Problem 1

This paper explores zeroth-order (ZO) optimization techniques, which eliminate the need for back-propagation by approximating gradients through function value differences rather than explicit gradient calculations. As LLMs continue to grow in size, the memory overhead generated by back-propagation during gradient computation presents a significant obstacle for applications like on-device training, where memory constraints are critical. And this technique addresses the challenge of memory inefficiency in fine-tuning large language models (LLMs) due to the extensive memory requirements of first-order (FO) optimization methods, such as stochastic gradient descent (SGD) and Adam. Though scaling up ZO optimization for deep model training is exceedingly challenging due to its high variance. Nevertheless, LLM pre-training offers a unique advantage by enabling the fine-tuner to start from a well-optimized pre-trained model state. This graceful model initialization makes ZO optimization potentially scalable to LLM finetuning tasks.

A key contribution of the paper is the benchmarking of six ZO optimization methods, such as ZO-SGD and ZO-Adam, and comparing them with FO methods. The benchmark reveals the trade-offs between memory efficiency and fine-tuning accuracy, showing that methods like ZO-SGD can significantly reduce memory usage while maintaining competitive accuracy, making them well-suited for memory-constrained settings. Furthermore, the authors introduce novel improvements to ZO optimization, including block-wise descent, which reduces variance in gradient estimation, and hybrid training, combining ZO and FO methods to strike a balance between memory efficiency and performance. These innovations demonstrate that ZO optimization offers a promising solution for scaling LLM fine-tuning to resource-limited environments.

Strength

- **Comprehensive Benchmarking**: This paper provides a detailed benchmark of different ZO optimization methods across multiple LLM architectures, tasks, and fine-tuning strategies. This helps researchers understand the trade-offs between different methods and gives a clearer picture of how ZO optimization compares to traditional FO approaches, particularly in terms of memory usage and query efficiency.
- **Innovation in Extended Study**: The introduction of block-wise descent and hybrid ZO-FO fine-tuning represents an important innovation. The proposed enhancements can further improve the finetuning accuracy while maintaining the memory efficiency.

Weakness

- **High Variance in Performance**: One of the weaknesses of ZO optimization identified in the paper is the high variance in performance across different ZO methods. For example, ZO-SGD and ZO-Adam exhibit fluctuating accuracy depending on the model and task, which can make it difficult to predict performance consistently across different LLMs. This indicates that the stability of ZO methods still needs improvement.
- **Limited Scalability for Larger Models and Complex Tasks**: While ZO optimization methods offer clear advantages in terms of memory efficiency, the paper highlights that as model size increases and task complexity grows, FO methods still outperform ZO methods in terms of accuracy. This suggests that ZO optimization might struggle with scaling up to the largest LLMs and more complex tasks.

Future Directions

- **Reducing Variance in ZO Optimization**: Further exploration of variance reduction techniques, such as importance sampling introduced in the previous paper, could help stabilize gradient estimates in ZO optimization. These techniques can reduce the noise introduced by ZO's gradient approximation and make the method more robust across different tasks and LLM architectures.
- **Privacy-Preserving Optimization**: An interesting possible direction for future research is applying ZO optimization in privacy-preserving machine learning. Since ZO methods do not require access to gradients, they can be more privacy-preserving compared to FO methods.

<u>Problem 2</u>

This paper introduces an input-aware dynamic backdoor attack, where the trigger is dynamically generated based on the input image which aims to improve the stealthiness and effectiveness of backdoor attacks on deep neural networks. Traditional backdoor attacks rely on static, fixed triggers that remain the same for all input images, making them detectable and vulnerable to defense methods. While this method ensures that each input has a unique trigger, significantly enhancing the stealthiness of the attack and bypassing existing defense mechanisms that rely on detecting fixed triggers.

The trigger generator used in this method is trained using a diversity loss to ensure that different input images produce distinct triggers, and a cross-trigger test ensures that triggers cannot be reused across images. By doing so, the attack can effectively poison models while maintaining high accuracy on clean data. The attack was tested on standard datasets such as MNIST, CIFAR-10, and GTSRB, achieving near-perfect attack success rates while bypassing state-of-the-art defense mechanisms like Neural Cleanse, STRIP, and Fine-Pruning. Additionally, the method was shown to resist common image regularization defenses such as spatial smoothing, further demonstrating its robustness.

Strength

- **Stealthiness and Resistance to Defenses**: By generating unique triggers for each input, the attack introduced in the paper becomes much harder to detect and defend against, as it breaks the assumption that backdoor triggers are static. This makes existing defense methods that rely on detecting fixed triggers ineffective.
- **Generalization Across Datasets**: The method is shown to be effective across different datasets and model architectures, demonstrating that it can generalize well to a variety of tasks. The high attack success rate coupled with the ability to maintain performance on clean data indicates that the attack is both powerful and adaptable.

Weakness

- **Lack of Real-World Testing**: While the method performs well in controlled experimental settings and it is claimed to be possibly useful in more scenarios, it has not yet been thoroughly evaluated in more complex, real-world applications, such as speech or face recognition tasks. Further validation is necessary to confirm the generalization of the attack to these domains.
- **Complexity in Implementation**: The input-aware trigger generation method requires careful design and training, including the use of a trigger generator and enforcing a diversity loss to prevent triggers from collapsing into fixed patterns. This added complexity may limit the practical deployment of the attack in many real-world scenarios .

Future Directions

- **Adversarial Noise as Triggers**: To address the problem mentioned in the paper that the generated triggers can appear unnatural, one possible approach would be to blend adversarial noise techniques with input-aware triggers, making the triggers indistinguishable from typical adversarial perturbations. This could further blur the line between backdoor attacks and common adversarial examples, making detection even more difficult.

- **Improving the Efficiency and Scalability of the Attack**: The current trigger generator architecture may introduce computational overhead, especially when applied to large-scale datasets or real-time applications. A future research direction could focus on improving the efficiency and scalability of the attack.
- **Adapting to Fine-Tuning Scenarios**: In modern AI, model fine-tuning on pre-trained models is a common practice. Then a key question is whether the dynamically generated triggers remain effective after the model undergoes fine-tuning on new tasks or datasets. Research could explore ways to ensure that the triggers are resilient to changes in the model weights, preserving their effectiveness even after fine-tuning.