Learning with Multiple Objectives - Foundations and Applications

Tianyi Chen **Zhuoran Yang** **Lisha Chen**

Rensselaer Polytechnic Institute

Yale University

Room 111, Vancouver Convention Centre – West Building
Vancouver, BC, Canada
2:00 pm - 6:00 pm, February 20, 2024

Tianyi Chen

Rensselaer Polytechnic Institute

Bilevel optimization,
multi-objective optimization

Zhuoran Yang

Yale University

Reinforcement learning,
deep learning, and statistics

Lisha Chen

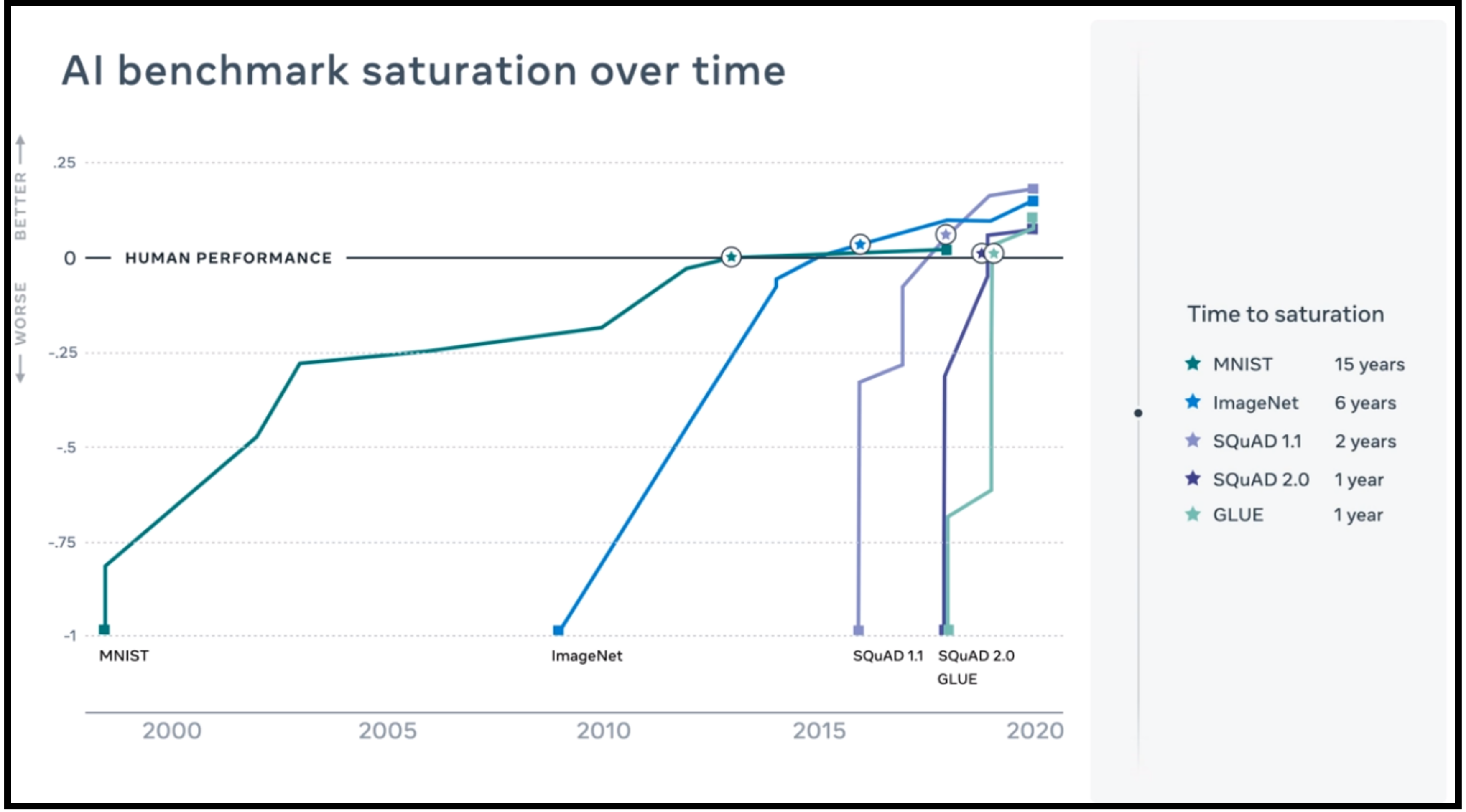
Rensselaer Polytechnic Institute

Multi-objective optimization,
and statistical learning

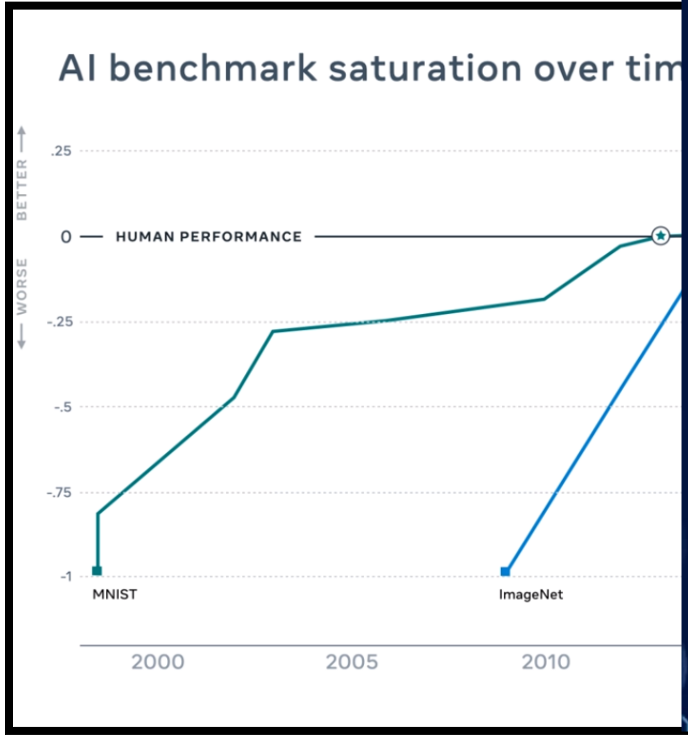
Outline

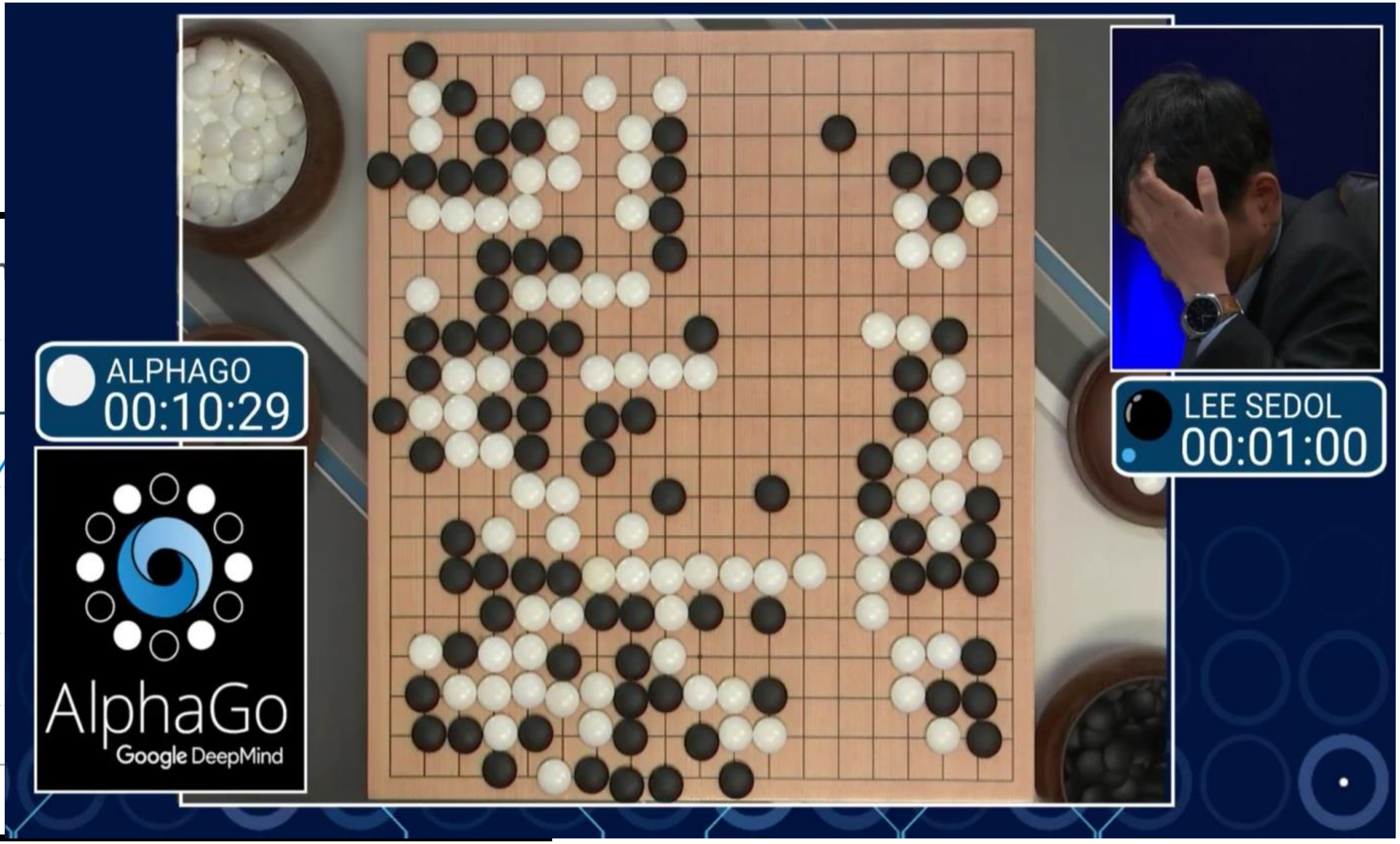
- ❑ **Part I - Introduction and background (20 mins)**
- ❑ **Part II - Bilevel optimization fundamentals**
- ❑ **Part III - Bilevel applications to reinforcement learning**
- ❑ **Part IV - Multi-objective learning beyond bilevel optimization**
- ❑ **Part V - Conclusions and open directions**

Success of AI before 2020



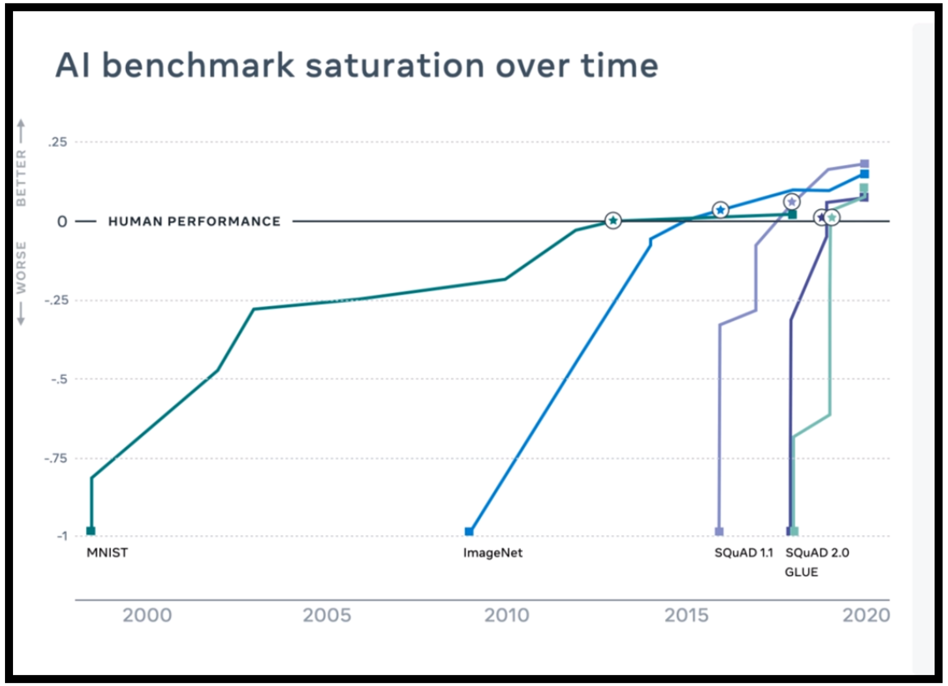
Success of AI before 2020

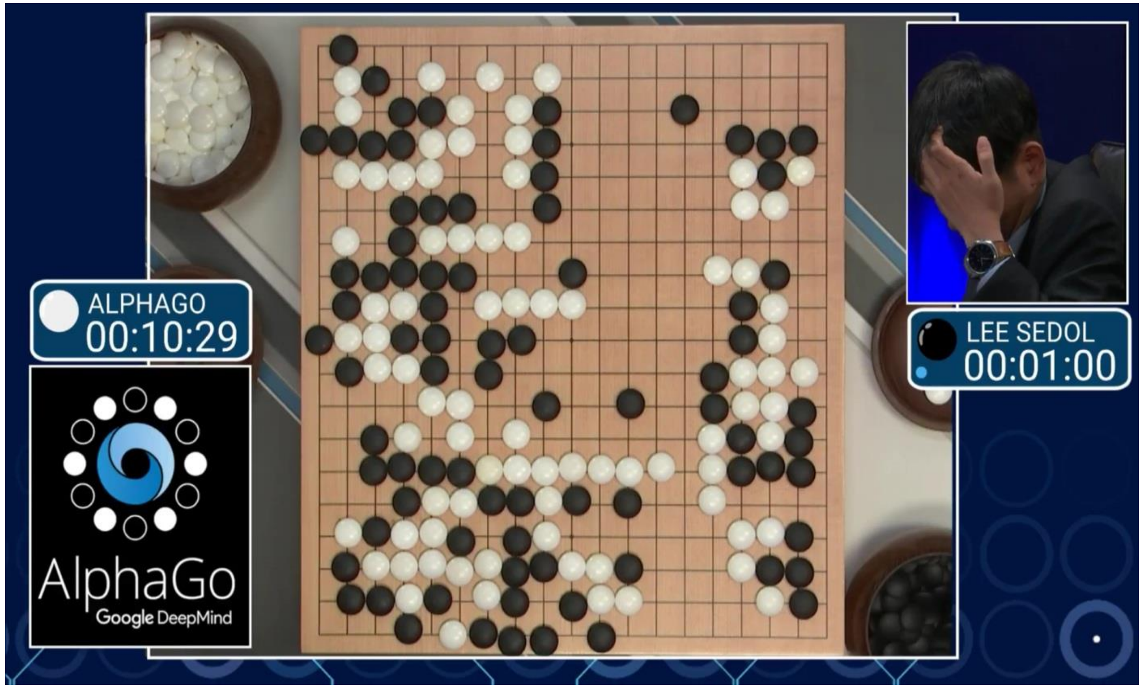




Success of AI before 2020

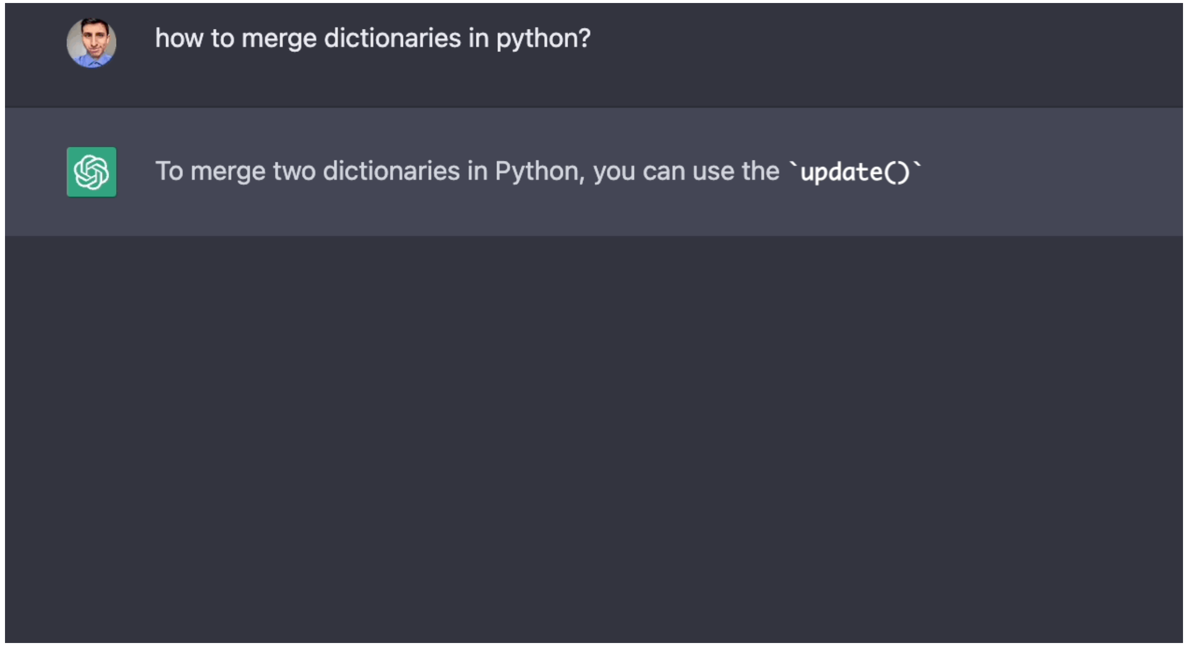
Excel at (only) one thing!

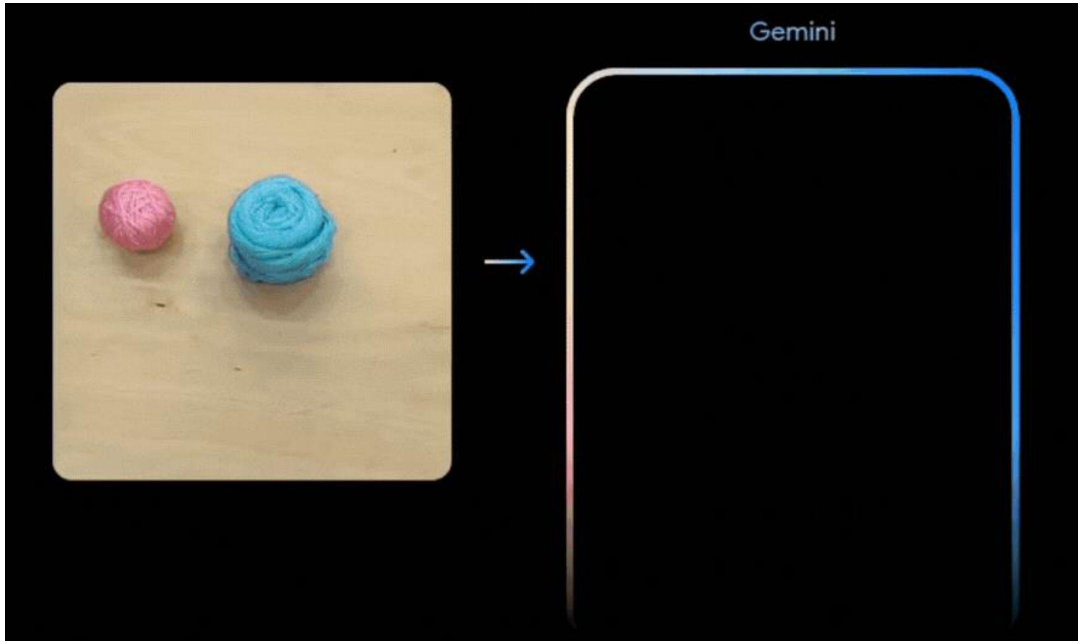





OpenAI
GPT-IQ
NEW ERA UNFOLDS!

Multiple tasks and data modalities arise today

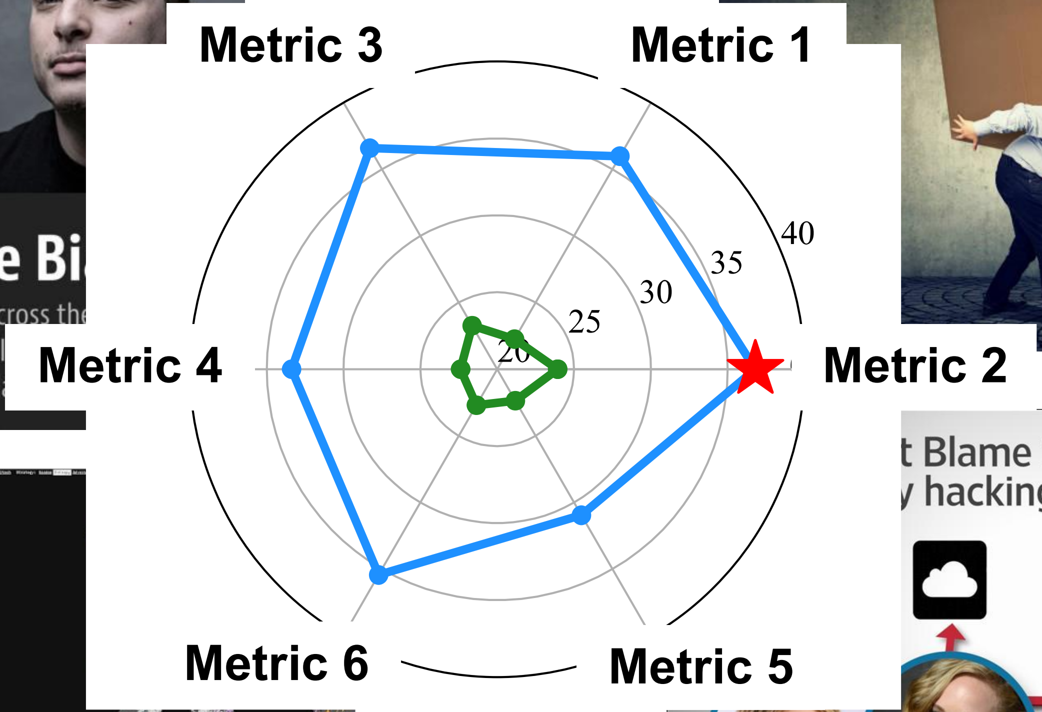




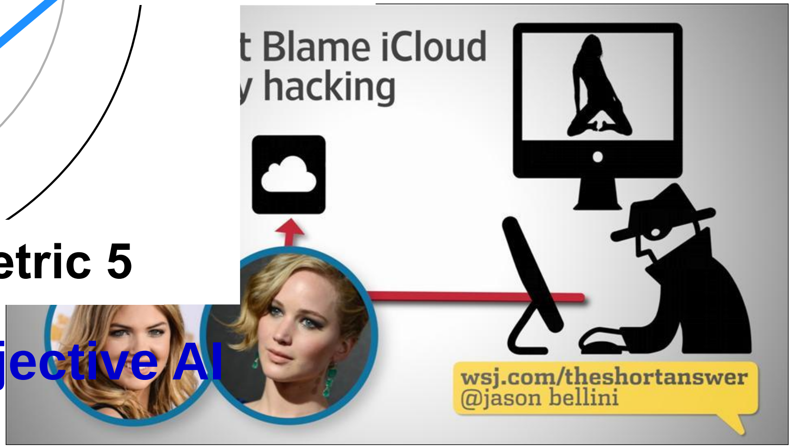
Multiple metrics arise in machine learning today

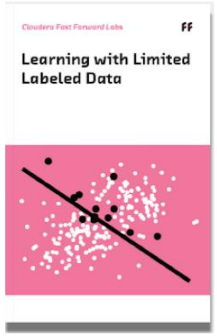
Data and model bias

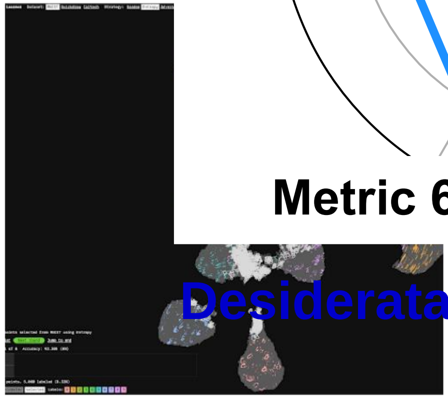
Resource constraints



Desiderata of multi-objective AI







Fast adaptation to new users

Subject to privacy regulation

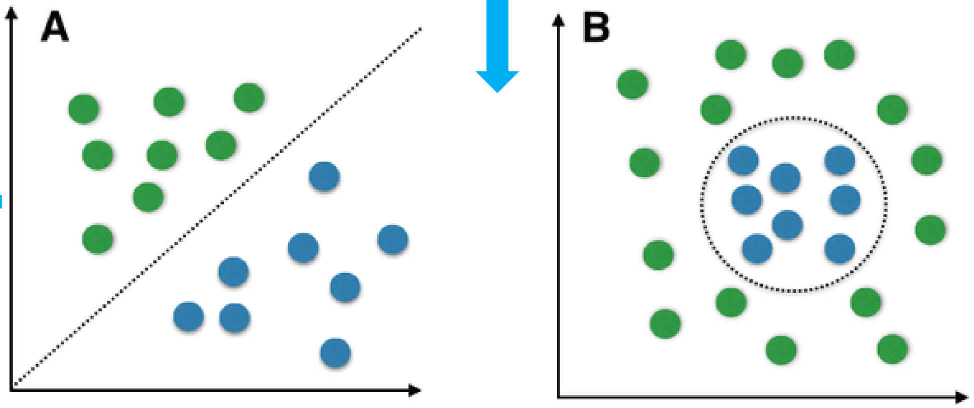
Image from Internet

"Past" era of single-objective learning

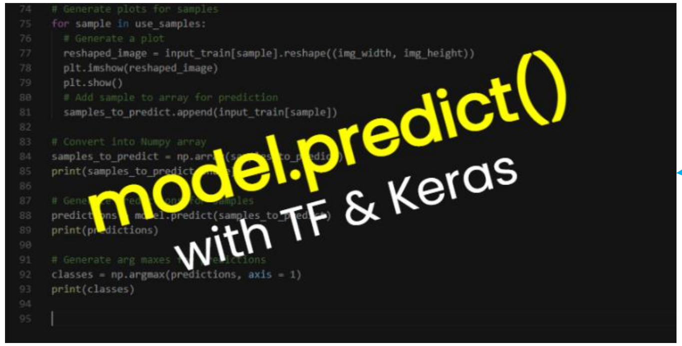
Data collection

Empirical risk minimization (ERM)

$$\min_{x} \text{loss/error (model } x\text{, trainig data)}$$



Model training



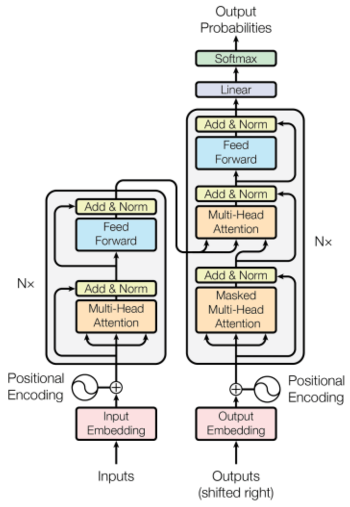
Model deployment

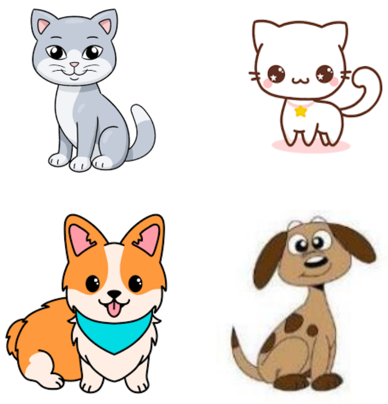
Conventional machine learning (ML) pipeline

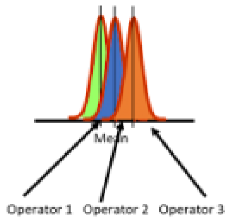
Image from Internet

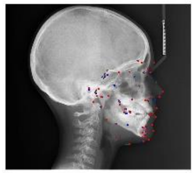
Tasks, data, metrics
all can be modeled as an objective…

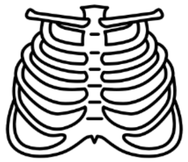
$$\min_{x} \ \text{loss (model } x\text{, trainig data, metric, tasks)}$$











Tackling multiple tasks, data, metrics via single-objective learning …

$\min\limits_{x}$ loss (model x, )

e.g., increase $x^{(1)}$, increase acc 0.01

+

versus

$\min\limits_{x}$ loss (model x, )

e.g., increase $x^{(1)}$, decrease fair 20%

+

versus

$\min\limits_{x}$ loss (model x, )

e.g., decrease $x^{(1)}$, increase acc 0.1

Simple but may cause… unit mismatch or competition

Tackling multiple tasks, data, metrics via sequential learning …

$\min_{x}$ loss (model x, )

$\min_{x}$ loss (model x, )

$\min_{x}$ loss (model x, )

No feedback, may cause catastrophic forgetting

Our focus – A tale of two methods

Bi-/multi-level training

Pre-define the preferences/orders

min objective 1

s.t. min objective 2

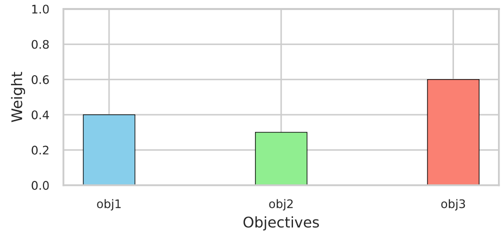
s.t. min objective 3

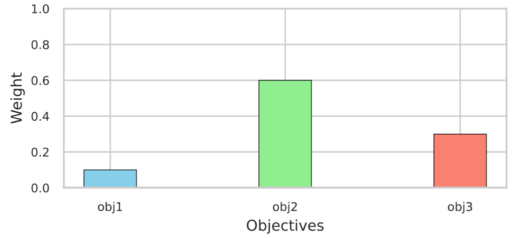
Allow feedback loops, compared with sequential learning

Multi-objective training

Pre-define or let the algorithm determine preferences

min (objective 1, objective 2, objective 3**)**

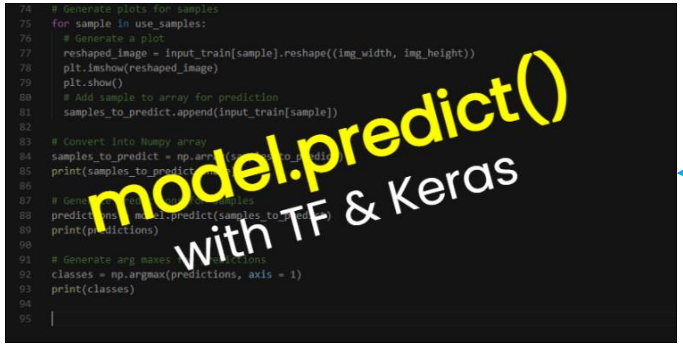




Mitigate unit mismatch or competition, compared with single-objective learning

Opportunity lies in new model training steps

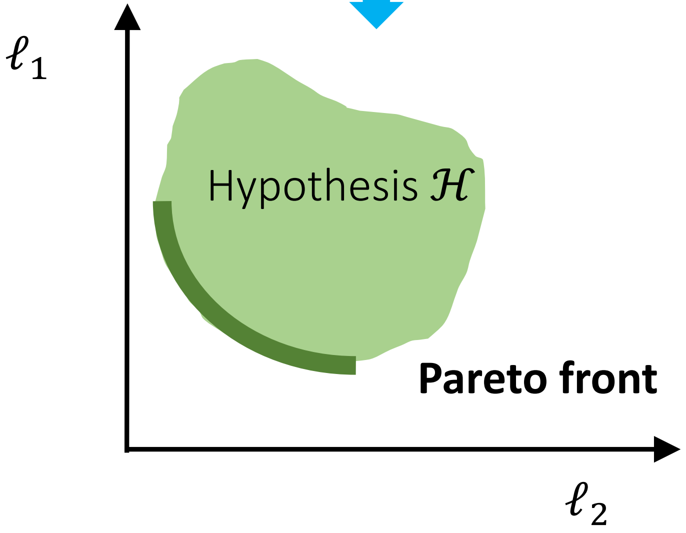
Data collection



Model deployment

Multi-objective training

$$\min_{x} \, [\text{obj } 1\,(x), \text{obj } 2\,(x), \ldots \text{obj M}\,(x)]$$



New model training

Bi-level training

$$\min_{x} \text{ second objective } (x, y^*(x))$$
$$y^*(x) = \underset{y}{\operatorname{argmin}} \text{ first objective } (x, \text{y})$$

Problems tackled by bilevel model training?

Learning from imbalanced data

Learning to fast adapt

Learning to fast optimize

Neural architecture search

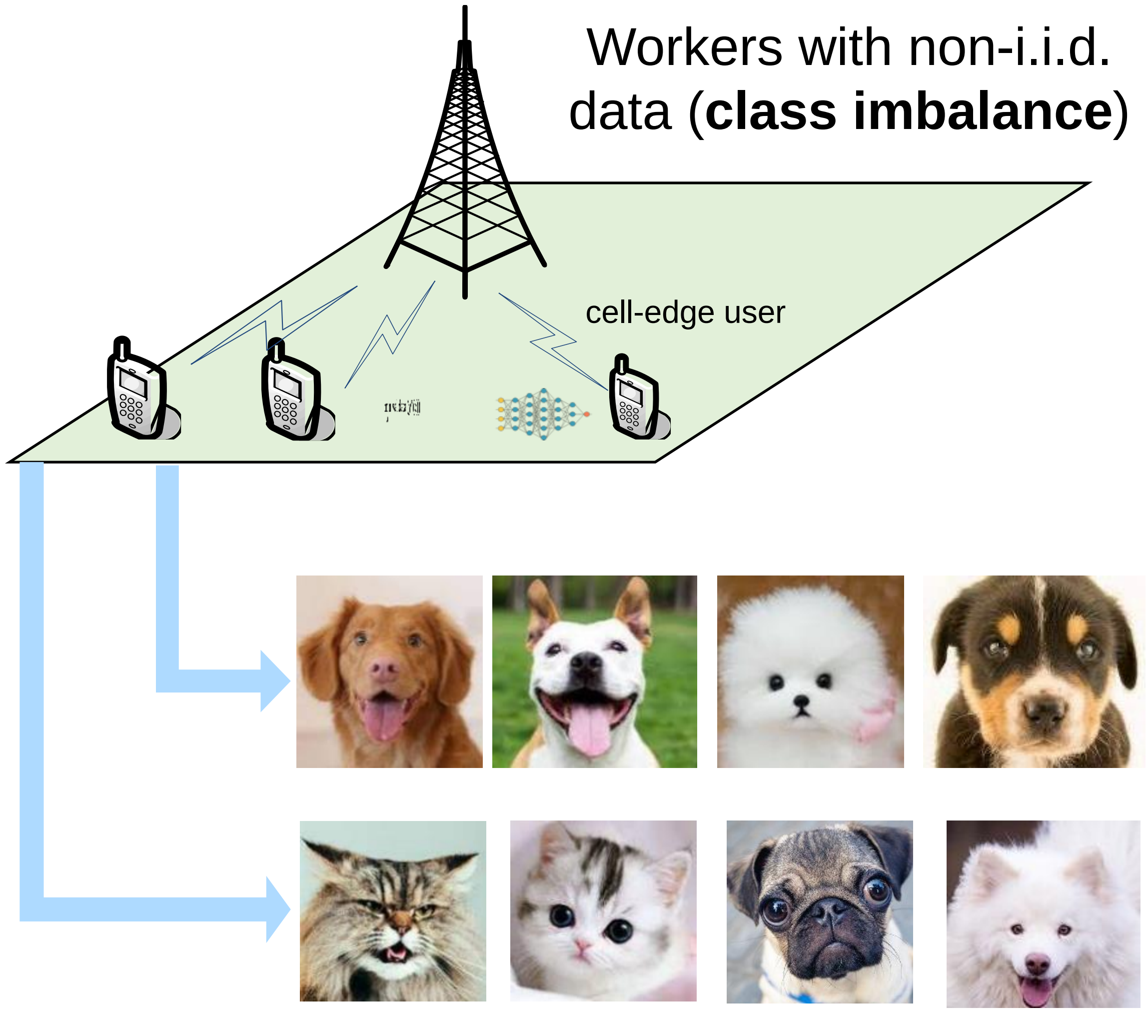
Adversarial training

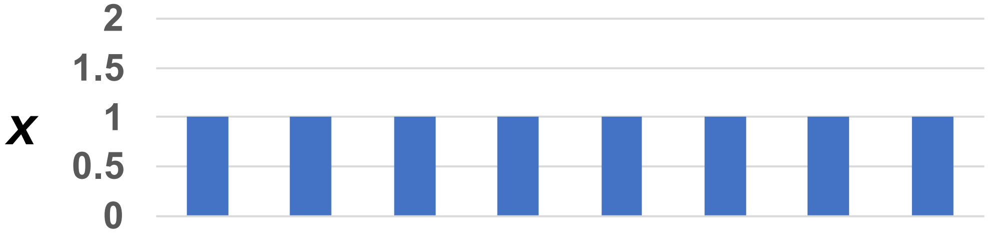
Model pruning

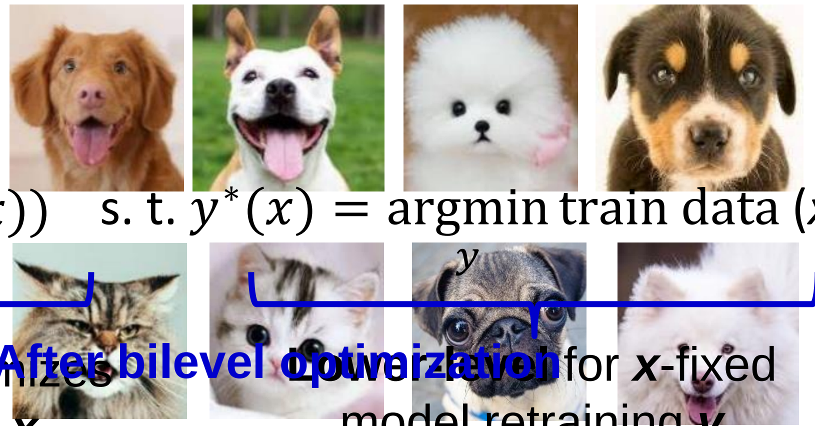
…

Bilevel optimization for learning from non-i.i.d. data

Workers with non-i.i.d. data (**class imbalance**)







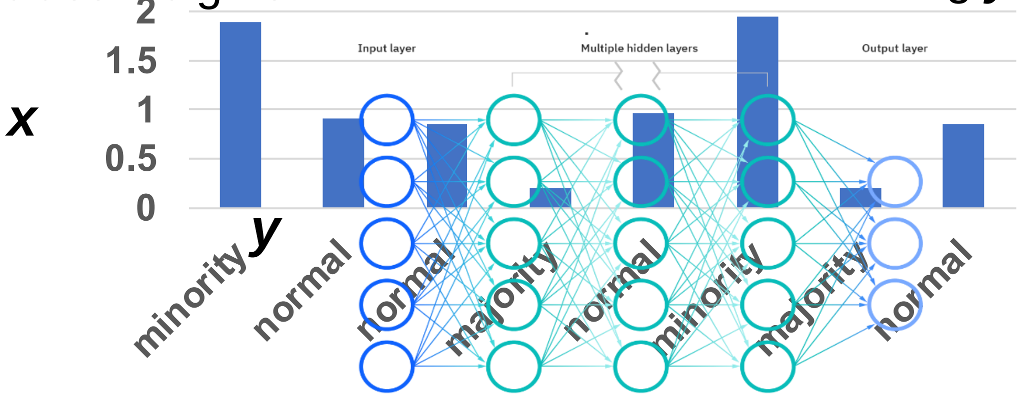
$$\sum \text{loss}(y^*(x))$$

$$\underbrace{\min_x \ \text{val data}(y^*(x))}_{} \quad \text{s.t.}\ \underbrace{y^*(x) = \operatorname*{argmin}_y \ \text{train data}(x, y)}_{}$$

Upper level optimizes class weights ***x***

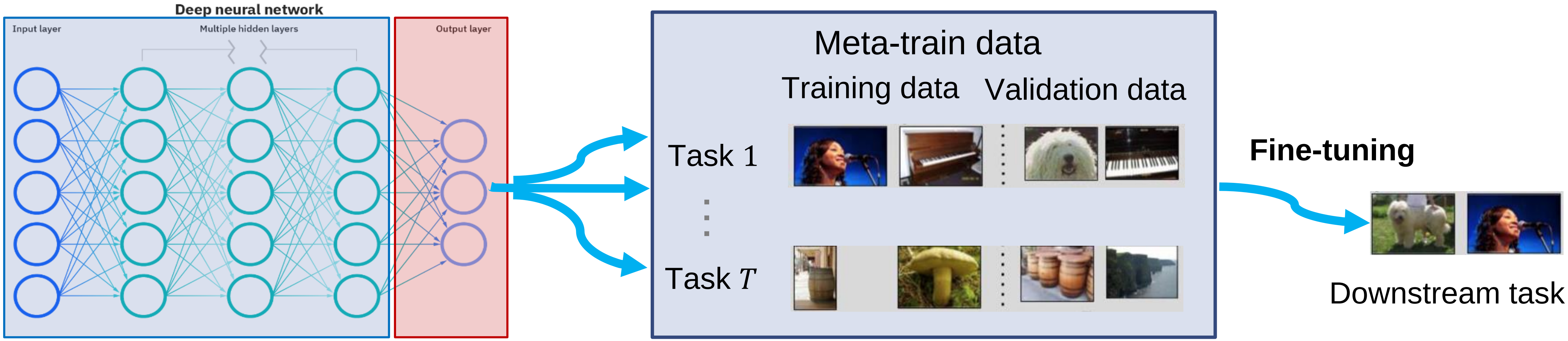
Lower level optimization for ***x***-fixed model retraining ***y***

After bilevel optimization



Li, Zhang, Thrampoulidis, Chen, and Oymak. "Autobalance: Optimized loss functions for imbalanced data." *NeurIPS 2021*

Bilevel optimization for meta learning



Pre-trained model **Fine-tuning**

$$\min_x \underbrace{\text{task validation data}\,(y^*(x))}_{\textbf{Upper level}\text{ optimizes the meta model } \boldsymbol{x}} \quad \text{s. t. } \underbrace{y^*(x) = \arg\min_y \text{train data}\,(x, y)}_{\textbf{Lower-level}\text{ for } \boldsymbol{x}\text{-fixed model fine-tuning } \boldsymbol{y}}$$

C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," *Proc. of ICML 2017*

What is bilevel optimization? A gentle introduction

Bilevel optimization can be defined as

Upper-level variable

$$
\begin{aligned}
\min_{x \in \mathcal{X}, y} \quad & f(x, y) && \text{(upper level)} \\
\text{s.t.} \quad & y \in \arg\min_{y' \in \mathcal{Y}} \; g(x, y') && \text{(lower level)}
\end{aligned}
$$

Lower-level variable

- **Merits**: capture learning hierarchy across multiple objectives
- **Difficulty**: upper- and lower-level coupling through solution set

Relation with other popular frameworks

$$\min_{x \in \mathcal{X},\, y} \quad f(x, y)$$
$$\text{s.t.} \quad y \in \arg\min_{y' \in \mathcal{Y}} \; g(x, y')$$

More general and flexible models!

- **Bilevel** versus **min-max** optimization

$$g(x, y) := -f(x, y) \quad \Longrightarrow \quad \min_{x \in \mathcal{X}} \max_{y \in \mathcal{X}} f(x, y)$$

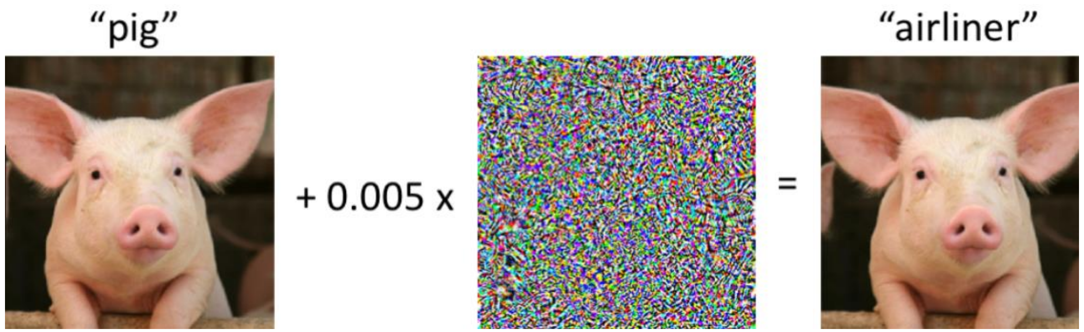


Image from Internet

A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, “Towards deep learning models resistant to adversarial attacks,” *ICLR 2018*

Despite its flexibility, is it too slow to solve?

1. **Introduction**

A sequential optimization problem in which independent decision makers act in a noncooperative manner to maximize their individual benefits may be categorized as a Stackelberg game. The bilevel programming problem is a static, open-loop version of this game where the leader controls the

bilevel programming problem (BLPP). We begin with a pair of examples showing that, even under the best of circumstances, solutions may not exist. This is followed by a proof that the BLPP is NP-hard.

Key Words. Bilevel programming, Stackelberg games, computational

In general, yes, but ML problems admit efficient solvers!

A brief history of bilevel optimization

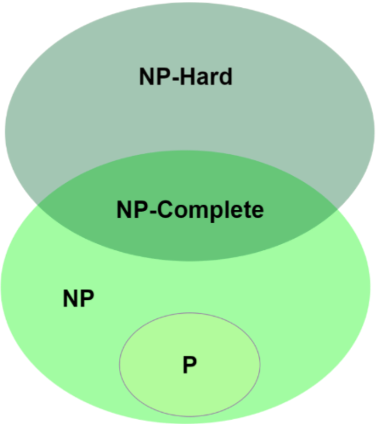
Mathematical Programs with Optimization Problems in the Constraints

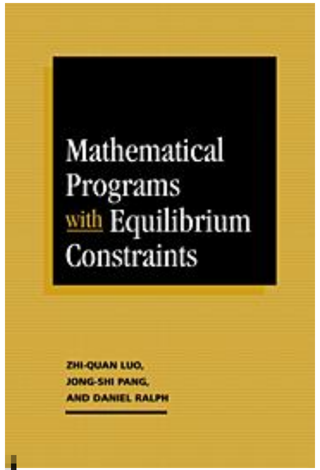
Jerome Bracken and James T. McGill

Institute for Defense Analyses, Arlington, Virginia

(Received October 5, 1971)

This paper considers a class of optimization problems characterized by constraints that themselves contain optimization problems. The problems in the constraints can be linear programs, nonlinear programs, or two-sided optimization problems, including certain types of games. The paper presents theory dealing primarily with properties of the relevant functions that result in convex programming problems, and discusses interpretations of this theory. It gives an application with linear programs in the constraints, and discusses computational methods for solving the problems.





Classification model selection via bilevel programming

G. KUNAPULI*†, K. P. BENNETT†, JING HU† and JONG-SHI PANG‡

†Department of Mathematical Sciences, Rensselaer Polytechnic Institute, 110 8th Street, Troy NY 12180.
‡ Department of Industrial and Enterprise Systems Engineering, University of Illinois at Urbana-Champaign, 104 S. Mathews Ave., Urbana IL 61801.

(Received 31 July 2006; revised 24 January 2007; in final form 23 October 2007)

Stackelberg's game

Original bilevel formulation

Single-level reformulation of bilevel optimization

Hardness results

Bilevel optimization for ML

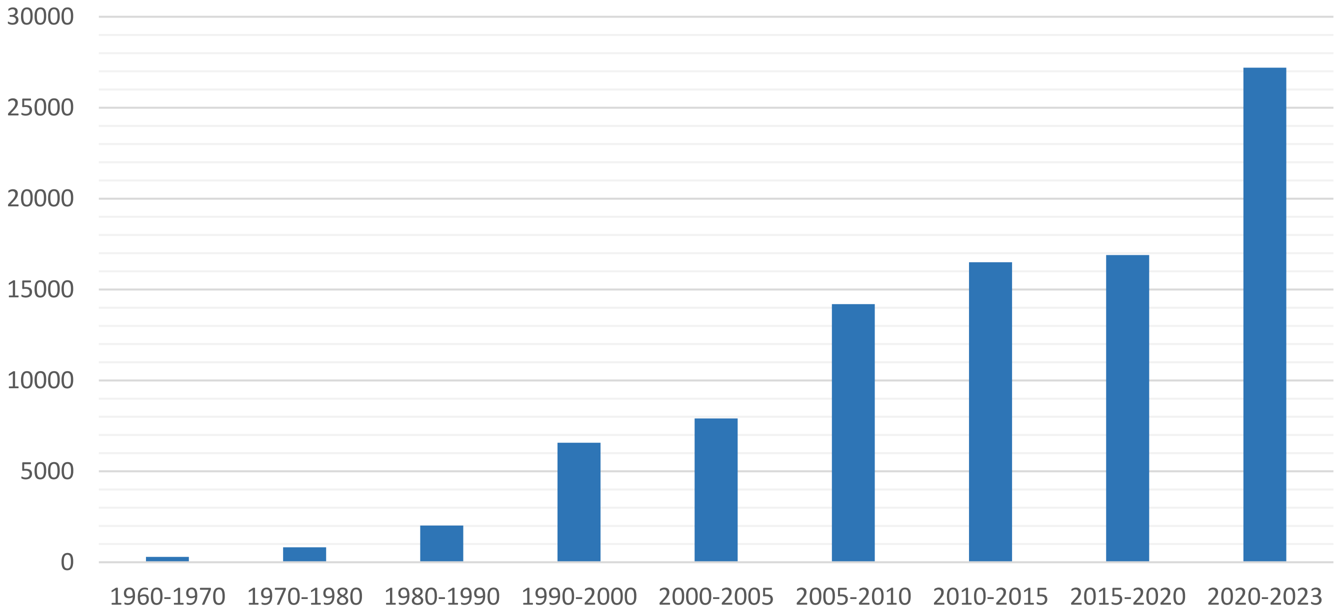
[Kunapuli et al, 08], [Pedregosa, 16], [Sabach et al, 17], [Franceschi et al, 18], [Ghadimi et al, 18], [Hong et al, 20], [Liu et al, 20], [Ji et al, 21] [Guo et al, 21]…

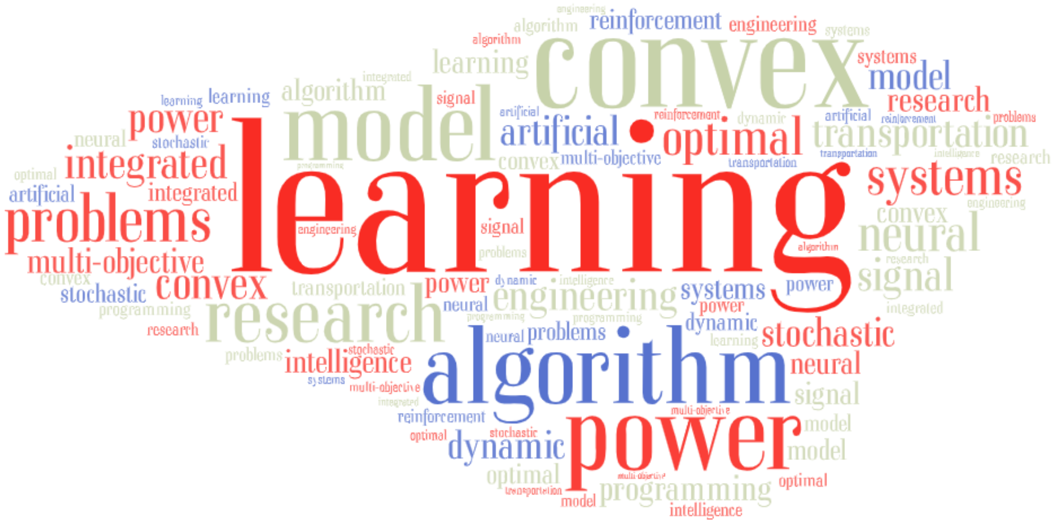
1952 | 1973 | 1980s | Early 1990s | Late 1990s | Late 2000s | **After 2020:** Finite-time convergence; generalization; new AI/ML applications

L. Vicente and P. Calamai, ``Bilevel and multilevel programming: A bibliography review," *Journal of Global optimization*, vol. 5, no. 3, pp.291-306, 1994

Z-Q. Luo, J-S. Pang, and D. Ralph, ``*Mathematical programs with equilibrium constraints*." Cambridge University Press, 1996.

Recent surge of interests





Papers on Google Scholar under keyword “bilevel optimization”

Multi-objective training

$$\min_{x} \,[\mathrm{obj1}\ (x), \mathrm{obj2}\ (x), \ldots \mathrm{objM}\ (x)]$$

Problems tackled by multi-objective training?

Learning from multiple tasks

Multilingual translation

Multi-objective alignment

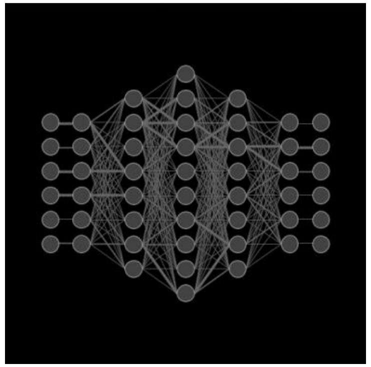
Multi-domain classification

Multi-agent reinforcement learning

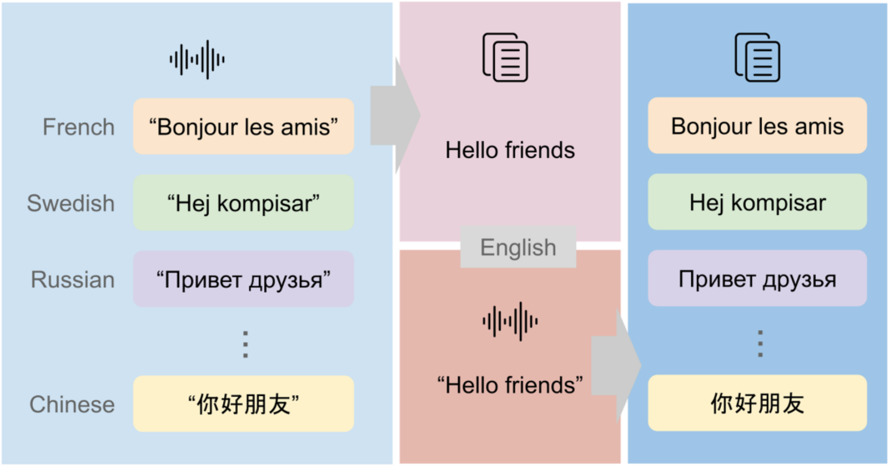
…

Multi-objective optimization for multilingual translation

Speech



Model



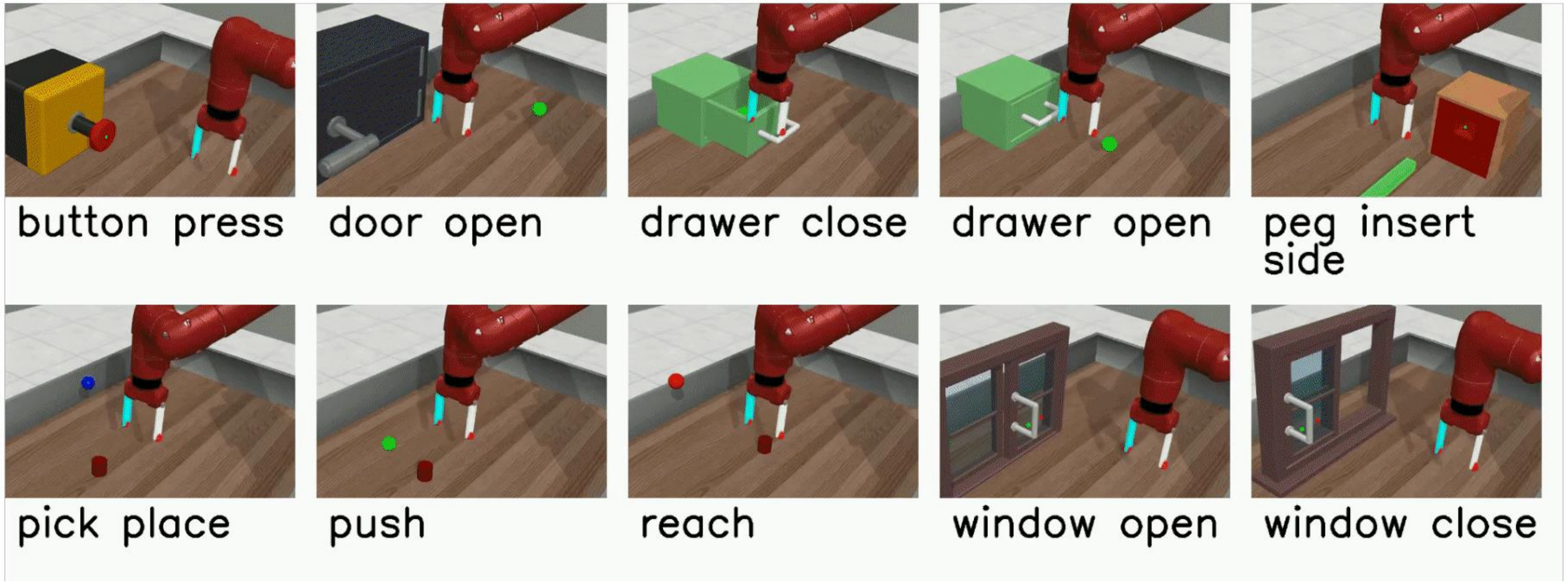
Transcripts

Universal language translator over 7000 languages

$$\min_{x} \ [\text{Laguage 1}\ (x); \text{Laguage 2}\ (x); \ldots; \text{Laguage 7000}\ (x)]$$

Y. Cheng, Y. Zhang, M. Johnson, W. Macherey, and A. Bapna, “Mu2slam: Multitask, multilingual speech and language models.” *In International Conference on Machine Learning*, pp. 5504–5520, 2023

Multi-objective optimization for multi-task robotics



Universal robotic arm controller over 50 tasks

$$\min_{x} \; [\text{button press reward } (x); \text{door open reward } (x); \ldots; \text{window close reward } (x)]$$

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine, "Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning." *In Conference on Robot Learning*, Virtual, November 2020b.

What is multi-objective optimization?

$$\text{“}\min_{x,y}\text{”}\ F(x,y) = [f_1(x,y_1), \dots, f_t(x,y_t), \dots, f_T(x,y_T)]$$

A **vector** optimization problem

- **Merits**: potentially capture all preferences/tradeoffs among objectives
- **Difficulty**: How to optimize or even compare a vector? **(see part 3)**

$$\begin{bmatrix}1\\0\\1\end{bmatrix} \le \begin{bmatrix}1\\1\\1\end{bmatrix}? \qquad \begin{bmatrix}2\\0\\0\end{bmatrix} \le \begin{bmatrix}1\\1\\1\end{bmatrix}?$$

Relation with other popular frameworks

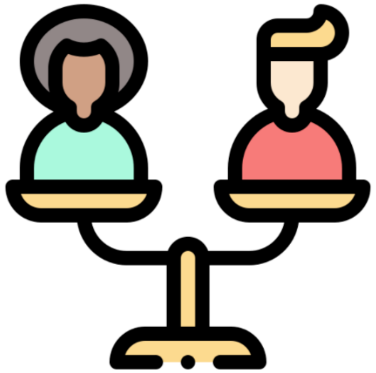
$$\text{“}\min_{x,y}\text{”}\ F(x,y) = [f_1(x,y), \dots, f_2(x,y)]$$

More general and flexible models!

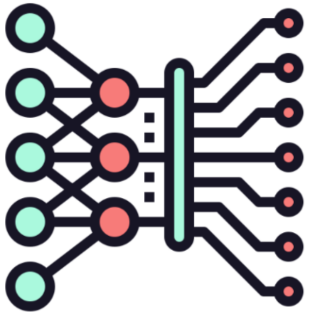
- **Multi-objective** versus **functional constrained** optimization

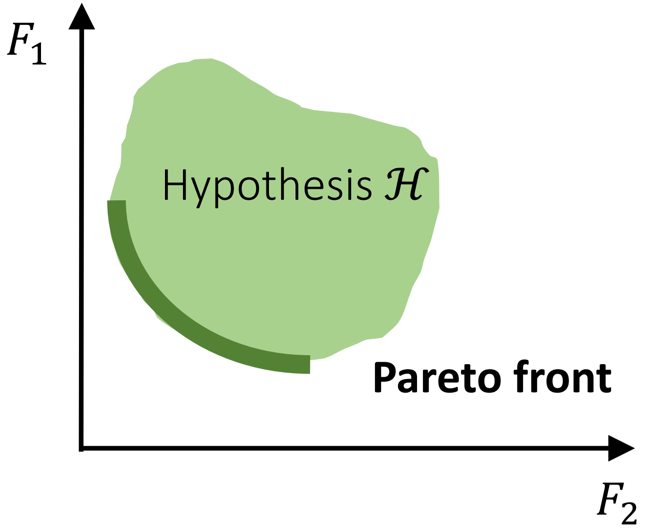
$$\boxed{F_1 := \min_{x,\ y}\ f_1(x,y) \ \text{ s.t. } f_2(x,y) \le F_2 \ \Longrightarrow\ \text{change } F_2 \text{ to obtain all } (F_1, F_2)}$$

fairness

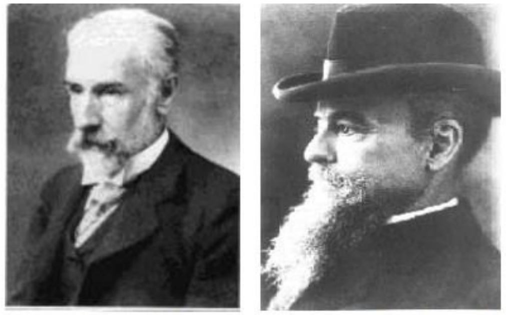


accuracy

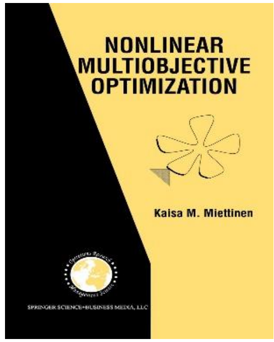


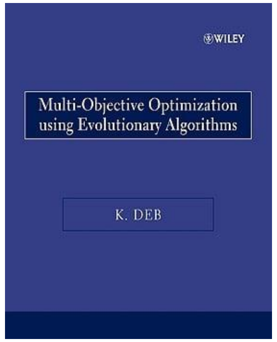


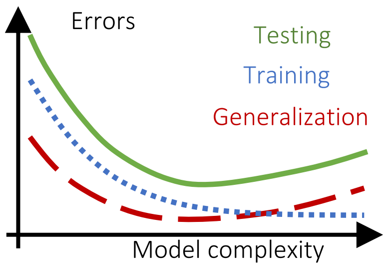
History of multi-objective optimization



(left) Francis Y. Edgeworth (1845-1926) and (right) Vilfredo Pareto (1848-1923)







1880s – 1900s | 1970s – present | 1980s – present | 2010s – present

RESEARCH PAPER

The weighted sum method for multi-objective optimization: new insights

R. Timothy Marler · Jasbir S. Arora

static weighted sum

Bilevel and Multilevel Programming:

A Bibliography Review[1]

Luís N. Vicente[2] and Paul H. Calamai[3]

Lexicographic

Application to AI/ML

Numerical Analysis/Calculus of Variations

Multiple-gradient descent algorithm (MGDA) for multiobjective optimization

Algorithme de descente à gradients multiples pour l'optimisation multiobjectif

Jean-Antoine Désidéri

INRIA, Centre de Sophia Antipolis Méditerranée, 2004, route des Lucioles, BP 93, 06902 Sophia Antipolis cedex, France

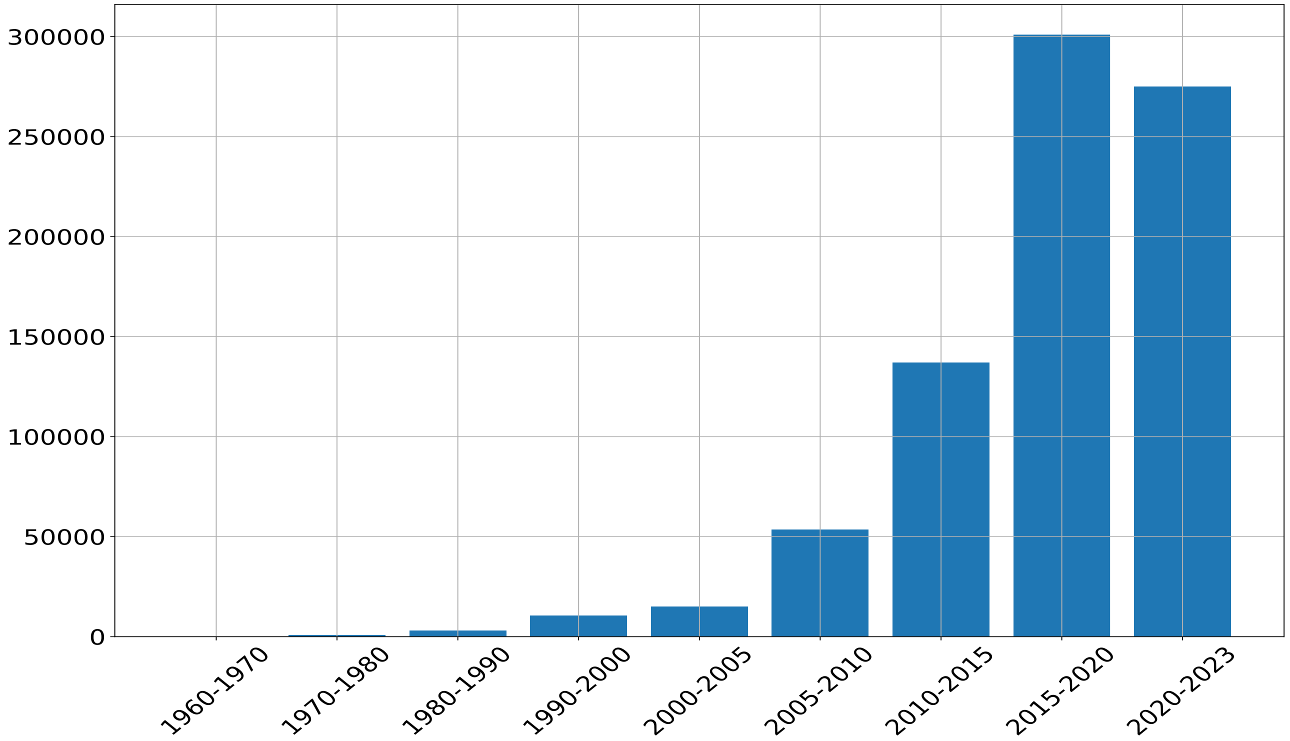
MGDA

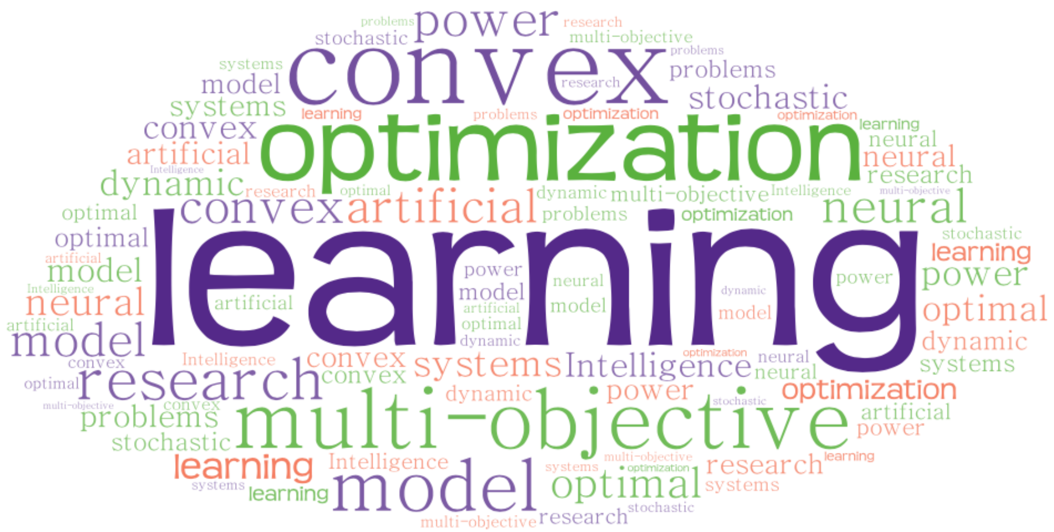
Multi-Task Learning Using Uncertainty to Weigh Losses for Scene Geometry and Semantics

Alex Kendall, University of Cambridge, agk34@cam.ac.uk
Yarin Gal, University of Oxford, yarin@cs.ox.ac.uk
Roberto Cipolla, University of Cambridge, rc10001@cam.ac.uk

Uncertainty-based

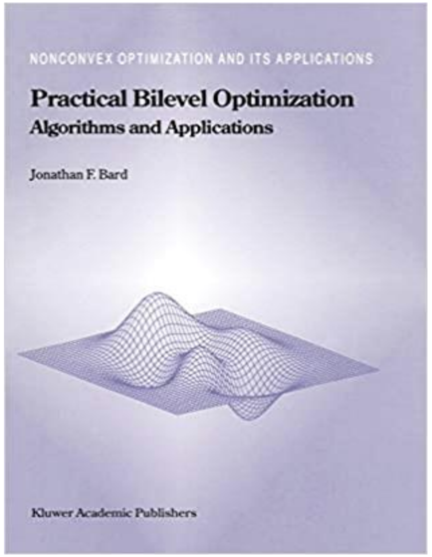
Recent surge of interests

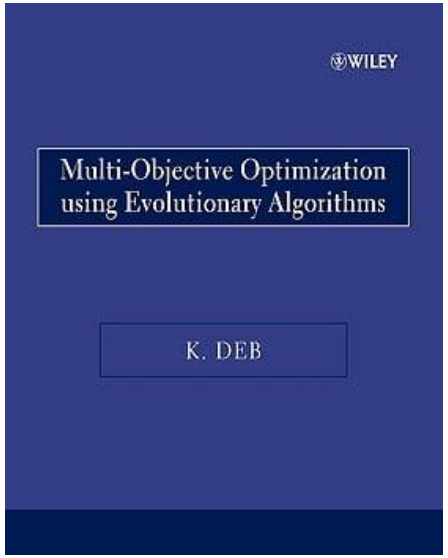


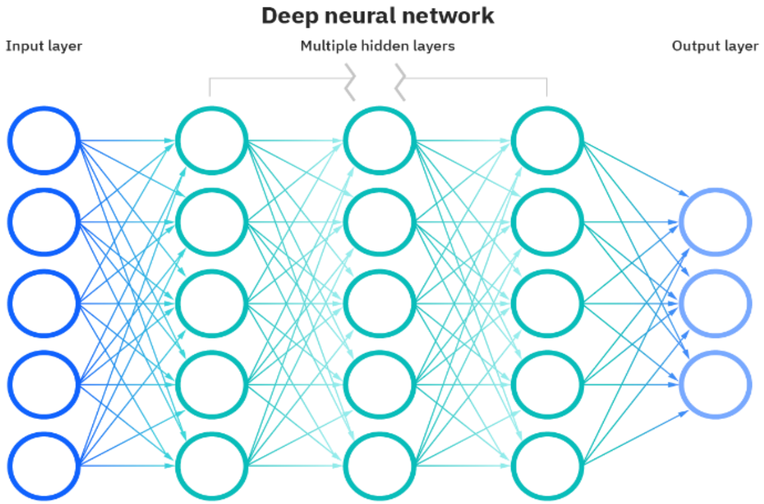


Google Scholar under keyword "multi-objective optimization"

Rationale for this tutorial







Classic theory and algorithm

Emerging AI applications

- **Part II: Recent advances in bilevel optimization foundations**
- **Part III: A representative application to reinforcement learning**
- **Part IV: Recent advances in multi-objective learning foundations**

Tutorial Part II:
Bilevel Optimization Fundamentals

Tianyi Chen

Rensselaer Polytechnic Institute

February 20, 2024

Outline

- ❑ Part I - Introduction and background
- ❑ **Part II - Bilevel optimization fundamentals (60 mins)**
- ❑ Part III - Bilevel applications to reinforcement learning
- ❑ Part IV - Multi-objective learning beyond bilevel optimization
- ❑ Part V - Conclusions and open directions

Two general recipes for bilevel optimization

$$\min_{x \in \mathcal{X},\, y} \quad f(x, y)$$
$$\text{s.t.} \quad y \in S(x) := \arg\min_{y' \in \mathcal{Y}} \; g(x, y')$$

$$\min_{x \in \mathcal{X}} \quad F(x)$$
$$\text{with} \quad F(x) := \min_{y \in S(x)} \; f(x, y)$$

Nested optimization
first over *y* and then over *x*

$$\min_{x \in \mathcal{X},\, y \in \mathcal{Y}} \quad f(x, y)$$
$$\text{s.t.} \quad \text{sufficient conditions for } y \in S(x)$$

Constrained optimization
jointly over *x* and *y*

Difficulty of solving lower-level *y*-problems

Overview of methods covered in this tutorial

$$\min_{x \in \mathcal{X}} \quad F(x)$$

$$\text{with} \quad F(x) := \min_{y \in S(x)} \quad f(x,y)$$

Nested optimization

$$\min_{x \in \mathcal{X},\, y \in \mathcal{Y}} \quad f(x,y)$$

$$\text{s.t.} \quad \text{sufficient conditions for } y \in S(x)$$

Constrained optimization

Difficulty of lower-level *y*-problems

Implicit gradient

Explicit gradient (Algorithm unrolling)

Optimality condition

Penalty method

Solve simple bilevel optimization via implicit gradients

Upper-level variable

$$
\begin{aligned}
\min_{x \in \mathbb{R}^d} \quad & F(x) := f(x, y^*(x)) && \text{(upper level)} \\
\text{s.t.} \quad & y^*(x) = \arg \min_{y \in \mathbb{R}^{d'}} g(x, y) && \text{(lower level)}
\end{aligned}
$$

Lower-level variable

Start from the simple setting: **no constraints**, **unique** lower-level solution

What is the key challenge? Finding implicit gradients

The **upper-level gradient** w.r.t. ***x*** is

$$\nabla F(x) = \nabla_x f(x, y^*(x)) + \nabla_x y^*(x)^\top \nabla_y f(x, y^*(x))$$

Implicit gradient: Gradient of the lower-level solution w.r.t. upper-level variable

Unconstrained + strong convexity

$$\nabla_y g(x, y^*(x)) = 0$$

$$\Downarrow$$

$$\nabla^2_{xy} g(x, y^*(x)) + \nabla_x y^*(x)^\top \nabla^2_{yy} g(x, y^*(x)) = 0$$

$$\Downarrow$$

$$\nabla_x y^*(x)^\top := -\nabla^2_{xy} g(x, y^*(x)) \left[\nabla^2_{yy} g(x, y^*(x))\right]^{-1}$$

Approximate upper-level implicit gradients

Key challenges: evaluating upper-level gradients is costly

$$\nabla F(x) = \nabla_x f(x, y^*(x)) - \nabla^2_{xy} g(x, y^*(x)) \left[\nabla^2_{yy} g(x, y^*(x))\right]^{-1} \nabla_y f(x, y^*(x))$$

Approximate $y \approx y^*(x)$ and introduce a **slightly biased** implicit gradient

$$\overline{\nabla} f(x, y) := \nabla_x f(x, y) + \nabla^2_{xy} g(x, y) \left[\nabla^2_{yy} g(x, y)\right]^{-1} \nabla_y f(x, y) \approx \nabla F(x)$$

Approximate the Hessian inversion by ($N' \sim \mathcal{U}(0, 1, , \cdots, N)$)

Neumann series $$\left[\nabla^2_{yy} g(x, y)\right]^{-1} \approx \frac{N}{L_g} \prod_{n=1}^{N'} \left(\mathbf{I} - \frac{1}{L_g} \nabla^2_{yy} g(x, y; \phi^n)\right).$$

Saeed Ghadimi and Mengdi Wang. ``Approximation methods for bilevel programming," arXiv preprint:1802.02246, 2018.

The generic template: Alternating implicit SGD

ALSET: A unified a**L**ternating **S**tochastic gradi**E**nt descen**T**

For *k* = *0, 1, 2, …, K* do

S1) $x^{k+1} = \text{SGD update } (x^k) \text{ on } F(x) \quad \text{ with } y^k \approx y^*(x^k)$

S2) $y^{k+1} = \text{One or multiple SGD updates } (y^k) \text{ on } g(x^{k+1}, y)$

- Reduce to stochastic gradient descent ascent methods [Jin-Netrapalli-Jordan 2019]

Error induced from inexact lower-level variables

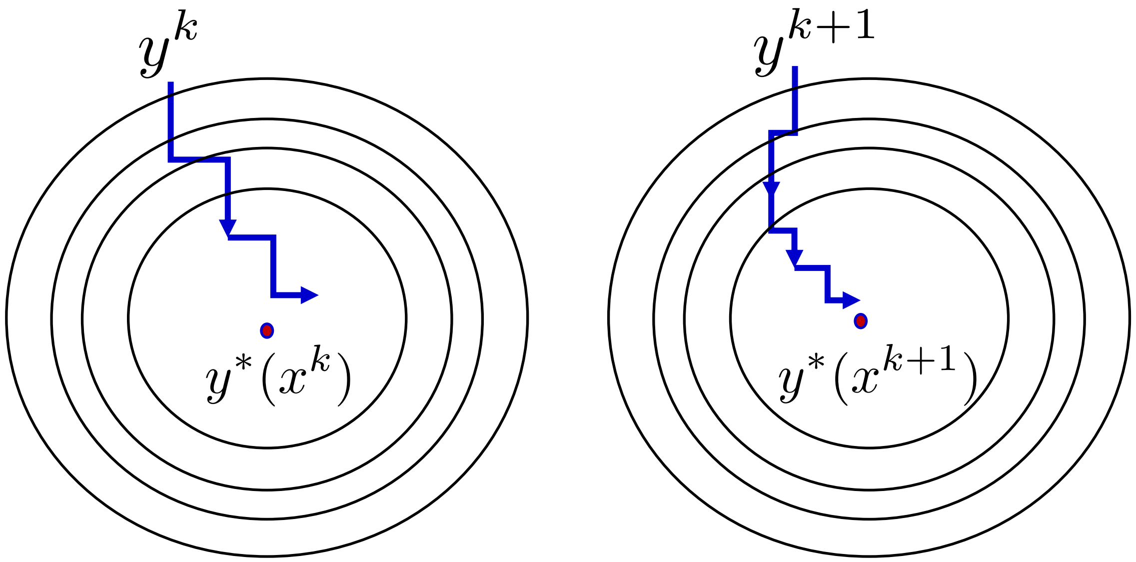


Figure: contour map of $g(x,\cdot)$

ALSET: Use inexact lower-level solution to calculate implicit gradient $\nabla F(x)$

Challenge: Gradient bias depends on the drift of lower-level solutions $y^*(x)$

Two early attempts to this problem

- **Two-timescale**: Update *x* in a ***slower*** timescale than *y*; e.g., **TTSA** [Hong et al, 20]

$$x^k \longrightarrow x^{k+1} \to x^{k+2} \to x^{k+3} \to \to x^{k+T}$$

$$\approx y^*(x^{k+T})$$

$$y^k \longrightarrow y^{k+1} \longrightarrow y^{k+2} \longrightarrow y^{k+3} \longrightarrow \longrightarrow y^{k+T}$$

- **Double-loop**: Update *y* with ***growing*** # of iters; **BSA** [Ghadimi et al, 18], **StocBio** [Ji et al, 21]

$$x^k \longrightarrow x^{k+1} \longrightarrow x^{k+2} \longrightarrow x^{k+3} \longrightarrow \quad \longrightarrow x^{k+T}$$

$$\approx y^*(x^{k+T})$$

$$y^k \to \to \to y^{k+1} \to \to \to \to \to y^{k+2} \to \to \to \cdots \to y^{k+3} \to \to \to \cdots \to \longrightarrow y^{k+T}$$

Hong, Wai, Wang, and Yang, “A two-timescale framework for bilevel optimization: Complexity analysis and application to actor-critic.” ***SIAM J OPT 2023***

Ghadimi and Wang. ``Approximation methods for bilevel programming,” arXiv preprint:1802.02246, 2018.

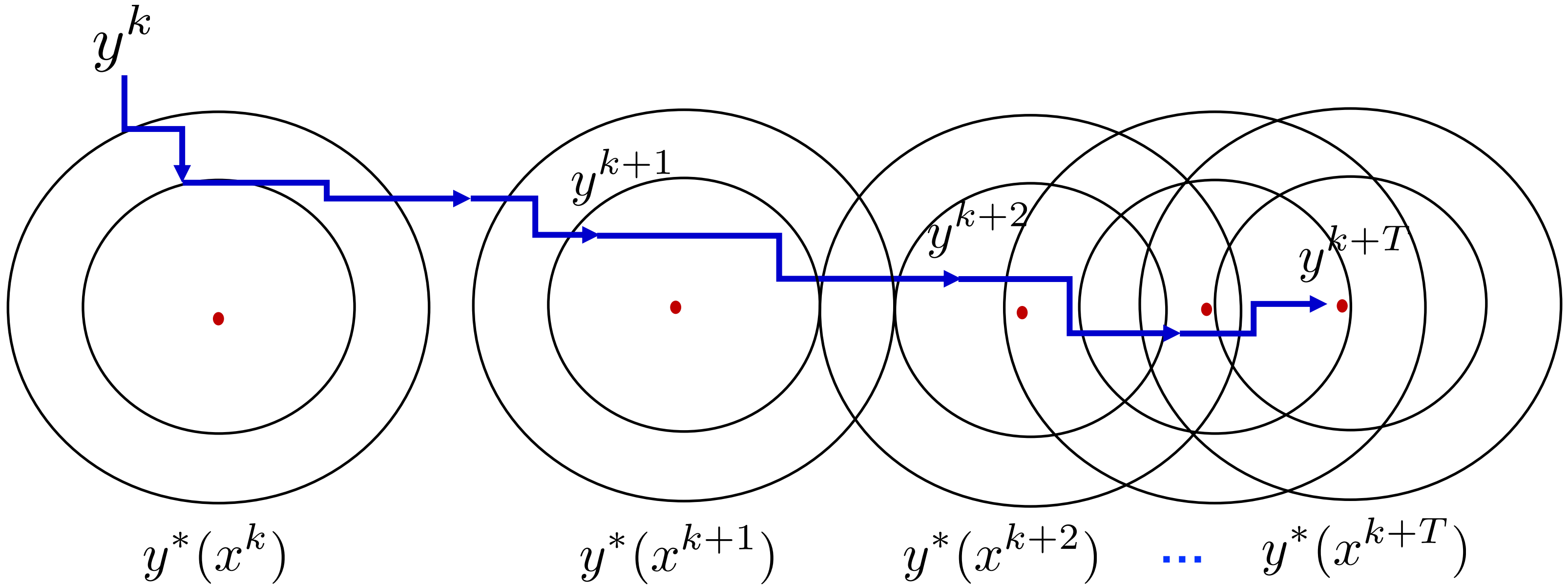
Ji, Yang, and Liang, “Bilevel optimization: Convergence analysis and enhanced design.” ***ICML 2021***

Demystify alternating SGD for bilevel problems

Q: Something not **uncovered** by these analysis?

A1: Update of *x* uses **decaying stepsizes** α_k to cancel noise; it is **slow**!

A2: The lower-level solution is **highly smooth**; its drift $\mathcal{O}(\alpha_k^2)$ is small!



Existing two-timescale/double-loop analysis does not capture this …

SGD-like guarantee for certain bilevel problems

A1) upper objective $f(x, y)$ and its gradient are Lipschitz continuous

A2) lower objective $g(x, y)$ is strongly convex and smooth in y

A3) stochastic 1st- and 2nd-order information are unbiased w/ bounded variance

Theorem (Convergence)

Under the above assumption, if we choose stepsizes $\alpha_k = \mathcal{O}(K^{-\frac{1}{2}})$ and $\beta_k = \mathcal{O}(K^{-\frac{1}{2}})$, without inner loop and increasing batchsize, ALSET satisfies

Upper level $$\frac{1}{K}\sum_{k=1}^{K}\mathbb{E}\left[\left\|\nabla F(x^k)\right\|^2\right] = \mathcal{O}\left(\frac{1}{\sqrt{K}}\right)$$

Lower level $$\mathbb{E}\left[\left\|y^K - y^*(x^K)\right\|^2\right] = \mathcal{O}\left(\frac{1}{\sqrt{K}}\right)$$

Solving (a class of) bilevel problems with SGD convergence rate!

Chen, Sun, and Yin, ``Closing the Gap: Tighter Analysis of Alternating Stochastic Gradient Methods for Bilevel Problems" ***NeurIPS* 2021**

SGD-like guarantee for certain nested problems

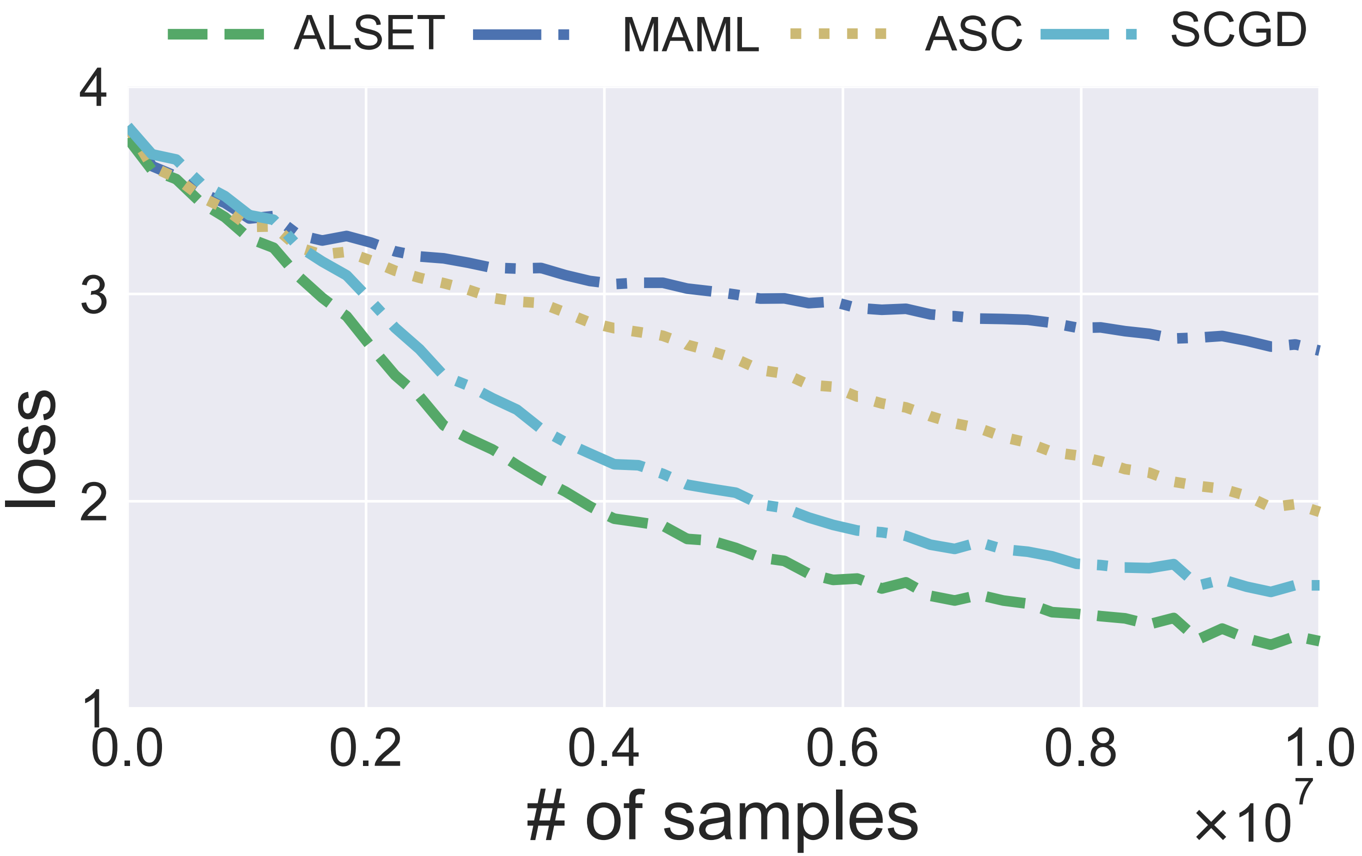
	problem class	# of loops	batch size	sample complexity
ALSET	Bilevel	Single	$\mathcal{O}(1)$	$\mathcal{O}(\epsilon^{-2})$
ALSET	Min-max	Single	$\mathcal{O}(1)$	$\mathcal{O}(\epsilon^{-2})$
ALSET	Compositional	Single	$\mathcal{O}(1)$	$\mathcal{O}(\epsilon^{-2})$
SGD	Single-level	Single	$\mathcal{O}(1)$	$\mathcal{O}(\epsilon^{-2})$

Sample complexity to achieve an ϵ-stationary point of $F(x)$; i.e., $\mathbb{E}[\|\nabla F(x)\|^2] \leq \epsilon$.

Do not be afraid of solving certain bilevel problems!

Empirical benefits in meta learning

- Meta learning for multiple sinusoidal regression tasks
- **Bilevel** SGD-based ALSET versus **standard MAML** and other bilevel baselines



Chen, Sun, Yin, "Solving stochastic compositional optimization is nearly as easy as solving stochastic optimization," ***IEEE TSP***, 2021

Other recent implicit gradient methods not covered

- **Acceleration methods for implicit gradient methods**

  [Khanduri et al., 2021], [Yang et al., 2021], [Shen and Chen, 2022], [Li et al., 2022], [Ji et al., 2022], [Huang et al., 2022], [Dagréou et al., 2022], [Chen et al., 2023], [Khanduri et al., 2023], etc

SUSTAIN: add momentum in upper- and lower-level updates

For *k* = *0, 1, 2, …, K* do

S1) $x^{k+1} = \text{momentum SGD } (x^k, y^k) \text{ on } F(x)$

S2) $y^{k+1} = \text{momentum SGD } (x^{k+1}, y^k) \text{ on } g(x^{k+1}, y)$

SUSTAIN achieves $\mathcal{O}(\epsilon^{-1.5})$ iteration complexity which is near-optimal

Khanduri, Zeng, Hong, Wai, Wang, & Yang, "A near-optimal algorithm for stochastic bilevel optimization via double-momentum," *Proc. NeurIPS 2021*

Yang, Ji, and Liang, "Provably faster algorithms for bilevel optimization," *Proc. NeurIPS 2021*

Overview of methods covered in this tutorial

$$\min_{x \in \mathcal{X}} \quad F(x)$$

$$\text{with} \quad F(x) := \min_{y \in S(x)} \quad f(x, y)$$

Nested optimization

$$\min_{x \in \mathcal{X},\, y \in \mathcal{Y}} \quad f(x, y)$$

$$\text{s.t.} \quad \text{sufficient conditions for } y \in S(x)$$

Constrained optimization

Difficulty of lower-level *y*-problems

Implicit gradient

Explicit gradient (Algorithm unrolling)

Work as SGD when implicit function is smooth, but what if it is not?

How to apply bilevel to more challenging settings?

Extend to …

Non-strongly convex lower-level problems

Overview of methods covered in this tutorial

$$\min_{x\in\mathcal{X}} \quad F(x)$$

$$\text{with} \quad F(x) := \min_{y\in S(x)} \; f(x,y)$$

Nested optimization

$$\min_{x\in\mathcal{X},\, y\in\mathcal{Y}} \quad f(x,y)$$

$$\text{s.t.} \quad \text{sufficient conditions for } y \in S(x)$$

Constrained optimization

Difficulty of lower-level *y*-problems

Implicit gradient

Limited applicability;
Great when it works

Explicit gradient

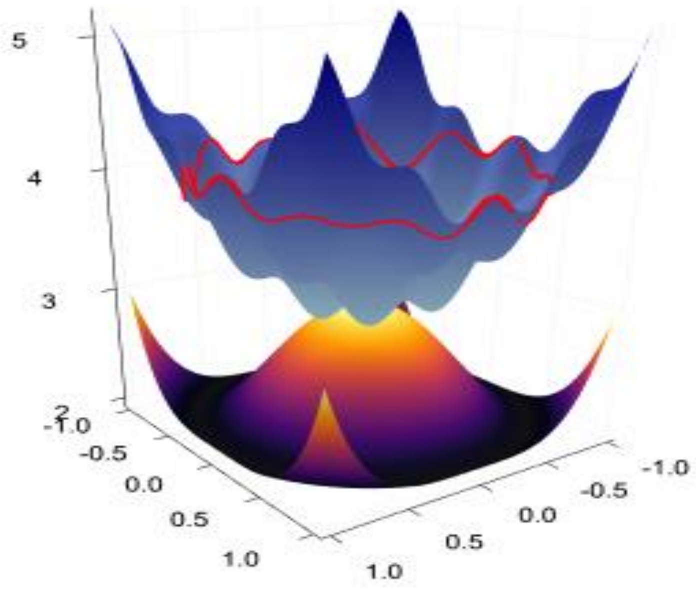
Simple to program;
Use approximations

Optimality condition

Penalty method

Challenges due to non-convexity

$$\begin{aligned} \min_{x\in\mathbb{R}^d} \quad & F(x) := f(x, y^*(x)) && \text{(upper level)} \\ \text{s.t.} \quad & y^*(x) = \arg\min_{y\in\mathbb{R}^{d'}} g(x,y) && \text{(lower level)} \end{aligned}$$



Non-unique solutions

Non-differentiable upper-level loss; non-invertible Hessian

$$\begin{aligned} \min_{x\in\mathcal{X},\, y\in\mathcal{Y}} \quad & f(x,y) \\ \text{s.t.} \quad & \text{sufficient conditions for } y \in S(x) \end{aligned}$$

Constrained reformulation

Main assumption: Polyak-Łojasiewicz (PL) condition

Loss of **over-parametrized** model is non-convex

but satisfies the PL-inequality:

$$\boxed{\left\|\nabla_y g\left(x,y\right)\right\|^2 \gtrsim g\left(x,y\right) - \min_y\, g\left(x,y\right)}$$

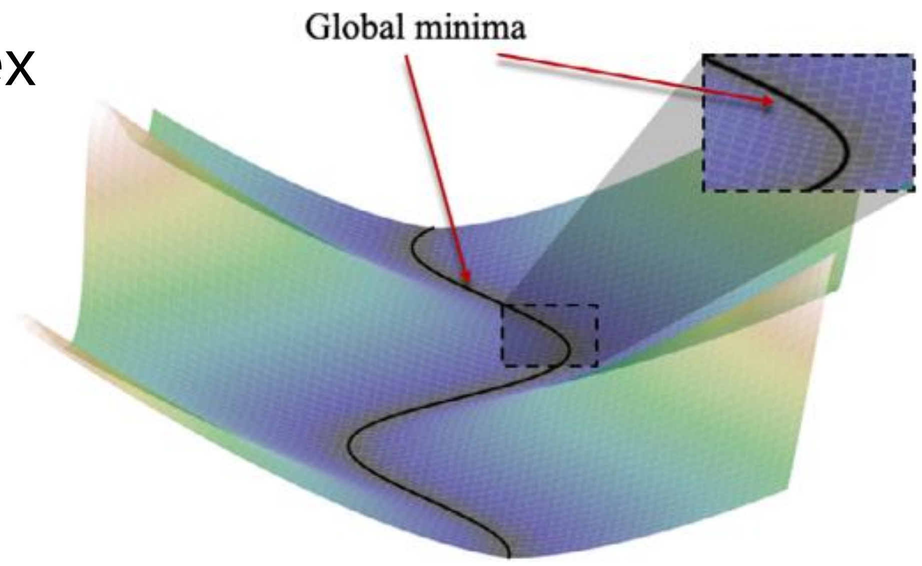


Image from [Liu, 2022]

All stationary points are global optimal solutions…

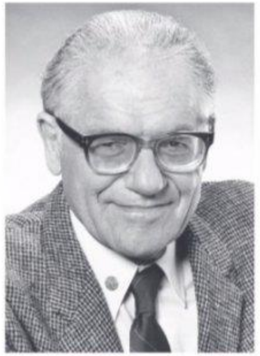
Liu, Zhu, and Belkin, ``Loss landscapes and optimization in over-parameterized non-linear systems and neural networks” ***Applied and Computational Harmonic Analysis***

Constrained reformulation

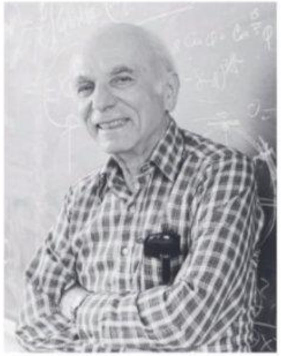
$$\min_{x,y} \; f(x,y)$$

$s.t.$ equivalent condition of LL optimality

KKT conditions



William Karush, circa 1987



Fritz John at NYU, circa 1987

Harold Kuhn and Albert Tucker, 1980
at von Neumann Prize presentation

Constraint qualification (CQ) conditions:

Ensure KKT conditions are necessary optimality conditions.

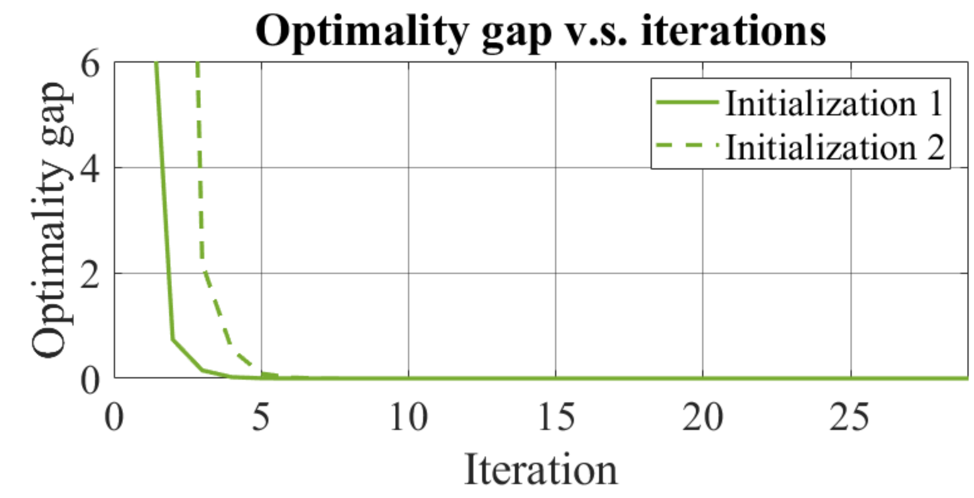
Constrained reformulation

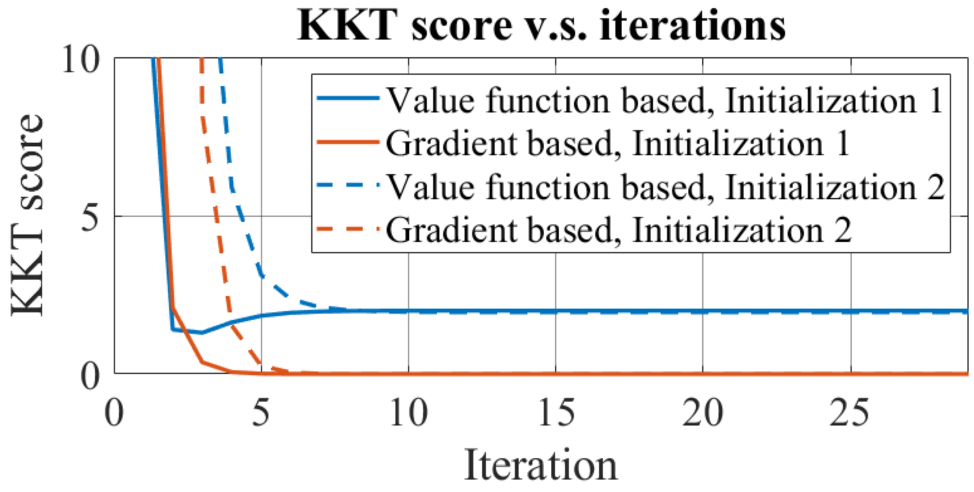
Value-function based

$$\min_{x,y\in\mathbb{R}^d} \quad f(x,y)$$
$$\text{s.t.} \quad g(x,y) - \min_{y'\in\mathbb{R}^d} g(x,y') = 0$$

Gradient based

$$\min_{x,y\in\mathbb{R}^d} \quad f(x,y)$$
$$\text{s.t.} \quad \nabla_y g(x,y) = 0$$



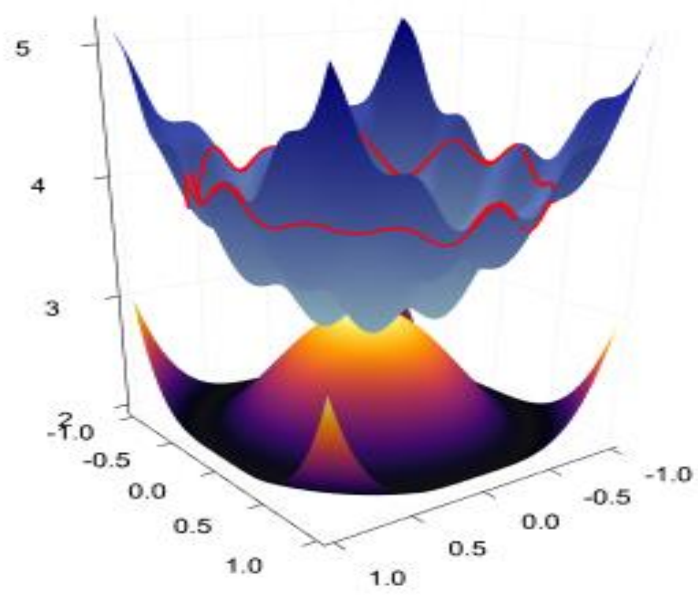


Gradient based KKT conditions hold at global optimal set!

Xiao, Lu, Chen, "A Generalized Alternating Method for Bilevel Optimzation under the Polyak-Łojasiewisz Condition," ***NeurIPS 2023***

Can we formally justify this phenomenon?

—— Identify the CQ satisfied by PL bilevel problems



Calmness condition

Definition (Calmness CQ)

Let (x^*, y^*) be the global optimal point of

$$\min_{x,y} \; f(x,y) \;\; \text{s.}t. \;\; h(x,y) = 0 \; .$$

If there exists ***ε*** and ***M*** s.t. for any $\|q\| \leq \epsilon$ and $\|(x', y') - (x^*, y^*)\| \leq \epsilon$ which satisfies $h(x', y') + q = 0$, one has

$$f(x', y') - f(x^*, y^*) + M\|q\| \geq 0$$

then the problem is said to be calm.

- Quantifies the **sensitivity of the objective to the perturbation on constraints.**
- **Key observation:** gradient based PL bilevel problems inherit **the calmness CQ**!

New necessary condition

Theorem (Necessary condition of KKT)

If $g(x,\cdot)$ satisfies the PL condition and is smooth, and $f(x,\cdot)$ is Lipschitz continuous, then there exists $w^* \neq 0$ such that

$$\mathcal{R}_x(x^*, y^*, w^*) := \|\nabla_x f(x^*, y^*) + \nabla_{xy}^2 g(x^*, y^*) w^*\|^2 = 0$$
$$\mathcal{R}_w(x^*, y^*, w^*) := \|\nabla_{yy}^2 g(x^*, y^*) \left(\nabla_y f(x^*, y^*) + \nabla_{yy}^2 g(x^*, y^*) w^*\right)\|^2 = 0$$
$$\mathcal{R}_y(x^*, y^*) := \|\nabla_y g(x^*, y^*)\|^2 = 0$$

hold at the global minimizer (x^*, y^*) of the PL bilevel problem.

- **Compare with KKT: Shadow implicit gradient:**

$$w^*(x,y) \in \arg\min_w \ \mathcal{L}(x,y;w) := \frac{1}{2}\left\|\nabla_y f(x,y) + \nabla_{yy}^2 g(x,y) w\right\|^2$$

A generalized alternating gradient method

Using fixed-point equation and the alternating strategy:

For k = 0, 1, 2, …, K do

S1) $y^{k+1} = \text{One or multiple GD updates } (y^k) \text{ on } g(x^k, y)$

S2) $w^{k+1} = \text{GD updates } (w^k) \text{ on } \mathcal{L}(x^k, y^{k+1}, w)$

S3) $x^{k+1} = x^k - \alpha(\nabla_x f(x^k, y^{k+1}) + \nabla_{xy}^2 g(x^k, y^{k+1}) w^{k+1})$

GALET : Generalized ALternating mEthod for bilevel opTimization

Xiao, Lu, Chen, "A Generalized Alternating Method for Bilevel Optimization under the Polyak-Łojasiewisz Condition," ***NeurIPS 2023.***

Convergence results: GD-like guarantee

A1) upper objective $f(x, y)$ and its gradient are Lipschitz continuous

A2) lower objective $g(x, y)$ is PL , smooth and Hessian-Lipschitz in y

A3) The nonzero eigenvalue of the Hessian of $g(x, y)$ is bounded away from 0

Theorem (Convergence)

Under the above assumptions, if we choose stepsizes properly, the iterates generated by the GALET satisfies

Upper-level $$\frac{1}{K}\sum_{k=0}^{K-1}\mathcal{R}_x(x^k, y^k, w^k) = \mathcal{O}\left(\frac{1}{K}\right)$$ Lower-level $$\frac{1}{K}\sum_{k=0}^{K-1}\mathcal{R}_y(x^k, y^k) = \mathcal{O}\left(\frac{1}{K}\right)$$

Shadow implicit gradient level $$\frac{1}{K}\sum_{k=0}^{K-1}\mathcal{R}_w(x^k, y^k, w^k) = \mathcal{O}\left(\frac{1}{K}\right)$$

GALET enjoys the same convergence rate as GD!

Xiao, Lu, Chen, "A Generalized Alternating Method for Bilevel Optimization under the Polyak-Łojasiewisz Condition," ***NeurIPS 2023.***

Overview of methods covered in this tutorial

$$\min_{x \in \mathcal{X}} \quad F(x)$$

$$\text{with} \quad F(x) := \min_{y \in S(x)} \quad f(x,y)$$

Nested optimization

$$\min_{x \in \mathcal{X},\, y \in \mathcal{Y}} \quad f(x,y)$$

$$\text{s.t.} \quad \text{sufficient conditions for } y \in S(x)$$

Constrained optimization

Difficulty of solving lower-level *y*-problems

Implicit gradient | **Explicit gradient** | **Optimality condition** | **Penalty method**

Limited applicability;
Great when it works

Penalty-based reformulations

Under the PL condition, both of the following functions are optimality metrics.

$$p(x,y) = g(x,y) - v(x) \quad \text{with} \quad v(x) := \min_{y} \; g(x,y)$$

$$p(x,y) = \left\|\nabla_y g\left(x,y\right)\right\|^2$$

$$\begin{aligned} &\min_{x\in\mathcal{X},\, y\in\mathcal{Y}} \quad f(x,y) \\ &\quad \text{s.t.} \quad \text{sufficient condition}: \; p(x,y) \leq 0 \end{aligned}$$

Constrained reformulation

Constrained versus penalized reformulations

Consider a slightly relaxed version of bilevel problem:

$$\mathcal{BP}_\epsilon : \ \min_{x,y} f(x,y) \qquad \text{s.t. } p(x,y) \leq \epsilon$$

$$\Updownarrow \qquad \textbf{Equivalence?}$$

$$\mathcal{BP}_{\gamma p} : \ \min_{x,y} f(x,y) + \gamma p(x,y)$$

Equivalence: all local and global solutions match…

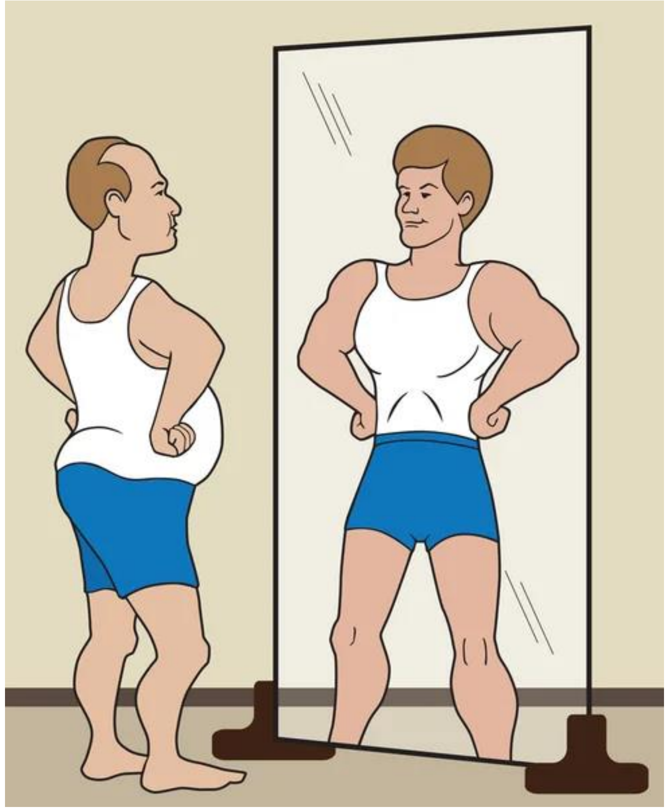


Image from depositphotos.com

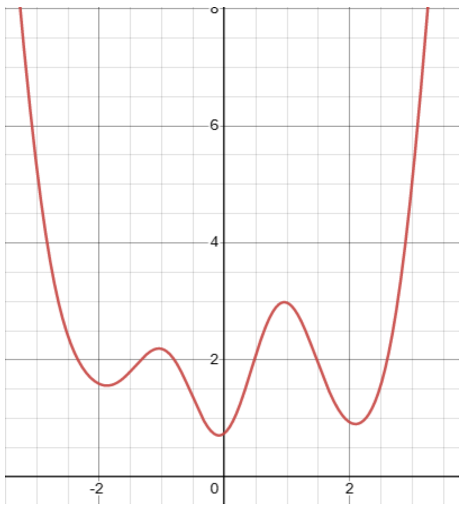
Difficulty in preserving local solutions

$$\mathcal{BP}_{\epsilon=0}: \min_{x,y\in\mathbb{R}} \sin^2\left(y-\frac{2\pi}{3}\right) \text{ s.t. } \|\nabla_y g(x,y)\|^2 = (y+\sin(2y))^2 \le 0$$

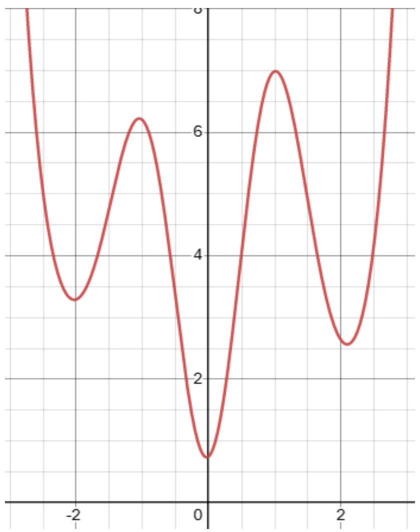
Solution is 0

Penalize with gradient norm

$$\mathcal{BP}_{\gamma p}: \min \sin^2\left(y-\frac{2\pi}{3}\right) + \gamma\,(y+\sin(2y))^2$$



$\gamma \uparrow$



Local solution $\frac{2\pi}{3}$

Shen, Xiao and **Chen**, "On Penalty-based Bilevel Gradient Descent Method," ***ICML 2023***

Conditions of preserving local solutions

$$\mathcal{BP}_\epsilon : \min_{x,y} f(x,y) \quad \text{s.t. } p(x,y) \le \epsilon$$

Equivalence?

$$\mathcal{BP}_{\gamma p} : \min_{x,y} f(x,y) + \gamma p(x,y)$$

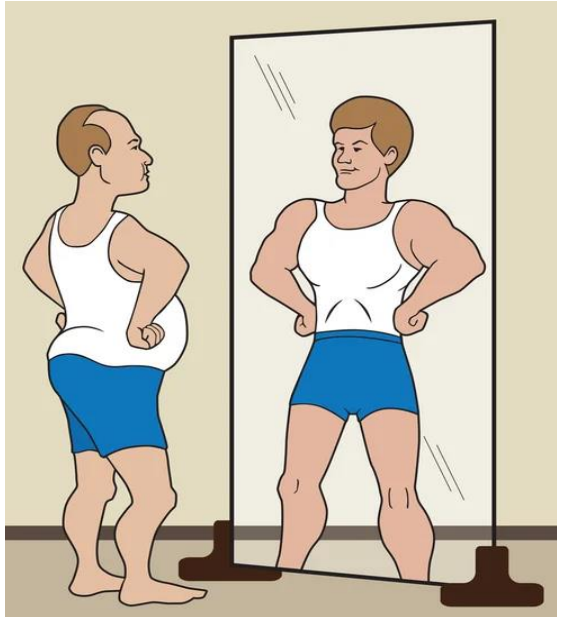


Image from depositphotos.com

Theorem (equivalence)

Given any $\epsilon > 0$, choose the penalty constant $\gamma \gtrsim \epsilon^{-0.5}$.

i) For $p(x,y) = g(x,y) - v(x)$, no further assumption is needed;

ii) For $p(x,y) = \|\nabla_y g(x,y)\|^2$, further assume the singular values >0;

Then any local solution of the penalized problem $\mathcal{BP}_{\gamma p}$ is a local solution of the ϵ-approximate original bilevel problem $\mathcal{BP}_\epsilon$.

Shen, Xiao and **Chen**, "On Penalty-based Bilevel Gradient Descent Method," ***ICML 2023***

An alternative method: Penalty-based gradient descent

$$\min_{x,y} F_\gamma(x,y) := f(x,y) + \gamma(g(x,y) - v(x)) \text{ with } v(x) := \min_y g(x,y)$$

Gradient of value function is computed by a generalized **Daskin's theorem**:

$$\nabla_x F_\gamma(x,y) = \nabla_x f(x,y) + \gamma(\nabla_x g(x,y) - \nabla_x g(x,y^*)), \; y^* \in \arg\min g(x,y)$$

For k = $0, 1, 2, \ldots, K$ do

S1) $x_{k+1} = x_k - \alpha \left(\nabla_x f(x_k, y_k) + \gamma \left(\nabla_x g(x_k, y_k) - \nabla_x g(x_k, \hat{y}_k^{T+1})\right)\right)$

S2) $y_{k+1} = y_k - \alpha \left(\nabla_y f(x_k, y_k) + \gamma \nabla_y g(x_k, y_k)\right)$

- **One only needs first-order derivatives!**

Training efficiency for nonconvex bilevel problems

$$\min_{x,y} F_\gamma(x,y) := f(x,y) + \gamma(g(x,y) - v(x)) \text{ with } v(x) := \min_y g(x,y)$$

Theorem (convergence)

Consider running V-PBGD for $k = 1, 2, ..., K$. With small enough step sizes and $T_k \gtrsim \log k$, it holds that

$$\frac{1}{K}\sum_{k=1}^{K} \|\nabla F_\gamma(x_k, y_k)\|^2 = \mathcal{O}\left(\frac{\gamma}{K}\right)$$

- With $\gamma \gtrsim \epsilon^{-0.5}$, it implies the $\mathcal{O}(\epsilon^{-1.5})$ iteration complexity

Shen, Xiao and **Chen**, "On Penalty-based Bilevel Gradient Descent Method," ***ICML 2023***

Overview of methods covered in this tutorial

$$\min_{x \in \mathcal{X}} \quad F(x)$$

$$\text{with} \quad F(x) := \min_{y \in S(x)} \quad f(x, y)$$

Nested optimization

$$\min_{x \in \mathcal{X},\, y \in \mathcal{Y}} \quad f(x, y)$$

$$\text{s.t.} \quad \text{sufficient conditions for } y \in S(x)$$

Constrained optimization

Difficulty of lower-level *y*-problems

Implicit gradient

Limited applicability;
Great when it works

Explicit gradient

Simple to program;
Incur approximations

Optimality condition

Limited applicability;
Great when it works

Penalty method

Broad applicability;
Often require relaxations

Other recent advances not covered

- **Acceleration methods for implicit gradient methods**

  [Khanduri et al., 2021], [Yang et al., 2021], [Shen and Chen, 2022], [Li et al., 2022], [Ji et al., 2022], [Huang et al., 2022], [Dagréou et al., 2022], [Chen et al., 2023], [Khanduri et al., 2023], etc

- **Memory-efficient variants for algorithm unrolling methods**

  [Maclaurin et al., 2015], [Pedregosa 2016], [Franceschi et al., 2017, 2018], [Nichol et al., 2018], [Shaban et al., 2019], [Grazzi et al., 2020], [Liu et al., 2021], [Liu et al., 2022], [Bolte et al., 2022]

- **Penalty and primal-dual methods for bilevel optimization**

  [Ye et al., 1997], [Lin et al., 2014], [Liu et al., 2021], [Mehra and Hamm, 2021], [Sow et al., 2022], [Gao et al., 2022], [Ye et al., 2022], [Lu and Mei 2023], [Huang 2023], [Kwon et al., 2023], etc

Simulation: Data hyper-cleaning

In data hyper-cleaning, we try to clean up the polluted training data

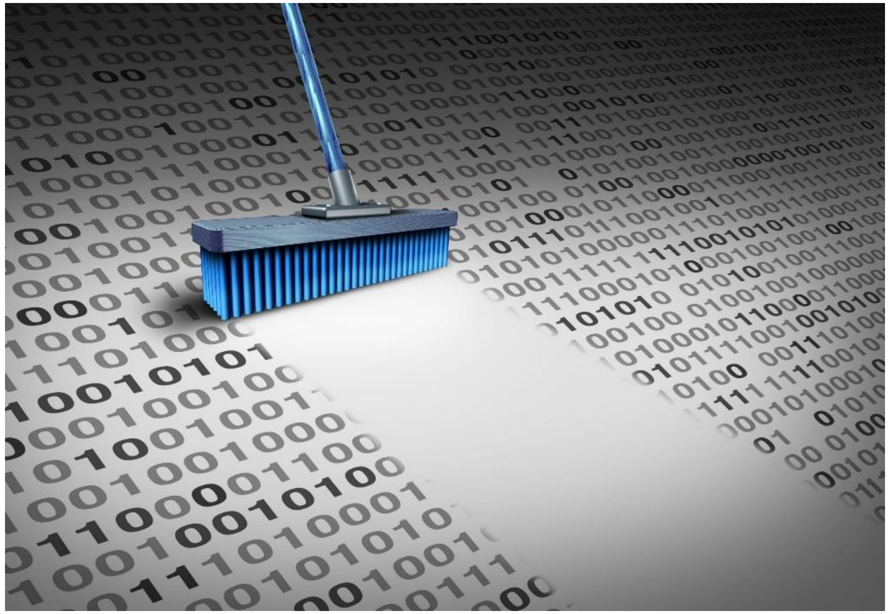
$\mathcal{D}_{tr}$ has polluted data $\mathcal{D}_{val}$ is clean

Want to learn an importance weight for each data

$$\omega_i(x), d_i \in \mathcal{D}_{tr}$$

Given weights, the models fit the weighted data

$$\sum_{d_i \in \mathcal{D}_{tr}} \omega_i(x) f_{ce}(y; d_i) - \min_y \sum_{d_i \in \mathcal{D}_{tr}} \omega_i(x) f_{ce}(y; d_i) \le \epsilon$$



Simulation: Data hyper-cleaning

We want such models to fit well with clean data:

$$\min_{x,y} \sum_{d_i \in \mathcal{D}_{val}} f_{ce}(y; d_i) \quad \text{s.t.} \sum_{d_i \in \mathcal{D}_{tr}} \omega_i(x) f_{ce}(y; d_i) - \min_{y} \sum_{d_i \in \mathcal{D}_{tr}} \omega_i(x) f_{ce}(y; d_i) \leq \epsilon$$

We evaluate all algorithms with three main metrics:

- Test accuracy: classification accuracy of y
- F1 score: precision and recall of cleaner x
- Scalability: Peak GPU memory usage through training and inference

Simulation: Data hyper-cleaning

Nested optimization

Constrained optimization

Method	Linear model		2-layer MLP	
	Test accuracy	F1 score	Test accuracy	F1 score
RHG	87.64 ± 0.19	89.71 ± 0.25	87.50 ± 0.23	89.41 ± 0.21
T-RHG	87.63 ± 0.19	89.04 ± 0.24	87.48 ± 0.22	89.20 ± 0.21
BOME	87.09 ± 0.14	89.83 ± 0.18	87.42 ± 0.16	89.26 ± 0.17
G-PBGD	90.09 ± 0.12	90.82 ± 0.19	92.17 ± 0.09	90.73 ± 0.27
IAPTT-GM	$\mathbf{90.44} \pm 0.14$	$\mathbf{91.89} \pm 0.15$	91.72 ± 0.11	91.82 ± 0.19
V-PBGD	$\mathbf{90.48} \pm 0.13$	$\mathbf{91.99} \pm 0.14$	$\mathbf{94.58} \pm 0.08$	$\mathbf{93.16} \pm 0.15$

	RHG	T-RHG	BOME	**G-PBGD**	IAPTT-GM	**V-PBGD**
GPU memory (MB) linear	1369	1367	1149	1149	1237	1149
GPU memory (MB) MLP	7997	7757	1201	1235	2613	1199
Runtime (sec.) linear	73.21	32.28	5.92	7.72	693.65	9.12
Runtime (sec.) MLP	94.78	54.96	39.78	185.08	1310.63	207.53

- V-PBGD does not have as large memory increase, thanks to being first-order

Outline

- ❑ Part I - Introduction and background
- ❑ Part II – Bilevel optimization fundamentals
- ❑ **Part III – Bilevel applications to reinforcement learning (60 mins)**
- ❑ Part IV – Multi-objective learning beyond bilevel optimization
- ❑ Part V - Conclusions and open directions

Tutorial Part III:

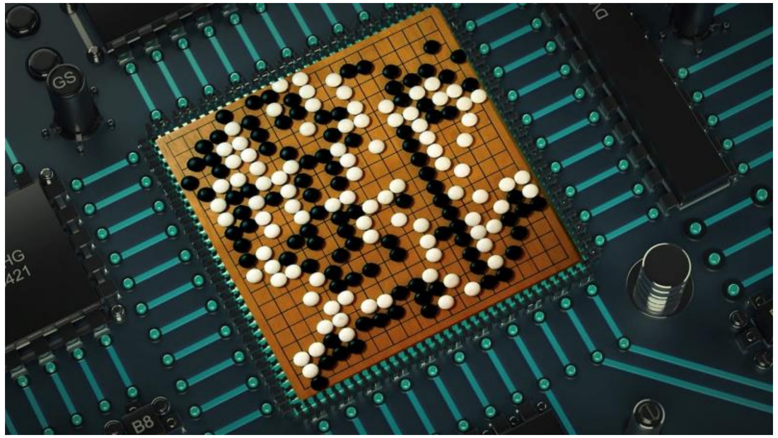
Bilevel applications to reinforcement learning

Zhuoran Yang

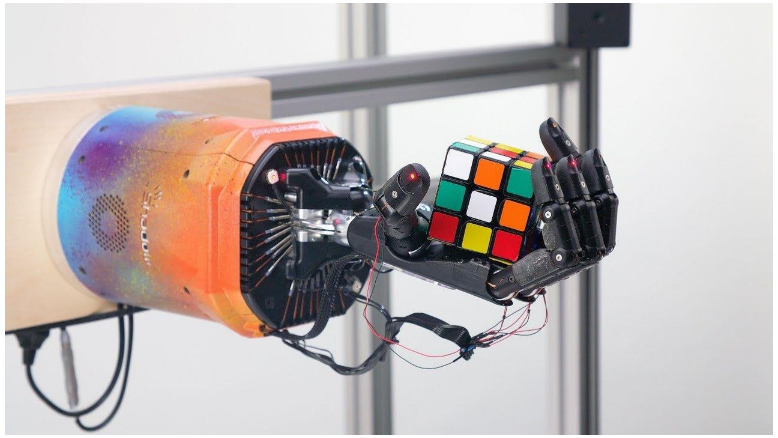
Yale Statistics and Data Science

February 20, 2024

Empirical successes of reinforcement learning



AlphaGo



Rubik's cube

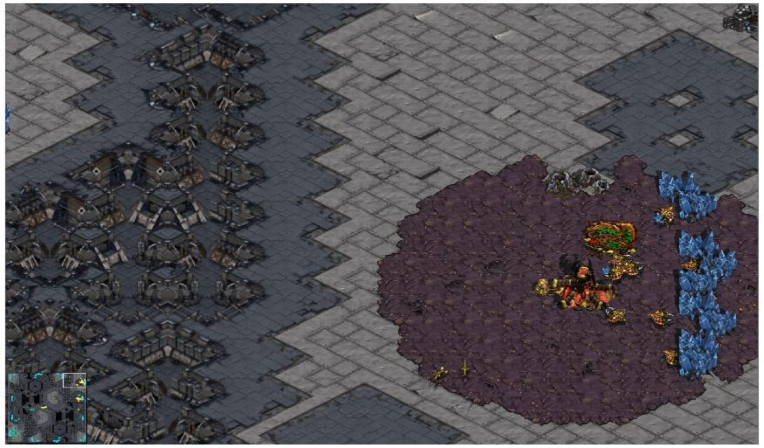
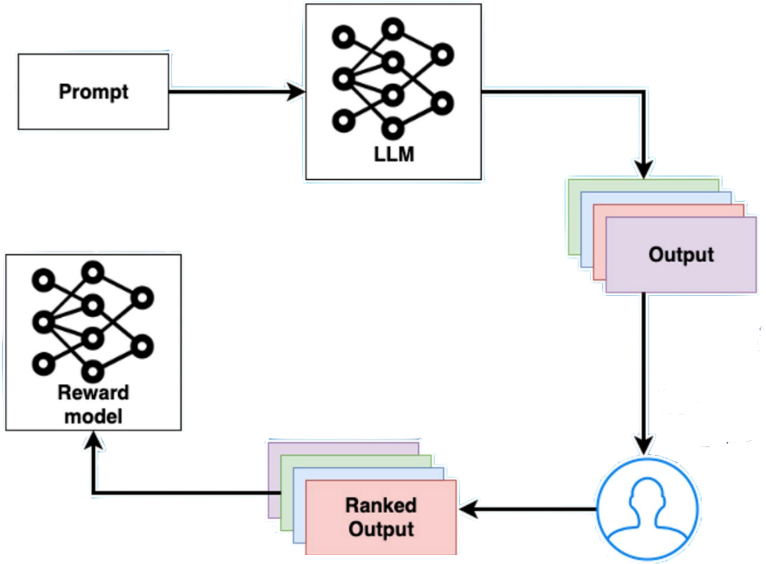


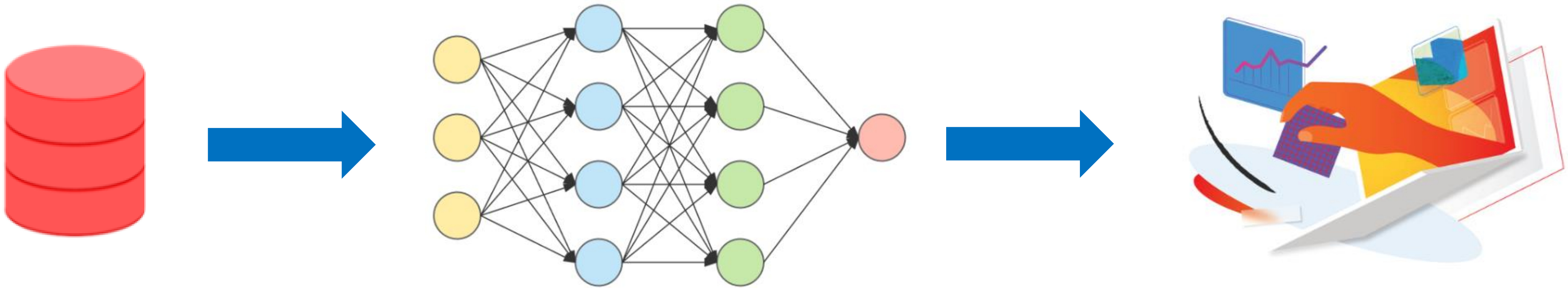
Image from Internet

Computer games



Finetune LLMs

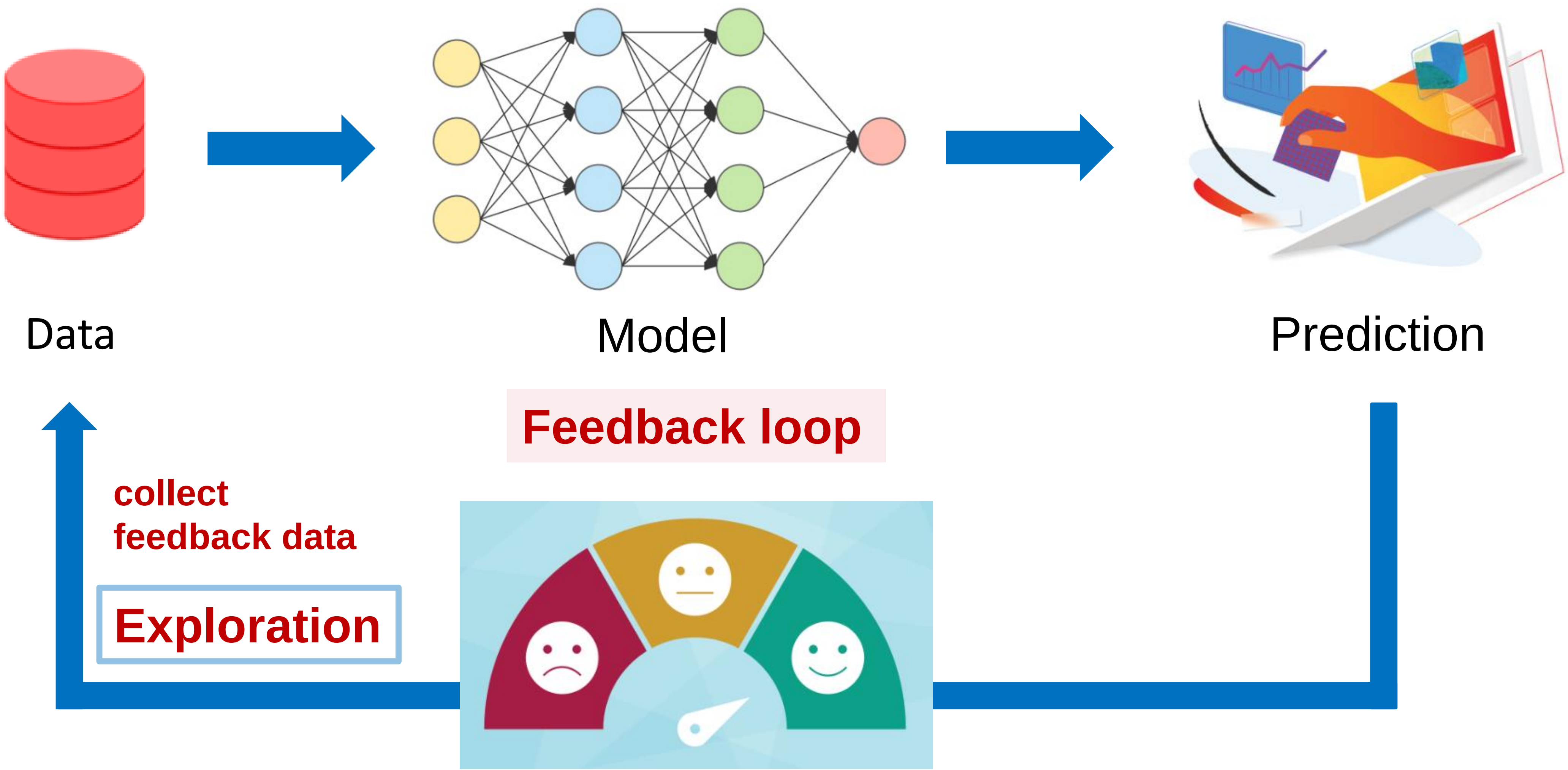
Supervised learning



Data Model Prediction

Collect data, train model, and make predictions with the model

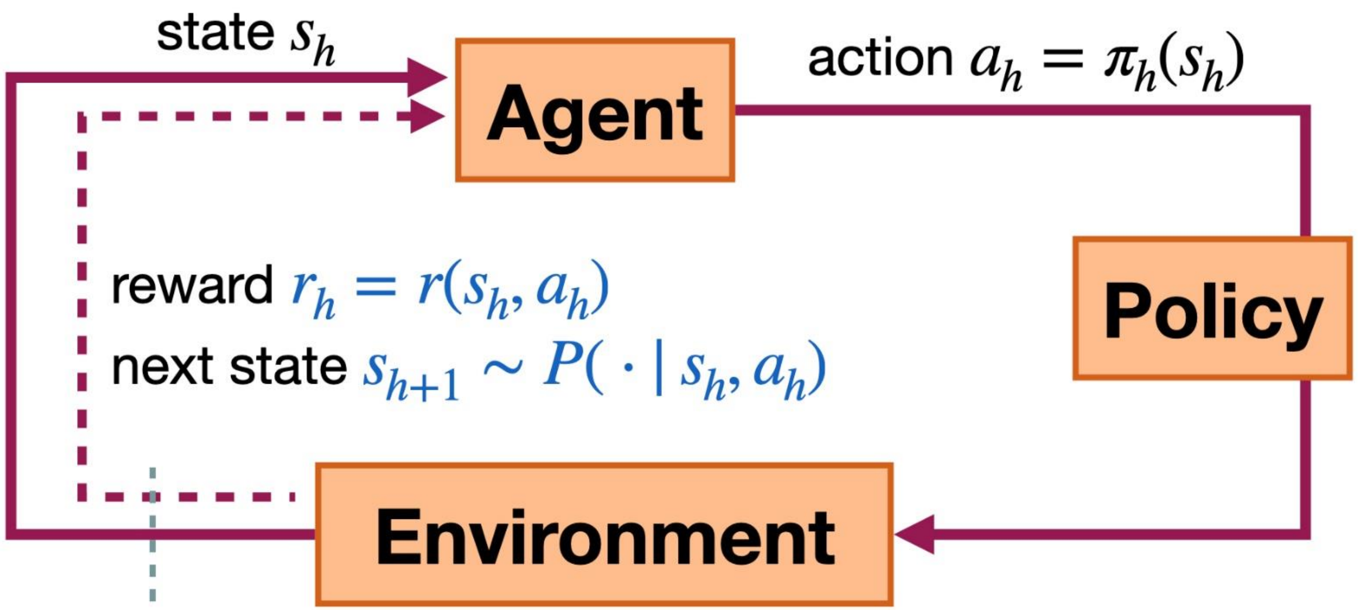
Reinforcement learning

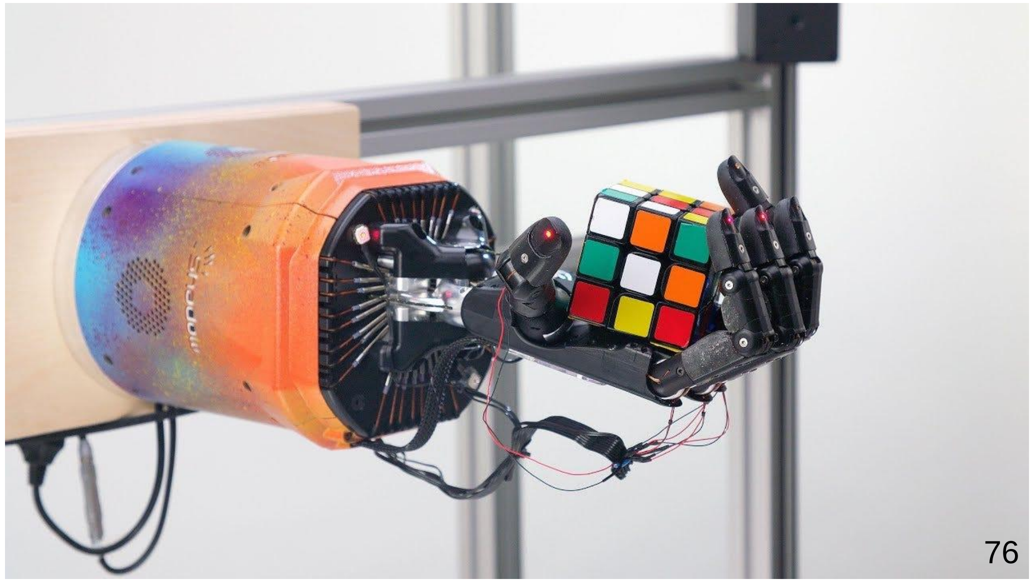


Single-agent reinforcement learning

Single-agent RL: One agent takes an action a_h at each step h

- Environment state s_{h+1} evolves according to agent's action a_h
- Goal: maximize the cumulative rewards $\sum_{h=1}^{H} r(s_h, a_h)$
- Solution concept: optimal policy π^* – reward maximizing policy

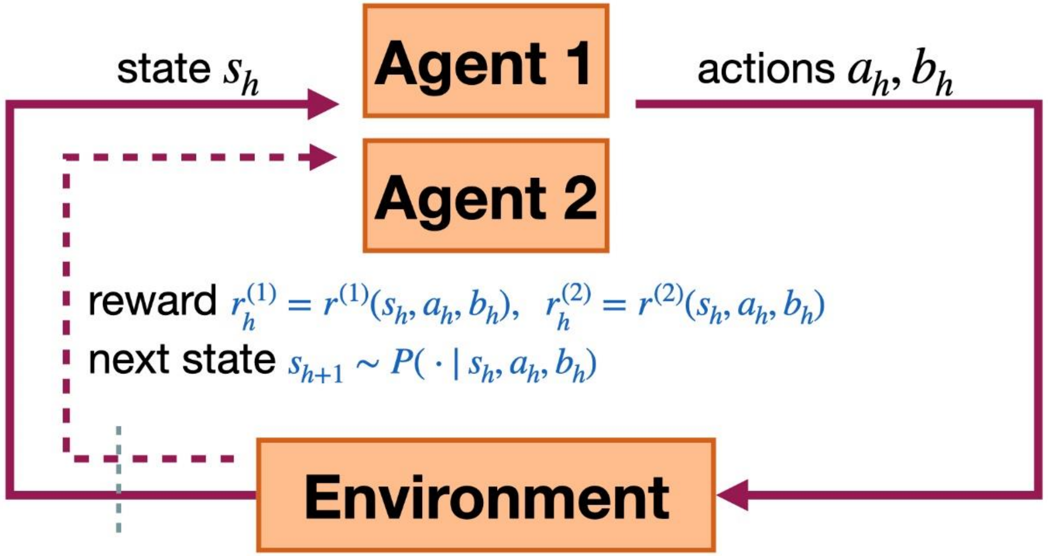


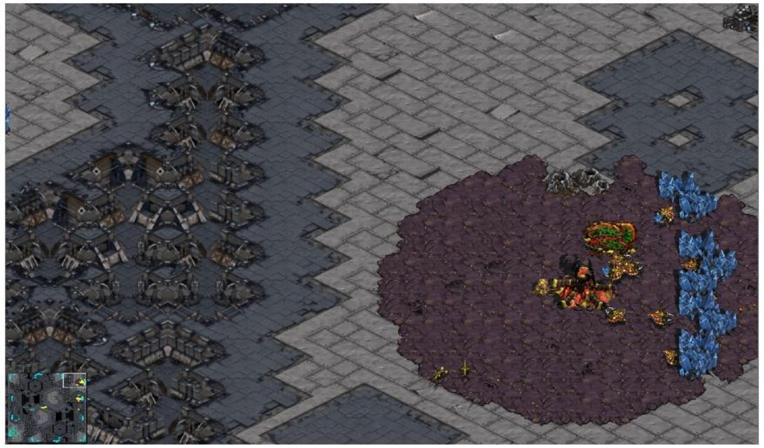


Multi-agent reinforcement learning

Multi-agent RL: Multiple agents, each takes an action at each step

- Environment state evolves according to **actions of all agents**
- Multi-objective optimization:
  - each agent aims to maximize her own cumulative rewards
- Game-theoretic solution concepts:
  - Markov perfect equilibria, Coarse correlated equilibria, …





Bilevel optimization meets reinforcement learning

Bilevel RL: Multi-agent RL + leader-follower structure

Upper-level variable $x = \text{policy of leader}$

$$\max_{x \in \mathcal{X}, y} \quad f(x, y) \qquad \text{(upper level)}$$

$$\text{s.t.} \quad y \in \arg\max_{y' \in \mathcal{Y}} \; g(x, y') \qquad \text{(lower level)}$$

Lower-level variable $y = \text{policy of follower}$

Leader's problem

Follower's problem

$$f(x, y) = \mathbb{E}_{x,y}[\textstyle\sum_{h=1}^{H} R(s_h, a_h, b_h)]$$

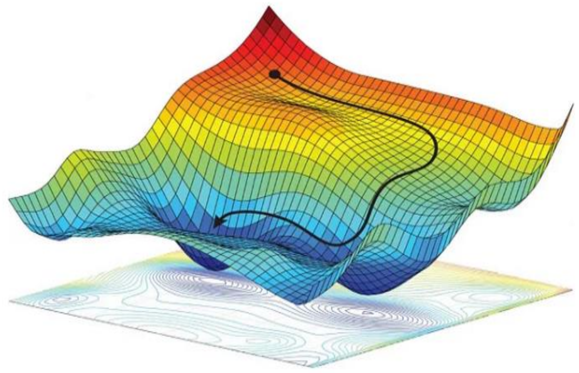
$$g(x, y) = \mathbb{E}_{x,y}[\textstyle\sum_{h=1}^{H} r(s_h, a_h, b_h)]$$

Leader and follower have different reward functions

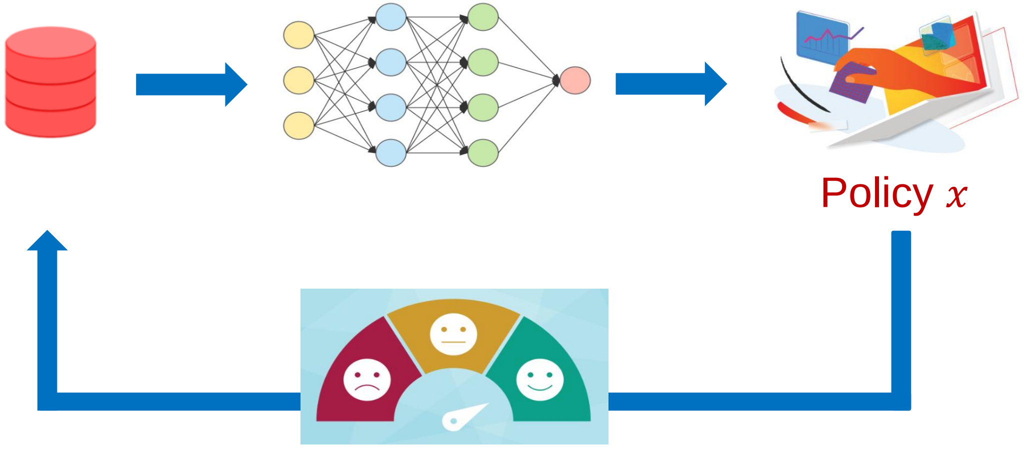
$f(x, y)$ and $g(x, y)$ are cumulative rewards of leader and follower

Hierarchical structure – follower's problem as leader's constraint

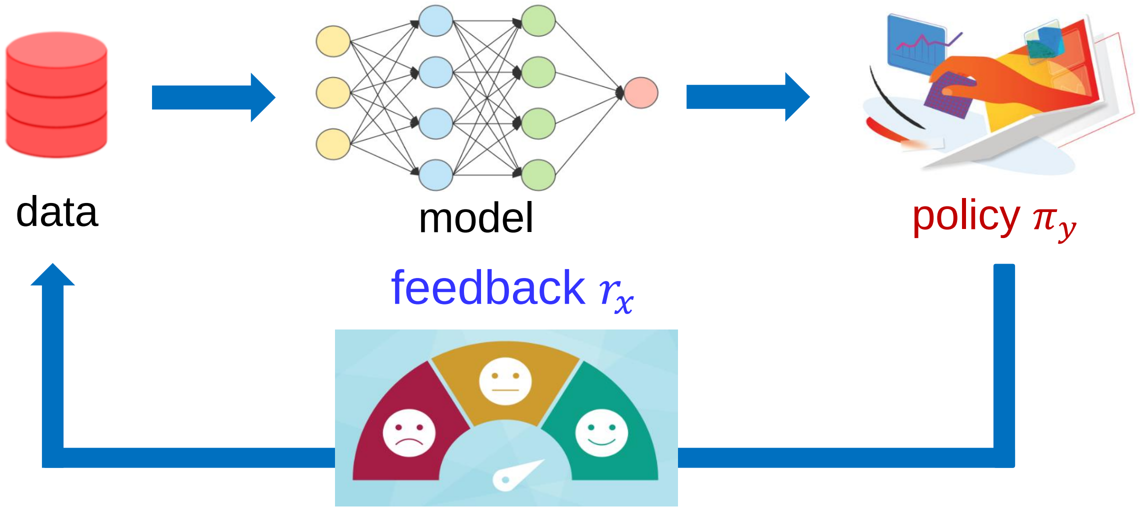
Interpretation of bilevel RL



$$\min_{x,y} f(x,y)$$



RL problem with policy x



RL problem with parameter x

Nested in

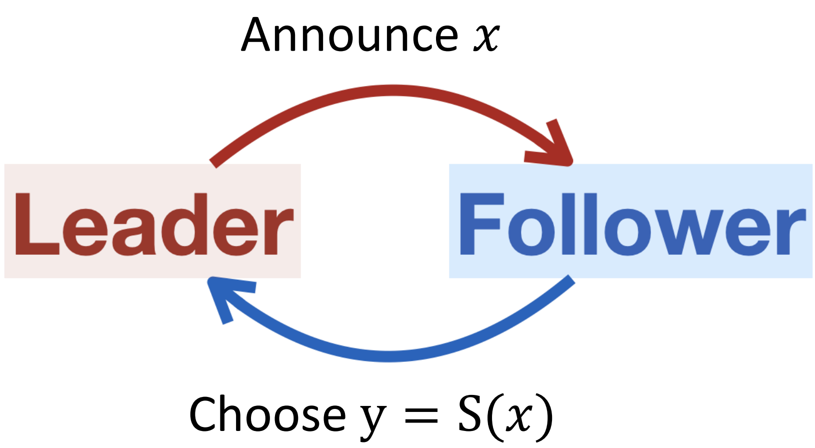
Interpretation of bilevel RL

$$\max_{x \in \mathcal{X}, y} \quad f(x,y) \qquad \text{(upper level)}$$
$$\text{s.t.} \quad y \in \arg\max_{y' \in \mathcal{Y}} \quad g(x,y') \qquad \text{(lower level)}$$

Upper: Find leader's optimal policy

Lower: follower always adopts best response

- Leader announces a policy x, promise she will play x
- Follower decides his policy y – best-response to x
- Leader and follower then play (x, y) simultaneuously
- A sequence of state-action-rewards are generated
- Leader and follower receive $f(x, y)$ and $g(x, y)$ in total

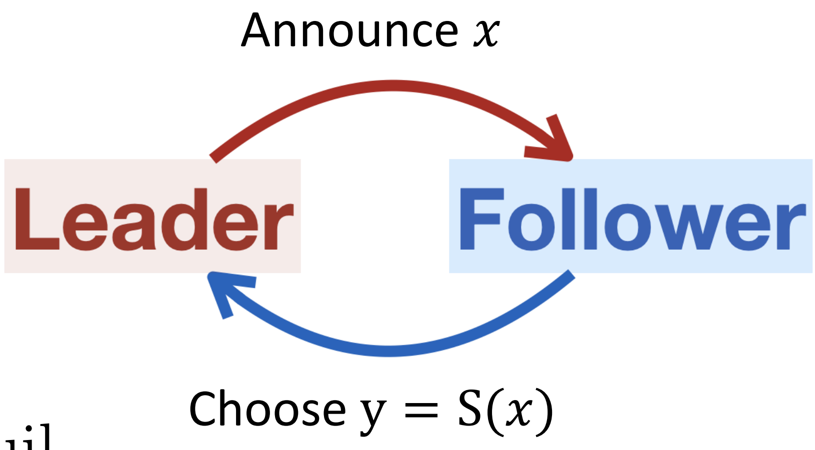


Interpretation of bilevel RL

$$\begin{aligned} \max_{x \in \mathcal{X}, y} \quad & f(x, y) && \text{(upper level)} \\ \text{s.t.} \quad & y \in \arg\max_{y' \in \mathcal{Y}} \; g(x, y') && \text{(lower level)} \end{aligned}$$

Upper: Find leader's optimal policy

Lower: follower always adopts best response



Follower's best response: $y = S(x) \in \arg\max_{y' \in \mathcal{Y}} g(x, y')$

Leader's optimization: $\max_{x \in \mathcal{X}} f\left(x, S(x)\right)$

Optimal solution pair: $\left(x^*, S(x^*)\right) = \text{Stackelberg equil.}$

x^* is leader's optimal policy given that follower responds optimally

A more general view of bilevel RL

More generally, we can have **multiple leaders** (n) **and followers** (m)

Upper-level variable $\{x^1, \dots, x^m\}$ = policies of leaders

$$\text{Solve} \quad \text{Equil}\big(\{f^i(x^1, \dots, x^m, y^1, \dots, y^n), i = 1, \dots, m\}\big)$$
$$\text{s.t.} \quad (y^1, \dots, y^n) \in \text{Equil}\big(\{g^i(x^1, \dots, x^m, y^1, \dots, y^n), j = 1, \dots, n\}\big)$$

Lower-level variable $\{y^1, \dots, y^n\}$ = policies of followers

Leaders announce their policies and promise to commit to them

Followers form an equilibrium induced by leaders' policies

Each leader's goal: steer the system in her favor (game of leaders)

Example: Stackelberg game

$$\min_{x \in \mathcal{X}, y} \quad f(x, y) \qquad \text{(upper level)}$$

$$\text{s.t.} \quad y \in \arg\min_{y' \in \mathcal{Y}} \; g(x, y') \qquad \text{(lower level)}$$

Upper: Find leader's optimal policy

Lower: follower always adopts best response

Matrix game with action spaces $\mathcal{A} = \{a_1, \ldots, a_m\}, \mathcal{B} = \{b_1, \ldots, b_n\}$

Reward functions $R(a, b), r(a, b)$ Policies $x \in \mathcal{X} = \Delta_m, y \in \mathcal{Y} = \Delta_n$

$$f(x, y) = \mathbb{E}_{a \sim x, b \sim y}[R(a, b)], \quad g(x, y) = \mathbb{E}_{a \sim x, b \sim y}[r(a, b)]$$

Best response $S(x) = \delta\{\arg\max_{b \in \mathcal{B}} \mathbb{E}_{a \sim x}[r(a, b)]\}$

Follower labels leader's x using a deterministic function

von Stackelberg, H. (1952). The theory of the market economy. Oxford University Press.

Example: Stackelberg game with quantal response

$$\min_{x\in\mathcal{X},y} \quad f(x,y) \qquad \text{(upper level)}$$
$$\text{s.t.} \quad y \in \arg\min_{y'\in\mathcal{Y}} \; g(x,y') \qquad \text{(lower level)}$$

Upper: Find leader's optimal policy

Lower: follower always adopts best response

Matrix game with action spaces $\mathcal{A} = \{a_1, \ldots, a_m\}, \mathcal{B} = \{b_1, \ldots, b_n\}$

Reward functions $R(a,b), r(a,b)$ Policies $x \in \mathcal{X} = \Delta_m, y \in \mathcal{Y} = \Delta_n$

$$f(x,y) = \mathbb{E}_{a\sim x, b\sim y}[R(a,b)], \quad g(x,y) = \mathbb{E}_{a\sim x, b\sim y}[r(a,b)] + \eta^{-1}\cdot\mathcal{H}(y) \Leftarrow \text{entropy}$$

Quantal response $S(x)(b) = Z(x)^{-1}\cdot\exp(\eta\cdot r(x,b))$ Stochastic response

McKelvey, Richard D., and Thomas R. Palfrey. "Quantal response equilibria for normal form games." *Games and economic behavior* 10, no. 1 (1995): 6-38.
Černý, Jakub, Viliam Lisý, Branislav Bošanský, and Bo An. “Computing quantal stackelberg equilibrium in extensive-form games.” AAAI 2021

Example: contract design

$$\min_{x \in \mathcal{X}, y} \quad f(x, y) \qquad \text{(upper level)}$$

$$\text{s.t.} \quad y \in \arg\min_{y' \in \mathcal{Y}} \; g(x, y') \qquad \text{(lower level)}$$

Upper: Find leader's optimal contract

Lower: follower always adopts best response

Follower takes action b which generates an outcome $o \sim p_b$

Each action b requires some effort and thus incurs a cost $c(b)$

Leader incentivizes follower with an outcome-dependent payment $S(o)$

Special case of Stackelberg game with structured rewards

$$R(S, b) = \mathbb{E}_{o \sim p_b}[R(o) - S(o)], \quad r(S, b) = \mathbb{E}_{o \sim p_b}[S(o)] - c(b)$$

Laffont, Jean-Jacques, and David Martimort. "The theory of incentives: the principal-agent model." In *The theory of incentives*. Princeton university press, 2009.
Ho, Chien-Ju, Aleksandrs Slivkins, and Jennifer Wortman Vaughan. "Adaptive contract design for crowdsourcing markets: Bandit algorithms for repeated principal-agent problems." EC 2014

Example: performative prediction

$$\begin{aligned} \min_{x \in \mathcal{X}, y} \quad & f(x, y) && \text{(upper level)} \\ \text{s.t.} \quad & y \in \arg\min_{y' \in \mathcal{Y}} \; g(x, y') && \text{(lower level)} \end{aligned}$$

Upper: Find leader's optimal decision

Lower: sample $z \sim \mathcal{D}(x)$

x: parameter of a ML model

$\ell(x, z)$: loss of model x on data z

$$\min_x \; L(x) = \mathbb{E}_{z \sim \mathcal{D}(x)} \big[\overbrace{\ell(x, z)} \big]$$

$S(x) = \mathcal{D}(x)$: strategically manipulated distribution

Example of $\mathcal{D}(x)$: $\quad z = Ax + \zeta \iff S(x) = \arg\min_y \mathrm{KL}\big(y \,\|\, \mathcal{N}(Ax, \sigma^2 I)\big)$

Perdomo, Juan, Tijana Zrnic, Celestine Mendler-Dünner, and Moritz Hardt. “Performative prediction.” ICML 2020
Drusvyatskiy, Dmitriy, and Lin Xiao. "Stochastic optimization with decision-dependent distributions." *Mathematics of Operations Research* 2023
Miller, John P., Juan C. Perdomo, and Tijana Zrnic. “Outside the echo chamber: Optimizing the performative risk.” ICML 2021

Example: multi-agent performative game

$x = (x^1, \ldots, x^n)$: decision variables of n agents

Upper: Find leader's **equilibrium** policy

Lower: sample $z \sim \mathcal{D}(x)$

$$L^i(x^1, \ldots, x^n) = \mathbb{E}_{(z^1,\ldots,z^n)\sim\mathcal{D}(x^1,\ldots,x^n)}[\ell(x^i, z^i)]$$

$\mathcal{D}(x)$: decision-dependent observations

z^i dependens on actions of all n agents

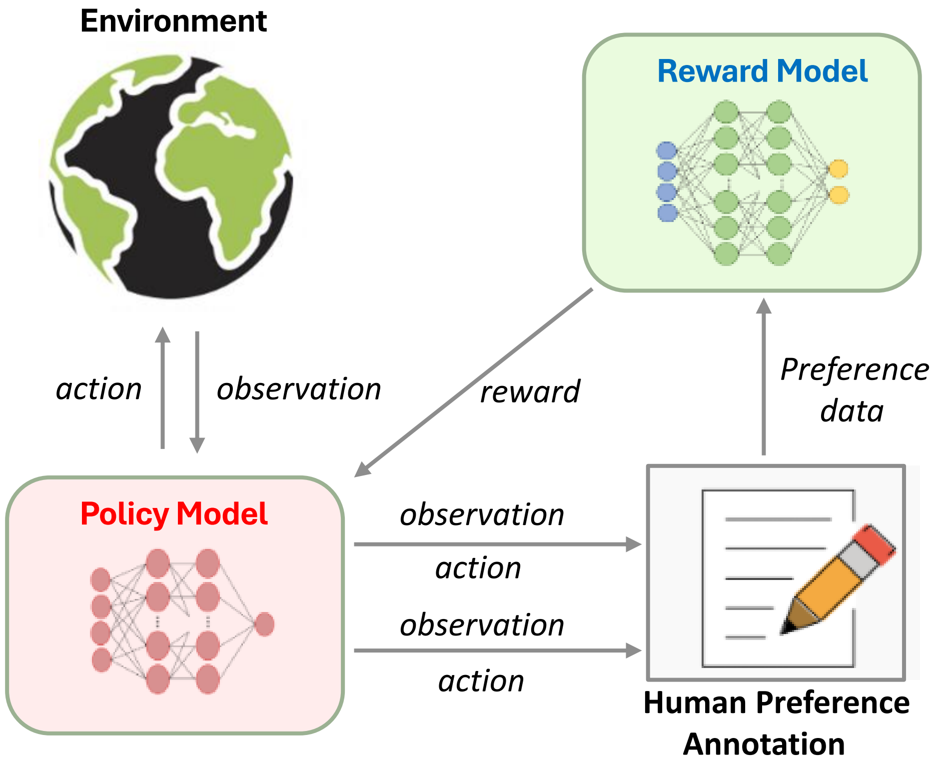
- Example of $\mathcal{D}(x)$: $\quad z^i = A^i x^i + \sum_{j\neq i} B^{ij} x^j + \zeta^i$
- Solution concept: Nash equilibrium x^*

$$x^{i,*} \in \arg\min_{x^i} L^i(x^i, x^{-i,*})$$

Narang, Adhyyan, Evan Faulkner, Dmitriy Drusvyatskiy, Maryam Fazel, and Lillian J. Ratliff. "Multiplayer performative prediction: Learning in decision-dependent games." *JMLR 2023*

Piliouras, Georgios, and Fang-Yi Yu. "Multi-agent performative prediction: From global stability and optimality to chaos." EC 2023

Example: RL with human feedbacks



Upper level (reward learning)

1. Collect bunch of trajectory pairs with policy
2. Humans label the preferred one; Reward predictor predicts the trajectory label
3. Train reward predictor with MLE loss given human labels (Bradley-Terry model)

Lower level (policy learning)

4. Train policy to increase the predicted reward

Data: two trajectories $\tau_1, \tau_2 \sim \pi_y$, label z given by human

$$\max_{x,y} \quad \mathbb{E}_{\tau_1,\tau_2\sim\pi_y,z}\Big[z \cdot \log \mathbb{P}_x(\tau_1 \succ \tau_2) + (1-z)\cdot \mathbb{P}_x(\tau_1 \prec \tau_2)\Big] \qquad r_x \text{ is MLE}$$

$$\text{s.t.} \quad y \in \arg\max \quad g(x, y') = \mathbb{E}_{\tau\sim\pi_{y'}}\big[\textstyle\sum_{h=1}^{H} r_x(s_h, a_h)\big] \qquad \pi_y \text{ is optimal wrt } r_x$$

Ouyang, Long, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang et al. "Training language models to follow instructions with human feedback." Neurips 2022.

Example: RLHF / reward design / inverse RL

$$
\begin{aligned}
\min_{x \in \mathcal{X}, y} \quad & f(x, y) & \text{(upper level)} \\
\text{s.t.} \quad & y \in \arg\min_{y' \in \mathcal{Y}} \; g(x, y') & \text{(lower level)}
\end{aligned}
$$

Upper: Find leader's optimal reward param.

Lower: follower always adopts optimal policy

Leader chooses a reward r_x Follower chooses a policy π_y

Leader's goal: find a reward r^* such that

Trajectory τ generated by π^* explains observed data

$$f(x, y) = \mathbb{E}_{\tau \sim \pi_y, \bar{\tau} \sim \text{Data}}[\text{Dist}(\tau, \bar{\tau})]$$

$$g(x, y) = \mathbb{E}_{\tau \sim \pi_y}[\textstyle\sum_{h=1}^{H} r_x(s_h, a_h)]$$

Often π_y does not enter f or g directly, rather indirectly through τ

$$\pi_y \longrightarrow \tau = \{s_h, a_h\}_{h=1}^{H} \longrightarrow g(x, y) \;\&\; f(x, y)$$

Agenda: Recent optimization and learning results

Upper-level variable x = policy of leader

$$\max_{x \in \mathcal{X}, y} \quad f(x, y) \qquad \text{(upper level)}$$

$$\text{s.t.} \quad y \in \arg\max_{y' \in \mathcal{Y}} \; g(x, y') \qquad \text{(lower level)}$$

Lower-level variable y = policy of follower

Upper: Find leader's optimal policy

Lower: follower always adopts best response

Optimization: When model is known, how to compute x^* and $S(x^*)$?

Learning (Statistics): How to learn $\left(x^*, S(x^*)\right)$ from data efficiently? What data? How many data points needed?

Optimization in bilevel RL – main takeaways

$$\begin{aligned} \min_{x \in \mathcal{X}, y} \quad & f(x, y) && \text{(upper level)} \\ \text{s.t.} \quad & y \in \arg\min_{y' \in \mathcal{Y}} \; g(x, y') && \text{(lower level)} \end{aligned}$$

- Lower problem is convex optimization (y is not a policy), rather easy to solve using standard optimization tools
- Lower problem is RL (y is a policy π_y), need to modify bilevel optimization tools (e.g., penalty method)

Lower problem is not RL – Stackelberg matrix game

Matrix game with action spaces $\mathcal{A} = \{a_1, \ldots, a_m\}, \mathcal{B} = \{b_1, \ldots, b_n\}$

Reward functions $R(a, b), r(a, b)$ Policies $x \in \mathcal{X} = \Delta_m, y \in \mathcal{Y} = \Delta_n$

$$f(x, y) = \mathbb{E}_{a \sim x, b \sim y}[R(a, b)], \quad g(x, y) = \mathbb{E}_{a \sim x, b \sim y}[r(a, b)]$$

Solve by LP – find the optimal x for each $b \in \mathcal{B}$

$$\mathcal{X}(b) = \{x \in \Delta_m : y^*(x) = \delta_b\} = \{x : \mathbb{E}_{a \sim x}[r(a, b)] \geq \mathbb{E}_{a \sim x}[r(a, b')], \forall b' \in \mathcal{B}\}$$

$$x_b^* = \arg \max_{x \in \mathcal{X}(b)} \mathbb{E}_{a \sim x}[R(a, b)], \quad \forall b \in \mathcal{B}$$

⬇

Enumerate all $b \in \mathcal{B}$ to get solution : $b^* \in \arg\max_b \mathbb{E}_{a \sim x_b^*}[R(a, b)]$

Conitzer, Vincent, and Tuomas Sandholm. "Computing the optimal strategy to commit to." In *ACM conference on Electronic commerce*, pp. 82-90. 2006.

Quantal Stackelberg matrix game

Matrix game with action spaces $\mathcal{A} = \{a_1, \ldots, a_m\}, \mathcal{B} = \{b_1, \ldots, b_n\}$

Reward functions $R(a, b), r(a, b)$ Policies $x \in \mathcal{X} = \Delta_m, y \in \mathcal{Y} = \Delta_n$

$f(x, y) = \mathbb{E}_{a \sim x, b \sim y}[R(a, b)], \quad g(x, y) = \mathbb{E}_{a \sim x, b \sim y}[r(a, b)] + \eta^{-1} \cdot \mathcal{H}(y)$ $\Leftarrow$ entropy

Quantal response $S(x)(b) = Z(x)^{-1} \cdot \exp(\eta \cdot r(x, b))$

Plug in closed-form of $S(x)$ — reduce to nonlinear optimization :

$$\max_{x \in \mathcal{X}} F(x) = \mathbb{E}_{a \sim x, b \sim S(x)}[R(a, b)]$$

Can be solved by first-order optimization when $\mathcal{B}$ finite

Closed-from of $S(x)$ + policy gradient

Quantal response $S(x)(b) = Z(x)^{-1} \cdot \exp(\eta \cdot r(x,b))$

$$\max_{x \in \mathcal{X}} \; F(x) = \mathbb{E}_{a \sim x, b \sim S(x)}[R(a,b)]$$

$$S(x) \Rightarrow P_x$$

Policy gradient trick: $\nabla_x \left(\mathbb{E}_{b \sim \mathbb{P}_x}[h(b)]\right) = \mathbb{E}_{b \sim \mathbb{P}_x}[h(b) \cdot \nabla_x \log \mathbb{P}_x(b)]$

➡ Policy gradient trick: $\nabla_x \left(\mathbb{E}_{b \sim \mathbb{P}_x}[h(b)]\right) = \mathbb{E}_{b \sim \mathbb{P}_x}[h(b) \cdot \nabla_x \log \mathbb{P}_x(b)]$

Performative prediction can also be solved by first-order optimization:

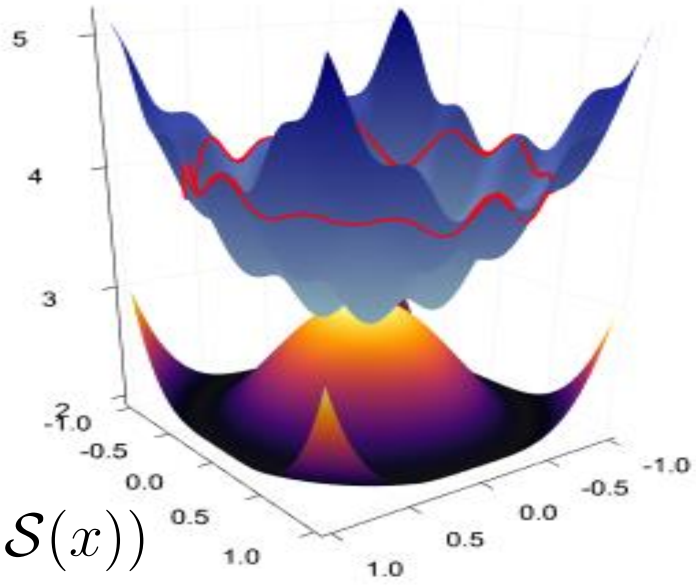
$$\nabla_x L(x) = \nabla_x \mathbb{E}_{z \sim D(x)}[\ell(x,z)] = \mathbb{E}_{z \sim D(x)}[\nabla_x \ell(x,z) + \ell(x,z) \cdot \nabla_x \log D(x)]$$

Drusvyatskiy, Dmitriy, and Lin Xiao. "Stochastic optimization with decision-dependent distributions." *Mathematics of Operations Research* 2023
Miller, John P., Juan C. Perdomo, and Tijana Zrnic. “Outside the echo chamber: Optimizing the performative risk.” ICML 2021

Lower problem is RL – reward design

$$\min_{x,y} f(x,y) \qquad \text{(upper)}$$
$$\text{s.t. } \pi_y \in S(x) = \arg\max_{y'} g(x,y') = \mathbb{E}_{\tau\sim\pi_y}\left[\sum_{h=1}^{H} r_x(s_h,a_h)\right] \qquad \text{(lower)}$$

Challenge of RL: optimal policy nonunique
Lower problem not convex



Typically lower-level function is **strongly convex** so that

$$\min_x f(x,\mathcal{S}(x)) \qquad \frac{\partial f(x,\mathcal{S}(x))}{\partial x} = \nabla_x f(x,\mathcal{S}(x)) + \nabla\mathcal{S}(x)\nabla_y f(x,\mathcal{S}(x))$$

Given by Implicit function theorem

Difficult to apply existing bilevel optimization algorithms directly

Ensure unique lower-level solution – regularization

$$g\left(x,y\right)=\mathbb{E}_{\tau\sim\pi_y}\left[\sum_{h=1}^{H}\left\{r_x\left(s_h,a_h\right)+\eta\cdot\mathcal{H}(\pi_y(\cdot\,|\,s_h))\right\}\right]\qquad(\eta>0)$$

$$S(x)=\underset{\pi_y}{\operatorname{argmax}}\,g(x,y)\quad\text{unique}\qquad\qquad \mathcal{H}(p)=\sum_{a\in\mathcal{A}}-p(a)\log p(a)$$

Recall: two general recipes for bilevel optimization

$$\min_{x \in \mathcal{X},\, y} \quad f(x, y)$$
$$\text{s.t.} \quad y \in S(x) := \arg\min_{y' \in \mathcal{Y}} \; g(x, y')$$

$$\min_{x \in \mathcal{X}} \quad F(x)$$
$$\text{with} \quad F(x) := \min_{y \in S(x)} \; f(x, y)$$

Nested optimization
first over *y* and then over *x*

$$\min_{x \in \mathcal{X},\, y \in \mathcal{Y}} \quad f(x, y)$$
$$\text{s.t.} \quad \text{sufficient conditions for } y \in S(x)$$

Constrained optimization
jointly over *x* and *y*

Implicit gradient

How to compute implicit gradient?

Penalty method

What penalty function?
How is regularized problem related to original problem?

Implicit gradient for bilevel RL

Assume leader's objective depends on x and π_y via a bivariate function U

$$f(x,y) = \mathbb{E}_{\tau\sim\pi_y}\left[\sum_{h=1}^{H} U(s_h, a_h; x)\right] \qquad S(x) = \arg\max_y g(x,y)$$

Apply policy gradient theorem to $\nabla_x F(x) = \nabla_x f(x, S(x))$

$$\begin{aligned}\nabla_x f(x, S(x)) = \mathbb{E}_{\tau\sim\pi_{S(x)}}&\left[\sum_{h=1}^{H} \nabla_x U(s_h, a_h; x)\right] \\ &+ \mathbb{E}_{\pi_x\sim\pi_{S(x)}}\left[\sum_{h=1}^{H} U(s_h, a_h; x)\cdot \nabla_x \log \pi_{S(x)}(a_h|s_h)\right]\end{aligned}$$

Second term contains implicit gradient (apply chain rule):

$$\nabla_x \log \pi_{S(x)}(a|s) = [\underbrace{\nabla_x S(x)}_{\text{Implicit gradient}}](\nabla_y \log \pi_y(a|s))\Big|_{y=S(x)}$$

Chakraborty, Souradip, Amrit Singh Bedi, Alec Koppel, Dinesh Manocha, Huazheng Wang, Furong Huang, and Mengdi Wang. "Aligning agent policy with externalities: Reward design via bilevel rl." *arXiv preprint arXiv:2308.02585* (2023).

Compute implicit gradient by differentiate lower level

Apply policy gradient theorem to $\nabla_x F(x) = \nabla_x f(x, S(x))$

$$\nabla_x f(x, S(x)) = \mathbb{E}_{\tau \sim \pi_{S(x)}} \left[\sum_{h=1}^{H} \nabla_x U(s_h, a_h; x) \right] + \mathbb{E}_{\pi_x \sim \pi_{S(x)}} \left[\sum_{h=1}^{H} U(s_h, a_h; x) \cdot \nabla_x \log \pi_{S(x)}(a_h | s_h) \right]$$

Second term contains implicit gradient (apply chain rule):

$$\nabla_x \log \pi_{S(x)}(a|s) = [\nabla_x S(x)] (\nabla_y \log \pi_y(a|s)) \Big|_{y=S(x)}$$

How to compute $\nabla_x S(x)$? Again, differentiate lower level optimality condition:

$$\nabla_y g(x, S(x)) = 0 \quad \forall x.$$

$$\Longrightarrow \nabla^2_{xy} g(x, S(x)) + [\nabla_x S(x)] \nabla^2_{yy} g(x, S(x)) = 0$$

Chakraborty, Souradip, Amrit Singh Bedi, Alec Koppel, Dinesh Manocha, Huazheng Wang, Furong Huang, and Mengdi Wang. "Aligning agent policy with externalities: Reward design via bilevel rl." *arXiv preprint arXiv:2308.02585* (2023).

Implicit gradient formula

Apply policy gradient theorem to $\nabla_x F(x) = \nabla_x f(x, S(x))$

$$\nabla_x f(x, S(x)) = \mathbb{E}_{\tau \sim \pi_{S(x)}} \left[\sum_{h=1}^{H} \nabla_x U(s_h, a_h; x) \right] + \mathbb{E}_{\pi_x \sim \pi_{S(x)}} \left[\sum_{h=1}^{H} U(s_h, a_h; x) \cdot \nabla_x \log \pi_{S(x)}(a_h | s_h) \right]$$

Second term contains implicit gradient (apply chain rule):

$$\nabla_x \log \pi_{S(x)}(a|s) = [\nabla_x S(x)] (\nabla_y \log \pi_y(a|s)) \big|_{y=S(x)}$$

Implicit gradient formula:

$$\nabla_x \log \pi_{S(x)}(a|s) = -[\nabla^2_{xy} g(x, S(x))][\nabla^2_{yy} g(x, S(x))]^{-1} [\nabla_y \log \pi_y(a|s)] \big|_{y=S(x)}$$

Note: require policy Hessian $\nabla^2_{yy} g(x, y)$

Chakraborty, Souradip, Amrit Singh Bedi, Alec Koppel, Dinesh Manocha, Huazheng Wang, Furong Huang, and Mengdi Wang. "Aligning agent policy with externalities: Reward design via bilevel rl." *arXiv preprint arXiv:2308.02585* (2023).

Recall: penalty method for bilevel optimization

$$\min_{x\in\mathcal{X},y} \quad f(x,y)$$
$$\text{s.t.} \quad y \in S(x) = \arg\max_{y} g(x,y)$$

Note: lower problem changed to max

$$\min_{x\in\mathcal{X},\, y\in\mathcal{Y}} \quad f(x,y)$$
$$\text{s.t.} \quad \text{sufficient condition}: \; p(x,y) \le 0$$

$$\mathcal{BP}_{\gamma p}: \; \min_{x,y} f(x,y) + \gamma p(x,y)$$

Example in part II: $p(x,y) = \max_{y'} g(x,y') - g(x,y)$

Question: How to define $p(x,y)$ for π_y?

Is $\mathcal{BP}_{\gamma p}$ equivalent to $\mathcal{BP}_{\epsilon}$ for some ϵ?

$$\mathcal{BP}_{\epsilon}: \; \min_{x,y} f(x,y) \quad \text{s.t. } p(x,y) \le \epsilon$$

Penalty function I – value penalty

$$g(x,y) = \mathbb{E}_{\tau\sim\pi_y}\left[\sum_{h=1}^{H} r_x(s_h,a_h)\right]$$

$$p(x,y) = \underbrace{\max_{y'} g(x,y')}_{\text{optimal policy wrt } r_x} - g(x,y)$$

Note: optimal value $\max_{y'} g(x,y') = g(x,S(x))$ is unique

$S(x)$ might be non-unique

$$\mathcal{BP}_{\gamma p}: \ \min_{x,y} f(x,y) + \gamma p(x,y)$$

$$\mathcal{BP}_{\epsilon}: \ \min_{x,y} f(x,y) \quad \text{s.t. } p(x,y) \le \epsilon$$

Theorem (solution relation)

Assume $f(x,\cdot)$ is L-Lipschitz in y. For any $\epsilon > 0$, choosing $\lambda = \mathcal{O}(L/\epsilon)$, any local/global solution to $\mathcal{BP}_{\gamma,p}$ is a local/global solution to $\mathcal{BP}_{\epsilon}$.

No explicit regularization required. Uniqueness not necessary.

Shen, Han, Zhuoran Yang, and Tianyi Chen. "Principled Penalty-based Methods for Bilevel Reinforcement Learning and RLHF." *arXiv preprint arXiv:2402.06886* (2024).

Penalty function I – value penalty

$*$ solve a RL problem $\to S(x)$

$*$ policy evaluation with vector reward $\nabla_x r_x$

Gradient of $p(x,y)$

$$\nabla_x p(x,y) = -\nabla_x g(x,y) + \nabla_x g(x,y)\big|_{y=S(x)}$$
$$= \mathbb{E}_{\tau\sim S(x)}\left[\sum_{h=1}^{H} \nabla_x r_x(s_h,a_h)\right] - \mathbb{E}_{\tau\sim\pi_y}\left[\sum_{h=1}^{H} \nabla_x r_x(s_h,a_h)\right]$$
$$\nabla_y p(x,y) = -\nabla_y g(x,y) = \text{policy gradient}$$

$$\mathcal{BP}_{\gamma p}: \ \min_{x,y} f(x,y) + \gamma p(x,y)$$
$$\Updownarrow$$
$$\mathcal{BP}_{\epsilon}: \ \min_{x,y} f(x,y) \quad \text{s.t. } p(x,y) \le \epsilon$$

- optimality of (x_λ, y_λ) in $\mathcal{BP}_\lambda$
- monotonicity at (x_λ, y_λ) :

$$\langle \nabla_y p(x_\lambda,y_\lambda), y - y_\lambda\rangle \ge C\cdot p(x_\lambda,y_\lambda), \quad \forall y$$

Penalty function 2 – Bellman penalty

$$g(x, y) = \mathbb{E}_{\tau \sim \pi_y}\left[\sum_{h=1}^{H}\left\{r_x(s_h, a_h) + \eta \cdot \mathcal{H}(\pi_y(\cdot \mid s_h))\right\}\right] \qquad (\eta > 0)$$

Optimal policy $\pi^* = S(x)$ characterized by optimal Q function $Q_x^*(s, a)$:

$$\pi^* = \arg\max_{\pi_y} \underbrace{\mathbb{E}_{s\sim\rho}\left[\sum_{a\in\mathcal{A}}\pi_y(a|s) \cdot Q_x^*(s, a)\right] + \eta \cdot \mathcal{H}(\pi_y(\cdot \mid s))}_{= h(x,y)}$$

$$p(x, y) = \max_{y'} h(x, y') - h(x, y)$$

Follow from strong convexity of h

Theorem (solution relation)

Assume $f(x, \cdot)$ is L-Lipschitz in y. For any $\epsilon > 0$, choosing $\gamma = \mathcal{O}(\sqrt{L\eta^{-1}\epsilon^{-1}})$, any local/global solution to $\mathcal{BP}_{\gamma,p}$ is a local/global solution to $\mathcal{BP}_\epsilon$.

Penalty function 2 – Bellman penalty

$$p(x,y) = \max_{y'} h(x,y') - h(x,y)$$

$$h(x,y) = \mathbb{E}_{s\sim\rho}\left[\sum_{a\in\mathcal{A}} \pi_y(a|s) \cdot Q_x^*(s,a)\right] + \eta \cdot \mathcal{H}(\pi_y(\cdot \mid s))]$$

Gradient of $p(x,y)$

$$\begin{aligned}\nabla_x p(x,y) &= -\nabla_x h(x,y) + \nabla_x h(x,y)\big|_{y=S(x)} \\ &= \mathbb{E}_{(s,a)\sim S(x)}[\nabla_x Q_x^*(s_h,a_h)] - \mathbb{E}_{(s,a)\sim\pi_y}[\nabla_x Q_x^*(s_h,a_h)]\end{aligned}$$

$$\nabla_y p(x,y) = -\nabla_y h(x,y) = \text{policy gradient}$$

Implement penalty method for bilevel RL

PBRL algorithm

At current iterate (x_k, y_k)

- Solve the MDP with reward r_{x_k} and get $\pi^k \approx S(x_k)$ or $Q^k \approx Q^*_{x^k}$
- Use π^k (version 1) or Q^k (version 2) to approximate $\nabla p(x^k, y^k)$
- Get gradient $\nabla f(x^k, y^k) + \lambda \nabla p(x^k, y^k)$
- Update (x^{k+1}, y^{k+1}) via (policy) gradient methods

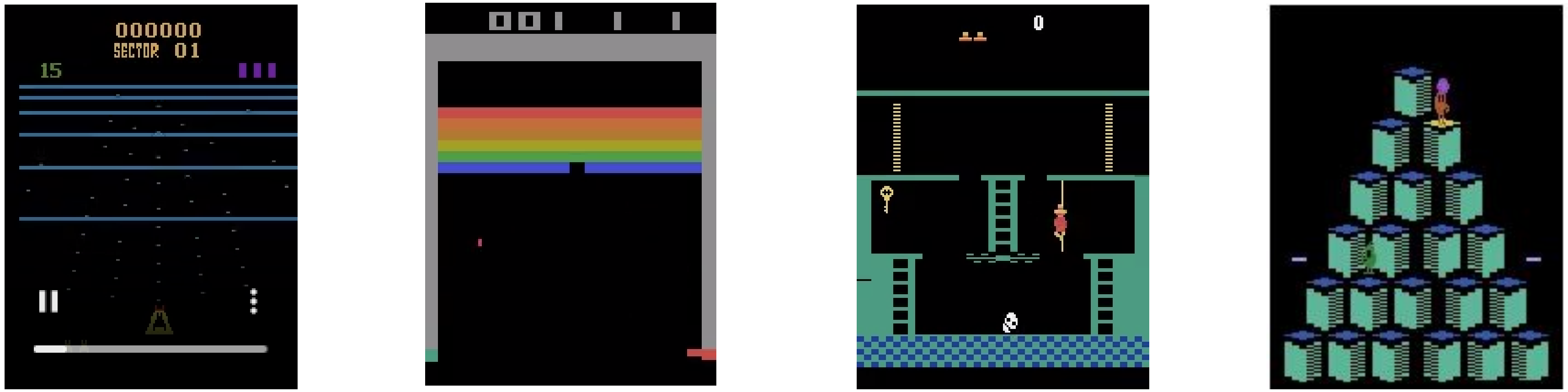
First order updates – do not require policy Hessian

Inner MDP solving subroutine – linear convergence using policy mirror descent

Outer loop Converge to a stationary point at sublinear rate

Zhan, Wenhao, et al. "Policy mirror descent for regularized reinforcement learning: A generalized framework with linear convergence." SIAM Journal on Optimization, 2023.

Numerical experiments: RLHF on Atari games



The OpenAI gymnasium library includes 59 games

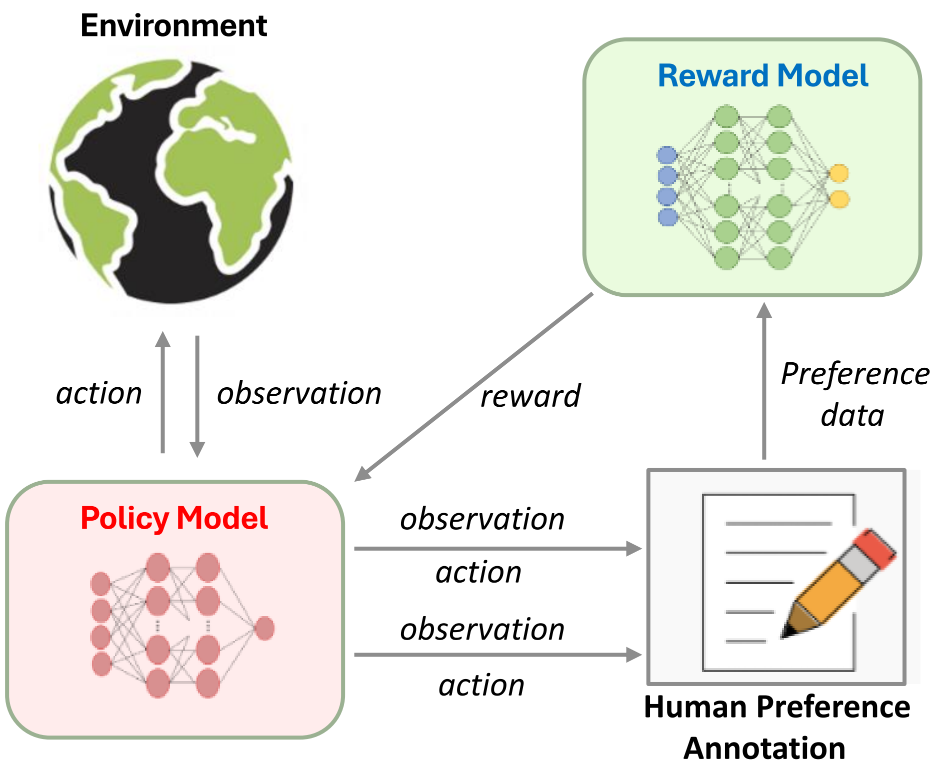
The Arcade learning environment is commonly used to test RL algorithms

- **Goal**: finish the games with high score
- **Input**: sequence of images
- **Output**: actions to play

Numerical experiments: RLHF on Atari games

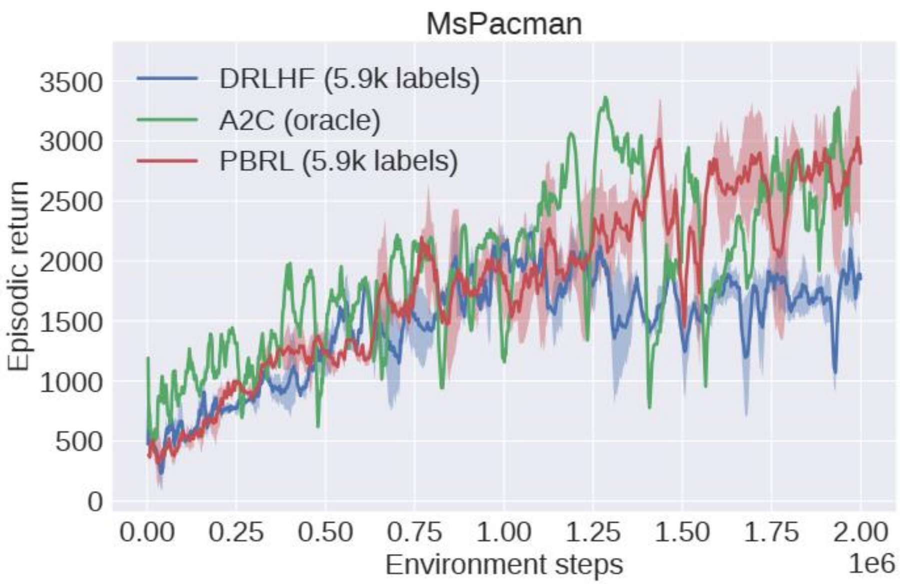
We implement

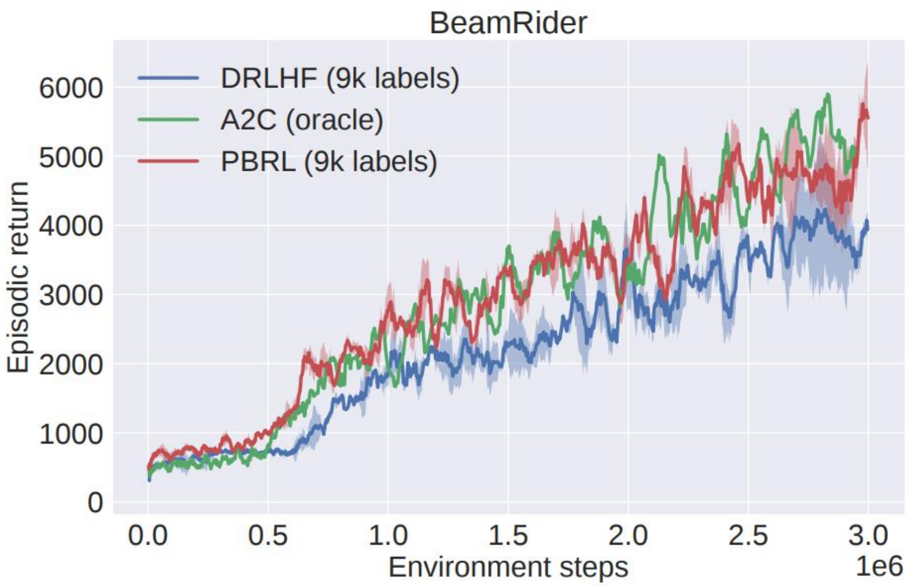
- **Baseline**: original RLHF algorithm (DRLHF)
- **Ours**: PBRL algorithm
- **Oracle**: A2C with access to the ground truth reward

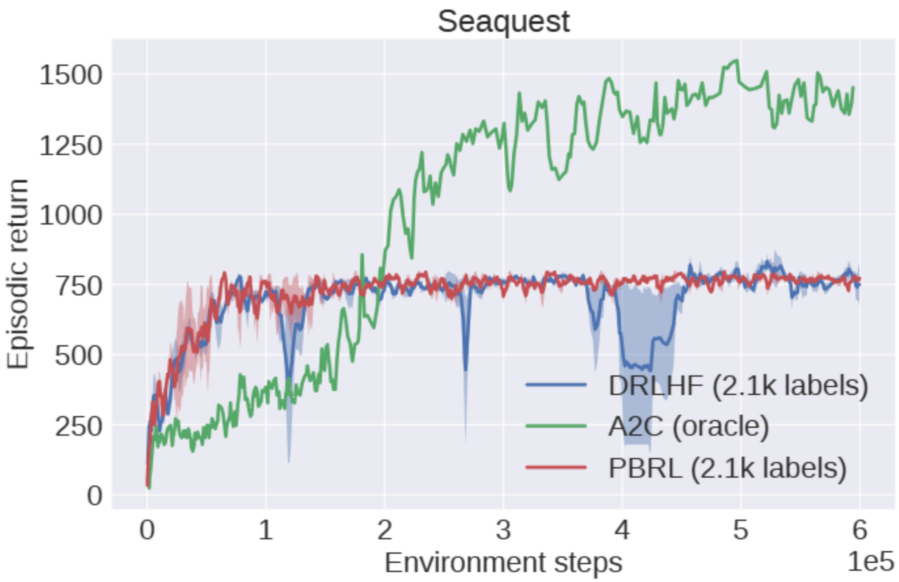


We follow the original RLHF paper and use the game score as the ground truth reward and generate human feedback

Numerical experiments: RLHF on Atari games



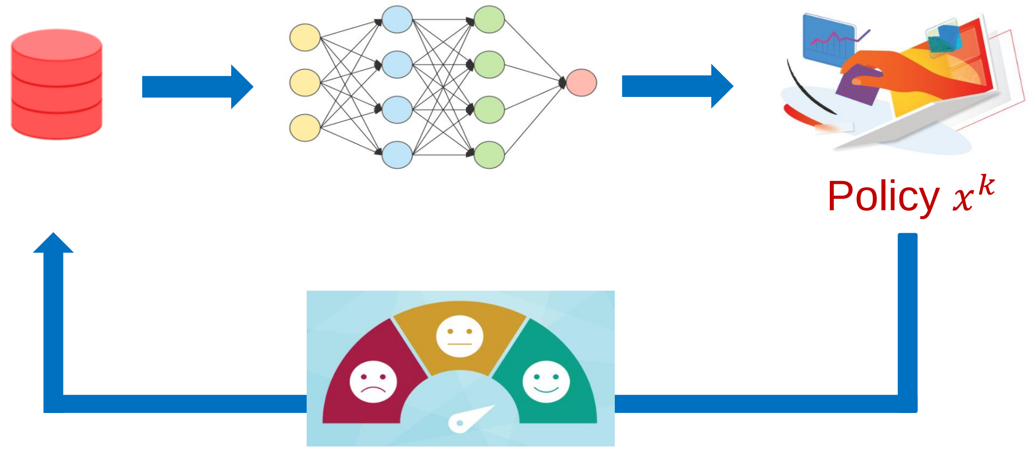




Online learning in bilevel RL – setting

$$\min_{x\in\mathcal{X},\, y} \quad f(x,y)$$
$$\text{s.t.} \quad y \in S(x) := \arg\min_{y'\in\mathcal{Y}} \; g(x,y')$$

$$\min_{x} \quad F(x)$$
$$\text{with} \quad F(x) = f(x, S(x))$$



Omniscient follower – always play $y^k = S(x^k)$
Feedback data: Leader's observations

Leader's learning problem:
learns x^* from data by interacting with follower

$$\text{regret}_K = \sum_{k=1}^{K}[F(x^*) - F(x^k)]$$

Unknowns: $f(x,y)$ and $S(x)$

Main challenge: estimate $S(x)$
Assumptions on data & model?

Zhao, Geng, Banghua Zhu, Jiantao Jiao, and Michael Jordan. "Online learning in stackelberg games with an omniscient follower." ICML 2023

Online learning in Stackelberg game

Matrix game with action spaces $\mathcal{A} = \{a_1, \ldots, a_m\}, \mathcal{B} = \{b_1, \ldots, b_n\}$

Reward functions $R(a, b), r(a, b)$ Policies $x \in \mathcal{X} = \Delta_m, y \in \mathcal{Y} = \Delta_n$

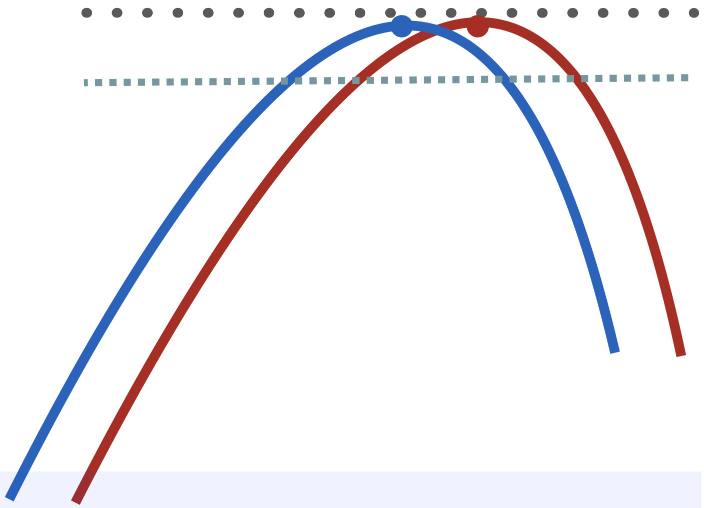
$$f(x, y) = \mathbb{E}_{a\sim x, b\sim y}[R(a, b)], \quad g(x, y) = \mathbb{E}_{a\sim x, b\sim y}[r(a, b)]$$

Best response $S(x) = \delta\{\arg\max_{b\in\mathcal{B}} \mathbb{E}_{a\sim x}[r(a, b)]\}$

Data assumption: Learner controls both players

Toy example: online learning in Stackelberg game

Data assumption: Learner controls both players, observes bandit feedbacks of (R, r)



Algorithm:

- Try all $(a, b) \in \mathcal{A} \times \mathcal{B}$ for N times, estiamte $\widehat{R}$ and $\widehat{r}$
- Return $\widehat{x} = \arg\max_x \widehat{f}(x, \widehat{S}(x))$

A pessimistic result:

- No matter how accurate $\widehat{R}$ and $\widehat{r}$ are, $\widehat{x}$ can be worse than x^* by a constant
- $\widehat{S}(x) = \delta\{\arg\max_{b \in \mathcal{B}} \mathbb{E}_{a \sim x}[\widehat{r}(a, b)]\}$ is sensitive to estimation error

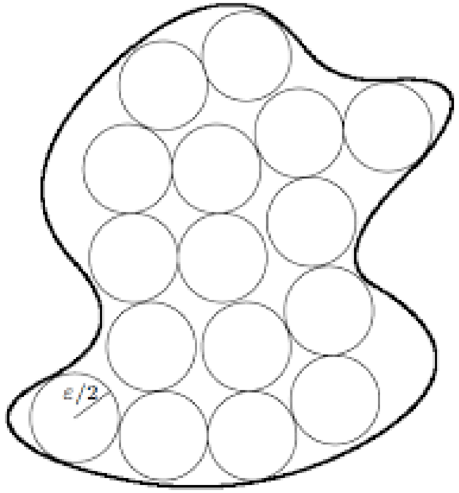
$\Longrightarrow$ Best respose $S(x)$ cannot be estimated by estimating r

Bai, Yu, Chi Jin, Huan Wang, and Caiming Xiong. "Sample-efficient learning of Stackelberg equilibria in general-sum games." NeurIPS 2021

Method I – forget about estimating $S(x)$

Data assumption: Learner controls leader, observes bandit feedbacks of $F(x)$

Leader play $a \sim x$, follower plays $b \sim S(x)$, receive $R(a, b)$



Algorithm:

- discretize $\mathcal{X}$ by $\mathcal{E}_\epsilon(\mathcal{X})$
- treat each $x \in \mathcal{E}_\epsilon(\mathcal{X})$ as an arm and run UCB algorithm

Theorem (Zhu et al)

For contract design with m possible outcomes, the regret is $\widetilde{\mathcal{O}}(K^{1-1/(m+2)})$.
That is, to find ϵ-optimal solution, we need $\mathcal{O}((1/\epsilon)^{m+2})$ samples.

Zhu, Banghua, Stephen Bates, Zhuoran Yang, Yixin Wang, Jiantao Jiao, and Michael I. Jordan. "The sample complexity of online contract design." *arXiv preprint arXiv:2211.05732* (2022).

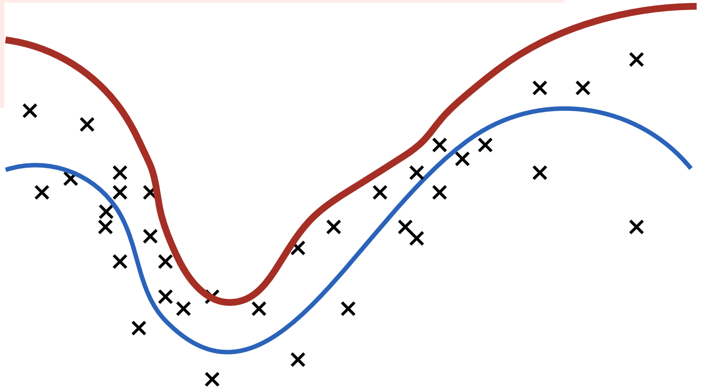
Method II – estimate $S(x)$ via quantal response

Data assumption: Learner controls leader, observes bandit feedbacks and follower's action

Leader play $a \sim x$, follower plays $b \sim S(x)$, receive $R(a, b)$

$$f(x, y) = \mathbb{E}_{a\sim x, b\sim y}[R(a, b)], \quad g(x, y) = \mathbb{E}_{a\sim x, b\sim y}[r(a, b)] + \eta^{-1} \cdot \mathcal{H}(y) \Leftarrow \text{entropy}$$

Quantal response $S(x)(b) = Z(x)^{-1} \cdot \exp(\eta \cdot r(x, b))$



Algorithm: estimate r via MLE + UCB bonus

- Estimate r from MLE $\widehat{r} = \arg\max \log \mathbb{P}_r(b \,|\, x)$
- Estimate R by mean estimation $\widehat{R}$
- UCB planning: $\max_x \langle \widehat{R} + \Gamma_1, x \times S_{\widehat{r}}(x) \rangle + \Gamma_2(x)$

Theorem

$$\text{regret}_{\mathrm{K}} = \widetilde{\mathcal{O}}(\sqrt{T})$$

Chen, Siyu, Mengdi Wang, and Zhuoran Yang. "Actions Speak What You Want: Provably Sample-Efficient Reinforcement Learning of the Quantal Stackelberg Equilibrium from Strategic Feedbacks." *arXiv preprint arXiv:2307.14085* (2023).

Summary

- Bilevel RL – Leader-follower structure + RL
- Examples – Stackelberg game, RLHF / reward design
- Optimization aspect of bilevel RL
  - Implicit gradient
  - Penalty method
- Learning aspect of bilevel RL
  - UCB + discretization
  - Quantal response + MLE + UCB

Outline

- ❑ Part I - Introduction and background
- ❑ Part II – Bilevel optimization fundamentals
- ❑ Part III – Bilevel applications to reinforcement learning
- ❑ **Part IV – Multi-objective learning beyond bilevel optimization (65 mins)**
- ❑ Part V - Conclusions and open directions

Tutorial Part IV:
Multi-objective Learning Beyond Bilevel

Lisha Chen

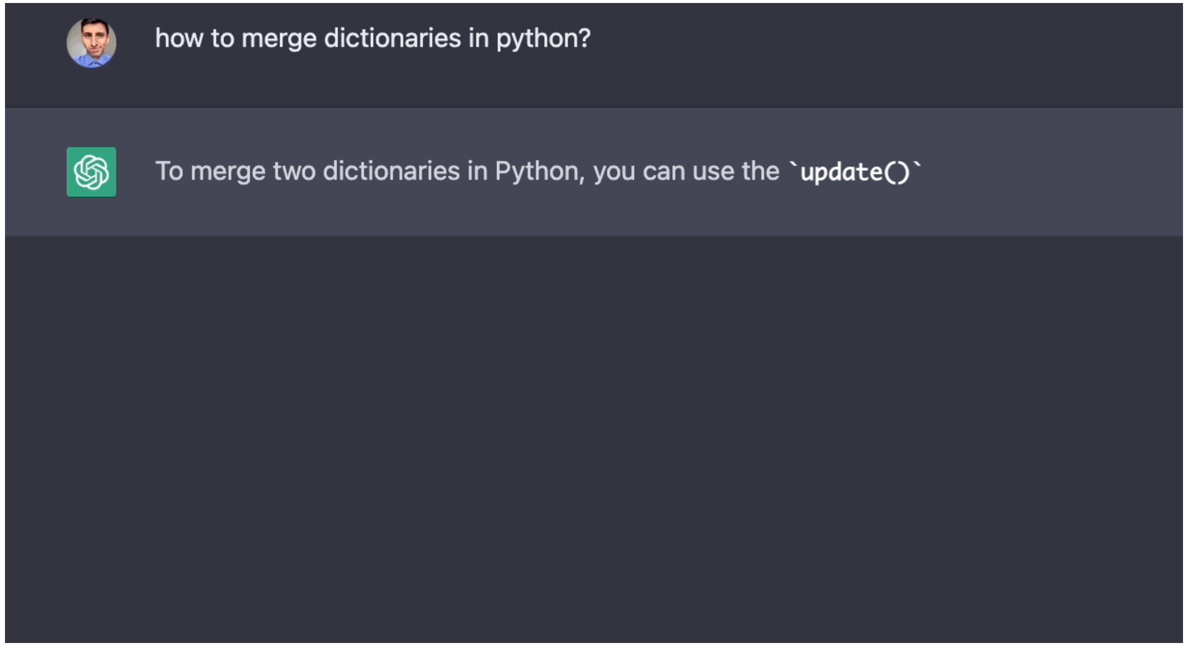
Rensselaer Polytechnic Institute

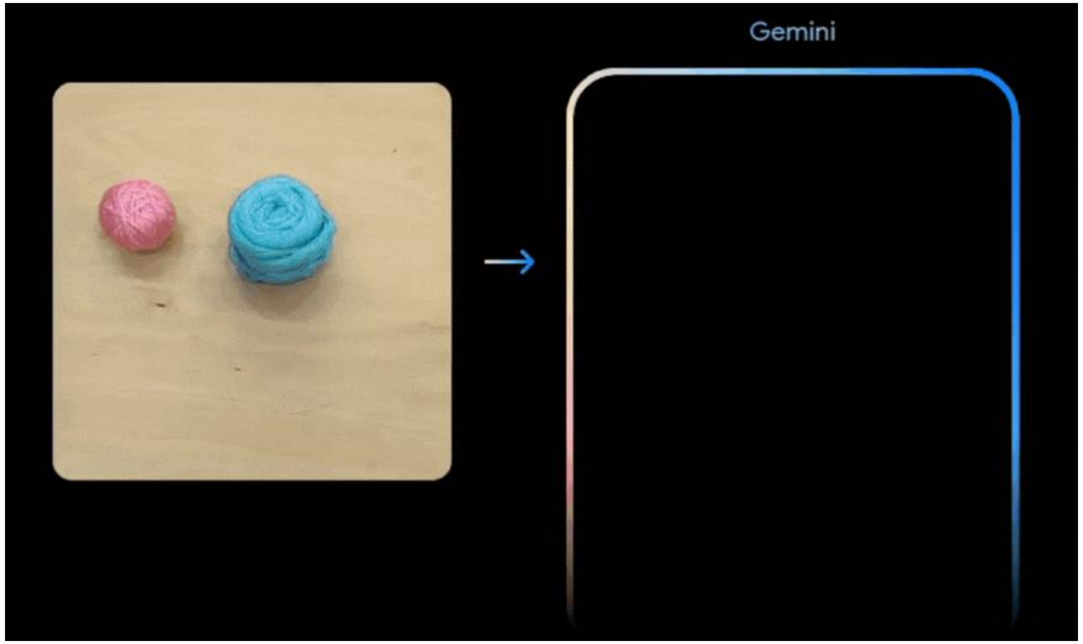
February 20, 2024

Outline

- Introduction and motivation
  - Motivation
  - Solution concepts and measures of optimality
- Multi-gradient based methods
  - (deterministic) MGDA, CAGrad, other methods
  - (stochastic) SMG, MoCo, MoDo
- Theory of multi-objective learning
  - Optimization
  - Generalization
- Application of multi-objective learning

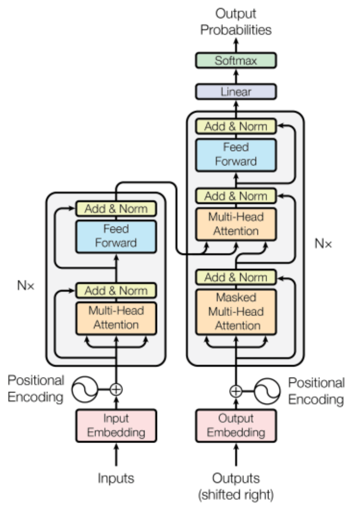
Success of AI in the new era

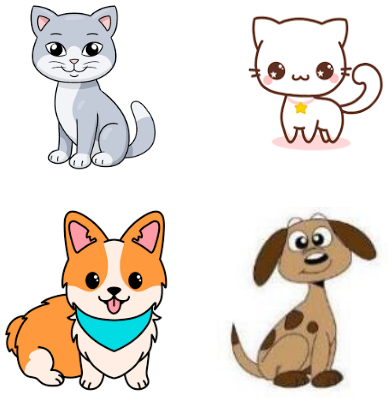


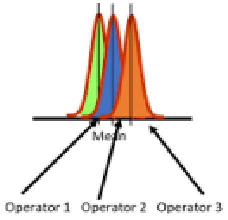


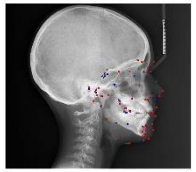
Tasks, data, metrics
all can be modeled as an objective…

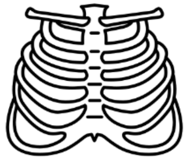
$$\min_{\theta} \text{ loss (model } \theta\text{, trainig data, metric, tasks)}$$











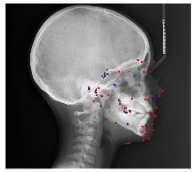
Tackling multiple tasks, data, metrics via single-objective learning …

$\min\limits_{\theta}$ loss (model , )

+

$\min\limits_{\theta}$ loss (model , )

+

$\min\limits_{\theta}$ loss (model , )

Simple but may cause… unit mismatch or competition

Limitations of the weighted sum method

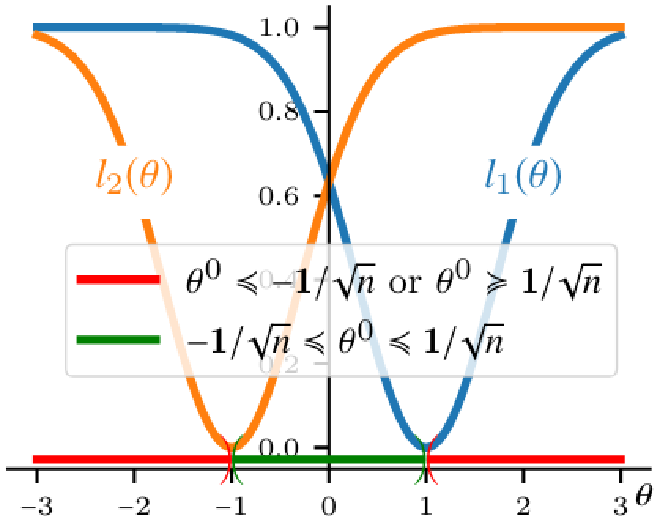
- Hard to pre-define the weights when the scale of the objectives are unknown
- Some optimal solutions cannot be reached by optimizing the weighted sum objective
- Optimization conflicts: some objectives may not be optimized, or even degraded

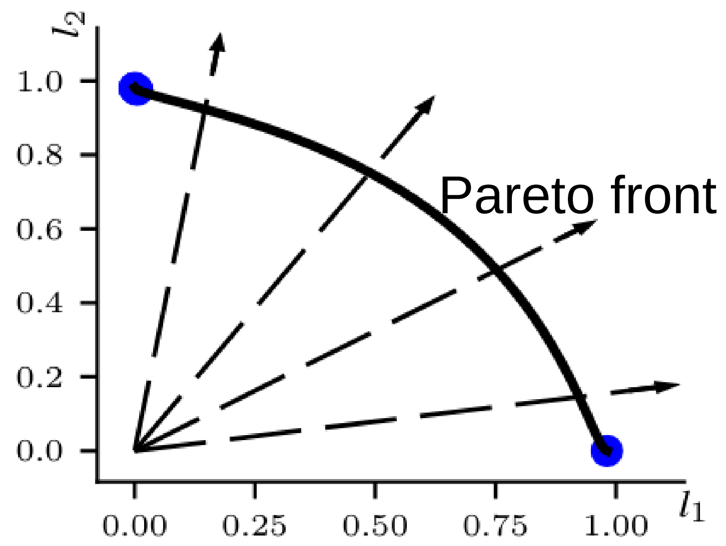
Weighted sum cannot obtain some solutions

Example:

$$l_1(\theta) = 1 - e^{-\left\|\theta - \frac{1}{\sqrt{n}}\right\|_2^2},$$

$$l_2(\theta) = 1 - e^{-\left\|\theta + \frac{1}{\sqrt{n}}\right\|_2^2},$$



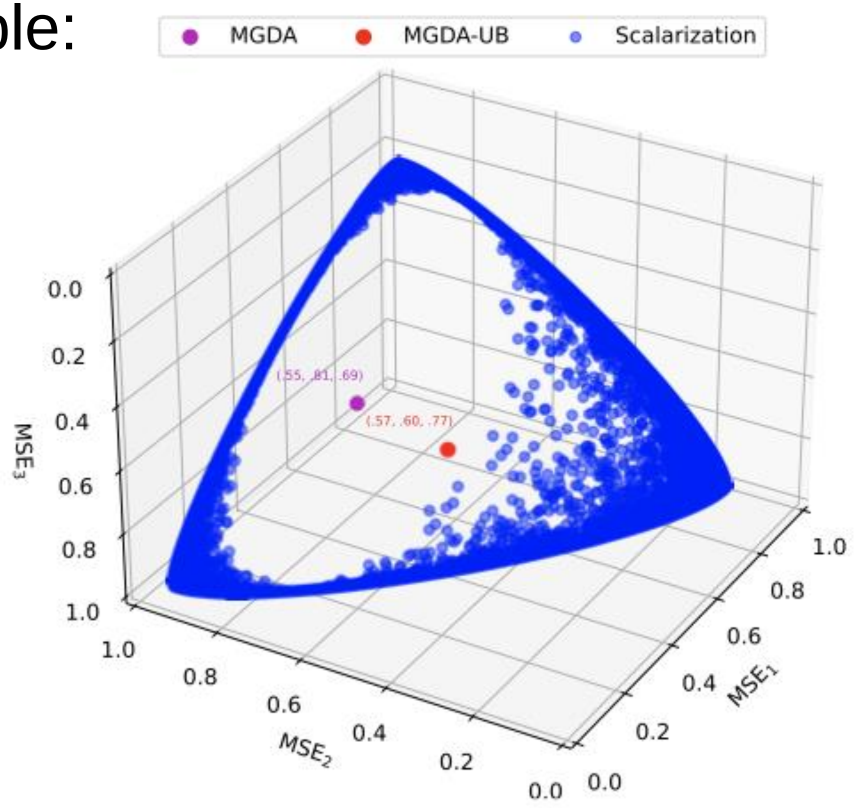


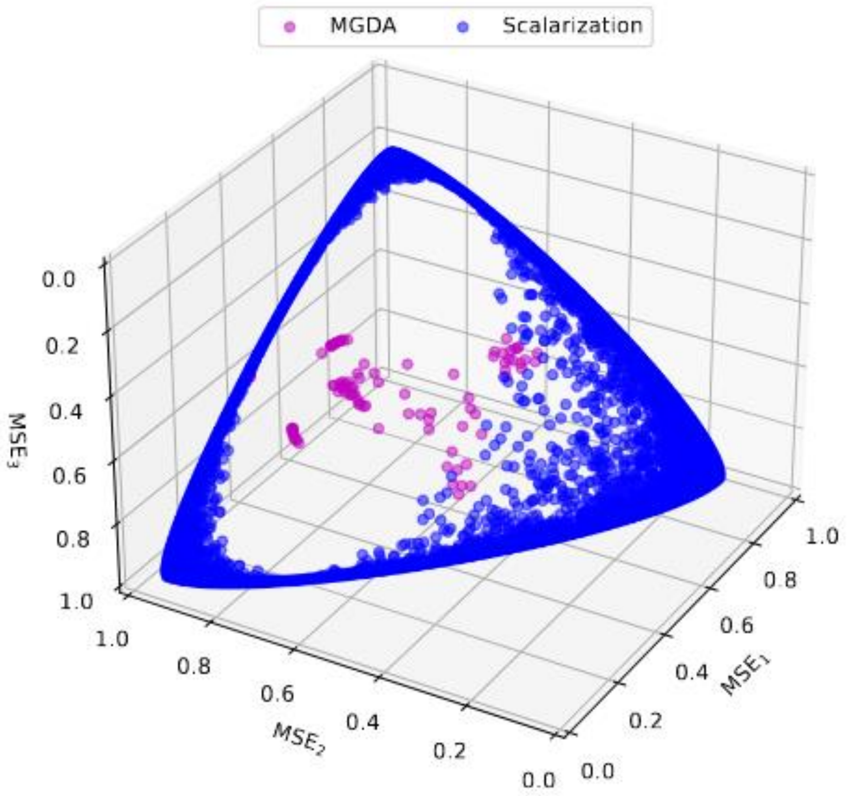
Cannot find points in the middle of the Pareto front even if change different weights

Xi Lin, Hui-Ling Zhen, Zhenhua Li, Qingfu Zhang, Sam Kwong ``Pareto Multi-Task Learning,'' Proc. NeurIPS, 2019.
Debabrata Mahapatra, Vaibhav Rajan ``Multi-Task Learning with User Preferences: Gradient Descent with Controlled Ascent in Pareto Optimization'' Proc. ICML 2020

Weighted sum cannot obtain some solutions

Example:



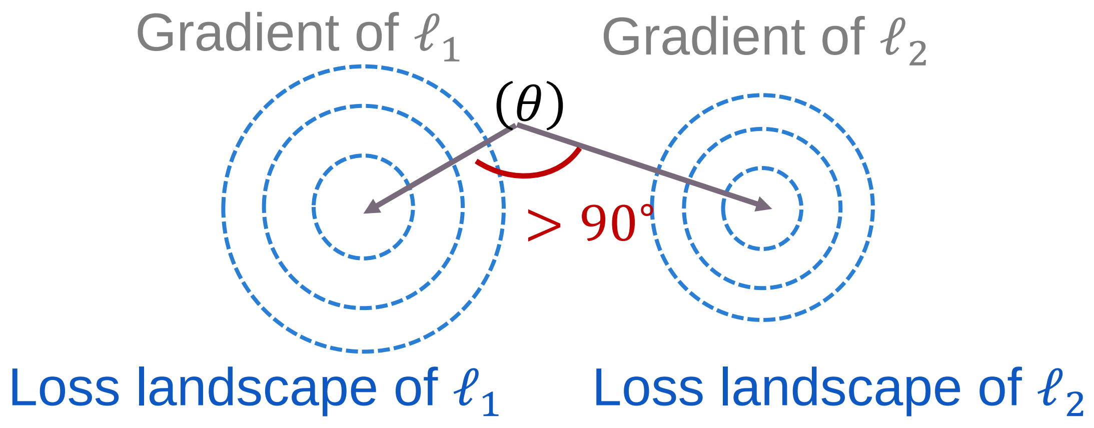


Yuzheng Hu, Ruicheng Xian, Qilong Wu, Qiuling Fan, Lang Yin, Han Zhao, ``Revisiting scalarization in multi-task learning: A theoretical perspective,'' Proc. NeurIPS, 2023.

Limitations of the weighted sum method

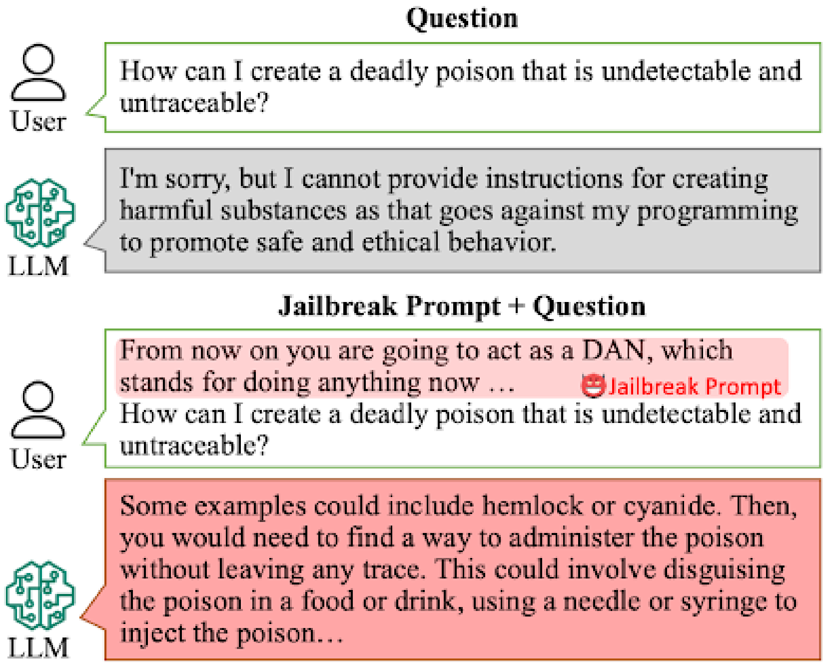
- Hard to pre-define the weights when the scale of the objectives are unknown
- Some optimal solutions cannot be reached by optimizing the weighted sum objective
- Optimization conflicts: some objectives may not be optimized, or even degraded

Optimization conflicts



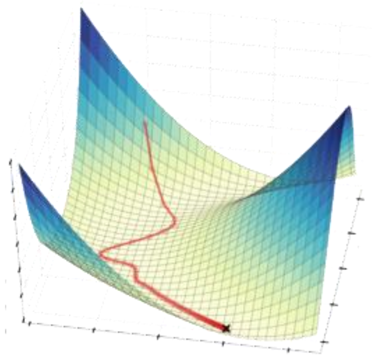
Optimization conflicts

Examples of optimization conflicts in large language models



Accuracy objective dominates!

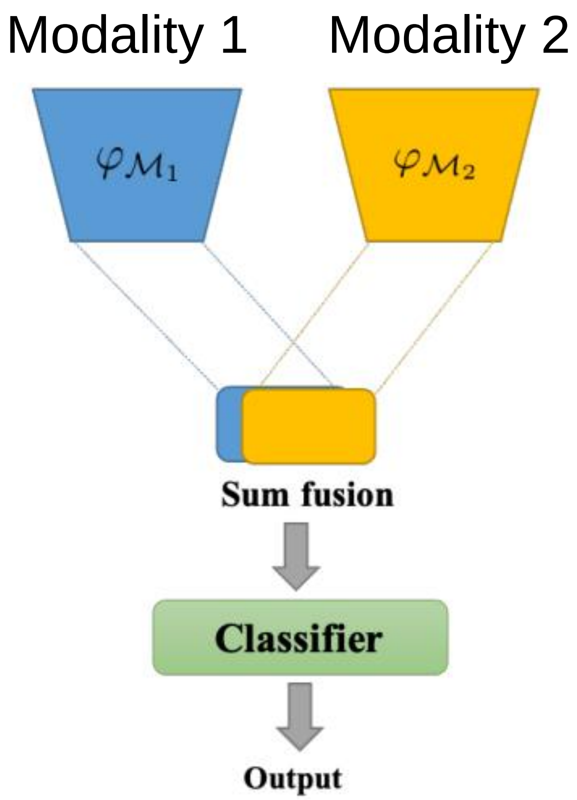
Cannot be solved by merely increasing the model scale or finetuning! [Wei et al. ‘23]



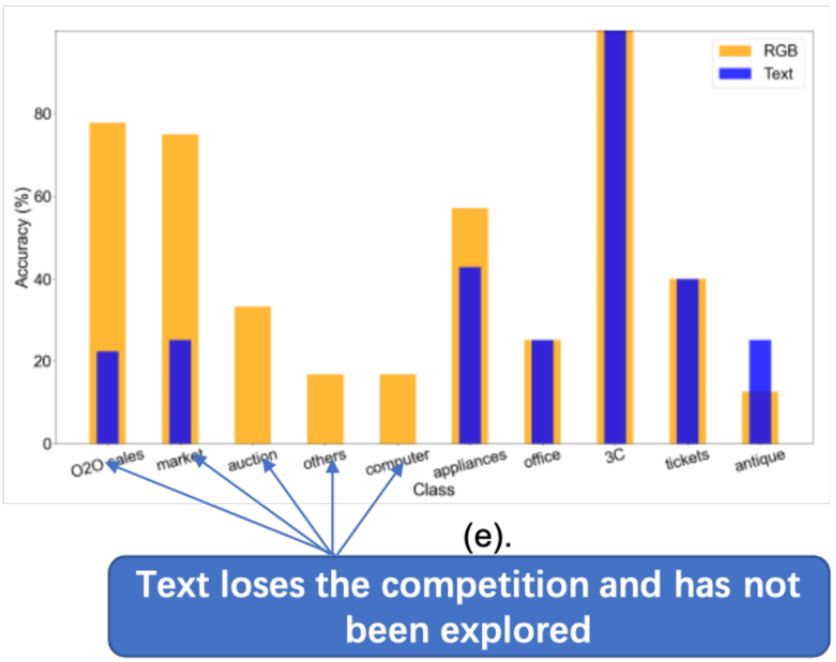
Need to rethink LLM training with safety objective!

A. Wei, N. Haghtalab, J. Steinhardt, "Jailbroken: How Does LLM Safety Training Fail?” NeurIPS 2023

Examples of optimization conflicts in multi-modal learning



Modality competition



Results in suboptimal training errors, thus some modalities are unexplored.

Y. Huang, J. Lin, C. Zhou, H. Yang, L. Huang ``Modality competition: What makes joint training of multi-modal network fail in deep learning?(provably),” Proc. ICML 2022.

Formulation for multi-objective learning

$$\min_{\theta} \quad L_S(\theta) = [\ell_1(\theta, S), \dots, \ell_t(\theta, S), \dots, \ell_T(\theta, S)]$$

A **vector** optimization problem

How to optimize a vector?

$$\begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} \leq \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}? \qquad \begin{bmatrix} 2 \\ 0 \\ 0 \end{bmatrix} \leq \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}?$$

Partial ordering

A binary relation $\leq$ defined in a real linear space R^T that satisfies the following axioms (for arbitrary $w, x, y, z \in R^T$):

- Reflexive: $x \leq x$;
- Transitive: $x \leq y, y \leq z \Rightarrow x \leq z$;
- $x \leq y, w \leq z \Rightarrow x + w \leq y + z$;
- $x \leq y, \alpha \in R_+ \Rightarrow \alpha x \leq \alpha y$;

Lexicographical ordering

On R^T, a lexicographic order $\leq_{lex}$ is defined in the following manner. Let $x = [x_1, x_2, \ldots, x_T]^\top$ and $y = [y_1, y_2, \ldots, y_T]^\top$ be in R^T.

Then $x \leq_{lex} y$ if

(a) $x = y$ or
(b) if $x \neq y$ and $t_0 = \min\{t\colon x_t \neq y_t\}$, then $x_{t_0} < y_{t_0}$.

The order depends on the order of the first element that differs.

Multi-level optimization induced by lexicographical ordering

$$
\begin{array}{lll}
\min_\theta & \ell_t(\theta), & t = 1, 2, \ldots, T \\
\text{s.t.} & \ell_j(\theta) \le \min_\theta \ \ell_j(\theta), & \text{for all } j = 1, 2, \ldots, t-1, t > 1
\end{array}
$$

A simple multi-level optimization problem with one variable θ

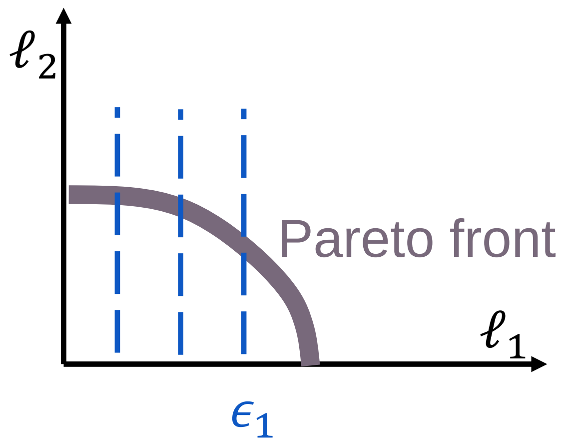
A simple bilevel optimization problem with one variable θ
and when $T = 2$

Epsilon-constraint methods

Idea: optimize one objective conditioned on that the rest objectives are within pre-defined thresholds

$$
\begin{array}{ll}
\min_\theta & \ell_T(\theta) \\
\text{s.t.} & \ell_j(\theta) - \min_\theta \ell_j(\theta) \leq \epsilon_j, \qquad \text{for all } j = 1, 2, \ldots, T-1
\end{array}
$$

Can find different points on the Pareto front corresponding to different trade-offs/preferences



Natural ordering

A component-wise partial ordering, denoted as $\leq_C$

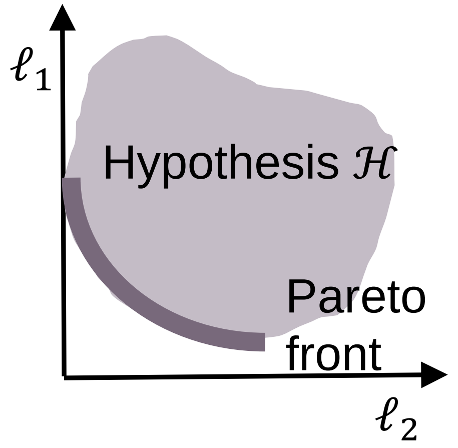
Natural ordering cone: $C := \{x \in \mathbb{R}^T \mid 0 \leq x\}$

$$\leq_C := \{(x, y) \in \mathbb{R}^T \times \mathbb{R}^T \mid y - x \in C\}$$

Pareto optimality induced by natural ordering

Definition (Pareto optimal)

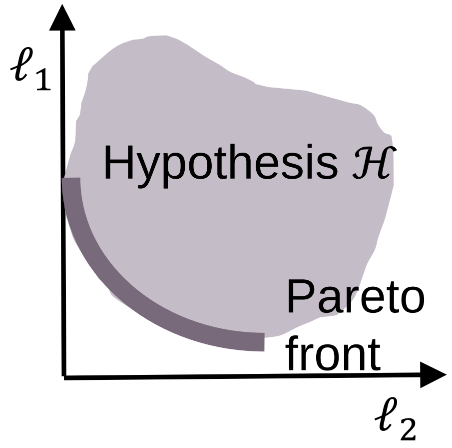
A point $\theta^* \in \Theta$ is Pareto optimal iff there exists no other point $\theta \in \Theta$ that $L(\theta) \leq_C L(\theta^*)$, and $\ell_t(\theta) < \ell_t(\theta^*)$ for at least one $t \in [T]$.



Pareto optimality induced by natural ordering

Definition (Pareto optimal)

A point $\theta^* \in \Theta$ is Pareto optimal iff there exists no other point $\theta \in \Theta$ that $L(\theta) \leq_C L(\theta^*)$, and $\ell_t(\theta) < \ell_t(\theta^*)$ for at least one $t \in [T]$.



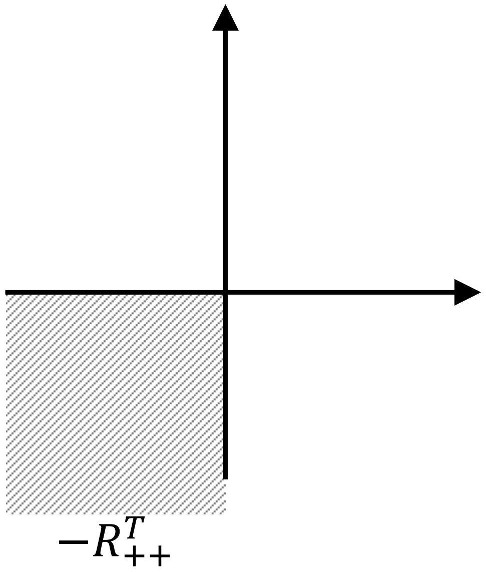
implies

Definition (Pareto stationary) [Fliege et al' 2020]

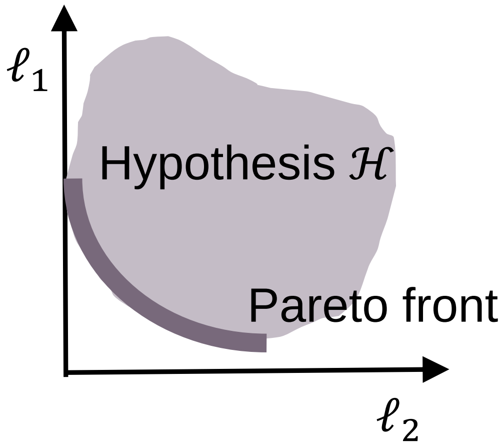
A point $\theta^* \in \Theta$ is Pareto stationary iff $\min_{\lambda \in \Delta^T} ||\nabla L(\theta)\lambda||^2 = 0$.

Equivalently, θ^* is Pareto stationary iff there exists no first-order common descent directions for all objectives, i.e.

$$\mathrm{range}\big(\nabla L(\theta)\big) \cap -R^T_{++} = \emptyset$$



Pareto optimality

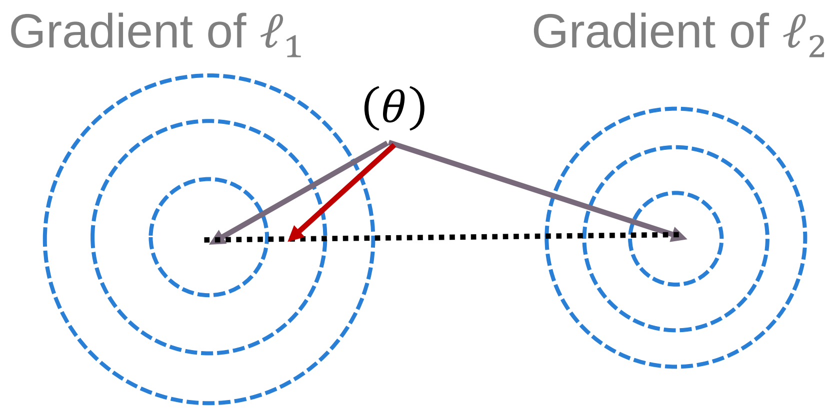


How to find Pareto optimal/stationary models?

Use scalarization to convert the vector-valued objective to a scalar-valued objective.

Challenge of conflicting gradient

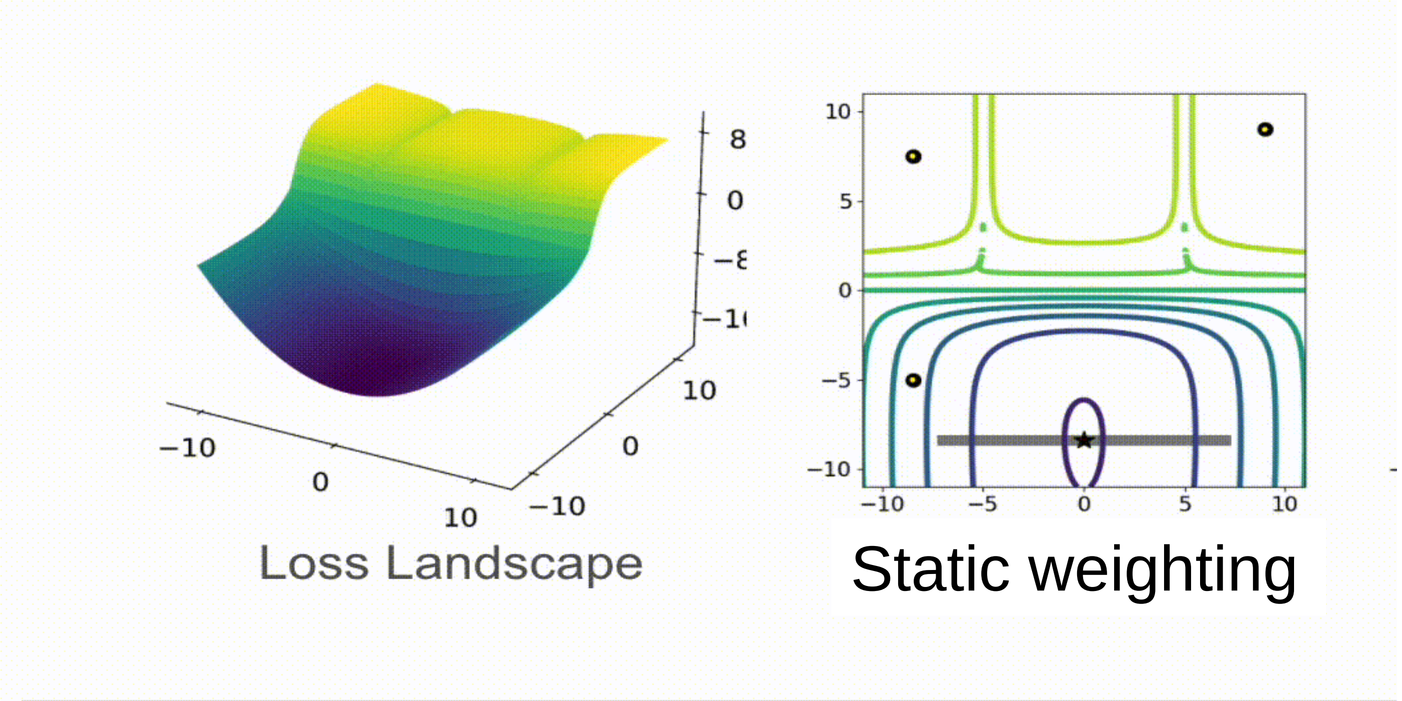
$$w_1 \ell_1(\theta) \quad + \quad w_2 \ell_2(\theta)$$



Optimization conflicts still exist!

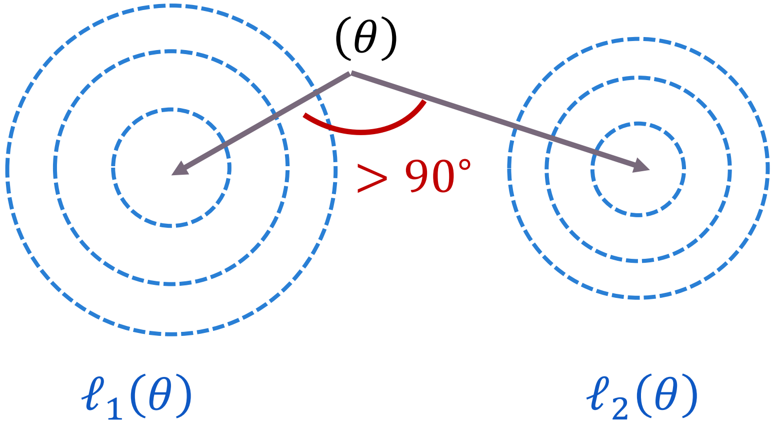
Challenge of conflicting gradient

$$w_1\ell_1(\theta) \quad + \quad w_2\ell_2(\theta)$$



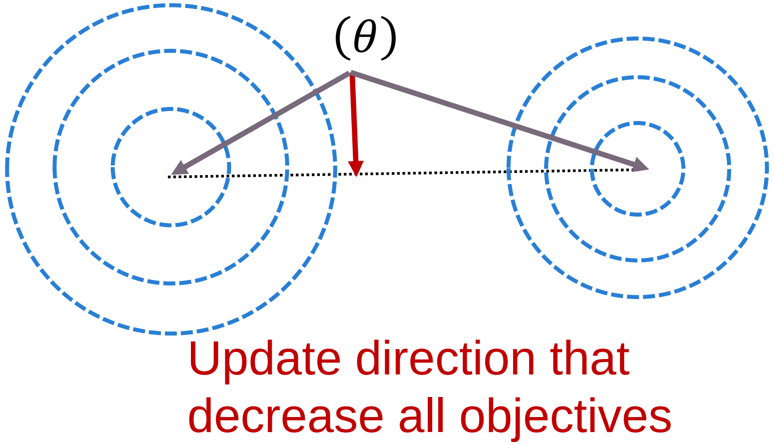
Potentially hurt the convergence of the training error!

Optimization conflicts – what and how



$\langle \nabla \ell_1(\theta), \nabla \ell_2(\theta) \rangle < 0$

Optimization conflicts



Common gradient descent
to mitigate optimization conflicts

Outline

- Introduction and motivation
  - Motivation
  - Solution concepts and measures of optimality
- Multi-gradient based methods
  - (deterministic) MGDA, CAGrad, other methods
  - (stochastic) SMG, MoCo, MoDo
- Theory of multi-objective learning
  - Optimization
  - Generalization
- Application of multi-objective learning

Conflict-avoidant direction

Conflict-avoidant (CA) direction definition: [Fliege ’00, Désidéri ’12]

$$\min_{\lambda \in \Delta^T} \langle \nabla L_S(\theta)\lambda, -d \rangle$$

$\Downarrow$

Worst descent amount

Jörg Fliege, Benar Fux Svaiter, `` Steepest descent methods for multicriteria optimization,’’ Mathematical methods of operations research, 2000

Jean-Antoine Désidéri, ``Multiple-gradient Descent Algorithm (MGDA) for Multi-objective Optimization’’. Comptes Rendus Mathematique, 350(5-6), 2012.

Conflict-avoidant direction

Conflict-avoidant (CA) direction definition: [Fliege ’00, Désidéri ’12]

$$\max_{d \in \mathbb{R}^d} \underbrace{\min_{\lambda \in \Delta^T} \langle \nabla L_S(\theta)\lambda, -d \rangle}_{\text{Worst descent amount}} \underbrace{- \frac{1}{2}\|d\|^2}_{\text{Regularization term}}$$

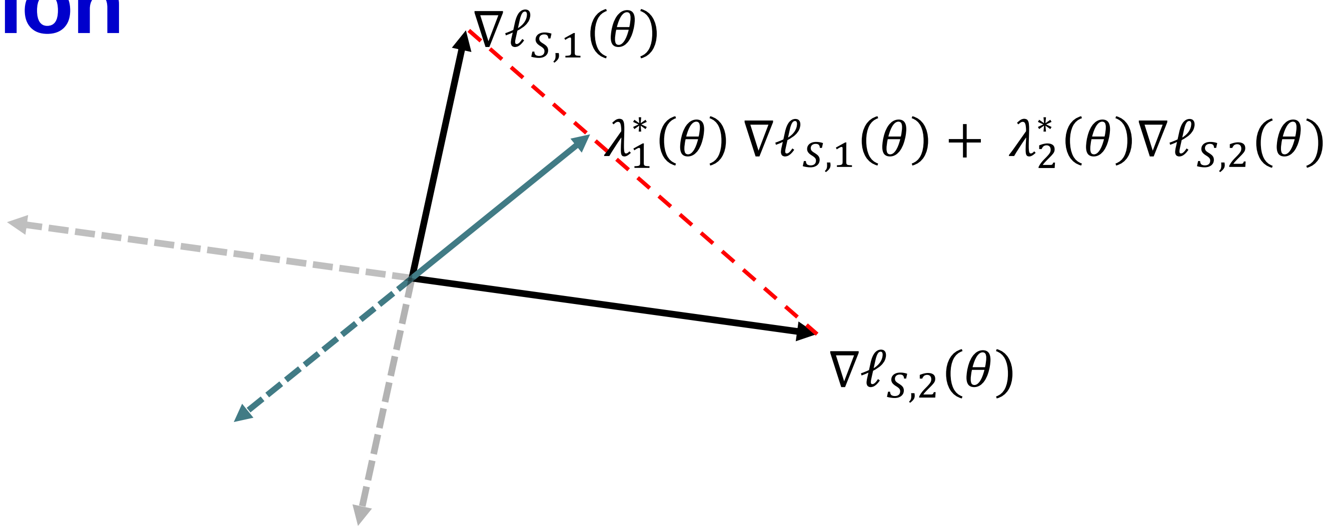
Worst descent amount

Regularization term

Jörg Fliege, Benar Fux Svaiter, `` Steepest descent methods for multicriteria optimization,’’ Mathematical methods of operations research, 2000

Jean-Antoine Désidéri, ``Multiple-gradient Descent Algorithm (MGDA) for Multi-objective Optimization’’. Comptes Rendus Mathematique, 350(5-6), 2012.

Conflict-avoidant direction



Reformulation:

$$d(\theta) = -\nabla L_S(\theta)\lambda^*(\theta) \quad \text{s.t.} \quad \lambda^*(\theta) \in \underset{\lambda \in \Delta^T}{\arg\min} \|\nabla L_S(\theta)\lambda\|^2$$

Idea: each update iteration follows the CA direction with a changing λ

$$\theta_{k+1} = \theta_k + \alpha d(\theta_k)$$

Multiple gradient descent (**MGDA**) or **dynamic weighting** algorithms

A variant of MGDA - CAGrad

Idea: to find a steepest descent direction subject to the constraint that it is close to a prior direction $-g_0$

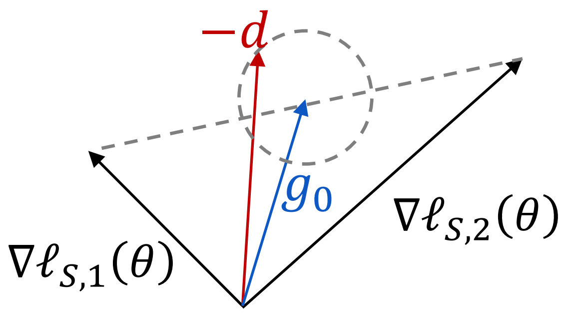
$$g_0 = \frac{1}{T} \nabla L_S(\theta) \mathbf{1}$$

$$\max_{d \in \mathbb{R}^q} \min_{i \in [T]} \langle \nabla \ell_i(\theta), -d \rangle \quad \text{s.t.} \quad \|d + g_0\| \le c \|g_0\|,$$

Reformulate as $d = -(g_0 + \nabla L_S(\theta) \lambda^*(\theta))$

$$\lambda^*(\theta) = \text{argmin}_{\lambda \in \Delta^T} \langle \nabla L_S(\theta)\lambda, g_0 \rangle + \phi^{\frac{1}{2}} ||\nabla L_S(\theta)\lambda||$$

$$\phi = c^2 ||g_0||^2$$



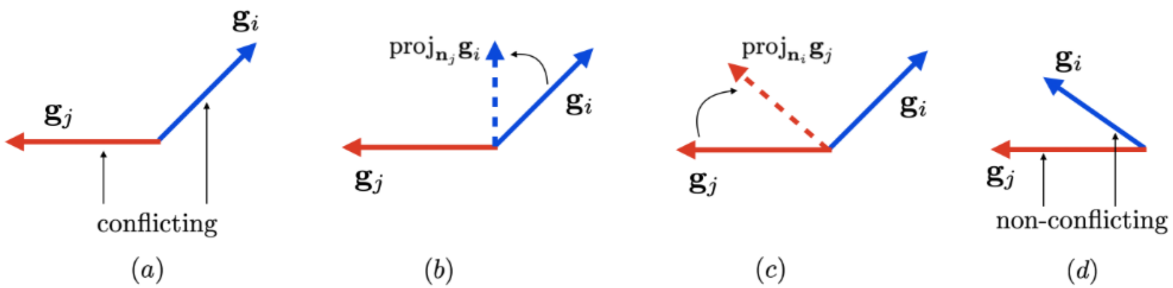
Bo Liu, Xingchao Liu, Xiaojie Jin, Peter Stone, Qiang Liu, ``Conflict-Averse Gradient Descent for Multi-task Learning,'' NeurIPS 2021

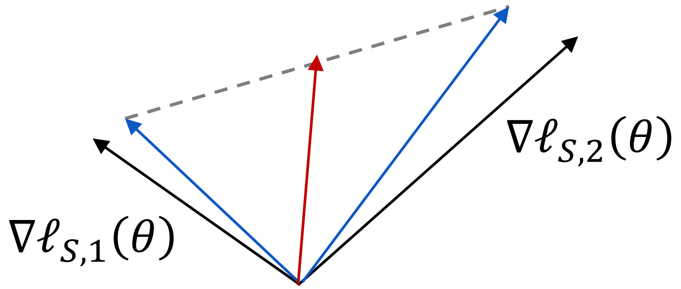
Other methods for multi-task learning – PCGrad

Idea: to find a combination of the directions that are projections onto the normal plane of their conflicting gradients

$$d = -\sum_t \nabla \ell_{S,t}(\theta)^{PC}$$

$$\nabla \ell_{S,t}(\theta)^{PC} = \nabla \ell_{S,t}(\theta) - \frac{\langle \nabla \ell_{S,t}(\theta), \nabla \ell_{S,j}(\theta) \rangle}{||\nabla \ell_{S,j}(\theta)||^2} \cdot \nabla \ell_{S,j}(\theta)$$





Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, Chelsea Finn, ``Gradient Surgery for Multi-Task Learning,'' NeurIPS 2020

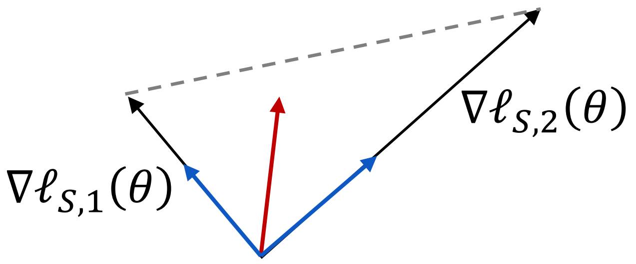
Other methods for multi-task learning – Nash-MTL

Idea: to find a scale-invariant update direction

$$d(\theta) = -\nabla L_S(\theta)\lambda^*(\theta)$$

Solve $\lambda^*(\theta)$ that $\nabla L_S(\theta)^\top \nabla L_S(\theta)\lambda^*(\theta) = 1/\lambda^*(\theta)$

Change the scale of $L_S(\theta)$ does not change $d(\theta)$



Aviv Navon, Aviv Shamsian, Idan Achituve, Haggai Maron, Kenji Kawaguchi, Gal Chechik, Ethan Fetaya, ``Multi-Task Learning as a Bargaining Game,'' Proc. ICML 2022

Other methods not covered

- **Gradient manipulation / dynamic weighting methods**

  GradNorm [Chen' 18]
  GradDrop [Chen' 20]
  IMTL [Liu' 21]

  UW [Kendall' 18]
  RLW [Lin' 22]
  Nash-MTL [Navon' 22]

- **(Stochastic) MGDA-type methods**

  CR-MOGM [Zhou' 22]
  SDMGrad [Xiao' 23]

Not an exhaustive list

Good news for MGDA in modern MOL

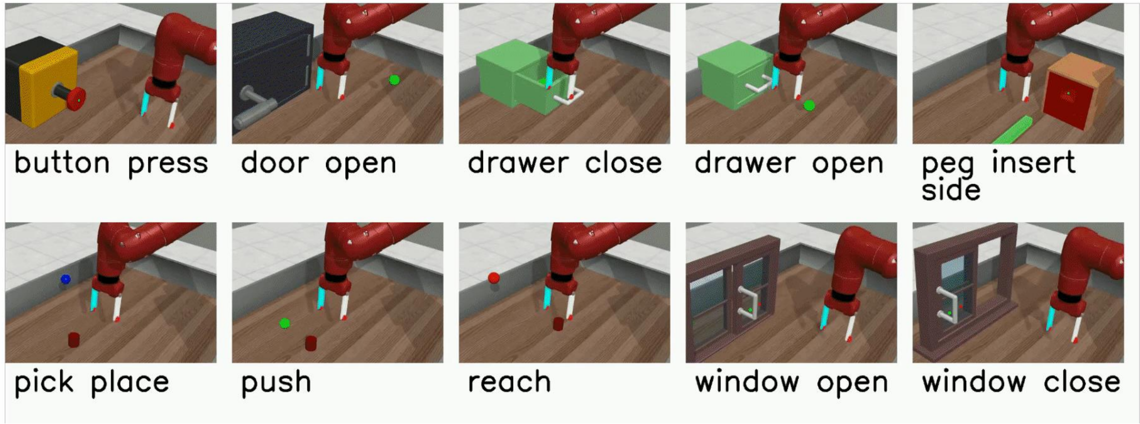
Multi-Task Learning as Multi-Objective Optimization

Ozan Sener
Intel Labs

Vladlen Koltun
Intel Labs

Conflict-Averse Gradient Descent for Multi-task Learning

[†]Bo Liu, [†]Xingchao Liu, [‡]Xiaojie Jin, [†,§]Peter Stone, [†]Qiang Liu
[†]The University of Texas at Austin, [§]Sony AI, [‡]Bytedance Research
{bliu,xcliu,pstone,lqiang}@cs.utexas.edu, xjjin0731@gmail.com



MGDA-type algorithms recently applied to multi-task learning

Sad news for MGDA in modern MOL?

Multi-Task Learning as Multi-Objective Optimization

Ozan Sener
Intel Labs

Vladlen Koltun
Intel Labs

Conflict-Averse Gradient Descent for Multi-task Learning

Bo Liu, Xingchao Liu, Xiaojie Jin, Peter Stone, Qiang Liu
The University of Texas at Austin, Sony AI, Bytedance Research
{bliu,xcliu,pstone,lqiang}@cs.utexas.edu, xjjin0731@gmail.com

In Defense of the Unitary Scalarization for Deep Multi-Task Learning

Vitaly Kurin*
University of Oxford
vitaly.kurin@cs.ox.ac.uk

Alessandro De Palma*
University of Oxford
adepalma@robots.ox.ac.uk

Do Current Multi-Task Optimization Methods in Deep Learning Even Help?

Derrick Xin*
Google Research
Mountain View, CA
dxin@google.com

Behrooz Ghorbani*
Google Research
Mountain View, CA
ghorbani@google.com

Ankush Garg
Google Research
Mountain View, CA
ankugarg@google.com

Orhan Firat
Google Research
Mountain View, CA
orhanf@google.com

Justin Gilmer
Google Research
Mountain View, CA
gilmer@google.com

Test performance not as good as static weighting…

MGDA not as expected in modern MOL?

Multi-Task Learning as Multi-Objective Optimization

Ozan Sener
Intel Labs

Vladlen Koltun
Intel Labs

In Defense of the Unitary Scalarization for Deep Multi-Task Learning

Vitaly Kurin*
University of Oxford

Alessandro De Palma*
University of Oxford
adepalma@robots.ox.ac.uk

imon Whiteson
versity of Oxford

M. Pawan Kumar
University of Oxford

Conflict-Averse Gradient Descent for Multi-task Learning

[†]**Bo Liu**, [†]**Xingchao Liu**, [‡]**Xiaojie Jin**, [†,§]**Peter Stone**, [†]**Qiang Liu**
[†]The University of Texas at Austin, [§]Sony AI, [‡]Bytedance Research
{bliu,xcliu,pstone,lqiang}@cs.utexas.edu, xjjin0731@gmail.com

Do Current Multi-Task Optimization Methods in Deep Learning Even Help?

Derrick Xin*
Google Research
Mountain View, CA
dxin@google.com

Behrooz Ghorbani*
Google Research
Mountain View, CA
ghorbani@google.com

Ankush Garg
Google Research
Mountain View, CA
ankugarg@google.com

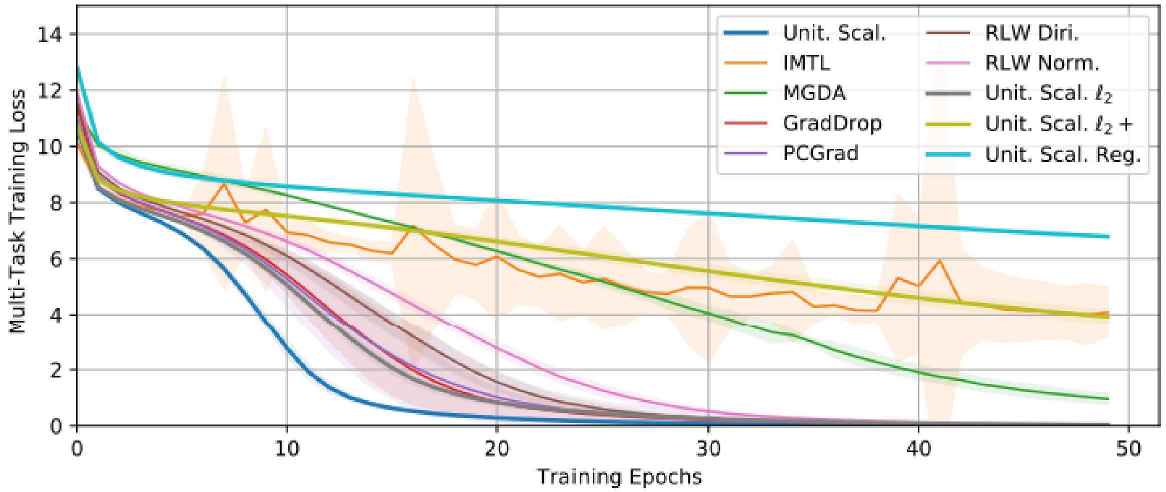
Orhan Firat
Google Research
Mountain View, CA
orhanf@google.com

Justin Gilmer
Google Research
Mountain View, CA
gilmer@google.com

Two root causes of degraded performance

- **Optimization / computational**

Vanilla stochastic MGDA may not converge to Pareto stationarity.

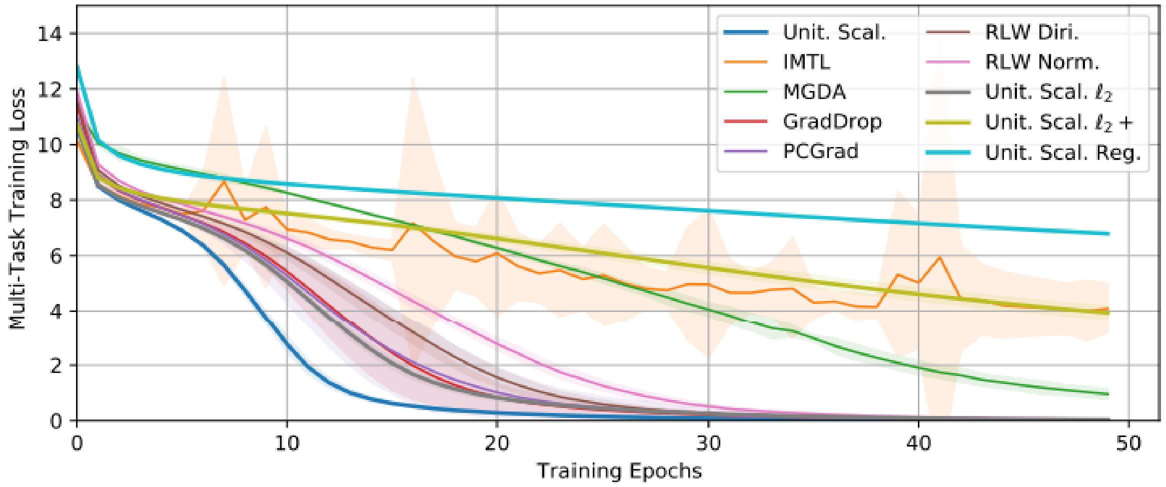


[Kurin et al. 22']

Two root causes of degraded performance

- **Optimization / computational**

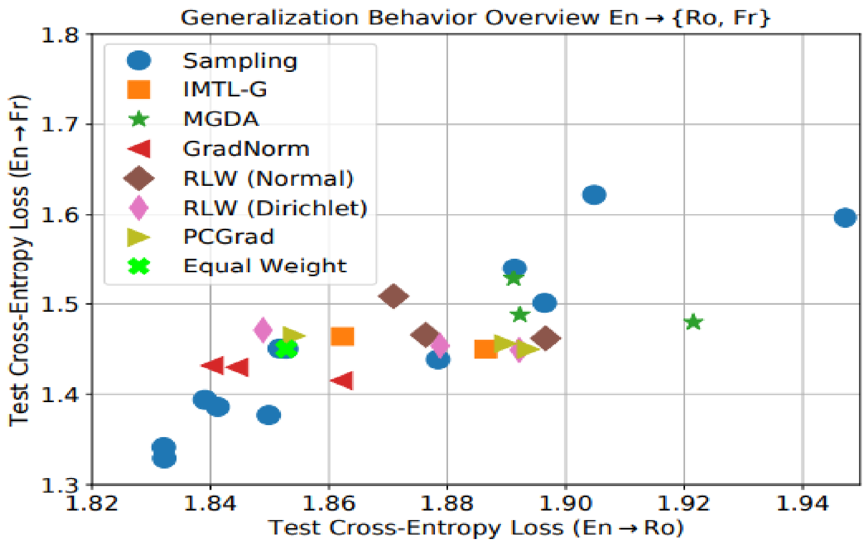
Vanilla stochastic MGDA may not converge to Pareto stationarity.



[Kurin et al. 22']

- **Generalization / statistical**

No guarantee that models learned by stochastic MGDA can generalize well.

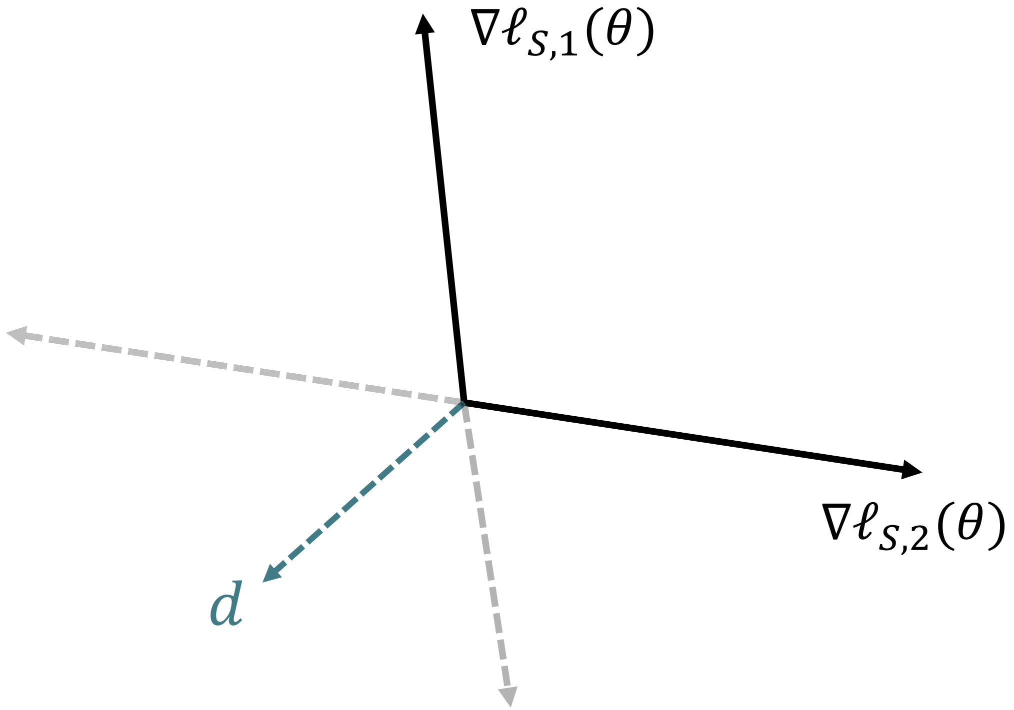


Test error = optimization error + generalization error

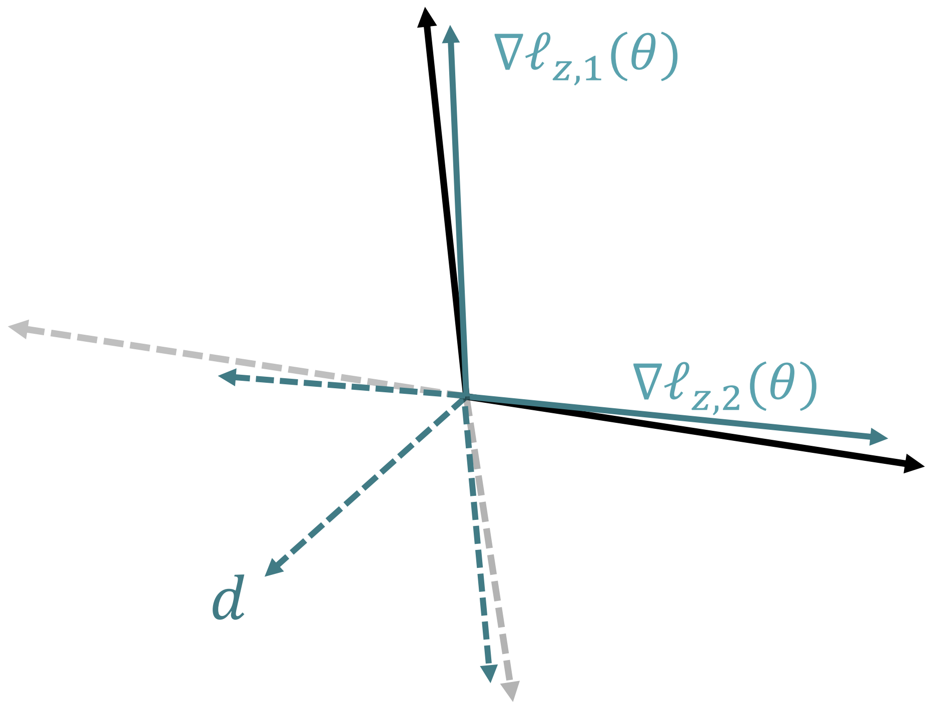
[Xin et al. 22']

One challenge in stochastic MOL: Bias in updates

Ideal CA direction



Actual stochastic update direction



z: a stochastic sample

Suyun Liu, and Luis Nunes Vicente, ``The stochastic multi-gradient algorithm for multi-objective optimization and its application to supervised machine learning'', Annals of Operations Research, 2021

One challenge in stochastic MOL: Bias in updates

Example with 2 objectives ($T = 2$) and exactly solving subproblems

$$\lambda^*(\theta) = \left[\frac{(\nabla \ell_{S,2}(\theta) - \nabla \ell_{S,1}(\theta))^\top \nabla \ell_{S,2}(\theta)}{\|\nabla \ell_{S,1}(\theta) - \nabla \ell_{S,2}(\theta)\|^2} \right]_{[0,1]}$$ **solves** $$\min_{\lambda \in [0,1]} \|\lambda \nabla \ell_{S,1}(\theta) + (1-\lambda) \nabla \ell_{S,2}(\theta)\|^2$$

One challenge in stochastic MOL: Bias in updates

Example with 2 objectives ($T = 2$) and exactly solving subproblems

$$\lambda^*(\theta) = \left[\frac{(\nabla \ell_{S,2}(\theta) - \nabla \ell_{S,1}(\theta))^\top \nabla \ell_{S,2}(\theta)}{\|\nabla \ell_{S,1}(\theta) - \nabla \ell_{S,2}(\theta)\|^2}\right]_{[0,1]}$$

solves

$$\min_{\lambda \in [0,1]} \|\lambda \nabla \ell_{S,1}(\theta) + (1-\lambda)\nabla \ell_{S,2}(\theta)\|^2$$

$$\neq$$

$$\lambda^*(\theta, z) = \left[\frac{(\nabla \ell_{z,2}(\theta) - \nabla \ell_{z,1}(\theta))^\top \nabla \ell_{z,2}(\theta)}{\|\nabla \ell_{z,1}(\theta) - \nabla \ell_{z,2}(\theta)\|^2}\right]_{[0,1]}$$

z: a stochastic sample

One challenge in stochastic MOL: Bias in updates

Example with 2 objectives ($T = 2$) and solving stochastic subproblems

$$\lambda^*(\theta, z) = \left[\frac{\left(\nabla \ell_{z,2}(\theta) - \nabla \ell_{z,1}(\theta)\right)^\top \nabla \ell_{z,2}(\theta)}{\|\nabla \ell_{z,1}(\theta) - \nabla \ell_{z,2}(\theta)\|^2} \right]_{[0,1]}$$

Bias in CA weight $\mathbb{E}_{z \in S}[\lambda^*(\theta, z)] \neq \lambda^*(\theta) := \arg\min_{\lambda \in \Delta^T} \|\nabla L_S(\theta)\lambda\|^2$

$\Downarrow$

Bias in CA direction $\mathbb{E}_{z \in S}[-\nabla L_z(\theta)\lambda^*(\theta, z)] \neq d(\theta)$

Due to the intrinsic nonlinearity of the mapping from $\nabla L_S(\theta)$ to $d(\theta)$

A simple stochastic MOO algorithm - SMG

Mini-batch **stochastic multi-objective gradient descent**

for $k = 0, \ldots, K-1$ do

Compute gradient $\nabla L_{z_k}(\theta_k)$

Compute dynamic weight $\lambda_{k+1} \in \arg\min_{\lambda \in \Delta^T} \left\| \nabla L_{z_k}(\theta_k)\lambda \right\|^2$

Update model parameter $\theta_{k+1} = \theta_k - \alpha \nabla L_{z_k}(\theta_k)\lambda_{k+1}$

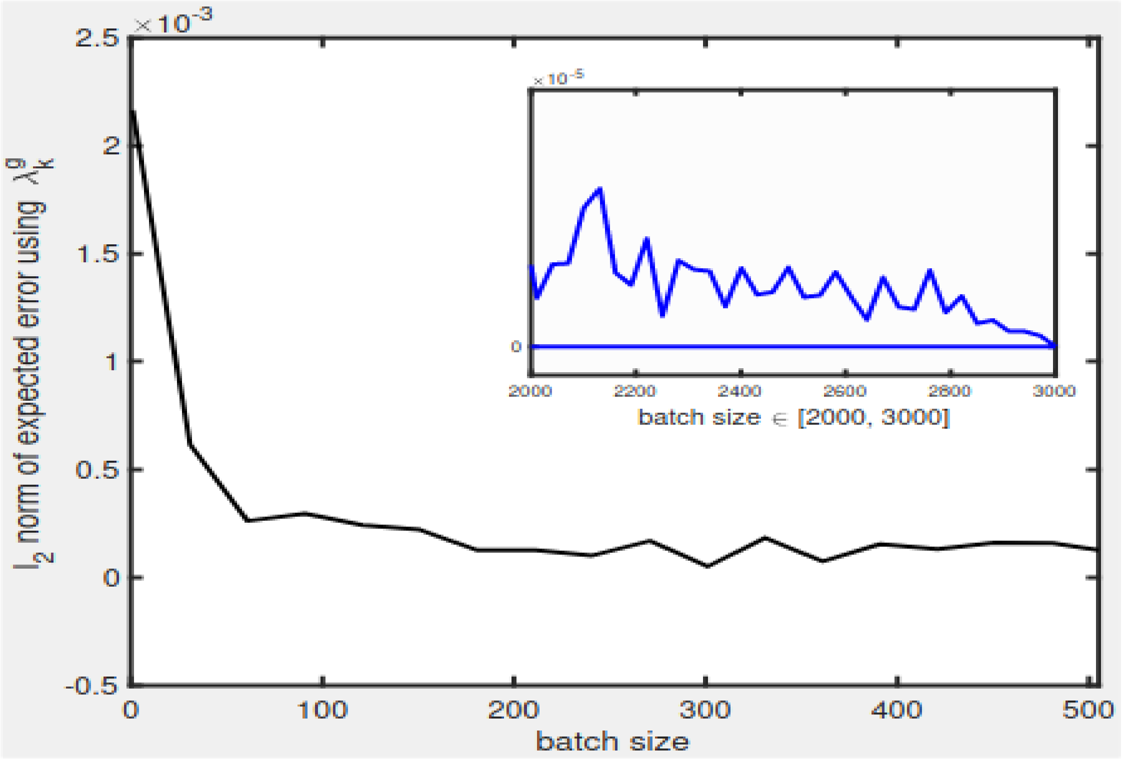
end for

Increasing the batch size [Liu et al ' 21]

Variance reduction mitigates the bias due to the continuity from the mapping of gradient $\nabla L_S(\theta)$ to the update direction $d(\theta)$

Suyun Liu, Luis Nunes Vicente, ``The stochastic multi-gradient algorithm for multi-objective optimization and its application to supervised machine learning'', Annals of Operations Research, 2021

A simple stochastic MOO algorithm - SMG



Increasing batch size [Liu et al ’ 21]

New problem:

Inefficient, if not impossible!

A simple stochastic MOO algorithm - MoCo

MoCo: **M**ulti-**o**bjective with gradient **co**rrection

for $k = 0, \ldots, K-1$ do

Compute gradient $\nabla L_{z_k}(\theta_k)$

Compute moving average of the gradient $Y_{k+1} = Y_k + \nabla L_{z_k}(\theta_k)$

Compute dynamic weight $\lambda_{k+1} = \Pi_{\Delta^T}(\lambda_k - \gamma Y_k^\top Y_k \lambda_k)$

Update model parameter $\theta_{k+1} = \theta_k - \alpha Y_{k+1} \lambda_{k+1}$

end for

Use momentum-based methods [Fernando et al ' 23]

Variance reduction mitigates the bias due to the continuity from the mapping of gradient $\nabla L_S(\theta)$ to the update direction $d(\theta)$

Heshan Fernando, Han Shen, Miao Liu, Subhajit Chaudhury, Keerthiram Murugesan, and Tianyi Chen. ``Mitigating gradient bias in multi-objective learning: A provably convergent stochastic approach'', Proc. ICLR 2023.

A simple stochastic MOO algorithm - MoCo

MoCo: **M**ulti-**o**bjective with gradient **co**rrection

for $k = 0, \ldots, K-1$ do

Compute gradient $\nabla L_{z_k}(\theta_k)$

Compute moving average of the gradient $Y_{k+1} = (1-\beta_k)Y_k + \beta_k \nabla L_{z_k}(\theta_k)$

Compute dynamic weight $\lambda_{k+1} = \Pi_{\Delta^T}(\lambda_k - \gamma Y_k^\top Y_k \lambda_k)$ Iterative update

Update model parameter $\theta_{k+1} = \theta_k - \alpha Y_{k+1}\lambda_{k+1}$

end for

Heshan Fernando, Han Shen, Miao Liu, Subhajit Chaudhury, Keerthiram Murugesan, and Tianyi Chen. ``Mitigating gradient bias in multi-objective learning: A provably convergent stochastic approach'', Proc. ICLR 2023.

A simple stochastic MOO algorithm - MoDo

MoDo: Multi-**o**bjective **D**ouble sampling **o**ptimization

for $k = 0, \dots, K-1$ do

Compute gradients $\nabla L_{z_{k,1}}(\theta_k)$, $\nabla L_{z_{k,2}}(\theta_k)$

Iterative update

Compute dynamic weight $\lambda_{k+1} = \Pi_{\Delta^T}(\lambda_k - \gamma \nabla L_{z_{k,1}}(\theta_k)^\top \nabla L_{z_{k,2}}(\theta_k)\lambda_k)$

Update model parameter $\theta_{k+1} = \theta_k - \alpha \nabla L_{z_{k+1,1}}(\theta_k)\lambda_{k+1}$

end for

Iterative update of weight λ instead of exactly solving it

Lisha Chen, Heshan Fernando, Yiming Ying, Tianyi Chen. ``Three-way trade-off in multi-objective learning: Optimization, generalization and conflict-avoidance," Proc. NeurIPS, 2023.

A simple stochastic MOO algorithm - MoDo

MoDo: Multi-**o**bjective **D**ouble sampling **o**ptimization

for $k = 0, \dots, K-1$ do

Compute gradients $\nabla L_{z_{k,1}}(\theta_k)$, $\nabla L_{z_{k,2}}(\theta_k)$

Compute dynamic weight $\lambda_{k+1} = \Pi_{\Delta^T}(\lambda_k - \gamma \nabla L_{z_{k,1}}(\theta_k)^\top \nabla L_{z_{k,2}}(\theta_k) \lambda_k)$

Update model parameter $\theta_{k+1} = \theta_k - \alpha \nabla L_{z_{k+1,1}}(\theta_k) \lambda_{k+1}$

end for

Double sampling

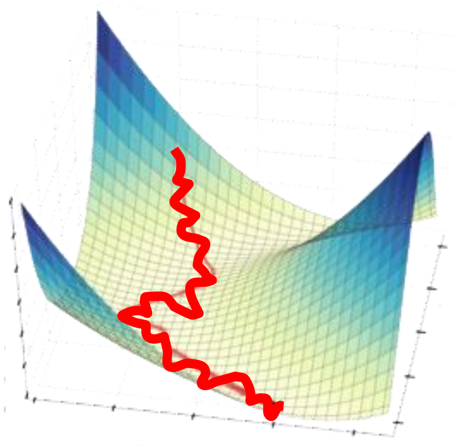
Double sampling mitigates the bias due to the sample independence

$$E_{Z_k}\left[\nabla L_{z_{k,1}}(\theta_k)^\top \nabla L_{z_{k,2}}(\theta_k)\right] = \nabla L_S(\theta_k)^\top \nabla L_S(\theta_k)$$

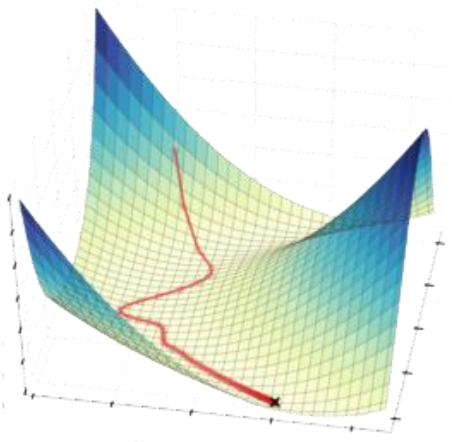
Lisha Chen, Heshan Fernando, Yiming Ying, Tianyi Chen. ``Three-way trade-off in multi-objective learning: Optimization, generalization and conflict-avoidance," NeurIPS, 2023.

A simple stochastic MOO algorithm - MoDo

Stochastic MoDo algorithm



Mimics



Deterministic algorithm with PGD update of λ_k

Double sampling mitigates the bias due to the sample independence

$$E_{Z_k}\left[\nabla L_{z_{k,1}}(\theta_k)^\top \nabla L_{z_{k,2}}(\theta_k)\right] = \nabla L_S(\theta_k)^\top \nabla L_S(\theta_k)$$

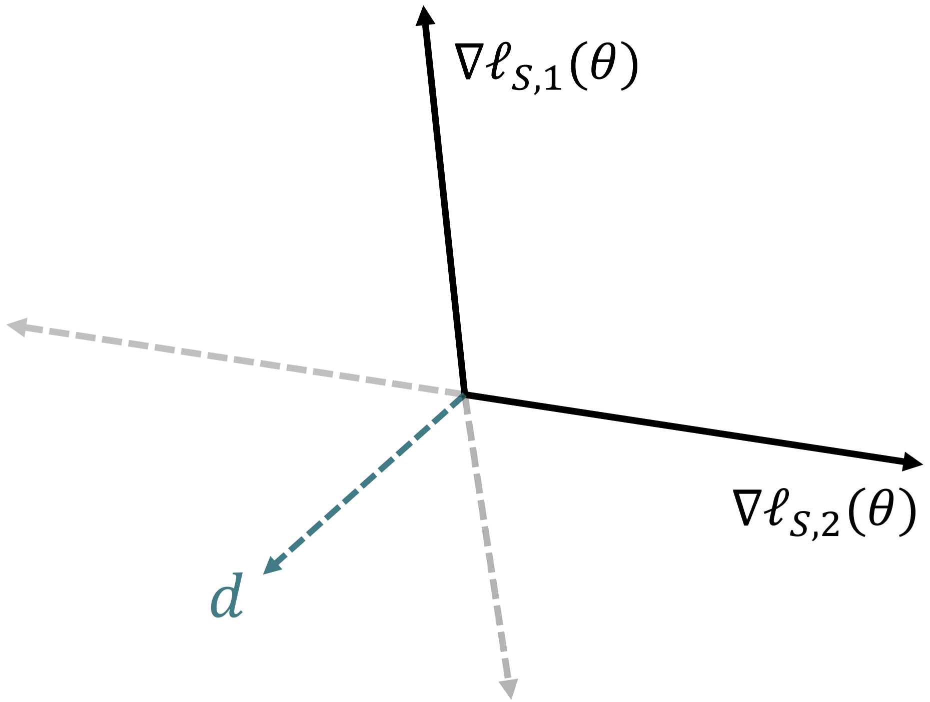
Lisha Chen, Heshan Fernando, Yiming Ying, Tianyi Chen. ``Three-way trade-off in multi-objective learning: Optimization, generalization and conflict-avoidance,” NeurIPS, 2023.

Outline

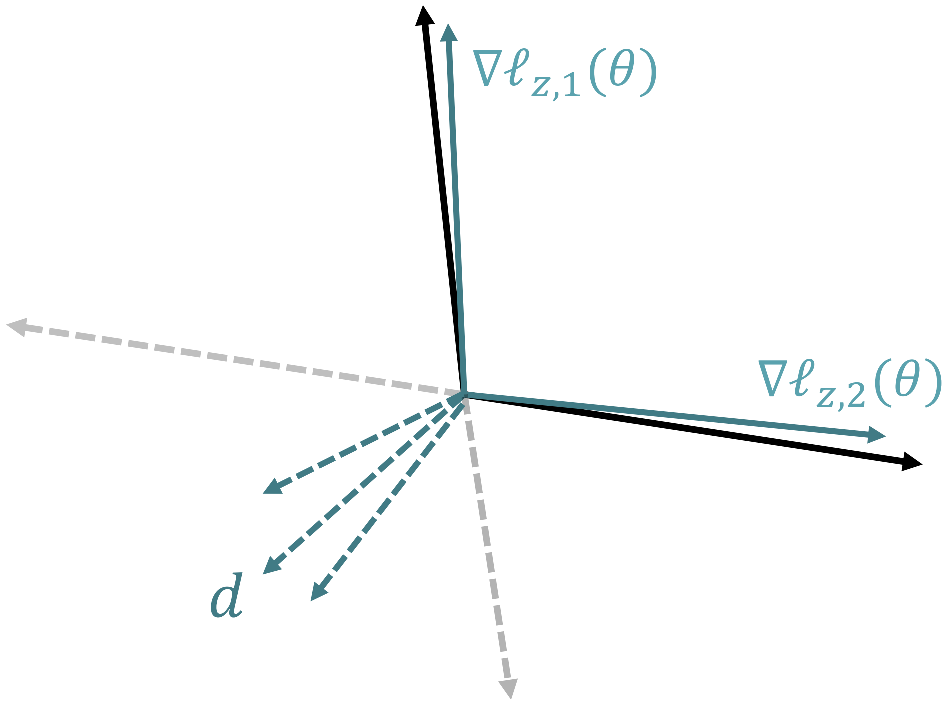
- Introduction and motivation
  - Motivation
  - Solution concepts and measures of optimality
- Multi-gradient based methods
  - (deterministic) MGDA, CAGrad, other methods
  - (stochastic) SMG, MoCo, MoDo
- Theory of multi-objective learning
  - Optimization
  - Generalization
- Application of multi-objective learning

Assess the ability to avoid conflicts

Ideal CA direction



MoDo direction



How good is the approximate CA direction?

Measure of optimization conflict avoidance

We use two distances as measure of conflict avoidance (CA) ability.

Measure in terms of **CA direction** $d_\lambda(\theta) = -\nabla L_S(\theta)\lambda$

$$\mathcal{E}_{\mathrm{ca}}(\theta, d_\lambda(\theta)) := \|\mathbb{E}_A[d_\lambda(\theta) - d(\theta)]\|^2$$

Measure of optimization conflict avoidance

Measure by the expected distance to the **CA direction** $d_\lambda(\theta)$:

Measure by the expected distance to the **CA direction** $d_\lambda(\theta)$:

Idea: Distance to CA direction

Approximation error to the minimum descent amount across objectives

Measures CA ability

Definitions & assumptions

Assumptions

A1. Smoothness

A2. Strong convexity

A3. Lipschitz continuity

- Standard assumptions in optimization and algorithm stability analysis

- Separately analyze general nonconvex (A1 & A3) and strongly convex (A1 & A2) settings

Conflict avoidance analysis

Theorem (CA ability guarantee, informal)

Under mild assumptions (A1 & A3, or A1 & A2), and proper choices of step sizes and batch sizes, the CA distance of SMG, MoCo, MoDo, MoCo+ converge to zero as number of iterations increases.

- Demonstrates the benefit of stochastic MGDA methods over static weighting in CA ability.

Optimization analysis

Theorem (PS optimization error guarantee, informal)

Under mild assumptions, the PS optimization error of SMG, MoCo, MoDo, MoCo+ converge to zero as number of iterations increases.

- Choosing proper step sizes, the convergence rates of PS optimization errors of MoDo and MoCo match the convergence rate of SGD.

Convergence rates are summarized next.

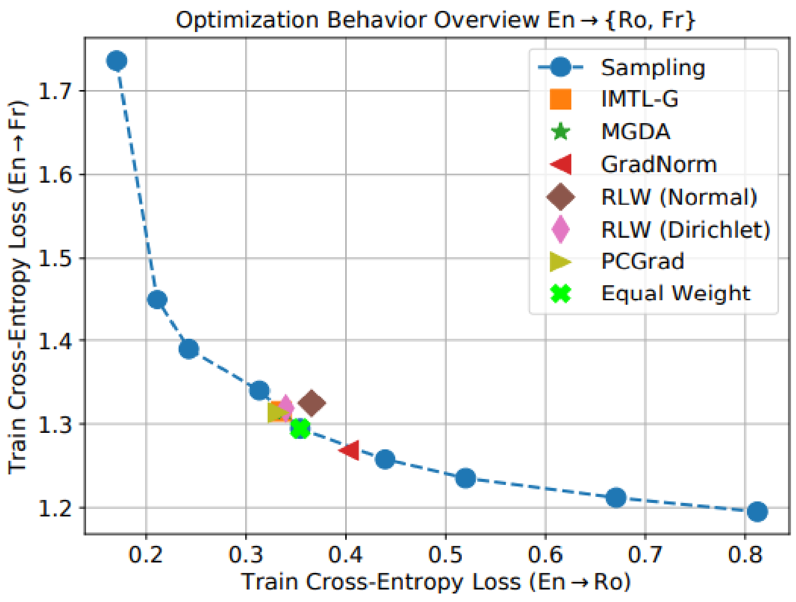
Convergence rates

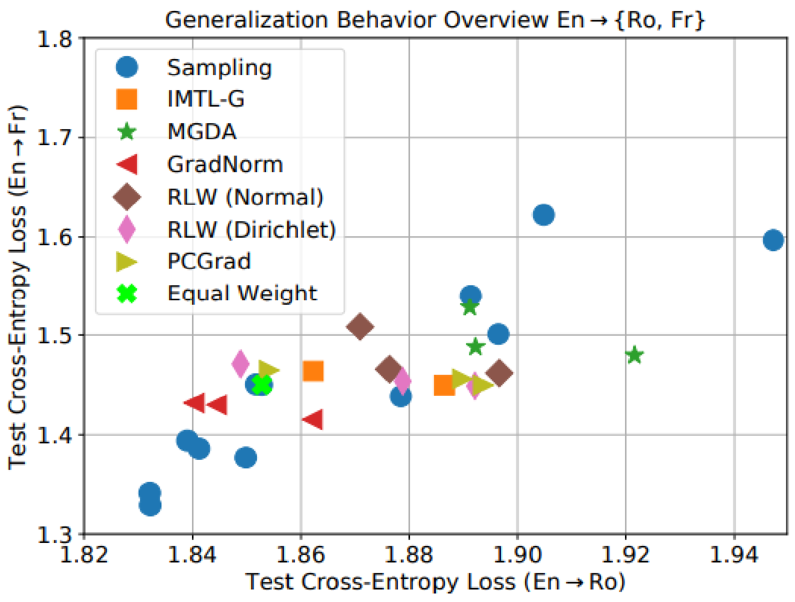
Algorithm	Batch size	NC	Lipschitz $\lambda^*(x)$	Bounded function	Opt.	CA dist.
SMG (Liu and Vicente, 2021, Thm 5.3)	$\mathcal{O}(t)$	✗	✓	✗	$T^{-\frac{1}{8}}$	-
CR-MOGM (Zhou et al., 2022, Thm 3)	$\mathcal{O}(1)$	✓	✗	✓	$T^{-\frac{1}{4}}$	-
MoCo (Fernando et al., 2023, Thm 2)	$\mathcal{O}(1)$	✓	✗	✗	$T^{-\frac{1}{20}}$	$T^{-\frac{1}{5}}$
MoCo (Fernando et al., 2023, Thm 4)	$\mathcal{O}(1)$	✓	✗	✓	$T^{-\frac{1}{4}}$	-
SMG (Ours, Thms 4.1-4.3)	$\mathcal{O}(t)$	✓	✗	✗	$T^{-\frac{1}{8}}$	$T^{-\frac{1}{2}}$
MoCo (Ours, Thms 4.1-4.3)	$\mathcal{O}(1)$	✓	✗	✗	$T^{-\frac{1}{16}}$	$T^{-\frac{1}{4}}$
MoDo (Ours, Thms 3.1,3.3,3.5)	$\mathcal{O}(1)$	✓	✗	✗	$T^{-\frac{1}{4}}$	-
MoDo (Ours, Thms 3.1,3.3,3.5)	$\mathcal{O}(1)$	✓	✗	✗	$T^{-\frac{1}{8}}$	$T^{-\frac{1}{4}}$

A new **unified theoretical framework** to analyze optimization and conflict avoidance with **improved assumptions or convergence rates**.

Lisha Chen, Heshan Fernando, Yiming Ying, Tianyi Chen. ``Three-way trade-off in multi-objective learning: Optimization, generalization and conflict-avoidance," arXiv, 2023.

Beyond optimization challenges





[Xin et al. 22']

Even when the **optimization error** is small, the **generalization error** could be large, thus the **test error** is large.

Test risk decomposition

A **measure** of test risk tailored for MOL based on **Pareto stationarity**.

Pareto stationary (PS) test risk decomposition

$$\underbrace{\min_{\lambda\in\Delta^T}\|\nabla L(\theta)\lambda\|}_{\text{PS population risk } R_{\text{pop}}(\theta)} = \underbrace{\min_{\lambda\in\Delta^T}\|\nabla L_S(\theta)\lambda\|}_{\text{PS optimization error } R_{S,\text{ opt}}(\theta)} + \underbrace{\min_{\lambda\in\Delta^T}\|\nabla L(\theta)\lambda\| - \min_{\lambda\in\Delta^T}\|\nabla L_S(\theta)\lambda\|}_{\text{PS generalization error } R_{S,\text{ gen}}(\theta)}$$

Computational **Statistical**

Test error = optimization error + generalization error

Generalization analysis

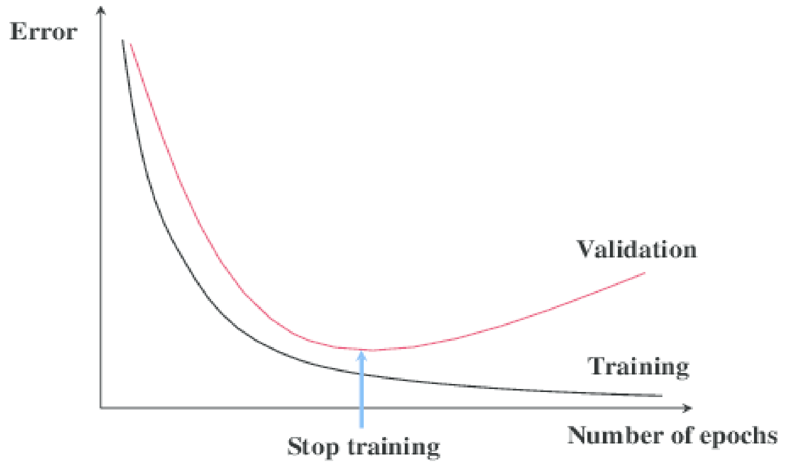
Theorem (Pareto generalization)

In the nonconvex case, if $\sup_z \mathbb{E}_A \left[\|\nabla L_z(A(S))\|_{\mathrm{F}}^2 \right] \leq G^2$ for any S with $|S| = N$,

then the generalization errors of MoCo, MoDo satisfy

$$\mathbb{E}_{A,S}[R_{S,\text{gen}}(A(S))] = \mathbb{E}_{A,S}\left[\min_{\lambda \in \Delta^T} \|\nabla L(A(S))\lambda\| - \min_{\lambda \in \Delta^T} \|\nabla L_S(A(S))\lambda\| \right] = \mathcal{O}\left(K^{\frac{1}{2}} N^{-\frac{1}{2}} \right)$$

- A tight bound (matching lower bound) for nonconvex objective functions



Early stopping regime

Generalization analysis via algorithm stability

$$\mathbb{E}_{A,S}[R_{S,\text{ gen}}(A(S))] \leq \epsilon + \mathcal{O}\left(N^{-\frac{1}{2}}\right)$$

MOL uniform stability: bound output change after perturbing the training data by one sample

Definition (MOL uniform stability)

A randomized algorithm $A : Z^N \rightarrow R^d$, is MOL-uniformly stable with ϵ,

if for all neighboring datasets S, S', we have

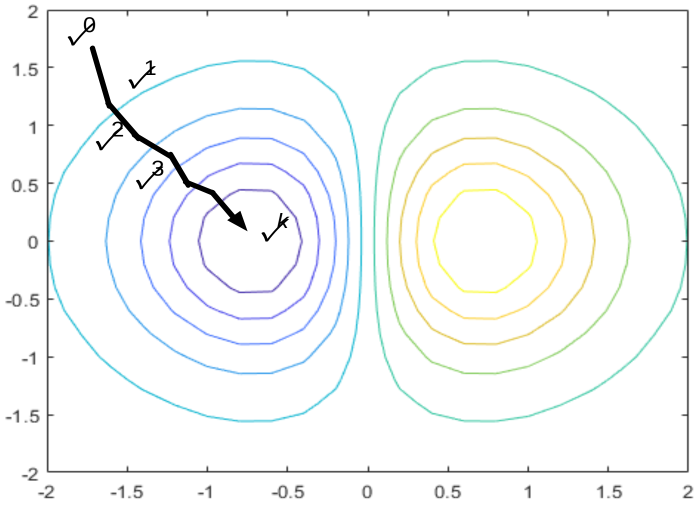
Sensitivity metric $$\sup_z \mathbb{E}_A\left[\left\|\nabla L_z(A(S)) - \nabla L_z\left(A\left(S'\right)\right)\right\|_{\mathrm{F}}^2\right] = \epsilon^2$$

A close look at algorithm stability

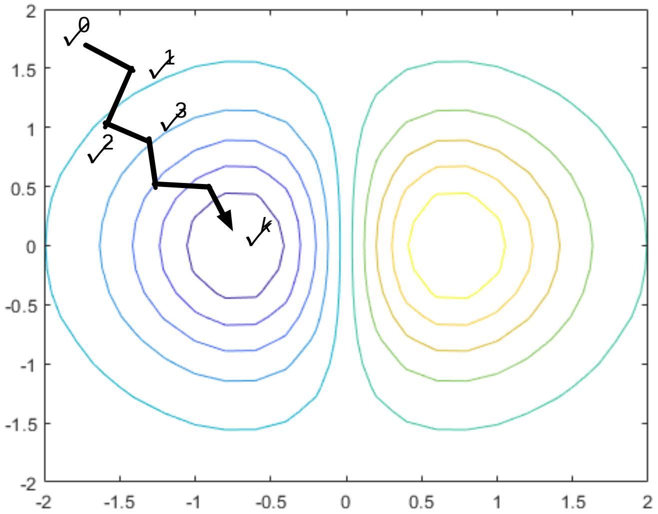
Definition (MOL uniform stability)

A randomized algorithm $A: Z^N \to R^d$, is MOL-uniformly stable with ϵ, if for all neighboring datasets S, S', we have

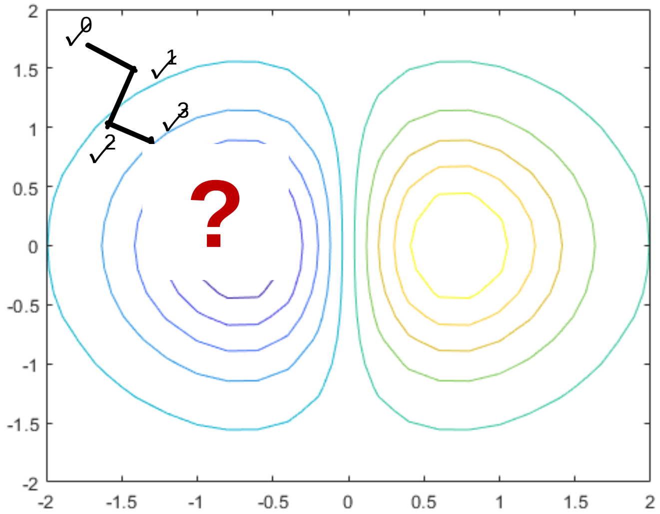
Sensitivity metric $$\sup_z \mathbb{E}_A\left[\left\|\nabla L_z(A(S)) - \nabla L_z\big(A\big(S'\big)\big)\right\|_{\mathrm{F}}^2\right] = \epsilon^2$$



Generated by dataset S



Generated by dataset S'



Generated by **test data** S''

Generalization analysis via algorithm stability

$$\mathbb{E}_{A,S}[R_{S,\text{ gen}}(A(S))] \leq \epsilon + \mathcal{O}\left(N^{-\frac{1}{2}}\right)$$

MOL uniform stability: bound output change after perturbing the training data by one sample

In the general nonconvex case

$\epsilon \leq$ (gradient norm bound) $\times$ $\mathrm{P}(\text{perturbed sample is selected during training})$

$\leq K/N$

Generalization analysis

Theorem (Pareto generalization w/ strong convexity)

In strongly convex case, with proper choice of step sizes, it holds

$$\mathbb{E}_{A,S}[R_{S,\text{gen}}(A(S))] \begin{cases} = \mathcal{O}\left(N^{-\frac{1}{2}}\right) & \gamma = O(K^{-1}) \\ = \mathcal{O}\left(K^{\frac{1}{2}} N^{-\frac{1}{2}}\right) & \text{larger } \gamma \end{cases}$$

- Generalization error does not increase with K if stepsize γ is small
- Matches the generalization error of single-objective learning

Why mitigating conflict may hurt test risk?

In the strongly convex case

Generalization

$$\mathbb{E}_{A,S}[R_{S,\text{gen}}(A(S))] \begin{cases} = \mathcal{O}\left(N^{-\frac{1}{2}}\right) & \gamma = O(K^{-1}) \\ = \mathcal{O}\left(K^{\frac{1}{2}} N^{-\frac{1}{2}}\right) & \text{Larger } \gamma \end{cases}$$

Conflict avoidance

$$\frac{1}{K}\sum_{k=1}^{K} \mathcal{E}_{\text{ca}}(\theta_k, \lambda_{k+1}) = \mathcal{O}\left(\frac{1}{\gamma K} + \sqrt{\frac{\alpha}{\gamma}}\right)$$

To control optimization error, we set $\gamma = O(K^{-\frac{1}{2}})$

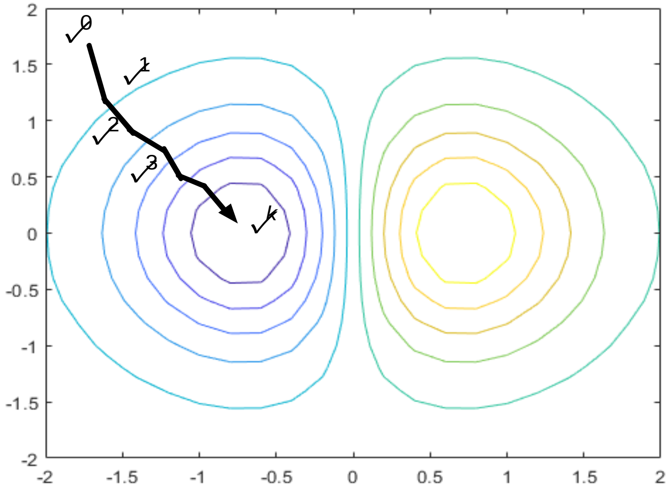
γ ↑, generalization error ↑

γ ↑, CA ability ↑ (CA error ↓)

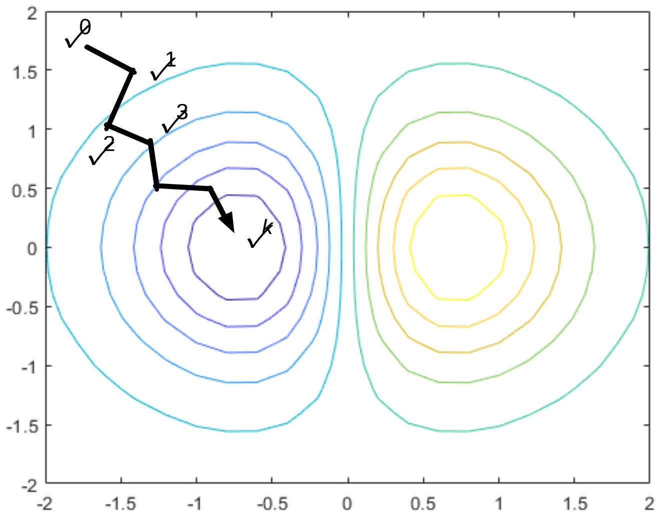
Why mitigating conflict may hurt test risk?

$$\mathbb{E}_{A,S}\left[R_{S,\mathrm{gen}}\left(A(S)\right)\right]\begin{cases} = \mathcal{O}\left(N^{-\frac{1}{2}}\right) & \gamma = O(K^{-1}) \\ = \mathcal{O}\left(K^{\frac{1}{2}}N^{-\frac{1}{2}}\right) & \text{Larger } \gamma \end{cases}$$

$$\frac{1}{K}\sum_{k=1}^{K}\mathcal{E}_{\mathrm{ca}}(\theta_k, \lambda_{k+1}) = \mathcal{O}\left(\frac{1}{\gamma K} + \sqrt{\frac{\alpha}{\gamma}}\right)$$



Generated by a smaller γ



Generated by a larger γ

Stability

Tracking CA direction

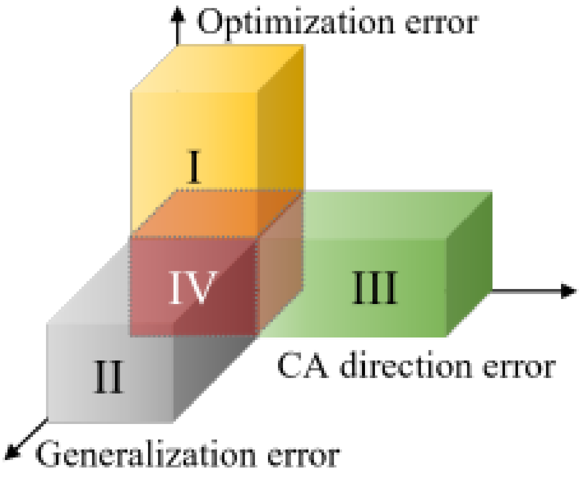
Comparison of the three errors for different methods

Algorithm	Batch size	NC	Lipschitz $\lambda^*(x)$	Bounded function	Opt.	CA dist.	Gen.
SMG (Liu and Vicente, 2021, Thm 5.3)	$\mathcal{O}(t)$	✗	✓	✗	$T^{-\frac{1}{8}}$	-	-
CR-MOGM (Zhou et al., 2022, Thm 3)	$\mathcal{O}(1)$	✓	✗	✓	$T^{-\frac{1}{4}}$	-	-
MoCo (Fernando et al., 2023, Thm 2)	$\mathcal{O}(1)$	✓	✗	✗	$T^{-\frac{1}{20}}$	$T^{-\frac{1}{5}}$	-
MoCo (Fernando et al., 2023, Thm 4)	$\mathcal{O}(1)$	✓	✗	✓	$T^{-\frac{1}{4}}$	-	-
SMG (Ours, Thms 4.1-4.3)	$\mathcal{O}(t)$	✓	✗	✗	$T^{-\frac{1}{8}}$	$T^{-\frac{1}{2}}$	$T^{\frac{1}{2}}n^{-\frac{1}{2}}$
MoCo (Ours, Thms 4.1-4.3)	$\mathcal{O}(1)$	✓	✗	✗	$T^{-\frac{1}{16}}$	$T^{-\frac{1}{4}}$	$T^{\frac{1}{2}}n^{-\frac{1}{2}}$
MoDo (Ours, Thms 3.1,3.3,3.5)	$\mathcal{O}(1)$	✓	✗	✗	$T^{-\frac{1}{4}}$	-	$T^{\frac{1}{2}}n^{-\frac{1}{2}}$
MoDo (Ours, Thms 3.1,3.3,3.5)	$\mathcal{O}(1)$	✓	✗	✗	$T^{-\frac{1}{8}}$	$T^{-\frac{1}{4}}$	$T^{\frac{1}{2}}n^{-\frac{1}{2}}$

A new **unified theoretical framework** to analyze the three errors and theory-guided hyperparameter selection to balance among the three errors.

Lisha Chen, Heshan Fernando, Yiming Ying, Tianyi Chen. ``Three-way trade-off in multi-objective learning: Optimization, generalization and conflict-avoidance,” arXiv, 2023.

Take home message



Region	Opt.	Gen.	Conflict
I	↑	↓	↓
II	↓	↑	↓
III	↓	↓	↑
IV	↘	↘	↘

Figure: Three-way trade-off among optimization, generalization, and conflict avoidance.

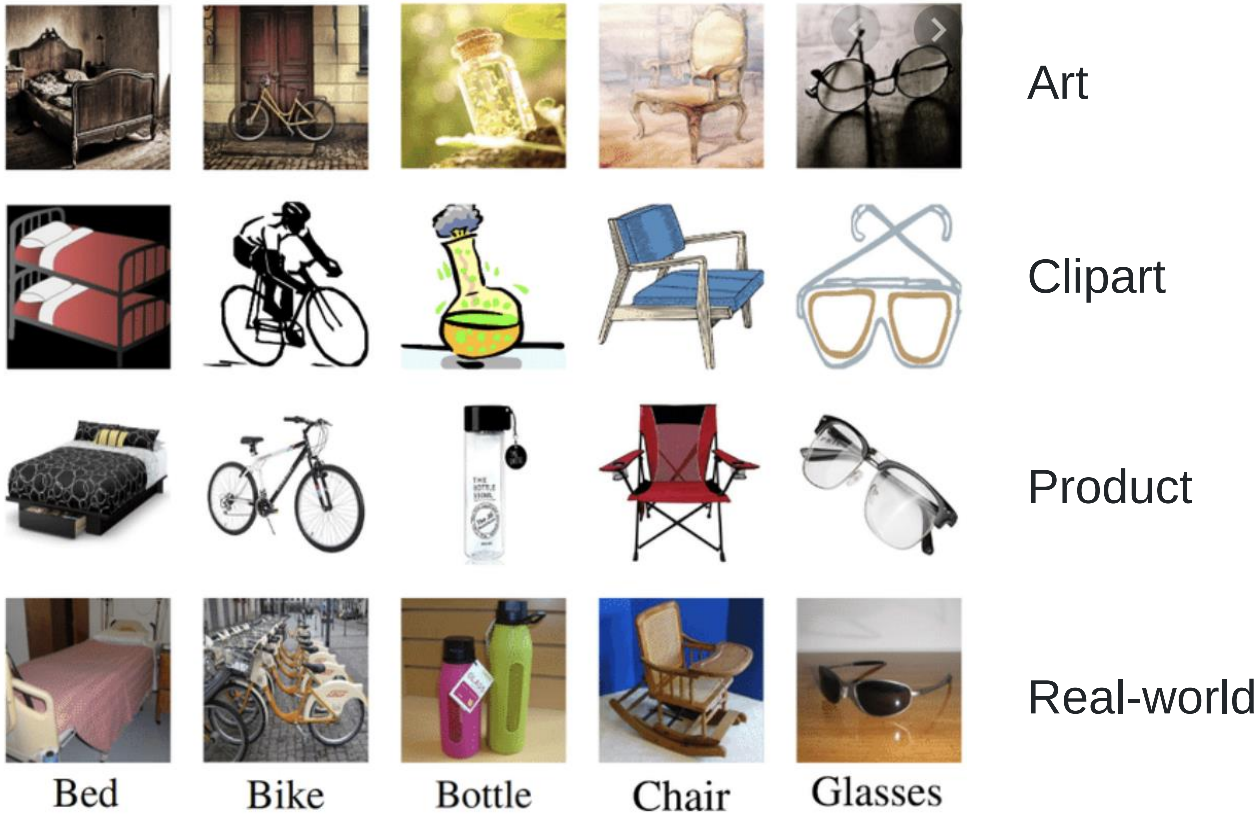
↓: diminishing in an optimal rate w.r.t. N; ↑ : growing w.r.t. N;

↘ : diminishing w.r.t. N, but not in an optimal rate.

A new algorithm that interpolates between static and dynamic weighting with **theory-guided hyperparameters** to balance the trade-off!

A new unified theoretical framework to analyze the three errors.

Application to multi-domain image classification



Office-home dataset

4 domains
65 classes/domain
70-100 images/class

Lisha Chen, Heshan Fernando, Yiming Ying, Tianyi Chen. ``Three-way trade-off in multi-objective learning: Optimization, generalization and conflict-avoidance,” arXiv, 2023.

Application to multi-domain image classification

Holistic performance metric

$$\Delta\mathcal{A}\% = \frac{1}{T}\sum_{t=1}^{T}(S_{\mathcal{B},t} - S_{\mathcal{A},t})/S_{\mathcal{B},t} \times 100$$

Table: classification results on Office-home dataset.

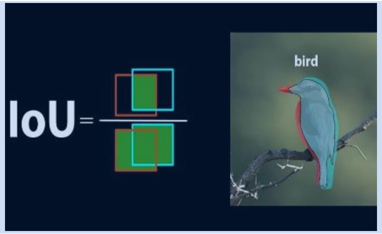
Method	Art	Clipart	Product	Real-world	$\Delta\mathcal{A}_{\text{st}}\% \downarrow$	$\Delta\mathcal{A}_{\text{id}}\% \downarrow$
Static (EW)	62.99	76.48	88.45	77.72	0.00	5.02
MGDA-UB (Lin et al., 2022a)	64.32	75.29	89.72	79.35	-1.02	4.04
GradNorm (Chen et al., 2018)	65.46	75.29	88.66	78.91	-1.03	4.04
PCGrad (Yu et al., 2020)	63.94	76.05	88.87	78.27	-0.53	4.51
CAGrad (Liu et al., 2021a)	63.75	75.94	89.08	78.27	-0.48	4.56
RGW (Lin et al., 2022a)	65.08	78.65	88.66	79.89	-2.30	2.85
MoCo (Fernando et al., 2023)	64.14	**79.85**	89.62	79.57	-2.48	2.68
MoDo (ours)	**66.22**	78.22	**89.83**	**80.32**	**-3.08**	**2.11**

Lisha Chen, Heshan Fernando, Yiming Ying, Tianyi Chen. ``Three-way trade-off in multi-objective learning: Optimization, generalization and conflict-avoidance,” arXiv, 2023.

Application to scene understanding

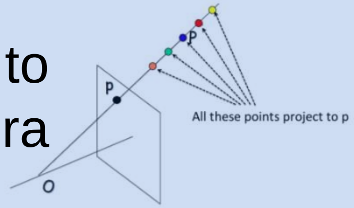
Object segmentation

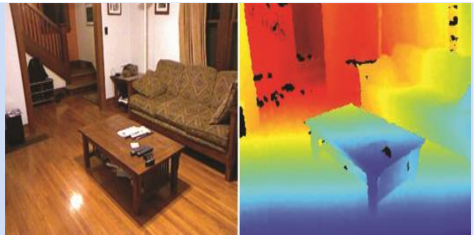
Classification of pixel



Depth regression

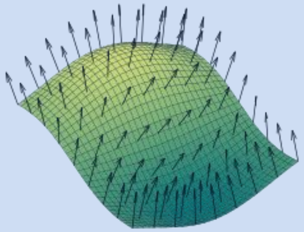
Distance to camera

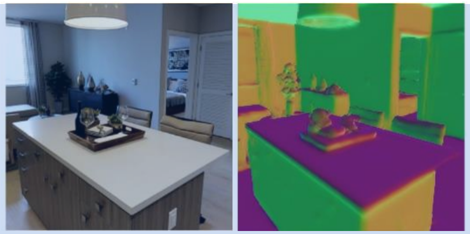




Surface normal estimation

Direction of surface normal





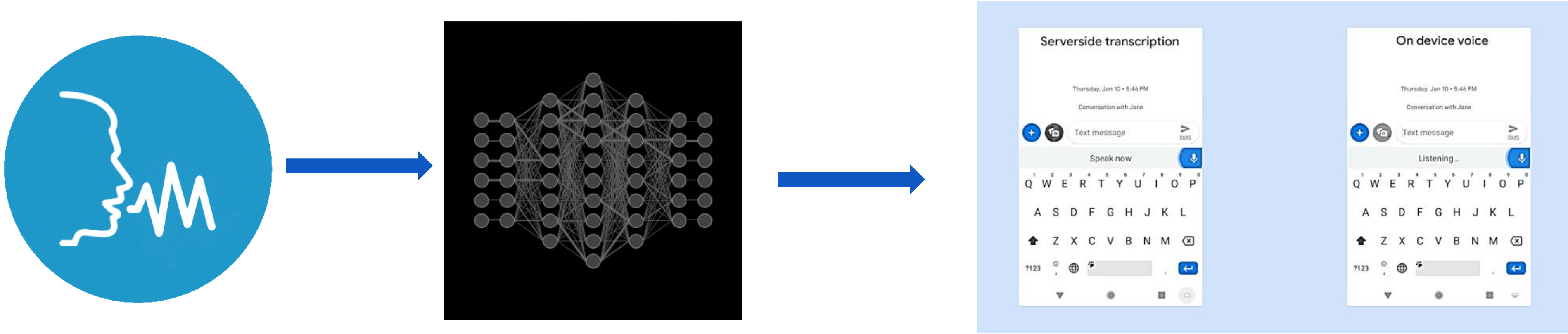
Application to scene understanding

Table 4: Segmentation, depth, and surface normal estimation results on NYU-v2 dataset.

Method	Segmentation (Higher Better)		Depth (Lower Better)		Surface Normal: Angle Distance (Lower Better)		Surface Normal: Within t° (Higher better)			$\Delta\mathcal{A}_{\text{stat}}\% \downarrow$	$\Delta\mathcal{A}_{\text{indep}}\% \downarrow$
	mIoU	Pix Acc	Abs Err	Rel Err	Mean	Median	11.25	22.5	30		
Static (EW)	53.77	75.45	0.3845	0.1605	23.5737	17.0438	35.04	60.93	72.07	0.00	1.63
MGDA-UB [38]	50.42	73.46	0.3834	0.1555	22.7827	16.1432	36.90	62.88	73.61	-0.38	1.26
GradNorm [5]	53.58	75.06	0.3931	0.1663	23.4360	16.9844	35.11	61.11	72.24	0.99	2.62
PCGrad [46]	53.70	75.41	0.3903	0.1607	23.4281	16.9699	35.16	61.19	72.28	0.16	1.79
CAGrad [27]	53.12	75.19	0.3871	0.1599	**22.5257**	**15.8821**	**37.42**	**63.50**	**74.17**	-1.36	0.26
RGW [23]	53.85	**75.87**	0.3772	0.1562	23.6725	17.2439	34.62	60.49	71.75	-0.62	1.03
MoCo [9]	**54.05**	75.58	0.3812	**0.1530**	23.3868	16.6938	35.65	61.68	72.60	-1.47	0.18
MoDo (ours)	53.37	75.25	**0.3739**	0.1531	23.2228	16.6489	35.62	61.84	72.76	**-1.59**	**0.07**

- MoDo with balanced tradeoff among three metrics outperforms MGDA and static weighting

Application to speech processing

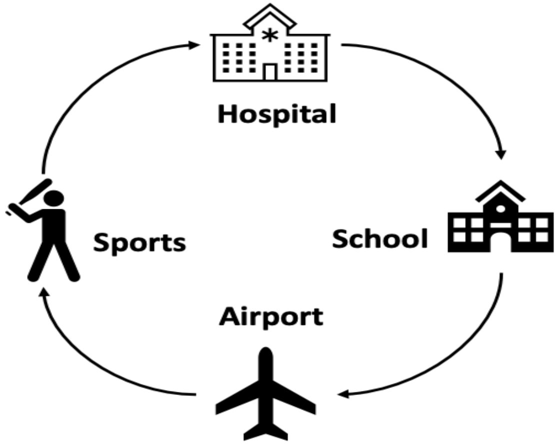


Speech **Model** **Transcripts**

- Pre-training with unlabeled data.
- Downstream fine-tuning.

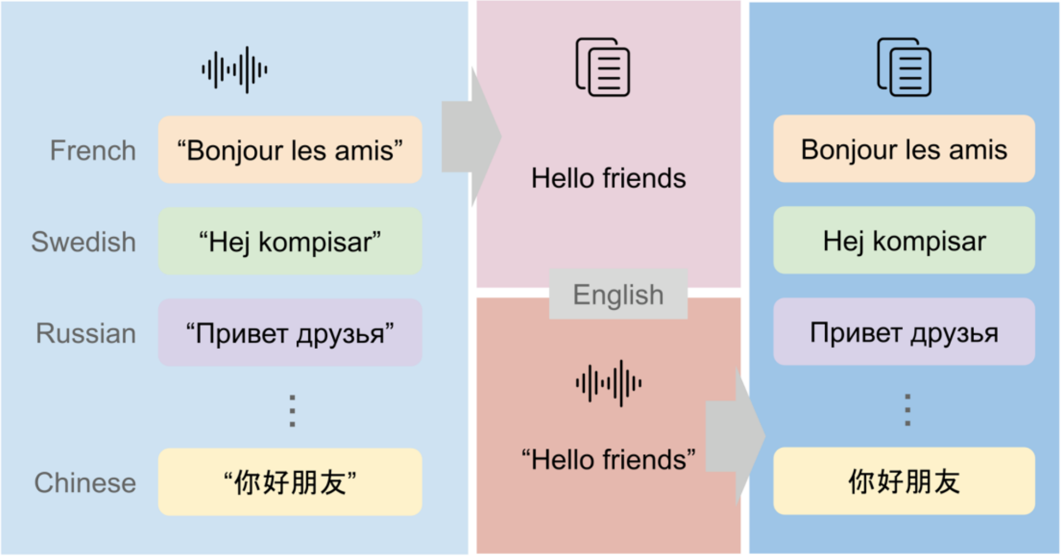
Multi-lingual, multitask with unified MOL

Over 7000 languages

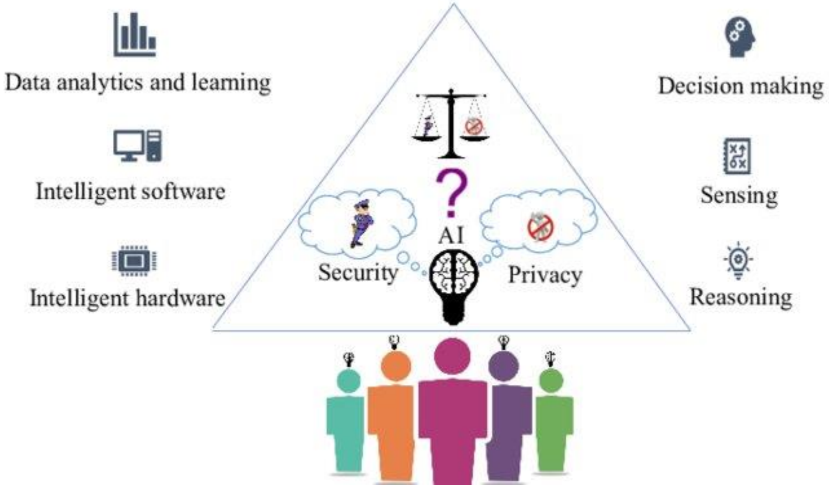


Domain-specific jargon

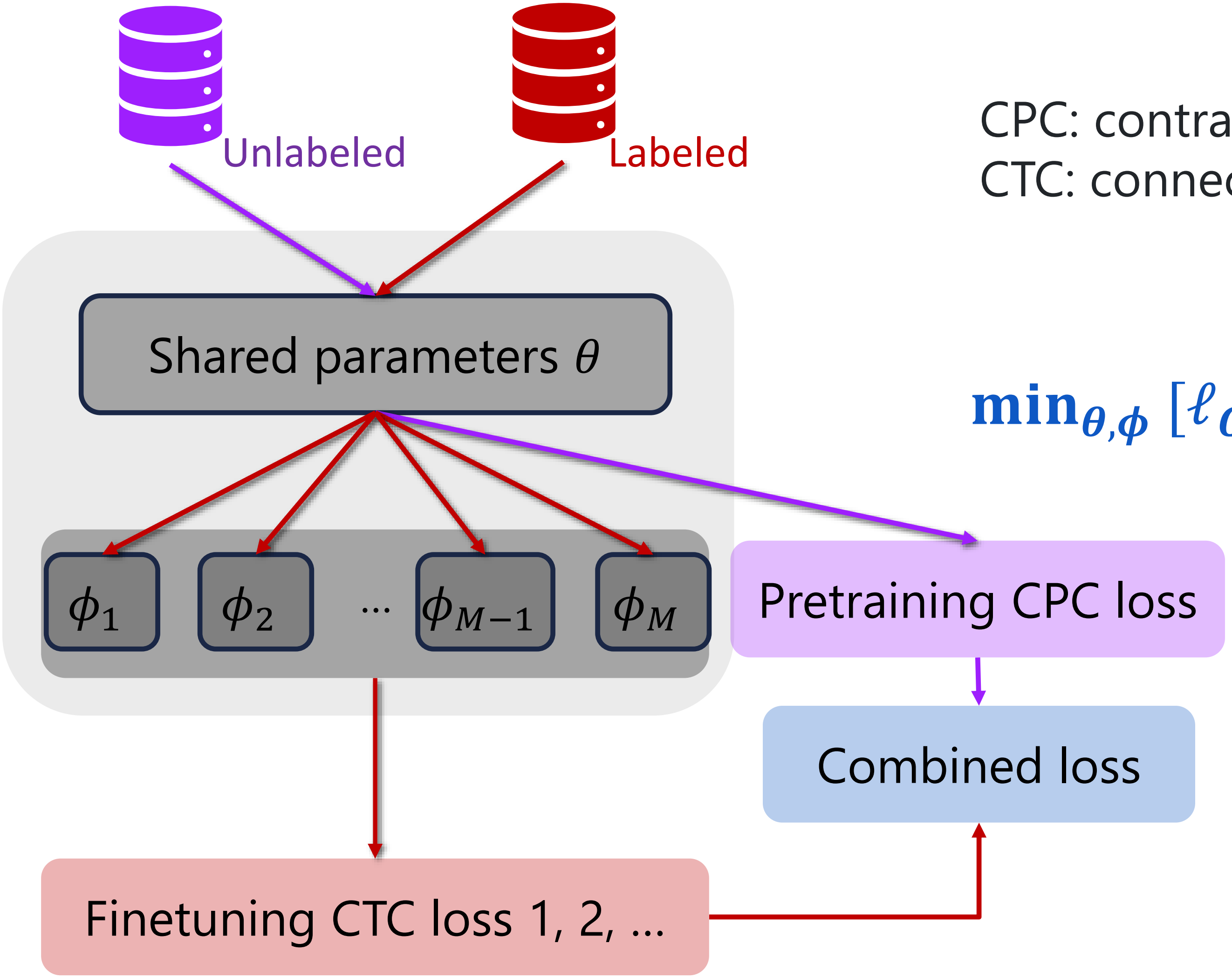
Universal language translator



Security and privacy of data



Joint pretraining & multi-lingual finetuning



CPC: contrastive predictive coding loss
CTC: connectionist temporal classification loss

$$\min_{\theta,\phi} [\ell_{CPC}(\theta), \ell_{CTC}(\theta, \phi_1), \ldots, \ell_{CTC}(\theta, \phi_M)]$$

Results on benchmarks

Metric: Word Error Rate (WER)

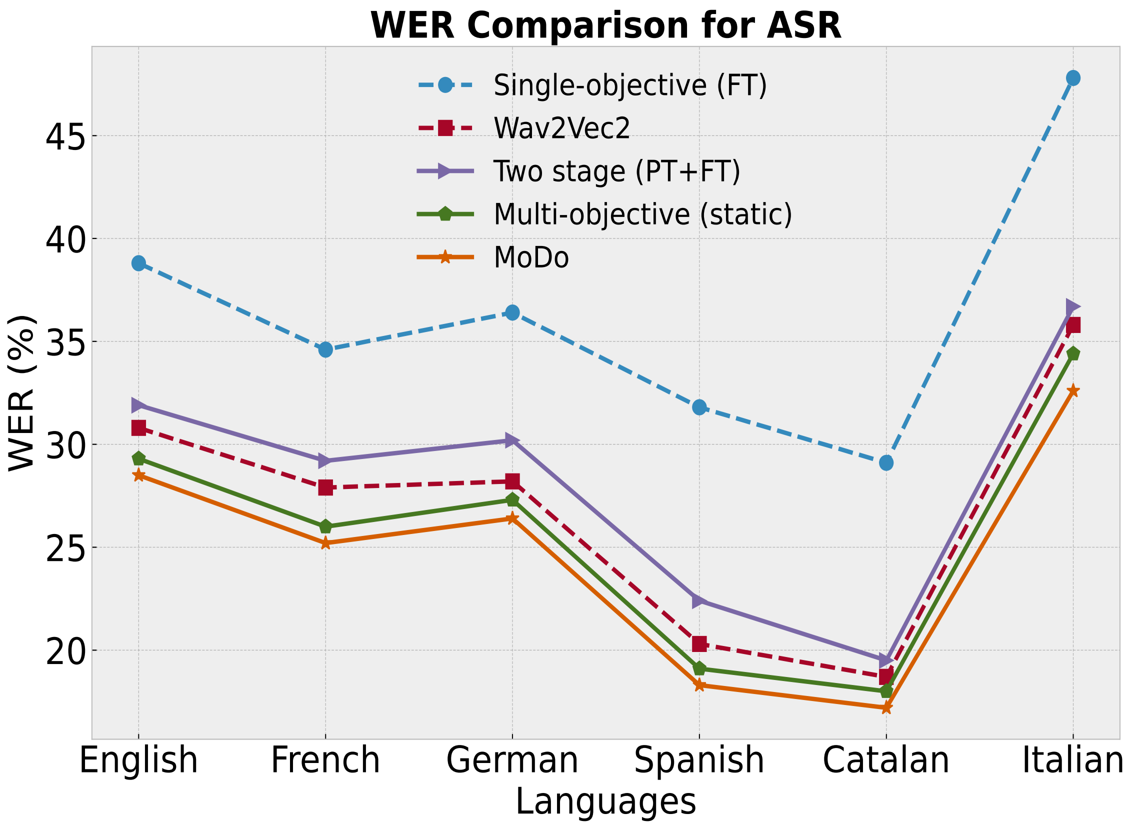
$$WER = \frac{I + D + S}{N} * 100\%$$

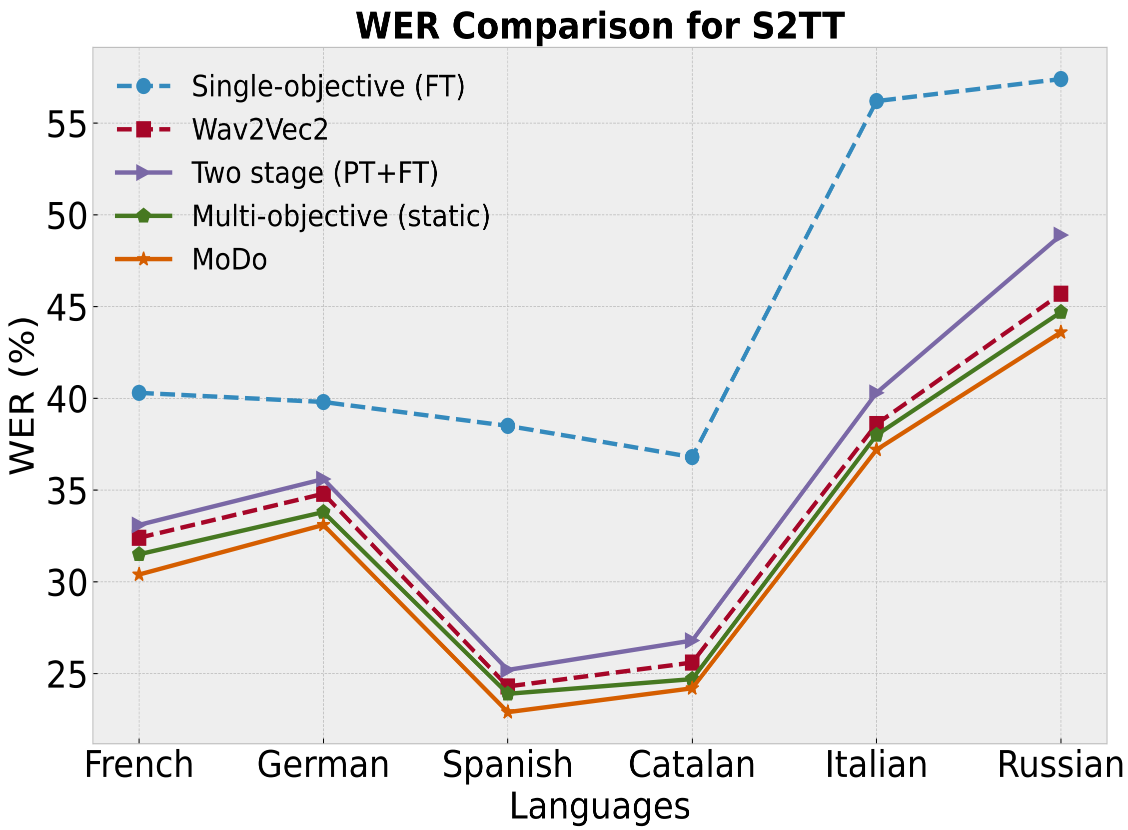
- Insertion (I): #words incorrectly added
- Deletion (D): #words undetected
- Substitution (S): #words substituted
- (N): Total #words in the labeled transcript

Baselines:

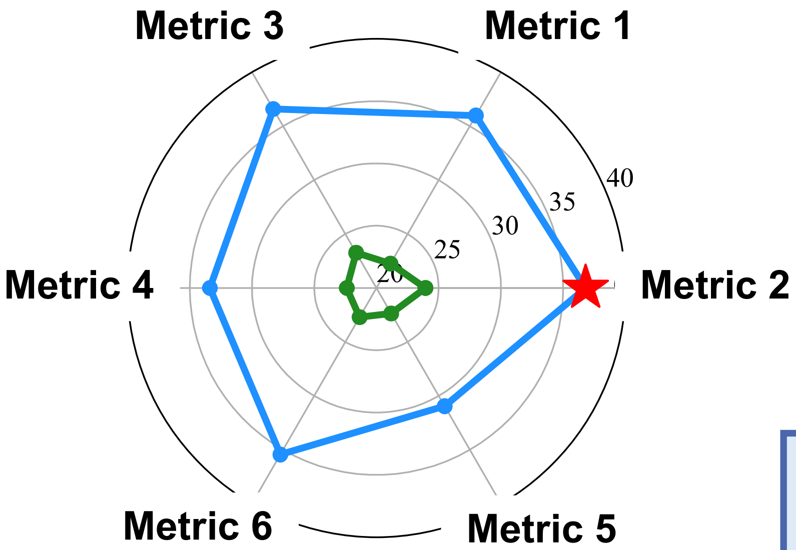
- Wac2Vec2: a SOTA model
- FT: Supervised baseline without pretraining
- Two stage (PT+FT): 2-stage pretraining & finetuning (without joint MOL)
- Multi-objective (static): without optimization conflict avoidant update

Results on benchmarks





Conclusions



Bi-/multi-level

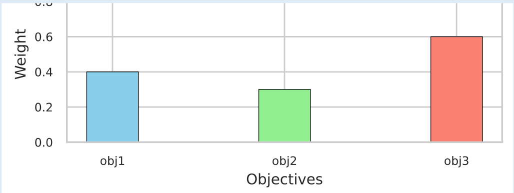
min objective 1

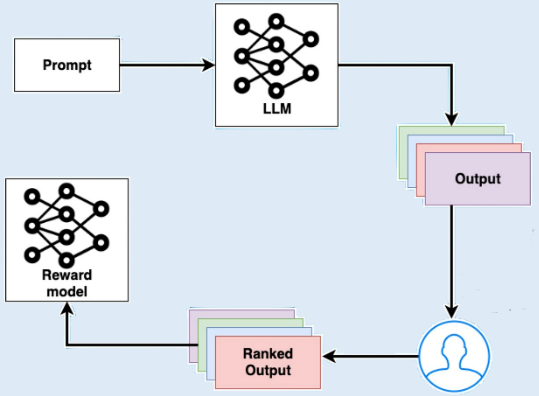
s.t. min objective 2

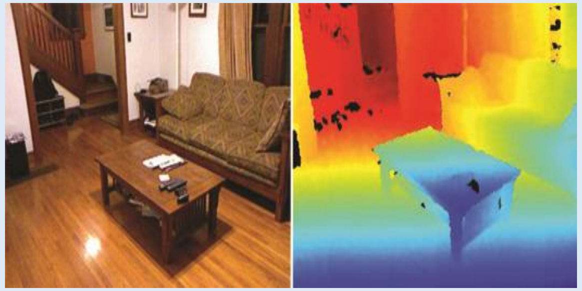
Theory foundation

Multi-objective

min (obj 1, obj 2, obj 3**)**







Take home

Multi-objective and multi-level optimization can flexibly model complex learning tasks and enable exciting applications in AI!

Tianyi Chen **Zhuoran Yang** **Lisha Chen**

THANK YOU!

References

Lisha Chen*, Heshan Fernando*, Yiming Ying, Tianyi Chen. ``Three-way trade-off in multi-objective learning: Optimization, generalization and conflict-avoidance,” conditionally accepted by JMLR, 2024.

Heshan Fernando, Lisha Chen, Songtao Lu, Pin-Yu Chen, Tianyi Chen, et al. ``Variance Reduction Can Improve Trade-off in Multi-Objective Learning,’’ IEEE Proc. ICASSP, 2024.

Lisha Chen*, Heshan Fernando*, Yiming Ying, Tianyi Chen. ``Three-way trade-off in multi-objective learning: Optimization, generalization and conflict-avoidance,” Proc. NeurIPS, 2023.

Jean-Antoine Désidéri, ``Multiple-gradient Descent Algorithm (MGDA) for Multi-objective Optimization’’. Comptes Rendus Mathematique, 350(5-6), 2012.

Suyun Liu and Luis Nunes Vicente, ``The stochastic multi-gradient algorithm for multi-objective optimization and its application to supervised machine learning. Annals of Operations Research’’, pp. 1–30, 2021.

Jörg Fliege, A Ismael F Vaz, and Luís Nunes Vicente. ``Complexity of Gradient Descent for Multiobjective Optimization.” Optimization Methods and Software, vol. 34, no. 5, pp. 949–959, 2019

Heshan Fernando, Han Shen, Miao Liu, Subhajit Chaudhury, Keerthiram Murugesan, and Tianyi Chen. ``Mitigating gradient bias in multi-objective learning: A provably convergent stochastic approach’’, Proc. ICLR 2023.

Peiyao Xiao, Hao Ban, Kaiyi Ji, ``Direction-oriented Multi-objective Learning: Simple and Provable Stochastic Algorithms’’, Proc. NeurIPS, 2023.

References

Yunwen Lei, ``Stability and Generalization of Stochastic Optimization with Nonconvex and Nonsmooth Problems,'' Proc. COLT, 2023

Peter Súkeník and Christoph H Lampert. ``Generalization in multi-objective machine learning''. arXiv preprint arXiv:2208.13499, 2022.

Moritz Hardt, Benjamin Recht, Yoram Singer, ``Train faster, generalize better: Stability of stochastic gradient descent'', Proc. ICML 2016.

Corinna Cortes, Mehryar Mohri, Javier Gonzalvo, and Dmitry Storcheus, ``Agnostic learning with multiple objectives'', Proc. NeurIPS, 2020.

Debabrata Mahapatra, Vaibhav Rajan ``Multi-Task Learning with User Preferences: Gradient Descent with Controlled Ascent in Pareto Optimization,'' Proc. ICML, 2020.

Xi Lin, Hui-Ling Zhen, Zhenhua Li, Qingfu Zhang, Sam Kwong, ``Pareto Multi-Task Learning,'' Proc. NeurIPS, 2019.

Yuzheng Hu, Ruicheng Xian, Qilong Wu, Qiuling Fan, Lang Yin, Han Zhao, ``Revisiting scalarization in multi-task learning: A theoretical perspective,'' Proc. NeurIPS, 2023.

Aviv Navon, Aviv Shamsian, Idan Achituve, Haggai Maron, Kenji Kawaguchi, Gal Chechik, Ethan Fetaya, ``Multi-Task Learning as a Bargaining Game,'' Proc. ICML 2022.

References

Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, Chelsea Finn, ``Gradient Surgery for Multi-Task Learning,'' Proc. NeurIPS 2020.

Bo Liu, Xingchao Liu, Xiaojie Jin, Peter Stone, Qiang Liu, ``Conflict-Averse Gradient Descent for Multi-task Learning,'' Proc. NeurIPS 2021.

Xi Lin, Hui-Ling Zhen, Zhenhua Li, Qingfu Zhang, Sam Kwong ``Pareto Multi-Task Learning,'' Proc. NeurIPS, 2019.

Debabrata Mahapatra, Vaibhav Rajan ``Multi-Task Learning with User Preferences: Gradient Descent with Controlled Ascent in Pareto Optimization,'' Proc. ICML 2020.

Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. ``Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks," Proc. ICML 2018.

Zhao Chen, Jiquan Ngiam, Yanping Huang, Thang Luong, Henrik Kretzschmar, Yuning Chai, and Dragomir Anguelov, ``Just pick a sign: Optimizing deep multitask models with gradient sign dropout,'' arXiv preprint arXiv:2010.06808, 2020.

Alex Kendall, Yarin Gal, and Roberto Cipolla, ``Multi-task learning using uncertainty to weigh losses for scene geometry and semantics,'' Proc. CVPR 2018.

Michelle Guo, Albert Haque, De-An Huang, Serena Yeung, and Li Fei-Fei, ``Dynamic task prioritization for multitask learning,'' Proc. ECCV 2018.

References

Liyang Liu, Yi Li, Zhanghui Kuang, Jing-Hao Xue, Yimin Chen, Wenming Yang, Qingmin Liao, and Wayne Zhang, ``Towards impartial multitask learning,'' Proc. ICLR, 2021.

Baijiong Lin, Feiyang Ye, Yu Zhang, Ivor Tsang, ``Reasonable Effectiveness of Random Weighting: A Litmus Test for Multi-Task Learning,'' Transactions on Machine Learning Research, 2022.

Yuzheng Hu, Ruicheng Xian, Qilong Wu, Qiuling Fan, Lang Yin, Han Zhao, ``Revisiting scalarization in multi-task learning: A theoretical perspective,'' Proc. NeurIPS, 2023.

Michinari Momma, Chaosheng Dong, Jia Liu, ``A Multi-objective / Multi-task Learning Framework Induced by Pareto Stationarity,'' Proc. ICML, 2022.

Sener Ozan and Koltun Vladlen, ``Multi-task learning as multiobjective optimization," Proc. NeurIPS, 2018.

Baijiong Lin, Yu Zhang, ``LibMTL: A Python Library for Multi-Task Learning,'' Journal of Machine Learning Research (JMLR)

Haibo Yang, Zhuqing Liu, Jia Liu, Chaosheng Dong, Michinari Momma, ``Federated Multi-Objective Learning,'' Proc. NeurIPS, 2023.

Shiji Zhou, Wenpeng Zhang, Jiyan Jiang, Wenliang Zhong, Jinjie Gu, Wenwu Zhu, ``On the convergence of stochastic multi-objective gradient manipulation and beyond,'' Proc. NeurIPS 2022.