BELIEFFORMER: BELIEF ATTENTION IN TRANSFORMER

Anonymous authors
Paper under double-blind review

ABSTRACT

In this paper, we consider modifying the attention layer in Transformer to improve its generalization performance. Conceptually speaking, the standard attention layer takes the softmax-based weighted summation of V vectors as the residual signal (with a linear mapping for dimensionality alignment) when performing the skip-connection operation. Inspired by distribution optimization, we propose to first perform an orthogonal projection of the softmax-based weighted summation of V vectors with respect to the original V vectors and then take the orthogonal projection instead as the residual signal (with a linear mapping for dimensionality alignment) when performing the skip-connection operation. By doing so, the token vectors are modified relatively more along their tangent directions compared to their magnitudes. Intuitively speaking, the orthogonal projection reflects a belief about the discrepancy between the weighted summation of V vectors and the V vectors themselves. We refer to the newly modified layer and the overall architecture as the belief-attention and the BeliefFormer, respectively. To further improve performance, we also design a variant of belief-attention by incorporating two types of orthogonal projections, referred to as belief-attention*. Extensive experiments show that the two new variants of attention layer in Transformers lead to better performance than the standard attention for image classification over ImageNet and natural language processing when training nano-GPT2.

1 INTRODUCTION

In recent years, Transformers (Vaswani et al., 2017) have made significant advances across a range of data analysis fields, including natural language processing (NLP) (Achiam et al., 2023; Touvron et al., 2023), computer vision (Dosovitskiy et al., 2021), image generation and editing (Peebles & Xie, 2023; Hatamizadeh et al., 2024; Zhang et al., 2023), and audio processing (Latif et al., 2023). A fundamental component of transformers is the attention layer, which enables the model to capture long-range dependencies within a sequence of tokens. This mechanism works by computing a weighted summation of the value (V) vectors based on the similarity between query (Q) and key (K) vectors, determined via a softmax function. Conceptually, the attention operation allows each token to aggregate relevant information from all other tokens. Following the attention layer, a feedforward network (FFN) processes each token independently, which can be interpreted as local information fusion. Recent large language models (LLMs) exploit a so-called mixture of experts (MoE) as an extension of basic FFN to improve the performance, where at the inference stage, only certain percentage of weights in the FFN layer are activated depending on the particular input.

One prominent research direction focuses on reducing the quadratic computational complexity inherent in the standard attention layer when processing long token sequences. Various simplified attention schemes have been proposed, which include, for example, LinFormer (Wang et al., 2020), LongFormer (Beltagy et al., 2020), ReFormer (Kitaev et al., 2020), FlashAttention (Dao, 2023), RingAttention (Liu et al., 2023), BurstAttention (Sun et al., 2023). FlashAttention is being widely used in practical situations as it reduces the computational complexity considerably without introducing any approximation in the standard attention layer.

In this work, we attempt to modify the attention layer to improve the generalization performance of Transformers. To do so, we draw inspiration from distributed optimization. From a high-level point of view, the attention-FFN framework in Transformers exhibits a certain similarity to the

framework of distributed optimization over an undirected graph. In general, a typical distributed optimization algorithm (see (Zhang & Heusdens, 2018; Boyd et al., 2011)) iteratively alternates between information-aggregation and information-fusion operations until all nodes in the graph reach consensus. Typical algorithms include alternating direction method of multipliers (ADMM) (Boyd et al., 2011) and primal-dual method of multipliers (PDMM) (Zhang & Heusdens, 2018). Considering PDMM as an example, it was primarily designed to solve the following separable convex optimisation problem over an undirected graph $G = (\mathcal{V}, \mathcal{E})$ representing a pear-to-pear (P2P) network from practice:

$$\begin{aligned} \text{minimise} \quad & \sum_{i\in\mathcal{V}} f_i(x_i) \\ \text{subject to} \quad & A_{ij}x_i + A_{ji}x_j = b_{ij}, \quad (i,j) \in \mathcal{E}, \end{aligned} \tag{1}$$

where each node i carries a local objective function $f_i(\cdot) : \mathbb{R}^{d_i} \to \mathbb{R}$ and each edge (i, j) carries a linear equality constraint as specified by the constant $(A_{ij}, A_{ji}, b_{ij}) \in (\mathbb{R}^{d_{ij}\times d_i}, \mathbb{R}^{d_{ij}\times d_j}, \mathbb{R}^{d_{ij}})$. As will be discussed in detail in Section 2, at each iteration of PDMM, each node in the network performs information aggregation from neighbours (corresponding to attention in Transformer) and local information fusion (corresponding to FFN). One key property of PDMM is that its update expression utilizes the consensus discrepancy in terms of the residual error of the linear edge-constraints in (1), which is essential to make the algorithm converge.

We consider extending the standard attention by drawing inspiration from PDMM. In particular, we propose to first perform orthogonal projection of the softmax-based weighted summation of V vectors with respect to their respective original V vectors. The orthogonal projection is then taken as the residual signal, and is further processed by a linear mapping for dimensionality alignment in preparation for skip-connection. This above newly designed attention, referred to as belief-attention, encourages updates to the token vectors more in their tangent directions and less in their magnitudes. The overall Transformer architecture with belief-attention is referred to as BeliefFormer. In brief, we make three contributions in the paper:

- Belief-attention is proposed as an extension of attention by taking the orthogonal projection as the residual signal before applying the linear mapping for dimensionality alignment. The orthogonal projection provides a belief about the discrepancy between the softmax-based weighted summation of V vectors and V vectors themselves.
- A variant of belief-attention is also proposed by combining two types of orthogonal projections for capturing more information, referred to as belief-attention*.
- Experimental on training nano-GPT2 for natural language processing (NLP), and training ViTs over Imagenet and CIFAR10, show that belief-attention and its variant belief-attention* demonstrate considerable improvement in validation performance.

2 Brief Review of PDMM

To facilitate node-oriented distributed optimization of (1) over a graph $G = (\mathcal{V}, \mathcal{E})$, PDMM introduces two Lagrangian multipliers $\lambda_{i|j}$ and $\lambda_{j|i}$ for the linear constraint over the edge $(i, j) \in \mathcal{E}$. Let $\mathcal{N}_i$ denote the set of neighbors for node i. At the kth iteration, each new update x_i^{k+1} is computed in terms of the information $\{(x_{j|i}^k, \lambda_{j|i}^k)|j \in \mathcal{N}_i\}$ from neighbors as

$$x_i^{k+1} = \arg\min_{x_i\in\mathbb{R}^{d_i}} \underbrace{\left[f_i(x_i) - x_i^T \overbrace{\Big(\sum_{i\in\mathcal{N}_i} A_{ij}^T\lambda_{j|i}^k\Big)}^{\text{1st info. aggregation}} + \overbrace{\sum_{j\in\mathcal{N}_i} \frac{\rho}{2}\|A_{ij}x_i + A_{ji}x_j^k - b_{ij}\|^2}^{\text{2nd info. aggregation}} \right]}_{\text{info. fusion}} \quad \forall i \in \mathcal{V}, \tag{2}$$

where the stepsize $\rho > 0$. Once x_i^{k+1} is available, the associated Lagrangian multipliers of node i are updated to be

$$\lambda_{i|j}^{k+1} = \lambda_{j|i}^{k} + \rho(b_{ij} - A_{ji}x_j^k - A_{ij}x_i^{k+1}) \quad \forall j \in \mathcal{N}_i \tag{3}$$

$$= \underbrace{\lambda_{i|j}^{k-1} + \overbrace{\rho(b_{ij} - A_{ji}x_j^k - A_{ij}x_i^{k-1}) + \rho(b_{ij} - A_{ji}x_j^k - A_{ij}x_i^{k+1})}^{\text{residual signals}}}_{\text{skip-connection}} \quad \forall j \in \mathcal{N}_i, \tag{4}$$

where the computation of $\lambda_{i|j}^{k+1}$ can be interpreted as performing skip-connection by adding two residual signals to $\lambda_{i|j}^{k-1}$. Detailed convergence results of the algorithm can be found in Zhang & Heusdens (2018); Sherson et al. (2019).

By inspection of (2), it is seen that the computation of x_i^{k+1} involves two weighted summations from neighbors, which are $\sum_{i\in\mathcal{N}_i} A_{ij}^T \lambda_{j|i}^k$ and $\sum_{j\in\mathcal{N}_i} \|A_{ij}x_i + A_{ji}x_j^k - b_{ij}\|^2$ as contributed by the first and second information aggregation terms. x_i^{k+1} is then obtained by solving a small-size optimization problem with the local function $f_i(\cdot)$, and can be viewed as local information fusion.

Next we study the Lagrangian multiplier $\lambda_{j|i}^k$ being explored in the computation of x_i^{k+1}. It is not difficult to conclude from (3)-(4) that $\lambda_{j|i}^k$ can be represented as a summation of the historical residual errors of the linear equality constraint for edge $(i,j) \in \mathcal{E}$. For the case of k being even, $\lambda_{j|i}^k$ can be represented as

$$\lambda_{j|i}^k = \lambda_{j|i}^0 + \rho \sum_{m=1}^{k/2} (b_{ij} - A_{ji}x_j^{2m-2} - A_{ij}x_i^{2m-1}) + \rho \sum_{m=1}^{k/2} (b_{ij} - A_{ji}x_j^{2m-1} - A_{ij}x_i^{2m}). \tag{5}$$

We take each residual error in (5) as the measurement of the consensus discrepancy between the pair of nodes (i,j). As a result, $\lambda_{j|i}^k$ is computed by accumulating the residual errors across the past iterations.

In addition to the Lagrangian multipliers for capturing the historical consensus discrepancy, it is clear from (2) that the set of quadratic penalty functions $\{\|A_{ij}x_i + A_{ji}x_j^k - b_{ij}\|\}_{j\in\mathcal{N}_i}$ are also included in computation of x_i^{k+1}. The penalty functions attempt to softly constrain x_i^{k+1} in a region that incurs small consensus discrepancy (with regard to the predefined edge-constraints) with respect to the neighbors $\{x_j^k\}_{j\in\mathcal{N}_i}$. The parameter $\rho > 0$ in front of the penalty functions and in (5) controls the contribution of the consensus discrepancy when updating the primal variables $\{x_i\}_{i\in\mathcal{V}}$.

3 Belief-attention via orthogonal projections

In this section, we first briefly revisit the standard attention in Transformer. We then motivate and present the orthogonal projections in designing belief-attention. Lastly, we briefly discuss the limitations of belief-attention.

3.1 Revisiting Attention in transformer

The original work (Vaswani et al., 2017) proposes the encoder-decoder structure in the transformer for NLP applications. The attention-FFN framework is slightly different in encoder and decoder. For the purposes of demonstration, we consider a simplified version of attention, represented as (see (MHA, 2023; Dosovitskiy et al., 2021))

$$\mathrm{H}_m(X) = \text{attention}(\overbrace{XW_m^Q}^{Q_m}, \overbrace{XW_m^K}^{K_m}, \overbrace{XW_m^V}^{V_m}) \quad m = 1, \ldots, M \tag{6}$$

$$\mathrm{MH}(X) = \text{Concat}(\mathrm{H}_1(X), \ldots, \mathrm{H}_M(X)) \tag{7}$$

$$X \Leftarrow \underbrace{X + \overbrace{\mathrm{MH}(X)W^o}^{\text{linear mapping}}}_{\text{skip-connection}} \tag{8}$$

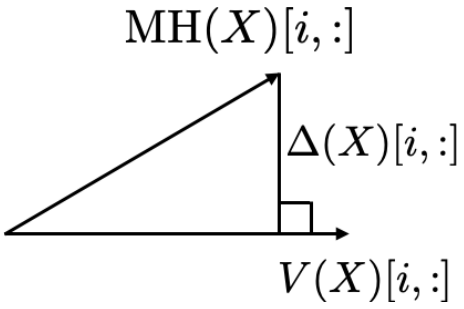

Figure 1: Orthogonal projection

```
#linear mapping
Wo = torch.nn.Linear(d_in,d)
##########################
# MH: multi-head attention
# V:  original V tensor
num = torch.sum(MH*V,dim=-1)
den = torch.sum(V*V,dim=-1)
Delta = MH - (num/den)*V
X = X + Wo(Delta)
```

Figure 2: Demonstration of pytorch code for belief-attention

where the tensor $X \in \mathbb{R}^{n\times d}$ is the input from the layer below in Transformer, (W_m^Q, W_m^K, W_m^V) are the three learnable matrices for computing $(Q_m, K_m, V_m) \in (\mathbb{R}^{n\times d_m}, \mathbb{R}^{n\times d_m}, \mathbb{R}^{n\times d_m})$ of the mth attention, and Concat$(\ldots)$ stacks up M attentions $\{\mathrm{H}_m(X)\}_{m=1}^M$, which is further processed by the linear mapping W^o to make its dimensionality be consistent with that of X. Lastly, the notations H and MH stand for "head" and "multi-head", respectively.

To facilitate the discussion of attention and PDMM later on, we first briefly explain the notations in (6)-(8). Following the convention of python-based implementation (e.g., pytorch) of attention, the tensor $X \in \mathbb{R}^{n\times d}$ has n tokens and each token is of dimension d. Therefore, the computation for (Q_m, K_m, V_m) in (6) and the linear mapping in (8) are conducted in a token-wise manner.

It is well-known that the attention operation in Equ. (6) is a QK-softmax-based weighted summation of the n row vectors in V_m, given by

$$\mathrm{H}_m(X) = \overbrace{\mathrm{softmax}\left(\frac{Q_m K_m^T}{\sqrt{d_m}}\right)}^{\text{info. aggregation}} V_m \tag{9}$$

where d_m is the dimension of the row vectors in Q_m. The softmax term computes the unified relevance of each token with respect to neighboring tokens, which generally stabilizes the training process in comparison to other forms of weighted summation. Similarly to that of PDMM, the computed weighted summation of the n row vectors in V_m can be taken as information aggregation from all neighbors.

In general, for a standard non-causal attention, every two tokens are neighbors, which corresponds to a fully connected graph in distributed optimization. On the other hand, a non-casual attention is actually associated with a sparse directed graph. This is because only earlier tokens could make contributions to the current considered token. We will not discuss those different types of graphs in relation with different types of attentions in detail, which is out of the scope in this paper.

3.2 Update expression of belief-attention

Motivation: By inspection of (4) for PDMM and (8) for the standard attention, both expressions have the skip-connection operations. In (4), PDMM exploits consensus discrepancy in the form of residual errors of the linear equality constraints in its update expression. However, in the expression (8) for the standard attention, the tensor MH(X) is not really a residual signal from the perspective of distributed optimization. This is because each term $\mathrm{H}_m(X)$ within MH(X) does not actually measure any discrepancy among the tokens. We argue that the Transformer architecture would benefit if certain type of discrepancy could be captured by the attention layer. By doing so, the learnable parameters in Transformer could promptly respond to the discrepancy among the tokens during the training process, thus making the learning procedure more efficient.

Taking orthogonal projection as the residual signal: Following the above guidance, we propose to perform orthogonal projection of each row vector in MH(X) with respect to its original row vector in

$$V(X) = \mathrm{Concat}(V_1, \ldots, V_M).$$

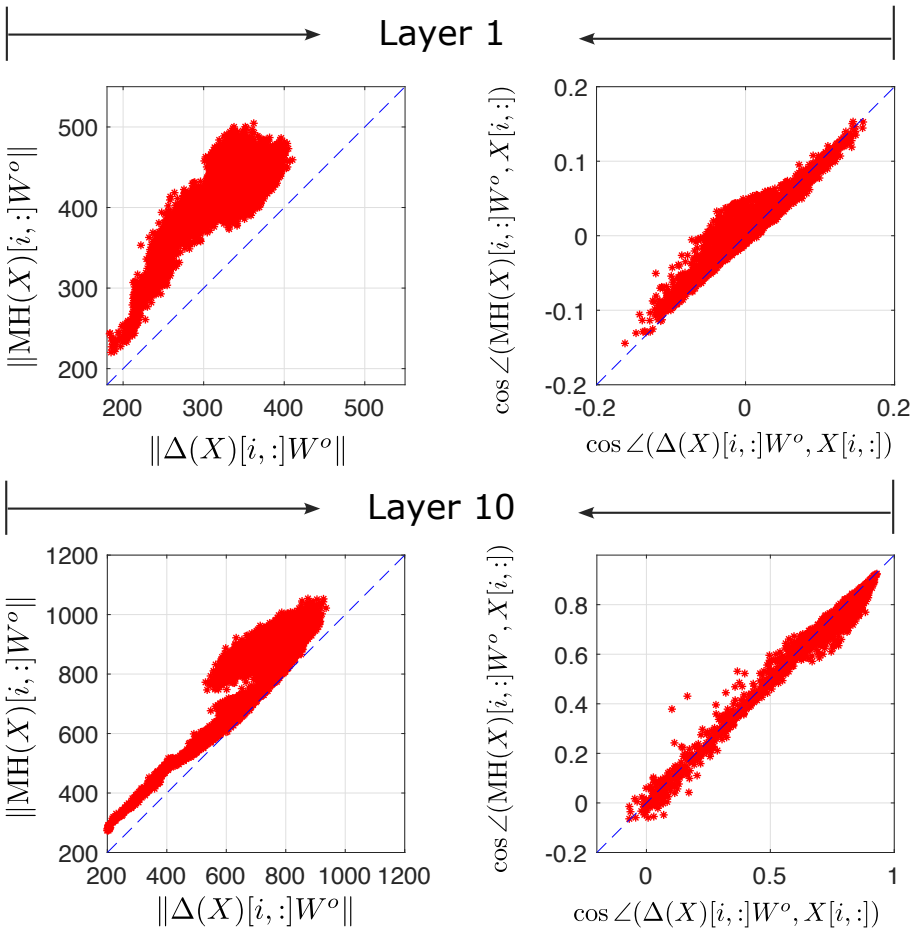

Figure 3: Demonstration of the impact of the orthogonal projection $\Delta X[i,:]$. The notation $\angle(\cdot,\cdot)$ stands for the angle between two vectors. The data points in the above four plots are collected when training BeliefFormer of 12 belief-attention layers over ImageNet for the 1st epoch (see Appendix A for explaining how the data points are collected in detail).

We use $\text{MH}(X)[i,:]$ and $V(X)[i,:]$ to denote the ith row vector of $\text{MH}(X)$ and $V(X)$, respectively. Their associated orthogonal projection is computed as

$$\Delta(X)[i,:] = \text{MH}(X)[i,:] - \alpha_i V(X)[i,:] \text{ where } \alpha_i = \frac{\langle \text{MH}(X)[i,:], V(X)[i,:]\rangle}{\langle V(X)[i,:], V(X)[i,:]\rangle}, \tag{10}$$

where $\langle\cdot,\cdot\rangle$ denotes the inner product of two vectors. As demonstrated in Fig. 1, $\Delta(X)[i,:]$ is orthogonal to $V(X)[i,:]$. By using algebra, one can easily show that the magnitude $\|\Delta(X)[i,:]\|$ is either small or equal to $\|\text{MH}(X)[i,:]\|$.

$$\|\Delta(X)[i,:]\| \leq \|\text{MH}(X)[i,:]\|, \tag{11}$$

where we use $\|\cdot\|$ to denote the l_2 norm of a vector.

The signal $\Delta(X)[i,:]$ reflects a belief about the discrepancy between the original vector $V(X)[i,:]$ and the newly computed vector $\text{MH}(X)[i,:]$ which is obtained via softmax-based weighted summation. A large magnitude of $|\Delta(X)[i,:]|$ indicates the the associated token should be adjusted significantly, and vice versa. .

We take $\Delta(X)$ as the residual signal when performing the skip-connection operation. Once the orthogonal projection $\Delta(X)$ is obtained, the token tensor X can be updated as follows:

$$X \Leftarrow \underbrace{X + \overbrace{\Delta(X)W^o}^{\text{linear mapping}}}_{\text{skip-connection}}, \tag{12}$$

where the linear mapping W^o is applied to the residual signal $\Delta(X)$ for dimensionality alignment. Fig. 2 demonstrates the pytorch code for realizing belief-attention. In practice, one can easily adopt the pytorch code to convert a standard attention into belief-attention. The main difference with respect to the standard attention is to subtract a scaled version of the original V tensor from the multi-head attention as represented in (10).

Impact of the update expression (12): We note that because of the linear mapping W^o in (12) and $\{W_m^V\}_{m=1}^M$ in (6), the transformed residual signal $\Delta(X)W^o$ will not be orthogonal to X. The expression (11) naturally leads to the following inequality:

$$\|\Delta(X)[i,:]W^o\| \leq \|\text{MH}(X)[i,:]W^o\|.$$

The obtained empirical results in Fig. 3 confirm that the magnitudes $\|\Delta(X)[i,:]W^o\|$ are considerably smaller than $\|\text{MH}(X)[i,:]W^o\|$. The plots in the figure also demonstrate that the two angles $\angle(\Delta(X)[i,:]W^o, X[i,:])$, and $\angle(\text{MH}(X)[i,:]W^o, X[i,:])$ are roughly the same.

The above analysis indicates that the update in (12) within belief-attention causes relatively more change in the tangent directions of the tokens (which are the row vectors in X) and less change in their magnitudes, compared to the update in the standard attention.

Discussion on an alternative choice of discrepancy metric: One might think that the vector difference $\text{MH}(X)[i,:] - V(X)[i,:]$ could also be taken as the residual signal. We argue that both the two terms in the above vector difference are dependent on $V(X)[i,:]$. It may occur that the magnitude of $\text{MH}(X)[i,:] - V(X)[i,:]$ is greater than the magnitude of $\text{MH}(X)[i,:]$. When adding $(\text{MH}(X) - V(X))W^o$ to the input tensor X, it may not serve the purpose of making more change in the tangent directions of the tokens and less change in their magnitudes.

3.3 Limitations of belief-attention w.r.t. attention

Before we discuss the limitations, we first emphasize that belief-attention does not introduce additional training parameters. That is, both belief-attention and the standard attention have the same number of parameters. Considering the time complexities, as belief-attention requires the additional orthogonal projection operations, it naturally leads to higher training and inference complexities. In the experimental section, we have quantitatively measured the training and/or inference complexities (see Table 1 and 2). It is found that the computational complexities are only slightly increased in comparison to those of the standard attention.

It is noted that the orthogonal projection is performed on a per-token basis. Therefore, in principle, its computation can be parallelized by using GPU to reduce the execution time. That is, each GPU core can take care of the computation for a small number of tokens.

4 A variant of belief-attention

In this section, we propose a variant of belief-attention by incorporating two types of orthogonal projections. Firstly, we note that in Subsection 3.2 (see also Fig. 1), we project the entire vector $\text{MH}(X)[i,:]$ w. r. t. $V(X)[i,:]$. Alternatively, we can also project the individual subvector $H_m(X)[i,:]$ w. r. t. the original subvector $V_m[i,:]$, which can be expressed as

$$\Delta_m^s(X)[i,:] = H_m(X)[i,:] - \beta_{m,i}V_m[i,:] \text{ where } \beta_{m,i} = \frac{\langle \text{H}_m(X)[i,:], V_m[i,:]\rangle}{\langle V_m[i,:], V_m[i,:]\rangle}, \tag{13}$$

for all $m = 1, \ldots, M$, and $i = 1, \ldots, n$.

Upon obtaining the two types of orthogonal projections in (10) and (13), we then exploit both of them when performing skip-connection. Our main purpose for doing so is to improve the performance of the overall neural architecture with the two types of discrepancy instead of one in belief-attention introduced earlier. The final update expression for X can be represented as

$$\Delta_s(X) = \text{Concat}(\Delta_1^s(X), \ldots, \Delta_M^s(X)) \tag{14}$$

$$X \Leftarrow \underbrace{X + \Delta(X)W^o + \Delta_s(X)W^s}_{\text{skip-connection}}, \tag{15}$$

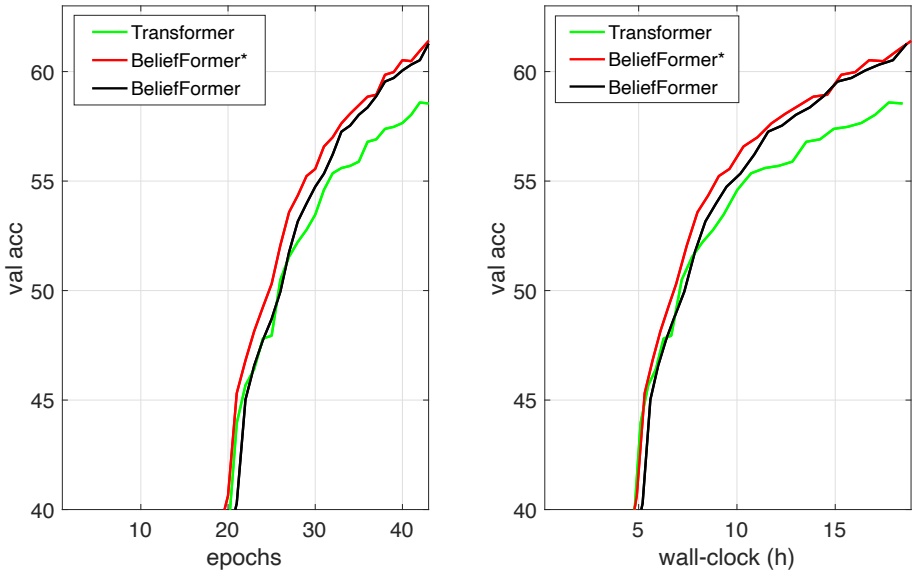

Figure 4: Performance comparison for image classification over ImageNet of 1000 classes.

	no. of parameters	time for evaluating val. dataset (s)
Transformer	22.2M (-)	37.46 (-)
BeliefFormer	22.2M (0%)	37.93 (1.3%)
BeliefFormer*	24.0M (8.1%)	38.87 (3.8%)

Table 1: Comparison of number of training parameters and computational complexities for image classification over ImageNet. We note that the input image size to the models changes dynamically over training time. Therefore, it is not feasible to measure the average training time per epoch. The values in the round bracket $(\cdot)$ account for the overhead of BeliefFormer* and BeliefFormer in comparison to Transformer in percentage.

where $\Delta_s(X)$ is obtained by stacking up M individual orthogonal projections $\{\Delta_m^s(X)\}_{m=1}^M$. In comparison to (12), an additional linear mapping W^s is required to make dimensionality alignment for $\Delta_s(X)$.

In brief, the update expressions (6)-(7), (10), and (13)-(15) together define a new type of attention layer, which we refer to as belief-attention*. Consequently, Transformer equipped with belief-attention* is referred to as BeliefFormer*. Based on the python code in Fig. 2 for belief-attention, one can easily develop the python code for belief-attention*.

4.1 Limitations of belief-attention*

Apparently, belief-attention* introduces an additional set of learnable parameters in W^s in comparison to the standard attention. Furthermore, since belief-attention* needs to perform two types of orthogonal projections, its computational complexity would be slightly higher than that of belief-attention. The results in Table 1 and 2 indicate that the overhead introduced in BeliefFormer* is acceptable given the fact that its performance gain w. r. t. that of Transformer (see Fig. 4 and 5) is remarkable.

5 Experiments

We evaluated BeliefFormer and its variant BeliefFormer* for three tasks: (1) image classification over ImageNet; (2) NLP over 5B tokens extracted from OpenWebText; (3) image classification over CIFAR10. Our experiments make use of three open-source repositories for the above three tasks, which are listed in Table 4 in the appendix. All the experiments were conducted on a computer with a single Nvidia Geforce A6000 GPU with 48GB memory.

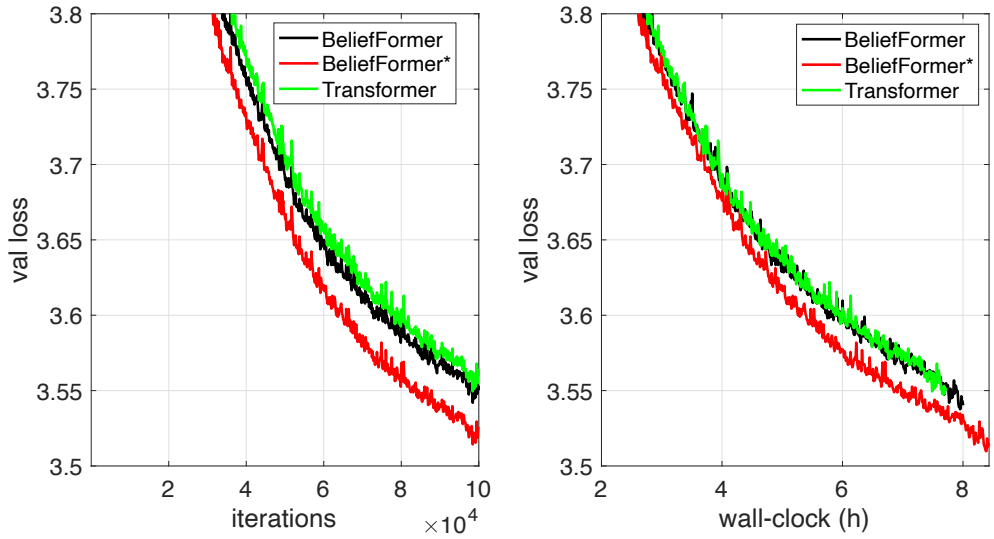

Figure 5: Performance comparison for NLP using 5B tokens extracted from OpenWebText. Transformer in the figure is in fact nano-GPT2.

	no. of parameters	training time (s) per iteration	tokens/s in training	inference time (s) per iteration
Transformer	123.5M (–)	0.274 (–)	2284	0.237 (–)
BeliefFormer	123.5M (0%)	0.284 (3.6%)	2245	0.241 (1.7%)
BeliefFormer*	130.6M (5.7%)	0.299 (9.1%)	2147	0.260 (9.7%)

Table 2: Comparison of number of parameters and computational complexities for NLP. Transformer in the table is in fact nano-GPT2. The values in the round bracket $(\cdot)$ account for the overhead of BeliefFormer* and BeliefFormer in comparison to Transformer in percentage.

In brief, it is found that both BeliefFormer and BeliefFormer* outperform Transformer consistently in all tasks. BeliefFormer* needs to introduce a small percentage of learnable parameters and marginal computational complexity in comparison to Transformer. BeliefFormer, on the other hand, only introduces marginal computational complexity.

5.1 Image classification over ImageNet

We adopted the 1st open-source repository in Table 4, which is for training a ViT over ImageNet (from 2012). There are 12 attention layers in the original ViT model (the model name is deit_small_patch16_224). We replaced each standard attention in ViT with belief-attention and belief-attention*, respectively. All the models were trained from scratch by using the ImageNet training data. The training setups in terms of the hyper-parameters follow directly from the original open source. After training, they are evaluated via the associated validation dataset.

Fig. 4 visualizes the obtained validation accuracy curves over epochs and over wall-clocks. It is clear that both BeliefFormer and BeliefFormer* outperforms Transformer (which is in fact the ViT model) significantly as the epoch index increases. The right plot in the figure against wall-clock suggests that the additional training time introduced in the two new models is negligible.

Table 4.1 summarizes the number of parameters and inference time for the three models. It is seen that the inference time for the three models are roughly the same when evaluating the valuation dataset, indicating that the orthogonal projection in the two new models can be efficiently computed by using GPU. For this particular task, BeliefFormer* introduces about 8% new parameters to handle two types of orthogonal projections.

5.2 NLP OVER A SUBSET OF OPENWEBTEXT

We adopted the 2nd open-source repository in Table 4 for this experiment. The open-source is for training nano-GPT2 over 5B tokens extracted from OpenWebText. Similarly, we replaced the standard attention layer in nano-GPT2 with belief-attention and belief-attention*, respectively. The training setups follow directly from the original open source. We refer to nano-GPT2 as Transformer in the context below.

Fig. 5 visualizes the validation loss curves over iterations and over wall-clock. Apparently, BeliefFormer* performs significantly better than the other two models even considering wall-clock instead of iterations. This suggests that it is indeed beneficial to include those two types of orthogonal projections as studied in Section 4. On the other hand, BeliefFormer performs slightly better than Transformer across iterations. If the training time complexity is taken into account, BeliefFormer and Transformer have a similar training speed.

Table 2 summarizes the number of parameters and time complexities of the three considered models. Similarly to the 1st task, BeliefFormer* slightly increases the number of parameters and computational complexity, but yields a noticeable improvement in validation performance. Considering BeliefFormer, it introduces only a small overhead in terms of computational complexity.

5.3 IMAGE CLASSIFICATION OVER CIFAR10

In this experiment, we adopted the 3rd open-source repository for training ViT-small over CIFAR10 in Table 4. Similarly, we replaced the standard attention layer by belief-attention and belief-attention* developed in this paper. In the context below, we refer to ViT-small as Transformer.

The SET-Adam optimizer was utilized Zhang (2024) in the training process for all the three models with the configuration $(\eta_0, \beta_1, \beta_2, \epsilon) = (1e-4, 0.9, 0.999, 1e^{-18})$, where η_0 denotes the initial learning rate. Each model was trained for 400 epochs. The remaining training setups follow directly from the original open source. Three experimental repetitions were performed per training setup to mitigate the effect of randomness.

Table 3 summarizes the obtained validation accuracy. It is clear that both BeliefFormer and BeliefFormer* produces considerably higher validation accuracy than Transformer. This indicates that the introduced orthogonal projections is a better choice than the softmax-based weighted summation of V vectors when performing the skip-connection in the attention layer.

Table 3: Validation accuracy and training time per epoch (in seconds) for image classification over CIFAR10. Transformer in the table is in fact ViT-small from the open-source.

Transformer		BeliefFormer		BeliefFormer*	
val. acc	validation time	val. acc	validation time	val. acc	validation time
88.15±0.55	1.66	**89.14±0.17**	1.82	88.64±0.08	2.17

6 CONCLUSIONS

In this work, we have proposed belief-attention and belief-attention* to replace attention in Transformer from a distributed optimization perspective. In particular, we first identify similarity between the update expressions of PDMM and the attention-FFN framework in Transformer. The softmax-based weighted summation in the standard attention can be viewed as information aggregation from neighboring tokens while the FFN operation can be taken as local information fusion. Inspired by PDMM that exploits the consensus discrepancy in its update expressions, we utilize the discrepancy in the form of the orthogonal projection between the weighted summation of V vectors and the original V vectors themselves when designing the two new variants of attention layer. As demonstrated in Fig. 3, usage of orthogonal projections in belief-attention and belief-attention* would make the tokens be updated relatively more in their tangent directions and less in their magnitudes. Experimental results over three tasks indicate that BeliefFormer ((aka Transformer with belief-attention)) BeliefFormer* (aka Transformer with belief-attention*) performs consistently better than Transformer in terms of the validation performance.

REFERENCES

Pytorch implimentation of multi-head attention. https://pytorch.org/docs/stable/generated/torch.nn.MultiheadAttention.html, 2023.

Josh Achiam, Steven Adler, Sandhini Agarwal, Florencia Leoni Aleman Lama Ahmad, Ilge Akkaya, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mo Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Lenny Bogdonoff Christopher Berner, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, SimÃşn Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, and Shawn Jain. Gpt-4 technical report. arXiv:2307.09288 [cs.CL], 2023.

I. Beltagy, M. E. Peters, and A. Cohan. Longformer: The long-document transformer. arXiv:2004.05150v2, 2020.

S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers. *In Foundations and TrendsÂő in Machine Learning*, 3(1):1–122, 2011.

T. Dao. Flashattention-2: Faster attention with better parallelism and work partitioning. arXiv preprint arXiv:2307.08691, 2023.

A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborna, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly abd J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.

A. Hatamizadeh, J. Song, G. Liu, J. Kautz, and A. Vahdat. DiffiT: Diffusion Vision Transformers for Image Generation. In *ECCV*, 2024.

N. Kitaev, A. Kaiser, and A. Levskaya. Reformer: The Efficient Transformer. In *ICLR*, 2020.

S. Latif, A. Zaidi, H. Cuayahuitl AZ, F. Shamshad, M. Shoukat, and J. Qadir. Transformers in speech processing: A survey. arXiv:2303.11607 [cs.CL], 2023.

H. Liu, M. Zaharia, and P. Abbeel. Ring attention with blockwise transformers for near-infinite context. arXiv:1706.03762 [cs. CL], 2023.

W. Peebles and S. Xie. Scalable Diffusion Models with Transformers. In *ICCV*, 2023.

Thomas William Sherson, Richard Heusdens, and W. Bastiaan Kleijn. Derivation and analysis of the primal-dual method of multipliers based on monotone operator theory. *IEEE Transactions on Signal and Information Processing over Networks*, 5(2):334–347, 2019. doi: 10.1109/TSIPN.2018.2876754.

A. Sun, W. Zhao, X. Han, C. Yang, Z. Liu, C. Shi, and M. Sun. Burstattention: An efficient distributed attention framework for extremely long sequences. arXiv:2403.09347 [cs.DC], 2023.

H. Touvron, L. Martin, P. Albert K. Stone, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, D. Bikel, L. Blecher, C. Canton Ferrer, M. Chen, G. Cucurull, D. Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux,

Thibaut Lavril, Jenya Lee, Yinghai Lu Diana Liskovich, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Andrew Poulton Yixin Nie, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Jian Xiang Kuan Adina Williams, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Aurelien Rodriguez Sharan Narang, Robert Stojnic, Sergey Edunov, and Thomas Scialom. Llama 2: Open foundation and fine-tuned chat models. arXiv:2307.09288 [cs.CL], 2023.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. arXiv:1706.03762 [cs. CL], 2017.

S. Wang, M. Khabsa B. Z. Li, H. Fang, and H. Ma. Linformer: Self-attention with linear complexity. arXiv:2006.04768v3, 2020.

G. Zhang. On Suppressing Range of Adaptive Stepsizes of Adam to Improve Generalisation Performance. In *ECML*, 2024.

G. Zhang and R. Heusdens. Distributed Optimization using the Primal-Dual Method of Multipliers. IEEE Trans. Signal and Information Processing over Networks, 2018.

Guoqiang Zhang, J. P. Lewis, and W. Bastiaan Kleijn. Exact diffusion inversion via bidirectional integration approximation. arXiv:2307.10829 [cs.CV], 2023.

ImagNet task	https://github.com/BorealisAI/efficient-vit-training
NLP task	https://github.com/KellerJordan/modded-nanogpt/tree/casted
CIFAR10	https://github.com/kentaroy47/vision-transformers-cifar10

Table 4: list of open-source repositories expoited in this paper.

A Regarding Generation of Figure 3.

We briefly explain how the data points were collected when generating the four plots of Fig. 3. When we trained BeliefFormer over ImageNet, we computed and collected the four quantities $\|\Delta(X)[i,:]\|$, $\|\text{MH}(X)[i,:]\|$, $\cos\angle(\Delta(X)[i,:]W^o, X[i,:])$, and $\cos\angle(\text{MH}(X)[i,:]W^o, X[i,:])$ for a particular token index $i = 0$ across different belief-attention layers and across different iterations in the first epoch. There are in total 12 belief-attention layers in the tested BeliefFormer. The behaviors of the above four quantities are similar across different layers.