Sharpness-aware Zeroth-order Optimization for Graph Transformers

Yang Liu[1], **Chuan Zhou**[1,2*], **Yuhan Lin**[3], **Shuai Zhang**[1], **Yang Gao**[4], **Zhao Li**[5] and **Shirui Pan**[6]

[1]Academy of Mathematics and Systems Science, Chinese Academy of Science
[2]School of Cyber Security, University of Chinese Academy of Science
[3]Fudan University
[4]Zhejiang University
[5]Hangzhou Yugu Technology
[6]Griffith University
{liuyang2020, zhouchuan, zhangshuai2021}@amss.ac.cn, yhlin22@m.fudan.edu.cn, gaoyang9507@zju.edu.cn, lzjoey@gmail.com, s.pan@griffith.edu.au

Abstract

Graph Transformers (GTs) have emerged as powerful tools for handling graph-structured data through global attention mechanisms. While GTs can effectively capture long-range dependencies, they introduce difficulties in optimization due to their complex, non-differentiable operators, which cannot be directly handled by standard gradient-based optimizers (such as Adam or AdamW). To investigate the above issues, this work adopts the line of Zeroth-Order Optimization (ZOO) technique. However, direct integration of ZOO incurs considerable challenges due to the sharp loss landscape and steep gradients within the GT parameter space. Under the above observations, we propose a Sharpness-aware Zeroth-order Optimizer (SZO) that combines Sharpness-Aware Minimization (SAM) technique facilitating convergence within a flatter neighborhood, and leverages parallel computing for efficient gradient estimation. Theoretically, we provide a comprehensive analysis of the optimizer from both convergence and generalization perspectives. Empirically, we conduct extensive experiments on various classical GTs across a wide range of benchmark datasets, which underscore the superior performance of SZO over the state-of-the-art optimizers.

1 Introduction

Graph Transformers (GTs) represent a significant advancement in machine learning, introducing an emerging class of network architectures purposefully designed to handle graph-structured data with considerable efficiency and flexibility [Dwivedi and Bresson, 2020; Liu *et al.*, 2024a]. In contrast to conventional graph neural networks (GNNs), GTs have demonstrated a strong capacity to address key issues such as over-smoothing and over-squashing [Topping *et al.*, 2021]. The incorporation of transformer-based architectures into graph models has proven to be highly adaptable and powerful. Notably, this approach has yielded remarkable empirical results in domains as varied as drug discovery [Liu *et al.*, 2022a], where intricate molecular interactions must be accurately modeled, and molecular structure prediction [Rong *et al.*, 2020], which demands precise representation of large, interconnected systems. The adaptability, robustness, and improved representational power of GTs drive ongoing research and offer significant potential for advancing various machine learning applications [Liu *et al.*, 2024b; Chang *et al.*, 2025].

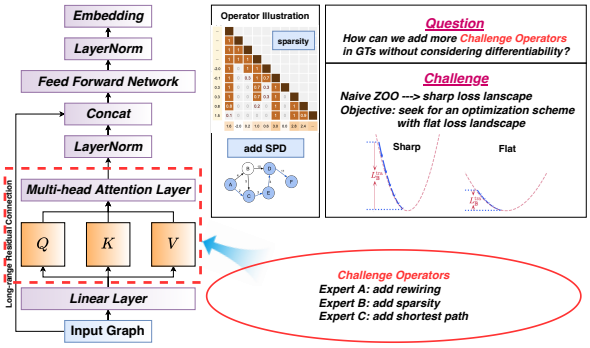

Figure 1: Illustration of our motivation and unveiling the power of SZO in graph transformers.

Motivation. Despite extensive efforts to develop advanced optimizers for Transformer-based architectures, a fundamental challenge arises from the presence of certain complex operators within GTs, such as shortest-path distance computations, which are inherently non-differentiable. These operators preclude the straightforward application of conventional gradient-based optimization techniques (e.g., ADAM) [Ying *et al.*, 2021]. In response to this challenge, Zeroth-Order Optimization (ZOO) has emerged as a promising alternative, providing a means of gradient estimation without direct differentiation. Such approaches have recently gained prominence in addressing optimization challenges in large-scale

*Corresponding author

Transformer-based language models [Zhang *et al.*, 2024]. However, the naive integration of ZOO methods in GTs faces substantial challenges in inferior and non-robustness, particularly as model complexity and dimensionality increase [Kunstner *et al.*, 2023]. These challenges arise from the steep gradients in the parameter space of GTs and the sharp loss landscape encountered during ZOO training [He *et al.*, 2022], where convergence to sharp local minima can significantly impair generalization [Foret *et al.*, 2020] and overall performance. Furthermore, we conduct a demonstration experiment which illustrates the poor performance of vanilla ZOO (see Figure 2). Consequently, novel strategies are in urgent need to leverage ZOO principles in GTs. To address the challenges associated with the loss landscape in ZOO, we consider incorporating Sharpness-Aware Minimization (SAM) [Foret *et al.*, 2020], which aims to find local minima with a flatter neighborhood and enhance the robustness of the optimizer. In technique, SAM updates the model parameters using two gradients (one from the neighborhood and one from the perturbed parameters), minimizing the average loss over the perturbed parameters, thereby enhancing generalization by reducing sensitivity to small variations in the parameter space.

In this work, we propose a Sharpness-aware Zeroth-order Optimizer (SZO) for graph transformers which basically incorporates SAM technique into the ZOO training process. By regulating the curvature of the loss landscape, SAM effectively steers the optimization process toward flatter minima, which in turn promotes improved generalization performance and heightened resilience against adversarial perturbations. Besides, we introduce a novel parallelized variant of the Randomized/Coordinate-wise Gradient Estimator which can reduce the computational burden. For the whole optimization framework, we present a thorough theoretical analysis from both the convergence properties and generalization guarantees. Empirically, we conduct extensive experiments involving a wide range of publicly available datasets and evaluate performance across various representative graph-oriented tasks. The results consistently demonstrate the effectiveness and versatility of the proposed methods, emphasizing their potential to drive advancements in the state-of-the-art in graph transformer optimization.

Our contributions are summarized as follows:

(1) Sharpness-aware Zeroth-order Optimization. We are the first to investigate the training performance of zeroth-order optimization for graph transformers and propose a sharpness-aware zeroth-order optimization (SZO) to enhance the robustness and generalization capabilities of the model.

(2) Comprehensive Theoretical Analysis. Our analysis provides an in-depth theoretical examination, addressing aspects of both generalization and convergence. We prove the convergence of this combined training approach and demonstrate that the generalization bound of SZO is confined to a neighborhood-wise loss.

(3) Experimental Evaluation. We evaluate SZO on various classical graph transformers and a wide range of public benchmark datasets. Extensive experimental results demonstrate the effectiveness in achieving superior performance compared to several modern optimizers and SAM variants.[1]

2 Mathematical Background

Notation. Given a training dataset $S \triangleq \bigcup_{i=1}^{n}\{(x_i, y_i)\}$ drawn i.i.d. from distribution $\mathcal{D} = \mathcal{X} \times \mathcal{Y}$, we aim to learn a graph transformer model that generalizes well. In particular, consider a family of models parameterized by $\theta \in \Theta \subseteq \mathbb{R}^d$. Given a per-data-point loss function $l : \Theta \times \mathcal{X} \times \mathcal{Y} \to \mathbb{R}_+$, we define the in-sample (empirical) training loss $L_S(\theta) \triangleq L_{in}(\theta) = \frac{1}{n}\sum_{i=1}^{n} l(\mathbf{w}, x_i, y_i)$ and the out-of-sample (theoretical) loss $L_{\mathcal{D}}(\theta) \triangleq L_{out}(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{D}}[l(\theta, x, y)]$. Having observed only S, the goal of training is to learn model parameters θ with lower loss $L_{\mathcal{D}}(\theta)$. For graph-level task, the input training data includes graphs $\{G^{(i)}\}_{i=1}^{N}$ (each graph $G^{(i)} = (X^{(i)}, A^{(i)})$) and associated labels $\{y^{(i)}\}_{i=1}^{N}$.

2.1 Graph Transformer

In this section, we present our focused framework, Graph Transformers (GTs). GTs enable each node in a graph to attend to all other nodes through a global attention mechanism. The architecture of GTs consists of two primary components: a self-attention module and a subsequent feed-forward neural network (FFN). In self-attention module, the input feature matrix $\mathbf{X} \in \mathbb{R}^{N\times d}$ is originally projected into three spaces: query matrix $\mathbf{Q} \in \mathbb{R}^{d\times d_q}$, key matrix $\mathbf{K} \in \mathbb{R}^{d\times d_k}$ and value matrix $\mathbf{V} \in \mathbb{R}^{d\times d_v}$ (where $d_q = d_k$):

$$\begin{aligned}\mathbf{Q} &= \mathbf{X}\mathbf{W}_{\mathbf{Q}} \in \mathbb{R}^{N\times d_k},\\ \mathbf{K} &= \mathbf{X}\mathbf{W}_{\mathbf{K}} \in \mathbb{R}^{N\times d_k},\\ \mathbf{V} &= \mathbf{X}\mathbf{W}_{\mathbf{V}} \in \mathbb{R}^{N\times d_v}.\end{aligned}$$

Then we can compute the self-attention score as:

$$\text{Attention}(\mathbf{X}) = \text{softmax}\left(\mathbf{Q}\mathbf{K}^T/\sqrt{d_v}\right)\mathbf{V} \in \mathbb{R}^{N\times d_v}. \quad (1)$$

The output of the self-attention module is followed by a skip-connection and a feed-forward network (FFN), which jointly compose a Transformer block:

$$\begin{aligned}\boldsymbol{X}'' &= \text{FFN}(\boldsymbol{X}') := \text{ReLU}(\boldsymbol{X}'\boldsymbol{W_1})\boldsymbol{W_2} + \boldsymbol{X}',\\ \boldsymbol{X}' &= \text{Attention}(\mathbf{X}),\end{aligned}$$

where $\boldsymbol{W_1}$ and $\boldsymbol{W_2}$ denote the parameters in FFN.

2.2 Zeroth-order Optimization

Zeroth-order optimization (ZOO) has gained significant attention due to its applicability in scenarios where the gradient of the objective function is not readily available or difficult to compute. In this section, we fisrt introduce two classical gradient estimators which are commonly employed to approximate the gradient of a target function. The first one, known as the Coordinate-wise Gradient Estimator (CGE), evaluates the partial derivatives of the function with respect to each separate coordinate. More formally, consider a differentiable function $f : \mathbb{R}^n \to \mathbb{R}$, and let $\boldsymbol{\theta} \in \mathbb{R}^n$ be the point where we aim to approximate the gradient $\nabla f(\boldsymbol{\theta})$. The

[1] https://github.com/liu-yang-maker/SZO

CGE method approximates each partial derivative $\frac{\partial f}{\partial \theta_i}$ by perturbing $\boldsymbol{\theta}$ along the i-th coordinate direction. Specifically, it computes $g_i(\boldsymbol{\theta}) \approx \frac{f(\boldsymbol{\theta}+\epsilon \boldsymbol{e}_i)-f(\boldsymbol{\theta})}{\epsilon}$, where $\epsilon > 0$ is a small constant and $\boldsymbol{e}_i$ represents the i-th standard basis vector in $\mathbb{R}^n$. By performing this operation for each coordinate i of $\boldsymbol{\theta}$, we obtain an approximation of the gradient $\nabla f(\boldsymbol{\theta})$. The formal CGE update procedure is thus defined by evaluating the above finite difference estimates for all coordinates and then assembling them into an approximate gradient vector as:

$$\boldsymbol{\theta_{t+1}} = \boldsymbol{\theta_t} - \eta \cdot \frac{1}{d} \sum_{i=1}^{d} \left[\frac{\ell(\boldsymbol{\theta_t} + \epsilon \boldsymbol{e_i}) - \ell(\boldsymbol{\theta_t})}{\epsilon} \boldsymbol{e_i} \right], \quad \text{(CGE)}$$

where $\boldsymbol{e_i}$ denotes the standard basis vector. Another zeroth-order gradient estimation approach is the Randomized Gradient Estimator (RGE), which offers a more flexible alternative to coordinate-wise perturbations by sampling random directions in the parameter space as:

$$\boldsymbol{\theta_{t+1}} = \boldsymbol{\theta_t} - \eta \cdot \frac{1}{q} \sum_{i=1}^{q} \left[\frac{\ell(\boldsymbol{\theta_t} + \epsilon \boldsymbol{u_i}) - \ell(\boldsymbol{\theta_t})}{\epsilon} \boldsymbol{u_i} \right], \quad \text{(RGE)}$$

where $\boldsymbol{u_i}$ represents a random direction vector, generally sampled from the standard Gaussian distribution $\mathcal{N}(0, \boldsymbol{I})$. The parameter q denotes the number of function queries, while $\epsilon > 0$ signifies a small perturbation step-size, also referred to as the smoothing parameter.

2.3 Sharpness-aware Minimization

The sharpness of the loss landscape in GTs under ZOO [Wang *et al.*, 2024] necessitates the introduction of an alternative optimization technique. To address this issue, we aim to flat the neighborhood of local minima, thereby enabling the development of a more robust and effective model. In this work, Sharpness-Aware Minimization (SAM) is utilized as a proposed solution [Foret *et al.*, 2020].

SAM Problem. Inspired by bounding generalization ability in terms of neighborhood-wise training loss, [Foret *et al.*, 2020] proposed Sharpness-Aware Minimization (SAM) problem as follows:

$$\min_{\boldsymbol{\theta}} L_S^{SAM}(\boldsymbol{\theta}) + \lambda \|\boldsymbol{\theta}\|_2^2, \tag{2}$$

where $L_S^{SAM}(\boldsymbol{\theta}) \triangleq \max_{\|\boldsymbol{\epsilon}\|_p \leq \rho} L_S(\boldsymbol{\theta} + \boldsymbol{\epsilon})$, $\rho \geq 0$ is a hyperparameter and $p \in [1, \infty]$.

Minimize $L_S^{SAM}(\theta)$. An efficient approximation of its gradient has been determined in as (details in Appendix)

$$\nabla_{\boldsymbol{\theta}} L_S^{\text{SAM}}(\boldsymbol{\theta}) \approx \nabla_{\boldsymbol{\theta}} L_S(\boldsymbol{\theta})\big|_{\boldsymbol{\theta}+\hat{\boldsymbol{\epsilon}}(\boldsymbol{\theta})}.$$

For each update step t, we seek to find the best update $\boldsymbol{\Delta} \in \mathbb{R}^d$ for $\boldsymbol{\theta}^{t+1} = \boldsymbol{\theta}^t - \eta \boldsymbol{\Delta}$, where $\eta \in \mathbb{R}_+$ denotes the learning rate. Then, we aim to solve the following optimization problem respect to $\boldsymbol{\Delta}$:

$$\min_{\boldsymbol{\Delta}} L(\boldsymbol{\theta} + \boldsymbol{\Delta}) \quad \text{s.t. } \mathrm{KL}\left(f(\boldsymbol{x}|\boldsymbol{\theta}) \| f(\boldsymbol{x}|\boldsymbol{\theta} + \boldsymbol{\Delta})\right) \leq \delta, \tag{3}$$

where δ represents the maximum step size and f refers to the original parametric model under consideration.

3 Proposed Approach

In this section, we describe SZO in details. First we give an overview procedure for SZO. We then describe how to design in technical view from the original version.

3.1 Proposed Optimization Procedure: SZO

Building on the SAM framework and the optimization challenges outlined earlier (see Eq.2 and Eq.3), we propose the Sharpness-aware Zeroth-order Optimization (SZO) scheme (here we take CGE as an example):

$$\begin{cases} \text{(I). } \boldsymbol{\epsilon}_t = \frac{1}{d} \sum_{i=1}^{d} \left[\frac{\ell(\boldsymbol{\theta}_t + \epsilon \boldsymbol{e}_i) - \ell(\boldsymbol{\theta}_t)}{\epsilon} \boldsymbol{e}_i \right], \\ \text{(II). } \hat{\boldsymbol{\epsilon}}_t = \rho \cdot \frac{\boldsymbol{\epsilon}_t}{\|\boldsymbol{\epsilon}_t\|_2}, \boldsymbol{\theta}_t^{\boldsymbol{\epsilon}} \triangleq \boldsymbol{\theta}_t + \hat{\boldsymbol{\epsilon}}_t, \\ \text{(III). } \hat{\boldsymbol{\Delta}}_t = \frac{1}{d} \sum_{i=1}^{d} \left[\frac{\ell(\boldsymbol{\theta}_t^{\boldsymbol{\epsilon}} + \epsilon \boldsymbol{e}_i) - \ell(\boldsymbol{\theta}_t^{\boldsymbol{\epsilon}})}{\epsilon} \boldsymbol{e}_i \right], \\ \text{(IV). } \boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta \cdot \hat{\boldsymbol{\Delta}}_t, \end{cases} \tag{4}$$

where ρ is the radius of the gradient stepping ball and η is the step size. A similar scheme with RGE can be derived by replacing the calculation of $\boldsymbol{\epsilon}_t$ and $\hat{\boldsymbol{\Delta}}_t$ above.

The training procedure of the SZO algorithm is outlined in Algorithm 1. The algorithm begins by initializing the model parameters θ (line 1). For each epoch t from 1 to *MaxIter* (line 2), a batch of graphs $\{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_b, y_b)\}$ is sampled from the training dataset $\mathbb{D}_{\text{train}}$ (line 3). In the First Differentiation Step, the approximate gradient $\boldsymbol{\epsilon}_t$ is computed using a coordinate-wise (or randomized) gradient estimator, achieved by perturbing each coordinate independently (line 4). The perturbed model parameters $\boldsymbol{\theta}_t^{\boldsymbol{\epsilon}}$ are then calculated (line 5). In the Second Differentiation Step, an additional gradient approximation $\hat{\boldsymbol{\Delta}}_t$ is computed (line 7). At the end of each epoch, the model parameters are updated by applying the computed gradient approximation via gradient descent (line 11).

3.2 Optimization Design

Given the huge complexity and parameter dimensions of graph transformers, traditional ZOO methods face efficiency bottlenecks. We implement algorithm acceleration using forward parallelization [Chen *et al.*, 2023]. Forward parallelization decouples parameter perturbations and leverages distributed computation, significantly reducing training time while maintaining performance. This acceleration is essential to meet the practical demands of large-scale graph data processing.

Decoupling Parameter Perturbations. For each coordinate in ZOO Perturbation, we select the active parameter to conduct the corresponding independent forward pass to allow for parallel computation. A demo procedure of parameters divided by $\boldsymbol{\theta_Q}, \boldsymbol{\theta_K}, \boldsymbol{\theta_V}$ (refer to $\boldsymbol{W_Q}, \boldsymbol{W_K}, \boldsymbol{W_V}$) can be described as

$$\boldsymbol{\epsilon} = \boldsymbol{\epsilon^Q} + \boldsymbol{\epsilon^K} + \boldsymbol{\epsilon^V}, \boldsymbol{\epsilon^I} = \sum_{i \in S_I} \left[\frac{\ell(\boldsymbol{\theta^I} + \epsilon \boldsymbol{e}_i) - \ell(\boldsymbol{\theta^I})}{\epsilon} \boldsymbol{e}_i \right], \tag{5}$$

where $\boldsymbol{I} \in \{\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}\}$ and $S_{\boldsymbol{I}}$ denote the active parameters assigned to module $\boldsymbol{I}$. In this case, only $|S_{\boldsymbol{I}}|$ forward passes are taken in each case.

Algorithm 1: Graph Transformer Training Procedure with SZO Optimizer

Input: Model Parameters $\boldsymbol{\theta}$; Some Hyper-parameters *MaxIter*, $K > 0$, batch size b, step size $\eta > 0$, neighborhood size $\rho > 0$.
Data: Training set $\mathbb{D}_{\text{train}} = \{(\boldsymbol{x}, y)\}$, testing graph set $\mathbb{D}_{\text{test}}$.
Output: Trained Parameters $\boldsymbol{\theta}$.

/* $\boldsymbol{\theta}$ denotes attention parameters. */
Initialize Parameters $\boldsymbol{\theta}$
/* begin update model parameters. */
for *epoch* $t = 1 \to$ *MaxIter* **do**
  Sample graph batch $\{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_b, y_b)\}$ from $\mathbb{D}_{\text{train}}$
  /* 1st differentiation step. */
  Compute gradient $\boldsymbol{\epsilon_t}$
  /* We only use one rule as: */
  [CGE Rule] $\boldsymbol{\epsilon}_t = \frac{1}{d}\sum_{i=1}^{d}\left[\frac{\ell(\boldsymbol{\theta}_t+\epsilon\boldsymbol{e}_i)-\ell(\boldsymbol{\theta}_t)}{\epsilon}\boldsymbol{e}_i\right]$
  [RGE Rule] $\boldsymbol{\epsilon}_t = \frac{1}{q}\sum_{i=1}^{q}\left[\frac{\ell(\boldsymbol{\theta}_t+\epsilon\boldsymbol{u}_i)-\ell(\boldsymbol{\theta}_t)}{\epsilon}\boldsymbol{u}_i\right]$
  Compute $\boldsymbol{\theta}_t^{\epsilon}$ by ZO perturbation $\boldsymbol{\theta}_t^{\epsilon} \triangleq \boldsymbol{\theta}_t + \rho \cdot \frac{\boldsymbol{\epsilon}_t}{\|\boldsymbol{\epsilon}_t\|_2}$
  /* 2nd differentiation step. */
  Compute gradient approximation $\hat{\boldsymbol{\Delta}}_t$ (see Eq.2)
  /* We only use one rule as: */
  [CGE Rule] $\hat{\boldsymbol{\Delta}}_t = \frac{1}{d}\sum_{i=1}^{d}\left[\frac{\ell(\boldsymbol{\theta}_t^{\epsilon}+\epsilon\boldsymbol{e}_i)-\ell(\boldsymbol{\theta}_t^{\epsilon})}{\epsilon}\boldsymbol{e}_i\right]$
  [RGE Rule] $\hat{\boldsymbol{\Delta}}_t = \frac{1}{q}\sum_{i=1}^{q}\left[\frac{\ell(\boldsymbol{\theta}_t^{\epsilon}+\epsilon\boldsymbol{u}_i)-\ell(\boldsymbol{\theta}_t^{\epsilon})}{\epsilon}\boldsymbol{u}_i\right]$
  Update Parameters $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \eta \cdot \hat{\boldsymbol{\Delta}}_t$
  if $\hat{\boldsymbol{\Delta}}_t$ *satisfy early stop condition* **then**
    Break

Distributed and Sparse Implementation. The computation is distributed across multiple processes (or GPUs). Each process is assigned a subset of the parameters to perturb and compute the corresponding forward passes. By distributing the parameter perturbations and forward passes across multiple machines, the training process is significantly accelerated. This approach avoids the limitations of traditional data parallelization, such as the potential for performance loss with large batch sizes. Furthermore, we use a top-K operator before each variable matrix in attention as $\mathbf{Y} = (\mathbf{X} \odot \mathbf{M}) \cdot \mathbf{W}^T$, where $\mathbf{M} \in \{0,1\}^{N \times D}$ denotes $\text{Top}_k(|\mathbf{X}|)$ which is the mask tensor that indicates the top-K activations in the input tensor $\mathbf{X}$ with absolute values and $\odot$ is the Hadamard product. This enhances the efficiency of SZO by ensuring high computational efficiency while maintaining performance.

3.3 SAM-based Design

We employ the Sharpness-Aware Minimization scheme (SAM, see Section 2.3) [Foret *et al.*, 2020; Wang *et al.*, 2024] for zeroth-order optimization.

SAM scheme. We introduce a SAM scheme in our method for better generalization [Wang *et al.*, 2024]. As discussed in Section 2.3, we perform the gradient estimation operation twice, regardless of whether using CGE, RGE, or a combination of both. This training scheme improves the model's generalization by facilitating a flatter loss landscape.

Perturbation size scheduler. $\epsilon_{i+1} = \epsilon_0 \cdot \gamma^{\text{epoch}/\lambda}$ where the hyperparameter λ determines the update frequency of ϵ, specifying the number of epochs per update, while γ regulates the scale of modifications in ϵ.

3.4 Theoretical Analysis

The training pipeline of SZO is illustrated in Algorithm 1. In this section, we provide a justification for the convergence and generalization ability of SZO to ensure a reliable training process. Note that the theorems only depend on the vanilla setting and ignore the sparsity improvement.

Convergence Justification

Theorem 3.1. *Considering an α-strongly convex and β-smooth loss function ℓ, if we run SZO optimizer starting at $\boldsymbol{\theta}_0$ with perturbation radius $\epsilon > 0$ and any learning rate $\eta < \frac{2}{\beta}$ to minimize ℓ, we have:*

$$\ell(\boldsymbol{\theta}_t) - \ell(\boldsymbol{\theta}^\star) \leq (1 - \alpha\eta(2-\eta\beta))^t(\ell(\boldsymbol{\theta}_0) - \ell(\boldsymbol{\theta}^\star) + \frac{\eta\beta^4\epsilon}{2\alpha(2-\eta\beta)}, \quad \forall t.$$

Generalization Justification

The original SAM [Foret *et al.*, 2020] approach was inspired by the established connection between the sharpness of the loss landscape and the model's generalization capabilities. To further elucidate this motivation, the following analogous theorem demonstrates how generalization ability can be bounded by neighborhood-wise training loss:

Theorem 3.2 (informal)**.** *For any $\epsilon > 0$ and any distribution $\mathcal{D}$, with probability $1 - \delta$ over the choice of the training set $\mathcal{S} \sim \mathcal{D}$, we have*

$$\ell_{\mathcal{D}}(\boldsymbol{\theta}) \leq \max_{i=1,\ldots,d} \ell_{\mathcal{S}}(\boldsymbol{\theta} + \epsilon\boldsymbol{e}_i) + h\left(\frac{\|\boldsymbol{\theta}\|_2^2}{\epsilon^2}\right)$$

where $h : \mathbb{R}^+ \to \mathbb{R}^+$ is a strictly increasing function under some technical conditions on $\ell_{\mathcal{D}}$ and we assumed $\ell_{\mathcal{D}}(\boldsymbol{\theta}) \leq \mathbb{E}_{\epsilon\sim\mathcal{N}(0,\delta')}[\ell_{\mathcal{D}}(\boldsymbol{\theta} + \epsilon\boldsymbol{e}_i)]$.

Remark 1. *Theorem 3.1 establishes a solid foundation for the reliability and effectiveness of the proposed SZO method. The convergence results demonstrate that SZO achieves guaranteed progress under both strongly convex and smooth loss functions, ensuring stable optimization with appropriate learning rates. The generalization analysis (see Theorem 3.2) highlights the role of the SAM framework in achieving flatter minima, which contributes to better robustness and improved generalization capabilities. Together, these theoretical results validate the efficiency of SZO for optimizing complex graph transformer models.*

Dataset	#Tasks	Task Type	#Molecule
BBBP	1	Graph Classification	2,039
Tox21	12	Graph Classification	7,831
Sider	27	Graph Classification	1,427
Clintox	2	Graph Classification	1,478
BACE	1	Graph Classification	1,522
ESOL	1	Graph Regression	1,128
Lipophilicity	1	Graph Regression	4,198
QM7	1	Graph Regression	7,165
QM8	23	Graph Regression	21,786

Table 1: Statistics of datasets.

Models	Graph Classification (ROC-AUC ↑)				
	BBBP	**Tox21**	**Sider**	**ClinTox**	**BACE**
GCN	$0.690_{\pm 0.041}$	$0.819_{\pm 0.031}$	$0.623_{\pm 0.022}$	$0.807_{\pm 0.044}$	$0.725_{\pm 0.027}$
MPNN	$0.901_{\pm 0.032}$	$0.834_{\pm 0.014}$	$0.634_{\pm 0.017}$	$0.881_{\pm 0.037}$	$0.791_{\pm 0.019}$
DMPNN	$0.912_{\pm 0.037}$	$0.845_{\pm 0.012}$	$0.646_{\pm 0.020}$	$0.897_{\pm 0.042}$	$0.809_{\pm 0.022}$
CMPNN	$0.925_{\pm 0.017}$	$0.837_{\pm 0.016}$	$0.640_{\pm 0.018}$	$0.918_{\pm 0.016}$	$0.826_{\pm 0.020}$
SPMM	$0.733_{\pm 0.023}$	-	$0.647_{\pm 0.023}$	$0.914_{\pm 0.045}$	$0.830_{\pm 0.010}$
PretrainGNN	$0.687_{\pm 0.022}$	$0.781_{\pm 0.020}$	$0.627_{\pm 0.029}$	$0.726_{\pm 0.018}$	$0.845_{\pm 0.019}$
Uni-Mol	$0.729_{\pm 0.027}$	$0.796_{\pm 0.029}$	$0.659_{\pm 0.018}$	$0.919_{\pm 0.012}$	$0.857_{\pm 0.017}$
MolXPT	$0.805_{\pm 0.050}$	$0.771_{\pm 0.044}$	$\mathbf{0.717}_{\pm 0.049}$	$\mathbf{0.953}_{\pm 0.002}$	$0.884_{\pm 0.011}$
GROVER	$0.917_{\pm 0.028}$	$0.822_{\pm 0.019}$	$0.649_{\pm 0.035}$	$0.853_{\pm 0.043}$	$0.871_{\pm 0.022}$
+SAM	$0.926_{\pm 0.022}$	$0.840_{\pm 0.035}$	$0.660_{\pm 0.043}$	$0.872_{\pm 0.044}$	$\underline{0.886}_{\pm 0.019}$
+GraphSAM	$0.928_{\pm 0.016}$	$\underline{0.846}_{\pm 0.012}$	$\underline{0.665}_{\pm 0.038}$	$0.866_{\pm 0.051}$	$0.882_{\pm 0.039}$
+SZO	$0.929_{\pm 0.018}$	$\mathbf{0.848}_{\pm 0.040}$	$\underline{0.707}_{\pm 0.035}$	$0.874_{\pm 0.050}$	$\underline{0.885}_{\pm 0.019}$
CoMPT	$0.948_{\pm 0.025}$	$0.828_{\pm 0.008}$	$0.621_{\pm 0.013}$	$0.914_{\pm 0.034}$	$0.863_{\pm 0.019}$
+SAM	$\underline{0.962}_{\pm 0.033}$	$0.839_{\pm 0.006}$	$0.643_{\pm 0.009}$	$0.927_{\pm 0.025}$	$0.876_{\pm 0.012}$
+GraphSAM	$\underline{0.961}_{\pm 0.012}$	$0.841_{\pm 0.004}$	$0.645_{\pm 0.013}$	$\underline{0.937}_{\pm 0.008}$	$0.880_{\pm 0.029}$
+SZO	$\mathbf{0.964}_{\pm 0.020}$	$\underline{0.843}_{\pm 0.006}$	$0.649_{\pm 0.010}$	$\underline{0.940}_{\pm 0.012}$	$\mathbf{0.888}_{\pm 0.044}$

Table 2: Graph Classification (ROC-AUC) Results. We use **boldface** to denote the best result, underline to denote the second best result, and wavy line to denote the third best result. Baseline results are from original papers and "-" indicates missing data in original papers.

4 Experiment

4.1 Experimental Setup

Dataset. Following established settings in molecular graph tasks, we utilize nine public benchmark datasets: BBBP, Tox21, Sider, ClinTox and BACE for classification, and ESOL, Lipophilicity, QM7 and QM8 for regression. We assess all models using a random split methodology as recommended by MoleculeNet [Wu *et al.*, 2018], dividing the datasets into training, validation, and testing sets with an 80%/10%/10% ratio. In Table 1, we summarize the statistics of the molecular graph datasets utilized in graph classification and graph regression tasks. Details see Appendix.

Baseline. We evaluate the effectiveness of SZO on two graph transformer models including GROVER and CoMPT [Rong *et al.*, 2020; Chen *et al.*, 2021] with different modern optimizers [Wang *et al.*, 2024]. Also, we add several competitive GNN-based models as baselines.

- GROVER [Rong *et al.*, 2020], which stands for Graph Representation frOm self-superVised mEssage passing tRansformer, is a novel framework designed to address the challenges in molecular representation learning. It leverages self-supervised tasks at the node, edge, and graph levels to learn rich structural and semantic information from unlabelled molecular data. GROVER integrates Message Passing Networks into a Transformer-style architecture, enhancing its capability to encode complex molecular information.
- CoMPT [Chen *et al.*, 2021]. The Communicative Message Passing Transformer (CoMPT) is a neural network designed to improve molecular graph representation. It enhances traditional Graph Neural Networks by strengthening interactions between nodes and edges using Transformer architecture. CoMPT includes a message diffusion mechanism to manage graph connectivity and prevent information overload, capturing both local and global structure.

Implementations. The implementations of the backbone models and their respective hyperparameter configurations are sourced from publicly available repositories as detailed in [Rong *et al.*, 2020] and [Chen *et al.*, 2021]. Both GROVER and CoMPT employ Adam as the base optimizer without employing any pre-training strategies. In our experiments, we solely adjust the hyperparameters introduced by SZO.

We implemented several optimizers (i.e. SGD, ADAM and SZO) on molecular graph data using two widely-adopted graph transformer backbones: GROVER [Rong *et al.*, 2020] and CoMPT [Chen *et al.*, 2021]. A comprehensive comparison was conducted against several baseline models including GCN [Kipf and Welling, 2016], MPNN [Gilmer *et al.*, 2017], DMPNN [Yang *et al.*, 2019] CMPNN [Song *et al.*, 2020], SPMM [Chang and Ye, 2024], PretrainGNN [Hu *et al.*, 2020], and MolXPT [Liu *et al.*, 2023b].

SAM and its Variants. Recent advancements in Sharpness-Aware Minimization (SAM) have introduced several efficient variants. The original SAM [Foret *et al.*, 2020] prevents sharp minima convergence but doubles computational costs. LookSAM [Liu *et al.*, 2022b] improves efficiency by intermittently computing gradients but neglects gradient magnitude, limiting its generalizability. AE-SAM [Jiang *et al.*, 2023] uses squared gradient norms to skip updates, though its performance varies across domains. RST [Zhao *et al.*, 2022] employs Bernoulli trials for stochastic updates, yielding suboptimal results. GraphSAM [Wang *et al.*, 2024] enhances efficiency for molecular graph transformers by reusing prior gradients, maintaining strong generalization.

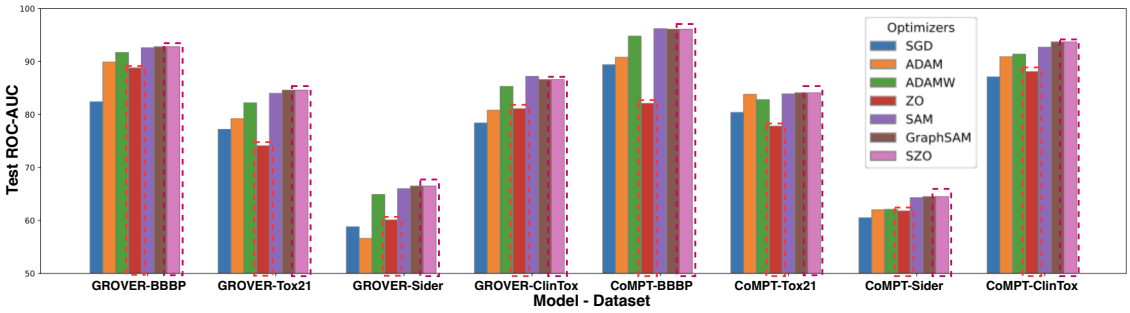

Figure 2: Testing ROC-AUC of graph transformers on graph classification benchmark datasets with SZO and other competitive optimizers. We emphasize the use of two optimizers, ZO and SZO, with bold box.

4.2 Performance on Graph Classification Task

We employed SZO on GROVER and CoMPT for an extensive evaluation against baseline methods across various datasets. The comparative results are presented in Table 2. It demonstrates that the impact of integrating SZO with GROVER and CoMPT models on graph classification tasks across six datasets (BBBP, Tox21, Sider, ClinTox, BACE). The SZO-enhanced versions of both models consistently outperform their baseline counterparts, achieving some of the highest ROC-AUC scores, particularly on BBBP, Tox21, and ClinTox datasets. Notable improvements include GROVER with SZO reaching 0.929 in BBBP and CoMPT with SZO achieving 0.940 in ClinTox. This highlights the effectiveness of SZO in boosting model performance.

4.3 Performance on Graph Regression Task

Table 3 displays the performance of various models on graph regression tasks, evaluated using RMSE and MAE across four datasets (ESOL, Lipophilicity, QM7, QM8). The integration of SZO with GROVER and CoMPT models generally leads to improved results. For instance, CoMPT with SZO achieves the lowest RMSE in ESOL (0.505) and the best MAE in QM8 (0.0139). Similarly, GROVER with SZO shows significant improvements, notably achieving the best RMSE in Lipophilicity (0.588). These results highlight the effectiveness of SZO in enhancing model accuracy for graph regression tasks.

Models	(RMSE ↓)		(MAE ↓)	
	ESOL	Lip.	QM7	QM8
GCN	$0.970_{\pm 0.071}$	$1.313_{\pm 0.149}$	131.1	0.0236
MPNN	$0.702_{\pm 0.042}$	$1.242_{\pm 0.249}$	94.5	0.0218
DMPNN	$0.665_{\pm 0.060}$	$1.159_{\pm 0.207}$	103.5	0.0190
CMPNN	$0.582_{\pm 0.055}$	$0.633_{\pm 0.029}$	81.9	0.0171
SPMM	$0.810_{\pm 0.066}$	$0.706_{\pm 0.223}$	-	-
PretrainGNN	-	$0.739_{\pm 0.118}$	113.2	0.0200
Uni-Mol	$0.788_{\pm 0.012}$	$\underline{0.603}_{\pm 0.017}$	**41.8**	0.0156
GROVER	$0.639_{\pm 0.087}$	$0.671_{\pm 0.047}$	78.9	0.0203
+SAM	$0.619_{\pm 0.089}$	$0.662_{\pm 0.052}$	76.4	0.0189
+GraphSAM	$0.625_{\pm 0.083}$	$0.654_{\pm 0.056}$	75.8	0.0185
+SZO	$0.598_{\pm 0.087}$	$\mathbf{0.588}_{\pm 0.119}$	73.3	0.0181
CoMPT	$0.562_{\pm 0.071}$	$0.618_{\pm 0.012}$	66.1	0.0159
+SAM	$\underline{0.517}_{\pm 0.025}$	$0.611_{\pm 0.015}$	64.3	$\underline{0.0145}$
+GraphSAM	$\underline{0.511}_{\pm 0.018}$	$\underline{0.608}_{\pm 0.007}$	$\underline{64.1}$	$\underline{0.0141}$
+SZO	$\mathbf{0.505}_{\pm 0.033}$	$0.609_{\pm 0.66}$	$\underline{62.2}$	**0.0139**

Table 3: Graph Regression (RMSE & MAE) Results. We use **boldface** to denote the best result, underline to denote the second best result, and wavy line to denote the third best result. Baseline results are sourced from the original papers, and "-" indicates missing results in the original papers.

4.4 Additional Verification

Ablation Study. The performance results of the ZO and SZO optimizer across four datasets are presented in Figure 2. It is evident that the ZO optimizer alone yields subpar outcomes, demonstrating a lack of effectiveness in achieving optimal performance on the datasets. However, when SAM is integrated into the optimization process, there is a substantial enhancement in performance. The addition of SAM helps to address the limitations of the ZO optimizer, significantly boosting the model's accuracy and stability across all datasets, thus showcasing the considerable benefits of combining SAM with the ZO optimization approach.

Different Optimizers. Table 4 shows the performance of GROVER and CoMPT models with various optimizers across different datasets for both graph classification (ROC-AUC) and graph regression (RMSE). For GROVER, the RSZO optimizer consistently achieves high scores, such as 0.929 in BBBP and 0.598 in ESOL. Similarly, for CoMPT, RSZO yields excellent results, including 0.964 in BBBP and 0.505 in ESOL. Figure 2 visualizes the total ROC-AUC scores for these models and datasets, clearly indicating that RSZO (purple) and SZO (pink) optimizers lead to superior performance, particularly in the datasets of BBBP, ClinTox, and Tox21. This demonstrates the effectiveness of SZO in optimizing model performance across various tasks.

5 Related works

Graph Transformer. The remarkable achievements of Transformers [Vaswani *et al.*, 2017] in natural language processing (NLP), and more recently in computer vision [Han *et al.*, 2021] and biological information [Bai *et al.*, 2023; Niu *et al.*, 2022; Wu *et al.*, 2022], have generated substantial interest in adapting Transformers for graph data analysis. One of the

Optimizers \ Tasks	Graph Classification (ROC-AUC ↑)				Graph Regression (RMSE ↓)	
GROVER+	**BBBP**	**Tox21**	**Sider**	**ClinTox**	**ESOL**	**Lipophilicity**
ADAM	$0.917_{\pm 0.028}$	$0.822_{\pm 0.019}$	$0.649_{\pm 0.035}$	$0.853_{\pm 0.043}$	$0.639_{\pm 0.045}$	$0.671_{\pm 0.077}$
SAM	$\underline{0.926}_{\pm 0.022}$	$\underline{0.840}_{\pm 0.035}$	$\underline{0.660}_{\pm 0.043}$	$\underline{0.872}_{\pm 0.044}$	$\underline{0.619}_{\pm 0.022}$	$0.662_{\pm 0.039}$
SAM-One	$0.900_{\pm 0.019}$	$0.801_{\pm 0.009}$	$0.610_{\pm 0.018}$	$0.818_{\pm 0.043}$	$0.671_{\pm 0.017}$	$0.715_{\pm 0.045}$
SAM-k	$0.892_{\pm 0.032}$	$0.815_{\pm 0.020}$	$0.631_{\pm 0.062}$	$0.836_{\pm 0.033}$	$0.650_{\pm 0.033}$	$0.698_{\pm 0.018}$
LookSAM	$0.889_{\pm 0.031}$	$0.829_{\pm 0.019}$	$0.653_{\pm 0.008}$	$0.847_{\pm 0.048}$	$0.633_{\pm 0.030}$	$0.680_{\pm 0.038}$
AE-SAM	$0.899_{\pm 0.027}$	$0.834_{\pm 0.028}$	$0.649_{\pm 0.023}$	$0.858_{\pm 0.035}$	$0.631_{\pm 0.013}$	$0.674_{\pm 0.077}$
RST	$0.911_{\pm 0.022}$	$0.818_{\pm 0.017}$	$0.625_{\pm 0.009}$	$0.845_{\pm 0.044}$	$0.638_{\pm 0.028}$	$0.690_{\pm 0.019}$
GraphSAM	$\underline{0.928}_{\pm 0.016}$	$\underline{0.846}_{\pm 0.012}$	$\underline{0.665}_{\pm 0.038}$	$0.866_{\pm 0.051}$	$\underline{0.625}_{\pm 0.026}$	$\underline{0.654}_{\pm 0.044}$
CSZO	$0.927_{\pm 0.022}$	$\mathbf{0.848}_{\pm 0.040}$	$0.662_{\pm 0.024}$	$\underline{0.870}_{\pm 0.035}$	$0.627_{\pm 0.103}$	$\underline{0.660}_{\pm 0.060}$
RSZO	$\mathbf{0.929}_{\pm 0.018}$	$\underline{0.846}_{\pm 0.032}$	$\mathbf{0.707}_{\pm 0.035}$	$\mathbf{0.874}_{\pm 0.050}$	$\mathbf{0.598}_{\pm 0.087}$	$\mathbf{0.588}_{\pm 0.119}$
CoMPT+	**BBBP**	**Tox21**	**Sider**	**ClinTox**	**ESOL**	**Lipophilicity**
ADAM	$0.948_{\pm 0.025}$	$0.828_{\pm 0.008}$	$0.621_{\pm 0.013}$	$0.914_{\pm 0.034}$	$0.562_{\pm 0.026}$	$0.618_{\pm 0.015}$
SAM	$\underline{0.962}_{\pm 0.033}$	$0.839_{\pm 0.006}$	$0.643_{\pm 0.009}$	$0.927_{\pm 0.025}$	$0.517_{\pm 0.071}$	$\underline{0.611}_{\pm 0.126}$
SAM-One	$0.933_{\pm 0.025}$	$0.788_{\pm 0.012}$	$0.595_{\pm 0.030}$	$0.879_{\pm 0.032}$	$0.601_{\pm 0.088}$	$0.638_{\pm 0.073}$
SAM-k	$0.959_{\pm 0.045}$	$0.819_{\pm 0.073}$	$0.610_{\pm 0.034}$	$0.895_{\pm 0.047}$	$0.575_{\pm 0.043}$	$0.630_{\pm 0.030}$
LookSAM	$0.955_{\pm 0.034}$	$0.825_{\pm 0.071}$	$0.625_{\pm 0.024}$	$0.916_{\pm 0.022}$	$0.543_{\pm 0.019}$	$0.621_{\pm 0.041}$
AE-SAM	$0.955_{\pm 0.022}$	$0.833_{\pm 0.075}$	$0.631_{\pm 0.050}$	$0.909_{\pm 0.024}$	$0.527_{\pm 0.043}$	$0.615_{\pm 0.020}$
RST	$0.960_{\pm 0.008}$	$0.815_{\pm 0.033}$	$0.615_{\pm 0.027}$	$0.890_{\pm 0.043}$	$0.588_{\pm 0.014}$	$0.633_{\pm 0.044}$
GraphSAM	$\underline{0.961}_{\pm 0.012}$	$\underline{0.841}_{\pm 0.004}$	$\underline{0.645}_{\pm 0.013}$	$\underline{0.937}_{\pm 0.008}$	$\underline{0.511}_{\pm 0.044}$	$\mathbf{0.608}_{\pm 0.030}$
CSZO	$\underline{0.961}_{\pm 0.024}$	$\mathbf{0.843}_{\pm 0.006}$	$\underline{0.644}_{\pm 0.09}$	$\mathbf{0.940}_{\pm 0.012}$	$\mathbf{0.505}_{\pm 0.033}$	$0.616_{\pm 0.014}$
RSZO	$\mathbf{0.964}_{\pm 0.020}$	$\underline{0.840}_{\pm 0.008}$	$\mathbf{0.649}_{\pm 0.010}$	$\underline{0.939}_{\pm 0.019}$	$\underline{0.510}_{\pm 0.013}$	$\underline{0.609}_{\pm 0.66}$

Table 4: Additional Results with Different Optimizers. We use **boldface** to denote the best result, underline to denote the second best result, and wavy line to denote the third best result.

widely recognized benefits of graph Transformers compared to message-passing neural networks (MPNNs) is their capacity to capture long-range interactions, effectively addressing challenges such as over-smoothing and over-squashing. The introduction of the fully-connected Graph Transformer [Dwivedi and Bresson, 2020] utilized the eigenvectors of the graph Laplacian for node positional encoding (PE). Subsequent research has introduced numerous PE methods to advance Graph Transformers. These include the invariant aggregation of Laplacian eigenvectors in SAN [Ranzato *et al.*, 2021], pair-wise graph distances in Graphormer [Ying *et al.*, 2021] and more. Future research on Graph Transformers includes integrating MPNNs, improving substructure encoding, and optimizing for directed graphs.

Zeroth-order Optimization. A subclass of gradient-free optimization known as zeroth-order (ZO) optimization [Liu *et al.*, 2020; Chen *et al.*, 2023; Liu *et al.*, 2023a] is used in many machine learning domains such as large language model. It is employed to handle optimization problems similar to gradient-based techniques. However, it only uses function values and does not require a gradient. The potential uses of ZO optimization are showcased, including but not limited to the assessment of the resilience of deep learning models [Chen *et al.*, 2017; Wang *et al.*, 2020] and the production of justifications from implicit systems, as well as the effective management of sensors in real-time scenarios.

SAM Optimization. Sharpness-Aware Minimization (SAM) has demonstrated significant effectiveness across diverse domains, including image classification [Foret *et al.*, 2020; Kwon *et al.*, 2021; Du *et al.*, 2022; Wang *et al.*, 2024; Ilbert *et al.*, 2024] and natural language processing [Bahri *et al.*, 2021]. [Foret *et al.*, 2020] proposed SAM during training to improve standard generalization. In particular, LookSAM [Liu *et al.*, 2022b] improves computational efficiency by eliminating the requirement to compute the updating gradient. [Du *et al.*, 2022] proposed SAF, an innovative trajectory loss designed to mitigate sudden decreases in loss at sharp local minima during weight updates, thereby reducing the associated time loss. Additionally, the lack of SAM research in the graph domain necessitates the development of GraphSAM [Wang *et al.*, 2024], designed to preserve SAM's generalization abilities while simultaneously improving computational efficiency.

6 Conclusion

In this work, our exploration into Sharpness-aware Zeroth-order Optimization (SZO) for graph transformers has revealed significant advancements in model performance and robustness. By integrating SAM and parallelized gradient estimation techniques, SZO has demonstrated robust generalization capabilities without sharp loss landscape and better performance across various GT models. This study underscores the potential of SZO to transform optimization practices in graph-based neural networks.

Acknowledgements

The AI-driven experiments, simulations and model training were performed on the robotic AI-Scientist platform of Chinese Academy of Science. This work was partially supported by the Strategic Priority Research Program of the Chinese Academy of Sciences (No. XDB0680101), the National Natural Science Foundation of China (No. 62472416), and the CAS Project for Young Scientists in Basic Research (No. YSBR-008).

References

[Bahri *et al.*, 2021] Dara Bahri, Hossein Mobahi, and Yi Tay. Sharpness-aware minimization improves language model generalization. *arXiv preprint arXiv:2110.08529*, 2021.

[Bai *et al.*, 2023] Peizhen Bai, Xianyuan Liu, and Haiping Lu. Geometry-aware line graph transformer pre-training for molecular property prediction. *arXiv preprint arXiv:2309.00483*, 2023.

[Chang and Ye, 2024] Jinho Chang and Jong Chul Ye. Bidirectional generation of structure and properties through a single molecular foundation model. *Nature Communications*, 15(1):2323, 2024.

[Chang *et al.*, 2025] Zongyu Chang, Feihong Lu, Ziqin Zhu, Qian Li, Cheng Ji, Zhuo Chen, Yang Liu, Ruifeng Xu, Yangqiu Song, Shangguang Wang, et al. Bridging the gap between llms and human intentions: Progresses and challenges in instruction understanding, intention reasoning, and reliable generation. *arXiv preprint arXiv:2502.09101*, 2025.

[Chen *et al.*, 2017] Pin-Yu Chen, Huan Zhang, Yash Sharma, Jinfeng Yi, and Cho-Jui Hsieh. Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In *Proceedings of the 10th ACM workshop on artificial intelligence and security*, pages 15–26, 2017.

[Chen *et al.*, 2021] Jianwen Chen, Shuangjia Zheng, Ying Song, Jiahua Rao, and Yuedong Yang. Learning attributed graph representations with communicative message passing transformer. *arXiv preprint arXiv:2107.08773*, 2021.

[Chen *et al.*, 2023] Aochuan Chen, Yimeng Zhang, Jinghan Jia, James Diffenderfer, Jiancheng Liu, Konstantinos Parasyris, Yihua Zhang, Zheng Zhang, Bhavya Kailkhura, and Sijia Liu. Deepzero: Scaling up zeroth-order optimization for deep model training. *arXiv preprint arXiv:2310.02025*, 2023.

[Du *et al.*, 2022] Jiawei Du, Daquan Zhou, Jiashi Feng, Vincent Tan, and Joey Tianyi Zhou. Sharpness-aware training for free. *Advances in Neural Information Processing Systems*, 35:23439–23451, 2022.

[Dwivedi and Bresson, 2020] Vijay Prakash Dwivedi and Xavier Bresson. A generalization of transformer networks to graphs. *arXiv preprint arXiv:2012.09699*, 2020.

[Foret *et al.*, 2020] Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. In *International Conference on Learning Representations*, 2020.

[Gilmer *et al.*, 2017] Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. PMLR, 2017.

[Han *et al.*, 2021] Kai Han, An Xiao, Enhua Wu, Jianyuan Guo, Chunjing Xu, and Yunhe Wang. Transformer in transformer. *Advances in neural information processing systems*, 34:15908–15919, 2021.

[He *et al.*, 2022] Xiaoyu He, Zibin Zheng, Zefeng Chen, and Yuren Zhou. Adaptive evolution strategies for stochastic zeroth-order optimization. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 6(5):1271–1285, 2022.

[Hu *et al.*, 2020] Weihua Hu, Bowen Liu, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. Strategies for pre-training graph neural networks. In *International Conference on Learning Representations*, 2020.

[Ilbert *et al.*, 2024] Romain Ilbert, Ambroise Odonnat, Vasilii Feofanov, Aladin Virmaux, Giuseppe Paolo, Themis Palpanas, and Ievgen Redko. Unlocking the potential of transformers in time series forecasting with sharpness-aware minimization and channel-wise attention. *arXiv preprint arXiv:2402.10198*, 2024.

[Jiang *et al.*, 2023] Weisen Jiang, Hansi Yang, Yu Zhang, and James Kwok. An adaptive policy to employ sharpness-aware minimization. *arXiv preprint arXiv:2304.14647*, 2023.

[Kipf and Welling, 2016] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[Kunstner *et al.*, 2023] Frederik Kunstner, Jacques Chen, Jonathan Wilder Lavington, and Mark Schmidt. Noise is not the main factor behind the gap between sgd and adam on transformers, but sign descent might be. *arXiv preprint arXiv:2304.13960*, 2023.

[Kwon *et al.*, 2021] Jungmin Kwon, Jeongseop Kim, Hyunseo Park, and In Kwon Choi. Asam: Adaptive sharpness-aware minimization for scale-invariant learning of deep neural networks. In *International Conference on Machine Learning*, pages 5905–5914. PMLR, 2021.

[Liu *et al.*, 2020] Sijia Liu, Pin-Yu Chen, Bhavya Kailkhura, Gaoyuan Zhang, Alfred O Hero III, and Pramod K Varshney. A primer on zeroth-order optimization in signal processing and machine learning: Principals, recent advances, and applications. *IEEE Signal Processing Magazine*, 37(5):43–54, 2020.

[Liu *et al.*, 2022a] Siyuan Liu, Yusong Wang, Yifan Deng, Liang He, Bin Shao, Jian Yin, Nanning Zheng, Tie-Yan Liu, and Tong Wang. Improved drug–target interaction prediction with intermolecular graph transformer. *Briefings in Bioinformatics*, 23(5):bbac162, 2022.

[Liu *et al.*, 2022b] Yong Liu, Siqi Mai, Xiangning Chen, Cho-Jui Hsieh, and Yang You. Towards efficient and scalable sharpness-aware minimization. In *Proceedings of the*

IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 12360–12370, 2022.

[Liu *et al.*, 2023a] Yang Liu, Chuan Zhou, Peng Zhang, Shuai Zhang, Xiaoou Zhang, Zhao Li, and Hongyang Chen. Decision-focused graph neural networks for graph learning and optimization. In *2023 IEEE International Conference on Data Mining (ICDM)*, pages 1151–1156. IEEE, 2023.

[Liu *et al.*, 2023b] Zequn Liu, Wei Zhang, Yingce Xia, Lijun Wu, Shufang Xie, Tao Qin, Ming Zhang, and Tie-Yan Liu. Molxpt: Wrapping molecules with text for generative pre-training. *arXiv preprint arXiv:2305.10688*, 2023.

[Liu *et al.*, 2024a] Yang Liu, Chuan Zhou, Huang Fang, Peng Zhang, Yiling Pang, Qingzhao Zhang, Zhao Li, and Hongyang Chen. Secure causal reasoning on coarsened graph for privacy-aware social web services. In *2024 IEEE International Conference on Web Services (ICWS)*, pages 557–567. IEEE, 2024.

[Liu *et al.*, 2024b] Yang Liu, Chuan Zhou, Peng Zhang, Zhao Li, Shuai Zhang, Xixun Lin, and Xindong Wu. Cl4co: A curriculum training framework for graph-based neural combinatorial optimization. In *2024 IEEE International Conference on Data Mining (ICDM)*, pages 779–784. IEEE, 2024.

[Niu *et al.*, 2022] Peisong Niu, Tian Zhou, Qingsong Wen, Liang Sun, and Tao Yao. Chemistry guided molecular graph transformer. In *NeurIPS 2022 AI for Science: Progress and Promises*, 2022.

[Ranzato *et al.*, 2021] Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors. *Advances in Neural Information Processing Systems 34*, 2021.

[Rong *et al.*, 2020] Yu Rong, Yatao Bian, Tingyang Xu, Weiyang Xie, Ying Wei, Wenbing Huang, and Junzhou Huang. Self-supervised graph transformer on large-scale molecular data. *Advances in neural information processing systems*, 33:12559–12571, 2020.

[Song *et al.*, 2020] Ying Song, Shuangjia Zheng, Zhangming Niu, Zhang-Hua Fu, Yutong Lu, and Yuedong Yang. Communicative representation learning on attributed molecular graphs. In *IJCAI*, volume 2020, pages 2831–2838, 2020.

[Topping *et al.*, 2021] Jake Topping, Francesco Di Giovanni, Benjamin Paul Chamberlain, Xiaowen Dong, and Michael M Bronstein. Understanding over-squashing and bottlenecks on graphs via curvature. *arXiv preprint arXiv:2111.14522*, 2021.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[Wang *et al.*, 2020] Zhongruo Wang, Krishnakumar Balasubramanian, Shiqian Ma, and Meisam Razaviyayn. Zeroth-order algorithms for nonconvex minimax problems with improved complexities. *stat*, 1050:22, 2020.

[Wang *et al.*, 2024] Yili Wang, Kaixiong Zhou, Ninghao Liu, Ying Wang, and Xin Wang. Efficient sharpness-aware minimization for molecular graph transformer models. In *International Conference on Learning Representations*, 2024.

[Wu *et al.*, 2018] Zhenqin Wu, Bharath Ramsundar, Evan N Feinberg, Joseph Gomes, Caleb Geniesse, Aneesh S Pappu, Karl Leswing, and Vijay Pande. Moleculenet: a benchmark for molecular machine learning. *Chemical science*, 9(2):513–530, 2018.

[Wu *et al.*, 2022] Qitian Wu, Wentao Zhao, Zenan Li, David P Wipf, and Junchi Yan. Nodeformer: A scalable graph structure learning transformer for node classification. *Advances in Neural Information Processing Systems*, 35:27387–27401, 2022.

[Yang *et al.*, 2019] Kevin Yang, Kyle Swanson, Wengong Jin, Connor Coley, Hua Gao, Angel Guzman-Perez, Timothy Hopper, Brian P Kelley, Andrew Palmer, Volker Settels, et al. Are learned molecular representations ready for prime time? 2019.

[Ying *et al.*, 2021] Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do transformers really perform badly for graph representation? *Advances in neural information processing systems*, 34:28877–28888, 2021.

[Zhang *et al.*, 2024] Yihua Zhang, Pingzhi Li, Junyuan Hong, Jiaxiang Li, Yimeng Zhang, Wenqing Zheng, Pin-Yu Chen, Jason D Lee, Wotao Yin, Mingyi Hong, et al. Revisiting zeroth-order optimization for memory-efficient llm fine-tuning: A benchmark. *arXiv preprint arXiv:2402.11592*, 2024.

[Zhao *et al.*, 2022] Yang Zhao, Hao Zhang, and Xiuyuan Hu. Randomized sharpness-aware training for boosting computational efficiency in deep learning. *arXiv preprint arXiv:2203.09962*, 2022.