| PriViT G | | MPCViT | | PriVit R | | MPCViT+ | |
|---|---|---|---|---|---|---|---|
| Acc | Latency (s) | Acc | Latency (s) | Acc | Latency (s) | Acc | Latency (s) |
| 96.31 | 21.13 | 94.3 | 10.39 | 94.45 | 18.74 | 93.3 | 5.57 |
| 95.31 | 19.27 | 94.2 | 9.85 | 92.45 | 17.83 | 93.9 | 6.38 |
| 95.58 | 15.99 | 94.1 | 9.39 | 92.36 | 13.26 | 94.2 | 7.39 |
| 95.14 | 14.43 | 93.6 | 8.96 | 92.39 | 13.01 | 94.3 | 6.83 |
| 94.52 | 14.37 | | | 91.08 | 10.24 | | |
| 94.44 | 11.6 | | | | | | |

Table 6: Benchmarking PriViT and MPCViT over CIFAR 10 dataset on SEMI2k protocol.

## A Supplementary results

**Additional PI results** Following Zeng et al. (2022) we benchmark our method over SEMI2k using secretflow framework, the client and server are 64GB RAM, Intel(R) Xeon(R) Platinum 8268 CPU @ 2.90GHz. We run PI over LAN settings between two nodes of HPC cluster, hence there is a variation of the total inference latency from what is reported in Zeng et al. (2022), but to keep a consistent comparison, we benchmark both PriViT and MPCViT under our system settings. We report additional bench marking results on CIFAR 10 data in the table 6.

**Analysis of performance degradation.** In this analysis, we aim to compare the performance of trained PriViT models with their finetuned versions. Our analysis is based on the class-level accuracy metric from the Tiny ImageNet dataset, which consists of 200 classes. We focus on three specific parameters to understand the performance degradation:

*Maximum Difference in Accuracy:* We assess the greatest disparity in accuracy across all 200 classes between the PriViT and finetuned models.

*Overall Accuracy Difference:* We compute the average accuracy difference between the finetuned and the PriViT models across all 200 classes.

*Variance in Accuracy Difference:* We analyze the consistency of the differences in accuracy across the 200 classes by calculating the variance.

Table 7 highlights that average accuracy degradation is anywhere between 1-13% for different non-linearity budgets but certain classes seem to be more adversely affected even in low budgets as the max class level difference in accuracy is consistent around 30%.

Table 7: Performance degradation of PriViT models compared to finetuned model on tinyimagenet.

| Accuracy | Latency (M) | Max Difference | Mean Difference | Variance ($10 \times 10^{-3}$) |
|---|---|---|---|---|
| 69.8 | 151.75 | 30.00% | 1.85% | 6.8 |
| 66.98 | 128.23 | 34.00% | 4.68% | 6.9 |
| 64.46 | 110.60 | 34.00% | 7.21% | 8 |
| 60.53 | 93.72 | 40.00% | 11.13% | 9 |
| 59.55 | 86.51 | 34.00% | 12.12% | 9.4 |
| 59.58 | 84.78 | 36.00% | 12.08% | 9.8 |
| 59.04 | 84.13 | 36.00% | 12.63% | 9.9 |
| 58.74 | 69.42 | 40.00% | 12.92% | 9.4 |
| 58.2 | 67.43 | 36.00% | 13.48% | 8.7 |

**Fine-grained versus layer-wise Taylorization** PriVit employs a unique approach where it selectively Taylorizes softmax and GELU operations in models. To probe the effectiveness of this method, we contrasted it with an alternative PriViT approach that Taylorizes a ViT model progressively, layer by layer.

As illustrated in Table 8, our observations underscored the superiority of selective Taylorization.

Table 8: Performance comparison of PriViT versus layerwise linearization of GeLU in a ViT model with 200k GeLUs. Twelve models were generated by sequentially replacing up to 12 GeLU layers with Identity. PriViT was also evaluated with varying GeLU budgets below 200k.

| Layerwise GELU linearizing | | Pri-ViT | |
|---|---|---|---|
| Gelus (K) | Accuracy | Gelus (K) | Accuracy |
| 197 | 96.07 | 200 | 95.59 |
| 196 | 96.07 | 150 | 95.34 |
| 193 | 95.91 | 100 | 95.58 |
| 190 | 95.35 | 50 | 94.98 |
| 187 | 94.28 | 10 | 94.24 |
| 184 | 93.75 | 1 | 93.96 |
| 181 | 93.33 | | |
| 178 | 92.99 | | |
| 174 | 93.04 | | |
| 171 | 92.88 | | |
| 164 | 92.06 | | |
| 123 | 82.48 | | |
| 0 | 56.64 | | |

This superiority was especially pronounced under constrained non-linearity budgets.

Delving deeper, our experiment commenced with a foundational ViT model populated with 200k GeLUs, while the remaining operations were Identity-based. From this foundation, we crafted a series of models, each with an increasing number of GeLU layers swapped for Identity, creating a spectrum from 1 to 12 GeLU replacements. Post-finetuning, the performance metrics of these models were recorded. In parallel, we evaluated PriViT under a gamut of GeLU budgets, all set below the 200k threshold, thereby exploring its capability for dynamic GeLU retention.

**Hyperparameter Tuning** Following (Hassani et al., 2021) we use CutMix (Yun et al., 2019), Mixup (Zhang et al., 2017), Randaugment (Cubuk et al., 2020), and Random Erasing (Zhong et al., 2020) as data augmentation strategy. We probed multiple hyperparameter strategies for the joint optimization phase of PriViT to ensure consistent good performance over multiple configurations of non-linearity budgets of softmax and GELUs. Specifically we describe these strategies as follows:

**Late-Binarized Epoch (Strategy 1)**: This strategy involved 10 post-linearization training epochs. The binarization of auxiliary parameters, $s$ and $c$, occurred late in the process, specifically after the linearization was complete. The penalty increment condition for this method was checked when the reduction in the softmax and GELU coefficients per epoch was less than 200 and 2, respectively. Both masks began with identical penalties, signifying an ’equal’ starting penalty.

**Late-Binarized Incremental (Strategy 2)**: This strategy also encompassed 10 training epochs with late binarization. Here, the penalty increment condition was activated with an increase in the softmax and GELU coefficients per epoch. The starting penalty for both masks was ’equal’.

**Late-Binarized Divergent Penalty (Strategy 3)**: Much like Strategy 2, this involved 10 epochs with late binarization and an increment condition based on softmax and GELU coefficient rises. However, the initial penalty was set to ’unequal’, making the softmax penalty 20 times higher than the GELU penalty.

**Early-Binarized Incremental (Strategy 4)**: This strategy shared several similarities with Strategy 2, including 10 training epochs and an increment condition based on coefficient increases. The difference, however, lay in its early binarization, occurring during the freezing of the auxiliary parameters. The starting penalty was kept ’equal’ for both masks.

**Prolonged Early-Binarized Epoch (Strategy 5)**: Spanning 50 post-linearization training epochs, this strategy adopted an early binarization approach. The penalty increment condition was activated when the reduction in softmax and GELU coefficients per epoch was under 200 and 2, respectively. The masks were initialized with ’equal’ penalties.

Each of these strategies offered unique configurations in terms of epoch durations, binarization timings, increment conditions, and starting penalties, enabling a comprehensive assessment of the PriViT algorithm's performance under various conditions.

We test the different finetuning strategies described here by taylorizing PriViT for different softmax and GELU budgets and compare the test accuracy of the resulting model over CIFAR100. Table 9 highlights the comparative performance of all the strategies that we described. Strategy 5 seems to be performing best over different configuration of nonlinearity budget which is important as we would want to find the best model peformance for a particular non-linearity budget.

Table 9: We test the different finetuning strategies described in A. We run PriViT for different softmax and GELU budgets and compare the test accuracy of the resulting model over CIFAR100. We observe that strategy 5 works the best across a wide range of target softmax and GELU budgets.

| # Softmax (K) | # Gelu (K) | Strategy 1 (Acc. %) | Strategy 2 (Acc. %) | Strategy 3 (Acc. %) | Strategy 4 (Acc. %) | Strategy 5 (Acc. %) |
|---|---|---|---|---|---|---|
| 10 | 5 | 77.68 | 76.74 | - | 77.82 | **78.83** |
| 5 | 5 | 76.27 | 75.99 | 75.72 | - | **77.63** |
| 5 | 1 | 76.73 | 75.21 | 76.24 | - | **77.08** |
| 2 | 10 | 76.04 | 75.23 | - | 74.65 | **76.35** |
| 2 | 1 | 75.92 | 74.84 | 76.45 | - | **76.97** |
| 1 | 5 | 76.12 | 74.99 | 76.32 | - | **76.96** |

**Grid search of softmax and GELU configuration.** In order to elucidate the nuanced trade-off between softmax and GeLU operations, we executed a systematic grid search across an extensive parameter space encompassing varied softmax and GeLU configurations. Upon analysis of models exhibiting iso-latencies, as demarcated by the red lines in figure 7, it became evident that the trade-off dynamics are non-trivial. Specifically, configurations with augmented softmax values occasionally demonstrated enhanced performance metrics, whereas in other scenarios, models optimized with increased GeLU counts exhibited superior benchmark results.

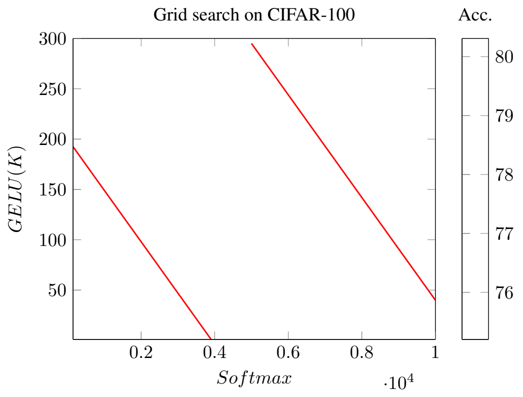


Figure 7: The PriViT algorithm produces a Pareto surface mapping the tradeoff between GeLU and softmax budgets over cifar 100.

**Taylorizing only one type of non-linearity.** The PriViT algorithm's standout capability is its simultaneous linearization of GELU and softmax operations, enabling a myriad of model configurations. In our focused experiment, we exclusively linearized GELU operations and anchored the auxiliary softmax parameter $S$, binarizing it to activate only the SoftmaxAttention mechanism. Despite extensive GELU substitutions, as reported in 8 the PriViT model displayed notable resilience on CIFAR10 and CIFAR100 datasets, with only slight performance drops, underscoring its robustness in varied setups.

**Effect of using pre-trained checkpoints.** To further investigate why using pretrained checkpoint is improving performance, we report the non-linear distributions searched by PriViT and compare it with PriViT without pretrain for the nonlinearity budget of 315k and 320k respectively. We observe from our findings in figures 9,10 that the distribution found by the two methods differs across each layer.

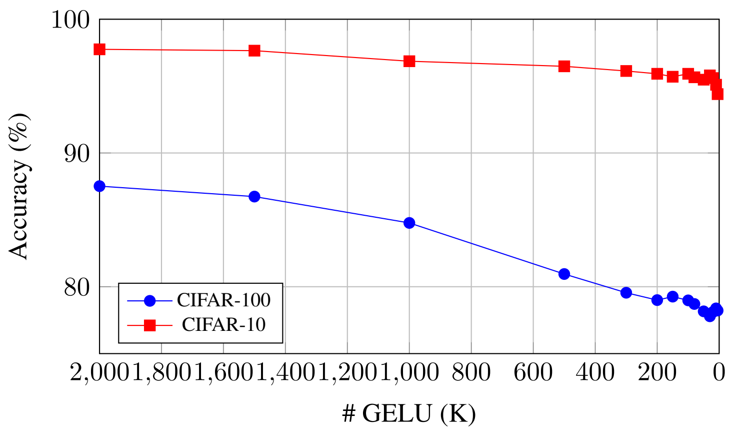


Figure 8: PriViT's ability to linearize GeLU operations visualized through performance on CIFAR datasets. As GELU operations decrease, CIFAR-100 and CIFAR-10 accuracies are affected, showcasing the trade-off between operation count and accuracy.

This supports our theory as to how PriViT operates under a strategic 'top-down' paradigm. Starting with a fine-tuned model, it has the advantage of an architecture that has not just discerned overarching generalization patterns but has also selectively pruned irrelevant information, streamlining its focus for a specific downstream task. This reduction of redundancy, undertaken from a vantage point of a pre-existing knowledge base, gives PriViT an edge.

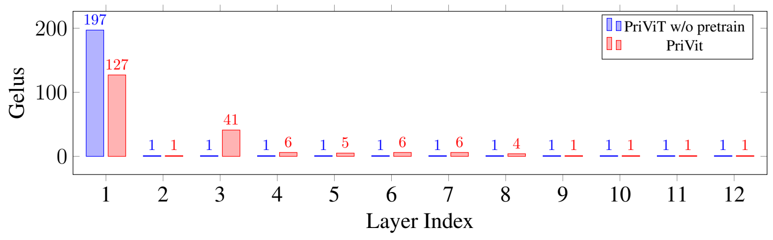


Figure 9: We compare the distribution of 208 GELU and 200 GELU operations distributed by PriViT w/o pretrain and PriViT respectively over tiny imagenet dataset.

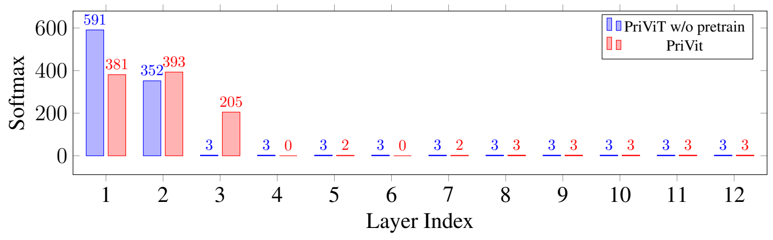
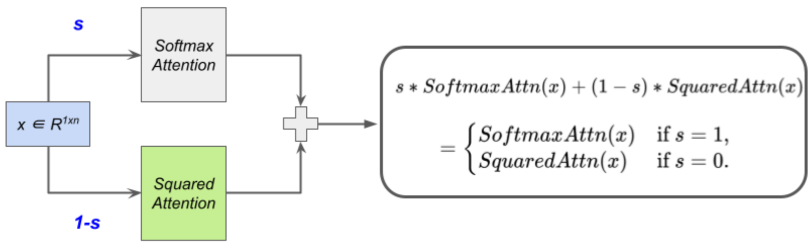


Figure 10: We compare the distribution of 973 softmax operations and 998 softmax operations operations distributed by PriViT w/o pretrain and PriViT respectively over tiny imagenet dataset.

## B SUPPLEMENTARY GRAPHICS

The following figure shows a graphical representation of the switching operation.

**Search granularity.** An important characteristic of PriViT is it's flexibility to search over different granularity of non-linearities. GELU is a pointwise functions, thus PriViT can search either at embedding level or at a token level. On the other hand, softmax is a token level operation, thus it


s
Softmax Attention
$x \in R^{1xn}$
Squared Attention
1-s
$s * SoftmaxAttn(x) + (1-s) * SquaredAttn(x)$
$= \begin{cases} SoftmaxAttn(x) & \text{if } s = 1, \\ SquaredAttn(x) & \text{if } s = 0. \end{cases}$

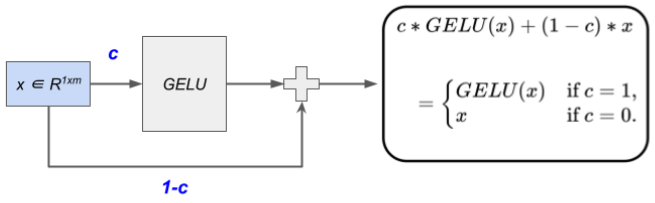

c
$x \in R^{1xm}$
GELU
1-c
$c * GELU(x) + (1-c) * x$
$= \begin{cases} GELU(x) & \text{if } c = 1, \\ x & \text{if } c = 0. \end{cases}$

Figure 11: Parameterized Gelu and Self-Attention operations. **Top**: Tokens undergo softmax and squared attention in training. Post-training, parameter $S$ is frozen and binarized, selecting only one operation. **Bottom**: Embeddings pass through GeLU and Identity during training. Afterwards, parameter $C$ is frozen and binarized, choosing a single operation.

cannot be broken into a finer search space. Note that softmax operations can be extended to search over the head space or layer space, and similarly GELU can be searched over the layer space. Fig 12 illustrates the search granularity over token and embedding space.

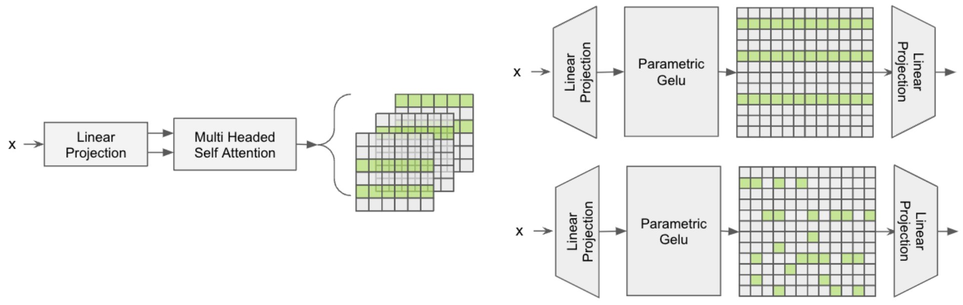


Figure 12: **Left**: The green blocks are SQUAREDATTENTION, and the grey blocks are Softmax Attention. For parametric attention, tokens emerge from a blend of softmax and square attention (refer to fig 11). Post-training, auxiliary variable $S$ is set to 0 or 1, resulting in $2^{N \times H}$ potential combinations per encoder block. **Right**: The green blocks are Identity function, and the grey blocks are GELU activation. Embeddings combine GELU and identity operations during training, as seen in fig 11. After training, parameter $C$ is frozen and binarized. This yields potential combinations of either $2^{H \times N}$ or $2^{N \times H \times m}$ for each ViT encoder block. Note that GELU being a pointwise function, we possess the flexibility to expand our search space either to tokens or directly to individual embeddings.

## C PRIVIT ALGORITHM

We provide detailed pseudocode for PriViT here.

**Algorithm 1** PRIVIT: Privacy Friendly ViT

```
1: Inputs: $f_{\mathbf{W}}$: pre-trained network, $\lambda_s$: Lasso coefficient for Softmax mask, $\lambda_g$: Lasso coefficient for GeLU
   mask, $\kappa$: scheduling factor, $G$: GeLU budget, $S$: Softmax budget, $\epsilon$: threshold.
2: Set $\mathbf{C} = 1$: same dimensions to all GeLU mask.
3: Set $\mathbf{S} = 1$: same dimensions to all Attention Heads.
4: Set $C_{\text{budget}} = False$: GeLU budget flag.
5: Set $S_{\text{budget}} = False$: Softmax budget flag.
6: $\overline{\mathbf{W}} \leftarrow (\mathbf{W}, \mathbf{C}, \mathbf{S})$
7: Lowest GeLU Count $\leftarrow \|\mathbb{1}(\mathbf{C} > \epsilon)\|_0$
8: Lowest Softmax Count $\leftarrow \|\mathbb{1}(\mathbf{S} > \epsilon)\|_0$
9: while GeLU Count $> G$ or Softmax Count $> S$ do
10:    Update $\overline{\mathbf{W}}$ via ADAM for one epoch.
11:    GeLU Count $\leftarrow \|\mathbb{1}(\mathbf{C} > \epsilon)\|_0$
12:    Softmax Count $\leftarrow \|\mathbb{1}(\mathbf{S} > \epsilon)\|_0$
13:    if Lowest GeLU Count - GeLU Count $< 2$ then
14:        $\lambda_g \leftarrow \kappa \cdot \lambda_g$.
15:    end if
16:    if Lowest Softmax Count - Softmax Count $< 200$ and $S_{\text{budget}} = False$ then
17:        $\lambda_s \leftarrow \kappa \cdot \lambda_s$.
18:    end if
19:    if Lowest GeLU Count $>$ GeLU Count then
20:        Lowest GeLU Count $\leftarrow$ GeLU Count
21:    end if
22:    if Lowest Softmax Count $>$ Softmax Count then
23:        Lowest Softmax Count $\leftarrow$ Softmax Count
24:    end if
25:    if GeLU count $<=$ G and $C_{\text{budget}} = False$ then
26:        $\mathbf{C} \leftarrow \mathbb{1}(\mathbf{C} > \epsilon)$
27:        $C_{\text{budget}} = True$
28:        $\overline{\mathbf{W}} \leftarrow (\mathbf{W}, \mathbf{S})$
29:    end if
30:    if Softmax count $<=$ S and $S_{\text{budget}} = False$ then
31:        $\mathbf{S} \leftarrow \mathbb{1}(\mathbf{S} > \epsilon)$
32:        $S_{\text{budget}} = True$
33:        $\overline{\mathbf{W}} \leftarrow (\mathbf{W}, \mathbf{C})$
34:    end if
35: end while
```

| PriViT | | MPCViT TinyImagenet | | MPCViT CIFAR10/100 | |
|---|---|---|---|---|---|
| Function | # ReluOps | Function | # ReluOps | Function | # ReluOps |
| Softmax(197) | 18586 | ReLU Softmax(257) | 4428 | ReLU Softmax(65) | 1133 |
| Layernorm(192) | 6504 | Layernorm(192) | 6504 | Layernorm(256) | 8614 |
| GeLU(1) | 270 | GeLU(1) | 270 | GeLU(1) | 270 |
| $x^2$(197) | 3248 | | | | |

Table 10: Non-linearity cost normalized to the cost of one ReluOp which is 1 ReLU operation over a scalar value. Bracket considers amortizing to a vector of inputs, e.g., a Layernorm(192) is an operation over a vector length of 192 is equivalent to 6504× than the cost of a ReLU.

## D LATENCY BENCHMARKS

We conduct thorough benchmarking by creating GC circuits for the non-linearity functions found in ViT, and also benchmark specific functions used in MPCViT so as to enable us to compare the two methods under the same protocol DELPHI. In order to compare the different cost of non-linearity we bring them down to a common benchmark of ReluOps, where 1 ReluOp is the cost incurred for performing a GC evaluation of ReLU of one scalar value. Figure 14 shows how we count the non-linearity cost of softmax. The front and end consider Secret Sharing similar to Circa Ghodsi et al. (2021). Since the GC cost of each operation is known, we add them up as the final cost of softmax.

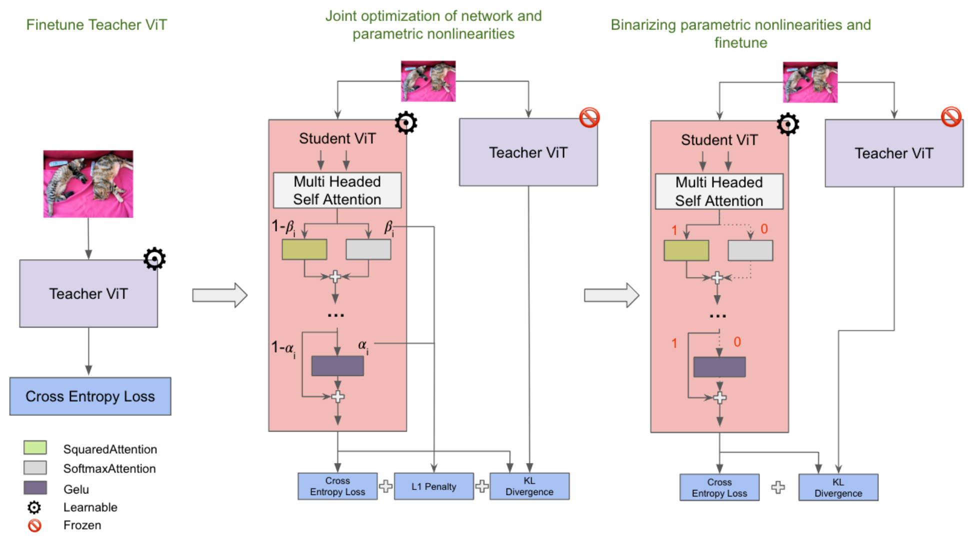


Figure 13: **Left**: Step 1 - Fine-tuning of a pretrained ViT over target dataset to produce the 'teacher ViT'. **Middle**: Step 2 - Duplicate teacher ViT, introduce parametric GELUs and attention mask to form 'student ViT'. Train using cross-entropy loss, KL divergence, and L1 penalty to gradually find a sparse mask. Binarize the mask post desired non-linearity budget. **Right**: Step 3 - With a frozen, binarized mask, further fine-tune the student model using cross-entropy loss and KL divergence with the teacher.

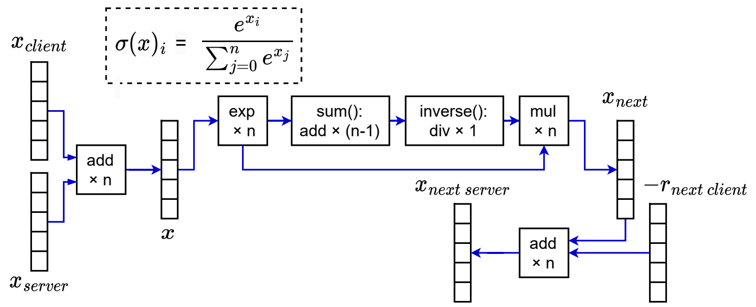


Figure 14: *Detailed steps of benchmarking the non-linearity cost for softmax. Denominator is calculated once and reused for all indices of the vector.*

## E Attention Variants

Here we describe formally the different attention variant we ablated. Uniform form attention is basically described by the following equation

$$\textsc{UniformAttn}(\mathbf{X}) = \frac{(1)}{N}\mathbf{W}_v\mathbf{X}, \tag{7}$$

Where N is the number of tokens, so for each token the attention is equal hence the name UniformAttention.

ScaleAttn is the softmax candidate used in the work Zeng et al. (2022) which is essentially described as

$$\textsc{ScaleAttn}(\mathbf{X}) = \frac{(\mathbf{X}\mathbf{W}_q\mathbf{W}_k\mathbf{X})}{N}\mathbf{W}_v\mathbf{X}, \tag{8}$$