РОЗДІЛ ІІ. ІНЖЕНЕРІЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

UDC 004.93
DOI https://doi.org/10.26661/2786-6254-2023-2-03

INCORPORATING ATTENTION SCORE TO IMPROVE FORESIGHT PRUNING ON TRANSFORMER MODELS

Melnychenko A. V.
Postgraduate Student at the Department of Software Engineering for Power Industry
National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute"
Beresteyskyi Ave., 37, Kyiv, Ukraine
orcid.org/0009-0000-3588-4772
artemxl@gmail.com

Zdor K. A.
Postgraduate Student at the Department of Software Engineering for Power Industry
National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute"
Beresteyskyi Ave., 37, Kyiv, Ukraine
orcid.org/0009-0008-7640-1499
kostya9919moonlight@gmail.com

Key words: *transformers, neural networks, pruning, efficiency, optimization.*

With rapid development of technologies and growing number of application of neural networks, the problem of optimization arises. Among other methods to optimize training and inference time, neural network pruning has attracted attention in recent years. The main goal of pruning is to reduce the computational complexity of neural network models while retaining performance metrics on desired level. Among the various approaches to pruning, Single-shot Network Pruning (SNIP) methods was designed as a straightforward and effective approach to optimize number of parameters before training. However, as neural network architectures have evolved, particularly with the growing popularity of transformers, a need to reevaluate traditional pruning methods arises. This paper aims to revisit SNIP pruning method, evaluate its performance on transformer model, and introduce an enhanced version of SNIP, specifically designed for transformer architectures.
The paper outlines the mathematical framework of SNIP algorithm, and proposes a modification, based on specifics of transformers models. Transformer models achieved impressive results because of their attention mechanisms for a multitude of tasks such as language modeling, translation, computer vision tasks and many others. The proposed modification takes into account this unique feature and combines this information with traditional loss gradients. Traditional method calculates importance score for weights of the network using only gradients from loss function, in the case of enhanced algorithm. In the enhanced version, the importance score is a composite metric that incorporates not only the gradient from the loss function but also from the attention activations.
To evaluate the efficiency of proposed modifications, a series of experiments were conducted on image classification task, using Linformer variation of transformer architectures. The results of experiments demonstrate the efficiency of incorporating attention scores in pruning. Conducted experiments show that model pruned by modified algorithm outperforms model pruned by original SNIP by 34% in validation accuracy, confirming the validity of the improvements introduced.

ПОКРАЩЕННЯ ПРУНІНГУ ПЕРЕД НАВЧАННЯМ ШЛЯХОМ ВРАХУВАННЯ ПОКАЗНИКА УВАГИ ДЛЯ МОДЕЛЕЙ АРХІТЕКТУРИ TRANSFORMER

Мельниченко А. В.
аспірант кафедри інженерії програмного забезпечення в енергетиці
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
просп. Берестейський, 37, Київ, Україна
orcid.org/0009-0000-3588-4772
artemxl@gmail.com

Здор К. А.
аспірант кафедри інженерії програмного забезпечення в енергетиці
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
просп. Берестейський, 37, Київ, Україна
orcid.org/0009-0008-7640-1499
kostya9919moonlight@gmail.com

Ключові слова: *transformers, нейронні мережі, прунінг, ефективність, оптимізація.*

Зі стрімким розвитком технологій і зростанням числа прикладних застосувань нейронних мереж постає проблема їх оптимізації. Серед інших методів оптимізації навчання та використання навчених моделей в останні роки багато уваги було приділено методам виключення вагів (прунінг) нейронної мережі. Основна мета прунінгу — зменшити обчислювальну складність моделей за умови збереження показників продуктивності на прийнятному рівні. Серед різноманітних підходів до прунінгу, було розроблено метод одноразового прунінгу (SNIP), що являє собою простий і ефективний підхід для оптимізації параметрів перед навчанням. Однак з появою нових архітектур нейронних мереж, особливо зі зростанням популярності архітектур типу transformer, виникає потреба переглянути підхід до методів прунінгу. Дана стаття має на меті переглянути метод SNIP, оцінити його ефективність на моделі transformer та представити покращену версію SNIP, спеціально допрацьовану для архітектур tranformer.

У статті викладено математичну основу алгоритму SNIP та запропоновано його модифікацію, виходячи зі специфіки моделей transformer. Моделі архітектури transformer досягли значних результатів завдяки своєму механізму уваги для багатьох завдань, таких як розробка мовних моделей, переклад, задачі комп'ютерного зору та багато інших. Запропонована модифікація враховує цю унікальну особливість і поєднує цю інформацію при обчисленні градієнту і функції втрат. Традиційний метод розраховує оцінку важливості для ваг мережі, використовуючи лише градієнти функції втрат. У розширеній версії оцінка важливості є складеним показником, який включає не лише градієнт функції втрати, але й активацію уваги.

Для оцінки ефективності запропонованої модифікації було проведено серію експериментів на завдані класифікації зображень, використовуючи варіацію архітектури transformer – Linformer. Результати експериментів демонструють ефективність врахування показників уваги при прунінгу. Проведені експерименти показують, що модель, оптимізована за модифікованим алгоритмом, має показник точності на 34% кращий, ніж модель, оптимізована за оригінальним методом SNIP, підтверджуючи достовірність внесених вдосконалень.

Introduction

Recent developments in the field of deep learning have revolutionized the use of technologies in many fields. Among other developments, introduction of transformer models, initially designed for natural language processing (NLP) tasks, made possible achieving state-of-the-art performance in other domains. Transformers have been successfully in machine translation [1], text summarization [2], and question-answering systems [3].

Beyond NLP, the versatility of transformer models has been demonstrated in other fields. With introduction of Vision Transformer (ViT), training neural network with this architecture allowed to improve efficiency in computer vision tasks such as image classification [4]. Transformers have also been applied in bioinformatics for protein folding prediction [5] and in reinforcement learning for optimizing control systems [6].

Despite their effectiveness, transformer models come with high computational and memory requirements, even compared to more traditional architectures. The self-attention mechanism usually comes at a quadratic cost, and many transformer architectures have high-dimensional embeddings and have many stacked layers. This makes training and deploying transformers challenging, particularly in resource-constrained environments like edge devices or embedded systems. While not specifically designed for transformers, pruning techniques aim to reduce the complexity of neural network architectures in general, making them more efficient and deployable without significantly compromising performance.

Existing Pruning Methods

In the field of neural network optimization, various pruning methodologies have been developed to address computational complexity and reduce memory requirements [7; 8].

Among weight pruning methods, magnitude-based pruning operates by selecting weights for elimination based on their absolute values. While straightforward, this technique may induce performance degradation, so resulting model requires fine-tuning. In contrast, gradient-based pruning leverages the gradients of the loss function to identify importance of weights for elimination, but at a computational cost that may extend convergence time [9].

Neuronal pruning approaches, such as activation-based and objective-based methods, offer another dimension to pruning [10]. Activation-based pruning eliminates neurons that exhibit consistently low activation values across a dataset and enhances model interpretability. However, the method risks neglecting neurons that are conditionally crucial [11]. Objective-based pruning focuses on detecting neurons that contribute less than others to the overall loss function and pruning them. This approach is computationally demanding and requires carefull hyperparameter tuning.

Structured pruning methods, including filter and block pruning, offer a more architectural focus [12]. Filter pruning removes entire filters in convolutional layers based on criteria like their L1-norm. Despite its effectiveness in reducing both parameter count and computational complexity, it may necessitate architectural adjustments [13]. Block pruning prunes contiguous blocks of weights or neurons, preserving the architecture but potentially removing salient features [14].

Foresight pruning methods such as SNIP and the Lottery Ticket Hypothesis introduce unique paradigms [15; 16]. SNIP calculates a saliency score for each weight based on its initial contribution to the loss function and prunes the least salient ones before training commences. However, this method may require dataset-specific adjustments. Recent work has emphasized the utility of SNIP for pre-training, marking it as a notable avenue for computational optimization in neural networks.

Purpose and objectives of the study

Despite the advancements in pruning methods, there is room for improvement, especially in the context of transformer models. This study aims to enhance the efficiency of transformer models by modifying the SNIP algorithm to incorporate attention scores into the weight scoring mechanism. Our contributions are as follows:

– A novel modification to the SNIP algorithm that integrates attention scores.

– An empirical evaluation of the modified algorithm's performance on Plant Disease Dataset.

– A comparative analysis with original pruning method to demonstrate the efficacy of our approach.

Methodology

The SNIP algorithm is designed to prune neural networks before training commences. It calculates a saliency score for each weight in the network, based on the impact of removing that weight on the loss function. Weights with lower scores are pruned, resulting in a sparse network that can be trained more efficiently.

Mathematical Framework

The goal of SNIP is to find a sparse subset of weights W_s where $W_s \subseteq W$ such that the loss $L(W_s)$ is minimized. Here, W represents the weights in the entire neural network, and $L(W)$ is the loss function that the network aims to minimize during training. Main steps of the algorithm are:

1. The neural network is initialized with a set of random weights, denoted as W.

2. A single forward and backward pass is performed on a mini-batch of the training data. This step is crucial for computing the gradients $\frac{\partial L}{\partial W}$ of the loss function with respect to each weight in W.

3. The importance s_i of each weight w_i in W is calculated using the formula:

$$s_i = | w_i \times \frac{\partial L}{\partial w_i} | \quad (1)$$

Where $\frac{\partial L}{\partial w_i}$ is the gradient of the loss function with respect to the weight w_i.

1. The calculated importance scores s_i are normalized to produce $\hat{s}_i$ using the equation:

$$\hat{s}_i = \frac{s_i}{\sum_j s_j} \quad (2)$$

2. The normalized importance scores $\hat{s}_i$ are sorted in ascending order. A fraction ρ of the weights with the lowest normalized importance scores are pruned from W, resulting in the pruned set $\mathcal{W}_s$:

$$\mathcal{W}_s = \{w_i \in \mathcal{W} : \hat{s}_i \geq \rho\} \quad (3)$$

6. The pruned network, now represented by $\mathcal{W}_s$, is trained using standard optimization algorithms.

Modified SNIP Algorithm with Attention Scores

Proposed modification to the SNIP algorithm involves the integration of attention scores generated by the transformer model. Its vital to focus on the attention in transformer optimization because the attention mechanism as a pivotal component of the Transformer architecture that has revolutionized the field of deep learning, particularly in natural language processing tasks. At its core, attention allows models to focus on different parts of the input data with varying degrees of emphasis, akin to how humans pay attention to specific details when comprehending information. The attention scores, which indicate the importance of different parts of the input sequence, are used to adjust the importance scores of the weights.

The proposed enhanced Single-shot Network Pruning (SNIP) algorithm incorporates a crucial modification: incorporating attention outputs into criteria evaluated for gradient calculation. The new criteria includes loss function, attention activations and output vector, and calculated using formula 3.

$$L' = L(W) + \sum_j A_j + O \quad (4)$$

In the new formula for criteria, A_j denotes sum of the outputs for *j*-th attention layer, and O denotes sum of the output tensor. Thus, the augmented importance score becomes:

$$s_i = \left| w_i \times \frac{\partial L'}{\partial w_i} \right| \quad (5)$$

The backward pass for the modified criteria increases gradient values depending on the sensitivity of each individual weight in the attention layers. Given the importance of attention mechanisms in transformer models, this knowledge is vital for effective pruning.

The rationale for the effectiveness of this approach lies in the specialized function of attention mechanisms in transformer architectures, which are critical for tasks such as language modeling and translation. By incorporating gradients from attention activations, the pruning strategy becomes more informed, possibly preserving essential features in the pruned model. The modified SNIP algorithm aims to offer a nuanced pruning methodology better suited to the complexities of transformer models.

Experimental Setup

The experiments were conducted on Plant Disease Dataset – Housing tens of thousands of images, the dataset offers a panoramic view of diverse plant species (fig. 1). Labels include disease description, allowing to train a classification model to discern between healthy and diseased plants.

Fig. 1. Sample batch of Plant Disease Dataset

The transformer block within ViT model is replaced by a Linformer architecture to achieve computational efficiency without sacrificing performance (fig. 2).

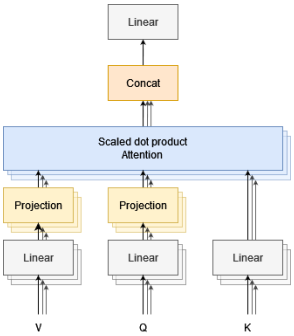

Fig. 2. Architecture diagram of Linformer model

It is engineered to efficiently handle long sequences by reducing the time complexity of the self-attention mechanism from $O(n^2)$ to $O(n)$ where n is the sequence length [17]. This is accomplished using following approaches:

– Fixed-Length Context: The self-attention mechanism in Linformer is designed to focus only on a fixed context window, thereby making the computational complexity invariant to the sequence length.

– Linear Self-Attention: Linformer approximates the full attention matrix by low-rank matrices, effectively reducing the time complexity to linear. This is particularly beneficial for tasks that involve long sequences.

– Kernelized Attention: The architecture employs kernelized attention patterns, which further optimize the computational requirements by eliminating the need for pairwise attention weight calculations.

– Shared Projections: Linformer utilizes shared projections for the key and value matrices in the

self-attention mechanism, thereby reducing the number of parameters and computational load.

In the Linformer model, the linearized attention for each head i is computed as $\overline{\text{head}_i} = \bar{P} \cdot (F_i V W_i^V)$. Here, $\bar{P}$ is the attention probability matrix, which is derived from the softmax normalization of the scaled dot-product term $\frac{QW_i^Q (E_i K W_i^K)^T}{\sqrt{d_k}}$. The matrices QW_i^Q, $E_i K W_i^K$, and $F_i V W_i^V$ represent the projections of the queries, keys, and values, respectively, for the i-th attention head (fig. 3).

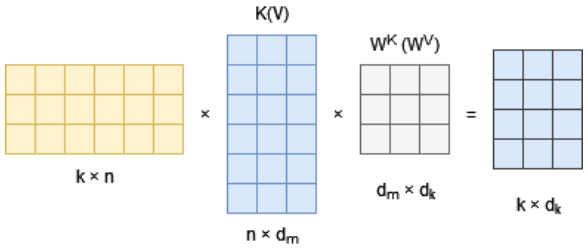

Fig. 3. Calculating projection layers of keys (K) and values (V) matricies

The learned context projection matrices E_i and F_i enable this low-rank approximation, effectively reducing the time complexity to $O(n)$ (eq. 5).

$$\overline{\text{head}_i} = \text{Attention}(QW_i^Q, E_i K W_i^K, F_i V W_i^V) = \underbrace{\text{softmax}\left(\frac{QW_i^Q (E_i K W_i^K)^T}{\sqrt{d_k}}\right)}_{\bar{P}:n\times k} \cdot \underbrace{F_i V W_i^V}_{k\times d} \quad (5)$$

The Linformer architecture in our experiment is configured with a sequence length of 50, which includes 49 patches and one class token. The model dimension is set to 128, and it comprises 12 layers (depth) with 8 attention heads. The kernel size for the attention mechanism is set to 64.

The Linformer block is integrated into a Vision Transformer model tailored for image classification tasks. The input images are resized to 224×224 pixels and are divided into patches of size 32×32. These patches are then linearly embedded into a 128-dimensional space, the same as the Linformer dimension. The input image channels are set to 3, corresponding to the RGB color space.

Results

The model was trained for 50 epochs and its performance was evaluated against the baseline SNIP algorithm. As shown in Table N, the modified method achieved a training accuracy of 98.9% and a validation accuracy of 94.9%, compared to the baseline's 67.9% and 57.9%, respectively. This translates to an improvement of 37% in validation accuracy. Additionally, the modified method also showed significantly lower loss values, with a training loss of 0.03 and a validation loss of 0.18, as opposed to the baseline's 1.01 and 1.22, respectively (table 1).

Table 1

Training results

Method	Train accuracy	Train loss	Val. accuracy	Val. loss
SNIP	67.9	1.01	57.9	1.22
Modified SNIP	98.9	0.03	94.9	0.18

The improvements were achieved on a very sparse model with only 275,738 trainable parameters, which constitutes just 10% of the initial parameters. The training curves, displayed in Figure 4, further substantiate the effectiveness of modified algorithm.

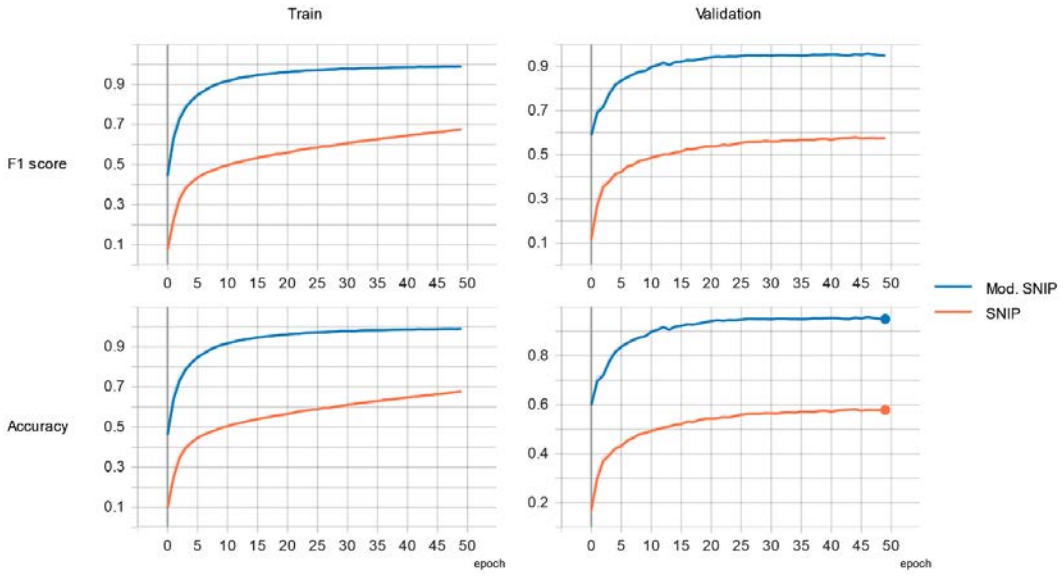

Fig. 4. Accuracy and F1 score comparison for original and modified algorithm

Conclusions

This study proposes a modification to the Sparse Networks from Scratch (SNIP) pruning algorithm by incorporating attention scores generated by the transformer model. Specifically, the hypothesis is that the by performing additional backward passes from attention activations and concatenating the gradients from original method, a more nuanced importance score can be calculated to increase accuracy of resulting model while pruning the same amount of weights.

The results confirm that integrating attention scores into the SNIP algorithm not only improves the model's accuracy by 34% on validation subset, maintaining a sparse parameter set. This opens up new avenues for research in model pruning, particularly in the context of transformer architectures.

REFERENCES

1. Vaswani A., Shazeer N., Parmar N., Uszkoreit J., Jones L., Gomez A. N., Kaiser L., Polosukhin I. (2017). Attention Is All You Need. Advances in Neural Information Processing Systems 30 (NIPS 2017). DOI: https://doi.org/10.48550/arXiv.1706.03762.
2. Liu Y., Lapata M. (2019). Text Summarization with Pretrained Encoders. ArXiv preprint, arXiv:1908.08345. DOI: https://doi.org/10.48550/arXiv.1908.08345.
3. Devlin J., Chang M-W., Lee K., Toutanova K. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. ArXiv preprint, arXiv:1810.04805. DOI: https://doi.org/10.48550/arXiv.1810.04805.
4. Dosovitskiy A., Beyer L., Kolesnikov A., Weissenborn D., Zhai X., Unterthiner T., Dehghani M., Minderer M., Heigold G., Gelly S., Uszkoreit J., Houlsby N. (2021). An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. Proceedings of the International Conference on Learning Representations (ICLR 2021). DOI: https://doi.org/10.48550/arXiv.2010.11929.
5. Rives A., Meier J., Sercu T., Goyal S., Lin Z., Liu J., Guo D., Ott M., Zitnick C. L., Ma J., Fergus R. (2021). Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. Proceedings of the National Academy of Sciences, 118 (15). DOI: https://doi.org/10.1073/pnas.2016239118.
6. Parisotto E., & Salakhutdinov R. (2017). Neural Map: Structured Memory for Deep Reinforcement Learning. ArXiv preprint, arXiv:1702.08360. DOI: https://doi.org/10.48550/arXiv.1702.08360.
7. Han S., Mao H., & Dally W. J. (2016). Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding. ArXiv preprint, arXiv:1510.00149. DOI: https://doi.org/10.48550/arXiv.1510.00149.
8. Molchanov P., Tyree S., Karras T., Aila T., & Kautz J. (2017). Pruning Convolutional Neural Networks for Resource Efficient Inference. ArXiv preprint, arXiv:1611.06440. Doi: https://doi.org/10.48550/arXiv.1611.06440.
9. Zhu M., & Gupta S. (2017). To prune, or not to prune: exploring the efficacy of pruning for model compression. ArXiv preprint, arXiv:1710.01878. DOI: https://doi.org/10.48550/arXiv.1710.01878.
10. Hu H., Peng R., Tai Y-W., & Tang C-K. (2016). Network Trimming: A Data-Driven Neuron Pruning Approach towards Efficient Deep Architectures. ArXiv preprint, arXiv:1607.03250. DOI: https://doi.org/10.48550/arXiv.1607.03250.
11. Li H., Kadav A., Durdanovic I., Samet H., & Graf H. P. (2017). Pruning Filters for Efficient ConvNets. Proceedings of the International Conference on Learning Representations (ICLR 2017). DOI: https://doi.org/10.48550/arXiv.1608.08710.
12. Narang S., Elsen E., Diamos G., & Sengupta S. (2017). Exploring Sparsity in Recurrent Neural Networks. Proceedings of the International Conference on Learning Representations (ICLR 2017). DOI: https://doi.org/10.48550/arXiv.1704.05119.
13. He Y., Zhang X., & Sun J. (2017). Channel Pruning for Accelerating Very Deep Neural Networks. Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp. 1389–1397. DOI: https://doi.org/10.48550/arXiv.1707.06168.
14. Wen W., Wu C., Wang Y., Chen Y., & Li H. (2016). Learning Structured Sparsity in Deep Neural Networks. Neural Information Processing Systems. DOI: https://doi.org/10.48550/arXiv.1608.03665.
15. Lee N., Ajanthan T., & Torr P. H. S. (2019). SNIP: Single-shot Network Pruning based on Connection Sensitivity. Proceedings of the International Conference on Learning Representations (ICLR 2019). DOI: https://doi.org/10.48550/arXiv.1810.02340.
16. Frankle J., & Carbin M. (2019). The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks. Proceedings of the International Conference on Learning Representations (ICLR 2019). DOI: https://doi.org/10.48550/arXiv.1803.03635.
17. Wang S., Li B. Z., Khabsa M., Fang H., & Ma H. (n.d.). Linformer: Self-Attention with Linear Complexity. ArXiv preprint, arXiv:2006.04768. DOI: https://doi.org/10.48550/arXiv.2006.04768.