

459 **A Architecture of Cross-Attention (CA) Layer**

460 The CA layer consists of H attention heads, taking x as input and output x_C .

$$x_C = x + \text{CA}(\text{norm}(x)), \quad x_n = \text{norm}(x), \quad \text{CA}(x_n) = \sum_{i=1}^H \text{Att}^{(i)}(x_e, x_n), \quad (28)$$

$$\text{Att}^{(i)}(x_e, x_n) = W_O^{(i)\top} W_V^{(i)} x_e \cdot \text{Softmax}((W_K^{(i)} x_e)^\top (W_Q^{(i)} x_n) / \sqrt{d}), \quad (29)$$

461 where x_e is the outputs of the encoder, $\text{norm}(x)$ represents the normalization function such as
 462 LayerNorm or RMSNorm, and $W_Q, W_K, W_V, W_O \in \mathbb{R}^{d_h \times d}$ represent the parameters of the CA
 463 layer. These parameters correspond to the query, key, value, and output matrices, respectively. Here,
 464 d denotes the hidden size, and $d_h = d/H$ denotes the attention head size.

465 **B Proof of Equation 7**

466 Let X be a matrix formed by concatenating a series of submatrices in the following manner:

$$X = [L^{(1)}(x_1), \dots, L^{(1)}(x_M), L^{(1 \sim 2)}(x_1), \dots, L^{(1 \sim 2)}(x_M), L^{(1 \sim N)}(x_1), \dots, L^{(1 \sim N)}(x_M)]. \quad (30)$$

467 Then, we have,

$$\arg \min_P \mathbb{E}_{x \in \mathbb{D}} \left(\sum_{i=1}^N \|L^{(1 \sim i)}(x) - PP^\top L^{(1 \sim i)}(x)\|_F^2 \right) \quad \text{s.t.} \quad P^\top P = \mathbf{I} \quad (31)$$

$$\iff \arg \min_P \sum_{j=1}^M \left(\sum_{i=1}^N \|L^{(1 \sim i)}(x_j) - PP^\top L^{(1 \sim i)}(x_j)\|_F^2 \right) \quad \text{s.t.} \quad P^\top P = \mathbf{I} \quad (32)$$

$$\iff \arg \min_P \sum_{i=1}^N \sum_{j=1}^M \|L^{(1 \sim i)}(x_j) - PP^\top L^{(1 \sim i)}(x_j)\|_F^2 \quad \text{s.t.} \quad P^\top P = \mathbf{I} \quad (33)$$

$$\iff \arg \min_P \sum_{i=1}^N \sum_{j=1}^M \|X[(i-1) * M + j] - PP^\top X[(i-1) * M + j]\|_F^2 \quad \text{s.t.} \quad P^\top P = \mathbf{I} \quad (34)$$

$$\iff \arg \min_P \|X - PP^\top X\|_F^2 \quad P^\top P = \mathbf{I}, \quad (35)$$

468 where $M = |\mathbb{D}|$ is the number of the sampled data instances, $X[(i-1) * M + j] = L^{(1 \sim i)}(x_j)$
 469 represents the submatrix associated with the i -th layer and the j -th data instance.

470 Therefore, Eq. 6 is equivalent to the following equation,

$$\arg \min_P \|X - PP^\top X\|_F^2 \quad \text{s.t.} \quad P^\top P = \mathbf{I}. \quad (36)$$

471 The optimization problem associated with Eq. 36 corresponds to one of the formulations of Principal
 472 Component Analysis [48]. Solving this problem yields the matrix U , which is obtained through the
 473 Singular Value Decomposition (SVD) of matrix X .

474 **C Normalization Function Approximation**

475 **LayerNorm.** For LayerNorm, we have,

$$\text{norm}(x) = \gamma * \frac{x - \mathbb{E}(x)}{\sqrt{\text{Var}(x) + \epsilon}} + \beta. \quad (37)$$

476 Then, we have

$$\text{norm}(P\hat{x}) = \gamma * \frac{P\hat{x} - \mathbb{E}(P\hat{x})}{\sqrt{\text{Var}(P\hat{x}) + \epsilon}} + \beta \quad (38)$$

$$\approx \hat{P} * (\gamma_n * \frac{\hat{x} - \mathbb{E}(\hat{x})}{\sqrt{\text{Var}(\hat{x}) + \epsilon}} + \beta_n) + \hat{b} \quad (39)$$

$$\approx \hat{P} * \text{norm}(\hat{x}) + \hat{b}, \quad (40)$$

477 where $\hat{P} = \gamma * P * \gamma_n^{-1}$, $\hat{b} = \beta - P_n * \beta_n$, and γ_n, β_n can take any value as needed. Empirically,
 478 we find that replacing $\mathbb{E}(P\hat{x})$, $\text{Var}(P\hat{x})$ with $P\mathbb{E}(\hat{x})$, $\text{Var}(\hat{x})$ does not affect the model performance.
 479 Here, γ and β are learnable parameters of LayerNorm, $\mathbb{E}(x)$ represents the mean of x over hidden
 480 units, and $\text{Var}(x)$ represents the variance of x over hidden units.

481 **BatchNorm.** For BatchNorm, we have

$$\text{norm}(x) = \gamma * \frac{(x - x_{\text{mean}})}{\sqrt{x_{\text{var}} + \epsilon}} + \beta. \quad (41)$$

482 Then, we have,

$$\text{norm}(P\hat{x}) = \gamma * \frac{(P\hat{x} - x_{\text{mean}})}{\sqrt{x_{\text{var}} + \epsilon}} + \beta \quad (42)$$

$$= \hat{P} * (\gamma_n * \frac{\hat{x} - 0}{1} + \beta_n) + \hat{b} \quad (43)$$

$$= \hat{P} * \text{norm}(\hat{x}) + \hat{b}, \quad (44)$$

483 where $\hat{P} = \frac{\gamma}{\sqrt{x_{\text{var}} + \epsilon}} P \gamma_n^{-1}$, $\hat{b} = \beta - \frac{\gamma}{\sqrt{x_{\text{var}} + \epsilon}} x_{\text{mean}} - \hat{P} \beta_n$, and γ_n, β_n can take any value as needed.
 484 Here, γ and β are learnable parameters of BatchNorm, and x_{mean} and x_{var} are the mean and variance
 485 of each dimension over data instances.

486 D Transformer Compression with Post-Normalization

487 For the transformer model with a post-normalization function (e.g., BERT), each layer of the model
 488 is formalized as:

$$L_n^{(l)}(x_n) = \text{norm}^{(l)}(L^{(l)}(x_n)), \quad L^{(l)}(x_n) = x_n + \text{Layer}(x_n), \quad x_n = \text{norm}^{(l-1)}(x), \quad (45)$$

489 where l denotes the index of layer. We further define,

$$L^{(1 \sim i)}(x) = L^{(i)} \circ \text{norm}^{(i-1)} \dots \text{norm}^{(1)} \circ L^{(1)} \circ \text{norm}^{(0)}(x) \quad (46)$$

$$L_n^{(1 \sim i)}(x) = \text{norm}^{(i)} \circ L^{(i)} \circ \text{norm}^{(i-1)} \dots \text{norm}^{(1)} \circ L^{(1)} \circ \text{norm}^{(0)}(x). \quad (47)$$

490 Then given an input x , we pass it through the transformer model starting from the first layer to the
 491 last N -th layer, resulting in the corresponding feature vector,

$$L_n^{(1 \sim N)}(x) = \text{norm}^{(N)} \circ L^{(N)} \circ \text{norm}^{(N-1)} \dots \text{norm}^{(1)} \circ L^{(1)} \circ \text{norm}^{(0)}(x). \quad (48)$$

492 According to Eq. 6, we can use the projection matrix P to add dimensionality reduction and dimen-
 493 sionality enhancement operations between each layer, while preserving the final output, as shown in
 494 the following equation,

$$L_n^{(1 \sim N)}(x) \approx \text{norm}^{(N)} \circ U \circ D \circ L^{(N)} \dots \text{norm}^{(1)} \circ L^{(1)} \circ U \circ D \circ \text{norm}^{(0)}(x) \quad (49)$$

$$\approx \hat{P}^{(N)}(\text{no}\hat{\text{rm}}^{(N)} \circ \hat{L}^{(N)} \dots \text{no}\hat{\text{rm}}^{(1)} \circ \hat{L}^{(1)} \circ \text{no}\hat{\text{rm}}^{(0)}(P^\top x)) \quad (50)$$

$$\approx \hat{P}^{(N)} \hat{L}_n^{(1 \sim N)}(P^\top x), \quad (51)$$

495 where $U(x) = Px$ represents the dimensionality enhancement operation, $D(x) = P^\top x$ represents
 496 the dimensionality reduction operation, $\hat{L}^{(i)} \circ \text{no}\hat{\text{rm}}^{(i-1)} = D \circ L^{(i)} \circ \text{norm}^{(i-1)} \circ U$ is the projected
 497 layer, and $\hat{L}_n^{(1 \sim N)} = \text{no}\hat{\text{rm}}^{(N)} \circ \hat{L}^{(N)} \dots \text{no}\hat{\text{rm}}^{(1)} \circ \hat{L}^{(1)} \circ \text{no}\hat{\text{rm}}^{(0)}$ is the projected model.

498 For a projected layer $\hat{L}^{(i)} \circ \text{no}\hat{\text{rm}}^{(i-1)}(\hat{x})$, we have,

$$\hat{L}^{(i)} \circ \text{no}\hat{\text{rm}}^{(i-1)}(\hat{x}) = D \circ L^{(i)} \circ \text{norm}^{(i-1)} \circ U(\hat{x}) \quad (52)$$

$$= P^\top (\text{norm}^{(i-1)}(P\hat{x}) + \text{Layer}(\text{norm}^{(i-1)}(P\hat{x}))). \quad (53)$$

499 According to appendix C, by the approximation of the post-normalization function, we can replace
 500 $\text{norm}(P\hat{x})$ with $\hat{P}\text{no}\hat{\text{rm}}(\hat{x})$ (For the sake of simplicity, we ignore bias here). Then we have,

$$\hat{L}^{(i)} \circ \text{no}\hat{\text{rm}}^{(i-1)}(\hat{x}) \approx P^\top (\hat{P}^{(i-1)} \text{no}\hat{\text{rm}}^{(i-1)}(\hat{x}) + \text{Layer}(\hat{P}^{(i-1)} \text{no}\hat{\text{rm}}^{(i-1)}(\hat{x}))). \quad (54)$$

501 With matrix fusion, we can compress the Layer function to $\hat{\text{L}}\text{ayer}$, i.e.,

$$\hat{L}^{(i)} \circ \text{no}\hat{\text{rm}}^{(i-1)}(\hat{x}) \approx P^\top \hat{P}^{(i-1)} \text{no}\hat{\text{rm}}^{(i-1)}(\hat{x}) + \hat{\text{L}}\text{ayer}(\text{no}\hat{\text{rm}}^{(i-1)}(\hat{x})). \quad (55)$$

502 There is an additional matrix $P^\top \hat{P}^{(i-1)}$ in Eq. 55, where $P^\top \hat{P}^{(i-1)} = P^\top \gamma P \gamma_n^{-1}$. Since we can set
 503 the value of γ_n arbitrarily, we approximate $P^\top \gamma P$ with $\text{diag}(P^\top \gamma P)$ and set γ_n to $\text{diag}(P^\top \gamma P)$.
 504 Then, we have $P^\top \hat{P}^{(i-1)} \approx \mathbf{I}$. Therefore, we finally have,

$$\hat{L}^{(i)} \circ \text{no}\hat{\text{rm}}^{(i-1)}(\hat{x}) \approx \text{no}\hat{\text{rm}}^{(i-1)}(\hat{x}) + \hat{\text{L}}\text{ayer}(\text{no}\hat{\text{rm}}^{(i-1)}(\hat{x})). \quad (56)$$

505 In our experiments, we find that compressing the first layer of the post-normalization transformer
 506 results in a significant decline in performance. Therefore, when compressing the post-normalization
 507 transformer, we choose to exclude the compression of its first layer.

508 E CA Layer Compression

509 For CA Layer, we have,

$$\hat{\text{CA}}(\hat{x}_n) = \sum_{i=1}^H P^\top W_O^{(i)\top} W_V^{(i)} \hat{P}_e \hat{x}_e \cdot \text{Softmax}((W_K^{(i)} \hat{P}_e \hat{x}_e)^\top (W_Q^{(i)} \hat{P} \hat{x}_n) / \sqrt{d}) \quad (57)$$

$$= \sum_{i=1}^H \hat{W}_O^{(i)\top} \hat{W}_V^{(i)} \hat{x}_e \cdot \text{Softmax}((\hat{W}_K^{(i)} \hat{x}_e)^\top (\hat{W}_Q^{(i)} \hat{x}_n) / \sqrt{d}), \quad (58)$$

510 where $\hat{W}_O^{(i)} = W_O^{(i)} P$, $\hat{W}_V^{(i)} = W_V^{(i)} \hat{P}_e$, $\hat{W}_K^{(i)} = W_K^{(i)} \hat{P}_e$, $\hat{W}_Q^{(i)} = W_Q^{(i)} \hat{P}$. The shape of these
 511 matrices is changed from $d_h \times d$ to $d_h \times k$, retaining k/d of the original number of parameters.

F Experiment Details

F.1 Experiment Setup

To establish the baseline models, we first download the pre-trained checkpoints from the HuggingFace Transformers repository [49]. For BERT, we conduct fine-tuning on the pre-trained model for 2 epochs, employing a batch size of 16 and a learning rate of $3e-5$ for tasks in the GLUE [41] and SQuAD [42, 50] benchmarks. For T5, we fine-tune the pre-trained model for 2 epochs, using a batch size of 16 and learning rates of $1e-4$ and $3e-5$ for tasks in the GLUE [41] and SQuAD [42, 50] benchmarks, respectively. Then, we employ a sample size of 2,000 to compress the transformer model. Finally, we refine the model weight using the same settings utilized during the fine-tuning of the baseline models. Other parameters are set to the default parameters provided by the framework.

F.2 Datasets

GLUE benchmark [41] consists of various tasks related to sentence similarity calculation, sentence classification, textual entailment, and natural language inference. It includes 10 tasks, namely AX, COLA, QQP, MNLI, MRPC, QNLI, QQP, RTE, SST-2, STS-B, and WNLI. The number of training examples for each task is as follows: 1.1k, 10.7k, 432k, 5.8k, 105k, 364k, 3k, 70k, 67k, 852, respectively. For our experiment, we select the datasets QQP, MNLI, MRPC, QNLI, QQP, SST-2, and STS-B based on the data scale and previous research [8]. The SQuAD 1.1 [42] and SQuAD 2.0 [50] datasets involve question and answering tasks, each containing 88K and 130K training examples, respectively.

F.3 Comparison of TCSP with Prior Methods on BERT

Following the experimental setup of Kwon et al. [8], we compare TCSP against previous pruning methods and low-rank factorization methods for the transformer model on four GLUE tasks: QQP, QNLI, SST-2, and MRPC. It is important to note that our evaluation solely relies on the experimental results without any additional knowledge distillation. For DRONE, as the paper only reported the speed-up metric, the compression ratio is not provided. Additionally, the results for MRPC in Sajjad et al., DynaBERT, Kwon et al., FWSVD, and TFWSVD are reported in terms of accuracy instead of F1 score. Please refer to Table 6 for the detailed experimental outcomes. Our method TCSP demonstrates comparable or superior results compared to the prior methods.

F.4 Performance Comparison of Compressed T5-base with T5-small

We conduct a comparison between the compressed T5-base and T5-small models. As shown in Table 7, the compressed T5-base, having twice the number of parameters, outperforms T5-small on all datasets. In the future, we aim to delve deeper into the layer compression algorithm to ensure that the compressed T5-base and T5-small models have an equal number of layers and hidden size. This will enable a fairer and more accurate comparison between the two models.

F.5 Performance Analysis of Compressed T5-base at Various Compression Rates

Table 8 illustrates the performance of the compressed T5-base model at different compression ratios. Notably, a reduction of 25% in the hidden size can be achieved while preserving accuracy.

F.6 Performance of Compressed LLaMa

We extend the application of TCSP to large transformer models, such as LLaMa-7B. Initially, We conduct LoRA-based fine-tuning on LLaMa-7B using the Stanford Alpaca dataset. After that, we apply TCSP to compress the transformer model. To maintain the performance, we also need to conduct LoRA-based fine-tuning on the Alpaca dataset after the compression process. To evaluate the performance of the compressed model in a zero-shot setting, we test it on three commonsense reasoning datasets: PIQA, HellaSwag, and WinoGrande. The accuracy of the model is reported for these three datasets. The experimental results shown in Table 9 demonstrate that we can compress the model by 12.5% of its parameters while incurring only a 3% degradation in performance.

Table 6: Performance comparison of TCSP with previous compression methods on BERT.

	Compression rate	QQP	QNLI	SST-2	MRPC	Diff.			
Flop [9]	baseline	-	91.6	92.7	90.9	-	0	0	0
	33.3	-	89.0	92.1	88.6	-	-2.6	-0.6	-2.3
SLIP [10]	baseline	90.6	91.6	92.7	90.9	0	0	0	0
	34.4	89.7	90.7	91.7	89.9	-0.9	-0.9	-1.0	-1.0
	38.5	88.9	89.5	91.8	88.1	-1.7	-2.1	-0.9	-2.8
Sajjad et al. [47]	baseline	91.1	91.1	92.4	88.0*	0	0	0	0
	33.3	90.6	89.7	90.6	79.4*	-0.4	-1.4	-1.8	-8.6
	50.0	90.4	87.6	90.3	80.2*	-0.7	-3.5	-2.2	-7.8
DynaBERT [23]	baseline	-	-	92.9	87.7*	-	-	0	0
	25.0	-	-	92.3	86.0*	-	-	-0.6	-1.7
	50.0	-	-	91.9	86.0*	-	-	-1.0	-1.7
EBERT [11]	baseline	87.9	91.5	93.2	-	0	0	0	-
	40.0	87.5	90.2	92.2	-	-0.4	-1.3	-1.0	-
	50.0	87.9	89.6	91.6	-	-0.7	-1.9	-1.6	-
BMP [12]	baseline	91.1	-	92.7	-	0	-	0	-
	50.0	90.4	-	90.7	-	-0.7	-	-2.0	-
Kwon et al. [8]	baseline	91.0	91.4	93.6	86.3*	0	0	0	0
	30.0	90.7	90.9	93.0	86.1*	-0.3	-0.5	-0.6	-0.2
	40.0	90.4	90.0	92.5	85.3*	-0.6	-1.4	-1.1	-1.0
	50.0	89.5	88.7	91.6	83.2*	-1.5	-2.7	-2.0	-3.1
DRONE [14]	baseline	90.9	91.4	92.3	89.5	0	0	0	0
	-	90.1	89.3	90.8	88.0	-0.8	-2.1	-1.5	-1.5
FWSVD [15]	baseline	87.8	91.3	93.0	87.4*	0	0	0	0
	40.0	87.6	89.5	91.2	88.0*	-0.2	-1.8	-1.8	+0.6
TFWSVD [16]	baseline	87.8	91.3	93.0	87.4*	0	0	0	0
	40.0	86.9	90.3	91.1	89.0*	-0.9	-1.0	-1.9	+1.6
TCSP	baseline	91.1	91.4	92.2	89.9	0	0	0	0
	40.0	90.8	90.6	91.1	89.1	-0.3	-0.8	-1.1	-0.8

Table 7: Comparison of compressed T5-base with T5-small

	MNLI	QQP	QNLI	SST-2	STS-B	MRPC	SQuAD _{1.1}	SQuAD _{2.0}
T5-small	81.6	89.5	90.7	91.5	87.4	91.0	83.3	63.9
T5-base w TCSP {25%, 25%} + ft.	86.1	90.9	92.0	93.2	89.3	92.3	87.0	77.5

G Algorithm

G.1 Projection Matrix Generation

The pseudo-code of projection matrix generation is shown in Algorithm 1.

G.2 TCSP

The pseudo-code of TCSP is shown in Algorithm 2. The pseudo-code of TCSP combined with other compression methods is shown in Algorithm 3.

Table 8: Performance of compressed T5-base at different compression ratios

	MNLI	QQP	QNLI	SST-2	STS-B	MRPC	SQuAD _{1.1}	SQuAD _{2.0}
T5-base	86.8	91.4	93.2	94.5	90.0	91.9	88.6	79.3
w TCSP {12.5%, 0%} + ft.	86.5	91.2	92.6	94.4	90.3	91.0	87.2	78.1
w TCSP {25%, 0%} + ft.	86.2	91.2	92.5	93.2	90.1	91.3	86.8	78.0
w TCSP {37.5%, 0%} + ft.	84.8	90.6	92.1	92.4	89.3	91.7	86.0	76.0
w TCSP {50%, 0%} + ft.	82.6	90.5	89.9	90.9	83.4	83.4	84.4	73.9

Table 9: Performance of compressed LLaMa

	PIQA	HellaSwag	WinoGrande
LLaMa-7B+LoRa-ft.	77.4	73.9	63.2
+TCSP{12.5%, 0%}	74.3	68.0	60.7
+TCSP{12.5%, 0%}+LoRa-ft.	75.4	71.1	61.0
+TCSP{25.0%, 0%}	68.4	62.6	56.0
+TCSP{25.0%, 0%}+LoRa-ft.	72.7	63.3	59.6

G.3 TCSP-filter-pruning

The pseudo-code of TCSP-filter-pruning is shown in Algorithm 4.

G.4 TCSP-head-size-compression

The pseudo-code of TCSP-head-size-compression is shown in Algorithm 5.

H Societal Impacts

We believe that our work will not have an immediate negative impact on society, as its primary objective is to accelerate model inference without compromising the output quality.

I Limitation and Future Work

There are two key considerations regarding TCSP. Firstly, it relies on performing SVD on the feature matrix for model compression. However, the computational overhead associated with SVD and the storage requirements for the feature matrix using a limited number of samples. Secondly, our current method exclusively supports compressing of models employing the Transformer structure. In the future, we intend to delve into alternative techniques for computing the projection matrix, offering greater flexibility for handling large-scale matrices. Additionally, we aim to extend the application of our compression methods to encompass diverse model structures.

Algorithm 1: Projection Matrix Generation (PMG)

Input: r : compression ratio, T : sampled tokens size, \mathbb{D} : a subset of training data, $L^{(1 \sim N)}$:
model, d : hidden size
Output: Projection Matrix P
 $X \leftarrow []$
for $m \leftarrow 0$ **to** M **do**
 $x^{(0)} \leftarrow \mathbb{D}[m]$
 for $i \leftarrow 1$ **to** N **do**
 $x^{(i)} \leftarrow L^{(i)}(x^{(i-1)})$
 $x_s^{(i)} = \text{Sample}(x^{(i)}, T)$
 $X \leftarrow \text{Concate}(X, x_s^{(i)})$
 end
end
 $U, \Sigma, V^\top \leftarrow \text{SVD}(X)$
 $k \leftarrow \lceil d * r \rceil$
 $P \leftarrow U_{:, :k}$

Algorithm 2: TCSP

Input: r : compression ratio, T : sampled tokens size, \mathbb{D} : a subset of the training data
Output: Compressed model $\hat{L}^{(1 \sim N)}$
 $P \leftarrow \text{PMG}(r, T, \mathbb{D})$
for $i \leftarrow 1$ **to** $|MHA|$ **do**
 for $j \leftarrow 1$ **to** H **do**
 $[\gamma, W_O, W_V, W_Q, W_K] \leftarrow MHA_j^{(i)}$ \triangleright Load MHA Parameters
 $\hat{P} \leftarrow \sqrt{\frac{d}{k}} \gamma P$
 $\hat{MHA}_j^{(i)} \leftarrow [\mathbf{I}, W_O P, W_V \hat{P}, W_Q \hat{P}, W_K \hat{P}]$ \triangleright Set MHA Parameters
 end
end
for $i \leftarrow 1$ **to** $|FFN|$ **do**
 $[\gamma, W_D, W_U] \leftarrow FFN^{(i)}$ \triangleright Load FFN Parameters
 $\hat{P} \leftarrow \sqrt{\frac{d}{k}} \gamma P$
 $\hat{FFN}^{(i)} \leftarrow [\mathbf{I}, W_D P, W_U \hat{P}]$ \triangleright Set FFN Parameters
end
 $\hat{M} \leftarrow [\hat{MHA}, \hat{FFN}]$

Algorithm 3: TCSP-combine

Input: r : compression ratio, T : sampled tokens size, \mathbb{D} : a subset of the training data

Output: Compressed model $\hat{L}^{(1 \sim N)}$

$P \leftarrow \text{PMG}(r, T, \mathbb{D})$

$\text{Mask} \leftarrow \text{TFP}(r, T, \mathbb{D})$

$U_M, V_M, U_{M'}, V_{M'} \leftarrow \text{THSC}(r, T, \mathbb{D})$

for $i \leftarrow 1$ **to** $|\text{MHA}|$ **do**

for $j \leftarrow 1$ **to** H **do**

$[\gamma, W_O, W_V, W_Q, W_K] \leftarrow \text{MHA}_j^{(i)}$

$\hat{P} \leftarrow \sqrt{\frac{d}{k}} \gamma P$

$\hat{\text{MHA}}_j^{(i)} \leftarrow [\mathbf{I}, V_{M',j}^{(i)\top} W_O P, U_{M',j}^{(i)\top} W_V \hat{P}, V_{M,j}^{(i)\top} W_Q \hat{P}, U_{M,j}^{(i)\top} W_K \hat{P}]$

end

end

for $i \leftarrow 1$ **to** $|\text{FFN}|$ **do**

$[\gamma, W_D, W_U] \leftarrow \text{FFN}^{(i)}$

$\hat{P} \leftarrow \sqrt{\frac{d}{k}} \gamma P$

$\hat{\text{FFN}}^{(i)} \leftarrow [\mathbf{I}, \text{Mask}^{(i)} W_D P, \text{Mask}^{(i)} W_U \hat{P}]$

end

$\hat{L}^{(1 \sim N)} \leftarrow [\hat{\text{MHA}}, \hat{\text{FFN}}]$

Algorithm 4: TCSP-filter-pruning (TFP)

Input: r : compression ratio, T : sampled tokens size, \mathbb{D} : a subset of training data, $L^{(1 \sim N)}$: model, d_f : number of filters

Output: TCSP Pruning Mask Mask

$\mathbb{E}[\mathbf{X}]_* \leftarrow 0$

$\mathbb{E}[\mathbf{X}^2]_* \leftarrow 0$

for $m \leftarrow 1$ **to** M **do**

$x^{(0)} \leftarrow \mathbb{D}[m]$

for $i \leftarrow 1$ **to** N **do**

$x^{(i)} \leftarrow L^{(i)}(x^{(i-1)})$

if $L^{(i)}$ is j -th FFN Layer $\text{FFN}^{(j)}$ **then**

$\mathbb{E}[\mathbf{X}]_j \leftarrow \text{Update}(\mathbb{E}[\mathbf{X}]_j, x^{(i)})$

$\mathbb{E}[\mathbf{X}^2]_j \leftarrow \text{Update}(\mathbb{E}[\mathbf{X}^2]_j, x^{(i)} * x^{(i)})$

end

end

end

$K \leftarrow \lceil d_f * r \rceil$

for $i \leftarrow 1$ **to** $|\text{FFN}|$ **do**

$\mathbb{E}_i \leftarrow \mathbb{E}[\mathbf{X}]_i$

$\text{std}_i \leftarrow \sqrt{\mathbb{E}[\mathbf{X}^2]_i - (\mathbb{E}[\mathbf{X}]_i)^2}$

$\text{Score}(\cdot) \leftarrow |W_{D,*}|(\mathbb{E}_{i,*} + \text{std}_{i,*})$

$\text{Index} \leftarrow \text{TopK_Index}(\text{Score}, K)$

$\text{Mask}_j \leftarrow \text{GenerateMask}(d_f, \text{Index})$

\triangleright Select Top-K important filters

$\triangleright \text{Mask}_j$ is a $\mathbb{R}^{K \times d_f}$ matrix for pruning filters

end

$\text{Mask} \leftarrow [\text{Mask}_1, \text{Mask}_2, \dots, \text{Mask}_{|\text{FFN}|}]$

Algorithm 5: TCSP-head-size-compression (THSC)

Input: r : compression ratio, T : sampled tokens size, \mathbb{D} : a subset of training data, $L^{(1 \sim N)}$: model, d : hidden size

Output: Projection Matrices $U_M, V_M, U_{M'}, V_{M'}$

for $i \leftarrow 1$ **to** $|MHA|$ **do**

$X_K^{(i)} \leftarrow \square$

$X_Q^{(i)} \leftarrow \square$

$X_V^{(i)} \leftarrow \square$

end

for $m \leftarrow 0$ **to** M **do**

$x^{(0)} \leftarrow \mathbb{D}[m]$

for $i \leftarrow 1$ **to** N **do**

$x^{(i)} \leftarrow L^{(i)}(x^{(i-1)})$

if $L^{(i)}$ is j -th MHA Layer $MHA^{(j)}$ **then**

$x_s^{(i)} = \text{Sample}(x^{(i)}, T)$

$X_Q^{(j)} \leftarrow \text{Concate}(X_Q^{(i)}, W_Q^{(j)} x_s^{(i)})$

$X_K^{(j)} \leftarrow \text{Concate}(X_K^{(i)}, W_K^{(j)} x_s^{(i)})$

$X_V^{(j)} \leftarrow \text{Concate}(X_V^{(i)}, W_V^{(j)} x_s^{(i)})$

end

end

end

for $i \leftarrow 1$ **to** N **do**

if $L^{(i)}$ is j -th MHA Layer $MHA^{(j)}$ **then**

\triangleright Solve Eq.26 and Eq.27 for each head

for $k \leftarrow 1$ **to** H **do**

$U_{M,k}^{(j)}, V_{M,k}^{(j)}, U_{M',k}^{(j)}, V_{M',k}^{(j)} \leftarrow \text{Solve}_k(X_Q^{(j)}, X_K^{(j)}, X_V^{(j)})$

end

end

end

$U_M \leftarrow [U_M^{(1)}, U_M^{(2)}, \dots, U_M^{(|MHA|)}]$

$V_M \leftarrow [V_M^{(1)}, V_M^{(2)}, \dots, V_M^{(|MHA|)}]$

$U_{M'} \leftarrow [U_{M'}^{(1)}, U_{M'}^{(2)}, \dots, U_{M'}^{(|MHA|)}]$

$V_{M'} \leftarrow [V_{M'}^{(1)}, V_{M'}^{(2)}, \dots, V_{M'}^{(|MHA|)}]$
