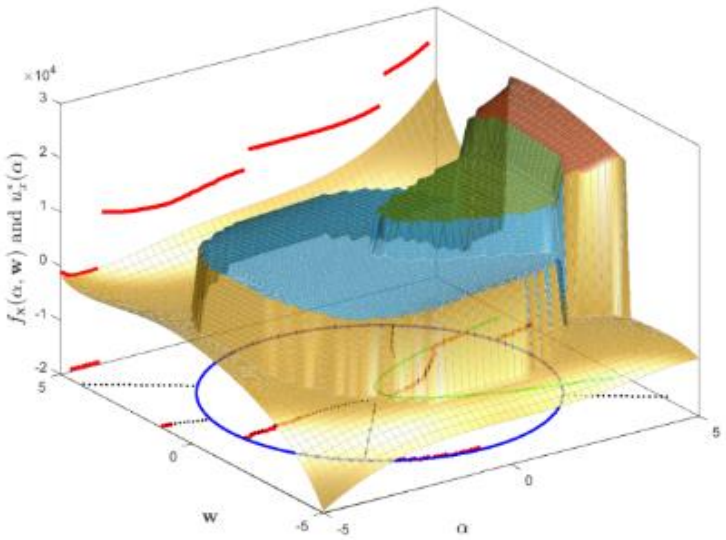
  [Balcan-Sharma-Khodak-Talwalkar, NeurIPS 2022] [Balcan-Sharma- Ngyuen, NeurIPS 2023]
- **Decision Tree Learning**: learning algorithms for building DTs

  [Balcan-Sharma, UAI 2024 (oral and Winner of Outstanding Student Paper Award)]
- **Hyperparameter tuning in neural networks**: structural components [Balcan-Sharma- Ngyuen, NeurIPS 2025]

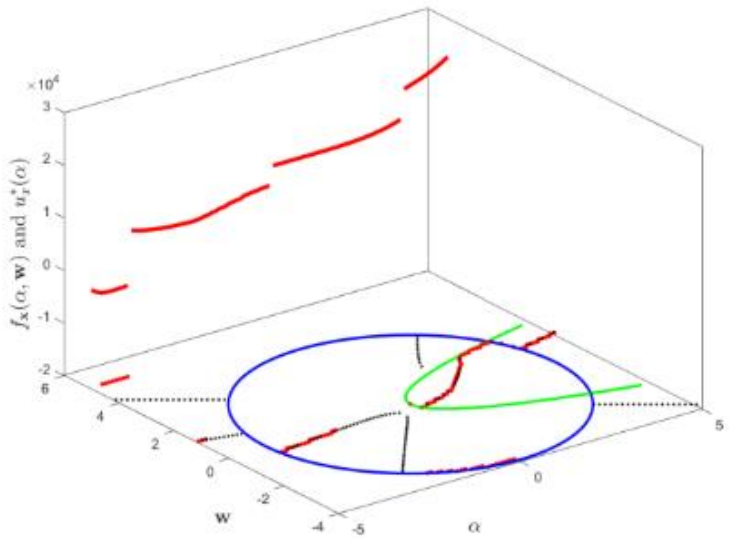
Hyperparameter Tuning in Neural Networks

Sample complexity of data-driven tuning of model hyperparameters in neural networks with structured parameter-dependent dual function

[Balcan-Sharma- Ngyuen, NeurIPS 2025]



(a) The piecewise structure of $u_{\boldsymbol{x}}^*(\alpha)$ and piecewise polynomial surface of $f_{\boldsymbol{x}}(\alpha, \boldsymbol{w})$ in sheer view.



(b) Removing the surface $f_{\boldsymbol{x}}(\alpha, \boldsymbol{w})$ for better view of $u_{\boldsymbol{x}}^*(\alpha)$, the boundaries, and the derivative curves.

$$u_{\boldsymbol{x}}^*(\alpha) = \sup_{\boldsymbol{w} \in \mathcal{W}} f_{\boldsymbol{x}}(\alpha, \boldsymbol{w}),$$

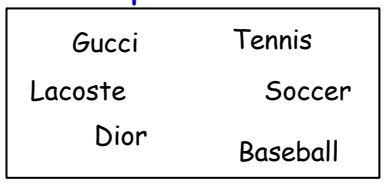
Online Algorithm Selection

Online alg. selection: instances arrive online, one by one

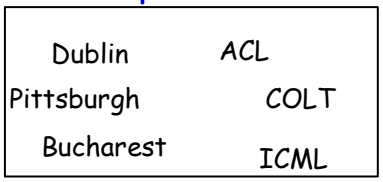
Select Algorithm: A_1 A_2 ... A_m

Get Input:

Input 1:



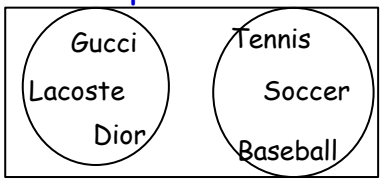
Input 2:



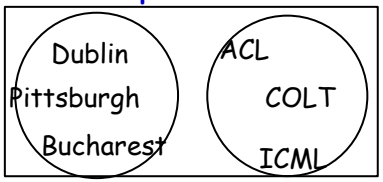
Input m:

Run algorithm on input:

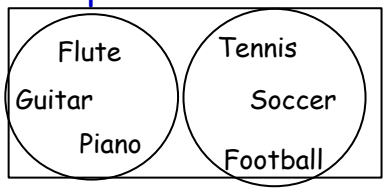
Input 1:



Input 2:



Input m:



Get cost: $cost_1$ $cost_2$ $cost_m$

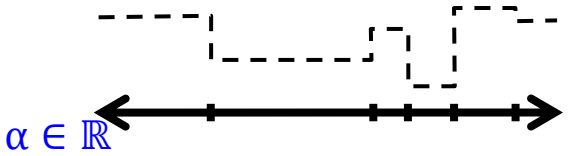
Guarantee: no regret - our cumulative performance comparable to performance of best parameter setting (algorithm) in hindsight.

Online Algorithm Selection

Instances arrive online, one by one. [Balcan-Dick-Vitercik, FOCS'18], [Balcan-Dick-Pedgen, UAI'20]

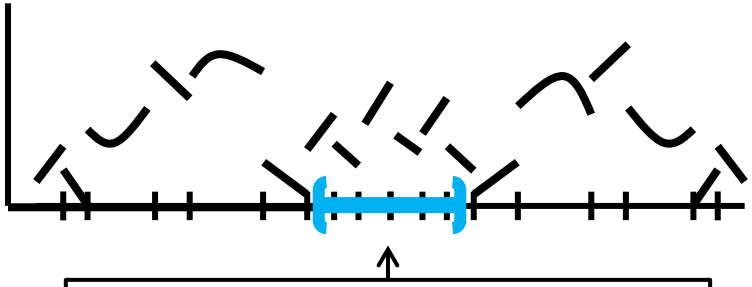
Guarantee: no regret - our cumulative performance comparable to performance of best algorithm from the family in hindsight.

Challenge: loss functions volatile.



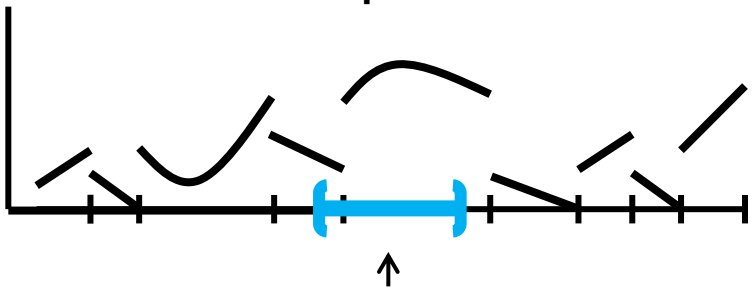
Our contribution: identify general properties (piecewise Lipschitz fns with dispersed discontinuities) sufficient for no regret guarantees.

Not dispersed



Many boundaries within interval

Dispersed



Few boundaries within any interval

Dispersion, Sufficient Condition for No-Regret

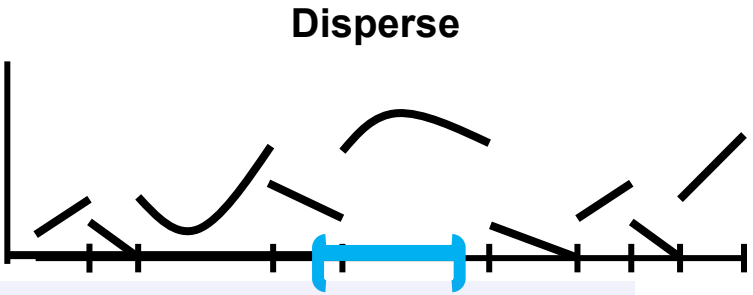
Full info: exponentially weighted forecaster [Cesa-Bianchi-Lugosi 2006]

On each round $t \in \{1, \ldots, T\}$:

- Sample $\boldsymbol{\alpha}_t$ from distr. p_t: $p_t(\boldsymbol{\alpha}) \propto \exp\left(\lambda \sum_{s=1}^{t-1} u_s(\boldsymbol{\alpha})\right)$

density of α exponential in its performance so far

Our Results:



Disperse fns, regret $\tilde{O}(\sqrt{Td}$ fnc of problem$)$.

Key Questions

- What are interesting tunable families of algos?
- How do we tune algos, to achieve best performance for a given domain, with provable guarantees?
  - **Data driven algo design as distributional learning**

Suffices to show that **dual class** $\{u_I(\cdot)$: instances $\mathbf{I}\}$ is structured.

  - **Data driven algo design as online learning**

[Balcan-Dick-Vitercik, FOCS'18], [Balcan-Dick-Pegden, UAI'20], [Balcan-Sharma, NeurIPS 2021]

Disperse fns, regret $\tilde{O}\left(\sqrt{Td}\text{ fnc of pb}\right)$.

Many Other Domains

Follow up work across different field. E.g.,

Low-rank approximation [Bartlett, Indyk, Wagner, COLT 2022]

Simulated Annealing [Blum, Dan, Seddighin, AISTATS 2021]

Optimization via Gradient Descent [Sharma 2026]

Integer and Linear Programming [Cheng and Basu 2024, Sakaue and Oki 2024]

Energy and Climate Science [Mathioudaki et al., 2023, Bostandoost et al. 2024]

Inventory Management [Ma, Xie, Xin, 2025]

Mechanism Design [Dütting, Feldman, Ponitka, Soumalias, 2025]

Open Questions

Sample Complexity:

- Data-dependent bounds (beyond regression, pricing).
  - Covering arguments.
- Lower bounds on learnability (past work: LB on Pdim & data indep. discretization for MIPs & clustering)
- Learning within an instance.

Algorithmic pb.: different from classic one (do well on average on training set, as opposed to do well on one instance).

- Output sensitive algorithms

[Balcan-Seiler-Sharma, NeurIPS 2024] [Balcan-Dick-Lang, ICLR 2020]

Online learning: other sufficient conditions for learnability.

Open Questions

Apply our tools for learning other structures. E.g, Shafi's talk (proofs, self-reducible fns)

Guarantees for LLMs, Agents for Algorithm Design.

- Leverage additional problem descriptions in text (in addition to past instances).

  E.g.: Lawless et al. [CPAIOR'25]: Cutting plane configuration for MILP solvers using LLM and text description. Retrieves relevant structure from problem descriptions. Suggests promising existing cutting-plane families.

Long term: **algorithmic paradigms as well as proof techniques of the type humans were not able to design by hand before?**

Multi-agent strategic AI systems

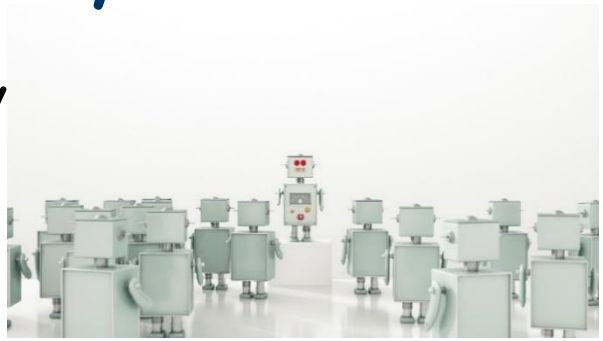
Learning in presence of other strategic agents.

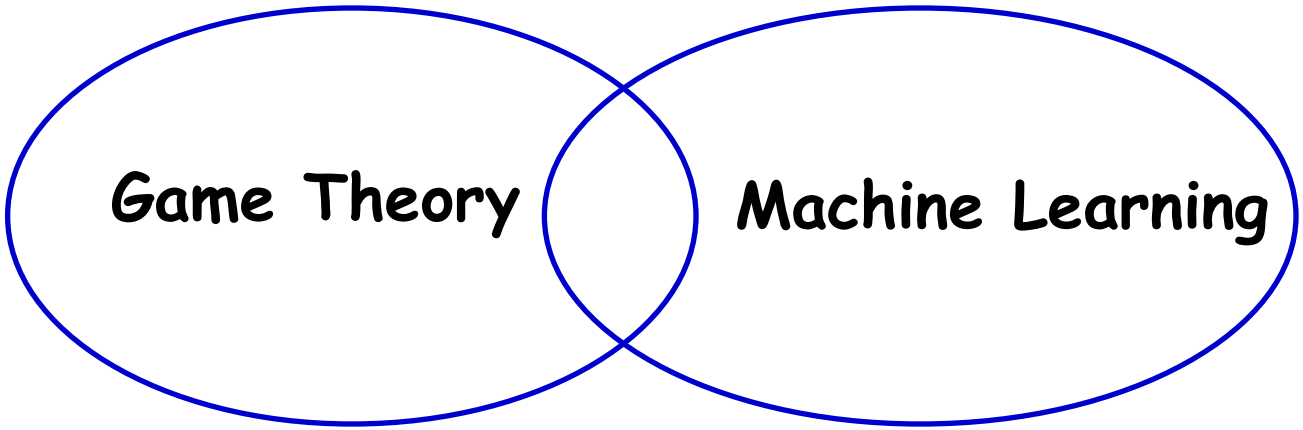
Classically:

- learning theory analyzes single learner.
- game theory analyzes static solution concepts (equilibria) or very simple learners (no feature information, just actions).

Multi-agent strategic AI systems

Use and expand game & learning theory to analyze learning in the presence of other learners.





Our work: Focus on competitive **strategic situations with commitment** used in security applications.

Stackelberg Games

- **Strategic interaction with commitment**: 2 players, leader is able to commit to a strategy before follower takes an action.
- Example: **security games**, resource-constrained defender tries to stop attacker.

Real-world impact. E.g.,

Airport security:

 Defender: TSA

 Attacker: terrorist

[PJWPTOKP, AAAI '08], …

Wildlife protection:

 Defender: park rangers

 Attacker: poacher

[HKFTC, AAAI '14], [FST, IJCAI '15], …

Stackelberg Security Games

Interactions between a defender and an attacker.

Defender's strategy space:

- (Randomized) deployment of resources to protect targets.

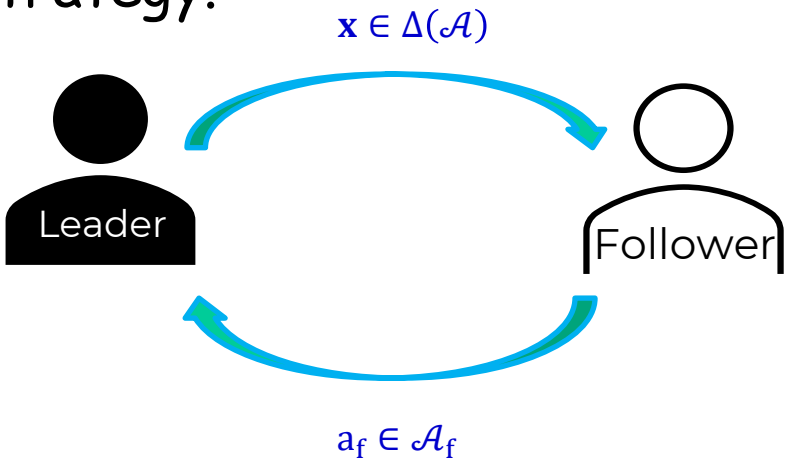
Attacker's strategy space: which target to attack.

Stackelberg solution concept:

1. Defender commits to defense strategy.

   E.g., (randomized) patrol strategy through park

2. Attacker best-responds.



Overview of Our Results

- **Prior Work**: compute equilibria in one shot game (known utilities). Learning, interaction with same attacker.

  [Conitzer-Sandholm, EC 2006] [Letchford-Conitzer-Munagala, SAGT 2009]

- Repeated Stackelberg (security) games in face of several different types of attackers. **Online learnability guarantees.**

  [Balcan-Blum-Haghtalab-Procaccia, EC 2015]

  - Interactions might be **repeated**.
  - **Different types** of **followers**.

- Online learning in repeated **contextual** Stackelberg games.

  [Harris-Wu-Balcan, NeurIPS 2024] [Balcan-Bernasconi-Castiglioni-Celli-Harris-Wu, ICLR 2026]

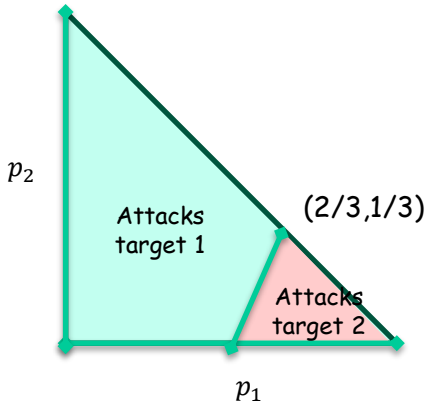
  [Balcan-Fragkia-Harris, ICML 2026]

Online learning in Security Games

Key Challenge: utility function of leader sharp transitions.

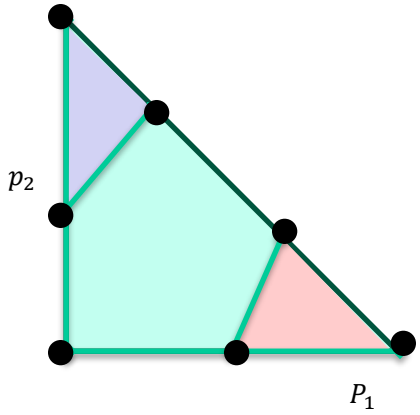
Ex: 2 targets, defender can defend only one

Attacker's utility	Target 1	Target 2
If not defended	1	0.5
If defended	0	0



Attacks target 1 if $p_2 \geq 2p_1 - 1$, otherwise, attacks target 2

Key Idea: exploit structure of the problem to identify a set of experts and reduce to online learning over those.

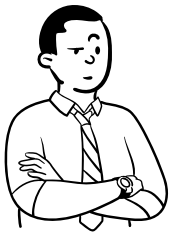


Contextual Stackelberg Security Games

Player utilities often depend on side information.

- Airport security: arrival/departure city, VIP passengers…
- Wildlife protection: weather, time of year…

Optimal attacker/defender strategies may change based on side information available.



How to design algorithms for Stackelberg (security) games with side information?

[Harris-Wu-Balcan, NeurIPS 2024] [Balcan-Bernasconi-Castiglioni-Celli-Harris-Wu, ICLR 2026]

[Balcan- Fragkia -Harris Arxiv ICML 2026]

Stackelberg Games. Summary

- Formalized repeated Stackelberg games through online learning & game theory lenses.
- Explored side information or context.

- Induced loss fns more volatile than in classic learning theory.

Discussion:

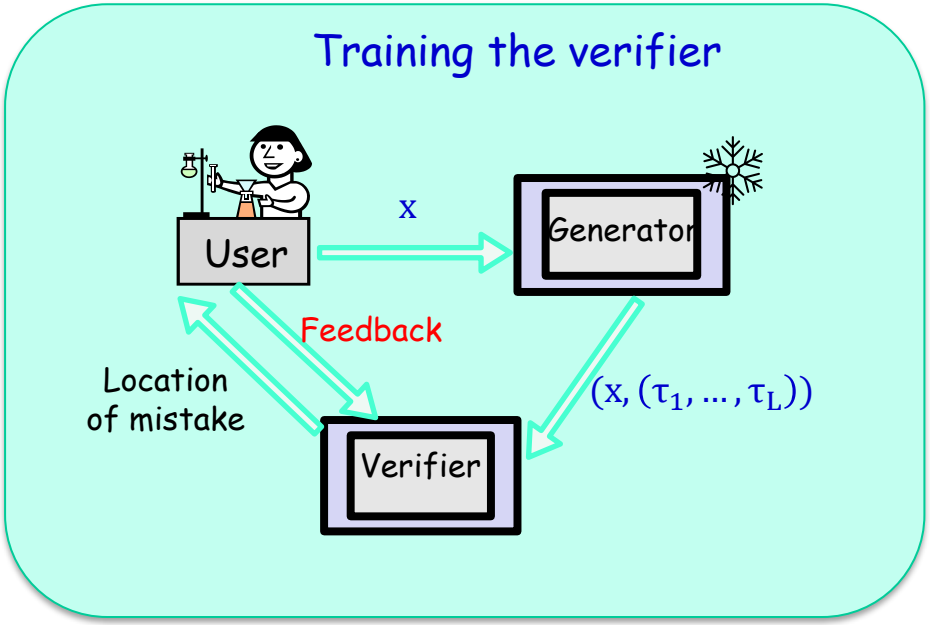
- Build on and expand learning & game theory to explore realistic learning in other multi-agent (game theoretic) contexts.

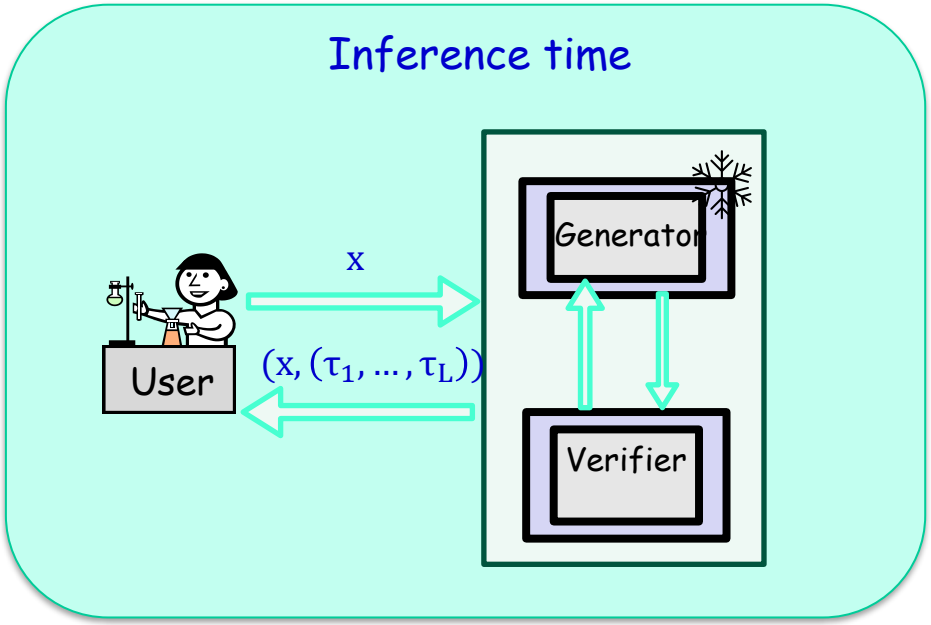
- Even more complex learners.

Theory of LLMs. Human-AI collaboration to train a verifier to improve accuracy, efficiency of a pre-trained LLM agent/generator.

[Balcan-Blum-Li-Sharma, NeurIPS 2025]

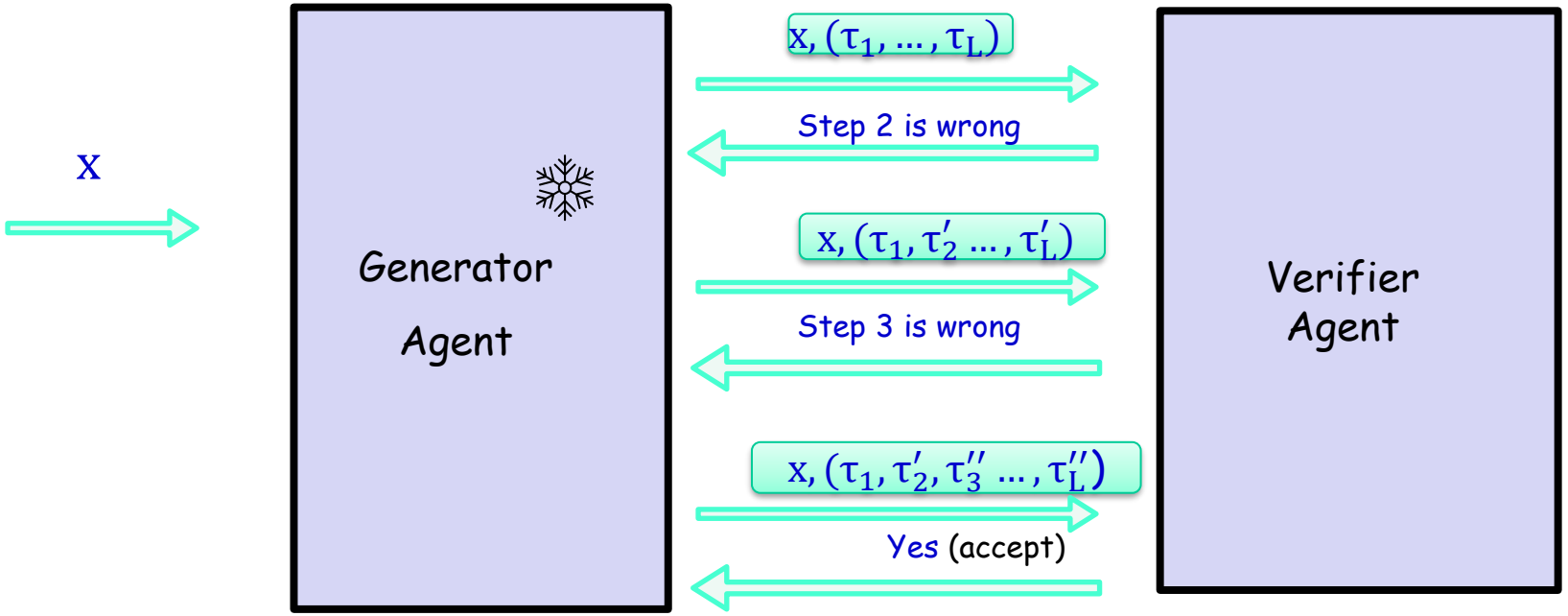
[Balcan-Blum-Fragkia-Li-Sharma, Arxiv 2026]





Modelling Challenges

When verifier used interactively with the generator agent, might go out-of-distribution if generator uses feedback to update its output.



Current AI: More Complex Systems

- Used for much **more complex tasks**.
  - Much more complex loss functions.
  - Theory substantially more interesting.
- **Learning with other agents** with complex utilities.
- Used for **much more high-stakes domains**.

Multi-faceted approach:

- extend ideas from many fields, (learning theory, game theory, algorithms, …) to analyze and better design AI.