Finding the Pillars of Strength for Multi-Head Attention

Jinjie Ni and **Rui Mao** and **Zonglin Yang** and **Han Lei** and **Erik Cambria**

Nanyang Technological University, Singapore

{jinjie001, leih0003}@e.ntu.edu.sg,

{rui.mao, zonglin.yang, cambria}@ntu.edu.sg

Abstract

Recent studies have revealed some issues of Multi-Head Attention (MHA), e.g., redundancy and over-parameterization. Specifically, the heads of MHA were originally designed to attend to information from different representation subspaces, whereas prior studies found that some attention heads likely learn similar features and can be pruned without harming performance. Inspired by the minimum-redundancy feature selection, we assume that focusing on the most representative and distinctive features with minimum resources can mitigate the above issues and lead to more effective and efficient MHAs. In particular, we propose Grouped Head Attention, trained with a self-supervised group constraint that group attention heads, where each group focuses on an essential but distinctive feature subset. We additionally propose a Voting-to-Stay procedure to remove redundant heads, thus achieving a transformer with lighter weights. Moreover, our method achieves significant performance gains on three well-established tasks while considerably compressing parameters.

1 Introduction

Transformers have shown promising performance across various tasks . However, they have some issues, e.g., redundancy and over-parameterization, which is mainly caused by Multi-Head Attention (MHA) (Michel et al., 2019; Voita et al., 2019) and Feed-Forward Network (FFN) (Sukhbaatar et al., 2019; Wu et al., 2019, 2020) of transformer. We aim to mitigate the redundancy and over-parameterization issues by optimizing the MHA module. The multi-heads were originally designed to attend to different representation subspaces of input (Vaswani et al., 2017). However, prior works (Michel et al., 2019; Voita et al., 2019) have shown that the attention heads are highly redundant and over-parameterized after training because some heads can be switched off with a negligible performance drop.

Such an issue is probably caused by their parallel design: the heads naturally work in the same way and likely attend to similar features (Cordonnier et al., 2020). The existing redundancy optimization methods are mainly based on homogenization, diversification, and head significance. However, they all have some limits. **(1)** The homogenization-based methods mitigate redundancy and over-parameterization by making heads similar and removing unnecessary parameters. Cordonnier et al. (2020) homogenized attention heads by sharing most weights between all heads, which reduced the redundant parameters but sacrificed the performance somewhat because of the lack of diversity. **(2)** The diversification-based methods diversify the heads to enrich features and reduce the inter-head redundancy. Li et al. (2018) found that diversifying attention heads by adding a regularization could force MHA to reduce inter-head information redundancy, yielding performance gains in Machine Translation. However, such strategy that retains all feature subsets is suboptimal, because it does not address the issue that MHA is over-parameterized. **(3)** The significance-based methods (Michel et al., 2019; Voita et al., 2019; Li et al., 2021) learn significance scores for the heads to prune unimportant ones. However, the retained important heads still remain inter-head redundancy without diversifying them.

Considering the issues of the above-mentioned methods, we hypothesize that attending to the most representative and distinctive feature subsets with minimum resources leads to more effective and efficient MHAs, which is inspired by the minimum-redundancy feature selection (Cordonnier et al., 2020). Accordingly, we propose a divide-and-conquer strategy, including Group-Constrained Training (GCT) and Voting-to-Stay (V2S), to achieve the setting of our assumption and mitigate the above-mentioned issues. We illustrate them below.

We first propose a strategy to group and distinguish attention heads, where a Grouped Head Attention (GHA) is obtained via the self-supervised GCT. By encouraging homogenization within a group and diversification between groups, the MHA is forced to divide its heads to work in several separate groups, where each group focuses on an essential but unique feature subset, being in line with the setting of our assumption. Note that the redundancy exists when the resources deployed by the model are more than enough to process current information (Cordonnier et al., 2020). GHA reduces the redundancy in two aspects:

- The intra-group homogenization reduces redundancy by encouraging similar intra-group heads and only retaining the most representative one later to lower the resource deployment.

- The inter-group diversification reduces redundancy by forcing heads to attend to more diversified features (with less overlap between heads) so that the unique information to process increases and matches the resources deployed.

Next, we show that GHA-PS (GHA with the Pillar of Strength), a lighter-weight GHA, can be achieved by excluding the redundant heads of GHA via the V2S procedure. V2S culls the redundant heads that share similar patterns with the most representative head (PS head) of a group, which is selected by voting on different training batches. Note that upon the convergence of the GCT, the heads are highly homogenized within a group, thus being redundant because they process similar information. As a result, once the redundant heads are culled, the PS heads can still achieve the essential utility of the original attention layer and yield comparable performance to the unculled model. The Lottery Ticket hypothesis (Frankle and Carbin, 2019) argues that subnetworks in an over-parameterized neural network can converge faster and achieve comparable or better performance than the original network. Our GHA-PS achieving better results is also in line with this hypothesis.

Such a divide-and-conquer combination resolves the issues of previous redundancy optimization methods: (1) Our model achieves better parameter efficiency, resolving the issue of previous diversification-based methods; (2) The feature diversity is guaranteed and the inter-head redundancy is reduced, resolving the problems of previous homogenization- and significance-based methods.

We evaluate our method on three benchmarking tasks. We denote the corresponding transformer architectures of GHA and GHA-PS as Grouped Head Transformers (GHT) and Grouped Head Transformers with the Pillars of Strength (GHT-PS), respectively. GHT and GHT-PS achieve significant improvements over the strong baselines in Machine Translation (MT) BLEU scores (+3.8% and +4.4% averaged on 7 datasets), Language Modeling (LM) perplexity (-2.8% and -2.9%), and Abstractive summarization (AS) F1-Rouge (+6.7% and +7.0% on average). GHT-PS exhibits higher efficiency in model size, inference speed, and floating-point operations (FLOPs). The light architecture of GHT-PS reduces 63.6% parameters of the vanilla transformer and yields comparable performance. The key contributions of our work[1] are threefold:

- We find that, in a certain range, higher compactness of attention heads (i.e., the intra-group heads become closer to each other and the inter-group ones become farther) improves MHA's performance, forcing MHA to focus on the most representative and distinctive features. It provides guidance for future architectural designs of MHA.

- We propose a divide-and-conquer strategy that consists of GCT and V2S. It mitigates the redundancy and over-parameterization issues of MHA. Our method uses fewer parameters and achieves better performance, outperforming the existing MHA redundancy/parameter reduction methods.

- We verify our methods on three well-established NLP tasks. The superior results on datasets with multiple languages, domains, and data sizes demonstrate the effectiveness of our method.

2 Related Work

Parameter efficiency. Different methods were proposed to achieve lightweight transformers: **(1)** replacing attention with lightweight modules, e.g., convolution modules, such as Dynamic Conv (Wu et al., 2019) and Lite Transformer (Wu et al., 2020); **(2)** removing or replacing the feed-forward layers, such as Sukhbaatar et al. (2019) and Wu et al. (2020); **(3)** pruning the model, such as Michel et al. (2019), Voita et al. (2019), and Li et al. (2021).

---

[1]https://github.com/SenticNet/GHA

Modified multi-head mechanism. Ahmed et al. (2017) learned to weight the projected output of different heads, performing weighted sum over them. Li et al. (2019) aggregated the output of different heads by dynamic routing; Cui et al. (2019) used different attention mechanisms, e.g., global/local and forward/backward attention for different heads; Shazeer et al. (2020) mixed different heads before and after the softmax operation in an attention function to achieve communication between heads.

Head redundancy optimization. Michel et al. (2019) and Voita et al. (2019) found that only a subset of the attention heads have significant utilities in transformer, where the important heads could be identified by Expected Sensitivity and Layer-wise Relevance Propagation (LRP) (Ding et al., 2017). Upon this, Li et al. (2021) learned per-head importance scores and pruned the heads. Cordonnier et al. (2020) homogenized the attention heads by sharing a part of the weights between heads, which lowered the number of parameters but sacrificed performance. Li et al. (2018) found that diversifying attention heads by adding a regularization can force MHA to reduce inter-head redundancy, yielding performance gains for Machine Translation. However, previous methods either traded performance for efficiency or retained extra parameters.

3 Methodology

There are two core components in our method, namely the Group-Constrained Training (GCT) and the Voting-to-Stay (V2S) procedure. GHA (Figure 1) is developed with GCT that removes head redundancy; GHA-PS is developed by removing the redundant parameters of GHA in V2S. In this section, we detail the process of developing the GHA and finding its Pillars of Strength (PS).

3.1 Grouped Head Attention with Hidden Units

First, we detail the core module of GHT, the GHA with hidden units, which is built based on MHA via the GCT. The GCT divides the attention heads of MHA into several groups and makes heads within a group become more similar, whereas heads between groups become more different. Thus, MHA is forced to divide its heads to work in several separate groups, where each group focuses on an essential but unique feature subset to reduce head redundancy. We will show the effectiveness in § 5.
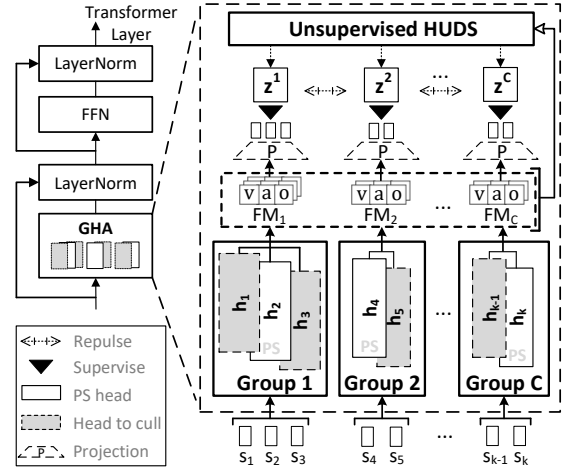

Figure 1: The Grouped Head Attention. The heads in a group are under self-supervision of the discovered group hidden units $\mathbf{Z}$ (Eq.4). The non-PS heads (dashed gray boxes in a group) will be culled in the VS procedure (Algorithm 1). S_k denotes the k-th representation subspace; FM_C denotes the C-th feature map group. The output of GHA is omitted for simplicity.

Given a transformer model $f(\mathbf{x}; \theta)$ with n attention layers, the set of heads at attention layer l is denoted as $\mathbf{H}_l = \{\mathbf{h}_{1,l}, ..., \mathbf{h}_{k,l}\}$, where k is the number of heads. The outputs of the attention heads are concatenated and projected with W^{out}, where the i-th head output $\mathbf{o}_{i,l}$ in layer l results from the computation of the projection matrices $W_{i,l}^{\mathbf{Q}}$, $W_{i,l}^{\mathbf{K}}$, and $W_{i,l}^{\mathbf{V}}$ of this head:

$$MHA_l(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = Concate(\mathbf{o}_{1,l}, ..., \mathbf{o}_{k,l})W^{out} \tag{1}$$

$$\mathbf{o}_{i,l} = softmax(\frac{(\mathbf{Q}W_{i,l}^{\mathbf{Q}})(\mathbf{K}W_{i,l}^{\mathbf{K}})^T}{\sqrt{d_k}})(\mathbf{V}W_{i,l}^{\mathbf{V}}). \tag{2}$$

Three feature maps (FMs) of GHA are extracted for the self-supervised GCT: **(1)** the result of $\mathbf{V}W_l^{\mathbf{V}}$, denoted as $\hat{\mathbf{V}}_l = \{\mathbf{v}_{1,l}, ..., \mathbf{v}_{k,l}\}$ (the value FM); **(2)** the attention weights of the l-th layer, denoted as $\mathbf{A}_l = \{\mathbf{a}_{1,l}, ..., \mathbf{a}_{k,l}\}$ (the attention FM); **(3)** the output of the l-th layer before the output projection W^{out}, denoted as $\mathbf{O}_l = \{\mathbf{o}_{1,l}, ..., \mathbf{o}_{k,l}\}$ (the head output FM). Moreover, $\hat{\mathbf{V}} = \{\hat{\mathbf{V}}_1, ..., \hat{\mathbf{V}}_l\}$, $\mathbf{A} = \{\mathbf{A}_1, ..., \mathbf{A}_l\}$, $\mathbf{O} = \{\mathbf{O}_1, ..., \mathbf{O}_l\}$. Given the FMs, a Hidden Unit Discovery System (HUDS) Ω assigns a hidden unit $\mathbf{z}_{i,l}^j$ for each head to represent its group property, where i denotes the i-th head and j denotes the j-th group hidden unit. $\mathbf{z}_{i,l}^j \in \hat{\mathbf{Z}}_l$, where $\hat{\mathbf{Z}}_l = \{\mathbf{z}_l^1, ..., \mathbf{z}_l^C\}$ represents the hidden unit candidates, and the hidden units assigned to the heads are denoted as $\mathbf{Z}_l = \{\mathbf{z}_{i,l}, ..., \mathbf{z}_{i,l}\}$.

$\mathbf{Z}_l$ is discovered by the HUDS Ω: $\mathbf{Z}_l = \Omega(\mathbf{E}_l)$, where $\mathbf{E}_l$ denotes either one of the $\hat{\mathbf{V}}_l$, $\mathbf{A}_l$, or $\mathbf{O}_l$. Here $\Omega(\cdot)$ is an unsupervised algorithm that divides the heads into C groups given their FMs, such as K-means[2]:

$$\Omega(\mathbf{E}_l) = \arg\min_{\mathbf{Z}_l} \sum_{i=1}^{C} \sum_{\mathbf{x} \in \hat{\mathbf{E}}_l^i} ||\mathbf{x} - \mu_i||^2, \quad (3)$$

where $\hat{\mathbf{E}}_l^i$ is the set of feature maps of the i-th head group in the l-th attention layer. Then, the feature map groups of the l-th attention layer are denoted as $\hat{\mathbf{E}}_l = \{\hat{\mathbf{E}}_l^1, ..., \hat{\mathbf{E}}_l^i, ..., \hat{\mathbf{E}}_l^C\}$. μ_i is the mean of the feature map vectors in $\hat{\mathbf{E}}_l^i$. The hidden units $\mathbf{Z} = \{\mathbf{Z}_1, ..., \mathbf{Z}_l\}$ are C-class categorical variables (Eq.4(A)) or continuous vectors (Eq.4(B)) to supervise the GCT. The objective of the self-supervised GCT is termed as:

$L_z(\mathbf{f}; \mathbf{A}, \hat{\mathbf{V}}, \mathbf{O}, \mathbf{Z}) =$

$$\begin{cases} -\dfrac{1}{kn}\alpha \sum_{l=1}^{n} \sum_{i=1}^{k} \log p_z(\mathbf{z}_{i,l}|\mathbf{v}_{i,l}, \mathbf{a}_{i,l}, \mathbf{o}_{i,l}) + \\ \dfrac{1}{(C-1)kn}\beta \sum_{l=1}^{n} \sum_{i=1}^{k} \sum_{j_2 \neq j_1} \log p_z(\mathbf{z}_l^{j_2}|\mathbf{v}_{i,l}^{j_1}, \mathbf{a}_{i,l}^{j_1}, \mathbf{o}_{i,l}^{j_1}) \quad \text{(A)} \\ \dfrac{1}{kn}\alpha \sum_{l=1}^{n} \sum_{i=1}^{k} \varphi(\mathbf{v}_{i,l}, \mathbf{a}_{i,l}, \mathbf{o}_{i,l}; \mathbf{z}_{i,l}) - \\ \dfrac{1}{\binom{C}{2}n}\beta \sum_{l=1}^{n} \sum_{j_1=1}^{C-1} \sum_{j_2=j_1+1}^{C} \varphi(\mathbf{z}_l^{j_1}; \mathbf{z}_l^{j_2}) \quad \text{(B)} \end{cases}$$

$$(4)$$

Either when $\mathbf{Z}$ are categorical variables (Eq.4(A)) or continuous vectors (Eq.4(B)), the objective is composed of a homogenization term and a diversification term[3]. $\mathbf{v}_{i,l}^j$, $\mathbf{a}_{i,l}^j$, and $\mathbf{o}_{i,l}^j$ denote the feature maps of the i-th head belonging to the j-th group. $p_z(\mathbf{z}_{i,l}|\mathbf{v}_{i,l}, \mathbf{a}_{i,l}, \mathbf{o}_{i,l})$ denotes the predicted probability of the assigned group hidden variable $\mathbf{z}_{i,l}$, given $\mathbf{v}_{i,l}$, $\mathbf{a}_{i,l}$, and $\mathbf{o}_{i,l}$. $\varphi(\mathbf{x}; \mathbf{y})$ denotes a cosine similarity measurement between $\mathbf{x}$ and $\mathbf{y}$ (following Li et al. (2018)). $\varphi(\mathbf{v}_{i,l}, \mathbf{a}_{i,l}, \mathbf{o}_{i,l}; \mathbf{z}_{i,l}) = \tau_1\varphi(\mathbf{v}_{i,l}; \mathbf{z}_{i,l}) + \tau_2\varphi(\mathbf{a}_{i,l}; \mathbf{z}_{i,l}) + \tau_3\varphi(\mathbf{o}_{i,l}; \mathbf{z}_{i,l})$, where τ is a coefficient, determined by the specific settings for each dataset & task. When $\mathbf{Z}$ are categorical variables, the grouping is a classification task whose classification heads project the output into C classes; when $\mathbf{Z}$ are continuous vectors, the grouping process is a metric learning task whose

similarity computations are conducted between $\mathbf{Z}$ and the projected FM representations. In both conditions, GHA is supervised by $\mathbf{Z}$ to make the heads in the same group yield similar patterns, whereas those in different groups repulse from each other. The overall objective is given by $L = L_t + L_z$, where L_t is the task-specific objective.

3.2 The Pillars of Strength

Being consistent with Lottery Ticket hypothesis (Frankle and Carbin, 2019), we establish the GHT-PS from GHT as its subnetwork by removing redundant heads from GHA to achieve higher parameter efficiency. We propose the V2S procedure to find the Pillars of Strength (PS) heads that constitute the core of the GHA and remove other heads. We first describe the V2S roughly. In GHA, the heads within each group exhibit similar patterns upon the convergence of the Group-Constrained Training (GCT). Then, we only keep the heads with the most explicit group patterns (the PS heads), and switch off the other ones within the same group via V2S. The main idea of V2S is to vote on all heads of the GHA, and only retain one head for each group – the head receiving the most votes. Specifically, it takes an entire epoch to collect the layer-wise votes $\mathbf{m}_l^b \in \{0,1\}^k$ from the whole training set (each data batch b creates one layer-wise vote $\mathbf{m}_l^b$ per attention layer), where k denotes the head number; 0 indicates that the corresponding head should be switched off and 1 indicates that a head is retained.

We assume that there are B mini-batches in the training set. Then, each attention layer receives B layer-wise votes within which each head-wise vote is denoted by either 0 or 1. For each group, the head receiving the most '1's are assigned a '1' in the final head mask $\mathbf{m}_l \in \{0,1\}^k$ for attention layer l, indicating that this head will be retained. Following Michel et al. (2019) and Voita et al. (2019), we mask out the output of heads as the equivalent operation of head removal[4]. The V2S procedure is outlined in Algorithm 1. We detail some of its definitions below. **(1)** ρ indicates the full convergence of GHT, i.e., the hidden units found by Ω have a center shift less than a threshold. **(2)** In Step 7-9, given feature maps $\hat{\mathbf{V}}_l$, $\mathbf{A}_l$, and $\mathbf{O}_l$ of the l-th attention layer, the vote vectors $\mathbf{m}_{l,v}^b$, $\mathbf{m}_{l,a}^b$, and $\mathbf{m}_{l,o}^b \in \{0,1\}^k$ are determined by the group pattern scores $\eta_{i,l}$ of each head, indicating the explicitness of group patterns.

---

[2]K-means fixes the group numbers for fair comparisons in §5. Other clustering algorithms may also be applicable.

[3]The coefficients α and β of Eq.4 respectively control the intra-group homology and inter-group diversity degrees to achieve different group intensities in different tasks/datasets.

[4]We perform real head removal when test inference speed.

Algorithm 1 The Voting-to-Stay (V2S) algorithm

1: **Procedure** Voting-to-Stay($\mathbf{f}, \hat{\mathbf{V}}, \mathbf{A}, \mathbf{O}, \mathbf{Z}$)
2: **if** satisfy ρ, and $\mathbf{m}$ is none **then**
3:     Start voting epoch; Freeze $\mathbf{f}$.
4:     $\mathbf{\Gamma}_l \leftarrow [\ ]$ ▷ Creat $\mathbf{\Gamma}_l$ to store votes
5:     **for** batch b in B training batches **do**
6:         **for** layer l in L layers **do**
7:             **for** $\mathbf{E}_l$ in $\{\hat{\mathbf{V}}_l, \mathbf{A}_l, \mathbf{O}_l\}$ **do**
8:                 Based on $\eta_l = \{\eta_{1,l}, ..., \eta_{1,k}\}$,
9:                 create $\mathbf{m}_{l,v}^b, \mathbf{m}_{l,a}^b, \mathbf{m}_{l,o}^b$.
10:                Add $\mathbf{m}_{l,v}^b, \mathbf{m}_{l,a}^b, \mathbf{m}_{l,o}^b$ to $\mathbf{\Gamma}_l$.
11:     **for** l in n **do**   ▷ Vote at each attn layer
12:         $\mathbf{m}_l \leftarrow VOTE(\mathbf{\Gamma}_l)$
13:     $\mathbf{m} \leftarrow [\mathbf{m}_1, ..., \mathbf{m}_n]$ ▷ Stack layer votes
14:     Unfreeze $\mathbf{f}$; end voting epoch.
15: $\mathbf{f} = \mathbf{f} \odot \mathbf{m}$ ▷ Mask GHT attn outputs with $\mathbf{m}$

We set the corresponding digit in the vote vectors as 1 for the head achieving the highest $\eta_{i,l}$ in its group, indicating the most representative head of the group. Here $\eta_{i,l} = p_z(\mathbf{z}_{i,l}|\mathbf{e}_{i,l})$ if $\mathbf{z}$ is categorical; otherwise $\eta_{i,l} = -\varphi(\mathbf{e}_{i,l}; \mathbf{z}_{i,l})$. $\mathbf{e}_{i,l}$ denotes the i-th head feature map (either one of the $\mathbf{v}_{i,l}$, $\mathbf{a}_{i,l}$, or $\mathbf{o}_{i,l}$). **(3)** $VOTE$ means counting the '1's for each head based on the *0-1* votes in $\mathbf{\Gamma}_l$ and only keeping the heads with the highest counts[5]. After V2S, a finetuning is applied to adapt the pruned network.

GHT-PS compresses considerable parameters. In the case of two head groups, GHT-PS reduces 75% parameters for an attention layer and 32.1% for the entire model[6]. We will show that V2S removing non-PS heads does not sacrifice model performance. Instead, it brings accuracy gains in some cases and improves inference speed.

4   Experimental Setup

In this section, we detail the key architectural configurations. Further training, model, dataset & evaluation setups are detailed in A.1, A.2, & A.3. We follow the transformer of Vaswani et al. (2017) as a backbone architecture for all datasets and tasks in our experiments. Following Wu et al. (2019, 2020), for Machine Translation and Abstractive Summarization, we adopt the same 8-head encoder-decoder architecture with 6 layers for both encoder

and decoder, where the model dimension $d_{model} = 512$ and feed-forward dimension $d_f = 2048$. For LM, we adopt the 16-head decoder-only architecture with 16 layers, where the model dimension $d_{model} = 1024$ and feed-forward dimension $d_f = 4096$. The layer normalization is applied before the residual connection of each layer. The parameters of decoder input and output projections are shared. Our models are based on fairseq (Ott et al., 2019) implementations.

We perform the GCT as a metric learning task because it does not introduce additional projection layers when the shapes of similarity inputs are identical (Eq.4(B)), which makes GHT weight-lighter. In addition, it performs better in our experiments compared to the classification-based grouping.

5   Results and Analysis

5.1   Machine Translation

Ours vs. vanilla transformer. We first report results by comparing GHT and GHT-PS with the vanilla transformer (Vaswani et al., 2017) which is the backbone of our model. As shown in Table 1, the models are compared at different parameter levels[7]. GHT does not have weight reduction, keeping the same parameter size as the vanilla transformer (44M, the same setting as transformer base (Vaswani et al., 2017)). In contrast, GHT-PS is compressed to 30M parameters via V2S. For a fair comparison, we first compare GHT-PS with two lite architectures, Transformer-Lite1 and Transformer-Lite2, whose parameter numbers are 30M as well. Keeping other settings unchanged, the encoder and decoder of Transformer-Lite1 are reduced to 4 layers, respectively. Transformer-Lite2 reduces the model dimension d_{model} to 424, and d_f to 1696.

GHT and GHT-PS consistently and significantly outperform their backbone models at the same parameter level across all datasets. On average, the GHT surpasses 44M vanilla transformer by 3.8% in BLEU (Papineni et al., 2002); GHT-PS surpasses Lite1 and Lite2 by 4.9% and 4.4%, respectively. Although GHT-PS reduces 32.1% parameters, it significantly outperforms both 44M and 30M vanilla transformers, which is comparable to GHT on all datasets. It shows that V2S reduces the parameter size of the original transformer without sacrificing accuracy on MT. Efficiency is analyzed later.

---

[5]Besides voting, there is an alternative way to create the mask. Instead of using *0-1* number as a discrete voting unit, the group pattern scores can be added up to rank the head pattern explicitness. We find that the two ways perform similarly.

[6]The encoder-decoder arch in Vaswani et al. (2017).

[7]The parameters analyzed in this paper exclude the embedding layer since they vary a lot between different datasets when the vocabulary sizes are different.

Model	Param ↓	Inference Speed ↑	FLOPs ↓	BLEU ↑	IWSLT				WMT	
				de-en	it-en	en-de	en-it	en-fr	en-de	en-fr
Vanilla Transformer	44M	1012.1 sent/s	1996M	34.4	32.3	28.0	30.8	40.1	27.3	38.1
GHT (ours)	44M	1016.4 sent/s	1996M	**35.4**	**32.8**	**29.1**	**31.6**	**41.5**	**28.6**	**40.7**
Transformer-Lite1	30M	1175.4 sent/s	1549M	33.8	31.9	27.9	29.3	39.9	26.9	37.7
Transformer-Lite2	30M	1108.7 sent/s	1465M	34.0	32.2	28.2	29.5	40.0	26.7	37.8
GHT-PS (ours)	30M	1122.1 sent/s	1558M	**35.2**	**32.7**	**28.9**	**31.6**	**41.4**	**28.2**	**40.5**

Table 1: Benchmark with vanilla transformer (backbone) on IWSLT and WMT Machine Translation datasets, measured by BLEU. All improvements are statistically significant with $p < 0.05$ under t-test.

Model	Param ↓	Inference Speed ↑	FLOPs ↓	BLEU ↑	IWSLT				WMT	
				de-en	it-en	en-de	en-it	en-fr	en-de	en-fr
Cordonnier et al. (2020)	44M	416.6 sent/s	2054M	34.4	31.8	28.2	31.0	40.7	27.6	38.5
Li et al. (2018)	44M	1011.2 sent/s	1996M	34.7	31.8	28.5	30.7	40.7	27.3	39.3
GHT (ours)	44M	1016.4 sent/s	1996M	**35.4***	**32.8***	**29.1***	**31.6***	**41.5***	**28.6***	**40.7***
Voita et al. (2019)	30M	1099.1 sent/s	1558M	32.2	30.8	26.5	30.3	39.8	22.0	34.0
Li et al. (2021)	30M	1116.9 sent/s	1558M	33.2	31.3	27.5	30.0	39.7	20.5	33.6
Dynamic conv	30M	1050.2 sent/s	1615M	34.8	**32.7**	28.7	31.1	40.6	24.0	36.5
Lite Transformer	30M	1096.6 sent/s	1809M	33.3	31.4	27.5	29.8	39.4	24.9	37.4
GHT-PS (ours)	30M	1122.1 sent/s	1558M	**35.2***	32.7	**28.9***	**31.6***	**41.4***	**28.2***	**40.5***

Table 2: Benchmark with state-of-the-art MHA redundancy/parameter optimization baselines on IWSLT and WMT Machine Translation datasets at the same parameter level, measured by BLEU. * denotes the improvement is statistical significant with $p < 0.05$ under t-test.

Ours vs. efficient attention models. We compare GHT with two state-of-the-art (SOTA) MHA redundancy optimization baselines. Cordonnier et al. (2020) and Li et al. (2018) are respectively homogenization- and diversification-based methods. In addition, we compare GHT-PS with four SOTA baselines that made major contributions to attention parameter compression and redundancy optimization[8]. Voita et al. (2019) and Li et al. (2021) are significance-based pruning methods. Dynamic Conv (Wu et al., 2019) and Lite Transformer (Wu et al., 2020) modify the MHA arch to reduce parameters.

Table 2 shows that GHT outperforms all its baselines on all datasets, exceeding the strongest baseline by 2.9% in averaged BLEU scores. GHT-PS outperforms all its baselines on 6 out of 7 datasets, exceeding the strongest baseline by 4.4% on average. Model compression of the baselines may sacrifice performance (especially on large datasets, e.g., WMT en-de and en-fr), while GHT-PS is al-

most not affected by the parameter reduction, even surpassing GHT's baselines with 44M parameters.

Model	BLEU ↑				
	de-en	it-en	en-de	en-it	en-fr
GHT	**35.4**	**32.8**	**29.1**	**31.6**	**41.5**
- w/o Diversifying	34.7	31.8	28.5	30.7	40.7
- w/o Homologizing	34.3	32.0	28.2	30.9	40.2
GHT-PS	**35.2**	**32.7**	**28.9**	**31.6**	**41.4**
- w/o GCT	33.8	31.9	28.1	30.5	39.8
- w/o GC	34.0	32.0	28.4	31.0	40.2
- w/o HUDS	33.7	32.0	28.1	30.9	40.3
- w/o PS stay	33.6	31.7	27.9	30.7	40.2
- w/ stage 2 GC	33.2	31.8	28.1	30.8	40.3
- w/ stage 1& 2 GC	33.4	31.9	27.7	30.6	40.2

Table 3: Ablation study on IWSLT'14. The results are statistically significant with $p < 0.05$ under t-test.

Ablation Study. We evaluate the impacts of the features we choose for GHT and GHT-PS (Table 3). We first ablate the diversification/homogenization term of GCT (see Eq.4), which lowers the BLEU scores. Next, we show the importance of GCT for V2S. **w/o GCT** denotes that we directly perform V2S at the very beginning without GCT. **w/o GC**
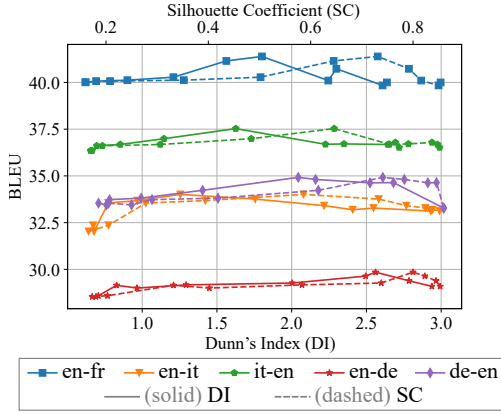
---

[8]Works optimizing parameters of transformer modules rather than the MHA are not compared. In addition, we do not compare to Michel et al. (2019) (post-pruning), because their method performs extremely bad when the parameter level is low, e.g., 30M (Li et al., 2021).

Figure 2: The final BLEU scores achieved by GHT (on IWSLT'14 dev set) first rise and then drop, as the final group patterns become more compact (indicated by the increasing SC and DI scores).

denotes that V2S is employed after normal training without Group Constrain (GC). Both ablation models yield lower BLEU, because they do not homogenize unnecessary heads and prepare them for pruning. Next, we validate the power of Pillars of Strength. **w/o HUDS** denotes we replace HUDS with randomly switching off heads after GCT; **w/o PS stay** denotes we keep random group members instead of the Pillars of Strength after GCT. We observe lower BLEU in **w/o HUDS** and **w/o PS stay**. Finally, we find that GC only needs to be added before V2S. We denote the training stages before and after V2S as stages 1 and 2. We compare the proposed Stage 1-based GHT-PS with models that perform GCT at Stage 2 (**w/ stage 2 GC**) and at both stages (**w/ stage 1& 2 GC**). BLEU scores of both ablation models decrease.

Effect of group compactness. We hypothesize that more compact group patterns bring performance gains to the GHT. Figure 2 shows the correlation between the compactness of the final group patterns and the final BLEU scores GHT achieved on 5 IWSLT'14 development sets when the GHT is fully converged in GCT. One data point corresponds to an independent run. We choose Silhouette Coefficient (SC) (Rousseeuw, 1987) and Dunn's Index (DI) (Bezdek and Pal, 1995) as the measurements of group pattern compactness, both of which increase as the intra-group samples become more similar and the inter-group ones become more separated. The SC and DI are computed with the FMs of GHA (§ 3.1) and controlled by tuning the α and β (Eq.4).
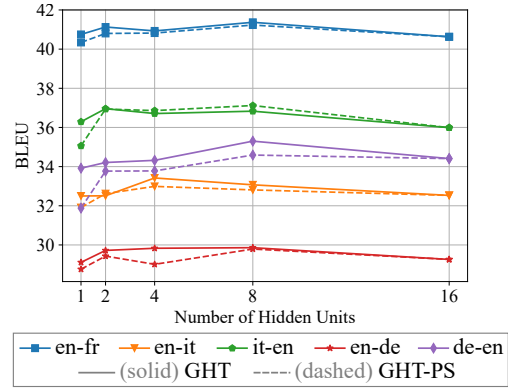

Figure 3: The BLEUs of GHT and GHT-PS by different numbers of hidden units (groups) on IWSLT'14 dev set.

Figure 2 shows that, within the normal range, the BLEU scores rise with higher SC/DI scores, which is in line with our assumption. The BLEUs start to drop after the peak as the SC/DI scores increase, because the very heavy group constraint prohibits the model from learning useful task-specific knowledge.

Effect of group number. Figure 3 shows the performance trends of 16-head GHT and GHT-PS by different numbers of group hidden units. For GHT, different datasets have different optimal hidden unit quantities, while a similar trend is observed. The optimal group number is between 2 and 8, which is in line with the claim that our group strategy is superior to sole homogenization (1 group) or diversification (16 groups) strategies. For GHT-PS, when the group number is larger than 1, it shows comparable performance to GHT on most datasets. This also verifies that non-PS heads can be switched off without sacrificing performance.
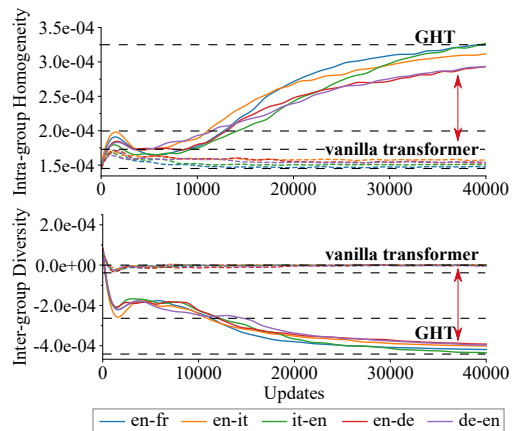

Figure 4: Intra-group homogeneity (upper) and inter-group diversity (lower) of GHT and vanilla transformer by training steps.

Model	Param ↓	Inference Speed ↑	FLOPs ↓	Rouge-1 ↑	Rouge-2 ↑	Rouge-L ↑
LSTM (Paulus et al., 2018)	-	-	-	38.30	14.81	35.49
CNN (Fan et al., 2018)	-	-	-	39.06	15.38	35.77
Light Conv (Wu et al., 2019)	86M	-	-	39.52	15.97	36.51
Dynamic Conv (Wu et al., 2019)	87M	-	-	39.84	16.25	36.73
Vanilla Transformer (Voita et al., 2019)	44M	208.77 sent/s	1996M	38.45	17.97	36.03
GHT (ours)	44M	208.77 sent/s	1996M	40.00	21.10	37.51
GHT-PS (ours)	**30M**	**257.62 sent/s**	**1558M**	**40.01**	**21.31**	**37.62**

Table 4: Abstractive Summarization results on CNN-DailyMail in terms of F1-Rouge and efficiency (parameter, inference speed, and FLOPs). All improvements are statistically significant with $p < 0.05$ under t-test.

Group pattern trends. Figure 4 shows the trends of intra-group homogeneity (given by the *1st* term of Eq.4(B)) and inter-group diversity (given by the *2nd* term of Eq.4(B)) of GHT and vanilla transformer in the training process on five IWSLT datasets. By training, GHT yields higher intra-group homogeneity and inter-group diversity absolute values, leading to more compact groups, while the vanilla transformer shows flattened trends. It shows that GCT can effectively homogenize intra-group heads and diversify inter-group heads.

Model	BLEU↑	Param↓	Infer Speed↑	FLOPs↓
Transformer base	25.8	44M	1016.4 sent/s	1996M
Transformer big	26.4	176M	707.1 sent/s	6635M
Lite Conv	26.6	166M	722.1 sent/s	6184M
Dynamic Conv	26.9	176M	710.0 sent/s	6315M
GHT-PS-LITE (ours)	26.6	**16M**	**1170.2** sent/s	**1181M**

Table 5: Efficiency comparison by parameter, inference speed (averaged on five runs), and FLOPs. All results are generated with beam size 5, batch size 256, max decoding length 10 on a NVIDIA Quadro RTX A6000.

Efficiency analysis. In Tables 1 and 2, the efficiency metrics are controlled to be identical. Our models with higher inference speed and lower FLOPs show efficiency by culling redundant parameters. We also compare the efficiency metrics by controlling BLEU scores. In Table 5, we select models from the works in Table 1 and 2 that are reported to achieve close BLEU scores on newstest2013 as the baselines. The GHT-PS-LITE is a light version of GHT-PS that has a d_f of 1024. Given BLEU ranges from 25.8 to 26.9, GHT-PS-LITE is much more efficient than the baselines. Noticeably, GHT-PS-LITE achieves 90.36% fewer parameters, 62.05% faster inference speed, and 80.90% fewer FLOPs against Lite Conv which yields the same BLEU as it.

5.2 Abstractive Summarization

We evaluate the ability of our model to process longer inputs via Abstractive Summarization on the CNN-DailyMail dataset. We take vanilla transformer as the backbone. Table 4 shows that both GHT and GHT-PS achieve higher F1-Rouge scores (Lin, 2004a) on this task. GHT-PS achieves 4.1% higher Rouge-1, 18.6% higher Rouge-2, and 4.4% higher Rouge-L against vanilla transformer. It also achieves 0.4% higher Rouge-1, 31.1% higher Rouge-2 and 2.4% higher Rouge-L against the best-performing baseline (Dynamic Conv). Meanwhile, GHT-PS only takes 68.18% parameters of the vanilla transformer and exhibits higher inference speed and fewer FLOPs.

Model	Param↓	Infer Spd↑	FLOPs↓	Valid↓	Test↓
S4	249M	-	-	19.69	20.95
BERT-L-CAS	395M	-	-	19.67	20.42
GPT-2 Large	762M	-	-	-	22.05
VT w/ AI	201M	9.9 tok/s	6106M	19.03	19.14
GHT (ours)	201M	9.9 tok/s	6106M	**18.57**	18.60
GHT-PS (ours)	**167M**	**19.0 tok/s**	**4573M**	18.58	**18.59**

Table 6: Language modeling results on WIKITEXT-103 by perplexity and efficiency (parameter, inference speed, and FLOPs). VT w/ AI denotes vanilla transformer with adaptive input. All improvements against the baselines are statistically significant with $p < 0.05$ under t-test.

5.3 Language Modeling

We evaluate LM performance on WIKITTEXT-103 dataset. The backbone is a decoder-only transformer with 16 layers and adaptive inputs (Baevski and Auli, 2019). We compare with the backbone model, as well as comparable SOTA LM models, including S4 (Gu et al., 2022), BERT-Large-CAS (Wang et al., 2019), and GPT-2 Large (Radford et al., 2018).

Table 6 shows that both GHT and GHT-PS achieve lower perplexity (Vajapeyam, 2014) than the baselines on both validation and test sets (2.9% and 9.0% less perplexity against the backbone and the best performing LM baseline, respectively). Meanwhile, GHT-PS achieves 16.92% parameter reduction, 2 times faster inference speed, and 75% FLOPs compared with the backbone.

6 Conclusion

In this paper, we assume that only focusing on the most representative and distinctive features with minimum resources may mitigate the redundancy and over-parameterization issues of MHA. Accordingly, we propose a divide-and-conquer strategy, including GCT and V2S to mitigate the issues. The improvements on three tasks and the extensive analysis verify our hypothesis and the effectiveness of our redundancy optimization methods. Our study may inspire future MHA design and training to achieve higher accuracy and efficiency.

Limitations

In this work, we evaluate the proposed models for NLP tasks only. However, tasks in other fields such as Computer Vision may present a very different input inductive bias, thus affecting the performance. Moreover, our models are trained from scratch, hence it is unknown whether the same divide-and-conquer strategy works for pre-trained models. We will study these limitations in the future to give a more extensive exploration.

Ethics Statement

This article follows the ACL Code of Ethics. The annotations are based on public datasets that do not contain private data. The algorithm we developed is an architectural optimization technique for improving model performance. To our best knowledge, there are no foreseeable potential risks to using this technique.

Acknowledgments

References

Karim Ahmed, Nitish Shirish Keskar, and Richard Socher. 2017. Weighted transformer network for machine translation. *CoRR*, abs/1711.02132.

Alexei Baevski and Michael Auli. 2019. Adaptive input representations for neural language modeling. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

James C Bezdek and Nikhil R Pal. 1995. Cluster validation with generalized dunn's indices. In *Proceedings 1995 second New Zealand international two-stream conference on artificial neural networks and expert systems*, pages 190–190. IEEE Computer Society.

Denny Britz, Anna Goldie, Minh-Thang Luong, and Quoc V. Le. 2017. Massive exploration of neural machine translation architectures. *CoRR*, abs/1703.03906.

Jean-Baptiste Cordonnier, Andreas Loukas, and Martin Jaggi. 2020. Multi-head attention: Collaborate instead of concatenate. *CoRR*, abs/2006.16362.

Hongyi Cui, Shohei Iida, Po-Hsuan Hung, Takehito Utsuro, and Masaaki Nagata. 2019. Mixed multi-head self-attention for neural machine translation. In *EMNLP-IJCNLP 2019, Hong Kong, November 4, 2019*, pages 206–214.

Yanzhuo Ding, Yang Liu, Huanbo Luan, and Maosong Sun. 2017. Visualizing and understanding neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1150–1159, Vancouver, Canada.

Sergey Edunov, Myle Ott, Michael Auli, David Grangier, and Marc'Aurelio Ranzato. 2018. Classical structured prediction losses for sequence to sequence learning. In *NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 355–364.

Angela Fan, David Grangier, and Michael Auli. 2018. Controllable abstractive summarization. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation, NMT@ACL 2018, Melbourne, Australia, July 20, 2018*, pages 45–54.

Jonathan Frankle and Michael Carbin. 2019. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Albert Gu, Karan Goel, and Christopher Ré. 2022. Efficiently modeling long sequences with structured state spaces. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Karl Moritz Hermann, Tomás Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *NIPs 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 1693–1701.

Jian Li, Zhaopeng Tu, Baosong Yang, Michael R. Lyu, and Tong Zhang. 2018. Multi-head attention with disagreement regularization. In *EMNLP 2018, Brussels, Belgium, October 31 - November 4, 2018*, pages 2897–2903.

Jian Li, Baosong Yang, Zi-Yi Dou, Xing Wang, Michael R. Lyu, and Zhaopeng Tu. 2019. Information aggregation for multi-head attention with routing-by-agreement. In *NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 3566–3575.

Jiaoda Li, Ryan Cotterell, and Mrinmaya Sachan. 2021. Differentiable subset pruning of transformer heads. *Trans. Assoc. Comput. Linguistics*, 9:1442–1459.

Chin-Yew Lin. 2004a. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Chin-Yew Lin. 2004b. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain.

Ilya Loshchilov and Frank Hutter. 2017. SGDR: stochastic gradient descent with warm restarts. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2017. Pointer sentinel mixture models. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Paul Michel, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 14014–14024.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA*, pages 311–318. ACL.

Razvan Pascanu, Tomás Mikolov, and Yoshua Bengio. 2013. On the difficulty of training recurrent neural networks. In *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*, volume 28 of *JMLR Workshop and Conference Proceedings*, pages 1310–1318. JMLR.org.

Romain Paulus, Caiming Xiong, and Richard Socher. 2018. A deep reinforced model for abstractive summarization. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

Gabriel Pereyra, George Tucker, Jan Chorowski, Lukasz Kaiser, and Geoffrey E. Hinton. 2017. Regularizing neural networks by penalizing confident output distributions. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*. OpenReview.net.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2018. Language models are unsupervised multitask learners.

Peter J Rousseeuw. 1987. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65.

Noam Shazeer, Zhenzhong Lan, Youlong Cheng, Nan Ding, and Le Hou. 2020. Talking-heads attention. *CoRR*, abs/2003.02436.

Sainbayar Sukhbaatar, Edouard Grave, Guillaume Lample, Hervé Jégou, and Armand Joulin. 2019. Augmenting self-attention with persistent memory. *CoRR*, abs/1907.01470.

Ilya Sutskever, James Martens, George E. Dahl, and Geoffrey E. Hinton. 2013. On the importance of initialization and momentum in deep learning. In *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*, volume 28 of *JMLR Workshop and Conference Proceedings*, pages 1139–1147. JMLR.org.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 2818–2826. IEEE Computer Society.

Sriram Vajapeyam. 2014. Understanding shannon's entropy metric for information. *CoRR*, abs/1405.2061.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 5797–5808.

Chenguang Wang, Mu Li, and Alexander J. Smola. 2019. Language models with transformers. *CoRR*, abs/1904.09408.

Felix Wu, Angela Fan, Alexei Baevski, Yann N. Dauphin, and Michael Auli. 2019. Pay less attention with lightweight and dynamic convolutions. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Zhanghao Wu, Zhijian Liu, Ji Lin, Yujun Lin, and Song Han. 2020. Lite transformer with long-short range attention. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

A Appendix

A.1 Trainig Settings

A.1.1 Machine Translation

We use Adam to optimize the MT models and set the $\beta_1 = 0.9, \beta_2 = 0.98$. We use the Inverse Square Root Schedule (Vaswani et al., 2017) where it first warms up for 4K steps until the learning rate reaches 5×10^{-4}, and then it exponentially decays the learning rate. We apply early stop as a termination condition. We apply a 0.3 dropout rate for all Machine Translation models. A weight decay of 10^{-4} is used for all IWSLT 2014 models, whereas for WMT models we use a weight decay of 0. We apply a 0.1 label smoothing (Szegedy et al., 2016; Pereyra et al., 2017) for the uniform prior distribution over the vocabulary.

A.1.2 Language Modeling

Following Baevski and Auli (2019), we use Nesterov's accelerated gradient method (Sutskever et al., 2013) with a momentum of 0.99. We clip the gradient norm if it exceeds 0.1 (Pascanu et al., 2013). The learning rate is linearly warmed up from 10^{-7} to 1 for 16K steps and then annealed using a cosine learning rate schedule (Loshchilov and Hutter, 2017) with multiple cycles. Each cycle doubles the number of updates than the previous cycle and we shrink the maximum and minimum learning rates by 0.75 compared to the previous cycle. The initial minimum learning rate is 10^{-4} and the maximum is 1. We apply 0.2 adaptive softmax dropout rate, 0.1 attention dropout rate, and 0.1 activation dropout rate.

A.1.3 Abstractive Summarization

We use the same training setup with IWSLT 2014 models. We apply 0.1 clip norm and 0.2 attention dropout. The model is warmed up for 10K updates.

A.2 Further Model Settings

Different α, β, and head feature maps ($\hat{\mathbf{V}}$, $\mathbf{A}$, and $\mathbf{O}$) are preferred for different datasets to achieve optimal performance. The Machine Translation configurations are detailed in Table 7; Language Modeling and Abstractive Summarization configurations are detailed in Table 8.

Note that $\varphi(\mathbf{v}_{i,l}, \mathbf{a}_{i,l}, \mathbf{o}_{i,l}; \mathbf{z}_{i,l}) = \tau_1 \varphi(\mathbf{v}_{i,l}; \mathbf{z}_{i,l}) + \tau_2 \varphi(\mathbf{a}_{i,l}; \mathbf{z}_{i,l}) + \tau_3 \varphi(\mathbf{o}_{i,l}; \mathbf{z}_{i,l})$, we set one of the $\{\tau_1, \tau_2, \tau_3\}$ to be 1, the other to be 0.

A.3 Datasets and Evaluation[9]

A.3.1 Efficiency Metrics settings

Inference speed. All inference speed results are generated with beam size 5, batch size 256, maximum decoding length 10 on a single NVIDIA Quadro RTX A6000.

FLOPs. We use the fvcore[10] to calculate the FLOPs, with a fixed input length of 30.

A.3.2 Machine Translation

To fully evaluate the effectiveness of our methods, we evaluate seven MT datasets of IWSLT'14 and WMT 2014 benchmarks. We closely follow the setup of Vaswani et al. (2017) for data preparation. WMT 2014 English-German dataset consists of about 4.5M sentence pairs. It is encoded with byte-pair encoding (Britz et al., 2017), having a shared source-target vocabulary of about 40K tokens. Following the standard setting (Vaswani et al., 2017), we validate on newstest2013 and test on newstest2014 for experiments on this dataset. The WMT 2014 English-French dataset consists of 36M sentence pairs and is encoded with a joint source-target BPE of about 43K vocabularies. Following the standard split, we validate on a concatenation of newstest2012 and newstest2013 and test on newstest2014. For IWSLT'14 German to English, IWSLT'14 English to German, IWSLT'14 English to French, IWSLT'14 English to Italian and IWSLT'14 Italian to English, we encode the sentence pairs with joint source-target BPE. Following Edunov et al. (2018), the validation set is randomly splited from the training set with a ratio of 1:23. The testset consists TED.tst2010, TED.tst2011, TED.tst2012 and TED.dev2010, TEDX.dev2012 for IWSLT'14 German to English, IWSLT'14 English to German, and IWSLT'14 English to French; the TEDX.dev2012 is replaced by TEDX.dev2014 for IWSLT'14 English to Italian and IWSLT'14 Italian to English.

For all Machine Translation datasets, we use detokenized BLEU. WMT 2014 English-German and WMT 2014 English-French are evaluated with a beam size 4 and length penalty 0.6; IWSLT'14 datasets are evaluated with a beam size 5 and without length penalty.

---

[9]For all three tasks, we follow the data pipeline of fairseq: https://github.com/facebookresearch/fairseq/blob/main/examples

[10]https://github.com/facebookresearch/fvcore

Model	IWSLT ($\alpha/\beta/FM$)					WMT ($\alpha/\beta/FM$)	
	de-en	it-en	en-de	en-it	en-fr	en-de	en-fr
GHT	0.7/0.5/$\hat{\mathbf{V}}$	0.3/0.5/$\hat{\mathbf{V}}$	0.3/0.1/$\mathbf{A}$	0.3/0.3/$\hat{\mathbf{V}}$	0.7/0.7/$\hat{\mathbf{V}}$	0.5/0.5/$\hat{\mathbf{V}}$	0.3/0.3/$\hat{\mathbf{V}}$
GHT-PS	0.5/0.7/$\mathbf{O}$	0.3/0.3/$\mathbf{A}$	0.3/0.7/$\mathbf{O}$	0.3/1/$\hat{\mathbf{V}}$	0.5/0.3/$\mathbf{A}$	0.5/0.5/$\hat{\mathbf{V}}$	0.3/0.3/$\hat{\mathbf{V}}$

Table 7: The configuration of α, β, and Feature Maps (FM, including $\hat{\mathbf{V}}$, $\mathbf{A}$, and $\mathbf{O}$) for GHT and GHT-PS on different Machine Translation datasets.

Model	$\alpha/\beta/FM$
GHT	0.5/0.5/$\hat{\mathbf{V}}$
GHT-PS	0.5/0.5/$\hat{\mathbf{V}}$

Table 8: The configuration of α, β, and Feature Maps (FM, including $\hat{\mathbf{V}}$, $\mathbf{A}$, and $\mathbf{O}$) for GHT and GHT-PS in Abstractive Summarization and Language Modeling.

A.3.3 Language Modeling

We evaluate LM on WIKITEXT-103 (Merity et al., 2017) which has about 100M tokens and a 260K BPE vocabulary. Following Baevski and Auli (2019), we use perplexity as an evaluation metric and a context window of 2047 at the inference stage.

A.3.4 Abstractive Summarization

We also evaluate on CNN-DailyMail (Hermann et al., 2015) for AS to test the ability of GHT in hard tasks with long inputs. The dataset comprises over 280K news articles paired with multi-sentence summaries. Following Wu et al. (2019), articles are truncated to 512 tokens and encoded with 50K BPE. We use F1-Rouge (Lin, 2004b) to evaluate the performance, including Rouge-1, Rouge-2 and Rouge-L.

A For every submission:

☑ A1. Did you describe the limitations of your work?
Section 7

☒ A2. Did you discuss any potential risks of your work?
The study is theoretical. It does not involve human in the experiment or involve ethic issues. It will not result in any potential risks.

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Section 1

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

Section 4; Appendix A.3

☑ B1. Did you cite the creators of artifacts you used?
Section 4; Appendix A.3

☒ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
The tools are for public use.

☒ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
No specific intended use is specified by the artifact creators.

☒ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
They does not contain any information that names or uniquely identifies individual people or offensive content.

☑ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Section 4; Appendix A.3

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Appendix A.3

C ☑ Did you run computational experiments?

Section 5

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Section 5; Appendix A.1

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Section 4, 5; Appendix A.1, 2, 3

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Section 5

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Appendix A.1, 2, 3

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
Not applicable. Left blank.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
Not applicable. Left blank.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
Not applicable. Left blank.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
Not applicable. Left blank.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
Not applicable. Left blank.