Lower-Level Agnostic Bilevel Optimization

Peiwen Qiu
Dept. of ECE
The Ohio State University
Columbus, OH 43210
qiu.617@osu.edu

Prashant Khanduri
Dept. of CS
Wayne State University
Detroit, MI 48202
khanduri.prashant@wayne.edu

Jia Liu
Dept. of ECE
The Ohio State University
Columbus, OH 43210
liu@ece.osu.edu

Abstract

Bilevel optimization is a fundamental framework for machine learning problems, where an upper-level (UL) objective depends on the solution of a nested lower-level (LL) problem. Existing bilevel algorithms typically assume that the LL objective is available for evaluation, differentiation, or unrolling. However, this assumption can fail when the LL response is produced by a black-box solver, simulator, adversary, or learned predictor. In this paper, we study *LL agnostic* bilevel optimization, where the UL learner has access only to an approximate LL solution $\bar{\mathbf{y}}(\mathbf{x})$ and cannot query the LL objective, its gradients, Hessian/Jacobian information, or the procedure that generates the LL response. We propose LeGo-BiO (Lower-level Agnostic Bilevel Optimization), a coordinate-wise pseudo-hypergradient method that estimates the missing Jacobian $\nabla_{\mathbf{x}}\bar{\mathbf{y}}(\mathbf{x})$ using finite differences of $\bar{\mathbf{y}}(\cdot)$. For each UL coordinate, LeGo-BiO constructs a decision vector subject to 1-sparse updates, i.e., restricting modifications to a single coordinate per step, thereby isolating the corresponding Jacobian column needed in the UL gradient computation. This avoids the directional-projection limitation of full-parameter response differences and, unlike derivative-free/zeroth-order approximations of the entire UL gradient, preserves the available first-order gradient information with respect to the UL variables for more accurate UL gradient evaluation. We establish convergence guarantees for LeGo-BiO in both deterministic and stochastic settings under an LL error-bound/Polyak–Łojasiewicz-type condition, without requiring LL strong convexity. Our bounds yield an $\mathcal{O}(T^{-1})$ rate in the deterministic setting and an $\mathcal{O}(T^{-1/2})$ rate in the stochastic setting when the LL approximation error decreases geometrically. Experiments on deep hyper-representation, adversarial training, and data weighting for LLM alignment demonstrate competitive performance of LeGo-BiO despite being agnostic to the LL objective.

1 Introduction

Bilevel optimization provides a natural framework for learning problems in which an upper-level (UL) decision depends on the solution of a nested lower-level (LL) problem. A standard bilevel optimization problem is formulated as

$$\min_{\mathbf{x}\in\mathbb{R}^{d_u}} \ell(\mathbf{x}) \triangleq f\left(\mathbf{x}, \mathbf{y}^*(\mathbf{x})\right) \quad \text{s.t.} \quad \mathbf{y}^*(\mathbf{x}) = \arg\min_{\mathbf{y}\in\mathbb{R}^{d_l}} g(\mathbf{x}, \mathbf{y}), \tag{1}$$

where $f : \mathbb{R}^{d_u} \times \mathbb{R}^{d_l} \to \mathbb{R}$ and $g : \mathbb{R}^{d_u} \times \mathbb{R}^{d_l} \to \mathbb{R}$ denote the UL and LL objectives, respectively. This framework has been widely adopted across a broad range of machine learning applications, including hyperparameter optimization [1, 2], meta-learning [3, 4], reinforcement learning [5, 6], neural architecture search [7,8], adversarial training [9–11], data hyper-cleaning [2,12], and dictionary learning [13, 14].

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

Most existing methods for solving Problem (1) are developed under the assumption that the LL objective is explicitly available and can be differentiated, queried, or optimized by the UL learner. However, this assumption may fail in many settings where the LL solution is produced by an external procedure whose objective or internal mechanism is inaccessible. For example, in adversarial training, the UL problem updates the model parameters, while the LL problem generates adversarial examples or perturbations against the current model. When this LL procedure is implemented through a black-box attack, simulator, or learned generator, the UL training algorithm may only observe the resulting adversarial examples, without direct access to the underlying LL objective, a differentiable description of the LL objective (i.e., gradient, Hessian, or Jacobian), or the internal optimization procedure that produce them.

Motivated by applications where the LL response is observable but the objective or mechanism generating it is unavailable to the UL learner, we study a new problem setting, which we call *LL agnostic* bilevel optimization. In this setting, the LL objective g still defines the target response $\mathbf{y}^*(\mathbf{x})$ in Eq. (1), but the UL algorithm does not have access to g, including its function values (*zeroth-order* information), gradients (*first-order* information), and Hessian/Jacobian information (*second-order* information). Instead, for a UL variable $\mathbf{x}$, the algorithm only observes an approximate LL solution $\bar{\mathbf{y}}(\mathbf{x})$, which may be produced by a black-box solver, simulator, or learned predictor.

This LL-agnostic setting goes beyond the scope of existing bilevel optimization algorithms. When the LL problem is strongly convex, classical bilevel optimization methods update $\mathbf{x}$ using the gradient of the UL objective (UL-gradient), which can be computed using the chain rule as:

$$\nabla \ell(\mathbf{x}) = \nabla_{\mathbf{x}} f(\mathbf{x}, \mathbf{y}^*(\mathbf{x})) + [\nabla_{\mathbf{x}} \mathbf{y}^*(\mathbf{x})]^\top \nabla_{\mathbf{y}} f(\mathbf{x}, \mathbf{y}^*(\mathbf{x})). \tag{2}$$

Approximate implicit differentiation (AID) methods [5, 15–21] estimate the Jacobian $\nabla_{\mathbf{x}} \mathbf{y}^*(\mathbf{x})$ in Eq. (2) by applying the implicit function theorem [22]. This requires derivative information from the LL objective, i.e., $\nabla^2_{\mathbf{xy}} g$ and $\nabla^2_{\mathbf{yy}} g$, to construct the resulting hypergradient. Iterative differentiation (ITD) methods [2,12,23–27] instead differentiate through the LL optimization trajectory, and therefore require access to the LL update rule or computational graph. Both classes of methods are *incompatible* with the LL-agnostic setting, where the UL algorithm *cannot* evaluate or differentiate the LL objective and *cannot* necessarily backpropagate through the procedure that returns $\bar{\mathbf{y}}(\mathbf{x})$.

Existing Hessian/Jacobian-free and derivative-free bilevel methods are also not directly applicable. Although these methods avoid second-order derivative computations, they assume an information model in which the LL objective is accessible in some form. Hessian/Jacobian-free first-order methods [28–32] avoid explicit second-order derivatives, but still rely on gradients of the LL objective or reformulations based on LL optimality conditions. Zeroth-order bilevel methods [33–36] typically construct finite-difference or smoothing-based estimators using function-value queries of the UL and LL objectives. A different line of derivative-free work considers direct-search methods that assume access to a black-box LL oracle returning approximate LL solutions and optimize the resulting inexact bilevel objective directly through function-value comparisons [37]. Such methods avoid differentiating the LL problem, but treat the induced UL objective as a derivative-free black box rather than exploiting its available first-order structure. In contrast, the LL-agnostic setting provides no function-value or gradient access to g, while the UL objective f remains differentiable and accessible. Our goal is therefore not to treat the entire bilevel problem as a black box. Instead, we exploit the available gradients of f and use only the observable LL response $\bar{\mathbf{y}}(\mathbf{x})$ to approximate the missing sensitivity $\nabla_{\mathbf{x}} \bar{\mathbf{y}}(\mathbf{x})$. Thus, the key difficulty is not merely the absence of Hessian/Jacobian information or derivative information from the induced bilevel objective, but the absence of a usable LL objective altogether.

To address this difficulty, we propose LeGo-BiO (Lower-level Agnostic Bilevel Optimization), a *coordinate-wise pseudo-hypergradient* method that uses the observable LL response map $\mathbf{x} \mapsto \bar{\mathbf{y}}(\mathbf{x})$ to estimate the missing Jacobian $\nabla_{\mathbf{x}} \mathbf{y}^*(\mathbf{x})$ in Eq. (2). Once $\bar{\mathbf{y}}(\mathbf{x})$ is used as a tractable proxy for $\mathbf{y}^*(\mathbf{x})$, the UL-gradient in Eq. (2) can be approximated by replacing $\mathbf{y}^*(\mathbf{x})$ with $\bar{\mathbf{y}}(\mathbf{x})$. The remaining challenge is to estimate the Jacobian $\nabla_{\mathbf{x}} \bar{\mathbf{y}}(\mathbf{x})$ without differentiating through g or through the mechanism that produces $\bar{\mathbf{y}}$. LeGo-BiO addresses this challenge by updating one UL coordinate at a time. Specifically, for each UL coordinate, LeGo-BiO constructs a decision vector subject to 1-sparse updates, i.e., restricting modifications to a single coordinate per step, thereby isolating the corresponding Jacobian column needed in the UL gradient computation. This coordinate-wise Jacobian estimate is then combined with the UL gradients (i.e., $\nabla_{\mathbf{x}} f$ and $\nabla_{\mathbf{y}} f$) to form a pseudo-hypergradient. As a result, LeGo-BiO requires only gradients of the UL objective and queries to

the approximate LL solution map without needing g, ∇g, $\nabla^2_{\mathbf{xy}} g$, or $\nabla^2_{\mathbf{yy}} g$, making it applicable to LL-agnostic setting.

The coordinate-wise design is important for both statistical and algorithmic reasons. A finite difference based on a full-parameter change in $\mathbf{x}$ only provides one directional derivative of $\bar{\mathbf{y}}$ along the chosen displacement, and therefore cannot identify the entire Jacobian matrix information needed to form a full UL-gradient without additional perturbations or structural assumptions. In contrast, LeGo-BiO isolates one Jacobian column at a time, which aligns naturally with coordinate-wise UL updates and avoids mixing the effects of multiple coordinates in a single response difference. Moreover, LeGo-BiO uses finite differences only to estimate the missing Jacobian within the UL-gradient, while retaining the available first-order gradients of the UL objective. This distinguishes it from the zeroth-order approaches that approximate the entire UL-gradient through function-value differences of the composite UL objective $f(\mathbf{x}, \bar{\mathbf{y}}(\mathbf{x}))$, which are typically more sensitive to noise and dimensionality [38].

Our main contributions are summarized as follows.

- We formulate and analyze a LL agnostic bilevel optimization setting in which the LL objective is unavailable to the UL algorithm, and the only available LL-side information is an approximate solution $\bar{\mathbf{y}}(\mathbf{x})$. This information model distinguishes the problem from both classical gradient-based bilevel optimization and standard zeroth-order bilevel optimization.
- We develop LeGo-BiO, a coordinate-wise pseudo-hypergradient algorithm that estimates the Jacobian of the approximate LL solution (i.e., $\nabla_{\mathbf{x}} \bar{\mathbf{y}}$ in Eq. (2)) *using only differences between LL responses*. This method avoids differentiating through the LL objective or the procedure that generates $\bar{\mathbf{y}}(\mathbf{x})$, and is therefore applicable to the LL-agnostic setting.
- We establish convergence guarantees for LeGo-BiO with nonconvex UL objectives in both deterministic and stochastic settings. The analysis does not require strong convexity of the LL objective (LLSC); instead, it relies on an LL error-bound/Polyak–Łojasiewicz-type condition. LeGo-BiO achieves an $\mathcal{O}(T^{-1})$ convergence rate in the deterministic setting and an $\mathcal{O}(T^{-1/2})$ rate in the stochastic setting when the LL approximation error decreases geometrically. Despite operating in the more challenging LL-agnostic setting, these rates match the state-of-the-art deterministic and stochastic convergence rates achieved by standard bilevel methods under the LLSC assumption.
- We evaluate the numerical performance of LeGo-BiO on three tasks: deep hyper-representation, adversarial training, and data weighting for LLM alignment. The results show that, despite having no access to the LL objective, LeGo-BiO achieves competitive or superior performance compared with baselines that use LL objective information during training.

2 Related Work

In this section, we provide an overview on three closely related lines of research to place our work in comparative perspectives.

1) Gradient-based Bilevel Optimization: Gradient-based bilevel optimization methods typically compute or approximate the UL-gradient of the value function $\ell(\mathbf{x}) = f(\mathbf{x}, \mathbf{y}^*(\mathbf{x}))$. Two major classes of methods are approximate implicit differentiation (AID) and iterative differentiation (ITD). AID-based methods use the implicit function theorem to approximate the implicit gradient of $\ell(\mathbf{x})$ [5, 15–21]. These methods generally require LL derivative information, such as $\nabla^2_{\mathbf{xy}} g$ and $\nabla^2_{\mathbf{yy}} g$, or Hessian-vector and Jacobian-vector products derived from the LL objective. ITD-based methods instead approximate the UL-gradient by differentiating through the unrolled LL optimization trajectory [2, 12, 23–27]. As a result, ITD methods require access to the LL update rule or computational graph, since the UL-gradient is obtained by backpropagating through the LL solver.

These approaches are effective when the LL objective is available and differentiable, but they are inapplicable for the LL-agnostic setting studied in this paper. In our setting, the UL learner cannot query the LL objective g, compute $\nabla^2_{\mathbf{xy}} g$ and $\nabla^2_{\mathbf{yy}} g$, form Hessian/Jacobian-vector products, or backpropagate through the procedure that produces the approximate LL solution $\bar{\mathbf{y}}(\mathbf{x})$. Therefore, neither AID nor ITD can be directly applied. In contrast, LeGo-BiO requires only gradients of the UL objective and queries to the observable LL response map $\mathbf{x} \mapsto \bar{\mathbf{y}}(\mathbf{x})$. This distinction is also reflected in the convergence analysis: many gradient-based bilevel methods rely on LL strong convexity to

control the solution map and hypergradient approximation errors, whereas our analysis uses an LL error-bound condition together with an explicit approximation-error sequence for $\bar{\mathbf{y}}(\mathbf{x})$.

2) Hessian/Jacobian-free First-order Bilevel Optimization: A recent line of work develops Hessian/Jacobian-free first-order bilevel methods to reduce the computational burden of second-order hypergradient estimation. These methods avoid explicitly computing LL Hessian or Jacobian matrices and instead replace second-order quantities with first-order approximations, optimality-based constraints, or penalty/value-function reformulations. For example, [28] approximates second-order information using first-order gradient information. Other methods reformulate the bilevel problem through LL optimality conditions and penalty/value-function-based objectives so that the resulting algorithms only require first-order gradient information [29–32].

Although these methods avoid explicit second-order derivatives, they still assume access to the LL objective in a usable form. In particular, they typically require LL gradients, LL optimality residuals, or penalty/value functions involving the LL objective. Hence, they remain incompatible with the LL-agnostic setting. The challenge in our setting is not merely that Hessian/Jacobian information is unavailable; rather, the LL objective itself is unavailable to the UL learner. LeGo-BiO therefore differs from Hessian/Jacobian-free first-order bilevel algorithms by estimating the missing Jacobian $\nabla_{\mathbf{x}}\bar{\mathbf{y}}$ directly from differences of approximate LL solutions, without using g or its gradients.

3) Derivative-free/Zeroth-order Bilevel Optimization: Derivative-free/zeroth-order bilevel optimization studies settings where derivative information is unavailable and algorithms rely on function-value queries or direct evaluations of the induced bilevel objective. Several works develop finite-difference, evolution-strategy, or Gaussian-smoothing estimators for bilevel optimization. For example, evolution-strategy-based methods have been used to avoid second-order derivatives in meta-learning [33]. Gaussian-smoothing-based methods estimate derivatives of both UL and LL objectives from noisy function values [34, 35]. More recent single-loop zeroth-order bilevel methods further improve the efficiency of finite-difference UL-gradient approximation [36]. Beyond gradient-estimation-based approaches, derivative-free direct-search methods have also been studied for bilevel optimization. In particular, [37] develop inexact direct-search schemes that use a fixed-accuracy black-box LL solver and optimize the resulting approximate bilevel objective through function-value comparisons, without explicitly constructing gradient estimators.

Our method is related to this literature because it also uses finite differences, but both the information model and the role of function evaluations are different. Standard zeroth-order bilevel methods assume access to function values of the objectives, including the LL objective g, and use finite differences or smoothing to estimate gradients of the UL and LL objectives. Likewise, direct-search bilevel methods treat the induced UL objective as a derivative-free black box and make optimization decisions based on its function values. In contrast, the LL-agnostic setting provides *no function-value access to g.* At the same time, the UL objective f remains differentiable and accessible. Therefore, LeGo-BiO does not treat either level, or the induced bilevel objective, as a fully black-box optimization problem. Instead, it preserves the chain-rule structure of the UL-gradient by using $\nabla_{\mathbf{x}} f$ and $\nabla_{\mathbf{y}} f$ directly, while applying finite differences only to estimate the missing Jacobian $\nabla_{\mathbf{x}}\bar{\mathbf{y}}(\mathbf{x})$ of the approximate LL solution. This targeted use of finite differences distinguishes LeGo-BiO from both gradient-estimation-based zeroth-order methods and direct-search-based derivative-free methods, and is central to its LL agnostic design. Since function-value-based zeroth-order estimators often incur additional sampling and dimension-dependent errors [38], retaining the available UL gradients in LeGo-BiO can lead to a more accurate and less noisy UL-gradient approximation.

3 The LeGo-BiO Algorithm

In this section, we propose LeGo-BiO, a coordinate-wise pseudo-hypergradient method for LL agnostic bilevel optimization. The main idea is to estimate the Jacobian of the observable LL response map $\mathbf{x} \mapsto \bar{\mathbf{y}}(\mathbf{x})$ by updating one UL coordinate at a time and measuring the resulting change in the approximate LL solution. This construction enables LeGo-BiO to approximate the missing Jacobian information of $\bar{\mathbf{y}}$ without accessing the LL objective g, its gradient ∇g, or its second-order derivatives, i.e., $\nabla^2_{\mathbf{xy}} g$ and $\nabla^2_{\mathbf{yy}} g$.

The LeGo-BiO algorithm is summarized in Algorithm 1. For each UL coordinate, LeGo-BiO constructs a decision vector subject to 1-sparse updates, which restricts modifications to a single coordinate per step, thereby isolating the corresponding Jacobian column needed in the UL gradient

computation. Specifically, at iteration t and for each coordinate $j \in \{1, \dots, d_u\}$, define the coordinate displacement as

$$s_t^j \triangleq x_{t-1}^j - x_t^j, \quad \widetilde{\mathbf{x}}_t^j \triangleq \mathbf{x}_{t-1} - s_t^j \mathbf{e}_j, \tag{3}$$

where $\mathbf{e}_j$ denotes the j-th canonical basis vector in $\mathbb{R}^{d_u}$. The point $\widetilde{\mathbf{x}}_t^j$ differs from $\mathbf{x}_{t-1}$ only in the j-th coordinate. Therefore, the difference $\bar{\mathbf{y}}(\mathbf{x}_{t-1}) - \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j)$ captures how the approximate LL response changes when only the j-th UL coordinate is updated. This provides a one-dimensional finite-difference estimate of the j-th column of the Jacobian $\nabla_{\mathbf{x}} \bar{\mathbf{y}}(\mathbf{x})$:

$$\frac{\bar{\mathbf{y}}(\mathbf{x}_{t-1}) - \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j)}{s_t^j} \approx \nabla_{\mathbf{x}} \bar{\mathbf{y}}(\mathbf{x}_t) \mathbf{e}_j.$$

Throughout the paper, we assume that $s_t^j \neq 0$ to ensure that the finite-difference estimate is well-defined. This column-wise estimate is exactly the quantity needed for the j-th UL-gradient component, because

$$\left[\nabla_{\mathbf{x}} \bar{\mathbf{y}}(\mathbf{x})^\top \nabla_{\mathbf{y}} f(\mathbf{x}, \bar{\mathbf{y}}(\mathbf{x}))\right]_j = \left\langle \nabla_{\mathbf{x}} \bar{\mathbf{y}}(\mathbf{x}) \mathbf{e}_j, \nabla_{\mathbf{y}} f(\mathbf{x}, \bar{\mathbf{y}}(\mathbf{x})) \right\rangle.$$

Algorithm 1 The LeGo-BiO algorithmic framework.

Input: Initial points $\mathbf{x}_{-1}, \mathbf{x}_0 \in \mathbb{R}^{d_u}$, and upper-level step sizes $\{\alpha_t\}_{t=0}^{T-1}$
for $t = 0, \dots, T-1$ **do**
  Receive the lower-level approximation $\bar{\mathbf{y}}(\mathbf{x}_{t-1})$
  for $j = 1, 2, \dots, d_u$ **do**
    Form the point $\widetilde{\mathbf{x}}_t^j$ as in Eq. (3)
    Receive the lower-level approximation $\bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j)$
    Compute pseudo-hypergradient $h_t^{f,j}$ via Eq. (4) in deterministic setting or Eq. (5) in stochastic setting
    Update $x_{t+1}^j = x_t^j - \alpha_t \left[h_t^{f,j}\right]_j$
  end for
end for

This explains why LeGo-BiO uses coordinate-wise response differences rather than a single full-parameter response difference. A full-vector perturbation would give

$$\bar{\mathbf{y}}(\mathbf{x}_{t-1}) - \bar{\mathbf{y}}(\mathbf{x}_t) \approx \nabla_{\mathbf{x}} \bar{\mathbf{y}}(\mathbf{x}_t)(\mathbf{x}_{t-1} - \mathbf{x}_t),$$

which is only one directional derivative of the response map. After contraction with $\nabla_{\mathbf{y}} f$, it yields one scalar projection

$$\left\langle \bar{\mathbf{y}}(\mathbf{x}_{t-1}) - \bar{\mathbf{y}}(\mathbf{x}_t), \nabla_{\mathbf{y}} f(\mathbf{x}_t, \bar{\mathbf{y}}(\mathbf{x}_t)) \right\rangle \approx \left\langle \mathbf{x}_{t-1} - \mathbf{x}_t, \nabla_{\mathbf{x}} \bar{\mathbf{y}}(\mathbf{x}_t)^\top \nabla_{\mathbf{y}} f(\mathbf{x}_t, \bar{\mathbf{y}}(\mathbf{x}_t)) \right\rangle,$$

not the full vector $\nabla_{\mathbf{x}} \bar{\mathbf{y}}(\mathbf{x}_t)^\top \nabla_{\mathbf{y}} f(\mathbf{x}_t, \bar{\mathbf{y}}(\mathbf{x}_t))$ needed to update all UL coordinates. Recovering the full vector from full-parameter perturbations would require multiple independent perturbations or a random-direction estimator, which introduces additional sampling error and dimension-dependent variance. The coordinate-wise construction instead isolates each component directly, leading to a more accurate UL-gradient estimate.

Deterministic pseudo-hypergradient. In the deterministic setting, the coordinate-wise pseudo-hypergradient is defined as

$$h_t^{f,j} = \nabla_{\mathbf{x}} f\left(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j)\right) + \left\langle \frac{\bar{\mathbf{y}}(\mathbf{x}_{t-1}) - \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j)}{s_t^j}, \nabla_{\mathbf{y}} f(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j)) \right\rangle \mathbf{e}_j. \tag{4}$$

Only the j-th component $[h_t^{f,j}]_j$ is used to update the coordinate x_t^j. The first term in Eq. (4) is the direct UL gradient with respect to $\mathbf{x}$. The second term estimates the indirect effect of changing $\mathbf{x}$ through the LL response map. The Jacobian $\nabla_{\mathbf{x}} \bar{\mathbf{y}}(\mathbf{x})$ is estimated through a one-dimensional finite difference of $\bar{\mathbf{y}}(\cdot)$. This coordinate-wise construction is the key mechanism that allows LeGo-BiO to construct a pseudo hypergradient using only LL responses rather than derivatives of the unknown LL objective. It is also important to distinguish Eq. (4) from a finite-difference approximation of the UL-gradient using composite UL objective $f(\mathbf{x}, \bar{\mathbf{y}}(\mathbf{x}))$. One could estimate the j-th derivative of the entire UL-gradient by

$$\frac{f(\mathbf{x}_{t-1}, \bar{\mathbf{y}}(\mathbf{x}_{t-1})) - f(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j))}{s_t^j},$$

but this would treat the entire UL objective as a black box. Such an estimator would discard the available UL gradients $\nabla_{\mathbf{x}} f$ and $\nabla_{\mathbf{y}} f$, mix the direct UL variation and the LL-response variation into a single scalar difference, and can amplify errors through the subtraction of inaccurate function values. Our LL-agnostic setting is different: the UL objective f remains differentiable and accessible, while only the LL objective g and its derivatives are unavailable. Therefore, LeGo-BiO preserves the chain-rule structure of the UL-gradient: it differentiates f directly and applies finite differences only to estimate the missing Jacobian $\nabla_{\mathbf{x}} \bar{\mathbf{y}}$. This targeted use of finite differences leads to a more accurate and less noisy UL-gradient surrogate.

Stochastic pseudo-hypergradient. We further consider the stochastic setting, where the UL and LL objectives are given by expectations:

$$f\left(\mathbf{x}, \mathbf{y}^*(\mathbf{x})\right) \triangleq \mathbb{E}_{\xi\sim\pi_f}\left[f\left(\mathbf{x}, \mathbf{y}^*(\mathbf{x}); \xi\right)\right], \quad g(\mathbf{x}, \mathbf{y}) \triangleq \mathbb{E}_{\zeta\sim\pi_g}[g(\mathbf{x}, \mathbf{y}; \zeta)],$$

where $\xi \sim \pi_f$ and $\zeta \sim \pi_g$ denote samples from the UL and LL data distributions, respectively. When only stochastic estimates are available, LeGo-BiO replaces the exact quantities in Eq. (4) with mini-batch estimates:

$$\widehat{h}_t^{f,j} = \nabla_{\mathbf{x}} f\left(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j); \mathcal{B}_t^{f_x}\right) + \left\langle \frac{\bar{\mathbf{y}}(\mathbf{x}_{t-1}; \mathcal{B}_{t-1}^y) - \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j; \mathcal{B}_t^{y,j})}{s_t^j}, \nabla_{\mathbf{y}} f(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j); \mathcal{B}_t^{f_y,j}) \right\rangle \mathbf{e}_j, \quad (5)$$

where $\mathcal{B}_t^{f_x}$ and $\mathcal{B}_t^{f_y,j}$ denote mini-batches used to estimate the UL gradients with respect to $\mathbf{x}$ and $\mathbf{y}$, respectively. $\mathcal{B}_t^{y,j}$ and $\mathcal{B}_{t-1}^y$ denote mini-batches used to obtain the stochastic LL responses at $\widetilde{\mathbf{x}}_t^j$ and $\mathbf{x}_{t-1}$. They are introduced only to characterize the stochastic approximation error and are not required to be known by the UL learner. This estimator highlights the three main sources of error in the stochastic LL-agnostic setting: the gradient approximation error from the UL sampling, the approximation error from replacing the optimal LL solution $\mathbf{y}^*(\mathbf{x})$ with $\bar{\mathbf{y}}(\mathbf{x})$, and the Jacobian-estimation error from approximating $\nabla_{\mathbf{x}}\bar{\mathbf{y}}(\mathbf{x})$ through finite differences of stochastic LL responses.

4 Theoretical Performance Analysis

In this section, we establish the convergence guarantees for LeGo-BiO in both deterministic and stochastic settings. A key feature of our analysis is that it does not require the LL objective to be strongly convex. Instead, we rely on an error-bound/PL-type condition for the LL problem.

4.1 Assumptions

We first state a set of assumptions that are needed to establish the convergence of LeGo-BiO. Although LeGo-BiO does not require direct access to the LL objective g, the assumptions below characterize the underlying bilevel problem and the accuracy required of the LL solution approximation.

Assumption 4.1 (UL Objective)**.** Let $\nabla f(\mathbf{x}, \mathbf{y}) = \nabla_{\mathbf{x}} f(\mathbf{x}, \mathbf{y}) + [\nabla_{\mathbf{x}}\mathbf{y}(\mathbf{x})]^\top \nabla_{\mathbf{y}} f(\mathbf{x}, \mathbf{y})$. The UL objective $f(\mathbf{x}, \mathbf{y})$ satisfies the following conditions.
1) The gradient map $(\mathbf{x}, \mathbf{y}) \mapsto \nabla f(\mathbf{x}, \mathbf{y})$ is L_f-Lipschitz continuous for some $L_f \geq 0$, i.e., $\|\nabla f(\mathbf{x}_1, \mathbf{y}_1) - \nabla f(\mathbf{x}_2, \mathbf{y}_2)\| \leq L_f(\|\mathbf{x}_1 - \mathbf{x}_2\| + \|\mathbf{y}_1 - \mathbf{y}_2\|)$.
2) For all $(\mathbf{x}, \mathbf{y}) \in \mathbb{R}^{d_u} \times \mathbb{R}^{d_l}$, we have $\|\nabla_{\mathbf{y}} f\left(\mathbf{x}, \mathbf{y}\right)\| \leq B_{f_y}$ for some $B_{f_y} \geq 0$.

Assumption 4.1 is standard in nonconvex bilevel optimization [5, 15, 20]. The Lipschitz condition controls the smoothness of the UL objective, while the boundedness of $\nabla_{\mathbf{y}} f$ ensures that errors in the LL response do not propagate arbitrarily into the UL hypergradient.

Assumption 4.2 (LL Objective)**.** The LL objective $g(\mathbf{x}, \mathbf{y})$ satisfies the following conditions.
1) For all $\mathbf{x} \in \mathbb{R}^{d_u}$, the LL problem admits a unique solution $\mathbf{y}^*\left(\mathbf{x}\right)$, which is differentiable.
2) For all $(\mathbf{x}, \mathbf{y}) \in \mathbb{R}^{d_u} \times \mathbb{R}^{d_l}$, we have $\|\mathbf{y} - \mathbf{y}^*\left(\mathbf{x}\right)\| \leq \kappa \left\|\nabla_{\mathbf{y}} g\left(\mathbf{x}, \mathbf{y}\right)\right\|$ for some $\kappa \geq 0$.
3) For all $\mathbf{y} \in \mathbb{R}^{d_l}$, the map $\mathbf{x} \mapsto \nabla_{\mathbf{y}} g(\mathbf{x}, \mathbf{y})$ is L_{g_y}-Lipschitz continuous for some $L_{g_y} \geq 0$, i.e., $\left\|\nabla_{\mathbf{y}} g\left(\mathbf{x}_1, \mathbf{y}\right) - \nabla_{\mathbf{y}} g\left(\mathbf{x}_2, \mathbf{y}\right)\right\| \leq L_{g_y} \left\|\mathbf{x}_1 - \mathbf{x}_2\right\|, \mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^{d_u}$.

Rather than assuming that the LL objective is μ_g-strongly convex, as is commonly adopted in the bilevel optimization literature [5, 15, 20, 39, 40], Assumption 4.2 imposes an LL error-bound/PL-type condition together with uniqueness of the LL solution. The PL-type condition is weaker than strong convexity and can hold for certain nonconvex objectives. Importantly, the uniqueness requirement does not imply strict or strong convexity; nonconvex LL objectives may still admit a unique, isolated solution while satisfying the stated PL-type error bound. Therefore, our analysis is not restricted to convex or strongly convex LL problems. As a special case, when $g(\mathbf{x}, \cdot)$ is μ_g-strongly convex, the LL PL-type condition is satisfied with $\kappa = 1/\mu_g$. Assumptions 4.2 and 4.1 are used to establish the Lipschitz continuity of $\nabla\ell(\mathbf{x})$.

Assumption 4.2(2) gives $\|\bar{\mathbf{y}}(\mathbf{x}) - \mathbf{y}^*(\mathbf{x})\| \leq \kappa\epsilon_t$, so $\delta_t = \kappa\epsilon_t$. A fixed LL budget yields an approximation-dependent neighborhood, whereas an increasing budget can make δ_t decrease and enable asymptotic stationarity. Note that the UL learner does **not** need to access g or know δ_t.

Assumption 4.3 (LL Approximation)**.** The approximate LL response map $\bar{\mathbf{y}}(\mathbf{x})$ satisfies the following conditions.
1) For all $t \in \{0, 1, \ldots, T-1\}$ and every $\mathbf{x}_t \in \mathbb{R}^{d_u}$, we have $\|\bar{\mathbf{y}}(\mathbf{x}_t) - \mathbf{y}^*(\mathbf{x}_t)\| \leq \delta_t$ for some $\delta_t \geq 0$.
2) The Jacobian map $\mathbf{x} \mapsto \nabla_{\mathbf{x}} \bar{\mathbf{y}}(\mathbf{x})$ is L_y-Lipschitz continuous for some $L_y \geq 0$, and $\|\nabla_{\mathbf{x}} \bar{\mathbf{y}}(\mathbf{x})\| \leq B_y$ for some $B_y \geq 0$.

Assumption 4.3 captures LeGo-BiO's two error sources: LL approximation, and finite-difference Jacobian estimation. Their global and uniform forms are sufficient conditions for analysis, while in practice they need only hold locally along the iterates and queried response segments. Assumption 4.3(1) follows naturally from practical LL stopping criteria. If an external LL solver returns $\bar{\mathbf{y}}(\mathbf{x})$ satisfying $\|\nabla_{\mathbf{y}} g(\mathbf{x}, \bar{\mathbf{y}}(\mathbf{x}))\| \leq \epsilon_t$, Assumption 4.3(2) is reasonable when the LL oracle induces a smooth and stable input-output map. For example, it holds for an exact isolated, nondegenerate LL solution when the LL Hessian is uniformly nonsingular and the relevant second derivatives of g are bounded and Lipschitz. It also holds when $\bar{\mathbf{y}}(\mathbf{x})$ is generated by a fixed number of smooth solver updates whose derivatives are bounded and Lipschitz on the region visited by the iterates. Note that the UL learner **need not** evaluate the Jacobian or know B_y and L_y; they only characterize the black-box map's regularity.

For the stochastic setting, we further impose the following assumptions on the stochastic estimators.

Assumption 4.4 (Stochastic Gradients)**.** For any $(\mathbf{x}, \mathbf{y}) \in \mathbb{R}^{d_u} \times \mathbb{R}^{d_l}$ and data batches $\mathcal{B}^{f_x}$, $\mathcal{B}^{f_y}$, and $\mathcal{B}^{y}$, the stochastic gradients of the UL objective are unbiased and have bounded variances:

$$\mathbb{E}[\|\nabla_{\mathbf{x}} f(\mathbf{x}, \mathbf{y}; \mathcal{B}^{f_x}) - \nabla_{\mathbf{x}} f(\mathbf{x}, \mathbf{y})\|^2] \leq \sigma_{f_x}^2, \quad \mathbb{E}[\|\nabla_{\mathbf{y}} f(\mathbf{x}, \mathbf{y}; \mathcal{B}^{f_y}) - \nabla_{\mathbf{y}} f(\mathbf{x}, \mathbf{y})\|^2] \leq \sigma_{f_y}^2.$$

In addition, the stochastic approximation used for estimating the Jacobian satisfies, for every j-th coordinate,

$$\mathbb{E}\left[\left\|\left(\nabla_{\mathbf{x}} \bar{\mathbf{y}}(\mathbf{x}; \mathcal{B}^{y}) - \nabla_{\mathbf{x}} \bar{\mathbf{y}}(\mathbf{x})\right) \mathbf{e}_j\right\|^2\right] \leq \sigma_y^2.$$

In Assumption 4.4, the variance terms vanish in the full-batch setting, and the Jacobian condition is an analytical mean-square regularity condition rather than an additional computational requirement.

Finally, we use the following stationarity criterion.

Definition 4.5 (ϵ-Stationarity)**.** A point $\mathbf{x}$ is an ϵ-stationary solution if $\mathbb{E}\left[\, \|\nabla \ell(\mathbf{x})\|^2 \,\right] \leq \epsilon$, where the expectation is taken over all randomness of the algorithm.

4.2 Main Convergence Results

We now state the main convergence result of LeGo-BiO for non-convex $\ell(\mathbf{x})$ in the deterministic setting, as established in Theorem 4.6:

Theorem 4.6 (Deterministic Setting)**.** *Under Assumptions 4.1–4.3, choose constant step-sizes $\alpha_t = \alpha$ for all $t \in \{0, 1, \ldots, T-1\}$, with $0 < \alpha \leq \bar{\alpha}$, where $\bar{\alpha}$ depends on L_l, d_u, d_l, B_{f_y} and L_y, and is specified in Theorem D.4. Then, the iterates generated by* LeGo-BiO *satisfy:*

$$\frac{1}{T} \sum_{t=0}^{T-1} \|\nabla \ell(\mathbf{x}_t)\|^2 = \mathcal{O}\left(\frac{\Delta_0}{T\alpha}\right) + \mathcal{O}\left(d_u \frac{1}{T} \sum_{t=0}^{T-1} \delta_{t+1}^2\right),$$

where $\Delta_0 = (\ell(\mathbf{x}_0) - \ell^*) + \|\nabla \ell(\mathbf{x}_0) - \sum_{j=1}^{d_u} [h_0^{f,j}]_j \mathbf{e}_j\|^2$.

The proof of Theorem 4.6 is provided in Section D. Theorem 4.6 shows that the convergence error consists of two terms: the standard optimization error $\mathcal{O}(\Delta_0/(T\alpha))$ and an additional bias term induced by the LL approximation error. This bias is controlled by the average squared error of the approximate LL solutions. Hence, the asymptotic convergence behavior depends on how the LL approximation error evolves over the course of optimization. If the LL approximation error δ_t is a non-vanishing constant, i.e., $\delta_t = \delta > 0$, for a fixed admissible step size, the optimization error vanishes as $T \to \infty$, but a residual bias remains. This does not imply divergence; rather, LeGo-BiO is guaranteed to reach an $\mathcal{O}\left(d_u \delta^2\right)$-stationary neighborhood. Such an error floor is expected when the LL oracle has persistent approximation bias, since exact stationarity of the original objective cannot generally be guaranteed from permanently inaccurate LL responses.

Moreover, if the accuracy of the returned LL responses improves sufficiently fast, the corresponding bias diminishes with T. Such behavior can naturally arise when the external LL oracle is progressively refined, for example, by increasing its computational budget or tightening its stopping tolerance across iterations. Importantly, LeGo-BiO does not require the UL learner to know or control the refinement mechanism; this condition only characterizes the quality of the responses returned by the external LL oracle. The following corollary makes this dependence explicit and establishes the resulting convergence rate under a decreasing LL approximation error.

Corollary 4.7. *Under the same conditions as in Theorem 4.6, choose $\alpha = \Theta(1)$ and assume that the LL approximation error decreases geometrically, i.e., $\delta_{t+1}^2 \leq (1-\gamma)\,\delta_t^2$ for some $0 < \gamma < 1$. Then, the iterates generated by* LeGo-BiO *satisfy: $\frac{1}{T}\sum_{t=0}^{T-1} \|\nabla \ell(\mathbf{x}_t)\|^2 = \mathcal{O}\left(T^{-1}\right)$.*

Corollary 4.7 shows that, when the LL approximation error decreases over time, LeGo-BiO achieves an $\mathcal{O}\left(T^{-1}\right)$ convergence rate for reaching a stationary solution. This rate matches the state-of-the-art deterministic convergence rate achieved by standard bilevel methods under stronger LL strong-convexity assumptions [15, 16].

We next present the main convergence result of LeGo-BiO for non-convex $\ell(\mathbf{x})$ in the stochastic setting, as given in Theorem 4.8:

Theorem 4.8 (Stochastic Setting)**.** *Under Assumptions 4.1–4.4, choose constant step-sizes $\alpha_t = \alpha$ for all $t \in \{0, 1, \ldots, T-1\}$, with $0 < \alpha \leq \bar{\alpha}$, where $\bar{\alpha}$ depends on L_l, d_u, d_l, B_{f_y} and L_y, and is specified in Theorem E.4. Then, the iterates generated by* LeGo-BiO *satisfy:*

$$\frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}\left[\|\nabla \ell(\mathbf{x}_t)\|^2\right] = \mathcal{O}\left(\frac{\Delta_0}{T\alpha}\right) + \mathcal{O}\left(d_u \frac{1}{T}\sum_{t=0}^{T-1} \delta_{t+1}^2 + \alpha d_u\left(\sigma_{f_x}^2 + \sigma_y^2 + \sigma_{f_y}^2 + \sigma_y^2\sigma_{f_y}^2\right)\right),$$

where $\Delta_0 = (\ell(\mathbf{x}_0) - \ell^*) + \|\nabla \ell(\mathbf{x}_0) - \sum_{j=1}^{d_u} [h_0^{f,j}]_j \mathbf{e}_j\|^2$.

The proof of Theorem 4.8 is provided in Section E. Compared with the deterministic bound, Theorem 4.8 contains additional variance terms arising from stochastic UL gradient estimation and stochastic Jacobian estimation. The product term $\sigma_y^2\sigma_{f_y}^2$ reflects the interaction between the stochasticity in the estimated Jacobian and the stochasticity in the UL gradient with respect to $\mathbf{y}$.

In Theorems 4.6 and 4.8, the LL approximation and stochastic-error terms scale linearly with the UL dimension d_u, while the LL dimension d_l enters the admissible step-size bound through the finite-difference error. Moreover, each full coordinate sweep requires $d_u + 1$ LL-response queries, yielding $\mathcal{O}(d_u T)$ oracle calls over T sweeps. Existing techniques for accelerating coordinate-wise methods can potentially be integrated into the LeGo-BiO framework to further improve its efficiency. We also discuss extensions of our convergence analysis to the case of set-valued, i.e., non-unique, LL solutions. Further details on both aspects are provided in Appendix B.

The following corollary gives the resulting stochastic convergence rate under a decreasing LL approximation error.

Corollary 4.9. *Under the same conditions as in Theorem 4.8, choose $\alpha = \Theta\left(T^{-1/2}\right)$ and suppose that the LL approximation error decreases geometrically, i.e., $\delta_{t+1}^2 \leq (1-\gamma)\,\delta_t^2$ for some $0 < \gamma < 1$. Then, the iterates generated by* LeGo-BiO *satisfy: $\frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}\left[\|\nabla \ell(\mathbf{x}_t)\|^2\right] = \mathcal{O}\left(T^{-1/2}\right)$.*

Corollary 4.9 demonstrates that LeGo-BiO achieves an $\mathcal{O}\left(T^{-1/2}\right)$ convergence rate for reaching a stationary solution. This rate matches the state-of-the-art stochastic convergence rate achieved by standard bilevel methods under stronger LL strong-convexity assumptions [15–19].

5 Numerical Experiments

In this section, we evaluate the performance of LeGo-BiO on three different bilevel optimization tasks: 1) deep hyper-representation; 2) adversarial training; and 3) data weighting for LLM alignment. The experiments are designed to examine two questions: whether LeGo-BiO can achieve competitive performance under a weaker LL information model than existing bilevel methods, and how sensitive the proposed pseudo-hypergradient is to the quality of the approximate LL solution. Due to space limitations, additional experimental details and results are provided in Appendix A.

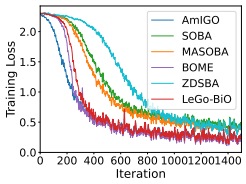
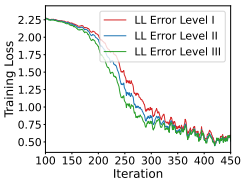
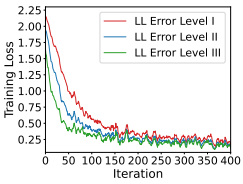


Figure 1: Performance among different algorithms on Task 1.



(a) SGD



(b) SciPy

Figure 2: Performance of LeGo-BiO under different LL approximation errors on Task 1 (I is the largest).

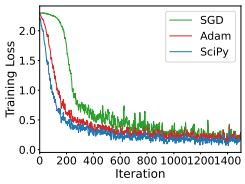


Figure 3: Performance of LeGo-BiO under different LL solvers on Task 1.

Task 1) Deep Hyper-Representation. We first evaluate LeGo-BiO on a deep hyper-representation task, where the goal is to learn a feature representation such that the LL classifier induced by this representation performs well on image classification. This task naturally reflects the LL-agnostic setting when the UL learner can observe the classifier returned by an external training procedure (an SGD-based classifier-training routine in our experiments), but does not have access to the LL training objective, its derivatives, or the procedure that generated the classifier. We conduct experiments on the MNIST dataset using a 5-layer LeNet network [41].

We compare LeGo-BiO with the AID-based methods AmIGO [18], SOBA [19], and MA-SOBA [42], the fully first-order method BOME [31], and the zeroth-order method ZDSBA [34]. AmIGO, SOBA, and MA-SOBA use LL derivative information to approximate hypergradients, while BOME still requires LL first-order gradients; ZDSBA instead constructs zeroth-order estimates from function-value queries of the UL and LL objectives. In contrast, LeGo-BiO uses only UL gradients and LL response queries.

Figure 1 compares the performance of different algorithms on the deep hyper-representation task. LeGo-BiO achieves a training loss comparable to those of AmIGO and BOME and lower than those of SOBA, MA-SOBA, and ZDSBA. These results demonstrate that, despite having no access to the LL objective or its derivative information, LeGo-BiO can construct an effective pseudo-hypergradient using only UL gradients and LL response queries alone. Moreover, the comparison with ZDSBA highlights the practical advantage of preserving the available UL first-order information rather than relying entirely on zeroth-order estimation.

We further study the robustness of LeGo-BiO under different levels of LL solution approximation error. Figure 2 reports the performance of LeGo-BiO under different LL approximation error levels, where Level I corresponds to the largest error. The results show that the final training losses remain comparable across different error levels. At the same time, when the LL approximation error is smaller, LeGo-BiO requires fewer iterations to reach the same training loss, due to a more accurate pseudo-hypergradient estimate. These results demonstrate that LeGo-BiO is robust to moderate LL approximation errors. Moreover, these observations are consistent with our theoretical finding, where the convergence depends explicitly on the LL approximation error.

To evaluate whether LeGo-BiO depends on a particular LL black-box solver, we have conducted a solver-diversity experiment using two simulated LL solvers, SGD and Adam, and an external SciPy-based solver. As shown in Figure 3, LeGo-BiO remains effective across all three solvers, and SciPy achieves the lowest training loss. These results indicate that LeGo-BiO can operate effectively with heterogeneous black-box response generators.

Task 2) Adversarial Training. We next evaluate LeGo-BiO on an adversarial training task, which aims to train a model that remains accurate under adversarial perturbations. In this bilevel formulation, the UL objective corresponds to the defender's model-training objective, while the LL problem corresponds to the attacker that returns adversarial perturbations against the current model. This task is well aligned with the LL-agnostic setting when the defender observes the perturbations generated by an external or black-box attacker

Table 1: Performance among different algorithms on Task 2.

	LeGo-BiO	FAST-BAT	FAST-AT	FAST-AT-GA
Standard Accuracy (%)	77.18	76.04	73.34	71.40
Robust Accuracy (%)	39.08	39.84	32.56	36.44

(implemented using projected gradient descent in our experiments), but cannot access the attack objective or differentiate through the attack-generation procedure. We conduct experiments by training the PreActResNet-18 [43] model on the CIFAR-10 [44] dataset.

We compare LeGo-BiO with three fast adversarial training baselines: FAST-AT [45], FAST-AT-GA [46], and FAST-BAT [10]. These baselines operate under a stronger information model, assuming that the attacker's objective is known to the defender and require access to it for specification or differentiation during training. In contrast, LeGo-BiO only uses the perturbations returned by the attacker together with UL gradient information, and does not require access to the attack objective.

Table 1 reports standard accuracy on clean test images and robust accuracy under adversarial perturbations. Both LeGo-BiO and FAST-BAT outperform FAST-AT and FAST-AT-GA in terms of both standard and robust accuracy, which supports the effectiveness of bilevel modeling for adversarial training. Moreover, LeGo-BiO achieves the highest standard accuracy among all compared methods, while attaining robust accuracy comparable to FAST-BAT and higher than FAST-AT and FAST-AT-GA. These results indicate that, despite not using the LL attack objective and its derivative information, LeGo-BiO can construct an accurate pseudo-hypergradient that approximates the UL-gradient comparably to hypergradient-based methods, thereby achieving a favorable standard-robust accuracy trade-off under a weaker information model.

Task 3) Data Weighting for LLM Alignment. We then evaluate LeGo-BiO on a data-weighting task for LLM Alignment, with the goal of learning data weights for LLM alignment. Specifically, we align the Llama-3.2-1B-Instruct model [47] on the HH-RLHF dataset [48]. For LeGo-BiO, LLM fine-tuning is treated as an external response oracle that maps data weights to fine-tuned LLM parameters; LeGo-BiO neither accesses the LL training objective nor differentiates through the fine-tuning procedure.

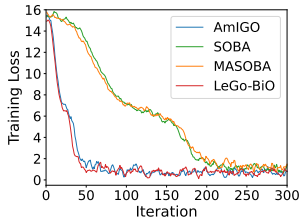


Figure 4: Performance among different algorithms on Task 3.

We compare LeGo-BiO with the white-box methods AmIGO [18], SOBA [19], and MA-SOBA [42], which have full access to the LL objective and its derivatives. This comparison therefore evaluates optimization performance under different information-access assumptions.

As shown in Figure 4, LeGo-BiO achieves a training loss comparable to those of AmIGO, SOBA and MA-SOBA. Moreover, LeGo-BiO and AmIGO converge substantially faster than SOBA and MA-SOBA, likely because they obtain more accurate LL solutions. These results demonstrate that, despite having no access to the LL objective or its derivative information, LeGo-BiO can construct an effective pseudo-hypergradient using only UL gradients and LL response queries.

6 Conclusion

This paper studied lower-level (LL) agnostic bilevel optimization, where the upper-level (UL) learner observes only an approximate LL solution $\bar{\mathbf{y}}(\mathbf{x})$ and has no access to the LL objective, its gradients, Hessian/Jacobian information, or the procedure that generates the response. We proposed LeGo-BiO, a coordinate-wise pseudo-hypergradient method that estimates the missing Jacobian $\nabla_{\mathbf{x}}\bar{\mathbf{y}}$ through finite differences of $\bar{\mathbf{y}}(\cdot)$ while retaining the available first-order gradients of the UL objective. The coordinate-wise construction isolates the Jacobian column required for each UL-gradient component, thereby avoiding the directional-projection limitation of full-parameter responses and the noise sensitivity of finite-differencing the entire UL-gradient. We established convergence guarantees for nonconvex UL objectives in both deterministic and stochastic settings under an LL error-bound/Polyak–Łojasiewicz-type condition, without requiring LL strong convexity. Our results yield an $\mathcal{O}(T^{-1})$ deterministic rate and an $\mathcal{O}(T^{-1/2})$ stochastic rate when the LL approximation error decreases geometrically. Experiments on deep hyper-representation, adversarial training, and data weighting for LLM alignment show that LeGo-BiO achieves competitive or superior performance despite having no access to the LL objective. These results suggest that LL response-based UL-gradient approximation offers a promising direction for bilevel optimization with black-box solvers, simulators, and learned LL modules.

Acknowledgements

This work is supported in part by NSF grants CAREER CNS-2110259, CNS-2112471, ECCS-233104, DARPA YFA D24AP00265, DARPA HR0011-25-2-0019, ONR grant N00014-24-1-2729, and a Meta Research Award INB3267726.

References

[1] F. Bao, G. Wu, C. Li, J. Zhu, and B. Zhang, "Stability and generalization of bilevel programming in hyperparameter optimization," *Advances in neural information processing systems*, vol. 34, pp. 4529–4541, 2021.

[2] L. Franceschi, P. Frasconi, S. Salzo, R. Grazzi, and M. Pontil, "Bilevel programming for hyperparameter optimization and meta-learning," in *International Conference on Machine Learning*. PMLR, 2018, pp. 1568–1577.

[3] A. Rajeswaran, C. Finn, S. M. Kakade, and S. Levine, "Meta-learning with implicit gradients," *Advances in neural information processing systems*, vol. 32, 2019.

[4] K. Ji, J. D. Lee, Y. Liang, and H. V. Poor, "Convergence of meta-learning with task-specific adaptation over partial parameters," *Advances in Neural Information Processing Systems*, vol. 33, pp. 11 490–11 500, 2020.

[5] M. Hong, H.-T. Wai, Z. Wang, and Z. Yang, "A two-timescale stochastic algorithm framework for bilevel optimization: Complexity analysis and application to actor-critic," *SIAM Journal on Optimization*, vol. 33, no. 1, pp. 147–180, 2023.

[6] H. Shen, Z. Yang, and T. Chen, "Principled penalty-based methods for bilevel reinforcement learning and rlhf," *Journal of Machine Learning Research*, vol. 26, no. 114, pp. 1–49, 2025.

[7] D. Lian, Y. Zheng, Y. Xu, Y. Lu, L. Lin, P. Zhao, J. Huang, and S. Gao, "Towards fast adaptation of neural architectures with meta learning," in *International Conference on Learning Representations*, 2019.

[8] Y. Hu, X. Wu, and R. He, "Tf-nas: Rethinking three search freedoms of latency-constrained differentiable neural architecture search," in *European Conference on Computer Vision*. Springer, 2020, pp. 123–139.

[9] Y. Tian, L. Shen, G. Su, Z. Li, and W. Liu, "Alphagan: Fully differentiable architecture search for generative adversarial networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 10, pp. 6752–6766, 2021.

[10] Y. Zhang, G. Zhang, P. Khanduri, M. Hong, S. Chang, and S. Liu, "Revisiting and advancing fast adversarial training through the lens of bi-level optimization," in *International Conference on Machine Learning*. PMLR, 2022, pp. 26 693–26 712.

[11] Y. Li, L. Song, X. Wu, R. He, and T. Tan, "Learning a bi-level adversarial network with global and local perception for makeup-invariant face verification," *Pattern Recognition*, vol. 90, pp. 99–108, 2019.

[12] A. Shaban, C.-A. Cheng, N. Hatch, and B. Boots, "Truncated back-propagation for bilevel optimization," in *The 22nd International Conference on Artificial Intelligence and Statistics*. PMLR, 2019, pp. 1723–1732.

[13] B. Lecouat, J. Ponce, and J. Mairal, "A flexible framework for designing trainable priors with adaptive smoothing and game encoding," *Advances in Neural Information Processing Systems*, vol. 33, pp. 15 664–15 675, 2020.

[14] J. Mairal, F. Bach, and J. Ponce, "Task-driven dictionary learning," *IEEE transactions on pattern analysis and machine intelligence*, vol. 34, no. 4, pp. 791–804, 2011.

[15] S. Ghadimi and M. Wang, "Approximation methods for bilevel programming," *arXiv preprint arXiv:1802.02246*, 2018.

[16] K. Ji, J. Yang, and Y. Liang, "Bilevel optimization: Convergence analysis and enhanced design," in *International Conference on Machine Learning*. PMLR, 2021, pp. 4882–4892.

[17] T. Chen, Y. Sun, and W. Yin, "Closing the gap: Tighter analysis of alternating stochastic gradient methods for bilevel problems," *Advances in Neural Information Processing Systems*, vol. 34, pp. 25 294–25 307, 2021.

[18] M. Arbel and J. Mairal, "Amortized implicit differentiation for stochastic bilevel optimization," in *International Conference on Learning Representations*, 2022.

[19] M. Dagréou, P. Ablin, S. Vaiter, and T. Moreau, "A framework for bilevel optimization that enables stochastic and global variance reduction algorithms," *Advances in Neural Information Processing Systems*, vol. 35, pp. 26 698–26 710, 2022.

[20] P. Khanduri, S. Zeng, M. Hong, H.-T. Wai, Z. Wang, and Z. Yang, "A near-optimal algorithm for stochastic bilevel optimization via double-momentum," *Advances in Neural Information Processing Systems*, vol. 34, pp. 30 271–30 283, 2021.

[21] F. Pedregosa, "Hyperparameter optimization with approximate gradient," in *International conference on machine learning*. PMLR, 2016, pp. 737–746.

[22] W. Rudin *et al.*, *Principles of mathematical analysis*. McGraw-hill New York, 1976, vol. 3.

[23] J. Domke, "Generic methods for optimization-based modeling," in *Artificial Intelligence and Statistics*. PMLR, 2012, pp. 318–326.

[24] D. Maclaurin, D. Duvenaud, and R. Adams, "Gradient-based hyperparameter optimization through reversible learning," in *International conference on machine learning*. PMLR, 2015, pp. 2113–2122.

[25] L. Franceschi, M. Donini, P. Frasconi, and M. Pontil, "Forward and reverse gradient-based hyperparameter optimization," in *International Conference on Machine Learning*. PMLR, 2017, pp. 1165–1173.

[26] R. Grazzi, L. Franceschi, M. Pontil, and S. Salzo, "On the iteration complexity of hypergradient computation," in *International Conference on Machine Learning*. PMLR, 2020, pp. 3748–3758.

[27] M. Mackay, P. Vicol, J. Lorraine, D. Duvenaud, and R. Grosse, "Self-tuning networks: Bilevel optimization of hyperparameters using structured best-response functions," in *International Conference on Learning Representations*, 2019.

[28] Y. Yang, P. Xiao, and K. Ji, "Achieving $\mathcal{O}\left(\epsilon^{-1.5}\right)$ complexity in hessian/jacobian-free stochastic bilevel optimization," *Advances in Neural Information Processing Systems*, 2023.

[29] J. Kwon, D. Kwon, S. Wright, and R. D. Nowak, "A fully first-order method for stochastic bilevel optimization," in *International Conference on Machine Learning*. PMLR, 2023, pp. 18 083–18 113.

[30] ——, "On penalty methods for nonconvex bilevel optimization and first-order stochastic approximation," in *The Twelfth International Conference on Learning Representations*, 2024.

[31] B. Liu, M. Ye, S. Wright, P. Stone, and Q. Liu, "Bome! bilevel optimization made easy: A simple first-order approach," *Advances in neural information processing systems*, vol. 35, pp. 17 248–17 262, 2022.

[32] H. Shen and T. Chen, "On penalty-based bilevel gradient descent method," in *International conference on machine learning*. PMLR, 2023, pp. 30 992–31 015.

[33] X. Song, W. Gao, Y. Yang, K. Choromanski, A. Pacchiano, and Y. Tang, "Es-maml: Simple hessian-free meta learning," in *International Conference on Learning Representations*, 2020.

[34] A. Aghasi and S. Ghadimi, "Fully zeroth-order bilevel programming via gaussian smoothing," *Journal of Optimization Theory and Applications*, vol. 205, no. 2, p. 31, 2025.

[35] A. Aghasi, J. Kwon, and S. Ghadimi, "Optimal zeroth-order bilevel optimization," *arXiv preprint arXiv:2510.03646*, 2025.

[36] M. Rando and S. Vaiter, "Zoba: An efficient single-loop zeroth-order bilevel optimization algorithm," *arXiv preprint arXiv:2601.21836*, 2026.

[37] Y. Diouane, V. Kungurtsev, F. Rinaldi, and D. Zeffiro, "Inexact direct-search methods for bilevel optimization problems," *Computational Optimization and Applications*, vol. 88, no. 2, pp. 469–490, 2024.

[38] J. C. Duchi, M. I. Jordan, M. J. Wainwright, and A. Wibisono, "Optimal rates for zero-order convex optimization: The power of two function evaluations," *IEEE Transactions on Information Theory*, vol. 61, no. 5, pp. 2788–2806, 2015.

[39] Z. Liu, X. Zhang, P. Khanduri, S. Lu, and J. Liu, "Interact: achieving low sample and communication complexities in decentralized bilevel learning over networks," in *Proceedings of the Twenty-Third International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing*, 2022, pp. 61–70.

[40] P. Qiu, Y. Li, Z. Liu, P. Khanduri, J. Liu, N. B. Shroff, E. S. Bentley, and K. Turck, "Diamond: Taming sample and communication complexities in decentralized bilevel optimization," in *IEEE INFOCOM 2023-IEEE conference on computer communications*. IEEE, 2023, pp. 1–10.

[41] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[42] X. Chen, T. Xiao, and K. Balasubramanian, "Optimal algorithms for stochastic bilevel optimization under relaxed smoothness conditions," *Journal of Machine Learning Research*, vol. 25, no. 151, pp. 1–51, 2024.

[43] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *European conference on computer vision*. Springer, 2016, pp. 630–645.

[44] A. Krizhevsky, "Learning multiple layers of features from tiny images," *Master's thesis, University of Tront*, 2009.

[45] E. Wong, L. Rice, and J. Z. Kolter, "Fast is better than free: Revisiting adversarial training," in *International Conference on Learning Representations*, 2020.

[46] M. Andriushchenko and N. Flammarion, "Understanding and improving fast adversarial training," *Advances in Neural Information Processing Systems*, vol. 33, pp. 16 048–16 059, 2020.

[47] Meta. (2024) meta-llama/llama-3.2-1b-instruct. [Online]. Available: https://huggingface.co/meta-llama/Llama-3.2-1B-Instruct

[48] Y. Bai, A. Jones, K. Ndousse, A. Askell, A. Chen, N. DasSarma, D. Drain, S. Fort, D. Ganguli, T. Henighan *et al.*, "Training a helpful and harmless assistant with reinforcement learning from human feedback," *arXiv preprint arXiv:2204.05862*, 2022.

A Experimental Details and Additional Results

In this section, we present additional experimental results and provide further details on the experimental setup and implementation.

A.1 Additional Experimental Results

A.1.1 Deep Hyper-Representation

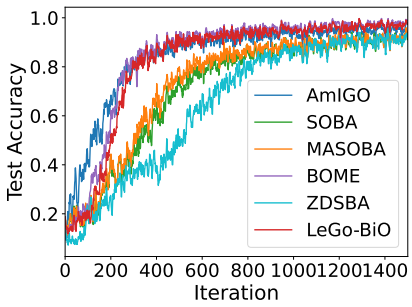
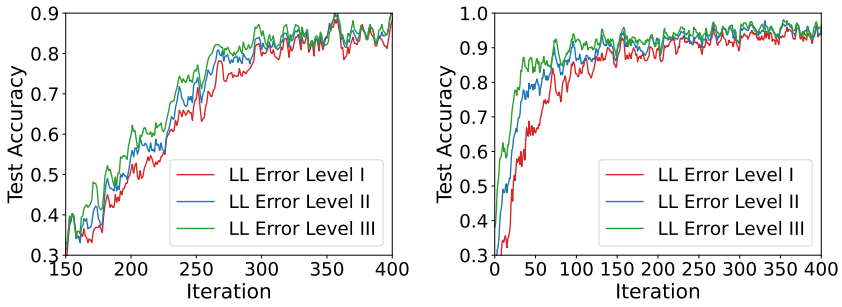


Figure 5: Comparison of test accuracy across different algorithms on Task 1.

Figure 5 compares the test accuracy of different algorithms on the deep hyper-representation task. LeGo-BiO achieves test accuracy comparable to AmIGO and BOME, while consistently outperforming SOBA, MA-SOBA, and ZDSBA. Together with the training-loss results in Figure 1, these results demonstrate that LeGo-BiO achieves competitive optimization and test performance despite having no access to the LL objective or its derivative information. In particular, the comparison with ZDSBA further demonstrates the benefit of preserving the available UL first-order information instead of estimating the entire UL gradient through zeroth-order function-value queries.



(a) SGD (b) SciPy

Figure 6: Test accuracy of LeGo-BiO under different LL approximation errors on Task 1 (I is the largest).

We further evaluate the effect of different LL approximation errors on the test accuracy of LeGo-BiO. As shown in Figure 6, LeGo-BiO achieves similar final test accuracy across all three LL error levels for both the SGD-based and SciPy-based LL solvers. Smaller LL approximation errors generally lead to faster improvements in test accuracy, which is consistent with the training-loss results in Figure 2. These observations indicate that LeGo-BiO is robust to moderate LL approximation errors, while more accurate LL responses can improve its convergence behavior.

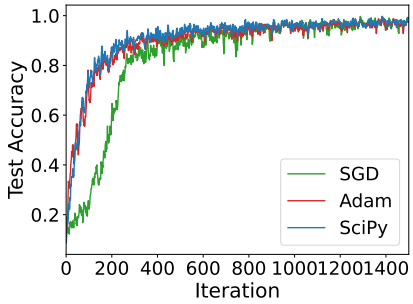


Figure 7: Test accuracy of LeGo-BiO under different LL solvers on Task 1.

Finally, Figure 7 reports the test accuracy of LeGo-BiO under different LL solvers. LeGo-BiO achieves comparable final test accuracy when the LL responses are generated by SGD, Adam, or SciPy, although their transient convergence behaviors differ. This result complements the training-loss comparison in Figure 3 and further shows that LeGo-BiO does not rely on a particular LL solver. Instead, it can effectively utilize approximate responses produced by heterogeneous black-box LL procedures.

A.2 Experimental Details

A.2.1 Deep Hyper-Representation

Deep hyper-representation learns a feature extractor by optimizing the validation performance of a classifier trained on the induced features. Let λ denote the parameters of the feature extractor ϕ_λ and let w denote the classifier parameters. The bilevel objective is

$$\min_{\lambda} \; f(\lambda, w^\star(\lambda)) := \frac{1}{|\mathcal{D}_{\text{val}}|} \sum_{(a_i,b_i)\in\mathcal{D}_{\text{val}}} \ell_{\text{ce}}\big(w^\star(\lambda); \phi_\lambda(a_i), b_i\big),$$

$$\text{s.t.} \; w^\star(\lambda) = \arg\min_{w} \; g(\lambda, w) := \frac{1}{|\mathcal{D}_{\text{tr}}|} \sum_{(a_i,b_i)\in\mathcal{D}_{\text{tr}}} \ell_{\text{ce}}\big(w; \phi_\lambda(a_i), b_i\big),$$

where (a_i, b_i) denotes a data sample, $\mathcal{D}_{val}$ and $\mathcal{D}_{tr}$ are the validation dataset and the training dataset, ℓ_{ce} corresponds to the cross-entropy loss.

In the LL-agnostic setting, LeGo-BiO does not access $g(\lambda, w)$ or differentiate through the LL classifier-training routine. For each queried representation parameter λ, the algorithm only receives an approximate classifier $\bar{w}(\lambda)$, which is produced by an external SGD-based classifier-training routine in our experiments, and uses coordinate-wise response differences of $\bar{w}(\cdot)$ to estimate the missing Jacobian contribution in the UL gradient.

We conduct experiments using a LeNet-5 network [41] (MIT License) on the MNIST dataset [41] (CC BY-SA 3.0 License) with batch size 64. Here, the UL variable $\mathbf{x} = \lambda$ has dimension 60,856, while the LL variable $\mathbf{y} = w$ has dimension 850. All methods use the same data preprocessing, model architecture, and train/test split. We compare LeGo-BiO against AmIGO [18], SOBA [19], and MA-SOBA [42]. For LeGo-BiO, we set the step size to $\alpha = 0.01$ and simulate the LL solver using stochastic gradient descent with step size $\beta = 0.001$. For both AmIGO and MA-SOBA, we set the step sizes to $\alpha = 0.01$, $\beta = 0.1$, and $\gamma = 0.1$. For AmIGO, we use 8 $\mathbf{y}$-update iterations and 4 $\mathbf{z}$-update iterations. For MA-SOBA, we set the momentum coefficient to 0.9. For SOBA, we set the step sizes to $\alpha = 0.01$, $\beta = 0.1$, and $\gamma = 0.01$. All deep hyper-representation experiments are conducted on an NVIDIA H200 NVL GPU.

A.2.2 Adversarial Training

We formulate adversarial training as a bilevel optimization problem, where the UL variable is the classifier parameter θ and the LL response is the adversarial perturbation. For each data-label pair (a, b), the LL attack problem is

$$\delta^\star(\theta) \in \arg\max_{\delta \in \mathcal{C}_\epsilon(a)} \ell_{\mathrm{ce}}(\theta; a + \delta, b), \quad \mathcal{C}_\epsilon(a) := \{\delta : \|\delta\|_\infty \le \varepsilon,\ a + \delta \in [0,1]^d\}, \tag{6}$$

where ε is the perturbation budget and ℓ_{ce} is the cross-entropy loss. The UL objective is the empirical robust training loss

$$\min_\theta \ \frac{1}{|\mathcal{D}_{\mathrm{tr}}|} \sum_{(a_i, b_i) \in \mathcal{D}_{\mathrm{tr}}} \ell_{\mathrm{ce}}\big(\theta; a_i + \delta_i^\star(\theta), b_i\big).$$

Although Eq. (6) defines the target LL response, LeGo-BiO does not use the LL attack objective or its derivatives. Instead, the attack routine is treated as an external response oracle: for a queried classifier parameter θ, the oracle returns an approximate perturbation $\bar{\delta}(\theta)$ based on projected gradient descent. The returned perturbation is detached from the attack-generation procedure, and LeGo-BiO uses finite differences of the returned perturbations to estimate the missing Jacobian information.

We conduct experiments using the PreActResNet-18 [43] model (MIT License) on the CIFAR-10 dataset [44] (CC BY 4.0 License) with batch size 128. Here, the UL variable $\mathbf{x} = \theta$ has dimension 11,172,170, while the LL variable $\mathbf{y} = \delta$ has dimension 3,072. The perturbation set is the ℓ_∞ ball with training and evaluation budget $\varepsilon = 10/255$. All methods use the same data preprocessing, model architecture, and train/test split. For standard accuracy, we evaluate the trained classifier on clean test images. For robust accuracy, we evaluate the classifier on adversarially perturbed test images generated under the same ℓ_∞ budget. Specifically, we use PGD-50-10, i.e., a 50-step projected gradient descent attack with 10 random restarts.

We compare LeGo-BiO against FAST-AT [45], FAST-AT-GA [46], and FAST-BAT [10]. For FAST-AT, we use a constant step size of 0.1. For both FAST-AT-GA and FAST-BAT, we use a constant step size of 0.05. For LeGo-BiO, we use a cyclic step size ranging from 0 to 2.0. All the experiments are conducted on an NVIDIA H200 NVL GPU.

A.3 Data Weighting for LLM Alignment

In this subsection, we describe the experimental setup for the data weighting task for LLM alignment. The goal is to learn a set of training-data weights such that fine-tuning the LLM on the reweighted training data yields low validation loss.

We partition the training data into N_T groups, denoted by $\mathcal{T}_n$, where $n = 1, 2, \ldots, N_T$. Each data sample $i = 1, 2, \ldots, |\mathcal{T}_n|$ consists of a prompt-response pair $\{p_{n,i}, r_{n,i}\}$ and its corresponding labeled score $s_{n,i}$. We introduce $\mathbf{x} = [x_1, x_2, \ldots, x_{N_T}]^\top$ as a vector of raw dataset weights and convert it into normalized weights through a softmax transformation. Let $\mathbf{y} \in \mathbb{R}^l$ denote the trainable

parameters of the LLM. The UL problem optimizes the dataset weights based on the loss evaluated on a validation set $\mathcal{V}$, while the LL problem fine-tunes the LLM using the weighted training data.

Accordingly, the bilevel optimization problem is formulated as

$$\min_{\mathbf{x}\in\mathbb{R}^{N_T}} \sum_{i=1}^{|\mathcal{V}|} \mathcal{L}(\tilde{r}_{0,i}, r_{0,i}; p_{0,i}, \mathbf{y}^*(\mathbf{x}))$$
$$\text{s.t.} \quad \mathbf{y}^*(\mathbf{x}) = \arg\min_{\mathbf{y}\in\mathbb{R}^l} \sum_{n=1}^{N_T} \frac{e^{x_n}}{\sum_{n'=1}^{N_T} e^{x_{n'}}} \sum_{i=1}^{|\mathcal{T}_n|} \mathcal{L}(\tilde{r}_{n,i}, r_{n,i}; p_{n,i}, \mathbf{y}),$$

where $\mathcal{L}(\tilde{r}_{n,i}, r_{n,i}; p_{n,i}, \mathbf{y})$ is the loss between the true response $r_{n,i}$ and the response $\tilde{r}_{n,i}$ generated by the LLM parameterized by $\mathbf{y}$ for prompt $p_{n,i}$.

We use Llama-3.2-1B-Instruct [47] (Llama 3.2 Community License) as the base model and fine-tune it using the low-rank adaptation (LoRA) technique with rank 8. Experiments are conducted on the Anthropic HH-RLHF dataset [48] (MIT License). The training samples are divided into two groups according to their preference labels, namely *chosen* and *rejected*, so that $N_T = 2$. The validation set contains only *chosen* responses. Consequently, an effective data-weighting strategy is expected to assign relatively larger weight to the *chosen* training group, thereby reducing the validation loss. Cross-entropy loss is used for both the UL and LL objectives.

For AmIGO [18], we use 5 update steps for both $\mathbf{y}$ and $\mathbf{z}$. For MA-SOBA [42], we set $\mu = 0.8$. All methods use step sizes $\alpha = 5 \times 10^{-3}$, $\beta = 2 \times 10^{-4}$, and $\gamma = 3 \times 10^{-4}$, with a batch size of 32. All experiments are conducted on an NVIDIA H200 NVL GPU.

B Discussion

B.1 Coordinate-Wise Design

Due to its coordinate-wise design, a full LeGo-BiO sweep requires $\mathcal{O}(d_u)$ LL-response queries, and this dimension dependence also appears in our convergence bounds.

This cost reflects an explicit "exploitation–exploration dilemma" in our LL-agnostic setting, which shows that the black-box bilevel optimization problem is *fundamentally hard*. A single full-parameter response difference provides only one directional projection of the Jacobian and therefore cannot recover the full pseudo-hypergradient without additional perturbations/explorations or structural assumptions. The coordinate-wise queries instead isolate each component, avoiding this projection ambiguity while preserving the available UL first-order information.

Consequently, the current full-sweep method is best suited to problems with a moderate effective UL dimension, or settings in which $\mathbf{x}$ has a structured low-dimensional parameterization, such as hyperparameters, layer-wise coefficients, or adapter parameters. The LL queries can also be batched or evaluated in parallel (if given an appropriate distributed computing environment) once the query points are formed, reducing the LL response latency.

For large networks with an appropriate symmetry, a natural scalable extension is to query only $k \ll d_u$ randomly selected coordinates per iteration, reducing the per-iteration LL-query cost to $\mathcal{O}(k)$. Another option is to apply LeGo-BiO to a structured low-dimensional reparameterization or develop block-wise or layer-wise estimators. These variants introduce additional stochastic or projection error and require a separate convergence analysis, which we identify as an important direction for future work.

B.2 Extension to Set-Valued LL Solutions

It is possible to extend our analysis to the *set-valued LL solution setting* (i.e., *non-unique* LL solutions) under a bounded diameter condition. Specifically, assume the LL solution set has bounded dimension D, and the oracle selects a differentiable branch $\mathbf{y}^\dagger(\mathbf{x})$ satisfying $\left\|\bar{\mathbf{y}}(\mathbf{x}_t) - \mathbf{y}^\dagger(\mathbf{x}_t)\right\| \le C_D \delta_t$, where C_D depends on the solution-set geometry, together with the corresponding Jacobian regularity of $\mathbf{y}^\dagger(\mathbf{x})$. Replacing $\mathbf{y}^*(\mathbf{x})$ with $\mathbf{y}^\dagger(\mathbf{x})$ and Assumption 4.3(1) accordingly, the existing proof will carry over with the LL approximation-error terms scaled by C_D^2.

C Supporting Lemmas

Lemma C.1. *Under Assumptions 4.1 and 4.2, we have* $\|\nabla \ell(\mathbf{x}_1) - \nabla \ell(\mathbf{x}_2)\| \leq L_l \|\mathbf{x}_1 - \mathbf{x}_2\|$*, for all* $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^{d_u}$*, where the Lipschitz constant is defined as* $L_l = L_f + \kappa L_f L_{g_y}$*.*

Proof. We have

$$\begin{aligned}
&\|\nabla \ell(\mathbf{x}_1) - \nabla \ell(\mathbf{x}_2)\| \\
&= \|\nabla f(\mathbf{x}_1, \mathbf{y}^*(\mathbf{x}_1)) - \nabla f(\mathbf{x}_2, \mathbf{y}^*(\mathbf{x}_2))\| \\
&\overset{(a)}{\leq} L_f \|\mathbf{x}_1 - \mathbf{x}_2\| + L_f \|\mathbf{y}^*(\mathbf{x}_1) - \mathbf{y}^*(\mathbf{x}_2)\|,
\end{aligned} \tag{7}$$

where (a) uses the Lipschitzness of $\nabla f(\mathbf{x}, \mathbf{y})$ (see Assumption 4.1).

Then, we bound $\|\mathbf{y}^*(\mathbf{x}_1) - \mathbf{y}^*(\mathbf{x}_2)\|$ in Eq. (7).

$$\begin{aligned}
&\|\mathbf{y}^*(\mathbf{x}_1) - \mathbf{y}^*(\mathbf{x}_2)\| \\
&\overset{(a)}{\leq} \kappa \|\nabla_{\mathbf{y}} g(\mathbf{x}_1, \mathbf{y}^*(\mathbf{x}_2))\| \\
&\overset{(b)}{=} \kappa \|\nabla_{\mathbf{y}} g(\mathbf{x}_1, \mathbf{y}^*(\mathbf{x}_2)) - \nabla_{\mathbf{y}} g(\mathbf{x}_2, \mathbf{y}^*(\mathbf{x}_2))\| \\
&\overset{(c)}{\leq} \kappa L_{g_y} \|\mathbf{x}_1 - \mathbf{x}_2\|,
\end{aligned} \tag{8}$$

where (a) and (c) are because of Assumption 4.2. (b) follows from $\nabla_{\mathbf{y}} g(\mathbf{x}_2, \mathbf{y}^*(\mathbf{x}_2)) = 0$.

Substituting Eq. (8) into Eq. (7) completes the proof. □

Lemma C.2. *For any function* $\varphi(x) : \mathbb{R} \to \mathbb{R}$*, define one-direction zeroth-order estimator as* $\widehat{\nabla}\varphi(x_1) = \frac{1}{h}(\varphi(x_2) - \varphi(x_1))$*, where* $h = x_2 - x_1$*. Assuming* $\nabla\varphi(x)$ *is Lipschitz with* $L_\varphi \geq 0$*, we have*

$$\left\| \widehat{\nabla}\varphi(x_1) - \nabla\varphi(x_1) \right\|^2 \leq \frac{h^2 L_\varphi^2}{4},$$

for all $x_1 \in \mathbb{R}$*.*

Proof. For $h \neq 0$, we have

$$\begin{aligned}
&\left\| \widehat{\nabla}\varphi(x_1) - \nabla\varphi(x_1) \right\| \\
&= \left\| \frac{1}{h}[\varphi(x_1 + h) - \varphi(x_1)] - \nabla\varphi(x_1) \right\| \\
&\overset{(a)}{=} \left\| \int_0^1 \big(\nabla\varphi(x_1 + \rho h) - \nabla\varphi(x_1)\big) d\rho \right\| \\
&\overset{(b)}{\leq} \int_0^1 L_\varphi \rho |h| d\rho \\
&= \frac{1}{2}|h| L_\varphi,
\end{aligned}$$

where (a) follows from the fundamental theorem of calculus. (b) utilizes the assumption that $\nabla\varphi(x)$ is Lipschitz.

Squaring proves the result. □

Lemma C.3. *Define* $q_t^j = \frac{\bar{\mathbf{y}}(\mathbf{x}_{t-1}) - \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j)}{x_{t-1}^j - x_t^j}$ *and* $\widehat{q}_t^j = \frac{\bar{\mathbf{y}}(\mathbf{x}_{t-1}; \mathcal{B}^y) - \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j; \mathcal{B}^y)}{x_{t-1}^j - x_t^j}$*. Under Assumptions 4.3 and 4.4, we have*

$$\left\| q_t^j \right\| \leq B_y, \qquad \mathbb{E}\left[\left\| \widehat{q}_t^j - q_t^j \right\|^2 \right] \leq \sigma_y^2.$$

Proof. Apply the mean-value theorem along the line segment connecting $\widetilde{\mathbf{x}}_t^j$ and $\mathbf{x}_{t-1}$.
Define the line segment

$$\nu_{t,j}(\rho) = \widetilde{\mathbf{x}}_t^j + \rho\left(\mathbf{x}_{t-1} - \widetilde{\mathbf{x}}_t^j\right), \qquad \rho \in [0,1].$$

Since $\mathbf{x}_{t-1} - \widetilde{\mathbf{x}}_t^j = \left(x_{t-1}^j - x_t^j\right)\mathbf{e}_j$, we have

$$\nu_{t,j}(0) = \widetilde{\mathbf{x}}_t^j, \qquad \nu_{t,j}(1) = \mathbf{x}_{t-1}, \qquad \nu_{t,j}'(\rho) = \left(x_{t-1}^j - x_t^j\right)\mathbf{e}_j.$$

We first prove $\left\|q_t^j\right\| \le B_y$.
By the fundamental theorem of calculus,

$$\bar{\mathbf{y}}(\mathbf{x}_{t-1}) - \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j) = \int_0^1 \frac{d}{d\rho}\bar{\mathbf{y}}(\nu_{t,j}(\rho))d\rho.$$

Using the chain rule,

$$\frac{d}{d\rho}\bar{\mathbf{y}}(\nu_{t,j}(\rho)) = \nabla_{\mathbf{x}}\bar{\mathbf{y}}(\nu_{t,j}(\rho))\nu_{t,j}'(\rho) = \nabla_{\mathbf{x}}\bar{\mathbf{y}}(\nu_{t,j}(\rho))\left(x_{t-1}^j - x_t^j\right)\mathbf{e}_j.$$

Therefore,

$$\bar{\mathbf{y}}(\mathbf{x}_{t-1}) - \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j) = \int_0^1 \nabla_{\mathbf{x}}\bar{\mathbf{y}}(\nu_{t,j}(\rho))\left(x_{t-1}^j - x_t^j\right)\mathbf{e}_j d\rho.$$

Divide both sides by $x_{t-1}^j - x_t^j$, we obtain

$$q_t^j = \int_0^1 \nabla_{\mathbf{x}}\bar{\mathbf{y}}(\nu_{t,j}(\rho))\mathbf{e}_j d\rho.$$

Now take the norm:

$$\begin{aligned}
\left\|q_t^j\right\| &= \left\|\int_0^1 \nabla_{\mathbf{x}}\bar{\mathbf{y}}(\nu_{t,j}(\rho))\mathbf{e}_j d\rho\right\| \\
&\overset{(a)}{\le} \int_0^1 \left\|\nabla_{\mathbf{x}}\bar{\mathbf{y}}(\nu_{t,j}(\rho))\mathbf{e}_j\right\| d\rho \\
&\le \int_0^1 \left\|\nabla_{\mathbf{x}}\bar{\mathbf{y}}(\nu_{t,j}(\rho))\right\| \left\|\mathbf{e}_j\right\| d\rho \\
&\overset{(b)}{\le} \int_0^1 B_y d\rho \\
&= B_y,
\end{aligned}$$

where (a) utilizes the triangle inequality for integrals. (b) is because of Assumption 4.3 and $\|\mathbf{e}_j\| = 1$.
Then, we prove $\mathbb{E}\left[\left\|\widehat{q}_t^j - q_t^j\right\|^2\right] \le \sigma_y^2$.
By the fundamental theorem of calculus,

$$\bar{\mathbf{y}}(\mathbf{x}_{t-1};\mathcal{B}^y) - \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j;\mathcal{B}^y) = \int_0^1 \nabla_{\mathbf{x}}\bar{\mathbf{y}}(\nu_{t,j}(\rho);\mathcal{B}^y)\left(x_{t-1}^j - x_t^j\right)\mathbf{e}_j d\rho.$$

Dividing by $x_{t-1}^j - x_t^j$, we have

$$\widehat{q}_t^j = \int_0^1 \nabla_{\mathbf{x}}\bar{\mathbf{y}}(\nu_{t,j}(\rho);\mathcal{B}^y)\mathbf{e}_j d\rho.$$

Hence,

$$\widehat{q}_t^j - q_t^j = \int_0^1 \left(\nabla_{\mathbf{x}} \bar{\mathbf{y}}(\nu_{t,j}(\rho); \mathcal{B}^y) - \nabla_{\mathbf{x}} \bar{\mathbf{y}}(\nu_{t,j}(\rho))\right) \mathbf{e}_j d\rho.$$

Using Jensen's inequality, we get

$$\begin{aligned}
&\mathbb{E}\left[\left\|\widehat{q}_t^j - q_t^j\right\|^2\right] \leq \int_0^1 \mathbb{E}\left[\left\|\left(\nabla_{\mathbf{x}} \bar{\mathbf{y}}(\nu_{t,j}(\rho); \mathcal{B}^y) - \nabla_{\mathbf{x}} \bar{\mathbf{y}}(\nu_{t,j}(\rho))\right) \mathbf{e}_j\right\|^2\right] d\rho \\
&\overset{(a)}{\leq} \int_0^1 \sigma_y^2 d\rho \\
&= \sigma_y^2,
\end{aligned}$$

where (a) uses Assumption 4.4.

These complete the proof. □

D Theoretical Analysis for LeGo-BiO in Deterministic Setting

Define the coordinate-wise estimator vector

$$\mathbf{d}_t^f \triangleq \sum_{j=1}^{d_u} \left[h_t^{f,j}\right]_j \mathbf{e}_j \in \mathbb{R}^{d_u}.$$

The update of the UL parameter can be written as

$$\mathbf{x}_{t+1} = \mathbf{x}_t - \alpha_t \mathbf{d}_t^f. \tag{9}$$

D.1 Descent in the Upper-Level Objective Function

Lemma D.1. *Under Assumptions 4.1 and 4.2 and choosing $\alpha_t \leq \frac{1}{2L_l}$, the following inequality holds for successive iterations of Algorithm 1:*

$$\ell(\mathbf{x}_{t+1}) - \ell(\mathbf{x}_t) \leq -\frac{\alpha_t}{2}\left\|\nabla \ell(\mathbf{x}_t)\right\|^2 - \frac{\alpha_t}{4}\left\|\mathbf{d}_t^f\right\|^2 + \frac{\alpha_t}{2}\left\|\nabla \ell(\mathbf{x}_t) - \mathbf{d}_t^f\right\|^2,$$

for all $t \in \{0, 1, \ldots, T-1\}$.

Proof. We have

$$\begin{aligned}
&\ell(\mathbf{x}_{t+1}) - \ell(\mathbf{x}_t) \\
&\overset{(a)}{\leq} \left\langle \nabla \ell(\mathbf{x}_t), \mathbf{x}_{t+1} - \mathbf{x}_t \right\rangle + \frac{L_l}{2}\left\|\mathbf{x}_{t+1} - \mathbf{x}_t\right\|^2 \\
&\overset{(b)}{=} -\alpha_t \left\langle \nabla \ell(\mathbf{x}_t), \mathbf{d}_t^f \right\rangle + \frac{\alpha_t^2 L_l}{2}\left\|\mathbf{d}_t^f\right\|^2 \\
&\overset{(c)}{=} -\frac{\alpha_t}{2}\left\|\nabla \ell(\mathbf{x}_t)\right\|^2 - \frac{\alpha_t}{2}\left\|\mathbf{d}_t^f\right\|^2 + \frac{\alpha_t}{2}\left\|\nabla \ell(\mathbf{x}_t) - \mathbf{d}_t^f\right\|^2 + \frac{\alpha_t^2 L_l}{2}\left\|\mathbf{d}_t^f\right\|^2 \\
&\overset{(d)}{\leq} -\frac{\alpha_t}{2}\left\|\nabla \ell(\mathbf{x}_t)\right\|^2 - \frac{\alpha_t}{4}\left\|\mathbf{d}_t^f\right\|^2 + \frac{\alpha_t}{2}\left\|\nabla \ell(\mathbf{x}_t) - \mathbf{d}_t^f\right\|^2,
\end{aligned}$$

where (a) uses the Lipschitz continuous gradients of ℓ (see Lemma C.1). (b) follows from the update rule in Eq. (9). (c) is because of $\langle x, y\rangle = \frac{1}{2}\|x\|^2 + \frac{1}{2}\|y\|^2 - \frac{1}{2}\|x - y\|^2$. (d) chooses $\alpha_t \leq \frac{1}{2L_l}$.

The proof of the lemma is complete. □

D.2 Descent in the Gradient Approximation Error

Lemma D.2. *Under Assumptions 4.1–4.3, the gradient approximation error in Algorithm 1 satisfies the following inequality:*

$$\left\|\nabla \ell(\mathbf{x}_t) - \mathbf{d}_t^f\right\|^2 \leq 4 d_u L_l^2 \alpha_{t-1}^2 \left\|\mathbf{d}_{t-1}^f\right\|^2 + B_{f_y}^2 d_l \frac{L_y^2}{2} \alpha_{t-1}^2 \left\|\mathbf{d}_{t-1}^f\right\|^2 + 4 d_u L_f^2 \delta_t^2,$$

for all $t \in \{0, 1, \ldots, T-1\}$.

Proof. We have

$$\begin{aligned}
&\left\|\nabla \ell\left(\mathbf{x}_t\right)-\mathbf{d}_t^f\right\|^2 \\
&=\sum_{j=1}^{d_u}\left(\left[\nabla \ell\left(\mathbf{x}_t\right)\right]_j-\left[h_t^{f, j}\right]_j\right)^2 \\
&\leq 2 \sum_{j=1}^{d_u}\left(\left[\nabla \ell\left(\mathbf{x}_t\right)\right]_j-\left[\nabla f\left(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j)\right)\right]_j\right)^2+2 \sum_{j=1}^{d_u}\left(\left[\nabla f\left(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j)\right)\right]_j-\left[h_t^{f, j}\right]_j\right)^2 . \quad (10)
\end{aligned}$$

First, we bound the first term on the right-hand side of Eq. (10).

$$\begin{aligned}
&\left(\left[\nabla \ell\left(\mathbf{x}_t\right)\right]_j-\left[\nabla f\left(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j)\right)\right]_j\right)^2 \\
&\leq\left\|\nabla \ell\left(\mathbf{x}_t\right)-\nabla f\left(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j)\right)\right\|^2 \\
&\leq 2\left\|\nabla \ell\left(\mathbf{x}_t\right)-\nabla \ell\left(\widetilde{\mathbf{x}}_t^j\right)\right\|^2+2\left\|\nabla \ell\left(\widetilde{\mathbf{x}}_t^j\right)-\nabla f\left(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j)\right)\right\|^2 \\
&\overset{(a)}{\leq} 2 L_l^2\left\|\mathbf{x}_t-\widetilde{\mathbf{x}}_t^j\right\|^2+2 L_f^2\left\|\mathbf{y}^*(\widetilde{\mathbf{x}}_t^j)-\bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j)\right\|^2 \\
&\overset{(b)}{\leq} 2 L_l^2\left\|\mathbf{x}_t-\widetilde{\mathbf{x}}_t^j\right\|^2+2 L_f^2 \delta_t^2 \\
&\leq 2 L_l^2 \alpha_{t-1}^2\left\|\mathbf{d}_{t-1}^f\right\|^2+2 L_f^2 \delta_t^2, \quad (11)
\end{aligned}$$

where (a) utilizes the Lipschitzness of $\nabla \ell\left(\mathbf{x}\right)$ and $\nabla f\left(\mathbf{x}, \mathbf{y}\right)$ (see Lemma C.1 and Assumption 4.1). (b) is due to Assumption 4.3.

Next, we bound the second term on the right-hand side of Eq. (10).

$$\begin{aligned}
&\left(\left[\nabla f\left(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j)\right)\right]_j-\left[h_t^{f, j}\right]_j\right)^2 \\
&\overset{(a)}{=}\left(\left[\nabla_{\mathbf{x}} f\left(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j)\right)\right]_j+\left[\left[\nabla_{\mathbf{x}} \bar{\mathbf{y}}\left(\widetilde{\mathbf{x}}_t^j\right)\right]^{\top} \nabla_{\mathbf{y}} f\left(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}\left(\widetilde{\mathbf{x}}_t^j\right)\right)\right]_j\right. \\
&\quad\left.-\left(\left[\nabla_{\mathbf{x}} f\left(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j)\right)\right]_j+\left[\left[\widehat{\nabla}_{\mathbf{x}} \bar{\mathbf{y}}\left(\widetilde{\mathbf{x}}_t^j\right)\right]^{\top} \nabla_{\mathbf{y}} f\left(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}\left(\widetilde{\mathbf{x}}_t^j\right)\right)\right]_j\right)\right)^2 \\
&\leq\left\|\nabla_{\mathbf{x}} \bar{\mathbf{y}}\left(\widetilde{\mathbf{x}}_t^j\right)-\widehat{\nabla}_{\mathbf{x}} \bar{\mathbf{y}}\left(\widetilde{\mathbf{x}}_t^j\right)\right\|^2\left\|\nabla_{\mathbf{y}} f\left(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}\left(\widetilde{\mathbf{x}}_t^j\right)\right)\right\|^2 \\
&\overset{(b)}{\leq} B_{f_y}^2\left\|\nabla_{\mathbf{x}} \bar{\mathbf{y}}\left(\widetilde{\mathbf{x}}_t^j\right)-\widehat{\nabla}_{\mathbf{x}} \bar{\mathbf{y}}\left(\widetilde{\mathbf{x}}_t^j\right)\right\|^2 \\
&=B_{f_y}^2 \sum_{d=1}^{d_l}\left\|\nabla_{\mathbf{x}} \bar{\mathbf{y}}_d\left(\widetilde{\mathbf{x}}_t^j\right)-\widehat{\nabla}_{\mathbf{x}} \bar{\mathbf{y}}_d\left(\widetilde{\mathbf{x}}_t^j\right)\right\|^2 \\
&\overset{(c)}{\leq} B_{f_y}^2 d_l \frac{L_y^2}{4}\left\|\mathbf{x}_{t-1}-\widetilde{\mathbf{x}}_t^j\right\|^2 \\
&=B_{f_y}^2 d_l \frac{L_y^2}{4}\left(x_{t-1}^j-x_t^j\right)^2, \quad (12)
\end{aligned}$$

where (a) uses the definitions of $h_t^{f, j}$ and $\nabla f\left(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}\left(\widetilde{\mathbf{x}}_t^j\right)\right)$. (b) is due to Assumption 4.1. (c) utilizes Lemma C.2.

Substituting Eq. (11) and Eq. (12) into Eq. (10), we have

$$\left\|\nabla \ell\left(\mathbf{x}_t\right)-\mathbf{d}_t^f\right\|^2$$

$$\leq 4d_u L_l^2 \alpha_{t-1}^2 \left\|\mathbf{d}_{t-1}^f\right\|^2 + 4d_u L_f^2 \delta_t^2 + B_{f_y}^2 d_l \frac{L_y^2}{2} \left\|\mathbf{x}_{t-1} - \mathbf{x}_t\right\|^2$$
$$\leq 4d_u L_l^2 \alpha_{t-1}^2 \left\|\mathbf{d}_{t-1}^f\right\|^2 + B_{f_y}^2 d_l \frac{L_y^2}{2} \alpha_{t-1}^2 \left\|\mathbf{d}_{t-1}^f\right\|^2 + 4d_u L_f^2 \delta_t^2.$$

Thus, the proof is complete. □

D.3 Descent in the Potential Function

We define the potential function W_t as follows:

$$W_t = \ell\left(\mathbf{x}_t\right) + K_l \left\|\nabla \ell\left(\mathbf{x}_t\right) - \mathbf{d}_t^f\right\|^2.$$

Lemma D.3. *Under Assumptions 4.1–4.3, choose α_t and K_l such that*

$$\alpha_t \leq \min\left\{\frac{1}{2L_l}, \frac{1}{32d_u L_l^2 K_l}, \frac{1}{4B_{f_y}^2 L_y^2 d_l K_l}\right\}, \quad K_l \geq \frac{\alpha_t}{2}.$$

Then, the iterates generated by Algorithm 1 satisfies: for all $t \in \{0, 1, \ldots, T-1\}$,

$$W_{t+1} - W_t \leq -\frac{\alpha_t}{2} \left\|\nabla \ell\left(\mathbf{x}_t\right)\right\|^2 + 4d_u L_f^2 \delta_{t+1}^2 K_l.$$

Proof. From Lemma D.1, we have

$$\ell\left(\mathbf{x}_{t+1}\right) - \ell\left(\mathbf{x}_t\right) \leq -\frac{\alpha_t}{2} \left\|\nabla \ell\left(\mathbf{x}_t\right)\right\|^2 - \frac{\alpha_t}{4} \left\|\mathbf{d}_t^f\right\|^2 + \frac{\alpha_t}{2} \left\|\nabla \ell\left(\mathbf{x}_t\right) - \mathbf{d}_t^f\right\|^2. \tag{13}$$

Based on Lemma D.2, we have

$$\begin{aligned}
&\left\|\nabla \ell\left(\mathbf{x}_{t+1}\right) - \mathbf{d}_{t+1}^f\right\|^2 - \left\|\nabla \ell\left(\mathbf{x}_t\right) - \mathbf{d}_t^f\right\|^2 \\
&\leq -\left\|\nabla \ell\left(\mathbf{x}_t\right) - \mathbf{d}_t^f\right\|^2 + 4d_u L_l^2 \alpha_t^2 \left\|\mathbf{d}_t^f\right\|^2 + B_{f_y}^2 d_l \frac{L_y^2}{2} \alpha_t^2 \left\|\mathbf{d}_t^f\right\|^2 + 4d_u L_f^2 \delta_{t+1}^2.
\end{aligned} \tag{14}$$

Combining Eq. (13) and (14), we get

$$\begin{aligned}
&W_{t+1} - W_t \\
&\leq -\frac{\alpha_t}{2} \left\|\nabla \ell\left(\mathbf{x}_t\right)\right\|^2 + C_h \left\|\mathbf{d}_t^f\right\|^2 + C_l \left\|\nabla \ell\left(\mathbf{x}_t\right) - \mathbf{d}_t^f\right\|^2 + 4d_u L_f^2 \delta_{t+1}^2 K_l,
\end{aligned}$$

where

$$\begin{aligned}
C_h &= -\frac{\alpha_t}{4} + 4d_u L_l^2 \alpha_t^2 K_l + B_{f_y}^2 d_l \frac{L_y^2}{2} \alpha_t^2 K_l, \\
C_l &= \frac{\alpha_t}{2} - K_l.
\end{aligned}$$

To ensure $C_h \leq 0$, we have

$$\begin{aligned}
C_h &= -\frac{\alpha_t}{4} + 4d_u L_l^2 \alpha_t^2 K_l + B_{f_y}^2 d_l \frac{L_y^2}{2} \alpha_t^2 K_l \\
&\overset{(a)}{\leq} -\frac{\alpha_t}{4} + \frac{\alpha_t}{8} + \frac{\alpha_t}{8} = 0,
\end{aligned}$$

where (a) uses $\alpha_t \leq \min\left\{\frac{1}{32d_u L_l^2 K_l}, \frac{1}{4B_{f_y}^2 L_y^2 d_l K_l}\right\}$.

To ensure $C_l \leq 0$, we choose $K_l \geq \frac{\alpha_t}{2}$.

Towards this end, the lemma is proved. □

D.4 Proof of Theorem 4.6

Theorem D.4 (Deterministic Setting)**.** *Under Assumptions 4.1–4.3, choose constant step-sizes* $0 < \alpha_t = \alpha \leq \bar{\alpha}$ *for all* $t \in \{0, 1, \ldots, T-1\}$*. Moreover, choose* $\bar{\alpha}$ *such that*

$$\bar{\alpha} \leq \min\left\{\frac{1}{2L_l}, \sqrt{\frac{2}{32d_u L_l^2}}, \sqrt{\frac{2}{4B_{f_y}^2 L_y^2 d_l}}\right\}.$$

Then, the iterates generated by LeGo-BiO *satisfy:*

$$\frac{1}{T}\sum_{t=0}^{T-1}\|\nabla \ell(\mathbf{x}_t)\|^2 = \mathcal{O}\left(\frac{\Delta_0}{T\alpha}\right) + \mathcal{O}\left(d_u \frac{1}{T}\sum_{t=0}^{T-1}\delta_{t+1}^2\right),$$

where $\Delta_0 = (\ell(\mathbf{x}_0) - \ell^*) + \|\nabla \ell(\mathbf{x}_0) - \sum_{j=1}^{d_u}[h_0^{f,j}]_j \mathbf{e}_j\|^2$.

Proof. Let $K_l = \frac{\alpha_t}{2}$, and choose α_t as a constant stepsize $\alpha_t = \alpha$. Summing the result in Lemma D.3 from $t = 0$ to $T-1$, and then dividing by T on both sides, we get

$$\frac{W_T - W_0}{T} \leq -\frac{\alpha}{2T}\sum_{t=0}^{T-1}\|\nabla \ell(\mathbf{x}_t)\|^2 + 2d_u L_f^2 \alpha \frac{1}{T}\sum_{t=0}^{T-1}\delta_{t+1}^2.$$

Rearranging the terms and multiplying by $2/\alpha$ on both sides, we have

$$\frac{1}{T}\sum_{t=0}^{T-1}\|\nabla \ell(\mathbf{x}_t)\|^2 \leq \frac{2(W_0 - \ell^*)}{\alpha T} + 4d_u L_f^2 \frac{1}{T}\sum_{t=0}^{T-1}\delta_{t+1}^2,$$

where $W_0 = \ell(\mathbf{x}_0) + K_l \left\|\nabla \ell(\mathbf{x}_0) - \mathbf{d}_0^f\right\|^2$.

Therefore,

$$\begin{aligned}
&\frac{1}{T}\sum_{t=0}^{T-1}\|\nabla \ell(\mathbf{x}_t)\|^2 \\
&= \mathcal{O}\left(\frac{(\ell(\mathbf{x}_0) - \ell^*) + \left\|\nabla \ell(\mathbf{x}_0) - \mathbf{d}_0^f\right\|^2}{\alpha T}\right) + \mathcal{O}\left(\frac{d_u}{T}\sum_{t=0}^{T-1}\delta_{t+1}^2\right).
\end{aligned}$$

The proof of the theorem is completed. □

E Theoretical Analysis for LeGo-BiO in Stochastic Setting

Define the coordinate-wise estimator vector

$$\widehat{\mathbf{d}}_t^f \triangleq \sum_{j=1}^{d_u}\left[\widehat{h}_t^{f,j}\right]_j \mathbf{e}_j \in \mathbb{R}^{d_u}.$$

The update of the UL parameter can be written as

$$\mathbf{x}_{t+1} = \mathbf{x}_t - \alpha_t \widehat{\mathbf{d}}_t^f. \tag{15}$$

E.1 Descent in the Upper-Level Objective Function

Lemma E.1. *Under Assumptions 4.1–4.4 and choosing* $\alpha_t \leq \frac{1}{4L_l}$*, the following inequality holds for successive iterations of Algorithm 1:*

$$\begin{aligned}
&\mathbb{E}\left[\ell(\mathbf{x}_{t+1}) - \ell(\mathbf{x}_t)\right] \\
&\leq -\frac{\alpha_t}{2}\mathbb{E}\left[\|\nabla \ell(\mathbf{x}_t)\|^2\right] - \frac{\alpha_t}{4}\mathbb{E}\left[\left\|\mathbf{d}_t^f\right\|^2\right] + \frac{\alpha_t}{2}\mathbb{E}\left[\left\|\nabla \ell(\mathbf{x}_t) - \mathbf{d}_t^f\right\|^2\right] + \alpha_t^2 L_l d_u \sigma_h^2,
\end{aligned}$$

for all $t \in \{0, 1, \ldots, T-1\}$*, where the expectation is taken over the stochasticity of the algorithm.*

Proof. We have

$$
\begin{aligned}
&\mathbb{E}\left[\ell\left(\mathbf{x}_{t+1}\right)-\ell\left(\mathbf{x}_t\right)\right] \\
&\overset{(a)}{\leq} \mathbb{E}\left[\left\langle\nabla \ell\left(\mathbf{x}_t\right), \mathbf{x}_{t+1}-\mathbf{x}_t\right\rangle+\frac{L_l}{2}\left\|\mathbf{x}_{t+1}-\mathbf{x}_t\right\|^2\right] \\
&\overset{(b)}{=} \mathbb{E}\left[-\alpha_t\left\langle\nabla \ell\left(\mathbf{x}_t\right), \widehat{\mathbf{d}}_t^f\right\rangle+\frac{\alpha_t^2 L_l}{2}\left\|\widehat{\mathbf{d}}_t^f\right\|^2\right] \\
&\overset{(c)}{=} \mathbb{E}\left[-\alpha_t\left\langle\nabla \ell\left(\mathbf{x}_t\right), \mathbf{d}_t^f\right\rangle+\frac{\alpha_t^2 L_l}{2}\left\|\widehat{\mathbf{d}}_t^f\right\|^2\right] \\
&\overset{(d)}{=} \mathbb{E}\left[-\frac{\alpha_t}{2}\left\|\nabla \ell\left(\mathbf{x}_t\right)\right\|^2-\frac{\alpha_t}{2}\left\|\mathbf{d}_t^f\right\|^2+\frac{\alpha_t}{2}\left\|\nabla \ell\left(\mathbf{x}_t\right)-\mathbf{d}_t^f\right\|^2+\frac{\alpha_t^2 L_l}{2}\left\|\widehat{\mathbf{d}}_t^f\right\|^2\right] \\
&\leq \mathbb{E}\left[-\frac{\alpha_t}{2}\left\|\nabla \ell\left(\mathbf{x}_t\right)\right\|^2-\frac{\alpha_t}{2}\left\|\mathbf{d}_t^f\right\|^2+\frac{\alpha_t}{2}\left\|\nabla \ell\left(\mathbf{x}_t\right)-\mathbf{d}_t^f\right\|^2+\alpha_t^2 L_l\left\|\widehat{\mathbf{d}}_t^f-\mathbf{d}_t^f\right\|^2+\alpha_t^2 L_l\left\|\mathbf{d}_t^f\right\|^2\right] \\
&\overset{(e)}{\leq} \mathbb{E}\left[-\frac{\alpha_t}{2}\left\|\nabla \ell\left(\mathbf{x}_t\right)\right\|^2-\frac{\alpha_t}{4}\left\|\mathbf{d}_t^f\right\|^2+\frac{\alpha_t}{2}\left\|\nabla \ell\left(\mathbf{x}_t\right)-\mathbf{d}_t^f\right\|^2+\alpha_t^2 L_l\left\|\widehat{\mathbf{d}}_t^f-\mathbf{d}_t^f\right\|^2\right], \qquad (16)
\end{aligned}
$$

where (a) uses the Lipschitz continuous gradients of ℓ (see Lemma C.1). (b) follows from the update rule in Eq. (15). (c) utilizes $\mathbb{E}[\widehat{\mathbf{d}}_t^f \mid \mathcal{F}_t] = \mathbf{d}_t^f$. (d) is because of $\langle x, y\rangle = \frac{1}{2}\|x\|^2 + \frac{1}{2}\|y\|^2 - \frac{1}{2}\|x-y\|^2$. (e) chooses $\alpha_t \leq \frac{1}{4L_l}$.

Next, we bound the last term on the right-hand side of Eq. (16).

Define

$$
q_t^j = \frac{\bar{\mathbf{y}}(\mathbf{x}_{t-1}) - \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j)}{x_{t-1}^j - x_t^j}, \qquad \widehat{q}_t^j = \frac{\bar{\mathbf{y}}(\mathbf{x}_{t-1}; \mathcal{B}_{t-1}^y) - \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j; \mathcal{B}_t^{y,j})}{x_{t-1}^j - x_t^j}.
$$

For each $j \in [d_u]$, we have

$$
\begin{aligned}
&\mathbb{E}\left[\left(\left[\widehat{\mathbf{d}}_t^f-\mathbf{d}_t^f\right]_j\right)^2\right] \\
&=\mathbb{E}\left[\left(\left[\nabla_{\mathbf{x}} f\left(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j); \mathcal{B}_t^{f_x}\right)-\nabla_{\mathbf{x}} f\left(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j)\right)\right]_j\right. \\
&\left.+\left[\left\langle\widehat{q}_t^j, \nabla_{\mathbf{y}} f(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j); \mathcal{B}_t^{f_y,j})\right\rangle-\left\langle q_t^j, \nabla_{\mathbf{y}} f(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j))\right\rangle\right]_j\right)^2\Bigg] \\
&=\mathbb{E}\left[\left(\left[\nabla_{\mathbf{x}} f\left(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j); \mathcal{B}_t^{f_x}\right)-\nabla_{\mathbf{x}} f\left(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j)\right)\right]_j\right. \\
&+\left[\left\langle q_t^j, \nabla_{\mathbf{y}} f(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j); \mathcal{B}_t^{f_y,j})-\nabla_{\mathbf{y}} f(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j))\right\rangle+\left\langle\widehat{q}_t^j-q_t^j, \nabla_{\mathbf{y}} f(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j))\right\rangle\right. \\
&\left.\left.+\left\langle\widehat{q}_t^j-q_t^j, \nabla_{\mathbf{y}} f(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j); \mathcal{B}_t^{f_y,j})-\nabla_{\mathbf{y}} f(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j))\right\rangle\right]_j\right)^2\Bigg] \\
&\leq 4\mathbb{E}\left[\left[\nabla_{\mathbf{x}} f\left(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j); \mathcal{B}_t^{f_x}\right)-\nabla_{\mathbf{x}} f\left(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j)\right)\right]_j^2\right]+4\mathbb{E}\left[\left|\left\langle\widehat{q}_t^j-q_t^j, \nabla_{\mathbf{y}} f(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j))\right\rangle\right|^2\right] \\
&+4\mathbb{E}\left[\left|\left\langle q_t^j, \nabla_{\mathbf{y}} f(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j); \mathcal{B}_t^{f_y,j})-\nabla_{\mathbf{y}} f(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j))\right\rangle\right|^2\right] \\
&+4\mathbb{E}\left[\left|\left\langle\widehat{q}_t^j-q_t^j, \nabla_{\mathbf{y}} f(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j); \mathcal{B}_t^{f_y,j})-\nabla_{\mathbf{y}} f(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j))\right\rangle\right|^2\right]. \qquad (17)
\end{aligned}
$$

Then, we bound each term on the right-hand side of Eq. (17).

$$
\mathbb{E}\left[\left[\nabla_{\mathbf{x}} f\left(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j); \mathcal{B}_t^{f_x}\right)-\nabla_{\mathbf{x}} f\left(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j)\right)\right]_j^2\right] \overset{(a)}{\leq} \sigma_{f_x}^2, \qquad (18)
$$

where (a) is based on Assumption 4.4.

$$\begin{aligned}
&\mathbb{E}\left[\left|\left\langle \widehat{q}_t^j - q_t^j, \nabla_{\mathbf{y}} f(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j))\right\rangle\right|^2\right] \\
&\overset{(a)}{\leq} \mathbb{E}\left[\left\|\widehat{q}_t^j - q_t^j\right\|^2 \left\|\nabla_{\mathbf{y}} f(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j))\right\|^2\right] \\
&\overset{(b)}{\leq} B_{f_y}^2 \sigma_y^2,
\end{aligned} \tag{19}$$

where (a) is due to Cauchy–Schwarz inequality. (b) uses Assumption 4.1 and Lemma C.3 with $\mathcal{B}_{t-1}^y = \mathcal{B}_t^{y,j} = \mathcal{B}^y$.

$$\begin{aligned}
&\mathbb{E}\left[\left|\left\langle q_t^j, \nabla_{\mathbf{y}} f(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j); \mathcal{B}_t^{f_y,j}) - \nabla_{\mathbf{y}} f(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j))\right\rangle\right|^2\right] \\
&\overset{(a)}{\leq} \mathbb{E}\left[\left\|q_t^j\right\|^2 \left\|\nabla_{\mathbf{y}} f(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j); \mathcal{B}_t^{f_y,j}) - \nabla_{\mathbf{y}} f(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j))\right\|^2\right] \overset{(b)}{\leq} B_y^2 \sigma_{f_y}^2,
\end{aligned} \tag{20}$$

where (a) uses Cauchy–Schwarz inequality. (b) is due to Assumption 4.4 and Lemma C.3.

$$\begin{aligned}
&\mathbb{E}\left[\left|\left\langle \widehat{q}_t^j - q_t^j, \nabla_{\mathbf{y}} f(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j); \mathcal{B}_t^{f_y,j}) - \nabla_{\mathbf{y}} f(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j))\right\rangle\right|^2\right] \\
&\overset{(a)}{\leq} \mathbb{E}\left[\left\|\widehat{q}_t^j - q_t^j\right\|^2 \left\|\nabla_{\mathbf{y}} f(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j); \mathcal{B}_t^{f_y,j}) - \nabla_{\mathbf{y}} f(\widetilde{\mathbf{x}}_t^j, \bar{\mathbf{y}}(\widetilde{\mathbf{x}}_t^j))\right\|^2\right] \\
&\overset{(b)}{\leq} \sigma_y^2 \sigma_{f_y}^2,
\end{aligned} \tag{21}$$

where (a) is because of Cauchy–Schwarz inequality. (b) uses Assumption 4.4 and Lemma C.3 with $\mathcal{B}_{t-1}^y = \mathcal{B}_t^{y,j} = \mathcal{B}^y$.

Substituting Eq. (18)–(21) into Eq. (17), we obtain

$$\mathbb{E}\left[\left(\left[\widehat{\mathbf{d}}_t^f - \mathbf{d}_t^f\right]_j\right)^2\right] \leq 4\sigma_{f_x}^2 + 4B_{f_y}^2\sigma_y^2 + 4B_y^2\sigma_{f_y}^2 + 4\sigma_y^2\sigma_{f_y}^2 = \sigma_h^2.$$

Summing over $j = 1, 2, \ldots, d_u$, we get

$$\mathbb{E}\left[\left\|\widehat{\mathbf{d}}_t^f - \mathbf{d}_t^f\right\|^2\right] \leq d_u\sigma_h^2. \tag{22}$$

Substituting Eq. (22) into Eq. (16) completes the proof. □

E.2 Descent in the Gradient Approximation Error

Lemma E.2. *Under Assumptions 4.1–4.4, the gradient approximation error in Algorithm 1 satisfies the following inequality:*

$$\begin{aligned}
\mathbb{E}\left[\left\|\nabla\ell\left(\mathbf{x}_t\right) - \mathbf{d}_t^f\right\|^2\right] \leq\ & 8d_u L_l^2 \alpha_{t-1}^2 \mathbb{E}\left[\left\|\mathbf{d}_{t-1}^f\right\|^2\right] + B_{f_y}^2 d_l L_y^2 \alpha_{t-1}^2 \mathbb{E}\left[\left\|\mathbf{d}_{t-1}^f\right\|^2\right] \\
&+ 8d_u L_l^2 \alpha_{t-1}^2 d_u \sigma_h^2 + B_{f_y}^2 d_l L_y^2 \alpha_{t-1}^2 d_u \sigma_h^2 + 4d_u L_f^2 \delta_t^2,
\end{aligned}$$

for all $t \in \{0, 1, \ldots, T-1\}$, where the expectation is taken over the stochasticity of the algorithm.

Proof. Following the proof of Lemma D.2 and using the update rule $\mathbf{x}_{t+1} = \mathbf{x}_t - \alpha_t \widehat{\mathbf{d}}_t^f$, we have

$$\begin{aligned}
&\mathbb{E}\left[\left\|\nabla\ell\left(\mathbf{x}_t\right) - \mathbf{d}_t^f\right\|^2\right] \\
&\leq 4d_u L_l^2 \alpha_{t-1}^2 \mathbb{E}\left[\left\|\widehat{\mathbf{d}}_{t-1}^f\right\|^2\right] + B_{f_y}^2 d_l \frac{L_y^2}{2} \alpha_{t-1}^2 \mathbb{E}\left[\left\|\widehat{\mathbf{d}}_{t-1}^f\right\|^2\right] + 4d_u L_f^2 \delta_t^2
\end{aligned}$$

$$\leq 8d_u L_l^2 \alpha_{t-1}^2 \mathbb{E}\left[\left\|\widehat{\mathbf{d}}_{t-1}^f - \mathbf{d}_{t-1}^f\right\|^2\right] + 8d_u L_l^2 \alpha_{t-1}^2 \mathbb{E}\left[\left\|\mathbf{d}_{t-1}^f\right\|^2\right]$$
$$+ B_{f_y}^2 d_l L_y^2 \alpha_{t-1}^2 \mathbb{E}\left[\left\|\widehat{\mathbf{d}}_{t-1}^f - \mathbf{d}_{t-1}^f\right\|^2\right] + B_{f_y}^2 d_l L_y^2 \alpha_{t-1}^2 \mathbb{E}\left[\left\|\mathbf{d}_{t-1}^f\right\|^2\right] + 4d_u L_f^2 \delta_t^2. \quad (23)$$

Substituting Eq. (22) into Eq. (23) completes the proof. □

E.3 Descent in the Potential Function

We define the potential function $\widehat{W}_t$ as follows:

$$\widehat{W}_t = \ell(\mathbf{x}_t) + K_l \left\|\nabla \ell(\mathbf{x}_t) - \mathbf{d}_t^f\right\|^2.$$

Lemma E.3. *Under Assumptions 4.1–4.4, choose α_t and K_l such that*

$$\alpha_t \leq \min\left\{\frac{1}{4L_l}, \frac{1}{64 d_u L_l^2 K_l}, \frac{1}{8B_{f_y}^2 L_y^2 d_l K_l}\right\}, \quad K_l \geq \frac{\alpha_t}{2}.$$

Then, the iterates generated by Algorithm 1 satisfies: for all $t \in \{0, 1, \ldots, T-1\}$,

$$\mathbb{E}\left[\widehat{W}_{t+1} - \widehat{W}_t\right]$$
$$\leq -\frac{\alpha_t}{2}\mathbb{E}\left[\|\nabla \ell(\mathbf{x}_t)\|^2\right] + \alpha_t^2 L_l d_u \sigma_h^2 + 8d_u L_l^2 \alpha_t^2 d_u \sigma_h^2 K_l + B_{f_y}^2 d_l L_y^2 \alpha_t^2 d_u \sigma_h^2 K_l + 4d_u L_f^2 \delta_{t+1}^2 K_l,$$

where the expectation is taken over the stochasticity of the algorithm.

Proof. From Lemma E.1, we have

$$\mathbb{E}\left[\ell(\mathbf{x}_{t+1}) - \ell(\mathbf{x}_t)\right]$$
$$\leq -\frac{\alpha_t}{2}\mathbb{E}\left[\|\nabla \ell(\mathbf{x}_t)\|^2\right] - \frac{\alpha_t}{4}\mathbb{E}\left[\left\|\mathbf{d}_t^f\right\|^2\right] + \frac{\alpha_t}{2}\mathbb{E}\left[\left\|\nabla \ell(\mathbf{x}_t) - \mathbf{d}_t^f\right\|^2\right] + \alpha_t^2 L_l d_u \sigma_h^2. \quad (24)$$

Based on Lemma E.2, we have

$$\mathbb{E}\left[\left\|\nabla \ell(\mathbf{x}_{t+1}) - \mathbf{d}_{t+1}^f\right\|^2 - \left\|\nabla \ell(\mathbf{x}_t) - \mathbf{d}_t^f\right\|^2\right]$$
$$\leq -\mathbb{E}\left[\left\|\nabla \ell(\mathbf{x}_t) - \mathbf{d}_t^f\right\|^2\right] + 8d_u L_l^2 \alpha_t^2 \mathbb{E}\left[\left\|\mathbf{d}_t^f\right\|^2\right] + B_{f_y}^2 d_l L_y^2 \alpha_t^2 \mathbb{E}\left[\left\|\mathbf{d}_t^f\right\|^2\right]$$
$$+ 8d_u L_l^2 \alpha_t^2 d_u \sigma_h^2 + B_{f_y}^2 d_l L_y^2 \alpha_t^2 d_u \sigma_h^2 + 4d_u L_f^2 \delta_{t+1}^2. \quad (25)$$

Combining Eq. (24) and (25), we get

$$\mathbb{E}\left[\widehat{W}_{t+1} - \widehat{W}_t\right]$$
$$\leq -\frac{\alpha_t}{2}\mathbb{E}\left[\|\nabla \ell(\mathbf{x}_t)\|^2\right] + C_h \mathbb{E}\left[\left\|\mathbf{d}_t^f\right\|^2\right] + C_l \mathbb{E}\left[\left\|\nabla \ell(\mathbf{x}_t) - \mathbf{d}_t^f\right\|^2\right]$$
$$+ \alpha_t^2 L_l d_u \sigma_h^2 + 8d_u L_l^2 \alpha_t^2 d_u \sigma_h^2 K_l + B_{f_y}^2 d_l L_y^2 \alpha_t^2 d_u \sigma_h^2 K_l + 4d_u L_f^2 \delta_{t+1}^2 K_l,$$

where

$$C_h = -\frac{\alpha_t}{4} + 8d_u L_l^2 \alpha_t^2 K_l + B_{f_y}^2 d_l L_y^2 \alpha_t^2 K_l,$$
$$C_l = \frac{\alpha_t}{2} - K_l.$$

To ensure $C_h \leq 0$, we have

$$C_h = -\frac{\alpha_t}{4} + 8d_u L_l^2 \alpha_t^2 K_l + B_{f_y}^2 d_l L_y^2 \alpha_t^2 K_l$$

$$\overset{(a)}{\leq} -\frac{\alpha_t}{4} + \frac{\alpha_t}{8} + \frac{\alpha_t}{8} = 0,$$

where (a) uses $\alpha_t \leq \min\left\{\frac{1}{64d_u L_l^2 K_l}, \frac{1}{8B_{f_y}^2 L_y^2 d_l K_l}\right\}$.

To ensure $C_l \leq 0$, we choose $K_l \geq \frac{\alpha_t}{2}$.

Towards this end, the lemma is proved. □

E.4 Proof of Theorem 4.8

Theorem E.4 (Stochastic Setting)**.** *Under Assumptions 4.1–4.4, choose constant step-sizes* $0 < \alpha_t = \alpha \leq \bar{\alpha}$ *for all* $t \in \{0, 1, \ldots, T-1\}$. *Moreover, choose* $\bar{\alpha}$ *such that*

$$\bar{\alpha} \leq \min\left\{\frac{1}{4L_l}, \sqrt{\frac{2}{64d_u L_l^2}}, \sqrt{\frac{2}{8B_{f_y}^2 L_y^2 d_l}}\right\}.$$

Then, the iterates generated by LeGo-BiO *satisfy:*

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\left\|\nabla\ell\left(\mathbf{x}_t\right)\right\|^2\right] = \mathcal{O}\left(\frac{\Delta_0}{T\alpha}\right) + \mathcal{O}\left(d_u\frac{1}{T}\sum_{t=0}^{T-1}\delta_{t+1}^2 + \alpha d_u\left(\sigma_{f_x}^2 + \sigma_y^2 + \sigma_{f_y}^2 + \sigma_y^2\sigma_{f_y}^2\right)\right),$$

where $\Delta_0 = \left(\ell\left(\mathbf{x}_0\right) - \ell^*\right) + \left\|\nabla\ell\left(\mathbf{x}_0\right) - \sum_{j=1}^{d_u}[h_0^{f,j}]_j\mathbf{e}_j\right\|^2$.

Proof. Let $K_l = \frac{\alpha_t}{2}$, and choose α_t as a constant stepsize $\alpha_t = \alpha$. Summing the result in Lemma E.3 from $t = 0$ to $T-1$, and then dividing by T on both sides, we get

$$\begin{aligned}\frac{\widehat{W}_T - \widehat{W}_0}{T} \leq& -\frac{\alpha}{2T}\sum_{t=0}^{T-1}\mathbb{E}\left[\left\|\nabla\ell\left(\mathbf{x}_t\right)\right\|^2\right] \\ &+ \alpha^2 L_l d_u\sigma_h^2 + 4d_u L_l^2\alpha^3 d_u\sigma_h^2 + \frac{1}{2}B_{f_y}^2 d_l L_y^2\alpha^3 d_u\sigma_h^2 + 2d_u L_f^2\alpha\frac{1}{T}\sum_{t=0}^{T-1}\delta_{t+1}^2.\end{aligned}$$

Rearranging the terms and multiplying by $2/\alpha$ on both sides, we have

$$\begin{aligned}\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\left\|\nabla\ell\left(\mathbf{x}_t\right)\right\|^2\right] \leq& \frac{2\left(\widehat{W}_0 - \ell^*\right)}{\alpha T} \\ &+ 2\alpha L_l d_u\sigma_h^2 + 8d_u L_l^2\alpha^2 d_u\sigma_h^2 + B_{f_y}^2 d_l L_y^2\alpha^2 d_u\sigma_h^2 + 4d_u L_f^2\frac{1}{T}\sum_{t=0}^{T-1}\delta_{t+1}^2,\end{aligned}$$

where $\widehat{W}_0 = \ell\left(\mathbf{x}_0\right) + K_l\left\|\nabla\ell\left(\mathbf{x}_0\right) - \mathbf{d}_0^f\right\|^2$.

Therefore,

$$\begin{aligned}&\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\left\|\nabla\ell\left(\mathbf{x}_t\right)\right\|^2\right] \\ &= \mathcal{O}\left(\frac{\left(\ell\left(\mathbf{x}_0\right) - \ell^*\right) + \left\|\nabla\ell\left(\mathbf{x}_0\right) - \mathbf{d}_0^f\right\|^2}{\alpha T}\right) + \mathcal{O}\left(d_u\frac{1}{T}\sum_{t=0}^{T-1}\delta_{t+1}^2 + \alpha d_u\sigma_h^2\right).\end{aligned}$$

The proof of the theorem is completed. □

F Broader Impacts

The work presented in this paper aims to advance the field of machine learning by developing algorithms for lower-level agnostic bilevel optimization. Our work has many potential positive social impacts. For example, by enabling upper-level learners to operate when the lower-level response is produced by a black-box solver, simulator, adversary, or learned predictor, our LeGo-BiO framework supports a broad range of applications, including adversarial training and deep hyper-representation learning. Since this work is foundational and not tied to a specific application domain, we do not anticipate direct negative societal impacts.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The paper's main contributions, as stated in the abstract and introduction, are substantiated in Sections 3–5.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: See Section B.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: The full set of assumptions is provided in Section 4, and the complete proofs are given in Appendices D and E.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: See Section A.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: The datasets used in our experiments are publicly available open-source datasets. We will release the experimental code upon acceptance of the paper.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: See Section A.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: All reported results are averaged over the training and test datasets, ensuring that our findings reflect overall model performance rather than variability from individual samples or runs.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: See Section A.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We adhere to the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: See Section F.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: The datasets and models used in our experiments are properly credited, and their licenses are explicitly reported in Section A.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: This research does not use LLMs as any important, original, or non-standard components.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.