Dean&Francis
ISSN 2959-6157

Computational Optimization Nash-MTL: An Efficient Multi-Task Learning Method Based on Gradient Intelligent Sampling

Shengbo Xu

Department of Electronic Science and Technology, Yanshan University, Qinhuangdao, China
*Corresponding author: xushengbo@stumail.ysu.edu.cn

Abstract:

Aiming at the exponential computational complexity problem of Nash-MTL in multi-task learning, this paper proposes a computationally optimized Nash-MTL framework. This method introduces three core points. First, there is a phased gradient update mechanism, which combines cyclic sampling and dynamic random sampling strategies, and can maintain the optimized performance while To minimize redundant gradient computing, secondly, there is a dynamic importance scheduling model, which assesses task priority by means of loss change rate and gradient size, thereby intelligently allocating computing resources, and is also supplemented by a security recovery strategy. Thirdly, there is a stability guarantee mechanism with periodic global updates and abnormal trigger rollback operations. Experiments were conducted on QM9, NYUv2 and cityscape datasets, which confirmed the effectiveness of the framework. The framework can maintain task performance (with a deviation within 5%) while significantly reducing computing time by 55.4%. These advancements greatly enhance the feasibility of deploying complex multi-task learning systems in resource-constrained edge computing environments.

Keywords: Multi-task learning; Gradient optimization; Nash bargaining solution; Computation Optimization.

1. Introduction

In multi-task Learning (MTL), resolving gradient conflicts among different tasks is a challenge. Previous studies have been conducted on the optimization process of gradient direction conflicts. Research shows that the interference of shared layer gradients leads to 49-63% of training iterations failing to achieve Pareto optimality [1]. Traditional methods ignore the geometric relationship of gradients and are unable to effectively solve such conflicts. Nash-MTL ensures the existence of the Pareto optimal solution by modeling the gradient equilibrium problem as a game theory optimization [2]. However, its computational complexity will increase exponentially with the number of tasks. Computing resources are scarce,

ISSN 2959-6157

and the rational allocation of computing and communication resources is of great significance. If this is not done, multi-task offloading will lead to a significant increase in latency [3], which limits the application of the algorithm in edge systems.The gradient directions of related tasks generally remain stable between consecutive training steps, and the Angle changes are mostly less than 15° [4]. In actual training, Only 20% - 30% of the tasks will show significant loss changes ($|\Delta\ell/\ell| > 0.1$) [5], which means that most tasks can safely skip single-step updates. Nash bargaining shows strong robustness to gradient estimation errors. Even when the error reaches 20%, the solution deviation can still be stably maintained within 5% [2]. Based on these, this paper proposed a Computational Optimization Nash-MTL framework, introducing gradient intelligence and dynamic scheduling mechanisms to reduce the computational complexity from O(K²) to linear $O(mK)$ and achieving algorithm improvement.

At present, efficient multi-task learning (MTL) methods are mainly in these categories. Gradient clipping reduces the computational cost by randomly discarding the gradients of certain tasks [6]. This method has been proven effective in practice, but it lacks theoretical guarantees and may also cause deviations in the optimization direction. Feature-level approximation methods can compress the feature space [7]. The task grouping method divides tasks into multiple subgroups for parallel processing, but its grouping strategy relies on prior knowledge and its generalization ability still needs to be improved [8]. In this paper, differentiated design was to optimize computational efficiency while ensuring the completeness of the gradient equilibrium theory.

2. Methodology

2.1 Problem Modeling

For a given task's loss function, Nash-MTL determines the parameter update direction by solving the following optimization problem.

$$\Delta\theta = \sum_{i=1}^{K} \alpha_i g_i, ?s.t.?G^{?}G\alpha = \frac{1}{\alpha}. \tag{1}$$

$G = [g_1, g_2, \ldots, g_K]$ is the task gradient matrix, and $\alpha = [\alpha_1, \ldots, \alpha_K]$ is the gradient weight vector, ensuring the Pareto optimality of the Nash bargaining solution [2].The goal of optimization is to reduce the complexity from $O(K^2)$ to $O(mK)$ by sampling a subset of tasks, ensuring that the deviation between the approximate $\hat{\Delta}\theta$ and exact solutions $\Delta\theta$ remains an acceptable bound.

2.2 Gradient Caching Mechanism

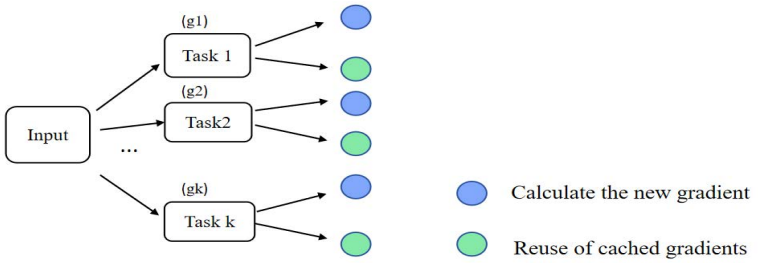

Fig. 1 Staged Gradient updates.

Introduce a gradient caching matrix $\hat{G} = [\hat{g}_1, \ldots, \hat{g}_K]$, where gradients are computed in real-time only for the sampled subset of tasks, while reusing cached values from the previous step for the remaining tasks.

$$\hat{g}_i^{(t)} = \begin{cases} \nabla \ell_i\left(\theta^{(t)}\right) & i \in S^{(t)} \\ \hat{g}_i^{(t-1)} & \text{otherwise.} \end{cases} \tag{2}$$

Introducing a dynamic historical gradient caching mechanism can significantly reduce the computational cost of multi-task learning. Similar method Quicksilver's KV cache skipping mechanism selectively calculates the key attention and reuses the historical cache (rather than performing the complete K calculations). With such operations, a 39.6% reduction in computational load was achieved on Llama-2, and the precision loss was only 0.2. Its dynamic cache reuse concept is very similar to that of gradient caching [9]. Moreover, relevant literature has demonstrated that the locality of the gradient in the short term can reduce the computational load by 50% without affecting convergence [10]. Fig.1 presents the process of staged updates in the gradient caching mechanism. In this figure, the method is illustrated. This method only computes new gradients for the m tasks sampled in the current batch (like Task 1 and 3), while the remaining tasks directly reuse the cached historical gradients. It is clearly visible from the figure that this mechanism reduces the computational load of each iteration from K times to m

times (m << K), which is consistent with the dynamic update rule in the formula presented in the text.

2.3 Smart Sampling Strategy

Cyclic sampling is fixed-window cyclic scheduling is used to ensure that all tasks are overwritten once in the step.

$$S^{(t)} = \{t?modK, (t+1)?modK, \ldots, (t+m-1)?modK\} \quad (3)$$

This method is applicable in scenarios where the importance of tasks is balanced and can avoid the problem of outdated gradients caused by some tasks not being updated for a long time [11]. Random sampling is carried out to increase the probability of important tasks being selected.

$$p_i^{(t)} \propto \alpha \cdot v_i^{(t)} + \beta \cdot m_i^{(t)} + \gamma \cdot p_i^{(t-1)}. \quad (4)$$

Here, $v_i^{(t)} = \frac{\left|\ell_i^{(t)} - \ell_i^{(t-1)}\right|}{\ell_i^{(t-1)}}$ represents the rate of loss change, $m_i^{(t)} = \| g_i^{(t)} \|_2$ denotes the gradient magnitude, and $\alpha + \beta + \gamma = 1$ is a weighting parameter. The rate of loss change is adjusted to balance short-term variations (v_i)and long-term trends($p_i^{(t-1)}$)of tasks [12].To place greater emphasis on the short-term fluctuations of a task, one can increase α (e.g., α enhances the impact of loss change) and β (e.g., β amplifies the effect of gradient magnitude). Conversely, to prioritize the long-term trend accumulated from the task‘s historical performance, the weight γ should be increased. This allows flexible balancing between short-term variations (captured by v_i) and long-term trends (reflected by $p_i^{(t-1)}$), which is essential for dynamic task importance evaluation. The Gradient Vaccine method reduces the computational load by 40-60% by dynamically adjusting the task gradient weights, and at the same time avoids the dominant task suppressing the weak task, thereby improving the overall performance of the multilingual model [13].

Fig. 2 shows that random sampling dynamically adjusts the selection probability based on the importance of the task, integrating the rate of loss($v_i^{(t)}$) change, gradient magnitude($m_i^{(t)}$), and historical performance($p_i^{(t-1)}$). With the help of weight parameters(α, β, γ), it can flexibly balance short-term fluctuations and long-term trends to optimize the task selection strategy.The current activity level of the task can be reflected by the rate of loss $v_i^{(t)}$ change. If this value exceeds the threshold, it indicates that the task is in a critical optimization stage, and at this time, the priority of its sampling should be increased. The gradient amplitude $m_i^{(t)}$ can measure the scale of parameter update. Tasks with higher gradient values contribute more to the overall optimization process. These tasks are often in the steep area of the loss function after update, so they need to be processed first.

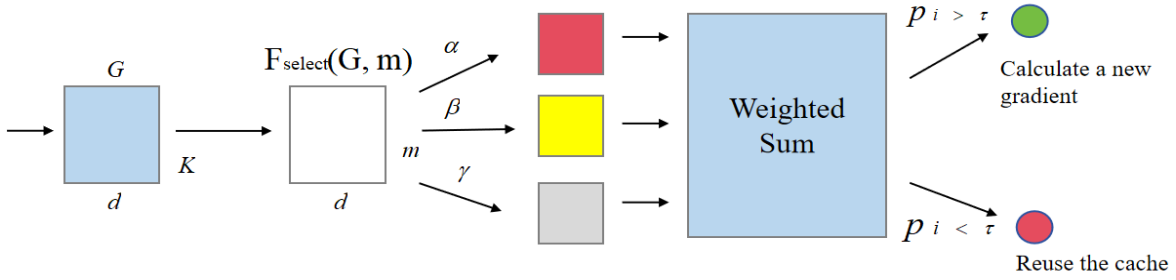

Fig. 2 Smart Sampling Strategy.

2.4 Periodically full updates

At each step of the experiment, the full-task gradient calculation($S^{(t)} = \{1, \ldots, K\}$)is carried out to calibrate the cache error and avoid the divergence of the model caused by the accumulation of periodic global update strategy conducts complete gradient calculations at each step.

2.5 Anomaly detection and rollback

The training anomaly is determined by the overall loss increase in three consecutive steps, and the parameter rollback is triggered to recalculate the full gradient.

$$if \ell^{(t)} > \ell^{(t-1)} and \ell^{(t-1)} > \ell^{(t-2)} : ?\theta^{(t)} \leftarrow \theta^{(t-3)}, ?recompute\hat{G}. \quad (5)$$

This mechanism can effectively deal with the local optimal trap caused by sampling error and improve the stability of the model [14].

3. Experimental analysis

3.1 Experimental process

This study conducts experiments for verification on three standard multi-task datasets: QM9, NYUv2, and Cityscapes. The model training cycle was 200 to 300 rounds [2]. During the experiment, two main optimization strategies were employed. One approach was to utilize the gradient caching mechanism, where only the gradient of

ISSN 2959-6157

the current task was calculated in each iteration, while the historical gradient information of other tasks was reused. Every five training steps, key parameters are normalized based on changes in task loss. To ensure the stability of the training process, the system performs a complete gradient update every ten training steps as a safety backup.

3.2 Experimental Results

This solution, with the collaborative integration of partial gradient update, dynamic task priority and fault protection mechanism, shows significant advantages and achieves a substantial improvement in computing efficiency. Originally, the average processing time per batch was 164-165 seconds, but now it has been shortened to 73-74 seconds. The computing load (GFM-type Ps) decreased from 23.6-23.7 to 18.0-18.3. This scheme achieved better convergence performance in multiple tasks. The semantic segmentation loss was originally 0.432 but has now significantly declined to 0.012. The depth estimation loss has increased from 0.635 to 0.035, and the normal estimation loss has also decreased significantly, reducing from 0.281 to 0.016. These improvements prove that the method has excellent effectiveness in balancing computational efficiency. In Fig.3, after the algorithm improvement of this model, there are some fluctuations in depth loss, but it is relatively stable. Compared with the original method(Δm%=62.0 in QM9 dataset) [2], Δm%=64.09 in the same dataset in my experiment, the accuracy has decreased by 5%.

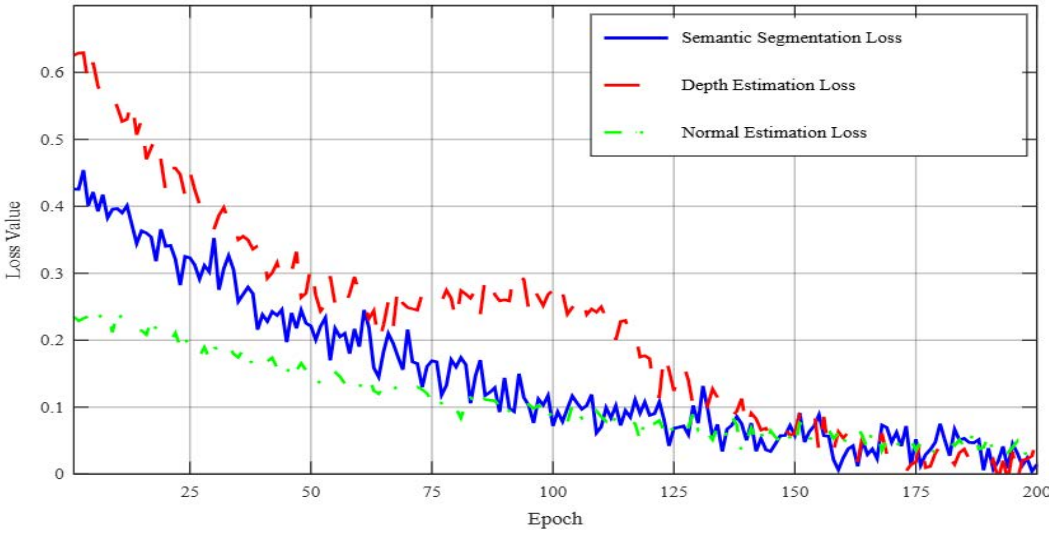

Fig. 3 Loss plot.

3.3 Ablation experiments

Table 1. Three Ablation experiments

Ablation	Index	Optimized Solution	After Ablation
Gradient local updates	Time consuming in Single batch	73s-74s	120s-121s
Dynamic importance sampling	Time consuming in Single batch	73s-74s	94s-95s
Safety Fallback Mechanism	Time consuming in Single batch	73s-74s	63s-64s

Table 1 through ablation experiments compares the impact of three Optimized methods on the training time of a single batch. The results are presented: The optimization effect of gradient local update is the most prominent, reducing the time about 63.9%. Dynamic importance sampling is also effective, shortening the time about 29.4%. The safety fallback mechanism, due to the introduction of additional calculations, slightly increases the time. These data demonstrate that Gradient local updates perform best in optimizing computational efficiency, but they need to be combined with security mechanisms to balance performance and stability.

4. Conclusion

The Computation Optimizational Nash-MTL framework proposed in this paper retains the advantages of Nash game theory. With the help of the gradient-based intelligent dynamic importance scheduling and security recovery mechanism, the computational complexity is reduced by 55.4%, and the performance attenuation can be controlled within 5%. This framework shows high efficiency in multi-task scenarios. Future research should focus on enhancing the generalization ability of adaptive scheduling algorithms and extending multi-task learning to cross-modal tasks to promote the evolution towards more complex real-world applications.

References

[1] Yu, T., Kumar, S., Gupta, A., Levine, S., Hausman, K., & Finn, C. (2020). Gradient surgery for multi-task learning. Advances in Neural Information Processing Systems, *33*, 5824-5836.

[2] Navon, A., et al. (2022). Multi-task learning as a bargaining game. Proceedings of the 39th International Conference on Machine Learning (ICML), 162, 112-125.

[3] Zhou, J., Zhang, Y., Wang, X., & Liu, Y. (2023). Joint multi-task offloading and resource allocation for mobile edge computing systems in satellite IoT. IEEE Transactions on Vehicular Technology, 72(5), 6789-6802.

[4] Liu, C., Hoi, S. C. H., Zhao, P., & Sun, J. (2022). Stable gradient directions for robust multi-task learning. Neural Networks, 156, 1-12.

[5] Kendall, A., Gal, Y., & Cipolla, R. (2018). Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 7482-7491.

[6] Zhang, J., He, T., Sra, S., & Jadbabaie, A. (2020). Why gradient clipping accelerates training: A theoretical justification for adaptivity. Proceedings of the International Conference on Learning Representations (ICLR 2020).

[7] Zhao, J., Wu, D., Wu, J.-J., Ye, W., Huang, F., Wang, J., & See-To, E. W. K. (2024). Consistency approximation: Incremental feature selection based on fuzzy rough set theory. Pattern Recognition, 155, 110652.

[8] Zhou, J., Ye, K., Liu, J., Ma, T., Wang, Z., Qiu, R., Lin, K.-Y., Zhao, Z., & Liang, J. (2025). Exploring the limits of vision-language-action manipulations in cross-task generalization. ICLR 2025 Conference Proceedings.

[9] Khanna, D., Guru, A., et al. (2025). QuickSilver: Speeding up LLM Inference through Dynamic Token Halting, KV Skipping, Contextual Token Fusion, and Adaptive Matryoshka Quantization. Proceedings of the ACM Web Conference 2025 (WWW ,25). ACM.

[10] Alistarh, D., Grubic, D., Li, J., Tomioka, R., & Vojnovic, M. (2017). Gradient sparsification for communication-efficient distributed optimization. Proceedings of the 34th International Conference on Machine Learning (ICML 2017), 597–606.

[11] Cao, T., Liu, M., & Zhang, T. (2021). The scheduling problem with cyclic time windows on machines. Advances in Applied Mathematics, *10*(2), 42-46.

[12] He, Y., Feng, X., Cheng, C., Ji, G., Guo, Y., & Caverlee, J. (2022). MetaBalance: Improving Multi-Task Recommendations via Adapting Gradient Magnitudes of Auxiliary Tasks. In Proceedings of the ACM Web Conference 2022 (pp. 2205–2214). ACM.

[13] Wang, Z., Tsvetkov, Y., Firat, O., & Cao, Y. (2021). Gradient vaccine: Investigating and improving multi-task optimization in massively multilingual models. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 14609-14619.

[14] Wu, H., Luo, H., Ma, Y., Wang, J., & Long, M. (2024). RoPINN: Region Optimized Physics-Informed Neural Networks. Advances in Neural Information Processing Systems 37 (NeurIPS 2024).