Bi-level Optimization and Continuous Bi-level Optimization Problems in Machine Learning

Combinatorial Optimization and Machine Learning | Lecture 3

Sanjeeb Dash / Parikshit Ram

June 26, 2023

Plan

1. Single-level Optimization in ML
   - Unconstrained
   - Constrained
2. Bi-level Optimization
   - Formulation & Terminology
   - Example
3. Bi-level Optimization in ML
   - Bi-level Reformulations
   - Inherently Bi-level Problems
4. Challenges with Bi-level Optimization

Outline

Single-level Optimization Problem

$$\min_{\theta \in \mathbb{R}^d} f(\theta) \tag{1}$$

subject to

$$P(\theta) \geq 0 \tag{2}$$

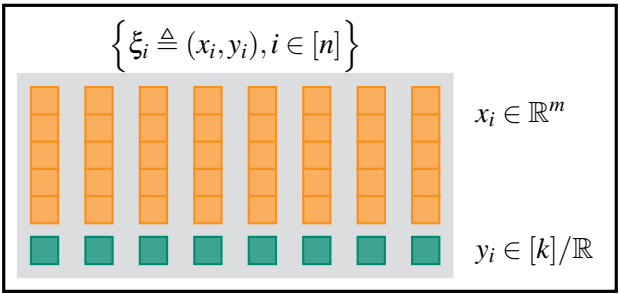
$$Q(\theta) = 0 \tag{3}$$

Terminology

- $\theta \in \mathbb{R}^d$ are the d decision variables
- $f : \mathbb{R}^d \to \mathbb{R}$ is the objective
- $P : \mathbb{R}^d \to \mathbb{R}^p$ are the p inequality constraints
- $Q : \mathbb{R}^d \to \mathbb{R}^q$ are the q equality constraints
- $\Omega_\theta \triangleq \{\theta \in \mathbb{R}^d : P(\theta) \geq 0, Q(\theta) = 0\}$ is the constraint set

Elements defining objectives/constraints in ML

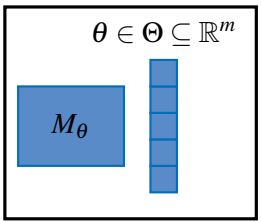
- **Data** – $\{\xi_i, i \in [n]\}$, $[n] \triangleq \{1, \ldots, n\}$
  - Samples $\xi_i \sim \mu$, for a data distribution μ
  - $\xi_i \triangleq (x_i, y_i) \in (\mathbb{R}^m \times [k])$ for k-class classification
  - $\xi_i \triangleq (x_i, y_i) \in (\mathbb{R}^m \times \mathbb{R})$ for regression
  - $\xi_i \in \mathbb{R}^m$ for unsupervised learning

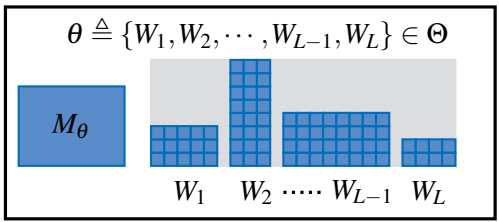


Optimization in Machine Learning

Elements defining objectives/constraints in ML

- ⇛ **Data** – $\{\xi_i, i \in [n]\}$
- ⇛ **Model** M_θ – Defines the decision variables θ
  - ⇛ Linear models: θ are the weights of the linear model
  - ⇛ Neural networks: θ are the weights of all the layers in the neural network



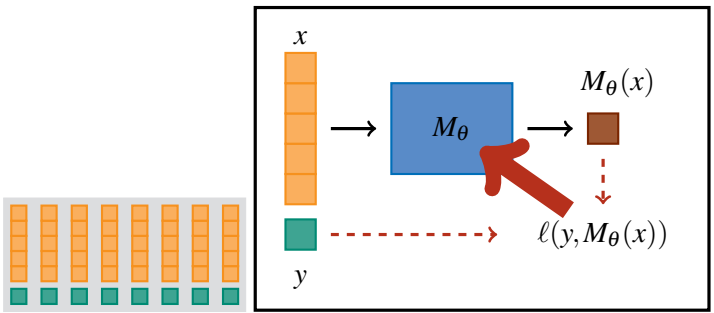


Optimization in Machine Learning

- **Data** – $\{\xi_i, i \in [n]\}$
- **Model** M_θ – Defines the decision variables θ
- **Loss function** ℓ
  - Mis-classification loss in k-class classification → $-\sum_{l \in [k]} \mathbb{I}(y = l) \log M_\theta(x)[l]$
  - Distortion loss in regression → $(y - M_\theta(x))^2$
  - (Negative) Likelihood in density estimation → $-\log M_\theta(x)$

Optimization in Machine Learning

- **Data** – $\{\xi_i, i \in [n]\}$
- **Model** M_θ – Defines the decision variables θ
- **Loss function** ℓ



Common Optimization in Machine Learning

Optimization type:

⇛ Many continuous, unconstrained

$$\min_{\boldsymbol{\theta} \in \mathbb{R}^d} f(\boldsymbol{\theta}) \tag{4}$$

Linear Regression via Empirical Risk Minimization (ERM)

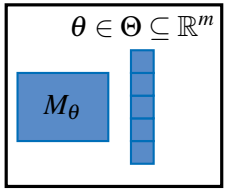
- ⇛ Data → $\{\xi_i, i \in [n]\}$, with $\xi_i \triangleq (x_i, y_i), x_i \in \mathbb{R}^m$, $y_i \in \mathbb{R}$
- ⇛ Model → $M_\theta(x) \triangleq \theta^\top x, \theta \in \mathbb{R}^d$ and $d = m$
- ⇛ Loss between true y and predicted y' → $\ell(y, y') \triangleq (y - y')^2$

Final objective

$$f(\theta) \triangleq \frac{1}{n} \sum_{i \in [n]} \ell(y_i, \theta^\top x_i) + \rho \|\theta\|_2^2. \qquad (5)$$

$$\min_{\theta \in \mathbb{R}^d} f(\theta) \qquad (6)$$

- ⇛ Continuous
- ⇛ Unconstrained



Neural Network via Empirical Risk Minimization (ERM)

- Data → $\{\xi_i, i \in [n]\}$, with $\xi_i \triangleq (x_i, y_i), x_i \in \mathbb{R}^m$, $y_i \in \mathbb{R}$
- Model → $M_\theta : \mathbb{R}^m \to \mathbb{R}, \theta \in \mathbb{R}^d$, d is the number of (learnable) parameters in the network
- Loss between true y and predicted y' → $\ell(y, y') \triangleq (y - y')^2$

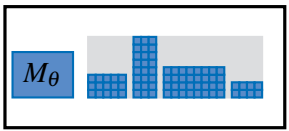
Final objective

$$f(\theta) \triangleq \frac{1}{n} \sum_{i \in [n]} \ell(y_i, M_\theta(x)). \tag{7}$$

$$M_\theta(x) \triangleq W_L \cdot \sigma(W_{L-1} \cdot \sigma(\cdots W_2 \cdot \sigma(W_1 x) \cdots)),$$
$$\theta \triangleq [W_1, W_2, \ldots, W_{L-1}, W_L] \tag{8}$$

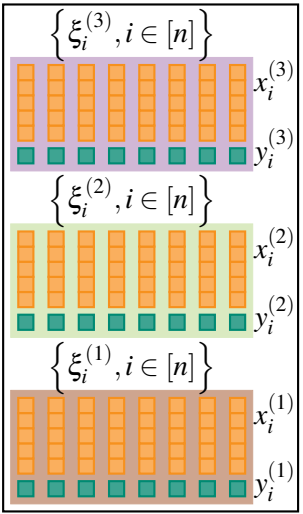
$$\min_{\theta \in \mathbb{R}^d} f(\theta) \tag{9}$$

- Continuous
- Unconstrained
- Nonlinear because of activation σ
- Sigmoid $\sigma(a) = 1/1+e^{-a}$
- Tanh $\sigma(a) = e^{a}-e^{-a}/e^{a}+e^{-a}$
- ReLU $\sigma(a) = \max\{a, 0\}$



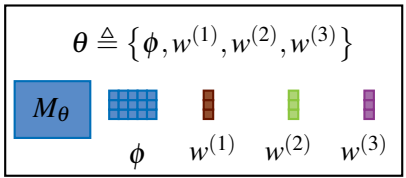
Multi-task Linear Regression via ERM

⇛ Data for T (related) "tasks" $\rightarrow$
$\{\xi_i^{(t)} \triangleq (x_i^{(t)}, y_i^{(t)}) \in (\mathbb{R}^m \times \mathbb{R}), i \in [n^{(t)}]\}$
for each $t \in [T]$



Multi-task Linear Regression via ERM

- Data for T (related) "tasks" $\rightarrow$ $\{\xi_i^{(t)} \triangleq (x_i^{(t)}, y_i^{(t)}) \in (\mathbb{R}^m \times \mathbb{R}), i \in [n^{(t)}]\}$ for each $t \in [T]$
- Model $\rightarrow M_\theta(x,t) \triangleq {w^{(t)}}^\top \phi \cdot x$
  - $\theta \triangleq [\phi, \{w^t, t \in [T]\}]$
  - Shared $\rightarrow \phi \in \mathbb{R}^{r \times m}$
  - Per-task $\rightarrow w^{(t)} \in \mathbb{R}^r \ \forall t \in [T]$
  - $d = (m+T)r$



Multi-task Linear Regression via ERM

- ⇛ Data for T (related) "tasks" $\rightarrow \{\xi_i^{(t)} \triangleq (x_i^{(t)}, y_i^{(t)}) \in (\mathbb{R}^m \times \mathbb{R}), i \in [n^{(t)}]\}$ for each $t \in [T]$
- ⇛ Model $\rightarrow M_\theta(x,t) \triangleq {w^{(t)}}^\top \phi \cdot x$, $\theta \triangleq [\phi, \{w^t, t \in [T]\}]$
- ⇛ Loss between true y and predicted $y' \rightarrow \ell(y,y') \triangleq (y-y')^2$

Final objective

$$f(\theta) \triangleq \sum_{t\in[T]} \frac{1}{n^{(t)}} \sum_{i\in[n^{(t)}]} \ell(y_i^{(t)}, {w^{(t)}}^\top \phi \cdot x_i^{(t)}) + \rho \sum_{t\in[T]} \|w^{(t)}\|_2^2 + \rho' \|\phi\|_F^2. \tag{10}$$

Multi-task Neural Network via ERM

- ⇛ Data for T (related) "tasks" → $\{\xi_i^{(t)} \triangleq (x_i^{(t)}, y_i^{(t)}) \in (\mathbb{R}^m \times \mathbb{R}), i \in [n^{(t)}]\}$ for each $t \in [T]$
- ⇛ Model → $M_\theta(x,t) \triangleq M_{\omega^{(t)}}(M_\phi(x))$
  - ⇛ $\theta \triangleq \left[\phi, \{\omega^{(t)}, t \in [T]\}\right]$
  - ⇛ Shared $\phi \in \mathbb{R}^{d_1}$, $M_\phi : \mathbb{R}^m \to \mathbb{R}^r$
  - ⇛ Per-task $\omega^{(t)} \in \mathbb{R}^{d_2}$ $\forall t \in [T]$, $M_{\omega^{(t)}} : \mathbb{R}^r \to \mathbb{R}$
  - ⇛ $d = d_1 + d_2$
- ⇛ Loss between true y and predicted y' → $\ell(y,y') \triangleq (y-y')^2$

Final objective

$$f(\theta) \triangleq \sum_{t\in[T]} \frac{1}{n^{(t)}} \sum_{i\in[n^{(t)}]} \ell\left(y_i^{(t)}, M_{\omega^{(t)}}\left(M_\phi\left(x_i^{(t)}\right)\right)\right). \tag{11}$$

$$M_\phi(x) \triangleq \sigma(W_L \cdot \sigma(\cdots W_2 \cdot \sigma(W_1 x)\cdots)), \quad \phi \triangleq [W_1, W_2, \ldots, W_{L-1}, W_L]$$

$$M_{\omega^{(t)}}(\mathrm{x}) \triangleq \sigma(w_l^{(t)} \cdot \sigma(w_{L-1}^{(t)} \cdot \sigma(\cdots w_2^{(t)} \cdot \sigma(w_1^{(t)}\mathrm{x})\cdots))), \quad \omega^{(t)} \triangleq [w_1^{(t)}, \ldots, w_l^{(t)}] \tag{12}$$

usually $l \ll L$

Solution with Gradient Descent

Gradient based continuous optimization algorithm:

$$\min_{\theta \in \mathbb{R}^d} f(\theta) \tag{13}$$

Algorithm 1 Gradient descent

Input: Initialization θ^0, initial learning rate α^0

for $k = 1, 2, \cdots, K$ **do**

  $\theta^{k+1} \leftarrow \theta^k - \alpha^k \cdot h^k$ // example $h^k \triangleq \nabla_\theta f(\theta)|_{\theta=\theta^k}$

  Learning rate update $\alpha^k \to \alpha^{k+1}$ // example $\alpha^k \propto \alpha^0/c^k$

return θ^{K+1}

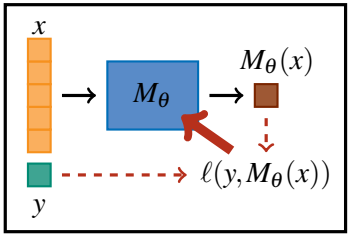
Stochastic Gradient Descent

Stochastic objective:

$$f(\theta) \triangleq \mathbb{E}_{\xi\sim\mu}\mathsf{f}(\theta;\xi) \approx \frac{1}{n}\sum_{i\in[n]}\mathsf{f}(\theta;\xi_i) \qquad (14)$$

Stochastic gradient estimate:

$$h^k \leftarrow \nabla_\theta \mathsf{f}(\theta;\xi^k)\big|_{\theta=\theta^k}$$
$$\triangleq \frac{1}{|B^k|}\sum_{i\in B^k\subset[n]}\nabla_\theta \ell(y_i, M_\theta(x_i))\Big|_{\theta=\theta^k} \qquad (15)$$



$$\theta^0 \xrightarrow{h^0} \theta^1 \xrightarrow{h^1} \theta^2 \dashrightarrow \theta^k \xrightarrow{h^k} \theta^{k+1} \dashrightarrow \theta^K$$

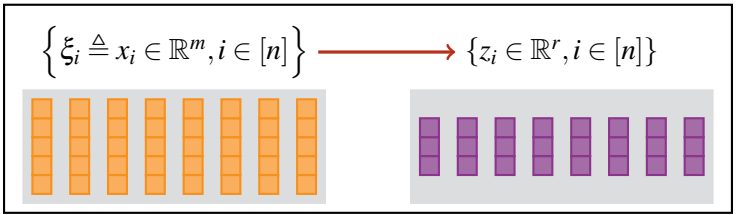
Common Constrained Optimization in Machine Learning

$$\min_{\theta \in \Theta \subset \mathbb{R}^d} f(\theta) \tag{16}$$

Common constraint types:

- ⇛ Rank constraints
- ⇛ Integrality constraints
- ⇛ Simplex constraints
- ⇛ Orthogonality constraints

Principal Component Analysis



- ⇛ Learn a projection matrix $V \in \mathbb{R}^{m \times r}$
- ⇛ Minimize distortion ($\equiv$ preserve variance) with smaller number of features
- ⇛ Orthogonality constraint on $V - V^\top V = I$

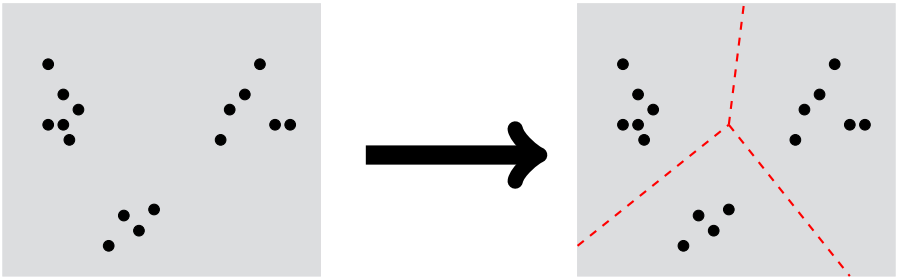
$$\min_{\theta} f(\theta) \triangleq \min_{\substack{V \in \mathbb{R}^{m \times r}, \\ \{z_i \in \mathbb{R}^r, i \in [n]\}}} \|x_i - V z_i\|_2^2 \quad (17)$$

subject to

$$V^\top V = I_r \quad (18)$$

$$\theta \triangleq [V, \{z_i, i \in [n]\}] \quad (19)$$

Discrete Clustering



Discrete Clustering

Clustering data $\{x_i, i \in [n]\}$ for a given pairwise affinity matrix $A \in \mathbb{R}^{n \times n}$:

$$\min_{\theta} f(\theta) \triangleq \min_{c_i \in \{0,1\}^k, i \in [n]} \sum_{i,j \in [n]} A_{ij} \|c_i - c_j\|_2^2 \quad (20)$$

⇛ Learn cluster assignments

$$\texttt{subject to}$$

⇛ Integrality cst

$$\mathbf{1}_k^\top c_i = 1 \forall i \in [n] \quad (21)$$

⇛ Simplex cst

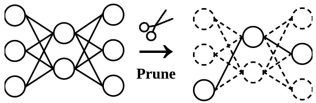
$$\texttt{rank}(C) = k, C \triangleq [c_i, i \in [n]] \in \{0,1\}^{k \times n} \quad (22)$$

⇛ Rank cst

Special cases

- ⇛ Euclidean clustering: $A_{ij} = -\|x_i - x_j\|_2^2$
- ⇛ Kernel clustering: $A_{ij} = \kappa(x_i, x_j)$ for some kernel function $\kappa : \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R}_{\geq 0}$
- ⇛ Spectral/graph clustering: $A_{ij} = \mathbb{I}((i,j) \in E)$ – that is, edge (i,j) in the graph $G = (V,E)$ with vertex set $V = [n]$ and edge set E

Neural Network Pruning & Compression



$$\min_{\theta \in \Theta} f(\theta) \triangleq \min_{\substack{m \in \{0,1\}^d, \\ \phi \in \mathbb{R}^d}} \sum_{i \in [n]} \ell(y_i, M_{m \odot \phi}(x_i)) \qquad (23)$$

subject to

$$\mathbf{1}_d^\top m = \alpha d \qquad (24)$$

$$\theta \triangleq [m, \phi] \in \mathbb{R}^{2d} \qquad (25)$$

Model Pruning

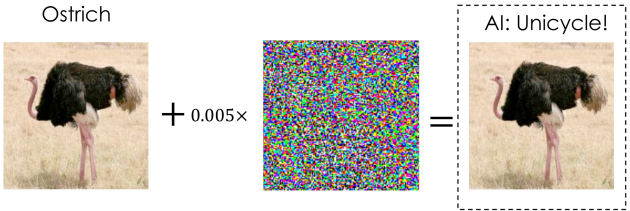
- ⇛ Remove redundant weights
- ⇛ Compress model
- ⇛ Speed up inference
- ⇛ Compression factor $\alpha \leq 0.1$

Constraints

- ⇛ Integrality cst
- ⇛ Simplex cst

Adversarial Attacks [Szegedy et al., 2013]

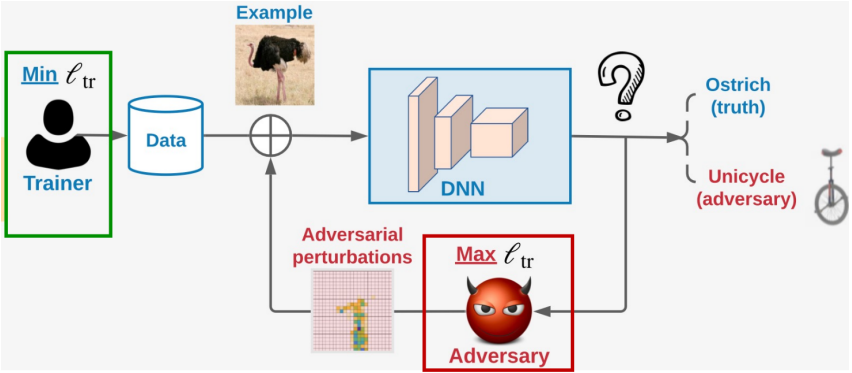
Problem. Neural networks (trained to high accuracy) can be easily “attacked” – *Models manipulated to make desired prediction with imperceptible perturbation*



Solution. Train models that are “robust” to adversarial perturbations

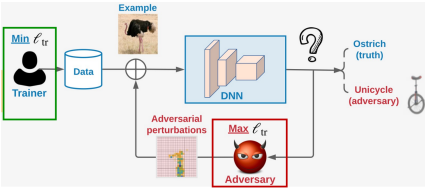
Adversarially Robust Learning

Problem. Neural networks (trained to high accuracy) can be easily “attacked” – *Models manipulated to make desired prediction with imperceptible perturbation*
Solution. Train models that are “robust” to adversarial perturbations



Adversarially Robust Learning

Solution. Train models that are "robust" to adversarial perturbations



$$\min_{\phi} \max_{\delta_i, \|\delta_i\|_\infty \leq \varepsilon, i \in [n]} \sum_{i \in [n]} \ell(y_i, M_\phi(x_i + \delta_i)) \tag{26}$$

$$\theta \triangleq [\phi, \{\delta_i, i \in [n]\}] \tag{27}$$

Gradient Descent with Constraints

Handling constraints with gradient-based continuous optimization algorithm:

$$\min_{\theta \in \Theta \subset \mathbb{R}^d} f(\theta) \tag{28}$$

Algorithm 2 Projected gradient descent

Input: Initialization θ^0, initial learning rate α^0

for $k = 1, 2, \cdots, K$ **do**

  $\theta^{k+1} \leftarrow \mathscr{P}_\Theta\left(\theta^k - \alpha^k \cdot h^k\right)$ `// closed-form projection`

  Learning rate update $\alpha^k \to \alpha^{k+1}$

return θ^{K+1}

$$\mathscr{P}_\Theta(\theta) \in \arg\min_{\vartheta \in \Theta} \|\theta - \vartheta\| \tag{29}$$

$$\theta^0 \xrightarrow{h^0} \tilde{\theta}^1 \xrightarrow{\mathscr{P}_\Theta(\cdot)} \theta^1 \xrightarrow{h^1} \tilde{\theta}^2 \xrightarrow{\mathscr{P}_\Theta(\cdot)} \theta^2 \dashrightarrow \theta^{k-1} \xrightarrow{h^k} \tilde{\theta}^k \xrightarrow{\mathscr{P}_\Theta(\cdot)} \theta^k \dashrightarrow \theta^K$$

Outline

Bi-level Formulation

$$\min_{\theta \in \Theta \subseteq \mathbb{R}^{d_u}, \phi} f_u(\theta, \phi) \tag{30}$$

$$\texttt{subject to} \quad P_u(\theta, \phi) \geq 0 \tag{31}$$

$$\phi \in S(\theta) \triangleq \arg \min_{\phi \in \Phi \subseteq \mathbb{R}^{d_l}} f_l(\theta, \phi) \tag{32}$$

$$\texttt{subject to} \quad P_l(\theta, \phi) \geq 0 \tag{33}$$

Terminology

- $\Rightarrow$ $\min_{\theta,\phi} f_u(\theta, \phi)$ is called the **upper level** (UL) problem or the leader's problem
- $\Rightarrow$ $\min_{\phi} f_l(\theta, \phi)$ is called the **lower level** (LL) problem or the follower's problem

Bi-level Formulation

$$\min_{\theta \in \Theta \subseteq \mathbb{R}^{d_u}, \phi} f_u(\theta, \phi)$$

$$\texttt{subject to} \quad P_u(\theta, \phi) \geq 0$$

$$\phi \in S(\theta) \triangleq \arg \min_{\phi \in \Phi \subseteq \mathbb{R}^{d_l}} f_l(\theta, \phi)$$

$$\texttt{subject to} \quad P_l(\theta, \phi) \geq 0$$

Terminology

- $f_u : \mathbb{R}^{d_u} \times \mathbb{R}^{d_l}$ is the UL objective
- $f_l : \mathbb{R}^{d_u} \times \mathbb{R}^{d_l}$ is the LL objective
- $\theta \in \Theta \subseteq \mathbb{R}^{d_u}$ are the UL decision variables
- $\phi \in \Phi \subseteq \mathbb{R}^{d_l}$ are the LL decision variables

Bi-level Formulation

$$\min_{\theta \in \Theta \subseteq \mathbb{R}^{d_u}, \phi} f_u(\theta, \phi)$$

$$\texttt{subject to} \quad P_u(\theta, \phi) \geq 0$$

$$\phi \in S(\theta) \triangleq \arg \min_{\phi \in \Phi \subseteq \mathbb{R}^{d_l}} f_l(\theta, \phi)$$

$$\texttt{subject to} \quad P_l(\theta, \phi) \geq 0$$

Terminology

- $P_u : \mathbb{R}^{d_u} \times \mathbb{R}^{d_l} \to \mathbb{R}^{p_u}$ are the UL inequality constraints
- $P_l : \mathbb{R}^{d_u} \times \mathbb{R}^{d_l} \to \mathbb{R}^{p_l}$ are the LL inequality constraints
- $S : \mathbb{R}^{d_u} \to \mathscr{S} \subset \mathbb{R}^{d_l}$ is a point-to-set mapping of the LL optimal solutions

Bi-level Formulation

$$\min_{\theta \in \Theta \subseteq \mathbb{R}^{d_u}, \phi} f_u(\theta, \phi)$$

$$\texttt{subject to} \quad P_u(\theta, \phi) \geq 0$$

$$\phi \in S(\theta) \triangleq \arg \min_{\phi \in \Phi \subseteq \mathbb{R}^{d_l}} f_l(\theta, \phi)$$

$$\texttt{subject to} \quad P_l(\theta, \phi) \geq 0$$

Terminology

- $\Omega \triangleq \{(\theta, \phi) \in \Theta \times \Phi : P_u(\theta, \phi) \geq 0, P_l(\theta, \phi) \geq 0\}$ is the shared constraint set
- $\Omega_\Theta \triangleq \{\theta \in \Theta : \exists \phi \in \Phi, (\theta, \phi) \in \Omega\}$ is the projection of the shared constraint set onto the θ-space
- Coupling constraints: UL constraints P_u that explicitly depend on the LL variable ϕ
- Linking variables: UL variables θ that explicitly appear in the LL constraints P_l

Alternate Bi-level Formulation

$$\min_{\substack{\theta \in \Theta \subseteq \mathbb{R}^{d_u}, \\ \phi \in \Phi \subseteq \mathbb{R}^{d_l}}} f_u(\theta, \phi) \tag{34}$$

subject to

$$P_u(\theta, \phi) \geq 0 \tag{35}$$

$$P_l(\theta, \phi) \geq 0 \tag{36}$$

$$f_l(\theta, \phi) \leq v(\theta) \tag{37}$$

$$v(\theta) \triangleq \min_{\phi \in \Phi} \{ f_l(\theta, \phi) : P_l(\theta, \phi) \geq 0 \} \tag{38}$$

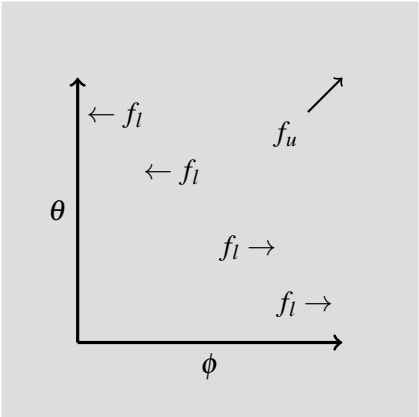
Terminology

$\Rightarrow$ $v : \mathbb{R}^{d_u} \to \mathbb{R}$ is called the Optimal Value Function

Example: Classical Pricing Problem

- Leader/UL
  - Decision variables: Price(s) of certain good(s)
  - Objective: Maximize revenue from selling goods
- Follower/LL
  - Decision variables: Amount spent on purchasing good(s)
  - Objective: Maximize utility
- Hierarchical structure
  - UL objective depends on the optimal LL variable
  - LL decision depends on the UL variables

Example: Classical Pricing Problem



Example: Classical Pricing Problem

$$\min_{\theta,\phi \triangleq \{\phi_1,\phi_2\}} -\theta^\top \phi_1 \equiv \max_{\theta,\phi \triangleq \{\phi_1,\phi_2\}} \theta^\top \phi_1 \tag{39}$$

`subject to`

$$A\theta \leq a \tag{40}$$

$$\phi \in \arg \min_{\varphi \triangleq \{\varphi_1,\varphi_2\}} (\theta + u_1)^\top \varphi_1 + u_2^\top \varphi_2 \tag{41}$$

`subject to`

$$B_1^\top \varphi_1 + B_2^\top \varphi_2 \geq b \tag{42}$$

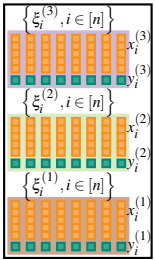
Example: Classical Pricing Problem

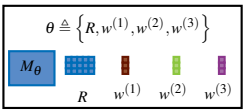
- UL variables θ are the prices
- UL objective maximizes the revenue from the goods ϕ_1
- UL constraints can bound the prices
- LL variables $\phi \triangleq \{\phi_1, \phi_2\}$ corresponds to the amount of goods (ϕ_1 might correspond to above UL seller and ϕ_2 might be an alternative available seller)
- LL objective minimizes the cost of goods
- LL constraints ensures a minimal amount of utility from the goods

Outline

Multi-task Linear Regression

- ⇛ Data for T (related) "tasks" → $\{\xi_i^{(t)} \triangleq (x_i^{(t)}, y_i^{(t)}) \in (\mathbb{R}^m \times \mathbb{R}), i \in [n^{(t)}]\}$ for each $t \in [T]$
- ⇛ Model → $M_\theta(x,t) \triangleq w^{(t)\top} R \cdot x$, $\theta \triangleq [R, \{w^t, t \in [T]\}]$
- ⇛ Loss between true y and predicted y' → $\ell(y,y') \triangleq (y-y')^2$



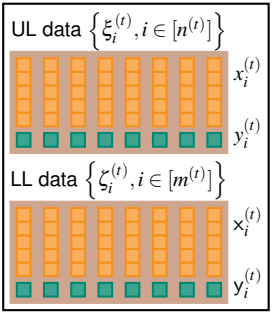


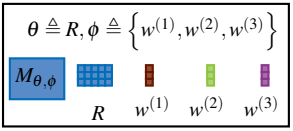
Final objective

$$f(\theta) \triangleq \sum_{t\in[T]} \frac{1}{n^{(t)}} \sum_{i\in[n^{(t)}]} \ell(y_i^{(t)}, w^{(t)\top} R \cdot x_i^{(t)}) + \rho \sum_{t\in[T]} \|w^{(t)}\|_2^2 + \rho' \|R\|_F^2. \tag{43}$$

Bi-level Multi-task Linear Regression

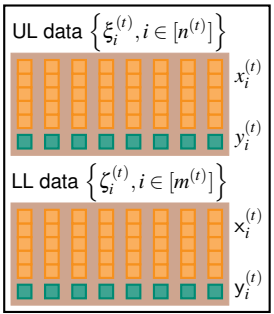
- Data for T (related) "tasks" $\rightarrow$
  - UL data $\{\xi_i^{(t)} \triangleq (x_i^{(t)}, y_i^{(t)}) \in (\mathbb{R}^m \times \mathbb{R}), i \in [n^{(t)}]\}$
  - LL data $\{\zeta_i^{(t)} \triangleq (\mathrm{x}_i^{(t)}, \mathrm{y}_i^{(t)}) \in (\mathbb{R}^m \times \mathbb{R}), i \in [m^{(t)}]\}$
- Model $\rightarrow M_{\theta,\phi}(x,t) \triangleq {w^{(t)}}^\top R \cdot x$
  - UL var $\theta \triangleq R$, LL var $\phi \triangleq [w^{(1)}, \ldots, w^{(T)}]$
- Loss $\rightarrow \ell(y,y') \triangleq (y-y')^2$
- Objectives:
  - UL: $\sum_{t\in[T]} \frac{1}{n^{(t)}} \sum_{i\in[n^{(t)}]} \ell(y_i^{(t)}, {w^{(t)}}^\top R \cdot x_i^{(t)}) + \rho' \|R\|_F^2$
  - LL: $\sum_{t\in[T]} \frac{1}{m^{(t)}} \sum_{i\in[m^{(t)}]} \ell(\mathrm{y}_i^{(t)}, {w^{(t)}}^\top R \cdot \mathrm{x}_i^{(t)}) + \rho \sum_{t\in[T]} \|w^{(t)}\|_2^2$





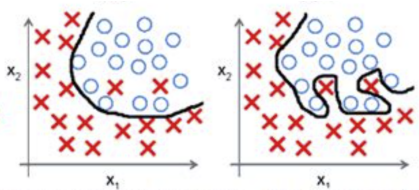
Bi-level Multi-task Nonlinear Representation Learning

- Data for T (related) "tasks" $\rightarrow$
  - UL data $\{\xi_i^{(t)} \triangleq (x_i^{(t)}, y_i^{(t)}) \in (\mathbb{R}^m \times \mathbb{R}), i \in [n^{(t)}]\}$
  - LL data $\{\zeta_i^{(t)} \triangleq (\mathrm{x}_i^{(t)}, \mathrm{y}_i^{(t)}) \in (\mathbb{R}^m \times \mathbb{R}), i \in [m^{(t)}]\}$
- Model $\rightarrow M_{\theta,\phi}(x,t) \triangleq M_{\omega^{(t)}} \circ M_R(x)$
  - UL var $\theta \triangleq R$, LL var $\phi \triangleq [\omega^{(1)}, \ldots, \omega^{(T)}]$
- Loss $\rightarrow \ell(y,y') \triangleq (y-y')^2$
- Objectives:
  - UL: $\sum_{t\in[T]} \frac{1}{n^{(t)}} \sum_{i\in[n^{(t)}]} \ell(y_i^{(t)}, M_{\omega^{(t)}} \circ M_R(x_i^{(t)}))$
  - LL: $\sum_{t\in[T]} \frac{1}{m^{(t)}} \sum_{i\in[m^{(t)}]} \ell(\mathrm{y}_i^{(t)}, M_{\omega^{(t)}} \circ M_R(\mathrm{x}_i^{(t)}))$



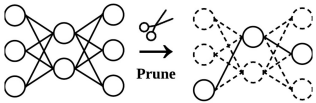
Digression: Overfitting and UL vs LL data

- Main mismatch between UL and LL obj are the data
  - UL data: $\{\xi_i^{(t)} \triangleq (x_i^{(t)}, y_i^{(t)}), i \in [n^{(t)}]\}$
  - LL data: $\{\zeta_i^{(t)} \triangleq (\mathrm{x}_i^{(t)}, \mathrm{y}_i^{(t)}), i \in [m^{(t)}]\}$
- In ML, this separation of data promotes better *generalization* – consistent performance on unseen data (**left figure**)
- Otherwise, optimization pushes model to learn the data really well, often *overfitting* – fitting noise and spurious signals (**right figure**)



Source: `https://math.mit.edu/ennui`

Neural Network Pruning & Compression



$$\min_{\theta \in \Theta} f(\theta) \triangleq \min_{\substack{m \in \{0,1\}^d, \\ \phi \in \mathbb{R}^d}} \sum_{i \in [n]} \ell(y_i, M_{m \odot \phi}(x_i)) \tag{44}$$

subject to

$$\mathbf{1}_d^\top m = \alpha d \tag{45}$$

$$\theta \triangleq [m, \phi] \in \mathbb{R}^{2d} \tag{46}$$

Model Pruning

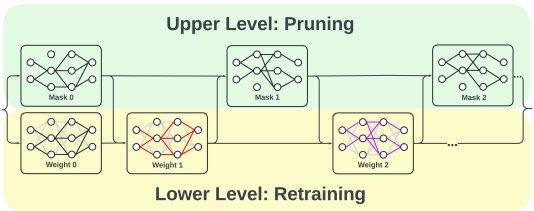
- ⇛ Remove redundant weights
- ⇛ Compress model
- ⇛ Speed up inference
- ⇛ Compression factor $\alpha \leq 0.1$

Constraints

- ⇛ Integrality cst
- ⇛ Simplex cst

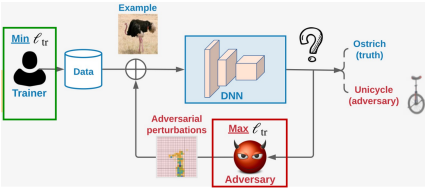
Bi-level Pruning [Zhang et al., 2022a]

- Data →
  - UL data $\{\xi_i \triangleq (x_i, y_i) \in (\mathbb{R}^m \times \mathscr{Y}), i \in [n]\}$
  - LL data $\{\zeta_i \triangleq (\mathrm{x}_i, \mathrm{y}_i) \in (\mathbb{R}^m \times \mathscr{Y}), i \in [m]\}$
- Model → $M_{\theta \odot \phi}(x)$
  - UL variable $\theta \in \{0,1\}^d$ – the parameter mask
  - LL variable ϕ – the model parameters (for the unmasked params)
- Loss → $\ell(y, y')$
- Objectives → UL: $\sum_{i \in [n]} \ell(y_i, M_{\theta \odot \phi}(x_i))$ LL: $\sum_{i \in [m]} \ell(\mathrm{y}_i, M_{\theta \odot \phi}(\mathrm{x}_i)) + \rho \|\phi\|_2^2$
- Constraints → UL: $\mathbf{1}_d^\top \theta = \alpha d, \alpha \ll 1$



Adversarially Robust Learning

Solution. Train models that are "robust" to adversarial perturbations



$$\min_{\phi} \max_{\delta_i, \|\delta_i\|_\infty \leq \varepsilon, i \in [n]} \sum_{i \in [n]} \ell(y_i, M_\phi(x_i + \delta_i)) \tag{47}$$

$$\theta \triangleq [\phi, \{\delta_i, i \in [n]\}] \tag{48}$$

Bi-level Adversarial Training [Zhang et al., 2022b]

- Data → $\{\xi_i \triangleq (x_i, y_i) \in (\mathbb{R}^m \times \mathscr{Y}), i \in [n]\}$
- Model → $M_\theta(x)$
  - UL variable θ – the model parameters
  - LL variable $\phi = [\delta_1, \ldots, \delta_n]$ – the per-example adversarial perturbations
- Loss →
  - UL: Learning loss $\ell(y, y')$
  - LL: attack loss $\ell_{\mathtt{atk}}(y, y')$
    - Measures attack success rate of adversary
    - Various different "threat models" / attack losses available
- Objectives → UL: $\sum_{i\in[n]} \ell(y_i, M_\theta(x_i + \delta_i))$ LL: $\sum_{i\in[n]} \ell_{\mathtt{atk}}(y_i, M_\theta(x_i + \delta_i))$
- Constraints → LL: $\|\delta_i\|_p \leq \varepsilon \forall i \in [n]$

Hyperparameter Optimization

Various "hyperparameters" need to be selected before learning/optimization:

- ⇛ Linear models: $f(\phi) \triangleq \frac{1}{n}\sum_{i\in[n]} \ell(y_i, \phi^\top x_i) + \rho \|\phi\|_p$
  - ⇛ Regularization penalty ρ
  - ⇛ Regularization form / norm p
  - ⇛ Loss function $|\cdot|$ vs $(\cdot)^2$ vs Poisson loss vs Gamma loss vs ...
- ⇛ Neural Networks: $M_\phi(x) \triangleq W_L \cdot \sigma(W_{L-1} \cdot \sigma(\cdots W_2 \cdot \sigma(W_1 x)\cdots))$
  - ⇛ Number of layers L
  - ⇛ Width/size of each layer
  - ⇛ Activation function σ
- ⇛ Optimizer:
  - ⇛ (initial) Learning rate
  - ⇛ Learning rate scheduling
  - ⇛ Momentum parameters
  - ⇛ many others ... (for modern optimizers such as Adam)

Bi-level Hyperparameter Optimization

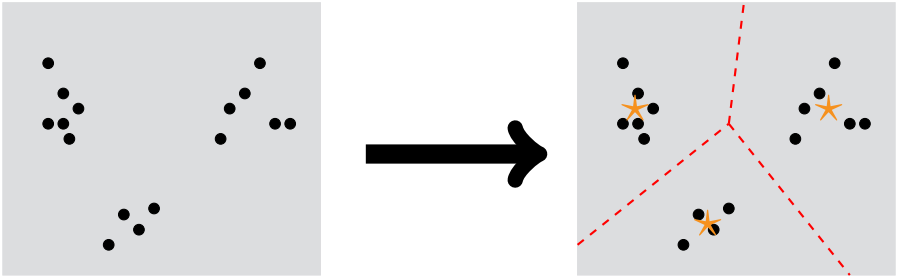
- Data →
  - UL **validation** data $\{\xi_i \triangleq (x_i, y_i), i \in [n]\}$
  - LL **training** data $\{\zeta_j \triangleq (\mathrm{x}_j, \mathrm{y}_j), j \in [m]\}$
- Model → M_ϕ
  - UL variable: θ hyperparameters
  - LL variable: ϕ ML model trained with selected hyperparameters
- Loss →
  - UL: Validation loss $\ell_{\mathsf{val}}(y, y')$
  - LL: Training loss $\ell_{\mathsf{tr}}(y, y', \theta)$ – often depends on the hyperparameter
- Objectives → UL: $\sum_{i\in[n]} \ell_{\mathsf{val}}(y_i, M_\phi(x_i))$ LL: $\sum_{i\in[m]} \ell_{\mathsf{tr}}(\mathrm{y}_i, M_\phi(\mathrm{x}_i), \theta)$
- Constraints →
  - UL: Often box constraints and hyperparameter dependency constraints,
  - LL: Model parameter constraints defined by the hyperparameter

Bi-level Hyperparameter Optimization

Final objective.

$$\begin{aligned} &\min_{\theta\in\Theta,\phi\in S(\theta)} \sum_{i\in[n]} \ell_{\mathsf{val}}(y_i, M_\phi(x_i)) \\ &\texttt{s.t.} \quad S(\theta) = \arg\min_{\phi\in\Phi(\theta)} \sum_{i\in[m]} \ell_{\mathsf{tr}}(\mathrm{y}_i, M_\phi(\mathrm{x}_i), \theta). \end{aligned} \tag{49}$$

k-means Clustering



Bi-level k-means clustering

- Data $\rightarrow \{\xi_i \triangleq x_i \in \mathbb{R}^m, i \in [n]\}$
- Model $\rightarrow M_{\theta,\phi}$
  - UL variable: θ cluster centers $[c_j \in \mathbb{R}^m, j \in [k]]$
  - LL variable: ϕ cluster assignments $[w_i \in \{0,1\}^k, i \in [n]]$
- Loss $\rightarrow$ distance to assigned cluster center $\sum_{j\in[k]} w_i[j]\|x_i - c_j\|_2^2$
- Objectives $\rightarrow$ UL/LL $\sum_{i\in[n]}\sum_{j\in[k]} w_i[j]\|x_i - c_j\|_2^2$
- Constraints $\rightarrow$ LL: $\mathbf{1}_k^\top w_i = 1 \, \forall i \in [n]\}$

Final objective.

$$\min_{\substack{\theta\in\Theta,\\ \phi\in\Phi}} \sum_{i\in[n]} \sum_{j\in[k]} w_i[j]\|x_i - c_j\|_2^2 \tag{50}$$

$$\theta \triangleq [c_j \in \mathbb{R}^m, j \in [k]] \in \Theta = \mathbb{R}^{m\times k} \tag{51}$$

$$\phi \triangleq \left[w_i \in [0,1]^k, i \in [n]\right] \in \Phi = \{0,1\}^{k\times n} \tag{52}$$

`subject to`

$$w_i \in \arg\min_{\omega\in[0,1]^k} \sum_{j\in[k]} \omega[j]\|x_i - c_j\|_2^2, \forall i \in [n] \tag{53}$$

$$\mathbf{1}_k^\top w_i = 1, \forall i \in [n] \tag{54}$$

Outline

Straightforward Approach: Alternating (S)GD

Unconstrained continuous bi-level problem

$$\min_{\theta,\phi} f_u(\theta,\phi) \quad \texttt{subject to} \quad \phi \in \arg\min_{\varphi} f_l(\theta,\varphi) \tag{55}$$

Algorithm 3 Alternating Gradient descent

Input: Initialization θ^0, ϕ^0, initial learning rates α^0, β^0 for UL and LL resp.

for $k = 1, 2, \cdots, K$ **do**

$\phi^{k+1} \leftarrow \phi^k - \beta^k \cdot \nabla_\phi f_l(\theta,\phi)\big|_{\theta=\theta^k,\phi=\phi^k}$ `// LL update`

$\theta^{k+1} \leftarrow \theta^k - \alpha^k \cdot \nabla_\theta f_u(\theta,\phi)\big|_{\theta=\theta^k,\phi=\phi^{k+1}}$ `// UL update`

Learning rate updates $\alpha^k \to \alpha^{k+1}$, $\beta^k \to \beta^{k+1}$

return θ^{K+1}, ϕ^{K+1}

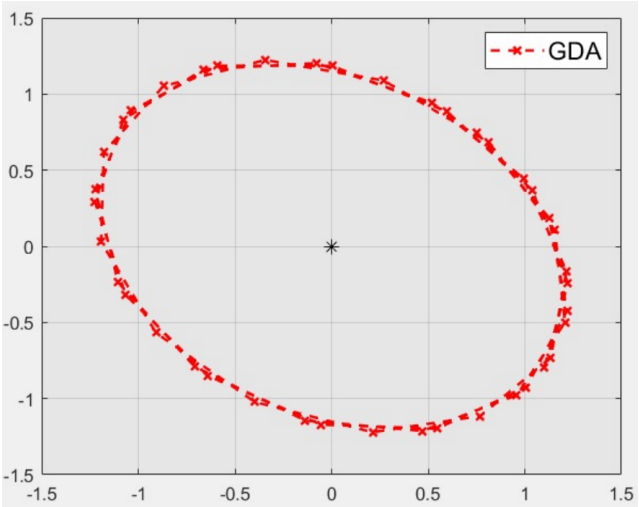
Alternating (S)GD fails!

- Convergence not guaranteed,
- Even if $f_u(\cdot, \phi)$ is strongly convex in θ and $f_l(\theta, \cdot)$ is strongly convex in ϕ

Example:

- $f_u(\theta, \phi) = \frac{1}{2}\theta^\top A\phi$
- $f_l(\theta, \phi) = -\frac{1}{2}\theta^\top A\phi$
- LL update: $\phi^{k+1} \leftarrow \phi^k + \beta^k \cdot A^\top \theta^k$
- UL update: $\theta^{k+1} \leftarrow \theta^k - \alpha^k \cdot A\phi^{k+1}$
- Alternating GD iterates cycle around the solution

Alternating (S)GD fails!



Constraints complicate matters!

- ⇛ LL problem might be infeasible for certain value of UL θ
- ⇛ The set of feasible optimal LL solutions for all UL feasible values can be nonconvex
- ⇛ UL constraints can further make this nonconvex set disconnected
- ⇛ Ignoring the LL optimality constraint often leads to suboptimal solutions

See Beck and Schmidt [2021, Example 1.12, page 16] for a precise example and explanation.

Resources

- Bi-level Introduction JPOC 2021 [Beck and Schmidt, 2021]
  `https://www.lamsade.dauphine.fr/poc/?q=node/76`
- Bi-level Optimization in Machine Learning: Foundations and Applications
  `https://sites.google.com/view/aaai2023tutorial/home`
- Surveys on Bi-level Optimization in Machine Learning
  - Investigating Bi-Level Optimization for Learning and Vision From a Unified Perspective: A Survey and Beyond [Liu et al., 2021]
  - Gradient-based Bi-level Optimization for Deep Learning: A Survey [Chen et al., 2022]

References

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013. URL https://arxiv.org/pdf/1312.6199.pdf.

Yihua Zhang, Yuguang Yao, Parikshit Ram, Pu Zhao, Tianlong Chen, Mingyi Hong, Yanzhi Wang, and Sijia Liu. Advancing model pruning via bi-level optimization. In *Annual Conference on Neural Information Processing Systems*, 2022a.

Yihua Zhang, Guanhua Zhang, Prashant Khanduri, Mingyi Hong, Shiyu Chang, and Sijia Liu. Revisiting and advancing fast adversarial training through the lens of bi-level optimization. In *International Conference on Machine Learning*, pages 26693–26712. PMLR, 2022b.

Yasmine Beck and Martin Schmidt. A gentle and incomplete introduction to bilevel optimization. 2021. URL https://optimization-online.org/2021/06/8450/.

Risheng Liu, Jiaxin Gao, Jin Zhang, Deyu Meng, and Zhouchen Lin. Investigating bi-level optimization for learning and vision from a unified perspective: A survey and beyond. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(12):10045–10067, 2021. URL https://ieeexplore.ieee.org/abstract/document/9638340.

Can Chen, Xi Chen, Chen Ma, Zixuan Liu, and Xue Liu. Gradient-based bi-level optimization for deep learning: A survey. *arXiv preprint arXiv:2207.11719*, 2022. URL https://arxiv.org/abs/2207.11719.

