LEVERAGING REDUNDANCY IN ATTENTION WITH REUSE TRANSFORMERS

Anonymous authors
Paper under double-blind review

ABSTRACT

Pairwise dot product-based attention allows Transformers to exchange information between tokens in an input-dependent way, and is key to their success across diverse applications in language and vision. However, a typical Transformer model computes such pairwise attention scores repeatedly for the same sequence, in multiple heads in multiple layers. We systematically analyze the empirical similarity of these scores across heads and layers and find them to be considerably redundant, especially adjacent layers showing high similarity. Motivated by these findings, we propose a novel architecture that reuses attention scores computed in one layer in multiple subsequent layers. Experiments on a number of standard benchmarks show that reusing attention delivers performance equivalent to or better than standard transformers, while reducing both compute and memory usage.

1 INTRODUCTION

Multi-head dot product attention is a key component of Transformer models (Vaswani et al., 2017). Each head in each Transformer layer allows aggregating information across different groups of tokens in a sequence, with the aggregation pattern in each head being input-dependent through the use of dot products between the learned query and key projections of token representations. This approach has proven empirically to be remarkably successful, with Transformers delivering state-of-the-art performance across a number of applications in language, vision, and beyond. Given this success, a considerable amount of research has focused on analyzing and interpreting the attention scores computed by trained Transformer models (Serrano & Smith, 2019; Jain & Wallace, 2019; Wiegreffe & Pinter, 2019; Clark et al., 2019; Rogers et al., 2020) to gain insight into how attention is useful for inference in specific applications.

Our paper begins with analysis of a different aspect of attention—namely, the redundancy of attention scores computed by a Transformer model. Recently, Bhojanapalli et al. (2021) analyzed the variability in attention scores computed from different inputs sampled from a typical data distribution. In this work, we instead focus on the variability of scores across different heads in different layers that are computed for the same input. We perform a systematic analysis looking at the similarities in attention scores computed in different layers, after matching the closest heads for a given pair of layers. These similarities are computed for each typical input example on trained language and vision Transformer models, and then aggregated across a large set of examples from the corresponding training set.

Surprisingly, we find a high degree of similarity between different layers, with adjacent layers in a model exhibiting the most similarity. This suggests that although a standard Transformer model recomputes attention scores multiple times, much of this computation is redundant: the number of *distinct* attention scores used for aggregation is much smaller than the total number of heads across all layers of a typical model. We also show that this is not an inherent characteristic of the Transformer architecture (random models do not produce similar attention scores), and that therefore this redundancy results from the structure of the problem domain.

Motivated by this analysis, we propose the *Reuse Transformer*: a modified Transformer architecture which saves on redundant attention computation to deliver reductions in both compute and memory usage. As illustrated in Figure 1, a Reuse Transformer layer uses a mix of standard exact computation heads with "reuse heads", that borrow attention scores computed in previous layers instead of computing their own via dot products of query-key projections. This Reuse layer can be included in standard Transformer models with different configurations: either by reusing a fraction of the heads in most layers, or by reusing all heads in a fraction of the layers.

Since reuse heads do not use their own query and key projections, this leads to a reduction in the number of model parameters (and associated memory needed to save these parameters and their gradient moments). But unlike other approaches that also reduce parameters by sharing query-key projection parameters in multiple layers (Dehghani et al., 2018; Lan et al., 2019), the Reuse Transformer also saves on the actual attention computation—thereby reducing the computational cost during both training and inference, and the memory needed to store intermediate projections.

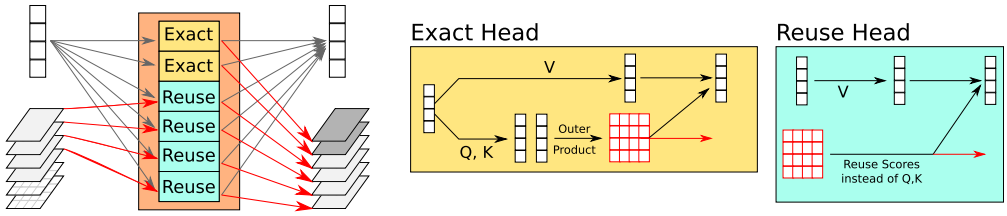

Figure 1: **Reuse Transformer.** We propose a modified architecture for Transformer layers that features a mix of standard "exact" and "reuse" attention heads, where the latter borrow attention scores computed in previous layers.

We evaluate Reuse Transformers in a variety of different settings: including on both language and vision tasks, and on encoder-only models like BERT (Devlin et al., 2018), encoder-decoder models like T5 (Raffel et al., 2020), and on Vision Transformers (ViT) (Dosovitskiy et al., 2021). Our experiments consider standard benchmarks where models are trained for specific tasks from scratch — Machine Translation (WMT 2018) (Bojar et al., 2018) and the Long Range Arena (LRA) (Tay et al., 2021) —as well as those involving pre-training and finetuning — GLUE (Wang et al., 2019b), SuperGlue (Wang et al., 2019a), SQuAD (Rajpurkar et al., 2016) and ImageNet (Deng et al., 2009). In addition to task performance, we also benchmark wall clock training time and memory usage to confirm that reusing attention translates readily to real world resource savings.

Through this extensive evaluation, we show that reusing attention scores saves compute and memory while yielding equivalent (and sometimes better) performance compared to standard Transformer models. We also show that when models with reuse are augmented with additional layers, to match baseline in terms of parameters, they perform better. Thus Reuse Transformers deliver a better performance and thus provide a better trade-off between resource usage and performance, indicating that attention score reuse is a useful inductive bias in Transformers.

In summary, our contributions in this paper are as follows.

- We systematically analyze the similarity of attention computed by different layers of standard Transformer models, and show that attention computed in different layers are very similar.
- We develop a novel architecture for Transformer models that reuses attention scores from earlier layers in subsequent layers, thus reducing compute and memory usage.
- We evaluate the proposed architecture on a wide variety of baseline models—BERT, T5, ViT— and benchmarks—GLUE, SuperGlue, SQuAD, ImageNet, WMT, and LRA—showing both actual savings in training time and memory usage, and at the same time, equivalent or better performance than standard Transformer models.

2 Background

2.1 Transformer

A Transformer encoder layer has two components: 1) a multi-head self-attention layer and 2) a tokenwise feed-forward (MLP) layer. A Transformer decoder layer in addition has a cross-attention layer, with attention between output of the encoder and the input to the decoder. The input to these models is a sequence of vectors that are usually embeddings of an input token sequence. We let $\boldsymbol{X} \in \mathbb{R}^{d \times n}$ denote the input embedding matrix of sequence length n with embedding size d. Note that we denote vectors and matrices with small (x) and capital ($\boldsymbol{X}$) bold letters respectively in this paper. The self-attention layer then updates these embeddings by computing pairwise dot product attention between the input embeddings. Both attention and feed-forward layers use layer normalization and skip connections.

The self-attention layer computes dot product based attention scores as

$$\boldsymbol{A}_{\boldsymbol{Z}} = \sigma\left(\boldsymbol{Z}^\top \boldsymbol{W}_Q^\top \boldsymbol{W}_K \boldsymbol{Z} / \sqrt{d}\right), \tag{1}$$

where $\boldsymbol{W}$ are trainable parameter matrices, and $\boldsymbol{Z}$ is the layer input. Here σ is a column-wise softmax operator. Projections $\boldsymbol{W}_Q \boldsymbol{Z}$ and $\boldsymbol{W}_K \boldsymbol{Z}$ are usually referred to as query and key projections respectively. The attention scores are then used to linearly combine token embeddings as follows: $\boldsymbol{Y} = \boldsymbol{A}_{\boldsymbol{Z}} \cdot \boldsymbol{Z}^\top \boldsymbol{W}_V^\top \cdot \boldsymbol{W}_0$, where $\boldsymbol{W}_V$ and $\boldsymbol{W}_0$ are referred to as value and output projections respectively. The output of the attention layer is fed into a tokenwise feedforward layer: $\boldsymbol{W}_2 \phi\left(\boldsymbol{W}_1 \boldsymbol{Y}^\top\right)$, with ϕ being an elementwise non-linear activation function such as a ReLU or GELU.

Multi-head attention involves multiple such trainable attention heads in a single layer, whose outputs are concatenated before multiplication with a common $\boldsymbol{W}_0$.

2.2 Related Work

Given the key role of the attention in Transformers, several works have analyzed the attention scores computed by these models. Many used probing tasks, such as syntax dependency and coreference resolution, to test the natural language understanding computed by attention layers (Clark et al., 2019; Hewitt & Manning, 2019; Voita et al., 2019; Rogers et al., 2020). Clark et al. (2019) found that heads in the initial layers of a BERT model tend to attend more broadly, while heads in later layers attend to specific tokens and perform better at language understanding tasks. Following these works, Raganato et al. (2020) explored replacing input dependent attention with a combination of fixed attention patterns and one learnable pattern per layer.

Another line of research Voita et al. (2019); Michel et al. (2019) showed that one can prune away many heads in an attention layer of a BERT model, after training. This pruning removes redundant attention heads, but leads to savings only during inference and not training. Through a systematic analysis of the similarity of attention scores computed by different layers, our work proposes a novel architecture that reduces redundant attention computation, leading to compute and memory savings both during training and inference.

It is well known that the cost of attention computation grows quadratically with input sequence length and poses challenges for training Transformers for long sequence length tasks. Several works address this issue by proposing efficient transformers that compute approximations to pairwise dot product attention—e.g. sparse (Child et al., 2019; Kitaev et al., 2020; Zaheer et al., 2020; Yun et al., 2020) and linear (Choromanski et al., 2020; Peng et al., 2020). These models reduce the computation and memory usage significantly but typically under-perform standard Transformers. We refer the reader to the survey by Tay et al. (2020) for a more detailed discussion. Our work focuses on reusing standard attention computation, and improves efficiency while maintaining performance.

Recently, Bhojanapalli et al. (2021) analyzed the variability of attention scores computed by a pre-trained model across different inputs. Finding them to be low-rank, they proposed a partial computation-based approach. While it reduced the cost of attention computation, also led to a drop in performance. Our work focuses on the similarity of attention scores computed in different layers for the same input, leading to a novel efficient architecture that does as well as or better than standard Transformers.

3 Attention Similarity Analysis

We now present an analysis of the similarity of attention scores computed in different layers by Transformer models trained for both language and vision tasks.

Preliminaries. Recall that each row of an attention score matrix is the output of a soft-max operator and thus lies on a probability simplex (all entries are non-negative and sum to one). Therefore, to compute the similarity between any pair of attention score matrices $\boldsymbol{A}$ and $\boldsymbol{A}'$, we use a metric based on the total variation (TV) distance between them as:

$$\mathcal{S}(\boldsymbol{A}, \boldsymbol{A}') = 1 - \frac{1}{n}\sum_{p=1}^{n} \text{TV}(\boldsymbol{A}[p,:], \boldsymbol{A}'[p,:]) = 1 - \frac{1}{n}\sum_{p=1}^{n} \frac{1}{2}\|\boldsymbol{A}[p,:] - \boldsymbol{A}'[p,:]\|_1. \tag{2}$$

Here n denotes the query sequence length (# rows of $\boldsymbol{A}$), $\|.\|_1$ the l_1 norm, and $\boldsymbol{A}[p,:]$ the pth row of the attention scores matrix $\boldsymbol{A}$—corresponding to scores for the pth query. Since attention scores form a probability distribution for each query, the total variation distance is always between 0 to 1. Hence the similarity also lies in $[0, 1]$.

Using equation 2 as our metric, we analyze the redundancy in attention scores computed by heads in different layers of a trained Transformer model on typical inputs from a dataset. Given T inputs, we pass each through a given model and let $\boldsymbol{A}^t_{l,h}$ denote the scores computed for the t^{th} example in the h^{th} head in layer l. Since there is no natural alignment between heads in different layers of a model, we define similarity from a reference layer l and head h to a target layer l', while finding the best target head in that layer. This similarity score $c_{(l,h),l'}$ is defined as:

$$c_{(l,h),l'} = \max_{h'} \frac{1}{T}\sum_{t=1}^{T} \mathcal{S}(\boldsymbol{A}^t_{l,h}, \boldsymbol{A}^t_{l',h'}). \tag{3}$$

Note that we compute the average similarity for each choice of the target h' before computing the best head, i.e., designating a common choice for the target head for all examples.

We use the above definitions of similarity to analyze scores from two BERT (Devlin et al., 2018) (12 layers and 12 heads, and 24 layers and 16 heads) and one Vision ViT (Dosovitskiy et al., 2021) (12 layers and 12 heads) model (please refer to Section 5 for training details). The similarities are computed for scores from 10k examples from the Wikipedia and ImageNet datasets for the BERT and ViT models respectively.

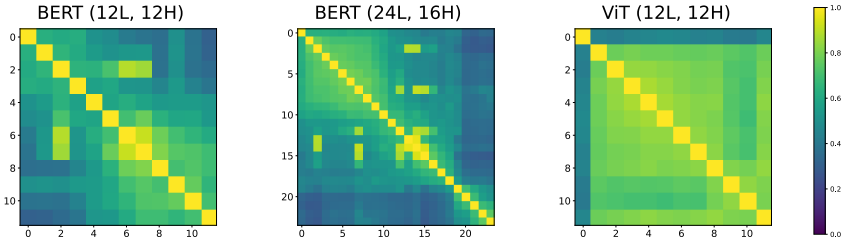

Figure 2: **All Pairs Similarity.** For all pairs of layers (indexed in x- and y- axis), we visualize the similarity between the best pairs of heads in that pair of layers. We show similarity scores for two BERT models on the Wikipedia dataset and one ViT model on the ImageNet dataset, using scores averaged over 10k examples in all cases.

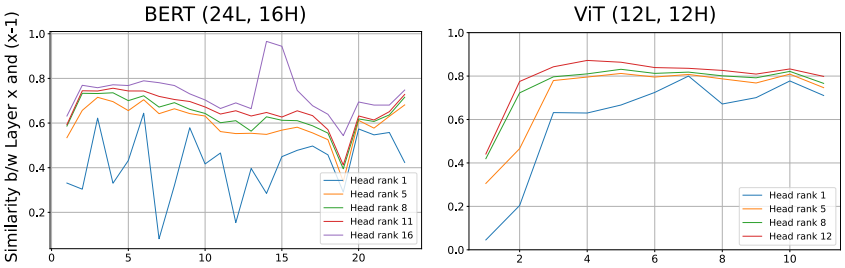

Figure 3: **Sequential Similarity**. We report similairty in adjacent layers for different heads in a (left) 24 layer BERT and (right) 12 layer ViT model. For each layer, we compute similarity for each head with its closest matching head in the previous layer. We then rank heads from lowest (rank 1) to highest similarity, and plot these across layers.

Results. We begin by looking at the similarity between all pairs of layers for all three models in Figure 2, looking at the best matched head in this case—for every pair of layers (l, l'), we visualize $\max_h c_{(l,h),l'}$. We find a surprisingly high degree of similarity in attention scores computed in different layers for all the three models. In particular, we see that the similarity between adjacent layers is especially high. Since we consider the best head here, these results imply that there is at least one head that is redundant in all pairs of layers that have high similarity. We also note that the ViT model shows a greater degree of similarity than the BERT models.

For a deeper understanding of similarity across different heads, we restrict ourselves to adjacent layers (where we find similarity to be the highest in Figure 2), and plot the similarity for different heads in the source layer, not just the best head. We rank heads from lowest (rank 1) to highest similarity, and plot these between successive layers in Figure 3 for the 24-layer BERT and 12-layer ViT models. We again notice that, for BERT, the best head (rank 16) has a high similarity of around 0.8 between successive layers. While the worst head has a low similarity (< 0.5), even rank 5 head has a high similarity of around 0.6. This suggests that the majority of the heads in a layer compute similar attention as in the previous layer. Some heads do seem to compute novel attention scores that are different from earlier layers.

Role of the Problem Domain. A natural question to ask is whether the similarity we observed above is inherent to the Transformer architecture. We evaluate this question both theoretically and empirically. First, we prove that a pair of random heads is expected to produce significantly different attention scores, with low similarity.

Lemma 1 (Random attention has less similarity)**.** *Let the entries of the query and key projection matrices of two heads* $\boldsymbol{W}_{q1}, \boldsymbol{W}_{k1}, \boldsymbol{W}_{q2}$ *and* $\boldsymbol{W}_{k2}$ *be i.i.d randomly with a zero mean distribution. Let* $\tilde{\boldsymbol{A}}_1 = \boldsymbol{X}^T * \boldsymbol{W}_{q1}^T * \boldsymbol{W}_{k1} * \boldsymbol{X}$ *and* $\tilde{\boldsymbol{A}}_2 = \boldsymbol{X}^T * \boldsymbol{W}_{q2}^T * \boldsymbol{W}_{k2} * \boldsymbol{X}$ *be the pre-softmax attention scores computed using these matrices for any fixed* $\boldsymbol{X}$*. Then,*

$$\mathbb{E}[(\tilde{\boldsymbol{A}}_1 - \tilde{\boldsymbol{A}}_2)_{ij}^2] = 2\mathbb{E}[(\tilde{\boldsymbol{A}}_1)_{ij}^2] = 2\mathbb{E}[(\tilde{\boldsymbol{A}}_2)_{ij}^2].$$

This lemma, proved in §C of the appendix, implies that even if the token embeddings input to attention heads in two different layers were the same, we would expect to see a large difference in their attention scores.

Therefore, the similarity that we observe is because the training process converges to models that find it beneficial, for the given inference task, to recompute similar attention scores in subsequent layers. We verify this empirically as well, by comparing a 6 layer BERT model trained on random data to the standard one trained on Wikipedia in Figure 4. We evaluate the similarity scores computed by both models on both Wikipedia data (1st and 2nd column) and on random data (3rd and 4th column). We notice that the degree of similarity depends more on the model, i.e., the data it was

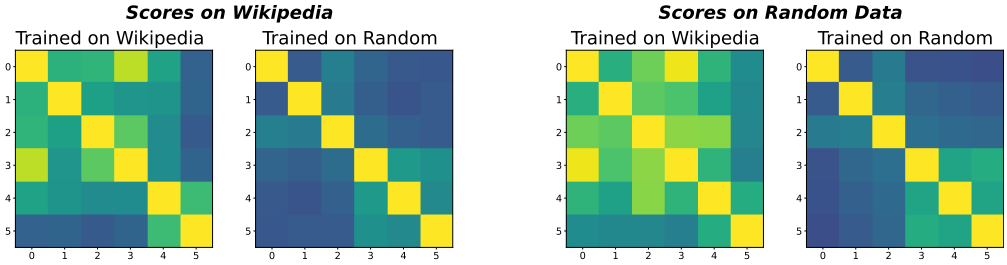

Figure 4: **Role of problem domain.** We visualize the all pairs best head attention similarity scores for (6 layer) Transformer models trained on Wikipedia vs random data. In the first two columns, we compare similarity in attention scores for both models computed over 10k examples from Wikipedia, and in the last two columns over random data. We notice that regardless of which examples are used at test time to compute similarities, the model trained on natural rather than random data exhibits higher similarity.

trained on, than on the data on which the attention scores are being computed. This suggests that the redundancy in attention is a result of training the model for a natural language problem domain.

4 Reuse Transformer

Attention score computation is one of the more expensive operations in a Transformer layer, since it scales quadratically with sequence length. Based on our observation that attention scores computed in different layers of a Transformer model are often redundant, we propose a novel Transformer layer which reuses attention scores computed in previous layers rather than computing them through a dot product of query-key projections.

Reuse Multihead Attention. We now describe the Transformer model with our proposed reuse layers in Algorithm 1. The model is also visually illustrated in Figure 1. In addition to the standard specification of the total number of layers L and heads per layer H, the Reuse Transformer architecture also depends on the choice on the number of layers in which to employ reuse $P < L$, and the number of heads $K < H$ to be reused in every such layer. Note that Alg. 1 describes only the modified attention computation. The remaining steps—query-key projection for exact heads, value projections for all heads and combination with attention scores, feed-forward layers, etc.—are the same as in standard Transformers as described in § 2.1. Further note that this mechanism can also be used for cross-attention layer in the decoder models (Vaswani et al., 2017), by interpreting the exact and reuse attention scores in Alg. 1 to refer to the ones from the cross-attention layer.

Algorithm 1 Reuse MultiHead Attention

Given: # Layers L and heads H, and reuse layers $P < L$ and reuse heads $K \leq H$.
Layer 1: Compute attention scores $\boldsymbol{A}_{1,h}, \forall h \in [1, H]$. ▷ First layer is always exact.
Set Reuse attention scores $\boldsymbol{R}_1 = [\boldsymbol{A}_{1,h\in[1,H]}]$.
for $l = 2, \cdots, P+1$ **do** ▷ Reuse K heads in the next P layers.
    Compute attention scores only for $H - K$ heads $\boldsymbol{A}_{l,h\in[1,H-K]}$.
    Reuse attention scores for K heads $\boldsymbol{A}_{l,h\in[H-K+1,H]} = \boldsymbol{R}_{l-1,h\in[1,K]}$.
    Set Reuse attention scores $\boldsymbol{R}_l = [\boldsymbol{A}_{l,h\in[1,H-K]}, \boldsymbol{A}_{l,h\in[H-K+1,H]}]$.
end for
for $l = P+2, \cdots, L$ **do** ▷ Remaining $L - P - 1$ layers are exact.
    Compute attention scores for all heads $\boldsymbol{A}_{l,h}, \forall h \in [H]$.
end for

We begin by noting from Alg. 1 that attention is computed exactly for all heads in the first layer—since the first layer does not have access to any previous attention scores to reuse—and that we choose to use reuse layers in the first P layers after the first one. Moreover, although we reuse attention scores, we still carry out the remaining steps of a self-attention layer, namely value projection and combining these values weighted by the attention scores. For ablation against variants that reuse different layers and skip attention computation entirely, please refer to § B.1 in the appendix.

Every layer l in our model outputs both the updated token embeddings as well as a set of H attention scores $\boldsymbol{R}_l$. In every layer that is reused, we compute exact attention scores for the first $H - K$ heads. For the remaining K heads, we copy over scores from the first K heads from $\boldsymbol{R}_{l-1}$. The layer then passes on this new set of H attention scores $\boldsymbol{R}_l$ as

input to layer $(l+1)$. Note that since we stack the new exact attention heads to the top of $\boldsymbol{R}_l$ and retain scores from the first few heads of $\boldsymbol{R}_{l-1}$, every reuse layer effectively uses the set of most recently computed H scores.

Reusing attention scores in this way is a reasonable approximation given our empirical observation that attention scores in trained models are redundant. We analyze this formally in the context of a simplified two-layer linear Transformer model (similar to (Levine et al., 2020)), and show that in the presence of high attention similarity, reusing attention generates similar outputs. We present a informal version of our result here, and include the more complete statement and proof in § C in the appendix.

Lemma 2 (Informal version of Lemma 3)**.** *Let attention computed by the two layers be ϵ apart for all inputs $\boldsymbol{X}$, i.e $\|\boldsymbol{A}_1 - \boldsymbol{A}_2\| \leq \epsilon$. Then, there exists a reuse Transformer, such that the error in the outputs scales as $O(\epsilon)$.*

Configurations. The parameters P, K control how much attention computation is reused in the Transformer model, reducing the number of attention computations from $L * H$ heads to $(L * H - P * K)$. Note that for a given reuse budget $(P * K)$ there are many ways of choosing parameters P and K. We consider two different settings for our experiments.

- **Partial layer Reuse.** In this setting, we always set the number of reuse layers P to be $L-2$ and vary K, such that all heads of the first and last layer compute attention scores, and rest of the layers reuse K heads. In this architecture, every layer has atleast one head (when $K < H$) that computes attention scores.
- **Full layer Reuse.** In this setting we always set K to be H and vary P, i.e., attention is not computed in P layers of the model and is reused from the earlier layer. Note that we need to again set the first layer exact to be able to reuse the attention scores in the following layers.

Computational Complexity. Reusing the attention scores reduces both memory and computation of attention layer as heads that reuse attention scores do not have to compute the query and key projections as well. Thus the model reduces the attention score computation cost in each layer from $H \cdot n^2$ to $(H-K) \cdot n^2$, for input sequence length n with K heads being reused. This reduces the overall computational complexity of the multihead attention layer from $4 \cdot d^2 \cdot n + 2 \cdot d \cdot n^2$ to $(1 - \frac{K}{2H}) \cdot \left[4 \cdot d^2 \cdot n + 2 \cdot d \cdot n^2\right]$. Similarly this reduces the number of parameters from $4 \cdot d^2$ to $(1 - \frac{K}{2H}) \cdot 4 \cdot d^2$.

5 Experiments

In this section we present experiments to show the advantage of reusing attention scores in reducing computational costs while matching or improving the performance of Transformers. We consider two different settings for our experiments 1) pre-training followed by finetuning, 2) training from scratch. For the first setting we consider BERT (Devlin et al., 2018), T5 (Raffel et al., 2020) and ViT (Dosovitskiy et al., 2021) models. For the second setting we consider Machine Translation on WMT2018 (Bojar et al., 2018) and the Long Range Arena (LRA) benchmark developed to test Transformers on long sequence length tasks in multiple domains (Tay et al., 2021). Note that our setup includes encoder-only as well as encoder-decoder models. In both these settings we will see that reusing attention not only reduces computation but can also improves performance in some settings. Moreover, we will show that, one can always achieve better performance by matching the number of parameters of the reuse Transformer with the standard Transformer. We use the publicly available implementations, with the **same hyperparameters** for all the experiments for a task, and report them in detail in Appendix (§ A).

5.1 Experimental Setup

We first describe our experimental setup for all the tasks.

BERT. BERT models are Transformers pre-trained with Masked Language Modeling (MLM) objective on Wikipedia and Books datasets (Devlin et al., 2018). These are encoder only models with bi-directional attention across input tokens. We follow a similar pre-training and finetuning recipe as BERT. We report the finetuning results on the MNLI (Williams et al., 2018) and SQuAD V1.1/V2.0 (Rajpurkar et al., 2016) tasks in Table 1.

We consider two models i.e., $\text{BERT}_{\text{BASE}}$ with 12 layers, 12 heads and $\text{BERT}_{\text{LARGE}}$ with 24 layers, 16 heads. We consider two different settings for reusing attention scores - 1) partial layer - reusing $6/8$ heads per layer , 2) full layer - reusing scores in the beginning $6/12$ layers (Algorithm 1) in the $\text{BERT}_{\text{BASE}}$ and $\text{BERT}_{\text{LARGE}}$ models respectively. Note that for both the settings the first layer computes attention scores for all heads as described in Section 4. Further we find it useful to have the last layer also compute attention scores for all heads in the first setting. In addition we also consider reuse transformers with more layers (Reuse 13L and 26L), that match the parameters of the baseline models.

T5. T5 models, unlike BERT, are encoder-decoder Transformer models that are pretrained on a similar objective as BERT models but on the C4 dataset (Raffel et al., 2020). With a unified text to text framework, these models are shown

to generalize easily to a variety of finetuning tasks. We consider the finetuning tasks from GLUE (Wang et al., 2019b) and SuperGLUE (Wang et al., 2019a) benchmarks for our experiments. Similar to the BERT experiments we consider models of two different sizes - T5_{BASE} and T5_{LARGE} with 12 and 24 layers respectively per encoder and decoder. Note that we reuse the attention in both the encoder and decoder, including in the cross attention layer.

ViT. We next consider experiments with Vision Transformers (Dosovitskiy et al., 2021). These models are pre-trained on JFT-300M (Sun et al., 2017), a dataset with 300M images and around 375M labels, and are finetuned on ImageNet. We consider the ViT-Base model, a 12 layer Transformer model with 12 heads per layer. For finetuning we use a resolution of 384x384 and a patch size of 16 resulting in a sequence length of 576 tokens per image. We consider again both partial and full layer reuse settings.

Machine Translation. We now consider the experiments on Machine Translation. For training we use two language pairs in both directions from WMT 2018 (Bojar et al., 2018) - English, German (en-de and de-en), and English, Czech (en-cs and cs-en). We report the test performance on Newstest 2018 datasets computed using the SacreBLEU (Post, 2018). We use a baseline Transformer model with 6 layers per encoder and decoder with 8 attention heads per layer following Vaswani et al. (2017).

LRA. Finally, we consider the Long Range Arena benchmark (Tay et al., 2021) developed to test Transformers on long sequence length tasks with input sequence lengths ranging from 1k to 8k. It consists of five tasks covering logical, textual and vision data. All tasks use a 4 layer Transformer model with the exception of Image classification task, that uses a 1 layer model. Hence, we do not report results on this task. We consider three different reuse settings.

5.2 Results

We now present our experimental results.

Table 1: **BERT.** Median pretraining and finetuning performance of BERT models from 3 independent runs. We highlight reuse Transformer cells that match (upto standard deviation) or improve over the baseline. We notice that reusing attention scores results in similar finetuning performance while reducing the computational requirement. We notice that both forms of reusing attention results in similar performance. We also see that matching the baseline in number of parameters (by increasing number of layers from 12/24 to 13/26 for the BASE/LARGE models) results in better performance showing that reusing attention results in better performance scaling.

Model	**Reuse** heads (K)	**Reuse** layers (P)	**Params**	**FLOPS**	**MLM** Acc	**MNLI** Acc	**SQuAD V1.1** F1	**SQuAD V2.0** F1
$\text{BERT}_{\text{BASE}}$ - 12H,12L	-	-	1.0	1.0	68.91	85.32 ± 0.05	89.93 ± 0.12	79.53 ± 0.71
Reuse	6	10	0.95	0.92	68.94	84.79 ± 0.31	89.29 ± 0.27	78.49 ± 0.01
Reuse	12	6	0.94	0.9	68.97	85.27 ± 0.2	89.61 ± 0.06	78.74 ± 0.25
Reuse 13L	12	6	1.0	0.98	69.32	85.38 ± 0.32	90.26 ± 0.32	79.55 ± 0.62
$\text{BERT}_{\text{LARGE}}$ - 16H,24L	-	-	1.0	1.0	73.76	87.97 ± 0.32	91.87 ± 0.17	82.41 ± 0.38
Reuse	8	22	0.93	0.91	73.93	87.6 ± 0.26	91.78 ± 0.12	83.58 ± 0.67
Reuse	16	12	0.92	0.9	73.64	87.75 ± 0.38	91.92 ± 0.33	82.56 ± 0.14
Reuse 26L	16	12	1.0	0.99	74.13	88.14 ± 0.10	92.37 ± 0.10	83.38 ± 0.35

BERT. We report the median finetuning results over 3 independent runs in Table 1. In addition to pretraining and finetuning metrics, we also report the relative computation cost required for each model. We first notice that models that reuse attention scores require lesser computation and parameters. Interestingly reusing attention scores results in similar performance on both pre-training and finetuning tasks. Further partial and full layer reuse have similar average performance. Finally the 13 and 26 layer reuse models achieve the best performance while having the same number of parameters as the $\text{BERT}_{\text{BASE}}$ and $\text{BERT}_{\text{LARGE}}$ models respectively.

T5. We again report the median finetuning results over 3 independent runs on the GLUE and SuperGLUE benchmarks in Table 2. We report the average scores across all tasks. We first notice that, interestingly, reusing attention results in better performance over T5_{BASE} and matches performance of T5_{LARGE}, while saving on computation resources. This is observed for both forms of reusing attention. In addition to model parameters and FLOPS we also report the training wall clock time, in terms of steps per second for these models on TPUv3 with 32 chips for T5_{BASE} and 64 chips T5_{LARGE} respectively. We notice that reusing attention leads to substantial speedups while improving the performance. Finally we also consider 13/26 layer reuse models that have the same number of parameters as the baseline, and achieve the best performance.

Table 2: **T5.** Median performance of T5 models on the GLUE and SuperGLUE finetuning tasks over 3 independent runs. We highlight reuse Transformer cells that match or improve over the baseline. We notice that reusing attention scores leads to an improvement in performance for both base and large models. We also report the relative number of parameters and compute required for all the models. We notice that reusing attention also leads to an improvement in compute and reduction of model parameters.

Model	**Reuse** heads (K)	**Reuse** layers (P)	**Params**	**FLOPS**	**Steps/Sec**	**Glue** Average	**SuperGlue** Average
$T5_{BASE}$ - 12H,12L	-	-	1.0	1.0	12.85	84.28±0.15	71.32±0.28
Reuse	6	10	0.92	0.9	13.7	84.66±0.36	71.09±0.15
Reuse	12	6	0.9	0.88	15.18	84.82±0.17	71.76±0.11
Reuse 13L	12	6	1.0	0.96	14.12	84.47±0.13	72.14±0.55
$T5_{LARGE}$ - 16H,24L	-	-	1.0	1.0	5.68	85.28±0.28	74.16±1.56
Reuse	16	12	0.9	0.88	6.65	85.52±0.23	73.44±0.23
Reuse 26L	16	12	1.0	0.92	6.16	85.65±0.23	73.81±1.41

Table 3: **ViT.** ImageNet finetuning accuracy of ViT models pretrained on JFT-300M.

Model	**Reuse** heads (K)	**Reuse** layers (P)	**Params**	**FLOPS**	**Steps/Sec**	**ImageNet Top-1** Acc
ViT - 12H,12L	-	-	1.0	1.0	4.02	84.51±0.13
Reuse	6	10	0.93	0.92	4.28	84.29±0.06
Reuse	12	6	0.92	0.90	4.51	83.55±0.18
Reuse 13L	6	10	1.01	0.99	3.96	84.69±0.24

ViT. We present the results in Table 3. We notice again that reusing attention leads to similar performance as baseline with reduction in parameters and compute. Interestingly, partial reuse has better performance over full layer reuse. Further increasing the reuse model size (13L), to match the baseline size, results in an improved performance.

Table 4: **Machine Translation.** Median translation performance (BLEU scores) on Newstest2018 dataset. We notice that reusing attention results in similar performance while saving on the computation and parameters.

Model	**Reuse** heads (K)	**Reuse** layers (P)	**Params**	**FLOPS**	**EN-DE**	**DE-EN**	**EN-CS**	**CS-EN**
Baseline - 8H,6L	-	-	1.0	1.0	39.22 ± 0.24	38.37 ± 0.09	18.27 ± 0.13	23.18 ± 0.12
Reuse	2	4	0.91	0.96	39.20 ± 0.17	38.41 ± 0.21	18.53 ± 0.16	23.49 ± 0.17
Reuse	4	4	0.90	0.92	39.05 ± 0.22	38.20 ± 0.14	18.16 ± 0.19	23.32 ± 0.11

Machine Translation. We report the median BLEU scores in Table 4. We see again that reusing attention scores leads to similar performance as the baseline while saving on computational resources.

LRA. We notice in Table 5 that reusing attention scores leads to better performance for the tasks in LRA benchmark. Interestingly we see better performance when more heads/layers reuse attention scores. We attribute this to the regularization effect of reusing the attention scores. Note that since some tasks only use a model with only 4 attention heads, the results for reusing 8 attention heads are the same as the ones with 4 attention heads.

5.3 Ablation

In this section we present ablation results with the reuse Transformers (Figure 5). We first train the reuse Transformer that reuses 6/12 attention layers ($P = 6, K = 12$) for the same amount of time as the baseline and notice significant performance gains ($> 1\%$) over the $T5_{BASE}$ model on the SuperGLUE benchmark (left). We next present an ablation varying the number of reuse heads (K), with $P = 10$, for the 12 layer $T5_{BASE}$ (middle) and ViT (right) models. We notice that reusing a few attention heads improves performance. Interestingly, even when reusing all the heads, the drop in performance is not much for the ViT models.

Table 5: **LRA.** Performance of reuse attention models on the Long Range Arena benchmark. We notice that reusing attention leads to sizeable performance improvement while reducing the computational cost.

Model	**Reuse** heads (K)	**Reuse** layers (P)	**Avg** Acc	**ListOps**	**Text**	**Retrieval**	**Path**
Baseline - 8H,4L	-	-	58.84±0.41	36.65	63.29	58.91	76.49
Reuse	4	2	59.47±0.71	37.4	64.07	58.08	78.33
Reuse	8	2	60.17±0.46	40.2	64.07	58.08	78.33
Reuse	8	3	61.26±1.48	39.95	63.95	58.72	82.4

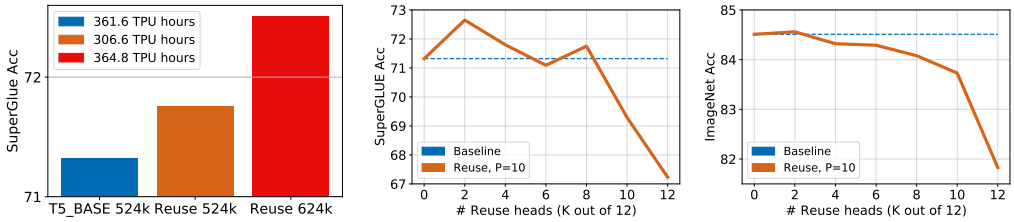

Figure 5: **Ablation.** Left - Training reuse models for the same amount of time as the T5_{BASE} results in $> 1\%$ improvement on the SuperGLUE benchmark. Middle, Right - Performance of the T5_{BASE} and ViT models with varying the number of reuse heads on SuperGLUE and ImageNet respectively. We see matching/improved performance with reusing a few heads. Interestingly performance doesn't drop much for ViT models even when $K = 12$, inline with the high attention similarity among its layers.

Table 6: **Computational savings.** Performance benchmark on Text classification task in the LRA benchmark for input sequence lengths from 1k to 4k. We see that reusing attention scores leads to significant gains both in increased steps per second and reduced memory usage. We indicate the percentage improvement for the 4k input sequence length setting in the parenthesis.

Model	**Reuse** heads (K)	**Reuse** layers (P)	**Steps/Second** 1K	2K	3K	4K	**Peak Mem Usage (GB)** 1K	2K	3K	4K
Baseline - 8H,4L	-	-	7.80	5.47	3.86	2.84	0.7	2	4.3	7.5
Reuse	4	2	8.25	5.83	4.37	3.24(13.9%)	0.57	1.7	3.6	6.2(17.3%)
Reuse	4	3	8.52	5.72	4.41	3.28(15.4%)	0.47	1.62	3.52	6.17(17.7%)

5.4 Computational savings

In this section we present comparison of models compute (steps per second) and memory usage during training for different sequence lengths on the text classification task in the LRA benchmark in Table 6. We use a batch size of 32 and train on TPUv3 with 16 chips. The baseline model has 4 layers with 4 attention heads per layer. We consider two different reuse settings.Both the settings show improvement in both speed and memory usage over the baseline with the best model achieving 15.4% speedup and 17.7% reduction in memory usage.

6 Conclusion

In this paper, we analyzed the similarity in attention scores computed at different layers of a Transformer model, and discovered them to be substantially redundant. Based on this observation, we proposed a new approach for reducing the compute and memory usage of Transformer models, both during training and inference, by reusing attention scores across layers. As our extensive experiments showed, this improved efficiency was borne out in terms of actual training speed and memory usage, and came with performance equivalent to or better than standard Transformers. More broadly, our work shows that developing a better understanding of the empirical behavior of state-of-the-art models can yield real dividends—in this case, in the form of a new architecture with improved performance-efficiency trade-offs.

7 Reproducibility Statement

All the experiments in this paper are done with models that have publicly available code. We have used the same default hyperparameters for all our experiments for a given task. We further list them in Appendix A. Proposed reuse attention (Algorithm 1) is a simple modification to standard attention layers.

References

Srinadh Bhojanapalli, Ayan Chakrabarti, Himanshu Jain, Sanjiv Kumar, Michal Lukasik, and Andreas Veit. Eigen analysis of self-attention and its reconstruction from partial computation. *arXiv preprint arXiv:2106.08823*, 2021.

Ondřej Bojar, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, and Christof Monz. Findings of the 2018 conference on machine translation (wmt18). In *Proceedings of the Third Conference on Machine Translation*, volume 2, pp. 272–307, 2018.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse Transformers. *arXiv preprint arXiv:1904.10509*, 2019.

Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, et al. Rethinking attention with performers. *arXiv preprint arXiv:2009.14794*, 2020.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D Manning. What does BERT look at? an analysis of BERT's attention. *arXiv preprint arXiv:1906.04341*, 2019.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Łukasz Kaiser. Universal Transformers. *arXiv preprint arXiv:1807.03819*, 2018.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional Transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations, ICLR*, 2021.

John Hewitt and Christopher D Manning. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4129–4138, 2019.

Sarthak Jain and Byron C Wallace. Attention is not explanation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 3543–3556, 2019.

Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. Reformer: The efficient Transformer. *arXiv preprint arXiv:2001.04451*, 2020.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*, 2019.

Yoav Levine, Noam Wies, Or Sharir, Hofit Bata, and Amnon Shashua. Limits to depth efficiencies of self-attention. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 22640–22651. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/ff4dfdf5904e920ce52b48c1cef97829-Paper.pdf.

Paul Michel, Omer Levy, and Graham Neubig. Are sixteen heads really better than one? In H. Wallach, H. Larochelle, A. Beygelzimer, F. dÁlché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/2c601ad9d2ff9bc8b282670cdd54f69f-Paper.pdf.

Hao Peng, Nikolaos Pappas, Dani Yogatama, Roy Schwartz, Noah Smith, and Lingpeng Kong. Random feature attention. In *International Conference on Learning Representations*, 2020.

Matt Post. A call for clarity in reporting bleu scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pp. 186–191, 2018.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67, 2020.

Alessandro Raganato, Yves Scherrer, and Jörg Tiedemann. Fixed encoder self-attention patterns in transformer-based machine translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pp. 556–568, 2020.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 2383–2392, 2016.

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. A primer in bertology: What we know about how bert works. *Transactions of the Association for Computational Linguistics*, 8:842–866, 2020.

Sofia Serrano and Noah A Smith. Is attention interpretable? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 2931–2951, 2019.

Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *Proceedings of the IEEE international conference on computer vision*, pp. 843–852, 2017.

Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. Efficient transformers: A survey. *arXiv preprint arXiv:2009.06732*, 2020.

Yi Tay, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler. Long range arena : A benchmark for efficient transformers. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=qVyeW-grC2k.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pp. 5998–6008, 2017.

Ashish Vaswani, Samy Bengio, Eugene Brevdo, Francois Chollet, Aidan N Gomez, Stephan Gouws, Llion Jones, Łukasz Kaiser, Nal Kalchbrenner, Niki Parmar, et al. Tensor2tensor for neural machine translation. *arXiv preprint arXiv:1803.07416*, 2018.

Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 5797–5808, 2019.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. Superglue: a stickier benchmark for general-purpose language understanding systems. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pp. 3266–3280, 2019a.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *7th International Conference on Learning Representations, ICLR 2019*, 2019b.

Sarah Wiegreffe and Yuval Pinter. Attention is not not explanation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 11–20, 2019.

Adina Williams, Nikita Nangia, and Samuel Bowman. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 1112–1122. Association for Computational Linguistics, 2018. URL http://aclweb.org/anthology/N18-1101.

Chulhee Yun, Yin-Wen Chang, Srinadh Bhojanapalli, Ankit Singh Rawat, Sashank Reddi, and Sanjiv Kumar. O(n) connections are expressive enough: Universal approximability of sparse transformers. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 13783–13794. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/9ed27554c893b5bad850a422c3538c15-Paper.pdf.

Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. Big bird: Transformers for longer sequences. *arXiv preprint arXiv:2007.14062*, 2020.

A EXPERIMENTAL SETUP

In this section we present details of our experimental setup. For all the models we closely follow the settings from their original implementation and use the publicly available code. We use TPUv2 and TPUv3s for our experiments.

A.1 BERT

We pretrain the BERT models on Wikipedia and Books datasets using the masked language model objective. We use a batch size of 512 and train for 1M steps using the Adam optimizer with $1e-4$ weight decay. We use dropout of 0.1. For finetuning on MNLI, we use a batch size of 128 and train for 3 epochs. We use $3e-5$ learning rate. For finetuning on SQuAD, we use a batch size of 48 and train for 2 epochs. We use $8e-5$ learning rate.

A.2 T5

We pretrain the T5 models on C4 dataset using the span corruption objective. We use a batch size of 128 with 512 input sequence length and 114 target sequence length. We pretrain for 524288 steps. We use Adafactor optimizer with 1.0 peak learning rate. We use linear learning rate warmup for 10k steps followed by square root decay. We use dropout of 0.1. For finetuning we train for additional 262,144 steps. We use a constant learning rate of $1e-3$. We use a target length of 84 and 62 for Glue and SuperGlue respectively.

A.3 VIT

We pretrain the ViT models on the JFT-300M dataset for 7 epochs. We use a batch size of 4096. We use Adam optimizer with a learning rate of 8e-4. We use 10k linear warmup steps and linear decay to 1e-5. We use a weight decay of 0.1. For finetuning on ImageNet, we train for 8 epochs with a batch size of 512. We use momentum SGD with a peak learning rate of 3e-2. We use 500 warmup steps and decay learning rate to 3e-4 using cosine schedule.

A.4 MACHINE TRANSLATION

We train the encoder-decoder Transformer on the WMT2018 datasets. We use a 6 layer model with a hidden size of 512 with 8 heads per layer. We use input sequence length of 256 with a batch size of 4096 with padding. We use Adam optimizer with linear warmup for 8k steps and square root decay. We use the Tensor2Tensor framework with default settings for training all the models (Vaswani et al., 2018).

A.5 LRA

We use the publicly available code for training models on this benchmark [1]. We train all the models with the default configs. We use a Transformer with 4 layers and 8 attention heads per layer with a hidden size of 512. We use Adam optimizer with linear warmup and square root decay. We use dropout and a weight decay of 1e-1.

B ADDITIONAL EXPERIMENTAL RESULTS

B.1 DIFFERENT CONFIGURATIONS OF REUSE

Table 7: Comparison of different configurations of reusing attention.

Model	**Reuse/skip** layers (P)	**MLM** Acc	**MNLI** Acc	**SQuAD V1.1** F1	**SQuAD V2.0** F1
$\text{BERT}_{\text{LARGE}}$	-	73.76	87.97	91.87	82.41
Reuse - Proposed	12	73.64	87.75	91.92	82.56
Reuse - Alternate	12	73.88	87.31	91.56	82.29
Reuse - AllEnd	12	74.07	87.24	91.55	81.7
Skip	12	73.4	87.3	91.65	81.83

In this section we present comparisons between different configurations of reuse attention. We consider full layer reuse setting. The proposed architecture in Algorithm 1, Reuse-Proposed, reuses the first P layers following the first layer.

[1] https://github.com/google-research/long-range-arena

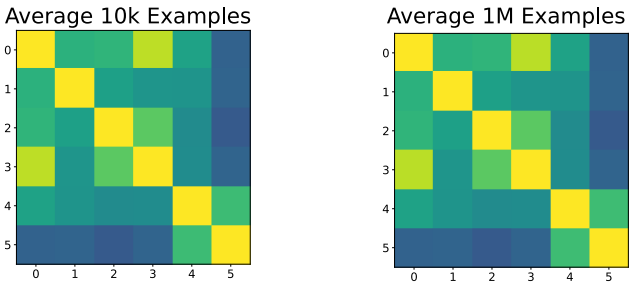

Figure 6: **Number of examples.** We show the all layer pair best head similarity scores for a six layer BERT model, averaged over (left) 10k examples as in the main paper, and over (right) 1M examples. The two averages are essentially identical, showing that our setting of using 10k examples is sufficient to draw conclusions about attention redundancy.

Here we consider three different variants, 1) Reuse - Alternate : here we reuse attention in alternate layers, i.e. we reuse attention in all the even numbered layers, with first layer being exact. 2) Reuse - AllEnd : here we reuse attention in the final P layers with last layer being exact, i.e., we reuse attention in layers $L-P-1$ to $L-1$. 3) Skip - in this architecture we skip the attention computation completely in P layers. The Transformer blocks in the Skip layers only consist of the tokenwise feedforward layers.

We compare above configurations using $\text{BERT}_{\text{LARGE}}$ (24 layers, 16 heads) as baseline in Table 7. We first notice that the Reuse-Proposed performs the best among all the configurations, with Reuse-Alternate performing slightly better than the Reuse-AllEnd and Skip, both of which have the worst performance.

B.2 Number of examples

We use 10k examples to compute the mean attention similarity for results in Section 3. Since we are only computing mean Total Variation similarity of probability distributions in 512 dimensions, $10K$ examples is enough. We also present mean similarity computed using $1M$ examples in Figure 6 and we got same results.

C Analysis

C.1 Attention similarity

In this section we present our analysis that shows that attention computed with random weights leads to less similarity in attention scores.

Proof of Lemma 1. Recall that $\tilde{\boldsymbol{A}}_1 = \boldsymbol{X}^T * \boldsymbol{W}_{q1}^T * \boldsymbol{W}_{k1} * \boldsymbol{X}$ and $\tilde{\boldsymbol{A}}_2 = \boldsymbol{X}^T * \boldsymbol{W}_{q2}^T * \boldsymbol{W}_{k2} * \boldsymbol{X}$. Let $\boldsymbol{X}_i$ be the ith column of $\boldsymbol{X}$. Now we write the expected error between attention scores for a single entry.

$$
\begin{aligned}
&\mathbb{E}[\left(\boldsymbol{X}^T * \boldsymbol{W}_{q1}^T * \boldsymbol{W}_{k1} * \boldsymbol{X} - \boldsymbol{X}^T * \boldsymbol{W}_{q2}^T * \boldsymbol{W}_{k2} * \boldsymbol{X}\right)_{ij}^2] \\
&\qquad = \mathbb{E}[(\boldsymbol{X}^T * \boldsymbol{W}_{q1}^T * \boldsymbol{W}_{k1} * \boldsymbol{X})_{ij}^2 + (\boldsymbol{X}^T * \boldsymbol{W}_{q2}^T * \boldsymbol{W}_{k2} * \boldsymbol{X})_{ij}^2 \\
&\qquad\qquad - 2(\boldsymbol{X}^T * \boldsymbol{W}_{q1}^T * \boldsymbol{W}_{k1} * \boldsymbol{X})_{ij}(\boldsymbol{X}^T * \boldsymbol{W}_{q2}^T * \boldsymbol{W}_{k2} * \boldsymbol{X})_{ij}] \\
&\qquad = \mathbb{E}[(\boldsymbol{X}_i^T * \boldsymbol{W}_{q1} * \boldsymbol{W}_{k1}^T * \boldsymbol{X}_j)^2] + \mathbb{E}[(\boldsymbol{X}_i^T * \boldsymbol{W}_{q2} * \boldsymbol{W}_{k2}^T * \boldsymbol{X}_j)^2] \\
&\qquad\qquad - 2\mathbb{E}[(\boldsymbol{X}_i^T * \boldsymbol{W}_{q1} * \boldsymbol{W}_{k1}^T * \boldsymbol{X}_j)(\boldsymbol{X}_i^T * \boldsymbol{W}_{q2} * \boldsymbol{W}_{k2}^T * \boldsymbol{X}_j)]
\end{aligned}
$$

Now the last term in the above inequality is 0 as $\boldsymbol{W}_{q1}, \boldsymbol{W}_{k1}$ are independent from $\boldsymbol{W}_{q2}, \boldsymbol{W}_{k2}$ and have zero mean. Hence,

$$
\begin{aligned}
&\mathbb{E}[\left(\boldsymbol{X}^T * \boldsymbol{W}_{q1}^T * \boldsymbol{W}_{k1} * \boldsymbol{X} - \boldsymbol{X}^T * \boldsymbol{W}_{q2}^T * \boldsymbol{W}_{k2} * \boldsymbol{X}\right)_{ij}^2] \\
&\qquad = \mathbb{E}[(\boldsymbol{X}_i^T * \boldsymbol{W}_{q1} * \boldsymbol{W}_{k1}^T * \boldsymbol{X}_j)^2] + \mathbb{E}[(\boldsymbol{X}_i^T * \boldsymbol{W}_{q2} * \boldsymbol{W}_{k2}^T * \boldsymbol{X}_j)^2] \\
&\qquad = 2\mathbb{E}[(\boldsymbol{X}_i^T * \boldsymbol{W}_{q1} * \boldsymbol{W}_{k1}^T * \boldsymbol{X}_j)^2]
\end{aligned}
$$

The last equality follows since $\boldsymbol{W}_{q1}, \boldsymbol{W}_{k1}$ have identical distribution with $\boldsymbol{W}_{q2}, \boldsymbol{W}_{k2}$. □

C.2 REUSE ATTENTION

In this section we present our analysis showing when can reuse attention approximate standard attention well. We consider a 2 layer 1 head Transformer architecture with simplifications for analysis. In particular following Levine et al. (2020) we consider an architecture that excludes the ReLU activation and layer-norm operation. Note that, unlike Levine et al. (2020), we allow for the softmax normalization. We emphasize that while these simplifications do affect performance they still preserve the main self-attention functionality - which is our main focus. Please see the discussion in Levine et al. (2020) for justification.

Additionally, we make the following approximation about the input to the attention scores in the second layer of the Transformer model.

Recall $\boldsymbol{A}_2 = \boldsymbol{Z}^T \boldsymbol{W}_{q2}^T * \boldsymbol{W}_{k2} * \boldsymbol{Z}$, where $\boldsymbol{Z}$ is the output of the first layer $\boldsymbol{Z} = \boldsymbol{X} + \Delta\boldsymbol{X}$, where $\Delta\boldsymbol{X} = \boldsymbol{A}_1 \boldsymbol{X} \boldsymbol{W}_1$. Hence,

$$\begin{aligned}\boldsymbol{A}_2 &= (\boldsymbol{X} + \Delta\boldsymbol{X})^T * \boldsymbol{W}_{q2}^T * \boldsymbol{W}_{k2} * (\boldsymbol{X} + \Delta\boldsymbol{X}) \\ &= \boldsymbol{X}^T \boldsymbol{W}_{q2}^T * \boldsymbol{W}_{k2} \boldsymbol{X} + \Delta\boldsymbol{X}^T \boldsymbol{W}_{q2}^T * \boldsymbol{W}_{k2} \boldsymbol{X} + \boldsymbol{X}^T \boldsymbol{W}_{q2}^T * \boldsymbol{W}_{k2} \Delta\boldsymbol{X} + \Delta\boldsymbol{X}^T \boldsymbol{W}_{q2}^T * \boldsymbol{W}_{k2} \Delta\boldsymbol{X}\end{aligned}$$

Note that usually $\|\Delta\boldsymbol{X}\|$ is much smaller than $\|\boldsymbol{X}\|$. Hence, we approximate the attention computation equation by ignoring the $\Delta\boldsymbol{X}$ terms, giving us $\boldsymbol{A}_2 = \boldsymbol{X}^T * \boldsymbol{W}_{q2}^T * \boldsymbol{W}_{k2} * \boldsymbol{X}$. We note that this is a reasonably approximation as we mainly consider attention in the first two layers in our analysis.

Under these simplification we can write the output of a 2 layer 1 head attention architecture as follows. Let input be a $\boldsymbol{X} \in \mathbb{R}^{n \times d}$. Output of this model is

$$\boldsymbol{Y} = \boldsymbol{X} + \boldsymbol{A}_1 \boldsymbol{X} \boldsymbol{W}_1 + \boldsymbol{A}_2 \boldsymbol{X} \boldsymbol{W}_2 + \boldsymbol{A}_2 \boldsymbol{A}_1 \boldsymbol{X} \boldsymbol{W}_1 \boldsymbol{W}_2.$$

Note that the different linear projections in the Transformer, value output, feedforward layer, all can be absorbed into a single projection $\boldsymbol{W}$ as they are all linear projections. Here $\boldsymbol{A}_1 = \sigma(\boldsymbol{X}^T * \boldsymbol{W}_{q1}^T * \boldsymbol{W}_{k1} * \boldsymbol{X})$ and $\boldsymbol{A}_2 = \sigma(\boldsymbol{X}^T * \boldsymbol{W}_{q2}^T * \boldsymbol{W}_{k2} * \boldsymbol{X})$ are the attention scores computed by the two layers.

Under the same setting, the output of the reuse attention model is

$$\widehat{\boldsymbol{Y}} = \boldsymbol{X} + \widehat{\boldsymbol{A}} \boldsymbol{X} \boldsymbol{W}_3 + \widehat{\boldsymbol{A}} \boldsymbol{X} \boldsymbol{W}_4 + \widehat{\boldsymbol{A}}^2 \boldsymbol{X} \boldsymbol{W}_3 \boldsymbol{W}_4.$$

Here $\widehat{\boldsymbol{A}} = \boldsymbol{X}^T * \widehat{\boldsymbol{W}}_q^T * \widehat{\boldsymbol{W}}_k * \boldsymbol{X}$ is the attention scores computed by the first layer and reused in the second layer. Let $\|.\|_2$ denote the spectral norm of a matrix.

Lemma 3. *Let attention computed by the two layers be ϵ apart for all inputs $\boldsymbol{X}$, i.e*

$$\|\boldsymbol{A}_1 - \boldsymbol{A}_2\|_2 \leq \epsilon.$$

Also let all input and parameter norms to be less than 1, i.e. $\|\boldsymbol{W}_1\|_2, \|\boldsymbol{W}_2\|_2, \|\boldsymbol{X}\|_2 \leq 1$. Then, there exists a choice for $\widehat{\boldsymbol{A}}, \boldsymbol{W}_3, \boldsymbol{W}_4$, such that the error scales as

$$\|\widehat{\boldsymbol{Y}} - \boldsymbol{Y}\|_2 \leq 2\epsilon + \frac{\epsilon^2}{2}.$$

This lemma shows us that if the attention scores are closer, then reuse attention the output remains close to the standard transformer output. Hence, this shows that we can reuse attention when there is high attention similarity without much output error.

Proof. We present a construction based proof. We set $\boldsymbol{W}_3 = \boldsymbol{W}_1$, $\boldsymbol{W}_4 = \boldsymbol{W}_2$, and $\widehat{\boldsymbol{W}}_q^T * \widehat{\boldsymbol{W}}_k = \frac{\boldsymbol{W}_{q1}^T * \boldsymbol{W}_{k1} + \boldsymbol{W}_{q2}^T * \boldsymbol{W}_{k2}}{2}$. Note that we can always find such matrices $\widehat{\boldsymbol{W}}_q, \widehat{\boldsymbol{W}}_k$ as they are all full dimensional and hence can be full rank. This implies $\widehat{\boldsymbol{A}} = \frac{\boldsymbol{A}_1 + \boldsymbol{A}_2}{2}$ for all $\boldsymbol{X}$.

$$\begin{aligned}\widehat{\boldsymbol{Y}} - \boldsymbol{Y} &= (\boldsymbol{A}_1 - \widehat{\boldsymbol{A}}) \boldsymbol{X} \boldsymbol{W}_1 + (\boldsymbol{A}_2 - \widehat{\boldsymbol{A}}) \boldsymbol{X} \boldsymbol{W}_2 + (\boldsymbol{A}_2 \boldsymbol{A}_1 - \widehat{\boldsymbol{A}}^2) \boldsymbol{X} \boldsymbol{W}_1 \boldsymbol{W}_2 \\ &= \frac{(\boldsymbol{A}_1 - \boldsymbol{A}_2)}{2} \boldsymbol{X} \boldsymbol{W}_1 + \frac{(\boldsymbol{A}_2 - \boldsymbol{A}_1)}{2} \boldsymbol{X} \boldsymbol{W}_2 + (\boldsymbol{A}_2 \boldsymbol{A}_1 - \widehat{\boldsymbol{A}}^2) \boldsymbol{X} \boldsymbol{W}_1 \boldsymbol{W}_2 \\ &= \frac{(\boldsymbol{A}_1 - \boldsymbol{A}_2)}{2} \boldsymbol{X} (\boldsymbol{W}_1 - \boldsymbol{W}_2) + \frac{1}{2} (\boldsymbol{A}_2 \boldsymbol{A}_1 - \boldsymbol{A}_1 \boldsymbol{A}_2 - \frac{1}{2} (\boldsymbol{A}_1 - \boldsymbol{A}_2)^2) \boldsymbol{X} \boldsymbol{W}_1 \boldsymbol{W}_2\end{aligned}$$

$$
\begin{aligned}
\|\tilde{\boldsymbol{Y}} - \boldsymbol{Y}\|_2 &\leq \|\frac{(\boldsymbol{A}_1 - \boldsymbol{A}_2)}{2}\boldsymbol{X}(\boldsymbol{W}_1 - \boldsymbol{W}_2)\|_2 + \|\frac{1}{2}(\boldsymbol{A}_2\boldsymbol{A}_1 - \boldsymbol{A}_1\boldsymbol{A}_2 - \frac{1}{2}(\boldsymbol{A}_1 - \boldsymbol{A}_2)^2)\boldsymbol{X}\boldsymbol{W}_1\boldsymbol{W}_2\|_2 \\
&\leq \frac{\epsilon}{2}\|\boldsymbol{X}(\boldsymbol{W}_1 - \boldsymbol{W}_2)\|_2 + \frac{\epsilon^2}{2}\|\boldsymbol{X}\boldsymbol{W}_1\boldsymbol{W}_2\|_2 + \|\frac{1}{2}(\boldsymbol{A}_2\boldsymbol{A}_1 - \boldsymbol{A}_1\boldsymbol{A}_2)\boldsymbol{X}\boldsymbol{W}_1\boldsymbol{W}_2\|_2 \\
&= \frac{\epsilon}{2}\|\boldsymbol{X}(\boldsymbol{W}_1 - \boldsymbol{W}_2)\|_2 + \frac{\epsilon^2}{2}\|\boldsymbol{X}\boldsymbol{W}_1\boldsymbol{W}_2\|_2 + \|\frac{1}{2}(\boldsymbol{A}_1(\boldsymbol{A}_1 - \boldsymbol{A}_2) + (\boldsymbol{A}_2 - \boldsymbol{A}_1)\boldsymbol{A}_1)\boldsymbol{X}\boldsymbol{W}_1\boldsymbol{W}_2\|_2 \\
&\leq \frac{\epsilon}{2}\|\boldsymbol{X}(\boldsymbol{W}_1 - \boldsymbol{W}_2)\|_2 + \frac{\epsilon^2}{2}\|\boldsymbol{X}\boldsymbol{W}_1\boldsymbol{W}_2\|_2 + \epsilon\|\boldsymbol{A}_1\|_2\|\boldsymbol{X}\boldsymbol{W}_1\boldsymbol{W}_2\|_2 \\
&\leq 2\epsilon + \frac{\epsilon^2}{2}.
\end{aligned}
$$

Note that $\boldsymbol{A}_1$ is a stochastic matrix with max singular value at most 1. □