
ViBid: Linear Vision Transformer with Bidirectional Normalization (Supplementary Material)

Jeongeun Song^{1,*}

Heung-Chang Lee^{1,*}

¹AI Lab & Service, Kakao Enterprise, Seongnam-si, South Korea

*Equal Contributions

Table 1: **Hyperparameter settings for our various models on ImageNet1k dataset.** Values in parentheses "(") mean values used in fine-tuning.

Hyperparameter	ViBid-U	ViBid-T	ViBid-S	ViBid-M	ViBid-B
Learning rate	5e-5			4e-5 (0.01)	
Warm-up LR	1e-6 (None)				
Batch size	4096 (4096)				
Optimizer	AdamW (SGD)				
LR scheduler	Cosine (Cosine)				
Gradient clip	0.5 (0.5)				
Stochastic depth	0.0	0.05	0.1	0.15 (0.15)	0.25 (0.25)
Warm-up epochs	5 (0)				
RandAugment	2, 7			2, 9 (2, 9)	2, 12 (2, 12)
Label smoothing	0.1 (0.1)				
Train epochs	400 (10)				
Weight decay	0.05 (0.0)				

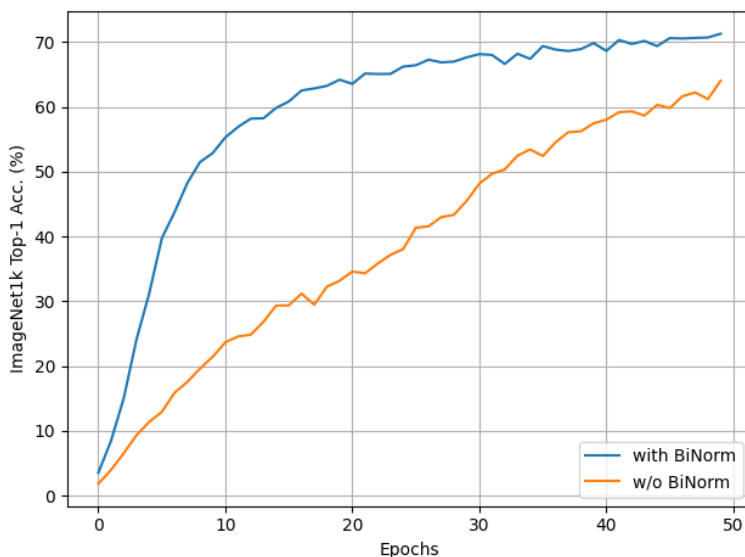


Figure 1: **Comparison of the effects of BiNorm presence or absence on early epoch training by using ViBid-M.** See Section 3 for details.

Algorithm 1: Python style pseudo-code of BiNorm-based attention.

```
1 def attend(sel f, x):
2     b, n, d = x.shape
3     qkv = sel f.qkv_proj(x)
4     h = qkv.shape[-1] // 3
5     qkv = qkv.reshape(b, n, 3, sel f.num_heads, h // sel f.num_heads)
6     qkv = qkv.permute(2, 0, 3, 1, 4)
7
8     q = output[0]
9     k = output[1]
10    v = output[2]
11
12    # we commented the lines of the original SA
13    # output = (q @ k.transpose(-2, -1)) * sel f.scale
14    # output = output.softmax(dim=-1)
15    output = k.transpose(-2, -1) @ v
16    output = normalize(output, dim=-2)
17    q = normalize(q, dim=-1) # Bi Norm
18
19    # output = (output @ v).transpose(1, 2).reshape(b, n, h)
20    output = (q @ output).reshape(b, n, h)
21    output = sel f.proj(output)
22    return output
```

Figure 2: **Visualized attentions.** Visualization of attention matrices using pseudo-inverse scheme. These matrices are extracted from class attention module of pretrained ViBid-S.