Performance Comparison of Reversible Vision Transformers with Integrated Momentum

Haolun Yang
Department of Computer Science
University of Exeter, Exeter, UK

Internal Supervision:
Guoqiang Zhang
Department of Computer Science

***Abstract*—This proposal investigates various reversible neural network architectures, analyzing their characteristics, strengths, and weaknesses to devise solutions that optimize memory usage and enhance performance. The research aims to explore the integration of momentum into Reversible Vision Transformers (Rev-ViTs) to improve learning efficiency, accuracy, computational performance, and memory optimization. By comparing the performance of momentum-enhanced Rev-ViTs with traditional models across different benchmarks, this study seeks to optimization that can enhance the efficacy and applicability of reversible networks in deep learning.**

I. Introduction

Deep learning has dramatically reshaped approaches in computer vision and natural language processing through advancements in artificial neural networks. Residual Neural Networks (ResNets) have enhanced deep learning by enabling significantly deeper networks using 'skip connections'. [1] Vision Transformers (ViTs), using self-attention mechanisms, process images to accentuate global feature interplays, improving complex representation learning. [2] However, the increased complexity of these models raises training memory consumption issues which poses a significant pressure for large-scale applications.

Reversible Vision Transformers (Rev-ViTs) address this by allowing input recovery from outputs without storing intermediate activations, introducing a significant shift towards memory efficiency. Integrating momentum into this framework could enhance reversible network performance and inspire future research. Initial benchmarks suggest that this integration maintains reversibility while potentially speeding up convergence and reducing memory usage.

The project proposal will concentrate on examining the performance comparison of Reversible Vision Transformers with integrated momentum, detailing Relevant Related Work, Project Aims, Network Design, Evaluation Plan, and Project Management.

II. Related work

In the field of deep learning, enhancing computational efficiency and performance is a key driver of technological progress. How to further improve the learning efficiency of these reversible models? If successfully introducing the concept of momentum into Reversible Vision Transformers (Rev-ViTs), this will further enhance the training efficiency of the model while maintaining the memory-saving benefits of the reversible architecture. This section will focus on analyzing the reversible structure of related models and their advantages and disadvantages.

As the ResNets become deeper and wider, coupled with the need to store the activation values of each layer during backpropagation, high the memory consumption becomes a problem. To address this issue, the reversible residual networks, RevNets, as a solution illustrated by A. N. Gomez and their team, allows for the reconstruction of the n^{th} layer using the activations of the $(n+1)^{th}$ layers. This means that during backpropagation, the activations of most layers, except for a few, do not need to be stored. The authors first divide the input of each layer into two groups, (x_1, x_2), with the outputs being (y_1, y_2), using two residual functions F and G. In the forward propagation, F acts on x_2 and is added to x_1 to get y_1, then G acts on y_1 and is added to x_2 to get y_2. Thus, x_1 and x_2 can be reconstructed from y_1 and y_2 during backpropagation, using the method $x_2 = y_2 - F(x_2)$, $x_1 = y_1 - F(x_2)$. It's important to note that, unlike traditional ResNets which can have layers with larger strides, the stride of the reversible blocks in RevNets must be 1, otherwise information of intermediate layers will be lost. To execute backpropagation without storing activations, the authors have proposed a modified method of computation. In backpropagation, the activations y_1, y_2 and their derivatives $\bar{y}_1, \bar{y}_2$ are used to calculate the inputs x_1, x_2 and their derivatives $\bar{x}_1, \bar{x}_2$, as well as the total derivatives of any parameters associated with the functions F and G, thus eliminating the need to store the activations for the layers. [3]

Overall, while the RevNets model resulted in longer training times and slightly reduced computational efficiency, the advantages in memory savings were quite significant. However, this comes with an increase in computational load, with RevNets requiring 4N operations compared to the traditional neural network's N plus 2N addition and multiplication operations.

The Momentum Residual Neural Networks are a variant of traditional Residual Neural Networks. As M. E. Sander's team shows, Momentum Residual Networks take this a step further by incorporating the concept of momentum, defining the function $f(x_n, \theta_n)$ in residual blocks as $v_{n+1} = \gamma v_n + (1-\gamma) f(x_n, \theta_n)$ and $x_{n+1} = x_n + v_{n+1}$, where v is the velocity term, γ is the momentum term, and f is the residual mapping function. With $x_n = x_{n+1} - v_{n+1}$ and $v_n = \frac{1}{\gamma}(v_{n+1} - (1-\gamma) f(x_n, \theta_n))$, the activations of layer n can be fully recovered from layer $n+1$, achieving network

reversibility. Since the activations of all layers during forward propagation can be recreated during backward propagation, there is no need to store these activations separately, which leads to less memory usage. By adjusting the value of the momentum parameter γ within the range of $[0, 1]$, this network structure can transition from approximating a traditional residual network to a symplectic scheme model similar to RevNet and Hamiltonian Networks. Momentum Residual Networks can also be interpreted as second-order ordinary differential equations in the continuous-time limit, allows them to learn more complex dynamics more finely. When fine-tuning a ResNet-152 network with a resolution of 500*500 on a GPU with 3GB RAM, Momentum ResNets are more convenient for fine-tuning compared to RevNet, which requires two separate networks. When learning to optimize sparse coding problems under the LISTA framework, Momentum Residual Networks can converge faster and handle larger batch sizes. This means faster training speeds and more efficient training of larger datasets. [4]

Momentum Residual Networks also may have some disadvantages. The introduction of the momentum term suddenly increases the complexity of the network, making it more difficult to understand and construct. Determining the value of the momentum term γ is also troublesome, requiring a large amount of experimentation and troubleshooting. Although the cost of memory is reduced, the corresponding computational cost of computing backward propagation increases, especially as the network depth increases, as the backward propagation process requires additional computations to restore the activations of previous layers.

Vision Transformers decouples model depth from GPU memory requirements, allowing the structure to scale effectively while utilizing memory efficiently. K. Mangalam's team adapted Vision Transformer and Multiscale Vision Transformers into reversible variants, which are called Reversible Vision Transformers, can significantly reduce memory usage by up to 15.5 times while maintaining similar model complexity and accuracy, and optimize additional computational burdens on deeper levels, making processing speeds up to 2.3 times faster than non-reversible models.The reversible transformation splits an input tensor I into two d-dimensional tensors and obtains tensor O through transformation T_1 as follows:

$$I = \begin{bmatrix} I_1 \\ I_2 \end{bmatrix}, \quad O = \begin{bmatrix} O_1 \\ O_2 \end{bmatrix}$$

where $O_1 = I_1$ and $O_2 = F(I_1)$ with F being an arbitrary differentiable function. Similarly, for the transposed transformation T_2 applied to I:

$$O = \begin{bmatrix} O_1 \\ O_2 \end{bmatrix} = \begin{bmatrix} I_1 + G(I_2) \\ I_2 \end{bmatrix}$$

Combining T_1 and T_2 into a composite transformation T, the result under transformation T can be represented as $\begin{bmatrix} I_1 + G(I_2 + F(I_1)) \\ I_2 + F(I_1) \end{bmatrix}$. Since both T_1 and T_2 allow for inverse transformations, the composite transformation T is also reversible, allowing the original input I to be recovered from the output O. Based on this, the team adapted Vision Transformers into a form with two residual streams, exchanging information through functions F and G to achieve reversibility. Function F represents the processing, extraction, and integration of information from the input using Multi-head attention, and function G represents the use of MLP to increase the nonlinearity and depth of the network, both of which are core operations within the reversible block. The authors also propose Reversible Multiscale Vision Transformer (Rev-MViT), including Stage-Transition Block and Stage-Preserving Block to adapt the reversible structure to Transformers with varying feature dimensions. The Stage-Transition Block allows for dimension transitions at the end of a constant dimension stack, using a fusion module to exchange information between the two input streams and the Stage-Preserving Block is used to maintain constant feature dimensions and forms the main part of reversible computation, which enables Rev-MViT to execute forward and backward propagation without storing the activation values of intermediate layers, thereby reducing the memory requirements for model training and inference. The authors trained and tested the model on image and video using Single Stage Transformers and Hierarchical Transformers at three different model sizes, demonstrating the wide adaptability of the model. [5]

All these design contribute to lower the usage of GPU memory, but reversible structure may make training convergence unstable as the model deepens.

B. Chang's team primarily analyzes three types of arbitrarily deep reversible structures. The Hamiltonian neural network structure is a key focus among them. The network's state is divided into two parts, Y and Z, used to simulate the position and momentum in physics. The two-layer structure Hamiltonian introduce two sets of convolution kernels, K_1 and K_2. $K(t)$ and $K(t)^T$ represent convolution and transpose convolution operations, respectively, which are mathematically reversible. Extending this to K_1 and K_2, the discretized Hamiltonian, as also recognised as Verlet method equations can be written as:

$$Y_{j+1} = Y_j + hK_j^T \sigma(K_{1j} Z_j + b_1), \tag{1}$$

$$Z_{j+1} = Z_j - hK_j^T \sigma(K_{2j} Y_{j+1} + b_2). \tag{2}$$

Moreover, since K_j and K_j^T are inverse operations of each other, it is straightforward to reverse solve for Y_j and Z_j from Y_{j+1} and Z_{j+1}, thereby achieving the reversibility of the network, while it is crucial to select appropriate network parameters to prevent the introduction of numerical instabilities. [6] However, although in theory a single-layer Hamiltonian network can approximate monotonically increasing continuous functions, the single-layer structure has limitations in capturing complex data patterns. In practice, certain controls are necessary to ensure that the network does not experience exponential growth during forward and backward propagation.

A. Gholami's team shows a ANODE framework employing the adjoint method, aims to reduce memory consumption and

stabilize the numerical solution process:

$$-\frac{d\alpha(t)}{dt} - \left(\frac{\partial f}{\partial z}\right)^T \alpha = 0, \quad t \in [0,1] \tag{3}$$

Here, $\alpha(t)$ represents the adjoint variable, crucial for gradient computations. To calculate the gradients efficiently, without storing all intermediate activations, They use:

$$g_\theta = \frac{\partial R}{\partial \theta} - \int_0^1 \left(\frac{\partial f}{\partial \theta}\right)^T \alpha \, dt \tag{4}$$

The ANODE framework mitigates memory demands in training Neural ODEs by compressing the memory requirement to $O(L)+O(N_t)$, employing checkpointing and Discretize-Then-Optimize (DTO) techniques for accurate and memory-efficient gradient calculations. [7]

However, under extreme memory constraints, one must start from the initial state and perform forward time steps up to the required point if a previously unsaved intermediate state is needed. This approach significantly increases computational costs since it leads to a quadratic number of recalculations in the number of time steps.

III. Aims & Objectives

The primary goal of this research is to investigate the impact of integrating momentum structures into Reversible Vision Transformers (Rev-ViTs) on model performance and optimization techniques.

Specifically, the research will address the following questions:

1) Can momentum improve convergence, learning efficiency, and accuracy in Rev-ViT models?
2) How does the incorporation of momentum affect memory efficiency, memory usage, and computational performance in RVTs?
3) Compared to conventional Rev-ViTs, how does the model with integrated momentum perform in various benchmark tests?

To explore the answers to these questions, the research objectives have been set as follows:

1) Modify the Rev-ViT model based on existing literature, integrate the momentum component, and test its specific impact on model performance.
2) Run both the traditional Rev-ViT and momentum-enhanced Rev-ViT on standard datasets to analyze and evaluate performance.
3) Ascertain the value of introducing momentum into the practical application of Rev-ViTs and explore directions for further optimization of the model structure.

This project will focus on the theoretical analysis of the model, architectural design, experimental testing, and result evaluation. It is not expected to include the implementation of momentum-enhanced Rev-ViTs in real-world application scenarios, which may be a part of subsequent research in image recognition and video processing.

IV. Design and Implementation

The overarching goal is to adjust the existing architecture of Rev-Vits to incorporate momentum terms. Key layers in Rev-Vits will be modified to maintain reversibility while using momentum to accelerate the training process and reduce memory usage. The following steps outline the implementation process:

1) Conduct a further analysis of the mathematical principles behind the momentum and Rev-Vits structures to understand how combine to improve reversible learning;
2) Based on the theoretical analysis of momentum mechanisms, design a momentum structure suitable for Rev-Vits;
3) Construct the model of momentum-augmented Rev-Vits within the PyTorch framework;
4) Train and test the new model on standard datasets to collect performance data for comparison.

Coding Environment Preparation: Development will take place within the PyTorch environment, managing all necessary dependencies via a virtual environment to ensure the reproducibility of the experiments.

Model Architecture: A foundational model of Rev-Vits will be established as the starting point for modifications. A momentum buffer layer is planned to be introduced into Rev-Vits in the form of a custom PyTorch layer. This layer will capture and update activation values in each transformer module, which will be adjusted according to momentum rules to optimize the flow of information.

Algorithmic Procedure: Updates to momentum will follow established optimization theories, with key formulas and algorithms being written into independent functions to ensure clarity and modularity. This process will affect the model's forward and backward propagation algorithms, requiring careful adjustments within PyTorch's automatic differentiation framework.

Training Process: A training pipeline is planned, featuring functionalities such as periodic model checkpointing, learning rate monitoring, and optimizer parameter tuning.

Debugging and Validation: Rigorous code reviews and unit testing will be carried out to verify each component of the model. The original Rev-Vits model will undergo unit tests to validate the stability and reversibility of the base architecture. Following each iteration of momentum structure addition, unit and functional tests will be re-executed to ensure that the new momentum structure does not compromise the model's fundamental performance and reversibility.

V. Evaluation Plan

In this section, I will elaborate on how I plan to assess the efficacy of the Reversible Vision Transformers with integrated momentum.

Experimental Design and Expected Goals: As previously mentioned, I will set up control experiments and experimental groups to compare traditional Rev-ViTs with those integrated with momentum mechanisms. Anticipate that this model will

demonstrate a marked improvement in learning efficiency and memory usage.

Specific Performance Metrics:

- **Learning Rate:** The rate of learning on the training set.
- **Accuracy:** The degree of prediction accuracy on the validation and test sets.
- **Convergence Speed:** The rate at which the model achieves accuracy across different training epochs.
- **Memory Efficiency:** Average and peak memory usage during training and testing.

Datasets: The datasets planned for use include CIFAR-10/100 and ImageNet, with necessary data preprocessing applied.

Hardware and Software Resources:

- A RTX 4060 Laptop GPU with 8GB VRAM.
- The latest stable release of NVIDIA CUDA 12.3 to fully utilize GPU acceleration features.
- A virtual environment running Python 3.8 under Windows 11 Anaconda, with PyTorch version 2.2.0 installed.

Assessment Procedures: The evaluation process will employ a range of tools including, but not limited to, PyTorch's built-in testing utilities and custom scripts for collecting and recording performance data. Visualization tools such as TensorBoard will be used to monitor training processes and evaluate results. Differences in performance between models will be analyzed, and the potential reasons for performance improvements brought about by the integration of momentum mechanisms into Rev-ViTs will be discussed.

VI. Project Management

In order to successfully complete this research project on schedule, the following is a list of milestones and deliverables with deadlines:

1. Theoretical Research and Proposal Writing (April 1 - April 30)
Objective: To complete preliminary theoretical research, review related literature, design the project plan, and draft the full proposal.
Deliverable: Submission of the project proposal document.
Deadline: May 3

2. Prototype Testing Phase (May 1 - May 14)
Objective: To test the foundational models of Reversible Vision Transformers and Momentum Residual Neural Networks, verifying basic functionality and performance.
Deliverable: Test report, including performance evaluation and operational confirmation.

3. Development Phase (May 15 - June 25)
Objective: To develop an integrated momentum mechanism in Reversible Vision Transformers, perform coding, and conduct preliminary functional tests.
Deliverable: Functional prototype and initial test results.

4. Debugging and Comparison Phase (June 26 - July 23)
Objective: To deeply debug the developed models, optimize performance and accuracy, and compare the new and original models.
Deliverable: Debugged model and detailed performance comparison analysis.

5. Report Writing and Continuous Updates (May 1 - August 8), Presentation Preparation (July 24 - August 8)
Objective: To continuously write and update the final project report while preparing the presentation starting July 24.
Deliverable: Complete project report and prepared presentation materials.
Deadline: August 8 for presentation preparation completion; August 9 for submission of the project presentation.

6. Final Report Revision and Improvement (August 9 - August 14)
Objective: To make final amendments and improvements to the report
Deliverable: Submission of the final project report and complete code.
Deadline: Submission of the final project report by August 15.

Technical Implementation Risk: Integration of momentum mechanisms into Reversible Vision Transformers may encounter unexpected technical compatibility and performance tuning issues.

Mitigation Strategy: Prompt collaboration with advisors will be pursued to ensure technical solutions are feasible and conceptually valid. Research targets may be slightly adjusted if necessary.

Project Schedule Risk: Technical difficulties or improper resource allocation may impede project progress.

Mitigation Strategy: A flexible time management plan will be established, with regular reviews of project progress and adjustments to resources and timelines as needed.

This project does not involve human or animal subjects and exclusively utilizes publicly available datasets, which minimizes ethical concerns. All research activities will adhere strictly to scientific ethical standards.

The datasets used are open to the public; however, the project will rigorously follow data protection laws to ensure all necessary precautions are taken in the storage and handling of data against any ethical breaches.

VII. Conclusion

This proposal outlines a plan to integrate momentum mechanisms into Reversible Vision Transformers (Rev-ViTs) with the aim of improving learning efficiency and memory optimization. Through theoretical analysis of existing models, I plan to modify Rev-ViTs by incorporating momentum terms and then conducting performance comparisons using standard datasets. The research is expected to optimize existing Rev-ViTs, aims to find ways to enhance memory efficiency and training performance of reversible deep neural networks in large-scale applications and provide new directions for the practice of deep learning and further research into reversible network models. I will strictly follow the research objectives and evaluation plan outlined in the proposal, and I anticipate that this project will contribute significantly to the field of deep learning.

REFERENCES

[1] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[2] K. Han, Y. Wang, H. Chen, X. Chen, J. Guo, Z. Liu, Y. Tang, A. Xiao, C. Xu, Y. Xu, Z. Yang, Y. Zhang, and D. Tao, "A survey on vision transformer," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 1, pp. 87–110, 2023.

[3] A. N. Gomez, M. Ren, R. Urtasun, and R. B. Grosse, "The reversible residual network: Backpropagation without storing activations," *Advances in neural information processing systems*, vol. 30, 2017.

[4] M. E. Sander, P. Ablin, M. Blondel, and G. Peyré, "Momentum residual neural networks," in *International Conference on Machine Learning*. PMLR, 2021, pp. 9276–9287.

[5] K. Mangalam, H. Fan, Y. Li, C.-Y. Wu, B. Xiong, C. Feichtenhofer, and J. Malik, "Reversible vision transformers," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 10 830–10 840.

[6] B. Chang, L. Meng, E. Haber, L. Ruthotto, D. Begert, and E. Holtham, "Reversible architectures for arbitrarily deep residual neural networks," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, no. 1, 2018.

[7] A. Gholami, K. Keutzer, and G. Biros, "Anode: Unconditionally accurate memory-efficient gradients for neural odes," *arXiv preprint arXiv:1902.10298*, 2019.