Transformer Architecture Implementation Details

Forward & Backward Pass Analysis

with Tensor Parallelism

November 4, 2025

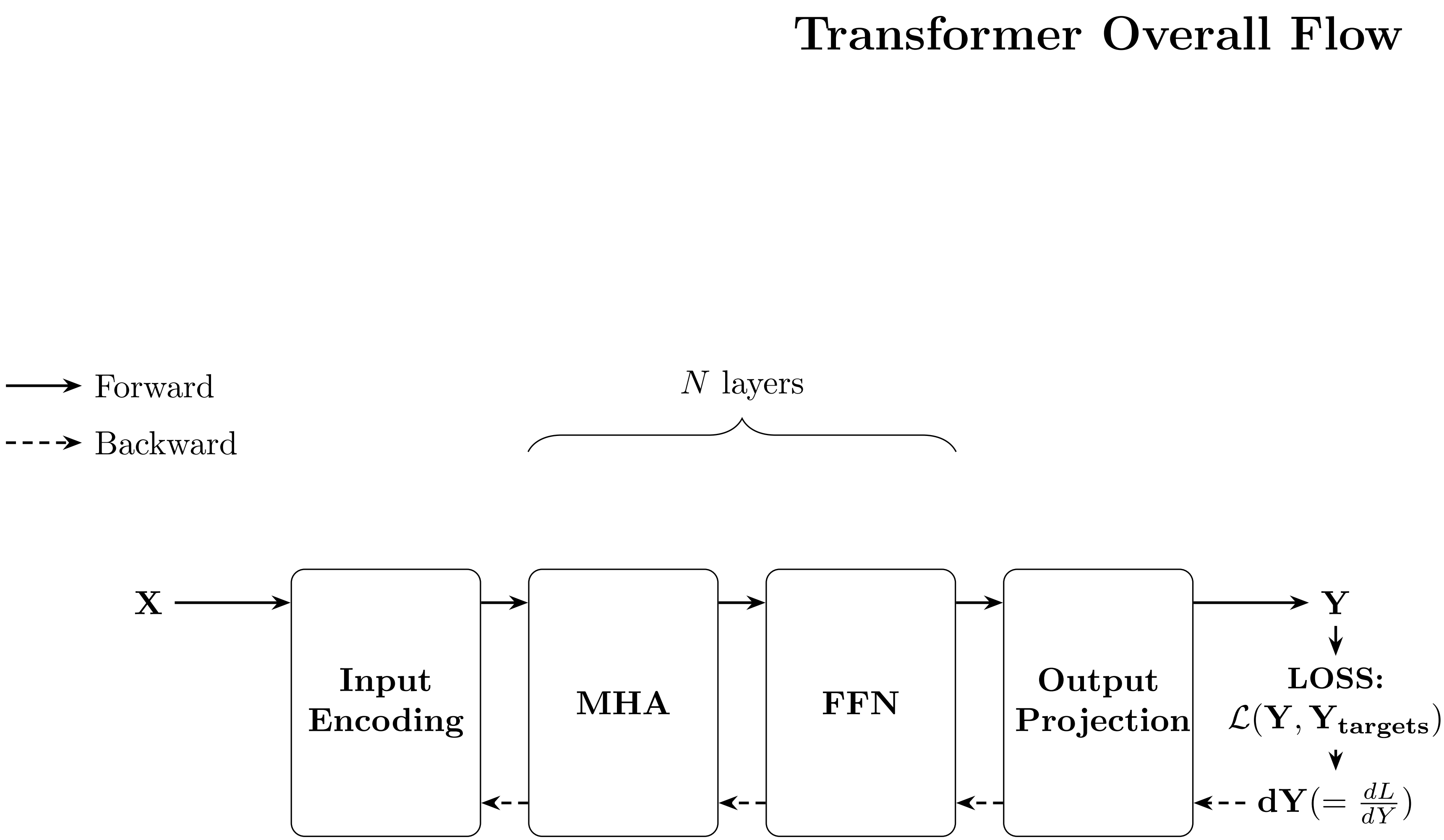
Contents

1 Single Node Transformer

This chapter covers the complete forward and backward pass of a Transformer model running on a single node (no parallelism).

1.1 Overall Architecture & Data Flow

The following diagram shows the high-level architecture of a Transformer layer, including both forward and backward passes.

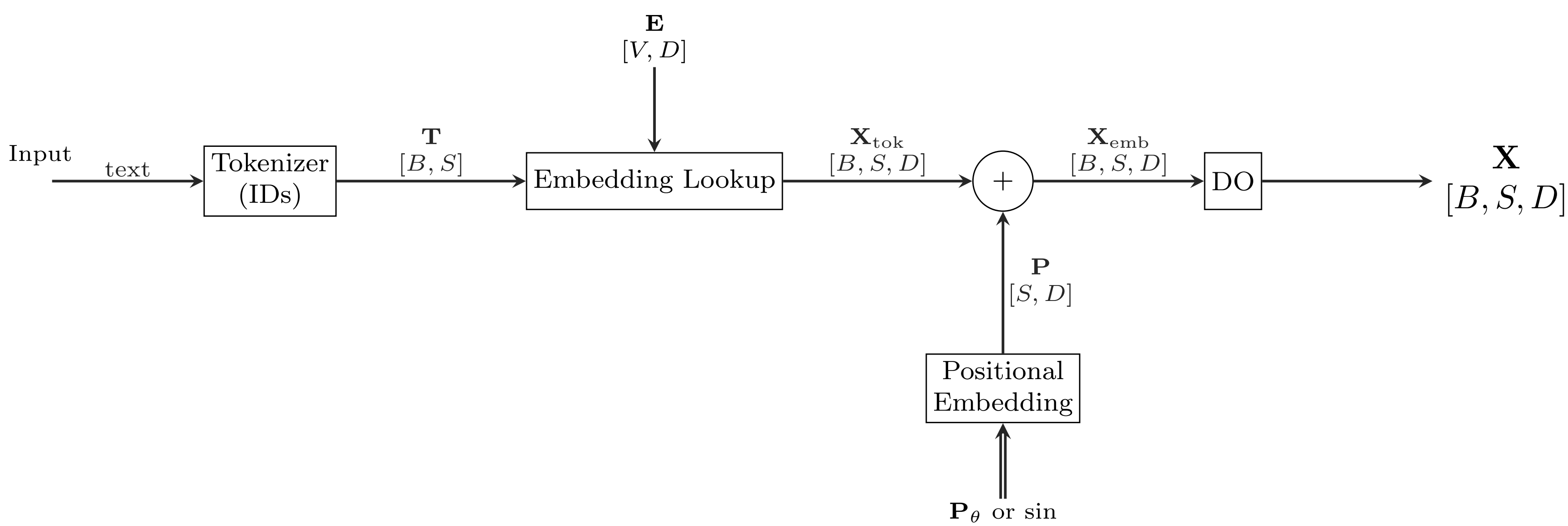


1.2 Input Embedding Layer

The input embedding layer converts token indices into dense vector representations and adds positional encodings.

1.2.1 Forward Pass

Input → Embedding → LN (Input to MHA)



1.2.2 Operations Summary

Operations (Ops)

Abbrev	Name	Type / Shape	Notes
Tokenizer	Tokenizer (IDs)	op	Maps raw text $\to$ integer ids $\mathbf{T} \in \mathbb{Z}^{[B,S]}$.
Embedding Lookup	Embedding Lookup	op	Gathers rows from $\mathbf{E} \in \mathbb{R}^{V \times D}$ using ids $\mathbf{T}$.
$+$	Element-wise Add (dashed circle)	op	Adds token and positional embeddings; broadcasting over B, S if needed.
DO	Dropout	op	Training-time stochastic dropout on $\mathbf{X}_{\text{emb}}$; identity at inference.
(none)	Broadcast $\mathrm{BC}_{B,S}(\cdot)$	op	Expands $[S, D]$ (or $[D]$) to $[B, S, D]$ across batch/sequence.

Data Tensors (Values)

Symbol	Name	Shape	Notes
text	Raw input text	—	Character/byte stream before tokenization.
$\mathbf{T}$	Token ids	$[B, S]$	Output of Tokenizer; integers in $\{0, \dots, V-1\}$.
$\mathbf{E}$	Embedding matrix (params)	$[V, D]$	Trainable; each vocab entry has a D-dim vector.
$\mathbf{X}_{\text{tok}}$	Token embeddings	$[B, S, D]$	$\mathrm{lookup}(\mathbf{E}, \mathbf{T})$.
$\mathbf{P}$	Positional embedding	$[S, D]$ (or $[B, S, D]$)	Learned $\mathbf{P}_\theta$ **or** sinusoidal (fixed); broadcast to $[B, S, D]$.
$\mathbf{X}_{\text{emb}}$	Sum of token+pos	$[B, S, D]$	$\mathbf{X}_{\text{tok}} + \mathrm{BC}_{B,S}(\mathbf{P})$.
$\mathbf{X}$	Input to MHA	$[B, S, D]$	After dropout (DO); goes to LN/MHA stack.
$\mathbf{P}_\theta$	Learned pos. params	matches $\mathbf{P}$	Used when positions are trainable; otherwise "sin" denotes fixed sinusoidal.

Shape symbols: B=batch size, S=sequence length, D=model dim, V=vocab size.

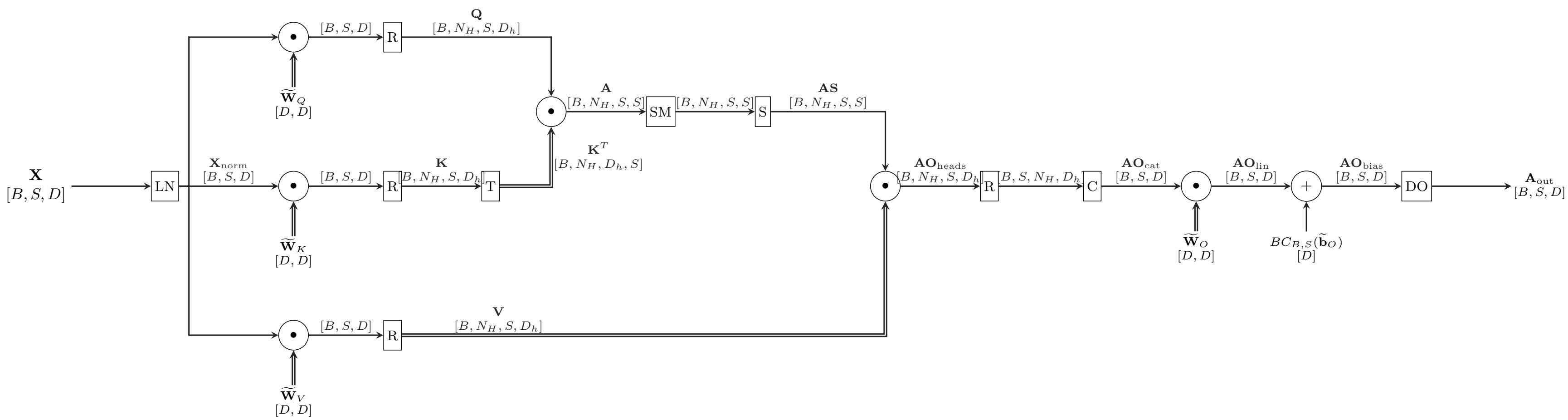
Notes: In practice, $\mathbf{P}$ may be pre-broadcast to $[B, S, D]$ or added per-token with implicit broadcasting.

1.3 Multi-Head Attention (MHA)

Multi-Head Attention enables the model to jointly attend to information from different representation subspaces.

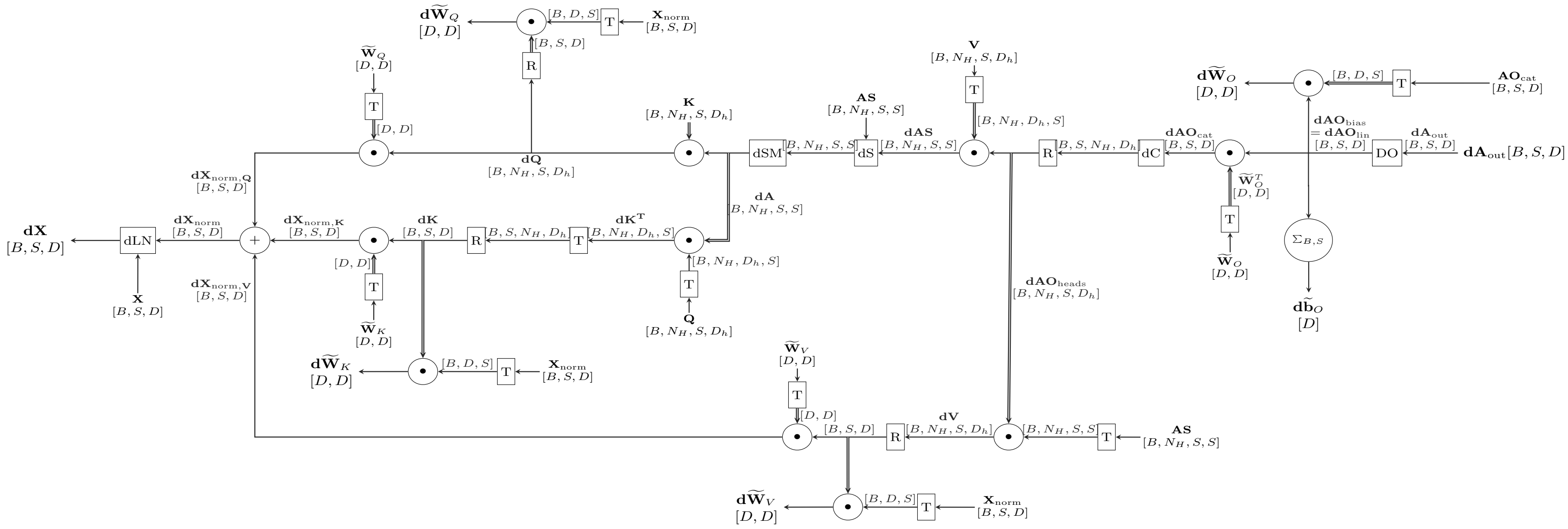
1.3.1 Forward Pass

Multi-Head Attention Forward Pass



1.3.2 Backward Pass

Multi-Head Attention Backward Pass



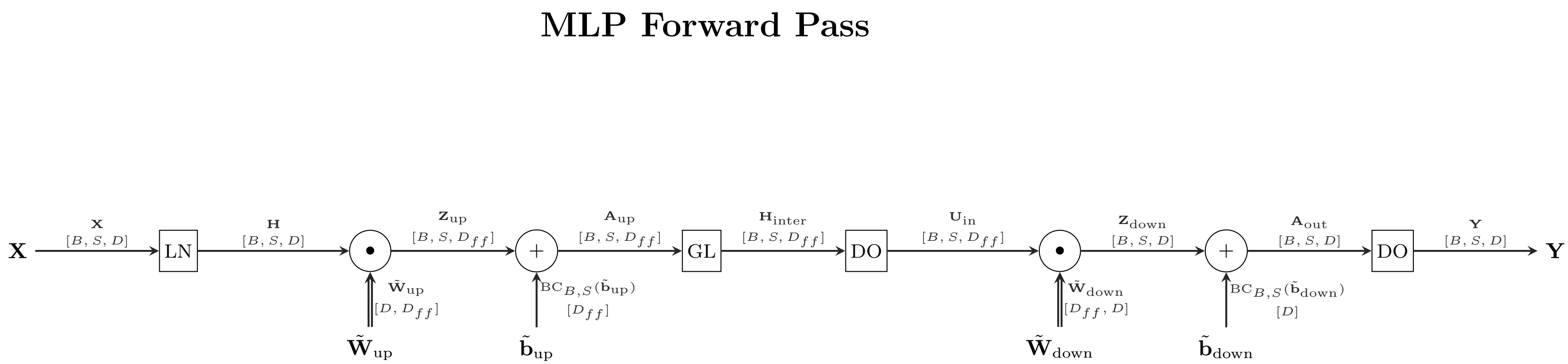
1.3.3 Operations Summary

Category	Symbol / Abbrev	Name	Shape / Type	Notes
Ops	LN	Layer Normalization	op	Normalizes per token (last dim D).
Ops	DO	Dropout	op	Training-time only; identity at inference.
Ops	$+$	Bias Add	op	Adds broadcast bias; see $\mathrm{BC}_{B,S}(\cdot)$.
Ops	T	Transpose	op	e.g., $[B, N_H, S, D_h] \to [B, N_H, D_h, S]$.
Ops	R	Reshape / Split / Merge	op	$[B, S, D] \leftrightarrow [B, N_H, S, D_h]$.
Ops	C	Concatenate	op	$[B, S, N_H, D_h] \to [B, S, D]$.
Ops	SM	Scale (+ Mask)	op	Multiply by $1/\sqrt{D_h}$ and apply mask.
Ops	S	Softmax	op	Over key length S per head.
Ops	$\mathrm{BC}_{B,S}(\cdot)$	Broadcast	op	Broadcast length-D (or D_h) to $[B, S, \cdot]$.
Ops	dS	Softmax Backward	op	Backprop through softmax over S.
Ops	dSM	Scale/Mask Backward	op	Backprop through scaling/masking.
Ops	dC	De-concat (Backward)	op	Split grads from concatenated heads.
Ops	dLN	LayerNorm Backward	op	Uses cached stats (μ, σ) and $\mathbf{X}$.
Data	$\mathbf{X}$	Input hidden states	$[B, S, D]$	Into MHA block (pre-LN).
Data	$\mathbf{X}_{\text{norm}}$	LN output	$[B, S, D]$	Result of LN($\mathbf{X}$).
Data	$\mathbf{Q}, \mathbf{K}, \mathbf{V}$	Query/Key/Value	$[B, N_H, S, D_h]$	From linear projections of $\mathbf{X}_{\text{norm}}$.
Data	$\widetilde{\mathbf{W}}_Q$	Q weight	$[D, D]$	Per-head realized via reshape (drawn fused).
Data	$\widetilde{\mathbf{W}}_K$	K weight	$[D, D]$	Same convention.
Data	$\widetilde{\mathbf{W}}_V$	V weight	$[D, D]$	Same convention.
Data	$\widetilde{\mathbf{W}}_O$	Output-proj weight	$[D, D]$	Maps concatenated heads to model dim.
Data	$\widetilde{\mathbf{b}}_O$	Output bias	$[D]$	Broadcast via $\mathrm{BC}_{B,S}$.
Data	$\mathbf{A}$	Attention scores	$[B, N_H, S, S]$	$\mathbf{Q}\mathbf{K}^T/\sqrt{D_h}$ (plus mask).
Data	$\mathbf{AS}$	Attention weights	$[B, N_H, S, S]$	softmax($\mathbf{A}$).
Data	$\mathbf{AO}_{\text{heads}}$	Per-head outputs	$[B, N_H, S, D_h]$	$\mathbf{AS} \cdot \mathbf{V}$.
Data	$\mathbf{AO}_{\text{cat}}$	Concatenated heads	$[B, S, D]$	After C.
Data	$\mathbf{AO}_{\text{lin}}$	Linear output	$[B, S, D]$	$\mathbf{AO}_{\text{cat}}\widetilde{\mathbf{W}}_O$.
Data	$\mathbf{AO}_{\text{bias}}$	Bias-added output	$[B, S, D]$	$+\ \widetilde{\mathbf{b}}_O$.
Data	$\mathbf{A}_{\text{out}}$	MHA output	$[B, S, D]$	After dropout; to next sublayer.
Data	$\mathbf{dA}_{\text{out}}$	Grad wrt MHA output	$[B, S, D]$	Backprop signal entering MHA.
Data	$\mathbf{dQ}, \mathbf{dK}, \mathbf{dV}$	Gradients for Q/K/V	$[B, N_H, S, D_h]$	From attention-core backward.
Data	$\mathbf{dK^T}$	Grad of K^T	$[B, N_H, D_h, S]$	Before transpose/reshape to $\mathbf{dK}$.
Data	$\mathbf{dAO}_{\text{heads}}$	Grad at heads	$[B, N_H, S, D_h]$	Split from $\mathbf{dAO}_{\text{cat}}$.
Data	$\mathbf{dX}_{\text{norm},Q}$	Grad wrt X_{norm} (Q)	$[B, S, D]$	Via W_Q^T.
Data	$\mathbf{dX}_{\text{norm},K}$	Grad wrt X_{norm} (K)	$[B, S, D]$	Via W_K^T.
Data	$\mathbf{dX}_{\text{norm},V}$	Grad wrt X_{norm} (V)	$[B, S, D]$	Via W_V^T.
Data	$\mathbf{dX}_{\text{norm}}$	Sum of above	$[B, S, D]$	Input to dLN.
Data	$\mathbf{dX}$	Grad wrt input X	$[B, S, D]$	Output of dLN.
Data	$\mathbf{d}\widetilde{\mathbf{W}}_Q$	Q weight grad	$[D, D]$	Standard matmul rule.
Data	$\mathbf{d}\widetilde{\mathbf{W}}_K$	K weight grad	$[D, D]$	Standard matmul rule.

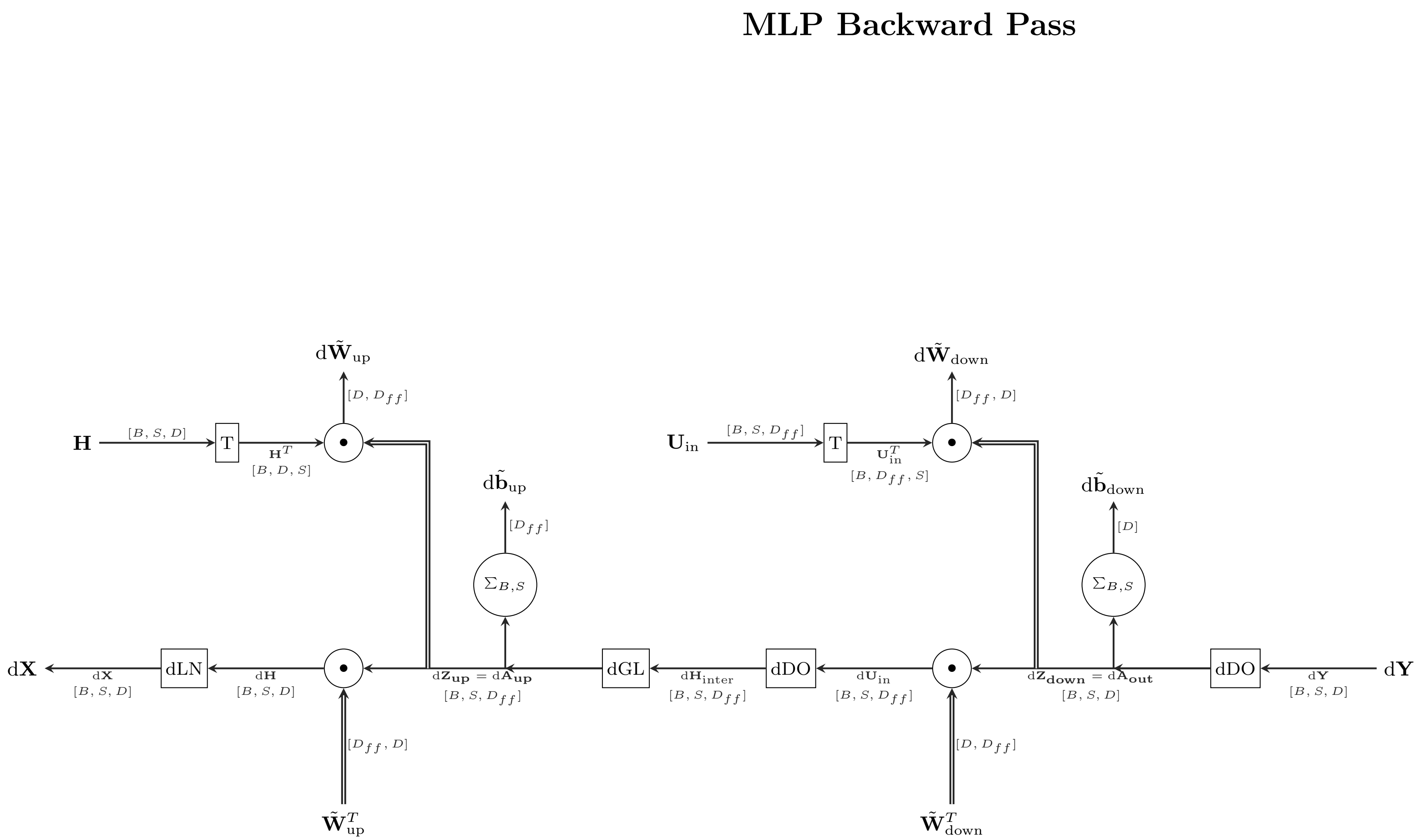
1.4 Feed-Forward Network (FFN/MLP)

The FFN consists of two linear transformations with a non-linear activation function in between.

1.4.1 Forward Pass



1.4.2 Backward Pass



1.4.3 Operations Summary

MLP (Feed-Forward) Block: Unified Table (Ops & Data)

Category	Symbol / Abbrev	Name	Shape / Type	Notes
Ops	LN	Layer Normalization	op	Normalize per token (last dim D).
Ops	$\bullet$	Linear (MatMul)	op	Used for up/down projections.
Ops	$+$	Bias Add	op	Adds broadcast bias; $\mathrm{BC}_{B,S}(\cdot)$.
Ops	GL	GELU (or activation)	op	Nonlinearity on D_{ff}.
Ops	DO	Dropout	op	Training-time only (identity at inference).
Ops	T	Transpose	op	Used in weight-grad computations.
Ops	$\sum_{B,S}$	Reduce-Sum	op	Bias-grad accumulation over batch, seq.
Data	$\mathbf{X}$	Input states	$[B,S,D]$	Block input.
Data	$\mathbf{H}$	LN output	$[B,S,D]$	$\mathrm{LN}(\mathbf{X})$.
Data	$\tilde{\mathbf{W}}_{\mathrm{up}}$	Up weight	$[D,D_{ff}]$	First projection.
Data	$\tilde{\mathbf{b}}_{\mathrm{up}}$	Up bias	$[D_{ff}]$	Broadcast to $[B,S,D_{ff}]$.
Data	$\mathbf{Z}_{\mathrm{up}}$	Pre-activation	$[B,S,D_{ff}]$	$HW_{\mathrm{up}}+b_{\mathrm{up}}$.
Data	$\mathbf{A}_{\mathrm{up}}$	Activated	$[B,S,D_{ff}]$	$f(\mathbf{Z}_{\mathrm{up}})$.
Data	$\mathbf{H}_{\mathrm{inter}}$	Post-DO	$[B,S,D_{ff}]$	After Dropout.
Data	$\tilde{\mathbf{W}}_{\mathrm{down}}$	Down weight	$[D_{ff},D]$	Second projection.
Data	$\tilde{\mathbf{b}}_{\mathrm{down}}$	Down bias	$[D]$	Broadcast to $[B,S,D]$.
Data	$\mathbf{Z}_{\mathrm{down}}$	Linear output	$[B,S,D]$	$H_{\mathrm{inter}}W_{\mathrm{down}}+b_{\mathrm{down}}$.
Data	$\mathbf{A}_{\mathrm{out}}$	Bias-added	$[B,S,D]$	Before dropout (out).
Data	$\mathbf{Y}$	Block output	$[B,S,D]$	After Dropout.
Data	$\mathrm{d}\mathbf{Y}$	Grad output	$[B,S,D]$	Incoming grad.
Data	$\mathrm{d}\mathbf{Z}_{\mathrm{down}}$	Grad lin-out	$[B,S,D]$	Equals $d\mathbf{A}_{\mathrm{out}}$.
Data	$\mathrm{d}\mathbf{U}_{\mathrm{in}}$	Grad into down	$[B,S,D_{ff}]$	To weight/bias grads.
Data	$\mathrm{d}\mathbf{Z}_{\mathrm{up}}$	Grad pre-act	$[B,S,D_{ff}]$	Equals $d\mathbf{A}_{\mathrm{up}}\cdot f'$.
Data	$\mathrm{d}\mathbf{H}$	Grad LN out	$[B,S,D]$	Into dLN.
Data	$\mathrm{d}\mathbf{X}$	Grad input	$[B,S,D]$	Block input grad.
Data	$\mathrm{d}\tilde{\mathbf{W}}_{\mathrm{up}}$	Weight grad up	$[D,D_{ff}]$	From H^T and dZ_{up}.
Data	$\mathrm{d}\tilde{\mathbf{W}}_{\mathrm{down}}$	Weight grad down	$[D_{ff},D]$	From U_{in}^T and dZ_{down}.
Data	$\mathrm{d}\tilde{\mathbf{b}}_{\mathrm{up}}$	Bias grad up	$[D_{ff}]$	$\sum_{B,S} dZ_{\mathrm{up}}$.
Data	$\mathrm{d}\tilde{\mathbf{b}}_{\mathrm{down}}$	Bias grad down	$[D]$	$\sum_{B,S} dZ_{\mathrm{down}}$.

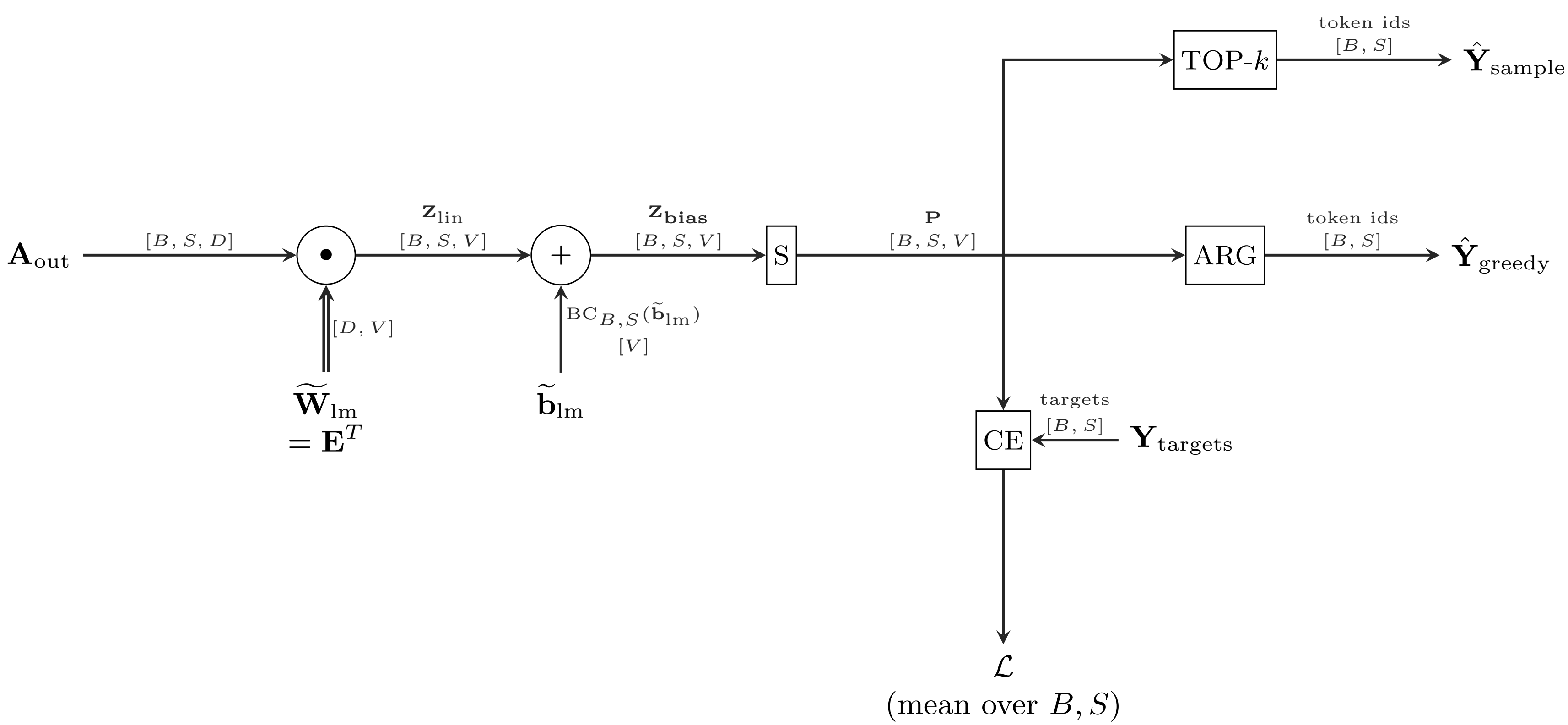
Shape symbols: B=batch, S=sequence, D=model dim, D_{ff}=FFN hidden dim (e.g., $4\times D$).

1.5 Output Projection & Loss Computation

The final layer projects the hidden states to vocabulary logits and computes the cross-entropy loss.

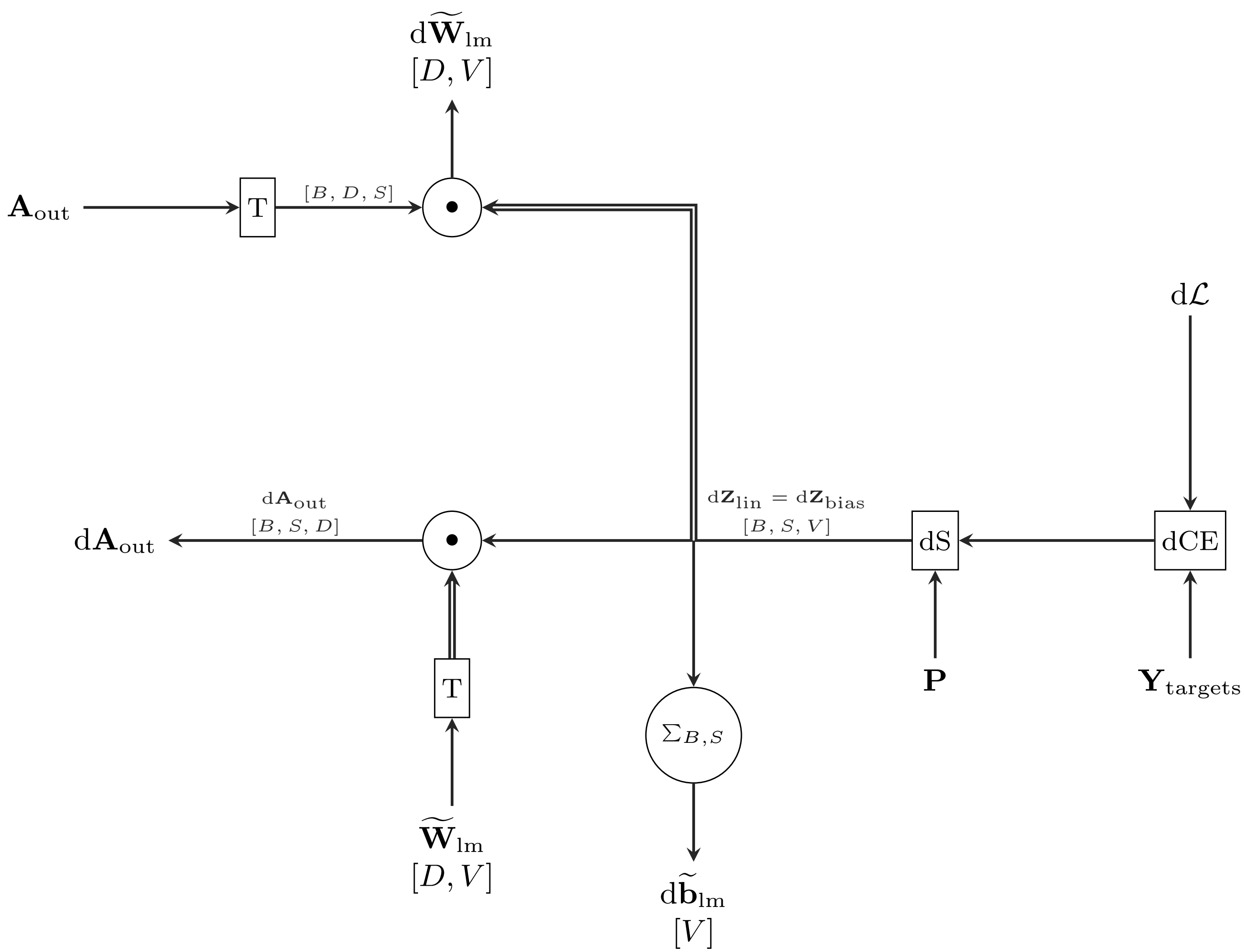
1.5.1 Forward Pass

Token Generation & Loss (Forward)



1.5.2 Backward Pass

Token Generation & Loss — Backward (Corrected)



1.5.3 Operations Summary

Operations (Ops)

Abbrev	**Name**	**Type / Shape**	**Notes**
S	Softmax	op	Over vocab axis V; outputs probabilities $\mathbf{P}$.
CE	Cross-Entropy	op	Usually *sparse* CE consuming label indices $\mathbf{Y}$.
ARG	Argmax (greedy)	op	argmax_V to get token ids (no gradient).
TOP-k	Top-k / sampling	op	Optional decoding path; no gradient.
T	Transpose	op	E.g., $\widetilde{\mathbf{W}}_{\text{lm}}^T \in \mathbb{R}^{V\times D}$.
$\text{BC}_{B,S}(\cdot)$	Broadcast	op	Expand $[V]\to[B,S,V]$ for bias add.
dS	Softmax backward	op	With CE: $\mathrm{d}\mathbf{Z}_{\text{bias}} = \mathbf{P} - \text{onehot}(\mathbf{Y})$.
dAddB	Addition (Bias) backward	op	Sends $\mathrm{d}\mathbf{Z}_{\text{bias}}$ to matmul and $\sum_{B,S}$.
$\sum_{B,S}$	Summation	op	Yields $\mathrm{d}\widetilde{\mathbf{b}}_{\text{lm}}$.

Data Tensors (Values)

Symbol	**Name**	**Shape**	**Notes**
$\mathbf{A}_{\text{out}}$	Transformer output (hidden)	$[B,S,D]$	Final hidden from the Transformer block(s).
$\widetilde{\mathbf{W}}_{\text{lm}}$	LM head weight (tied)	$[D,V]$	Typically tied to $\mathbf{E}^T$.
$\widetilde{\mathbf{b}}_{\text{lm}}$	LM head bias	$[V]$	Broadcast-added over $[B,S,V]$.
$\mathbf{Z}_{\text{lin}}$	Logits (linear output)	$[B,S,V]$	$\mathbf{A}_{\text{out}}\widetilde{\mathbf{W}}_{\text{lm}}$.
$\mathbf{Z}_{\text{bias}}$	Logits (final/Softmax input)	$[B,S,V]$	$\mathbf{Z}_{\text{lin}} + \widetilde{\mathbf{b}}_{\text{lm}}$.
$\mathbf{P}$	Probabilities	$[B,S,V]$	$\text{softmax}(\mathbf{Z}_{\text{bias}})$.
$\mathbf{Y}$	Target token ids	$[B,S]$	Ground-truth indices (sparse labels).
$\mathcal{L}$	Loss	scalar or $[B,S]$	Typically mean over B,S.
$\mathrm{d}\mathcal{L}$	Loss gradient	scalar-grad	Starting signal for backward pass.
$\mathrm{d}\mathbf{Z}_{\text{bias}}$	Final Logits gradient	$[B,S,V]$	From CE+Softmax: $\mathbf{P} - \text{onehot}(\mathbf{Y})$.
$\mathrm{d}\mathbf{Z}_{\text{lin}}$	Linear output grad	$[B,S,V]$	Same as $\mathrm{d}\mathbf{Z}_{\text{bias}}$.
$\mathrm{d}\widetilde{\mathbf{W}}_{\text{lm}}$	LM weight grad	$[D,V]$	$= \mathbf{A}_{\text{out}}^T\mathrm{d}\mathbf{Z}_{\text{lin}}$.
$\mathrm{d}\widetilde{\mathbf{b}}_{\text{lm}}$	LM bias grad	$[V]$	$= \sum_{B,S}\mathrm{d}\mathbf{Z}_{\text{bias}}$.
$\mathrm{d}\mathbf{A}_{\text{out}}$	Hidden grad	$[B,S,D]$	$= \mathrm{d}\mathbf{Z}_{\text{lin}}\,\widetilde{\mathbf{W}}_{\text{lm}}^T$.

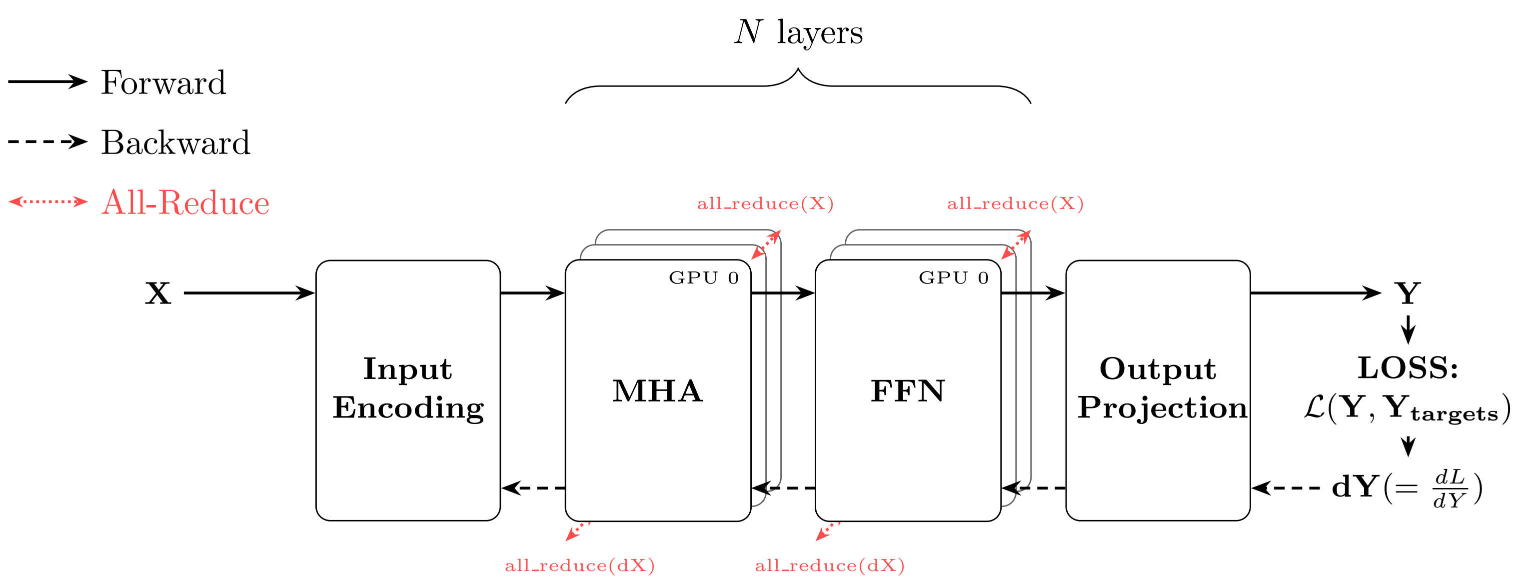
Shapes: B=batch, S=sequence length, D=hidden dim, V=vocab size.

2 Tensor Parallelism (TP)

[This chapter will cover Tensor Parallelism implementation details]

2.1 TP Overview

Transformer Overall Flow (TP with 3 GPUs)



Tensor Parallelism:
- Each GPU processes a shard of the weight matrix
- All-Reduce synchronizes partial results
- Forward: after row-parallel ops
- Backward: after column-parallel ops

[To be added]

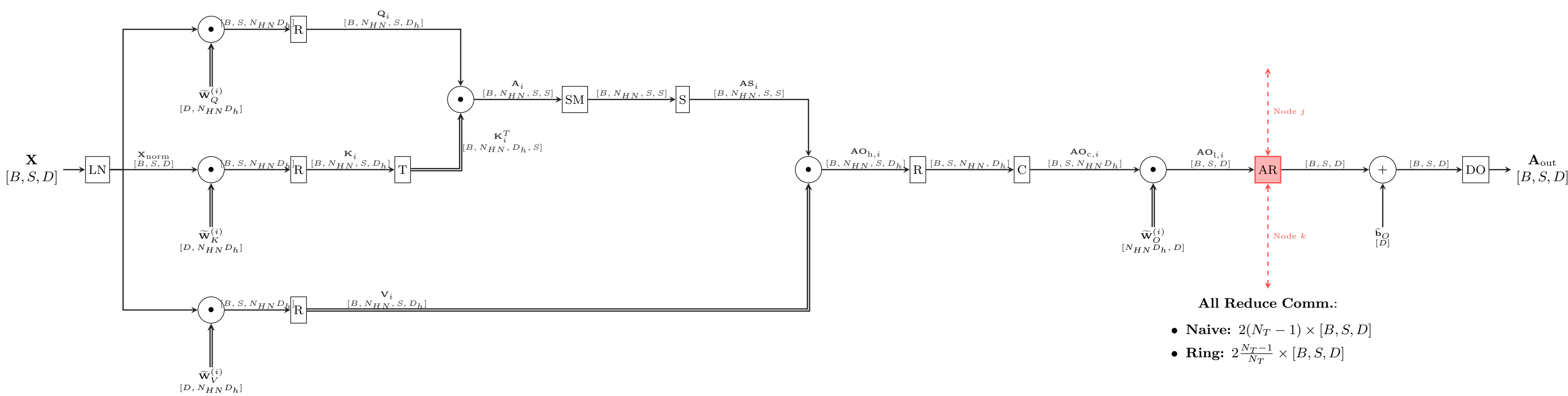
2.2 Forward Pass

In the tensor-parallel setting, the multi-head attention is distributed across N_T devices (GPUs). The notation used in the diagram is defined as follows:

- N_H: Total number of attention heads (global)
- N_T: Tensor parallelism degree (number of devices)
- N_{HN}: Number of heads per device (local), where $N_{HN} = \frac{N_H}{N_T}$

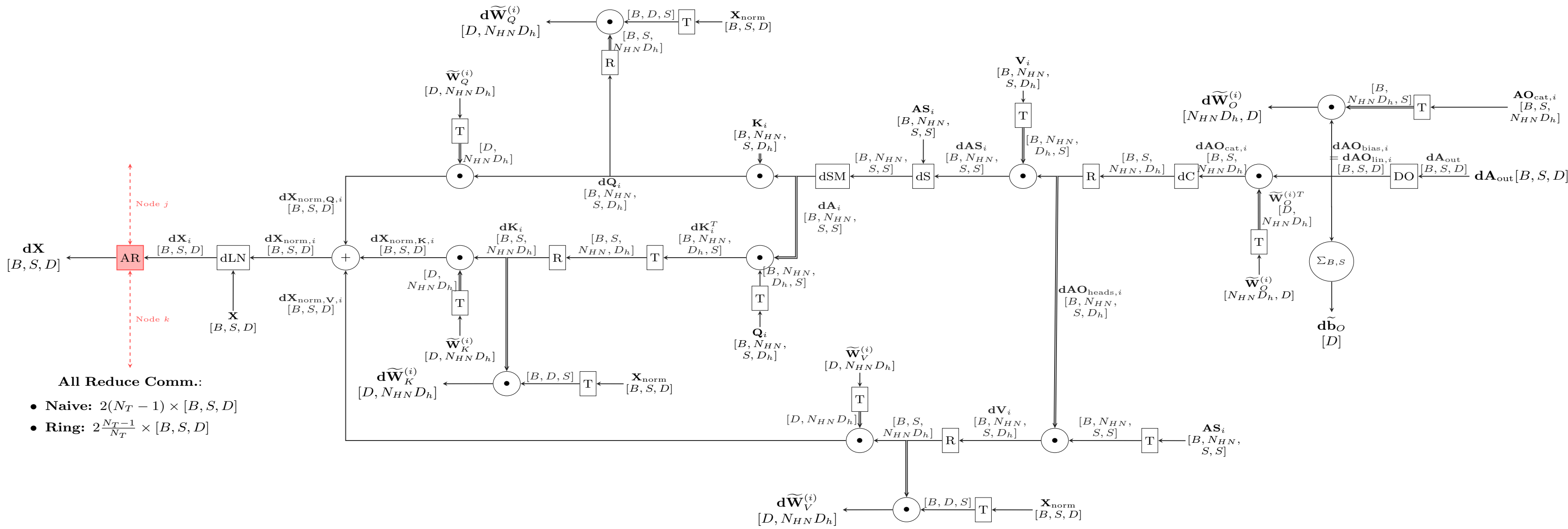
Each device processes N_{HN} heads independently. The weight matrices $\widetilde{\mathbf{W}}_Q$, $\widetilde{\mathbf{W}}_K$, $\widetilde{\mathbf{W}}_V$, and $\widetilde{\mathbf{W}}_O$ are column-partitioned across devices, with each device holding a $[D, N_{HN} \cdot D_h]$ slice of the original $[D, D]$ matrix.

Multi-Head Attention Forward Pass (Node i)

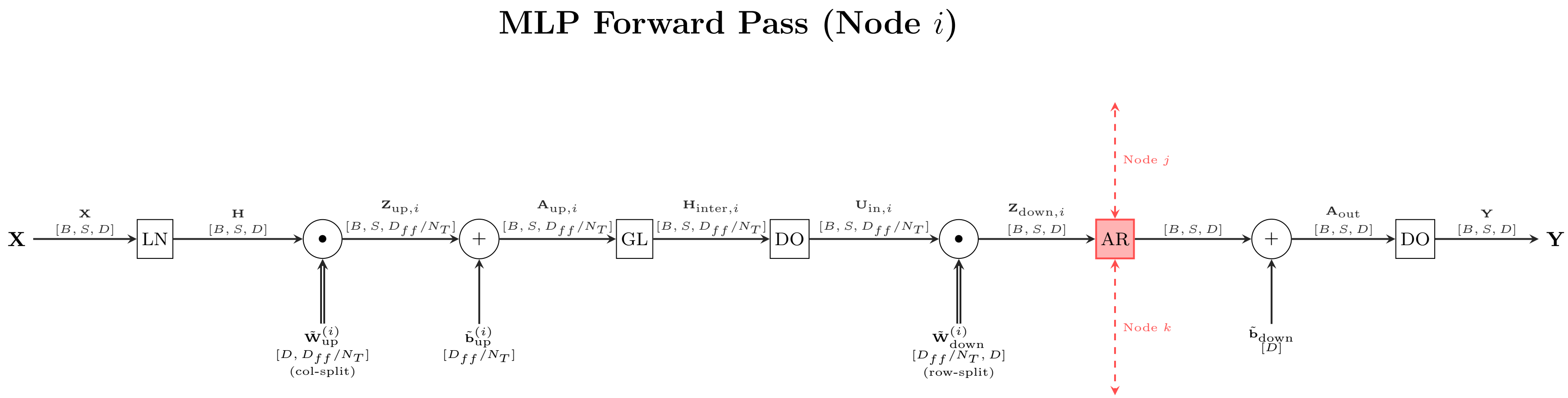


2.3 MHA Backward Pass

Multi-Head Attention Backward Pass (Node i)



2.4 MLP Forward Pass

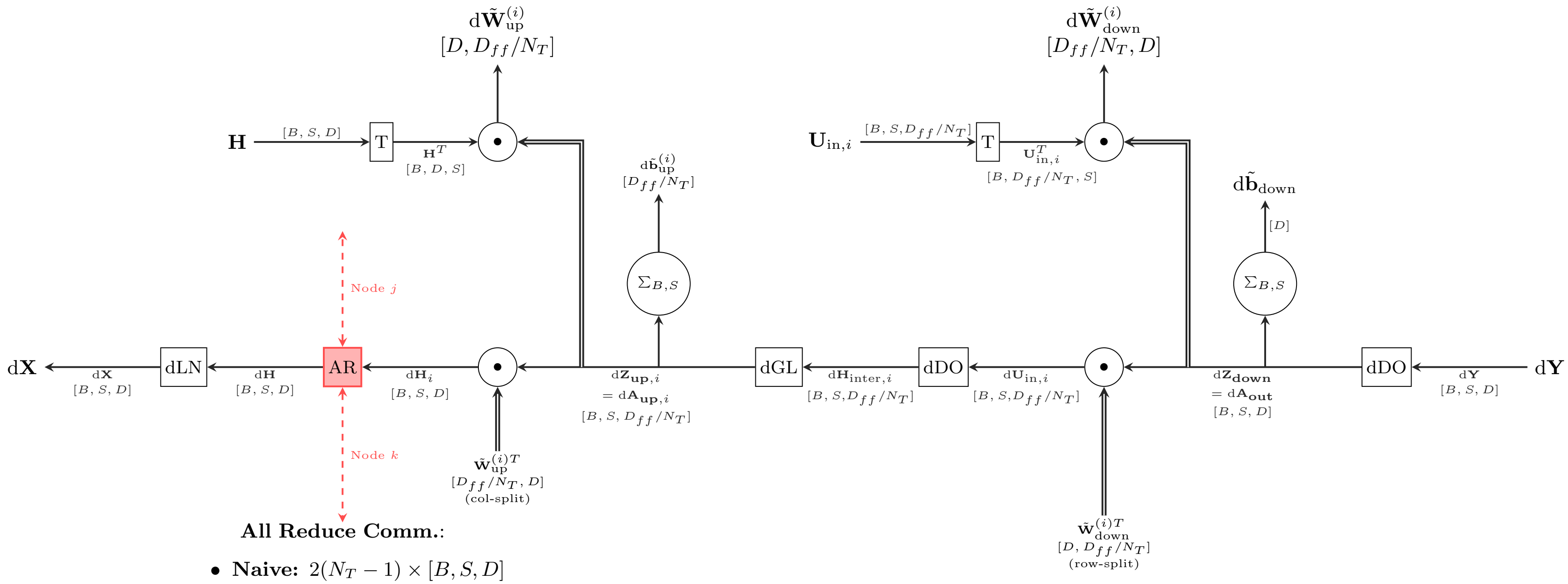


All Reduce Comm.:

- **Naive:** $2(N_T - 1) \times [B, S, D]$
- **Ring:** $2\frac{N_T-1}{N_T} \times [B, S, D]$

2.5 MLP Backward Pass

MLP Backward Pass (Node i)



All Reduce Comm.:

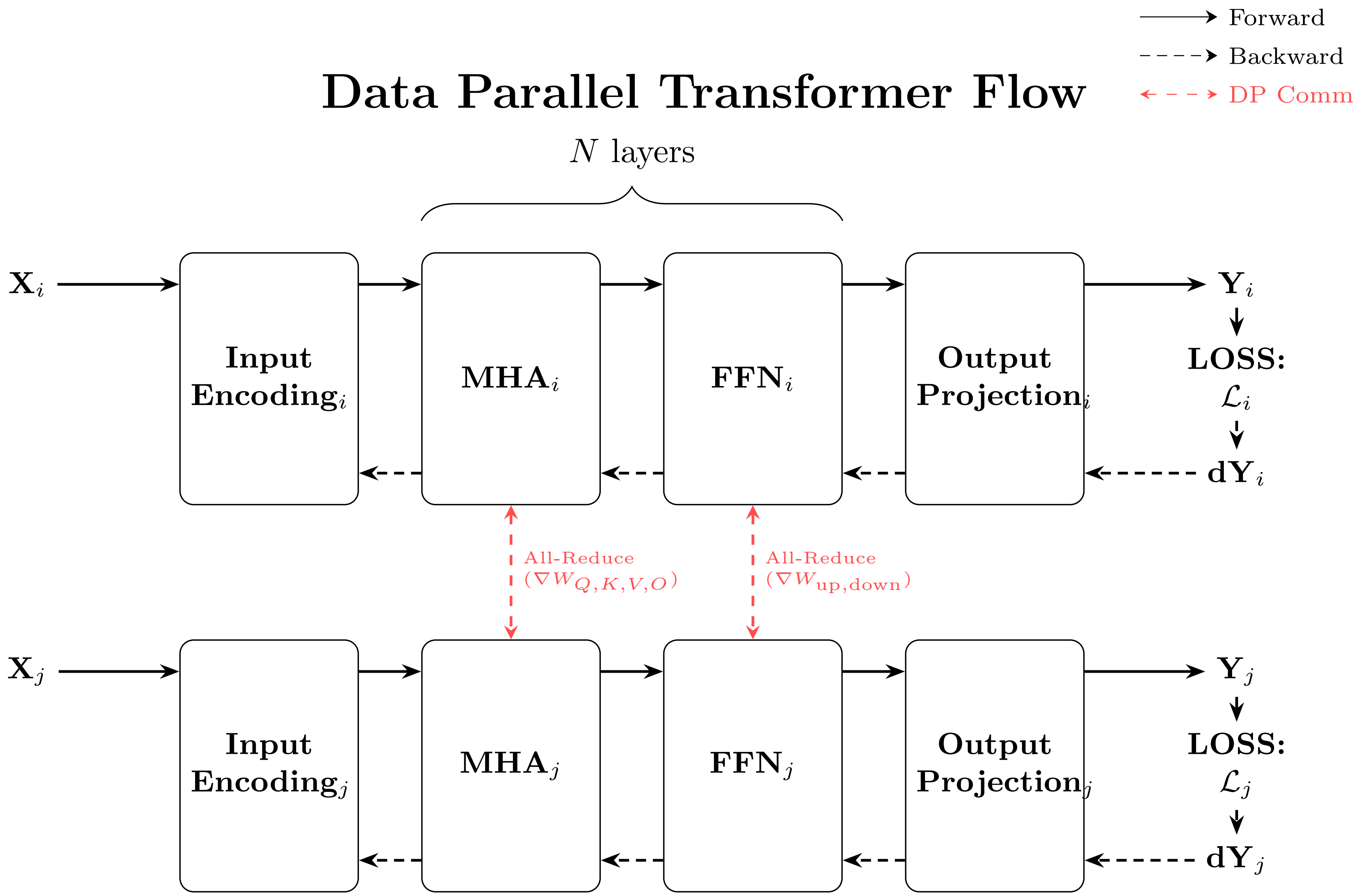
- **Naive:** $2(N_T - 1) \times [B, S, D]$
- **Ring:** $2\frac{N_T-1}{N_T} \times [B, S, D]$

2.6 Communication Patterns

[To be added]

3 Data Parallelism (DP)

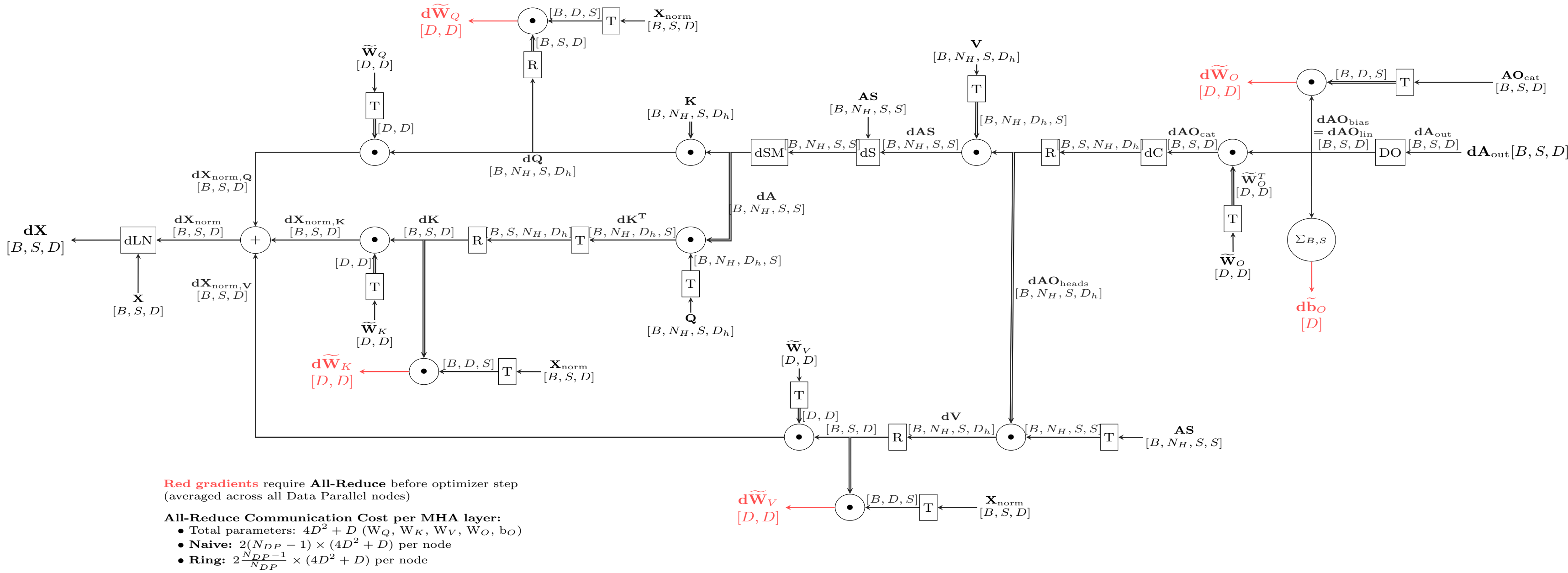
3.1 DP Overview



3.2 MHA Forward Pass

3.3 MHA Backward Pass

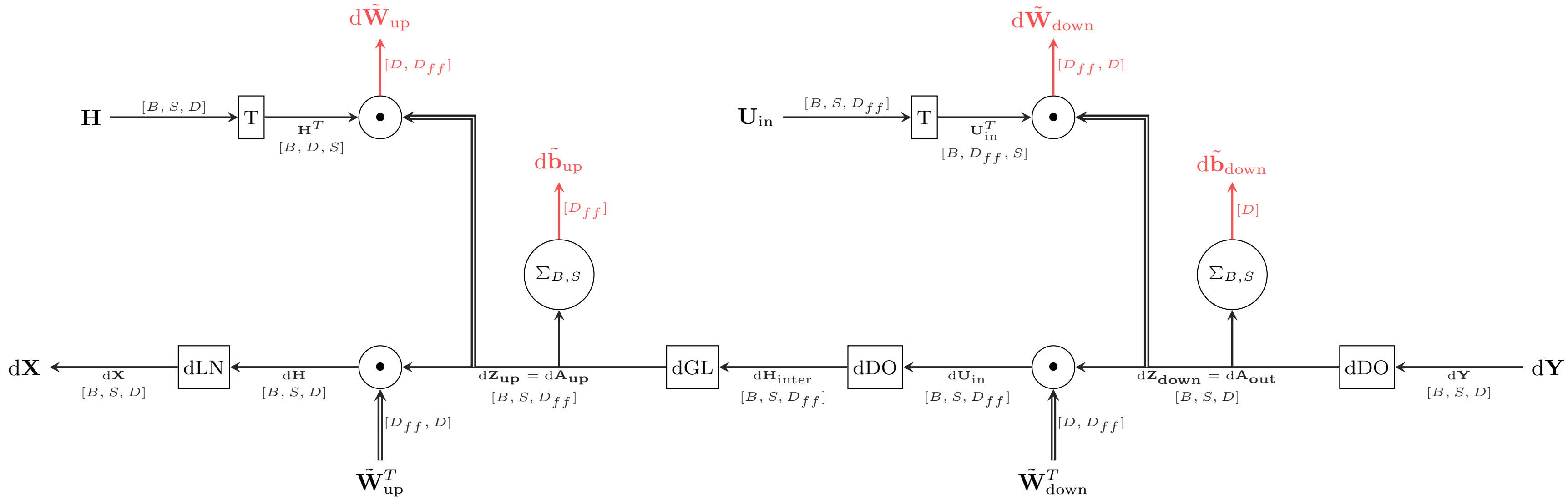
Multi-Head Attention Backward Pass (Data Parallel)



3.4 MLP Forward Pass

3.5 Mlp Backward Pass

MLP Backward Pass (Data Parallel)



Red gradients require **All-Reduce** before optimizer step

MLP All-Reduce Cost: $\sim 2DD_{ff}$ parameters (W_{up}, W_{down})

- **Naive:** $2(N_{DP} - 1) \times 2DD_{ff}$ per node
- **Ring:** $2\frac{N_{DP}-1}{N_{DP}} \times 2DD_{ff}$ per node

(Gradients averaged across N_{DP} data parallel nodes)