BIVIT: EXPLORING BINARY VISION TRANSFORMERS

Anonymous authors
Paper under double-blind review

ABSTRACT

We introduce BiViT, a Binary Vision Transformer that tackles the extremely difficult problem of quantizing both the weights and activations of a ViT model to just 1 bit. Initially, we observe that the techniques used to binarize transformers in NLP don't work on Vision Transformers (ViTs). To address this, we introduce some simple yet critical architectural changes, improving 28% over a baseline binarized ViT. Then, we improve 11% over from-scratch training by employing our normalized BiViT distillation scheme, which we find to be crucial for dense distillation in vision. Overall, BiViT can achieve a 58x reduction in operations and a 20x compression in model size, while bringing top-1 accuracy on ImageNet-1k in line with similar benchmarks for binary transformers in NLP. We hope BiViT can be the first step toward even more powerful binary ViT models.

1 INTRODUCTION

Transformers have emerged as the dominant architecture across deep learning, originally in NLP (Vaswani et al., 2017) as popularized by BERT (Kenton & Toutanova, 2019), and now also in vision with ViT (Dosovitskiy et al., 2020) and more vision-specific transformers such as SWIN (Liu et al., 2021a). While effective these transformers are huge, reaching billions of parameters in both NLP (Radford et al., 2019) and vision (Zhai et al., 2021). Because of this, these models can be difficult to run on consumer hardware.

One way to address this issue is to use quantization, reducing the size of each parameter and the time taken per operation. In the extreme, we can use binary quantization (Courbariaux et al., 2016), where each parameter and activation of the model is reduced to a single bit. However, the literature has not fully caught up to the current progress in vision architectures. There are a few works about binarizing NLP transformers such as BERT (Bai et al., 2020; Qin et al., 2022), but none about fully binarizing Vision Transformers.

Perhaps this is simply because creating a 1-bit ViT is difficult. First, the existing binary literature in computer vision (e.g., Liu et al. (2018; 2020)) relies heavily on the architectural design choices of convnets such as ResNet (He et al., 2016) or MobileNet (Howard et al., 2017). For instance, the ability to add extra skip connections (Liu et al., 2018) and the type and position of normalization layers in the network (see Sec. 3.1) have a large impact on the performance of binary models. The architectural design choices made in ViT (Dosovitskiy et al., 2020; Liu et al., 2022b) are very different to the convnets of the last decade, and thus many architectural techniques in the binary quantization literature don't apply.

To remedy this, we make some critical changes in the ViT architecture to enable 1-bit weights and activations: we return to BatchNorm (Ioffe & Szegedy, 2015) between every layer, as it gives a more balanced post-sign distribution than LayerNorm (Ba et al., 2016) (Sec. 3.1); we add generalized nonlinearities to better model ViT's intermediate features (Sec. 3.2); and we introduce a new mixed precision attention that works well with binary activations (Sec. 3.3). Overall, these changes increase performance on ImageNet-1k (Deng et al., 2009) by 28% compared to naively applying off-the-shelf techniques from vision (Xu et al., 2021) to the original ViT architecture.

Next, another central roadblock is that training even full precision ViTs is a challenge. Doing so requires either large pretraining datasets (Dosovitskiy et al., 2020; Steiner et al., 2021), a rich pretraining task (He et al., 2021; Xie et al., 2021), new optimization techniques (Foret et al., 2020; Chen et al., 2021), or distillation from existing convnets (Touvron et al., 2020; Ridnik et al., 2022). And on top of that, binary training is significantly more difficult and takes longer than floating point.

The training time on ImageNet-1k for binary training methods like ReCU (200 epochs, Xu et al. (2021)) or ReActNet (240 epochs, Liu et al. (2020)) is over twice that of a full precision ResNet (88 to 120 epochs, Leclerc et al. (2022); He et al. (2016)). What's worse is that ViTs typically train for 300 epochs or more (Dosovitskiy et al., 2020; Touvron et al., 2020)! So how long would we need to train a binary ViT?

Luckily, NLP has a solution for fast transformer training: dense distillation. Instead of learning only the outputs of a teacher model (Hinton et al., 2015), it's common in NLP to distill intermediate layers of a transformer (Jiao et al., 2019). However, while this technique is used by binary NLP transformers (Qin et al., 2022), we find that 1-bit ViTs perform abysmally with dense distillation (see Tab. 3). We identify the cause of this issue to be the placement of LayerNorm in the original ViT architecture (see Sec. 3.2) and fix it using a normalized distillation loss (Sec. 4.1). Then, we improve the results of distillation using two-stage training, where we first perform dense distillation with only 1-bit activations then fine-tune using output distillation with both 1-bit weights and activations (Sec. 4.2). All together, using BiViT distillation offers an 11% improvement over training from scratch and a 6% improvement over existing two-stage distillation methods (Liu et al., 2020).

In this paper, we identify that existing techniques for binarizing transformers from the NLP literature don't perform well for 1-bit vision transformers. To address this, we make the following contributions. First, we introduce a BiViT architecture (Sec. 3) using simple components to fix normalization issues (Sec. 3.1, 3.2) and a novel mixed-precision attention block (Sec. 3.3) to address difficulty in quantization. Then, we follow with a BiViT distillation scheme (Sec. 4) that unlocks the use of dense distillation on ViTs using our normalized distillation loss (Sec. 4.1) and then use that to create a two-stage training method that outperforms existing distillation techniques (Sec. 4.2). Finally, we fine-tune on CIFAR to show that these results can be transferred to other datasets without additional distillation. Altogether, this paper introduces the first binary vision transformer model and training approach which achieves cost and accuracy on par with that of binary NLP transformers.

2 Related Work

In this paper, we set out to create the first binary ViT (Dosovitskiy et al., 2020), where each weight and activation are represented by a single bit (1w1a).

Binary Quantization. Binary or 1-bit quantization is the most extreme version of quantization, where all weights and activations are either +1 or -1. This both compresses the size of the model up to 32x and allows for fast evaluation using `xor` and `popcnt` instructions. Binary quantization began with BNN (Courbariaux et al., 2016) and has since seen rapid development with more effective quantization schemes (Rastegari et al., 2016; Bulat & Tzimiropoulos, 2019), better architectures (Liu et al., 2018; 2020), improved 1-bit training methods (Lin et al., 2020; Xu et al., 2021), and two-stage training (Martinez et al., 2020; Liu et al., 2020). However, most of this work in vision has been performed on convnets. It's not clear how much, if any, of these improvements would also work for ViTs.

Binary NLP Transformers. Recently, BiBERT (Qin et al., 2022) has become the first to address 1w1a quantization for NLP transformers. In BiBERT, the authors employ a specialized Direction Matching Distillation (DMD) scheme to align the activations of a 1-bit BERT with its full precision teacher, but we find this to not perform well enough on ViTs (see Tab. 2). Similarly, the single-stage dense distillation used by BiBERT dosn't work on ViTs (see Tab. 3). We address these shortcomings in Sec. 3.3 and Sec. 4.1.

Quantized Vision Transformers. Vision Transformers (Dosovitskiy et al., 2020; Touvron et al., 2020; Liu et al., 2021a) are a relatively new innovation in vision. Thus, so too are the works that attempt to quantize them. There are several methods that employ quantization of both weights and activations, either with 8w8a (Liu et al., 2021b; Lin et al., 2021), 2w8a (Xu et al., 2022), or 3w3a (Li et al., 2022). We are the first to attempt 1w1a ViT quantization.

3 BiViT Architecture

In this work, we set out to create a Binary Vision Transformer by densely distilling (Jiao et al., 2019) from an existing ViT model (Dosovitskiy et al., 2020). To show that this is a non-trivial

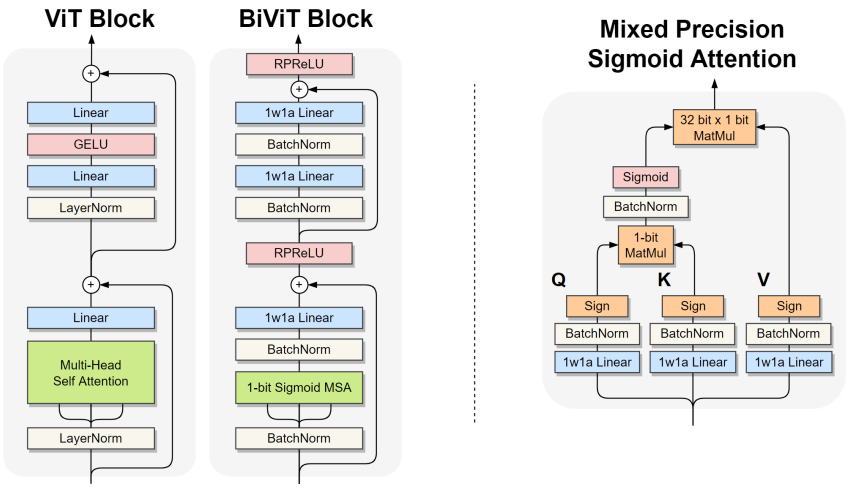


Figure 1: **Proposed BiViT Architecture.** The BiViT block compared to the original ViT architecture (left) and the proposed mixed precision sigmoid attention (right). BatchNorms are a cheap way to balance binary features across the dataset (Sec. 3.1), RPReLUs assist in modeling the ViT's intermediate features (Sec. 3.2), and mixed precision sigmoid attention offers the best cost performance trade-off of all attention methods we test (Sec. 3.3).

exercise, we apply our best distillation method from Sec. 4 to a baseline 1-bit ViT-S/32 architecture on ImageNet-1k using standard techniques from Rastegari et al. (2016); Bulat & Tzimiropoulos (2019); Xu et al. (2021). This baseline results in a staggering drop in accuracy from the teacher: 76% all the way down to 24% (Tab. 1). In this section, we investigate why this happens and provide several architectural solutions to increase accuracy.

3.1 Revisiting Normalization

Borrowing from NLP (Vaswani et al., 2017), modern vision transformer architectures use LayerNorm (Ba et al., 2016) for feature normalization instead of BatchNorm (Ioffe & Szegedy, 2015). Moreover, vision transformer architectures such as ViT place the LayerNorm *before* the corresponding linear or attention layer, while convnets such as ResNet (He et al., 2016) place their normalization *after* the corresponding layer.

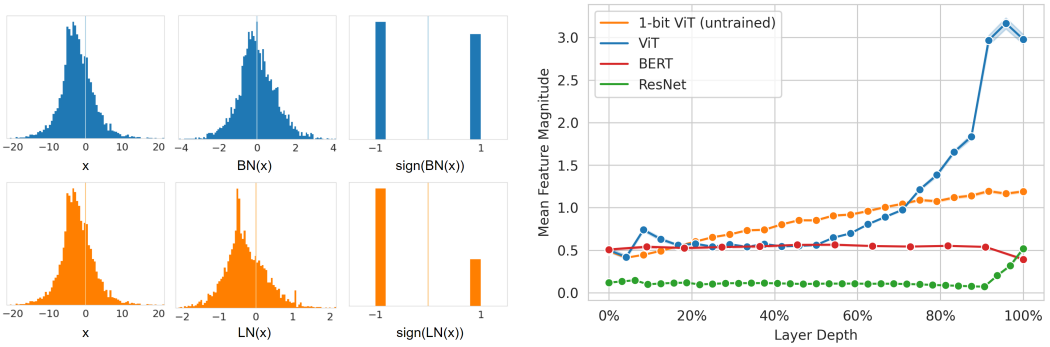
While these changes might seem subtle, they have a huge impact on 1-bit activations. In a convnet, normalization happens after each conv layer and thus after activations are quantized. However, in ViT normalization happens *before* the corresponding linear layers, and thus before the activations are quantized. Because of this, the normalization we use directly affects the distribution of binary features in a ViT. Ignoring the learned scalar transform, this takes the form

$$y = \text{sign}\left(\frac{x-\mu}{\sigma}\right) \qquad \text{where} \qquad \text{sign}(x) = \begin{cases} +1 & \text{if } x \geq 0 \\ -1 & \text{otherwise} \end{cases} \tag{1}$$

where μ and σ denote the mean and standard deviation as computed by the normalization scheme. Because $\text{sign}(x)$ ignores the scale of the input (at least in the forward pass), σ can be left out, and μ essentially becomes the threshold for the sign operation: everything greater than or equal to μ is set to $+1$, while everything below μ is set to -1.

Thus, if we want our activations to be balanced between -1 and +1, we need to be careful of which normalization scheme we use. Assuming that the mean of the incoming features is close to the median, for whatever normalization dimension we choose (e.g., batch or channel), half of the features along that dimension will be -1 and half will be +1 after taking the sign.

In the case of LayerNorm, this means -1s and +1s will be distributed evenly across the channel dimension, while with BatchNorm, they will be distributed evenly across the batch dimension (and in the limit, the dataset itself). We show an example of this difference in Fig. 2a. Because it offers a balanced distribution of -1s and +1s across the dataset, we opt to use BatchNorm over LayerNorm.



(a) BatchNorm produces more balanced post-sign distributions over the data than LayerNorm.

(b) The magnitude of skip connections in ViT increases drastically compared to other architectures.

Figure 2: **Architectural Challenges.** The ViT architecture differs in many ways from prior convnet architectures, and several of those changes are detrimental to binary quantization. LayerNorm is worse for binarization than BatchNorm (left), and the placement of norms causes intermediate features to balloon in magnitude (right), which is hard to model well with binary activations (orange line).

And because normalization is so important to balance binary weights, we apply it before every binary linear operation in the network, including the calculation of k, q, and v (see Fig. 1). These two simple changes combined lead to a massive +16.4% improvement (see Tab. 1). Just adding extra LayerNorms alone accounts for 14.7% of that improvement, while swapping the LayerNorms out for BatchNorms adds an additional 1.7% on top, with the added benefit of having a lower op count.

Note BatchNorm is faster than LayerNorm because it can be reduced to a constant at evaluation time. Moreover, because of its position in the network, it can be treated as a threshold for the sign function. Since most implementations of binary networks (that use `xor` and `popcnt`) already have to compare to a non-zero constant to compute the sign function, this makes BatchNorm essentially "free" at run-time. Thus, putting a BatchNorm before every layer actually *reduces* the operation count compared to the baseline.

3.2 Skip Connection Scale

As it turns out, placing the norm layer before its corresponding attention or linear layer has other consequences. Namely, the magnitude of the skip connection in ViT is vastly different to both BERT (Kenton & Toutanova, 2019) and convnets (Raghu et al., 2021). ViT makes this change because this pre-norm was found to perform better for deeper transformers than the post-norm that BERT uses (Wang et al., 2019). However, as we show in Fig. 2b, this results in vastly different scales for skip connection features compared to other networks.

This is a problem, then, for binarizing ViT through dense distillation, as the skip connections in a 1-bit ViT model don't conform to the same magnitudes (see Fig. 2b). To fix this, we add a generalized RPReLU operation after every skip connection as in Liu et al. (2020) that can scale the skip connection branch. This change offers a +3.4% improvement (see Tab. 1) while adding a minimal number of operations.

3.3 Mixed Precision Attention

Multi-head Self Attention (Vaswani et al., 2017), the heart of modern transformers, is extremely difficult to quantize (Qin et al., 2022). In this section, we will examine our options for reducing the compute cost of this layer. Self Attention, as used by ViT, is composed of two expensive matrix multiplications:

$$\text{MSA}(x) = Av \qquad \text{where} \qquad A = \text{softmax}\left(\frac{qk^\intercal}{\sqrt{d_h}}\right) \tag{2}$$

where d_h is the number of features for attention head h with q, k, v produced by linear layers evaluated on x. This operation is repeated for each head, making it overall quite expensive.

Block Design	a bits	OPs ($\times 10^7$)	Top-1 (%)
ViT Block*	32	230	75.98
Baseline 1-bit ViT	1	7.75	24.06
+ Norm (Sec. 3.1)		7.90	+14.7
+ LN $\rightarrow$ BN (Sec. 3.1)		7.66	+1.7
+ Scale (Sec. 3.2)		7.71	+3.4
+ MP Attn (Sec. 3.3)		8.37	+8.2
BiViT Block	1	8.37	52.05

Table 1: **Building the BiViT Block.** ViT-S/32 with 1-bit activations suffers from extremely poor performance. We fix issues in normalization, scale, and attention to create our BiViT block. Models are 32w1a using BiViT distillation (Sec. 4). OPs are for the entire network. * from Steiner et al. (2021).

Attention Design	Bits qkv/A	Attn OPs ($\times 10^5$)	Top-1 (%)
FP Softmax	32/32	23.5	54.05
1-bit Softmax	1/1	3.76	43.83
1-bit w/ DMD		3.76	48.25
1-bit w/ BN_h		3.76	49.63
MP Softmax	1/32	8.86	50.49
MP Softmax w/ BN_h		8.86	51.06
MP Sigmoid w/ BN_h		8.71	**52.05**

Table 2: **BiViT Attention.** 1-bit softmax attn (Eq. 3) performs poorly. DMD (Qin et al., 2022) or adding BN_h (Eq. 4) can help, but still result in large accuracy drops. We compromise between accuracy and OPs by using mixed precision (MP) attention, where q, k, v are 1-bit, while A is FP.

Thus, we'd like to binarize the two matrix multiplications $qk^\top$ and Av. As observed in BiBERT (Qin et al., 2022), simply binarizing the Softmax operation doesn't cut it. In our setting, this reduces overall performance by 10.22% (see Tab. 2), which is clearly unacceptable for a single module. Thus, BiBERT uses Direction Matching Distillation (DMD) with boolean attention:

$$A = \text{bool}\left(\frac{q_{\text{bin}}k_{\text{bin}}^\top}{\sqrt{d_h}}\right) \qquad \text{where} \qquad \text{bool}(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

where q_{bin} and k_{bin} are binary. While this was originally proposed for BERT (Kenton & Toutanova, 2019), it lowers the accuracy deficit to 5.80% (see Tab. 2) in our 1-bit ViT model.

However, we find that DMD is not necessary to achieve these results. The goal of DMD is to produce a good feature distribution around $\text{bool}(x)$'s threshold of 0, but we can instead simply choose a different threshold for the $\text{bool}(x)$ function that fits the features as they are. As described in Fig. 2a, this can be done by adding a BatchNorm before the bool operation:

$$A_{\text{bin}} = \text{bool}\left(\text{BN}_h(q_{\text{bin}}k_{\text{bin}}^\top)\right) \tag{4}$$

where BN_h is a scalar batch norm, one for each head. This replaces the the scale factor d_h, and because BN_h reduces to a constant threshold used for $\text{bool}(x)$ at evaluation, it's no more expensive. This outperforms boolean attention with DMD (see Tab. 2) and is much simpler.

However, while effective, it's still a 4.42% drop over full precision self attention. Instead, we propose to use mixed precision attention: 1-bit q, k, and v, but full precision A. While not binary throughout, it still significantly cuts the operation count of the layer down to a third of full precision, and most of the accuracy drop comes from quantizing A in the first place.

Out of the box and using the equation for A from Eq. 2 but with binary q, k, v, this compromise performs only 3.56% worse than full precision. However, binarizing q and k means that $q_{\text{bin}}k_{\text{bin}}^\top$ has a very limited range of magnitudes. To address this, we can simple use the same trick as before: replace the scale factor d_h with the scalar BatchNorm BN_h. This reduces the disparity further down to 3%. Finally, if we replace the Softmax operation with Sigmoid, we gain another percent, meaning our mixed precision attention performs only 2% worse than full precision, while having 3x fewer OPs:

$$A = \text{sigmoid}(\text{BN}_h(q_{\text{bin}}k_{\text{bin}}^\top)) \tag{5}$$

This mixed precision sigmoid attention gives us a +8.2% improvement in accuracy over naive 1-bit attention, while only slightly increasing the number of OPs in the network overall (see Tab. 1). Thus, we employ it in our final model, as illustrated in Fig. 1.

Note that Rastegari et al. (2016) would consider this 32×1 bit Av_{bin} multiplication to be equivalent in operations to full precision. However, we find this ignores the far fewer memory accesses in this case. As a compromise, we've assigned 32×1 bit multiplication half the OPs of a full precision matrix multiplication, though in practice it may be far less.

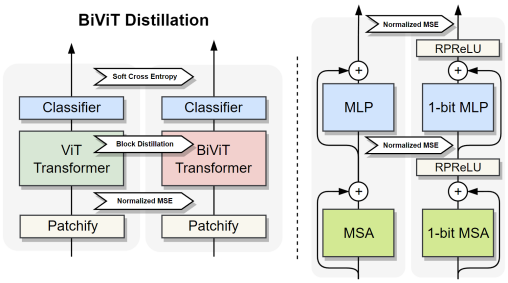


Figure 3: **BiViT Distillation.** We densely distill patch embeddings, intermediate features, and output probabilities to a 1-bit activation 32-bit weight student model. Then we fine-tune with 1-bit weights using output probability distillation.

Training Method	Epochs 32w1a	Epochs 1w1a	Top-1 (%)
Teacher Model	·	·	75.98
From Scratch	·	70	33.99
Output Distillation	·	70	36.94
Dense Distillation	·	70	20.30
w/ Normalized MSE	·	70	**37.74**
Output Distillation	30	40	38.92
BiViT Distillation	30	40	**44.77**

Table 3: **Training Methods.** 1-bit ViTs are difficult to train. Standard training and distillation methods perform very poorly. In BiViT distillation, we employ dense 32w1a pretraining into 1w1a output only fine-tuning to overcome these challenges.

4 BiViT Distillation

In order to train BiViT, we borrow a technique from NLP transformers: dense distillation of intermediate layers (Jiao et al., 2019). While this may seem directly applicable to ViT, due to the subtle differences in architecture between ViT and BERT (discussed in Sec. 3), TinyBERT distillation doesn't actually work out of the box. To show this, we use our final architecture from Sec. 3 to test different distillation methods for 1w1a training on ImageNet-1k (see Tab. 3). Here, dense distillation performs absymally: over 13% worse than just training from scratch. In this section we explore why this happens and propose a simple fix leading to a distillation method that outperforms existing techniques by almost 6%.

4.1 Dense Distillation

The original TinyBERT (Jiao et al., 2019) distills 3 components of the teacher model to the student model: intermediate features, attention matrices, and output probabilities. Like BiBERT (Bai et al., 2020), we find that the attention matrix distillation in TinyBERT does not help for binary attention, so we opt to omit it. Thus, for BiViT, we only distill intermediate features and output probabilities, as illustrated in Fig. 3.

Intermediate Features. Like in BiBERT, we sample intermediate features at every skip connection, rather than only at the end of the transformer block as in TinyBERT. Both BiBERT and TinyBERT use a simple MSE loss for intermediate features:

$$\ell_{\text{feats}}(f_s, f_t) = \text{MSE}(f_s W, f_t) \tag{6}$$

where f_s and f_t are the student and teacher features, and $W \in \mathbb{R}^{d_s \times d_t}$ is a 32-bit projection from the feature space of the student to that of the teacher (used only during training) that is randomly initialized and learned through backprop. However, while this MSE loss works for BERT, it performs very poorly for ViT (see Dense Distillation in Tab. 3).

The reason for this is simple: as discussed in Sec. 3.2, BERT uses a post-norm scheme, while ViT uses a pre-norm scheme. This causes the scale of the intermediate features to balloon with increased depth in ViT (see Fig. 2b) compared to consistent feature magnitudes in BERT. Because of this, the MSE loss in Eq. 6 is wildly imbalanced for different layers in the network, culminating in a few orders of magnitude difference between the first and last layer.

To combat this, we introduce a subtle but critical variation to the loss function in Eq. 6. Namely, since this is a scale mismatch issue, we simply normalize the features of the student and teacher so that scale isn't a factor in the loss. Specifically, in order to match the behavior of TinyBERT, we apply a parameter-less LayerNorm to both the student and teacher:

$$\hat{\ell}_{\text{feats}}(f_s, f_t) = \text{MSE}(\text{LN}(f_s W), \text{LN}(f_t)) \tag{7}$$

We use LayerNorm here instead of BatchNorm to closely match the behavior of Eq. 6 in BERT, where these LayerNorms are present in the BERT architecture. Note that we find applying W before the LayerNorm performs slightly better in our case than after.

Output Probabilities. To distill output probabilities, we find no issue with using the original Soft Target Cross Entropy as used in TinyBERT and BiBERT:

$$\ell_{\text{out}}(p_s, p_t) = \text{SCE}(p_s, p_t) \tag{8}$$

Note that while DeiT (Touvron et al., 2020) finds that using hard targets is better than soft targets, we don't observe this to be the case, at least with our limited training schedule. It could be that training longer with hard targets would outperform soft targets even with binary networks, but we leave that to future work.

4.2 Two-Stage Training

Real-to-Binary (Martinez et al., 2020) and ReActNet (Liu et al., 2020) observe higher accuracy when first training with 32w1a and then fine-tuning with 1w1a. We see this in our setting as well, and thus we incorporate this two-stage training into our design.

First Stage. In the first stage, we use the losses described in Sec. 4.1 and train with full precision weights. We place a normalized feature distillation loss $\hat{\ell}_{\text{feats}}$ after the position encoding is added in the "patchify" stem and after the two skip connections in each transformer block, for a total of $2B + 1$ loss targets as shown in Fig. 3, where B is the number of transformer blocks. Note that for the student, we use the features directly after the RPReLU, as these activations can help fit to the teacher model's feature distribution (see Appendix C). Finally, we place the output loss ℓ_{out} at the end. Unlike other ViT distillation methods (e.g. Touvron et al. (2020)), we do not use the original label when distilling outputs, as we find it to make little difference in our case.

Thus, our overall loss target for the first stage of training is:

$$L_1(s_{\text{32w1a}}, t) = \ell_{\text{out}}(p_s, p_t) + \sum_{i=0}^{2B} \hat{\ell}_{\text{feats}}^{[i]}(f_s^{[i]}, f_t^{[i]}) \tag{9}$$

where the student s_{32w1a} has 32-bit weights and 1-bit activations.

Second Stage. In the second training stage, we fine tune with binary weights using XNORNet (Rastegari et al., 2016) approximation and ReCU (Xu et al., 2021) training. As for the loss, we find that using dense distillation in the second step is too restrictive. This is the same conclusion found by Martinez et al. (2020), albeit with different loss functions. Thus, in the second stage we turn off all losses other than the output loss:

$$L_2(s_{\text{1w1a}}, t) = \ell_{\text{out}}(p_s, p_t) \tag{10}$$

This second stage formulation is similar to other ViT distillation methods Thus, our first stage becomes a "head start" for the network to quickly find a solution with floating point weights, and then we fine-tune that solution with binary weights using standard distillation.

5 Results

We use the BiViT architecture (Sec. 3) and distillation method (Sec. 4) to train several BiViT models on ImageNet-1k (Deng et al., 2009) distilled from different ViT models (Steiner et al., 2021). We compare these to their full precision teacher in Tab. 4, to existing binary convnets in Tab. 5, and fine-tune them on CIFAR in Tab. 6.

Training Configuration. For all experiments, we train using AdamW (Loshchilov & Hutter, 2017) with an effective batch size of 384 and an initial learning rate of `6e-4` with a linear decay schedule, no warm up, and a weight decay of `1e-4`. We use standard Inception-style data augmentations (Szegedy et al., 2016) along with color jitter. We train the first stage for 30 epochs and the second for another 40 epochs. Both stages use the same training parameters. For the ablations in Tab. 1 and Tab. 2, we train only the first stage as we find its accuracy representative of final performance.

Method	Bit Width w/a	Size (MiB)	BinOPs (1×10^9)	FLOPs (1×10^9)	OPs (1×10^9)	OPs Saved	Accuracy (%) Top-1	Top-5
ViT-S/32*	32/32	87.3	·	2.3	2.3	1x	75.98	93.20
BiViT-S/32	1/1	9.0	1.07	0.07	0.084	27x	44.77	69.73
BiViT-S/32 x1.5	1/1	15.4	2.41	0.10	0.138	17x	52.69	76.77
ViT-S/16*	32/32	84.1	·	9.2	9.2	1x	81.20	96.06
BiViT-S/16	1/1	5.7	4.36	0.16	0.229	40x	53.30	77.14
BiViT-S/16 x1.5	1/1	10.4	9.68	0.24	0.392	23x	60.72	83.28
ViT-B/32*	32/32	336.6	·	8.7	8.7	1x	80.70	95.62
BiViT-B/32	1/1	23.1	4.27	0.14	0.203	43x	58.23	80.91
BiViT-B/32 x1.5	1/1	42.2	9.59	0.20	0.347	25x	63.93	85.25
ViT-B/16*	32/32	330.3	·	35.1	35.1	1x	84.18	97.21
BiViT-B/16	1/1	16.4	17.09	0.33	0.601	58x	65.78	86.51
BiViT-B/16 x1.5	1/1	32.2	38.18	0.47	1.067	33x	71.07	90.07

Table 4: **BiViT Results.** Full BiViT results on the ImageNet-1k (Deng et al., 2009) validation set compared to their full precision teacher models. We also include x1.5 models with more features (see Sec. 5.1). OPs saved are relative to the corresponding FP model. See Sec. 5 for an explanation of the calculations. * models from Steiner et al. (2021). ViT FLOPs are from Zhai et al. (2021).

Binarization Method. We use XNORNet (Rastegari et al., 2016) for binary approximation and ReCU (Xu et al., 2021) with default parameters for binary training. We do not quantize the patchify stem and classification head, as is standard practice in binary quantization.

OPs and Model Size. To count the number of operations, we employ the strategy used by Liu et al. (2020), where binary and floating point operations are counted separately then reported as 1 FLOP equaling 64 BinOPs. As discussed in Sec. 3.3, we assign the mixed precision Av_{bin} operation to 32 BinOPs per mixed precision operation. Model sizes are calculated by assigning FP parameters a size of 4 bytes and binary parameters a size of 1 bit. Note that OPs are equivalent to MACs in this case.

5.1 Adding Features

In Tab. 4, we observe that the accuracy drop for larger models is lower than that of smaller models. This happens with convnets too (such as ResNet-18 vs ResNet-20 in Xu et al. (2021)), but not to this extent. ViTs rely on the features to learn inductive biases, rather than explicitly enforcing them with conv layers. Thus, a 1-bit feature space can be harsh on smaller models.

To remedy this, we use the fact that our distillation method allows us to train a student model with a different number of features compared to the teacher. In Tab. 4, we include x1.5 models with 50% more features and the same number of attention heads. We find that these x1.5 models achieve a better cost-to-performance ratio compared to the next tier of model.

5.2 Comparison to SotA ConvNets

In Tab. 5, we compare our BiViT-B models to the highly developed state-of-the-art in 1-bit convnets on ResNet-18. While BiViT does not yet reach the same speed-accuracy trade-off of these models, the fact that we can reach similar accuracies is a good signal. As the first exploration into binary vision transformers, the goal of this work is not to outright beat the established state-of-the-art, but instead to create a strong starting point and recipe for future binary vision transformers.

Moreover as discussed in Sec. 5.1, ViT's lack of implicit inductive biases that are afforded to convnets makes binarization a much harsher operation. It remains to be seen whether the same is true for transformers with inductive biases built in such as SWIN (Liu et al., 2021a) and LeViT (Graham et al., 2021). We hope future work can iterate on our design to produce even more powerful binary vision transformers.

Network	Method	OPs ($\times 10^8$)	Top-1 (%)
ResNet-18	BiRealNet	1.63	56.4
	Real-to-Binary	1.83	65.4
	ReActNet	1.63	65.9
	ReCU	1.63	66.4
ViT-B/32	**BiViT**	2.03	58.2
	BiViT x1.5	3.47	63.9
ViT-B/16	**BiViT**	6.01	65.8

Table 5: **1-bit ConvNets.** Comparison to BiReal-Net (Liu et al., 2018), Real-to-Binary (Martinez et al., 2020), ReactNet (Liu et al., 2020), and ReCU (Xu et al., 2021). Despite being the first binary transformer in vision, BiViT can reach comparable accuracies to established convnet models, though not yet at the same level of compute.

Dataset	Model	Source	Top-1 (%)
CIFAR-10	BiViT-B/32	Scratch	75.5
		Pretrain	**92.6**
	BiViT-B/16	Scratch	69.9
		Pretrain	**94.7**
CIFAR-100	BiViT-B/32	Scratch	46.3
		Pretrain	**75.8**
	BiViT-B/16	Scratch	39.1
		Pretrain	**78.2**

Table 6: **Fine-Tuning.** Fune-tuning the BiViT ImageNet-1k models trained in Tab. 4 on CIFAR (Krizhevsky et al., 2009) compared training from scratch (random initialization). BiViT is able to transfer well without additional distillation.

5.3 Fine-tuning on CIFAR

Because our method uses dense two-stage distillation on ImageNet-1k, it's not clear whether this training technique also needs to be performed when fine-tuning on downstream datasets. To test this, we fine-tune two ImageNet-1k pretrained BiViT models on CIFAR-10 and CIFAR-100 in Tab. 6 and compare it to the same models trained from scratch. For both, we use cross entropy with the same hyperparameters we use for pretraining. We train for 50 epochs when fine-tuning and 200 epochs when training from scratch for both datasets.

As expected, fine-tuning a BiViT model pretrained on ImageNet-1k produces a substantially more accurate model than training from scratch—even without distillation on the target dataset. This indicates that two-stage distillation step only needs to be performed once, and subsequent downstream tasks can use normal binary training.

5.4 Discussion

Our goal was to create a binary transformer that performs similarly to BiBERT but on vision. And to that extent, we succeeded. Despite none of the techniques from NLP carrying over to vision and many techniques from vision not being applicable, we obtain similar performance drop compared to BiBERT on the base model: 13-18% drop for BiViT on ImageNet-1k vs. 15-21% drop for BiBERT on GLUE. We break down the this accuracy drop in Appendix B and it seems that almost all of the remaining quantization error stems from binarizing activations. This is promising because concurrent work by Liu et al. (2022a) in NLP finds a huge boost in accuracy from rethinking how activations are quantized, which could also work here. We hope future work can explore this direction.

6 Conclusion

In this paper, we introduced the first fully binary ViT model, BiViT. Initially, we found that both the ViT architecture and existing binary distillation techniques produces extremely poor performance when binarizing ViTs. To address this, we introduce a BiViT architecture and dense distillation method in order to recover a lot of this lost performance. These contributions bring our BiViT in line with existing binary transformers in NLP, despite not being able to use any of the same techniques. Overall, we were able to produce a BiViT model with 58x fewer operations than its corresponding full precision ViT model while maintaining 65.78% accuracy on ImageNet-1k. We hope that this paper can be used as the first step toward accurate 1-bit Vision Transformers.

REFERENCES

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv*, 2016.

Haoli Bai, Wei Zhang, Lu Hou, Lifeng Shang, Jing Jin, Xin Jiang, Qun Liu, Michael Lyu, and Irwin King. Binarybert: Pushing the limit of bert quantization. *arXiv*, 2020.

Adrian Bulat and Georgios Tzimiropoulos. Xnor-net++: Improved binary neural networks. *arXiv*, 2019.

Xiangning Chen, Cho-Jui Hsieh, and Boqing Gong. When vision transformers outperform resnets without pre-training or strong data augmentations. *arXiv*, 2021.

Matthieu Courbariaux, Itay Hubara, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1. *arXiv*, 2016.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2020.

Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. *arXiv*, 2020.

Benjamin Graham, Alaaeldin El-Nouby, Hugo Touvron, Pierre Stock, Armand Joulin, Hervé Jégou, and Matthijs Douze. Levit: a vision transformer in convnet's clothing for faster inference. In *CVPR*, 2021.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. *arXiv*, 2021.

Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. Distilling the knowledge in a neural network. *arXiv*, 2015.

Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv*, 2017.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. Tinybert: Distilling bert for natural language understanding. *arXiv*, 2019.

Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, 2019.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Guillaume Leclerc, Andrew Ilyas, Logan Engstrom, Sung Min Park, Hadi Salman, and Aleksander Madry. ffcv. https://github.com/libffcv/ffcv/, 2022.

Zhexin Li, Tong Yang, Peisong Wang, and Jian Cheng. Q-vit: Fully differentiable quantization for vision transformer. *arXiv*, 2022.

Mingbao Lin, Rongrong Ji, Zihan Xu, Baochang Zhang, Yan Wang, Yongjian Wu, Feiyue Huang, and Chia-Wen Lin. Rotated binary neural network. *NeurIPS*, 2020.

Yang Lin, Tianyu Zhang, Peiqin Sun, Zheng Li, and Shuchang Zhou. Fq-vit: Fully quantized vision transformer without retraining. *arXiv*, 2021.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021a.

Zechun Liu, Baoyuan Wu, Wenhan Luo, Xin Yang, Wei Liu, and Kwang-Ting Cheng. Bi-real net: Enhancing the performance of 1-bit cnns with improved representational capability and advanced training algorithm. In *ECCV*, 2018.

Zechun Liu, Zhiqiang Shen, Marios Savvides, and Kwang-Ting Cheng. Reactnet: Towards precise binary neural network with generalized activation functions. *arXiv*, 2020.

Zechun Liu, Barlas Oguz, Aasish Pappu, Lin Xiao, Scott Yih, Meng Li, Raghuraman Krishnamoorthi, and Yashar Mehdad. Bit: Robustly binarized multi-distilled transformer. *arXiv*, 2022a.

Zhenhua Liu, Yunhe Wang, Kai Han, Wei Zhang, Siwei Ma, and Wen Gao. Post-training quantization for vision transformer. *NeurIPS*, 2021b.

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. *arXiv*, 2022b.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv*, 2017.

Brais Martinez, Jing Yang, Adrian Bulat, and Georgios Tzimiropoulos. Training binary neural networks with real-to-binary convolutions. *arXiv*, 2020.

Haotong Qin, Yifu Ding, Mingyuan Zhang, Qinghua Yan, Aishan Liu, Qingqing Dang, Ziwei Liu, and Xianglong Liu. Bibert: Accurate fully binarized bert. *arXiv*, 2022.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 2019.

Maithra Raghu, Thomas Unterthiner, Simon Kornblith, Chiyuan Zhang, and Alexey Dosovitskiy. Do vision transformers see like convolutional neural networks? *NeurIPS*, 2021.

Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *ECCV*, 2016.

Tal Ridnik, Hussam Lawen, Emanuel Ben-Baruch, and Asaf Noy. Solving imagenet: a unified scheme for training any backbone to top results. *arXiv*, 2022.

Andreas Steiner, Alexander Kolesnikov, Xiaohua Zhai, Ross Wightman, Jakob Uszkoreit, and Lucas Beyer. How to train your vit? data, augmentation, and regularization in vision transformers. *arXiv*, 2021.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *CVPR*, 2016.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. *arXiv*, 2020.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *NeurIPS*, 2017.

Qiang Wang, Bei Li, Tong Xiao, Jingbo Zhu, Changliang Li, Derek F Wong, and Lidia S Chao. Learning deep transformer models for machine translation. *arXiv*, 2019.

Zhenda Xie, Zheng Zhang, Yue Cao, Yutong Lin, Jianmin Bao, Zhuliang Yao, Qi Dai, and Han Hu. Simmim: A simple framework for masked image modeling. *arXiv*, 2021.

Sheng Xu, Yanjing Li, Teli Ma, Bohan Zeng, Baochang Zhang, Peng Gao, and Jinhu Lu. Tervit: An efficient ternary vision transformer. *arXiv*, 2022.

Zihan Xu, Mingbao Lin, Jianzhuang Liu, Jie Chen, Ling Shao, Yue Gao, Yonghong Tian, and Rongrong Ji. Recu: Reviving the dead weights in binary neural networks. *arXiv*, 2021.

Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling vision transformers. *arXiv*, 2021.

A COMPARISON TO RESNET-18

Because ViT-B/16 is quite a large model, it's hard to gauge if BiViT-B/16 x1.5 offers good performance for its operation count. Thus, we also compare it against full precision ResNet-18 in Tab. 7. Compared to ResNet-18, BiViT-B/16 x1.5 has a smaller model size, is more accurate, and has 1.7x fewer ops. While there are of course more efficient models than a full precision ResNet-18, this shows that binary vision transformers can offer promising level of performance, even when most parameters and activations are 1-bit.

Method	Bit Width w/a	Model Size (MiB)	BinOPs (1×10^9)	FLOPs (1×10^9)	OPs (1×10^9)	OPs Saved	Accuracy Top-1 (%)
ResNet-18 FP	32/32	44.7	·	1.83	1.83	1.0x	69.3
BiViT-B/16 x1.5	1/1	**32.2**	38.18	0.47	**1.07**	**1.7x**	**71.1**

Table 7: **Comparison to ResNet-18.** Our BiViT-B/16 x1.5 model is 1.8% more accurate on ImageNet-1k than a full precision ResNet-18 (He et al., 2016) while having 1.7x fewer ops and a smaller model size. ResNet-18 numbers were obtained from Martinez et al. (2020).

B ACCURACY DROP BREAKDOWN

BiViT distillation trains first with 1-bit activations and 32-bit weights and then fine tunes with 1-bit activations and 1-bit weights. By comparing the accuracy of the model at different training stages, we can gauge how much of the accuracy loss compared to full precision is due to binary activations, and how much is is due to binary weights. We display the results of this comparison in Tab. 8.

Architecture	Image Size	Accuracy ViT 32/32	→	BiViT (ours) 32/1	→	1/1
S/32	224	75.98	(-24.48)	51.50	(-6.73)	44.77
S/32 x1.5	224	75.98	(-16.81)	59.17	(-6.48)	52.69
S/16	224	81.20	(-18.85)	62.35	(-9.05)	53.30
S/16 x1.5	224	81.20	(-12.74)	68.46	(-7.74)	60.72
B/32	224	80.70	(-16.83)	63.87	(-5.64)	58.23
B/32 x1.5	224	80.70	(-11.34)	69.36	(-5.43)	63.93
B/16	224	84.18	(-12.61)	71.57	(-5.79)	65.78
B/16 x1.5	224	84.18	(-10.50)	73.68	(-2.61)	71.07

Table 8: **Quantization Error.** The breakdown of accuracy lost due to quantizing activations (32/1) and then subsequently quantizing weights (1/1) from the original full precision ViT teacher model (32/32). As described in the paper, most of the quantization error comes from 1-bit activations.

B.1 1-BIT ACTIVATIONS

Most of the quantization error occurs in the transition between full precision (32/32) and the first stage of BiViT distillation with full precision weights and 1-bit activations (32/1). In fact, for the smaller models this drop is extreme to the point of making these models unusable in practice (e.g., a 24.48% drop for S/32). As mentioned in the paper, quantizing to 1-bit activations is much more taxing on the model than 1-bit weights. This could be due to a lack of inductive biases, as the model has to learn a good feature space without any task-specific assistance, which binary activations restrict.

B.2 1-BIT WEIGHTS

Even for the smaller models, the drop in accuracy between stages 1 and 2 of BiViT distillation (32/1 → 1/1, i.e. binarizing weights) is significantly smaller than that for 1-bit activations. For the

largest model (B/16 x1.5), this only amounts to a 2.61% drop. For future work, it would likely be beneficial to focus on improving the feature space of a binary transformer, either by adding inductive biases or using better activation functions.

B.3 Adding Features

One way to improve the feature space of quantized models is simply to add more parameters. In the main paper, we introduce x1.5 models with a 50% larger hidden dimension compared the corresponding teacher. This increases the number of parameters in the network by increasing the size of each weight matrix. However, despite the primary increase in computation being due to extra weights, the accuracy boost we get from these models actually comes from the activations.

In fact, for most of the models (especially the /32 models), multiplying the hidden dimension by 1.5 has a negligible effect on the quantization error from 1-bit weights. Instead, most of the benefit comes from less activation quantization error. This lends more support to the idea that binary ViTs suffer more from reducing the representational capacity of the feature space than from the weights.

C RPReLU Distribution

In the main paper, we note that RPReLU helps the student BiViT model better learn the teacher ViT model's feature distribution. In order to show this, we conduct the following experiment.

For a batch of images, we take the input and output of one transformer block in the teacher model. Then, we attempt to fit a binary transformer model to the teacher block's features using MSE. Finally, we repeat this experiment with and without RPReLU and report the final MSE loss and visualize the learned feature distributions.

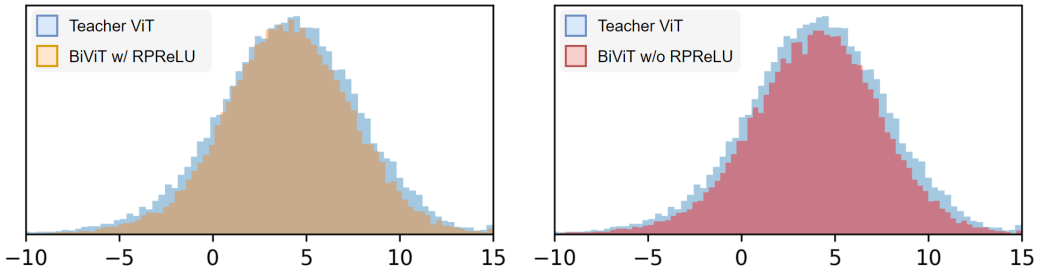


Figure 4: **Adding RPReLU.** We show that adding RPReLU on the skip connection branch improves performance when distilling from a floating point teacher model. Here we show that slight improvement in learned distribution for a single layer. Note that this slight benefit compounds over the 12 layers of the network.

The BiViT block with RPReLUs obtains a final MSE loss after convergence of 5.22. Adding RPReLUs in the skip connection branch lowers this loss down to 4.77. This result can be seen visually in Fig. 4, where the BiViT block with RPReLU (orange) matches the teacher distribution (blue) slightly better than the block without RPReLU (red). While this difference in one block is small, it adds up over the 12 layers of the network to equal a +3.4% accuracy improvement.