Greedy-Gnorm: A Gradient Matrix Norm-Based Approach Replacing Attention Entropy for Head Pruning

Yuxi Guo
Southwestern University of Finance and Economics
yuxiguo@udel.edu

October 31, 2024

Abstract

This paper introduces Greedy-Gnorm, a novel method for attention head pruning in transformer models. Greedy-Gnorm utilizes the norms of gradient matrices (Gnorm) to guide pruning decisions through a greedy algorithm, focusing on heads with lower importance scores. Gnorm effectively addresses the issue of uncontrollable gradient changes after pruning heads, ensuring a more stable and reliable pruning process. Additionally, Gnorm and ϵ-rectification addresses limitations of existing methods like Attention Entropy, which can suffer from underflow issues. Experimental results demonstrate that Greedy-Gnorm achieves superior performance and stability across multiple transformer architectures, including BERT, ALBERT, RoBERTa, and XLM-RoBERTa. By dynamically assessing head importance through gradient norms, the proposed method maintains model efficiency without significant accuracy loss.

Keywords: Gnorm, Attention Entropy, greedy algorithm, model compression

1 Introduction

In recent years, deep learning models, especially transformer-based architectures such as BERT, ALBERT and so on, have demonstrated remarkable performance across various natural language processing tasks, like GPT-3 [1] whose size of parameters can reach over 150B. However, the increasing model size presents challenges in terms of parameters redundancy[9] and deployment efficiency, particularly in resource-constrained environments. To define the importances of parameters, we need to find the particular functions of them. Dodrio tells us that different heads play various roles, like semantic or syntactic functions to understand the sentences.[14]Under the assumption that lower entropy signifies a more focused and crucial attention head. However, this method is not without limitations. In certain scenarios, entropy may suffer from underflow issues, which can lead to NAN value spreading phenomenon. Then this paper will give a scheme to solve this problem.

We choose four models to validate the effectiveness of new indicator: Gnorm (Norms of gradient matrix), and this research gives the comparison between Attention Entropy and Gnorm. Models are respectively BERT[4], ALBERT[6], RoBERTa[8] and XLM-RoBERTa[3]. The datasets are also corresponding to their focused tasks(financial-sentiment-analysis, mnli, tweets-sentiment-analysis and language-identification).

2 Related Work

After previous researches, there are some indicators that evaluate the importance of heads in Transformer. Attention Entropy is one of solutions to prune redundancy heads in BERT, but when text becomes longer, tokens will be paid less attention for average. Then Attention Entropy will bring underflow phenomenons for small values of a_{ij}, so that $log(0)$ will appear for computer to spread NAN value. The validation dataset is denoted as set $\mathcal{X}$, then the t_{x_k} means the total input tokens of the k^{th} validation sentence. Here, $a_{ij}^{(h)}$ means the attention score that i^{th} token points towards j^{th} token in the h^{th} head:

$$E\left(AE\left(h\right)\right) = -\frac{1}{|\mathcal{X}|}\sum_{k=1}^{|\mathcal{X}|}\frac{1}{t_{x_k}}\sum_{i=1}^{t_{x_k}}\sum_{j=1}^{t_{x_k}} a_{ij}^{(h)} \cdot log(a_{ij}^{(h)}) \tag{1}$$

Attention Entropy doesn't attach importance to gradients. In contrast, Attr(i.e. self-attention attribution score[5]) and TIS(i.e. Taylor-expansion Importance Scores[17]) takes the gradients of parameters in Transformers into account:

$$Attr_h\left(A\right) = A_h \odot \int_{\alpha=0}^{1} \frac{\partial F\left(\alpha A\right)}{\partial A_h} d\alpha \in R^{n \times n} \tag{2}$$

$$\widetilde{Attr_h\left(A\right)} = \frac{A_h}{m} \odot \sum_{k=1}^{m} \frac{\partial F\left(\frac{k}{m} A\right)}{\partial A_h} \tag{3}$$

$$I_h = E_x\left[max\left(Attr_h\left(A\right)\right)\right] \tag{4}$$

$$TIS_h = E_x\left|A_h^T \frac{\partial \mathcal{L}\left(x\right)}{\partial A_h}\right| \tag{5}$$

These two indicators depends on partial derivatives. Actually, when we prune one head, scores of original, Attr or TIS which rely on gradients will update values. Given that the gradients alter during pruning all the time, we need to calculate new scores after pruning each head, which causes huge computational cost.

3 Our method: Gnorm and AE rectification

3.1 Gnorm

3.1.1 Definition

We present a novel method to evaluate the importances of heads during the dynamic pruning process: Greedy-Gnorm(Greedy Pruning based on Norms of gradient matrix). Transformers are renowned for attention mechanism. Q, K and V weights help models to interpret the text information like human.[13] We calculated the gradients of QKV weights. Select the method of element-by-element multiplication to multiply the norm of the corresponding head QKV gradient matrix to get the Gnorm score of each head, and then use the greedy algorithm to cut, select the head with the smallest Gnorm score to cut. Independent variable n means that there are n heads pruned in models. Gnorm score in Greedy-Gnorm method is defined as following:

$$S(n) = G_{Q(n)} \odot G_{K(n)} \odot G_{V(n)} \tag{6}$$

$G_{Q(n)}$ means the expectation of norms matrix of Q weights gradient matrix after pruning n heads.

$$G_{Q(n)} = E(G_{q(n)}) \tag{7}$$

$G_{q(n)}$ is the gradient norms matrix of all heads. It is defined as following:

$$G_{q(n)} = \begin{pmatrix} ||G_{q(n)}^{(11)}|| & ||G_{q(n)}^{(12)}|| & \cdots & ||G_{q(n)}^{(1H)}|| \\ ||G_{q(n)}^{(21)}|| & ||G_{q(n)}^{(22)}|| & \cdots & ||G_{q(n)}^{(2H)}|| \\ \vdots & \vdots & \ddots & \vdots \\ ||G_{q(n)}^{(L1)}|| & ||G_{q(n)}^{(L2)}|| & \cdots & ||G_{q(n)}^{(LH)}|| \end{pmatrix} \in \mathbb{R}^{L \times H} \tag{8}$$

Where $G_{q(n)}^{(ij)}$ means the gradient matrix of Q weights at i^{th} layer j^{th} head. Correspondingly, $G_{k(n)}^{(ij)}$ and $G_{v(n)}^{(ij)}$ mean the gradient matrix of K and V weights at in i^{th} layer j^{th} head. Based on different inputs in datasets, we need to consider the expectations of the gradient matrix norms and select the head with the smallest Gnorm value. After pruning each head, we all need to find the new Gnorm stages. So S(n) tells that after pruning n heads, which head is the least important to do next pruning. Given the demand of reduce computational complexity, we choose to calculate the L2 norm and $F_n(X)$ means outputs of the Transformer after pruning n heads:

$$||G_{q(n)}^{(ij)}|| = \sqrt{\sum_m (\frac{\partial ||F_n(X)||}{\partial q_m^{(ij)}})^2} \tag{9}$$

$$G_{Q(n)} = E(G_{q(n)}) =$$
$$\begin{pmatrix} \frac{1}{||X||}\sum_{x\in X}\sqrt{\sum_m(\frac{\partial||F_n(x)||}{\partial q_m^{(11)}})^2} & \cdots & \\ \vdots & \ddots & \vdots \\ & \cdots & \frac{1}{||X||}\sum_{x\in X}\sqrt{\sum_m(\frac{\partial||F_n(x)||}{\partial q_m^{(LH)}})^2} \end{pmatrix} \tag{10}$$

3.1.2 Greedy Pruning

We give a general pruning scheme to prune the heads of model, including how to prune the heads based on Gnorm and greedy algorithms and how to address the collapsed gradient matrix after pruning some heads.

Algorithm 1 HOW TO PRUNE HEADS BASED ON GNORM

Input : $G_{l\times h} = [Gnorm_{ij}]_{l\times h}$: *Gnorm matrix of model*, layer
Output : *Accuracy scores of pruned models*
Initialization : $M_{l\times h} = [m_{ij}]_{l\times h}$: *Mask matrix filledwith* $1(m_{ij} = 1)$
procedure PRUNING($G, M, model$)
  while $M \neq [0]_{l\times h}$ **do**
    $A \leftarrow (Gmax - G) \odot M)$
    #Gmax is full of the maximum of G and have the same shape
    find the a_{ij} *which is equal to* $max(A)$
    $m_{ij} \leftarrow 0$ #Prune the j^{th} head in the i^{th} layer of model
  end while
  return $model, newM$
end procedure
procedure EXPANDGRADMATRIX($Grad, M$)
  #Grad: Incomplete gradient matrix;M: Masks list in one layer
  $E \leftarrow \emptyset$
  $cursor \leftarrow 0$
  for all masks in M **do**
    if mask==1 **then**
      put the elements of G *from cursor to cursor* $+ d_k$ *into* E
      $cursor \leftarrow cursor + d_k$
    else
      put $O_{1\times d_k}$ *into* E #$O_{1\times d_k}$ is a zero vector
    end if
  end for
  return E # E: Filled gradient matrix
end procedure
$model, M \leftarrow$ PRUNING($G, M, model$)
Calculate accuracy score
Get new incomplete gradient matrix Grad after pruning
for all layers in model **do**
  $Grad_l \leftarrow$ EXPANDGRADMATRIX($Grad, M[layer]$)
end for #concat all $Grad_l$ into complete gradient matrix Grad for calculating next Gnorm

Step 1 Calculate the Gnorm of QKV weights, find the useless head based on Gnorm

Step 2 Prune the head with smallest Gnorm, the algorithm is given in Algorithm 1. We get the next gradient matrix(Grad).

Step 3 The collapsed gradient matrix is extended so that it retains its shape, and then we go back to Step 1. With filled gradient matrix, we continue to calculate new Gnorm with gradient matrix(Grad) from **Step 2**.

Step 4 The shear schemes at the inflection points of the accuracy curve can be taken out separately as pruning schemes for testing and production.

3.2 AE rectification

3.2.1 ϵ Fine-tune

Because of too many tokens, attention values are dispersed into small values on each tokens which causes that log(0) often appears during calculating Attention Entropy. To solve the underflow issues, we rectify the AE as following:

$$A(a_1, a_2, ..., a_n) = -\sum_{i=1}^{n} a_i log(a_i) \tag{11}$$

$$B(a_1, a_2, ..., a_n) = -\sum_{i=1}^{n} a_i log(a_i + \varepsilon) \tag{12}$$

$$C(a_1, a_2, ..., a_n) = -\sum_{i=1}^{n} (a_i + \varepsilon) log(a_i + \varepsilon) \tag{13}$$

$$C - A = C - B + B - A$$

Where ϵ is an extremely small positive value($\epsilon + a_{ij} < 1$). Obviously, $C - B > 0; B - A < 0$. $B - A < 0$ is easy to proof so that we omit it. Given that $\sum a_{ij} = 1$ and $\epsilon \ll 1/2 \leq \frac{n-1}{n}$, $C - B > 0$ can be considered as following:

$$(\prod_{i} (a_i + \varepsilon))^{\frac{1}{n}} < \frac{n\varepsilon + 1}{n} < 1 \tag{14}$$

Here we choose C as rectification of AE, and the reason for this choice is that $C > B$. Because when B is less than A and A has underflow, logarithmic item in every term of B will no longer have underflow, but we hope that the result of entropy calculation itself will not underflow, so we choose the larger result of C.

3.2.2 Invariant pruning order

$F(a_1, a_2, ..., a_n) = \sum_{i=1}^{n} a_i log(a_i)$ is continuous in R^+. Given that $\sum_{i=1}^{n} x_i = \sum_{i=1}^{n} y_i = 1$ we discuss following mathematic problems:

Let δ be the difference between $F(P_2)$ and $F(P_1)$, $\delta > 0$. (i.e $\delta = F(P_2) - F(P_1) > 0$), then if $P_1' = (x_1 + \varepsilon, x_2 + \varepsilon, ..., x_n + \varepsilon)$ and $P_2' = (y_1 + \varepsilon, y_2 + \varepsilon, ..., y_n + \varepsilon)$, new difference $\delta' = F(P_2') - F(P_1')$, is δ' also positive?

Actually, let's consider a line segment in n-dimensional space that can be represented as follows:

$$\vec{l} = (l_1, l_2, ..., l_n) = \frac{\vec{P_1 P_2}}{||\vec{P_1 P_2}||} = \frac{\vec{P_1' P_2'}}{||\vec{P_1' P_2'}||} \tag{15}$$

In fact, point P_1 and point P_2 in n-dimensional space form a line segment in N-dimensional space, and the quality of the length of the line segment is the value of the curve integral. The density of the line is the directional gradient of the function F, so actually the directional length of the integrated line segment does not change, but the density does. The original difference can be represented as following:

$$\delta = F(P_2) - F(P_1) = \int_{P_1}^{P_2} \bigtriangledown F \vec{l}\, dl > 0 \tag{16}$$

Where $\bigtriangledown F = (log(a_1) + 1, log(a_2) + 1, ..., log(a_n) + 1)$:

$$\bigtriangledown F \vec{l} = \sum_{i=1}^{n} l_i + \sum_{i=1}^{n} l_i log(a_i) = \sum_{i=1}^{n} (y_i - x_i) + \sum_{i=1}^{n} l_i log(a_i) = \sum_{i=1}^{n} l_i log(a_i) \tag{17}$$

Because the function is continuous, when ε is small enough, it falls in the small neighborhood and the gradient sign remains unchanged:

$$\bigtriangledown F' \vec{l} = \sum_{i=1}^{n} l_i log(a_i + \varepsilon) \in N_r(\bigtriangledown F \vec{l}) \tag{18}$$

$$s.t.\ \delta' = F(P_2') - F(P_1') = \int_{P_1'}^{P_2'} \bigtriangledown F' \vec{l}\, dl > 0 \tag{19}$$

Here r can control the change of the integral value in a small range, so that the sign of the integral value remains unchanged. Then the size relationship of attention entropy remains unchanged, and thus ensure that our rectification can be smoothly clipped without changing the original clipping order, finally solving the underflow problem of AE.

4 Experiments

4.1 Experimental Setup

To keep the comparison as meaningful as possible, we choose four different kinds of models which are respectively BERT, ALBERT, RoBERTa and XLM-RoBERTa. Their tasks focus on financial-sentiment-analysis, mnli, tweets-sentiment-analysis and language-identification. With textpruner as our analysis tools[16], inverse pruning is used to validate the effectiveness of Gnorm. However, ALBERT's structure is single-layer reused for 12 times so that the masks only have 12 to control 144 heads. Fig. 1 shows the example for pruning process with merely 3 mask.

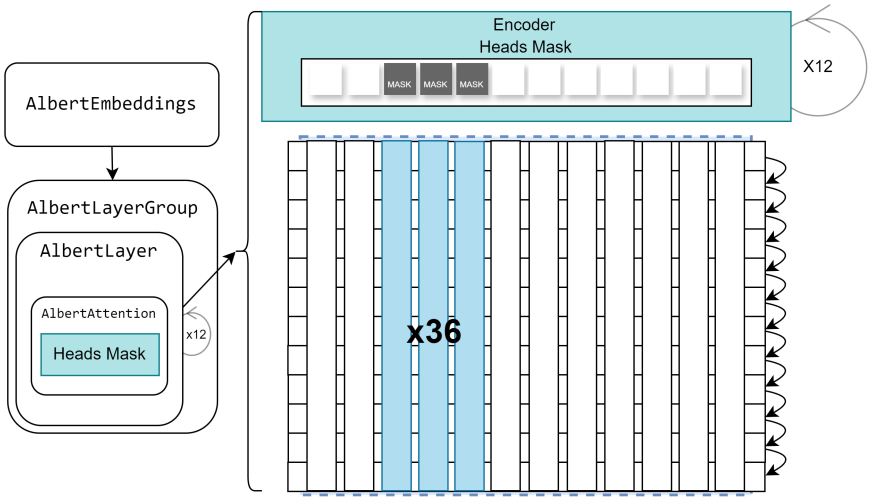

Figure 1: One example: Special structure of ALBERT during pruning based on only 3 mask

Datasets Applied datasets to test changes of accuracies are "hw2942/financial-news-sentiment", "nyu-mll/glue", "siebert/sentiment-roberta-large-english" and "papluca/language-identification".

Models During doing experiments, we have to pay attention to the special parameters sharing mechanism in ALBERT. Each mask in ALBERT means pruning 12 heads in 12 layer at the same position.

Selected models includes:

- BERT "hw2942/bert-base-chinese-finetuning-financial-news-sentiment-v2";
- ALBERT "Alireza1044/albert-base-v2-mnli";
- RoBERTa "rahmaabusalma/tweets_sentiment_analysis";
- XLM-RoBERTa "papluca/xlm-roberta-base-language-detection".

4.2 Pruning test of gradient matrix

We need to test a phenomenon first: after cutting off a head, the parameter gradient of other heads changes. If there is a change, indicating that the greedy algorithm cut a head to calculate a Gnorm idea is more reasonable.

The variation of the gradient matrix is visualized here and presented as a heat map. We cut the heads outside the observed object layer and the heads inside the observed object layer. After capturing two gradient matrices of pruned Q weights respectively, and the original gradient matrix is different to observe the heat maps of their differences. If heat maps do not have exactly the same color, it is obvious that the gradient changes.

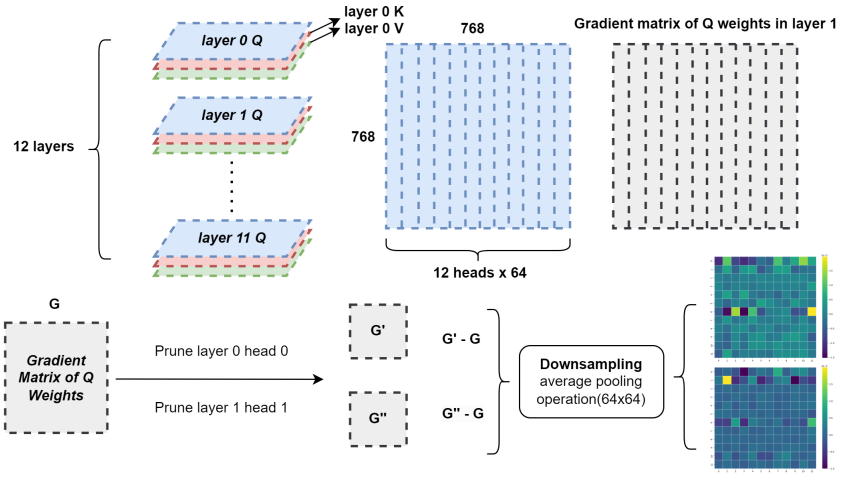

Figure 2: Sampled gradient matrix changes after pruning heads

For the convenience of presentation, downsampling can reduce the 768×768 gradient matrix into 12×12 matrix with 64×64 pooling layer. The results show whether the pruned heads are at layers above object layer or not. The gradient matrix always changes, which means that we have to calculate the importance scores after pruning each head of models.

4.3 Comparison Experiments

We compare the four kinds of prunings: AE Pruning, Inverse AE Pruning, Greedy Gnorm Pruning, Inverse Greedy Gnorm Pruning. Greedy-Gnorm performs excellently when pruning BERT, ALBERT and XLM-RoBERTa. Unlike BERT popular to prune[10], RoBERTa(larger model) has 384 heads, Fig. 3 shows that useful heads take up about 30%. After inverse pruning of Greedy Gnorm, when 150 heads are pruned, the accuracy remains constant level. For four curves of Inverse Greedy Gnorm Pruning results in four models, about 30% heads are really useful, playing an importance role for text understanding. After pruning these heads, the models can be regarded as untrained models which are disable for their tasks.

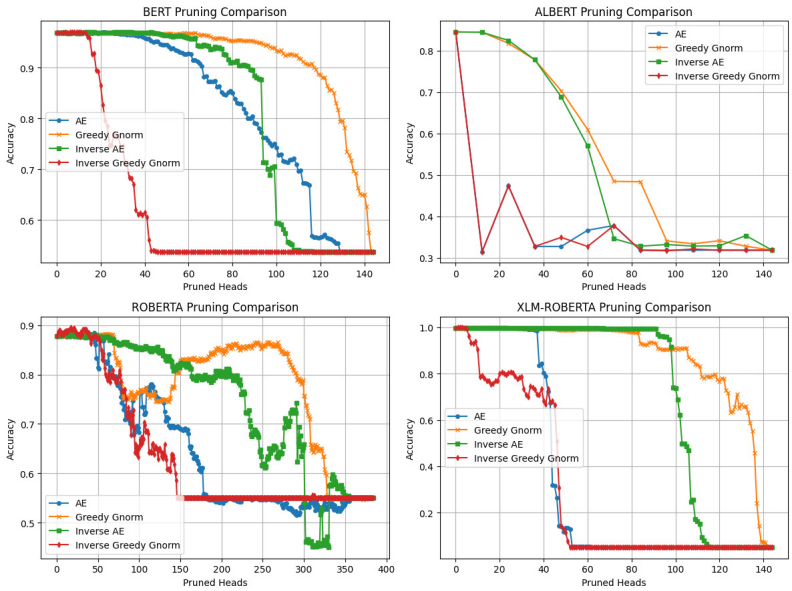

Figure 3: Greedy-Gnorm and AE Pruning Comparison

We adopted Heads Gates method to prune the heads in four models.[12] The heads mask component is set up during concatenating the different outputs of each head. So the heads gates will be closed one by one as

greedy algorithms work for pruning. The gates which first closed is the least important based on Gnorm. Left heads at last always have great considerable gradient to affect the outputs for model.

Fig. 3 shows that the Greedy-Gnorm pruning can outstand AE pruning method. Especially for BERT, despite the left heads less than 20%, the accuracy can still maintain 90%. At least, it means that Greedy-Gnorm can tell which head is really important for models.

4.4 Pruning solutions

Fig. 4 shows our final chosen solutions for four models. After pruning heads, their accuracies are kept as following: 90.08%, 77.76%, 86.40% and 90.97%.

Table 1: The solutions of Greedy-Gnorm for four models(BP means "Before pruning", AP means "After pruning")

	BERT	ALBERT	RoBERTa	XLM-RoBERTa
Accuracy(BP)	96.82%	84.48%	87.80%	99.73%
Accuracy(AP)	90.08%	77.76%	86.40%	90.97%
Size(BP)	390.13	44.58	1355.60	1060.71
Size(AP)	302.29	42.33	1110.42	981.88

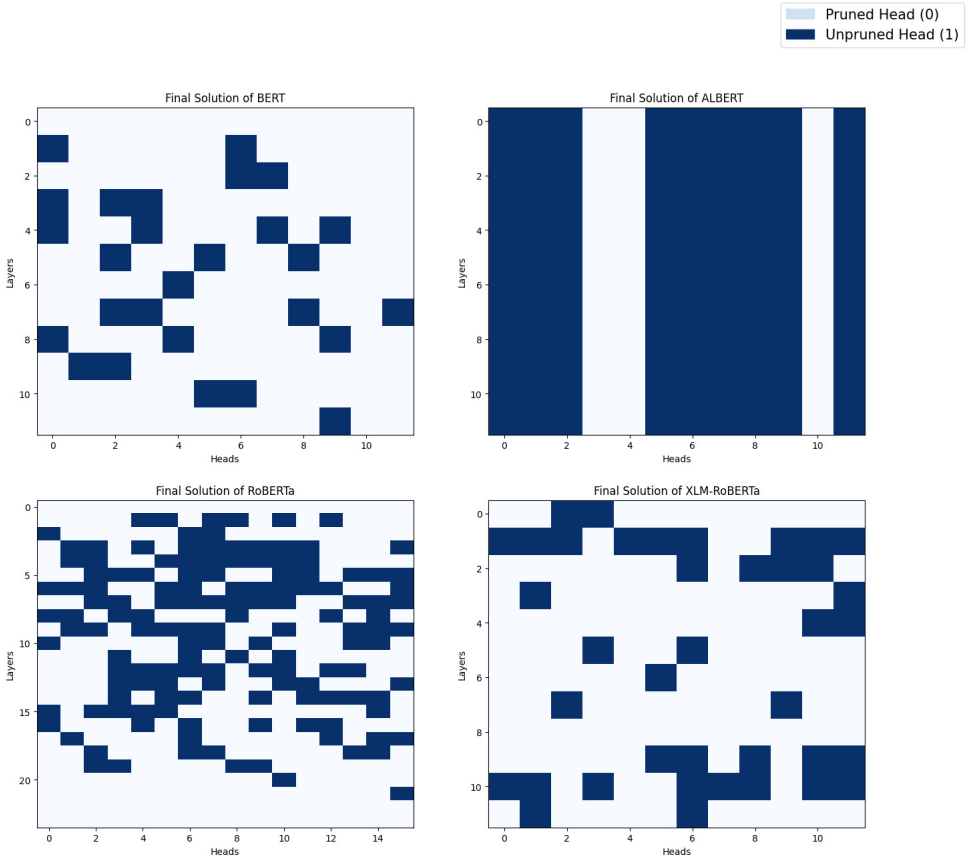

Figure 4: The final pruning solutions for four kinds of model

BERT The pruning solution for BERT reveals a selective pruning strategy. The retained heads are scattered across all 12 layers, indicating that attention heads in varying depths contribute differently to model performance. This solution suggests the presence of specialized functions performed by different heads at various layers, highlighting the flexibility of BERT's architecture.

ALBERT In contrast, the pruning result for ALBERT displays a uniform pattern with clear vertical bands of retained heads. This observation indicates that specific attention heads consistently hold higher importance across all layers, while others are consistently pruned. This result could be due to ALBERT's parameter-sharing mechanism, which enforces identical attention configurations across its layers, promoting the uniformity of the solution.

RoBERTa The heatmap for RoBERTa reveals a more complex pruning pattern, where retained heads are distributed irregularly across 24 layers. This irregularity suggests that RoBERTa's attention heads serve a diverse range of functions across layers, with some layers relying on a higher number of unpruned heads. The intricate nature of this pruning pattern reflects the robustness of RoBERTa's design, where more heads are retained to support nuanced attention mechanisms.

XLM-RoBERTa For XLM-RoBERTa, the heatmap indicates a selective pruning scheme that retains attention heads across layers in a dispersed manner. While the distribution of retained heads appears more balanced than in RoBERTa, there is a focus on preserving certain heads within specific layers, suggesting these heads may play crucial roles in multilingual tasks or cross-lingual transfer learning.

Here we take the BERT as example to observe the differences of all parts in model after pruning.

Table 2: The parameters condition of original BERT and pruned BERT(BP means "Before pruning", AP means "After pruning")

Layer Name	Params(BP)	Ratio(BP)	MB(BP)	Params(AP)	Ratio(AP)	MB(AP)
Model	102,269,955	100.00%	390.13	79,244,355	100.00%	302.29
Bert	102,267,648	100.00%	390.12	79,242,048	100.00%	302.28
Embeddings	16,622,592	16.25%	63.41	16,622,592	20.98%	63.41
Encoder	85,054,464	83.17%	324.46	62,028,864	78.28%	236.62
Pooler	590,592	0.58%	2.25	590,592	0.75%	2.25
Classifier	2,307	0.00%	0.01	2,307	0.00%	0.01
Weight	2,304	0.00%	0.01	2,304	0.00%	0.01
Bias	3	0.00%	0.00	3	0.00%	0.00
Accuracy		0.9682			0.9008	

5 Conclusion

In this study, we proposed Greedy-Gnorm as a new head pruning method for transformer-based models, leveraging the norms of gradient matrices to select and prune less essential attention heads. By replacing Attention Entropy with Gnorm, our approach successfully mitigates underflow issues and provides a more stable pruning mechanism. The results indicate that Greedy-Gnorm outperforms existing pruning methods in terms of maintaining accuracy, even after substantial model compression. Future work could explore extending Greedy-Gnorm to additional transformer models like Electra[2], BART[7], T5[11] and mT5[15], as well as investigating potential improvements in gradient-based pruning strategies. The findings suggest that Greedy-Gnorm holds promise for efficient and reliable transformer model compression, enhancing deployment feasibility in resource-constrained environments.

Bibliography

[1] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners. *CoRR*, abs/2005.14165, 2020.

[2] K. Clark, M. Luong, Q. V. Le, and C. D. Manning. ELECTRA: pre-training text encoders as discriminators rather than generators. *CoRR*, abs/2003.10555, 2020.

[3] A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, and V. Stoyanov. Unsupervised cross-lingual representation learning at scale. *CoRR*, abs/1911.02116, 2019.

[4] J. Devlin, M. Chang, K. Lee, and K. Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

[5] Y. Hao, L. Dong, F. Wei, and K. Xu. Self-attention attribution: Interpreting information interactions inside transformer. *CoRR*, abs/2004.11207, 2020.

[6] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut. ALBERT: A lite BERT for self-supervised learning of language representations. *CoRR*, abs/1909.11942, 2019.

[7] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *CoRR*, abs/1910.13461, 2019.

[8] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019.

[9] P. Michel, O. Levy, and G. Neubig. Are sixteen heads really better than one? *CoRR*, abs/1905.10650, 2019.

[10] A. Parnami, R. Singh, and T. Joshi. Pruning attention heads of transformer models using a* search: A novel approach to compress big NLP architectures. *CoRR*, abs/2110.15225, 2021.

[11] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *CoRR*, abs/1910.10683, 2019.

[12] K. Shim, I. Choi, W. Sung, and J. Choi. Layer-wise pruning of transformer attention heads for efficient language modeling. *CoRR*, abs/2110.03252, 2021.

[13] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017.

[14] Z. J. Wang, R. Turko, and D. H. Chau. Dodrio: Exploring transformer models with interactive visualization. *CoRR*, abs/2103.14625, 2021.

[15] L. Xue, N. Constant, A. Roberts, M. Kale, R. Al-Rfou, A. Siddhant, A. Barua, and C. Raffel. mt5: A massively multilingual pre-trained text-to-text transformer. *CoRR*, abs/2010.11934, 2020.

[16] Z. Yang, Y. Cui, and Z. Chen. TextPruner: A model pruning toolkit for pre-trained language models. In V. Basile, Z. Kozareva, and S. Stajner, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 35–43, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[17] Y. Zhong and Y. Zhou. Heat: Head-level parameter efficient adaptation of vision transformers with taylor-expansion importance scores, 2024.