Separable Batch Normalization for Robust Facial Landmark Localization Supplementary Material

BMVC 2021 Submission # 585

Abstract

In this document, more implementation details are provided to facilitate the reimplementation of our work, and more ablation study experiments results are reported to explain our choice of hyper-parameters. Additionally, diverse qualitative results are presented to show the effect of our SepBN module.

1 Implementation Details

All our experiments were implemented in PyTorch on a platform with one Intel Core i7-7700K CPU and one NVIDIA GeForce GTX 1080Ti GPU. We used the Stochastic Gradient Descent (SGD) optimizer and set the weight decay to 5e-4, momentum to 0.9 and batch size to 8. For learning rate, we used the cosine annealing scheduler and set the maximum and minimum learning rate to $1e-3$ and $5e-6$, with 120 warm-up epochs. Each model was trained for 500 epochs with the L1 loss function.

We followed [2] to perform data augmentation using the geometric augmentation methods. We randomly rotated the training images between [-25, 25] and perturbed the bounding box corner within 15% of the bounding box size. In addition, horizontal flip and shear transformations were applied. For a fair comparison, all the test images were cropped and resized according to the bounding boxes provided without any transformation.

Following [1], the temperature-controlled softmax function with an extra parameter τ is used in the attention network as:

$$\pi = \frac{\exp(\mathcal{X}_{ex}/\tau)}{\sum_k \exp(\mathcal{X}_{ex}/\tau)}. \tag{1}$$

In this equation, the original softmax function can be considered as a special case of the temperature-controlled softmax function, *i.e.* $\tau = 1$. The temperature parameter adjusts the sensitivity of the softmax function to the characteristics of the input data. By setting τ to a larger value, the output tends to a uniform distribution. According to [1], the temperature-based annealing strategy helps the network converge better and provides better performance by setting a large temperature value for early epochs and gradually reducing it to 1 as the training progresses. The initial $\tau = 30$ is applied in our experiments and this number drops to 1 in the first 30 training epochs.

2 More Ablation Study

We performed a number of ablation studies on AFLW and WFLW using our Vanilla CNN to find the best configuration of the module.

Aggregation method: There are two methods to obtain the final output of the SepBN module, hard aggregation (*i.e.* $\hat{\gamma}_{n,g} = \arg\max_k \pi_{n,g,k}\gamma_k$, $\hat{\beta}_{n,g} = \arg\max_k \pi_{n,g,k}\beta_k$) and soft aggregation (*i.e.* $\hat{\gamma}_{n,g} = \sum_{k=1}^{K} \pi_{n,g,k}\gamma_k$, $\hat{\beta}_{n,g} = \sum_{k=1}^{K} \pi_{n,g,k}\beta_k$). The results of these two different aggregation methods obtained on AFLW using our Vanilla CNN are shown in Table 1. We can see that the soft aggregation method performs much better. When hard-aggregation is applied, the parameters of the attention part cannot be effectively optimized, resulting in random attention weights, so the final performance is even worse than that of the baseline method using the classical BN layer.

Temperature annealing of attention: For SepBN, the hyper-parameter τ used in the temperature-controlled softmax function can be set to a fixed value or adjusted in the annealing way. We report the results obtained by different settings in Table 2. Clearly, the annealing strategy outperforms all the other configurations.

Number of separate routes K: The number of sets of mapping parameters (*i.e.* K) may affect the performance of the trained network as well. We report the results of our Vanilla CNN trained with different values of K in Table 2. We can see that, by setting K to 3, we have the best result. Even though $K = 7$ brings the same result, we prefer to choose the smaller one to reduce the growth of the number of parameters.

Most suitable configuration in Vanilla CNN: To find out how SepBN modules affect the performance across layers, we replaced the classical BN layers with the SepBN modules after different convolution operations in Vanilla CNN and compared the performance on WFLW in Table 3. The results show that replacing all the original BN layers with SepBN *except the last one* achieves the best performance for our Vanilla CNN.

Application to Modern Network architectures: For the use of our SepBN in ResNeXt-50 and MobileNetV2, experiments were conducted to find the best solution to combine Bottleneck blocks with the proposed SepBN module. As shown in Table 4, by replacing the original BN_3 in the Bottelneck blocks with our proposed SepBN module, both ResNeXt-50 and MobileNetV2 show the largest performance boost as compared with the baseline network.

3 Qualitative Results

Additional qualitative results are illustrated in this section. Figure 1 , 2 , 3 are the visualization results of the K sets of scale and shift parameters learned by the SepBN modules in Vanilla CNN, MobileNetV2 and ResNeXt50 on COFW, respectively. And Figure 4 , 5 , 6 are the visualization results of the K sets of scale and shift parameters learned by the SepBN modules

Table 1: Hard-aggregation vs. Soft-aggregation

	NME(%)	
	AFLW-Full	AFLW-Frontal
Vanilla CNN (BN)	1.65	1.40
Vanilla CNN (SepBN, Hard-aggre)	1.86	1.60
Vanilla CNN (SepBN, Soft-aggre)	**1.55**	**1.39**

Table 2: A comparison of different τ(left) and K(right) configurations on the AFLW dataset using the Vanilla CNN network.

Temperature	NME(%)
$\tau = 1$	1.69
$\tau = 5$	1.63
$\tau = 10$	1.64
$\tau = 20$	1.67
$\tau = 30$	1.66
Annealing τ	**1.55**

Separated routes	NME(%)
$K = 2$	1.62
$K = 3$	**1.55**
$K = 4$	1.58
$K = 5$	1.59
$K = 6$	1.56
$K = 7$	**1.55**

Table 3: A study of the impact of the SepBN modules after different convolution operations in Vanilla CNN. ✓ indicates replacing BN with SepBN, – indicates reserving the classical standard BN.

Convolution operations in Vanilla CNN						WFLW-test
Conv1	Conv2	Conv3	Conv4	Conv5	Conv6	NME(%)
–	–	–	–	–	✓	5.76
–	–	–	–	✓	✓	5.70
–	–	–	✓	✓	✓	5.64
–	–	✓	✓	✓	✓	5.66
–	✓	✓	✓	✓	✓	5.57
✓	✓	✓	✓	✓	✓	5.49
✓	✓	✓	✓	✓	–	**5.48**
✓	✓	✓	✓	–	–	5.53
✓	✓	✓	–	–	–	5.59
✓	✓	–	–	–	–	5.65
✓	–	–	–	–	–	5.69
–	–	–	–	–	–	5.70

Table 4: A study of the best combination solution of SepBN module and Bottleneck Block validated on COFW with ResNeXt-50 (left) and MobileNetV2 (right). ✓ indicates replacing BN with SepBN, – indicates reserving the original standard BN, NaN indicates the network shows no convergence.

Different locations			COFW
BN_1	BN_2	BN_3	NME(%)
–	–	–	3.95
✓	–	–	4.03
–	✓	–	4.12
–	–	✓	**3.51**
✓	✓	–	4.03
✓	–	✓	3.80
–	✓	✓	NaN
✓	✓	✓	NaN

Different locations			COFW
BN_1	BN_2	BN_3	NME(%)
–	–	–	5.07
✓	–	–	4.07
–	✓	–	4.03
–	–	✓	**3.96**
✓	✓	–	4.08
✓	–	✓	4.02
–	✓	✓	4.10
✓	✓	✓	4.05

in Vanilla CNN, MobileNetV2 and ResNeXt50 on WFLW, respectively.

As can be told from these 6 figures, when SepBN module is really useful to improve performance, the learned K sets of mapping parameters show great differences as shown in Figure 2 , 3 , 4 and 5 . In contrast, when it is difficult for SepBN to effectively improve

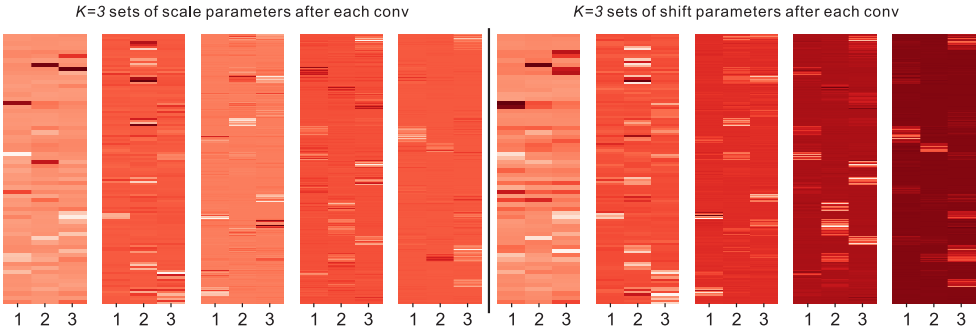

Figure 1: Visualization of the 5 SepBN modules with $K = 3$ sets of mapping parameters (left: scale parameters; right: shift parameters) applied in Vanilla CNN trained on COFW.

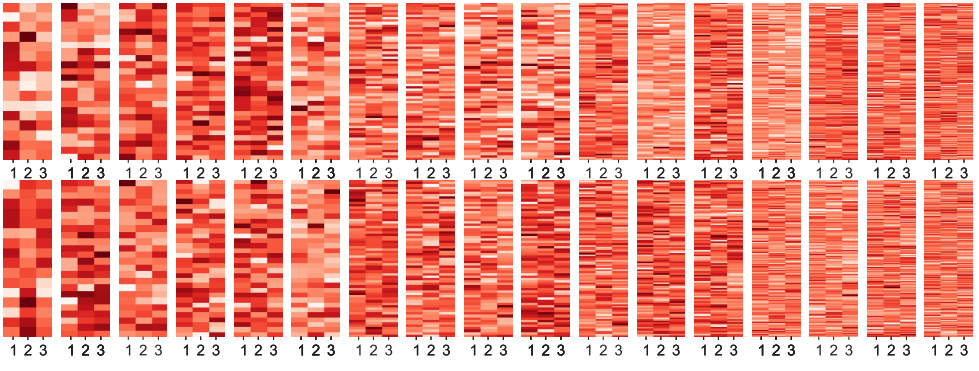

Figure 2: Visualization of the 17 SepBN modules with $K = 3$ sets of mapping parameters (up: scale parameters; down: shift parameters) applied in MobileNetV2 trained on COFW.

performance, the K sets of parameters learned will be very similar as shown in Figure 1 and 6 , in other words, it almost degenerates into a classical BN layer. We can also use this to judge whether SepBN is really beneficial to the network learning.

Figure 7 illustrates the face image and its corresponding attention weights ($G = 2$, $K = 3$, $G * K = 6$ in column) of the 5 SepBN modules (column by column, 5 columns in total), using Vanilla CNN network trained on WFLW. Face images are randomly chosen from the WFLW test set. It can be known from the picture that the network does output different attention weights for different images, and finally generates feature-adaptive mapping parameters for mapping.

References

[1] Yinpeng Chen, Xiyang Dai, Mengchen Liu, Dongdong Chen, Lu Yuan, and Zicheng Liu. Dynamic convolution: Attention over convolution kernels. In *Proceedings of the*

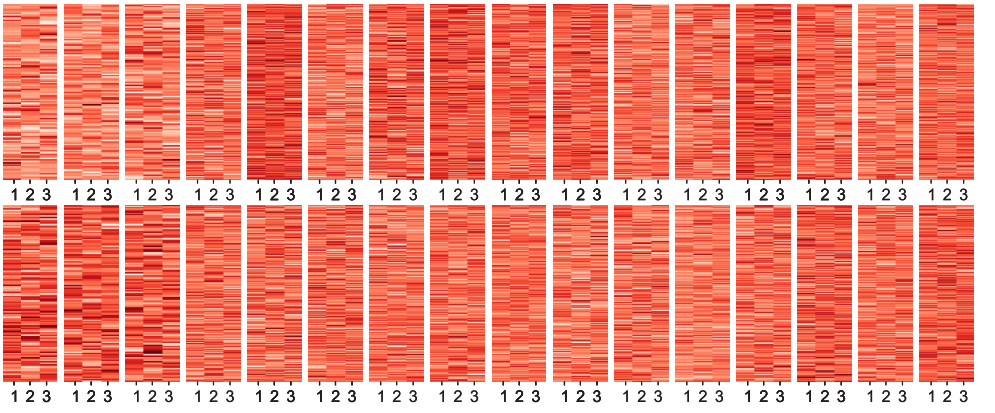

Figure 3: Visualization of the 16 SepBN modules with $K = 3$ sets of mapping parameters (up: scale parameters; down: shift parameters) applied in ResNeXt50 trained on COFW.

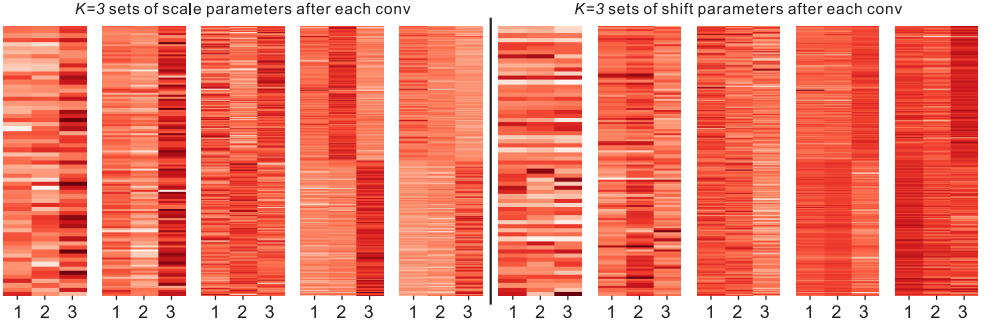

Figure 4: Visualization of the 5 SepBN modules with $K = 3$ sets of mapping parameters (left: scale parameters; right: shift parameters) applied in Vanilla CNN trained on WFLW.

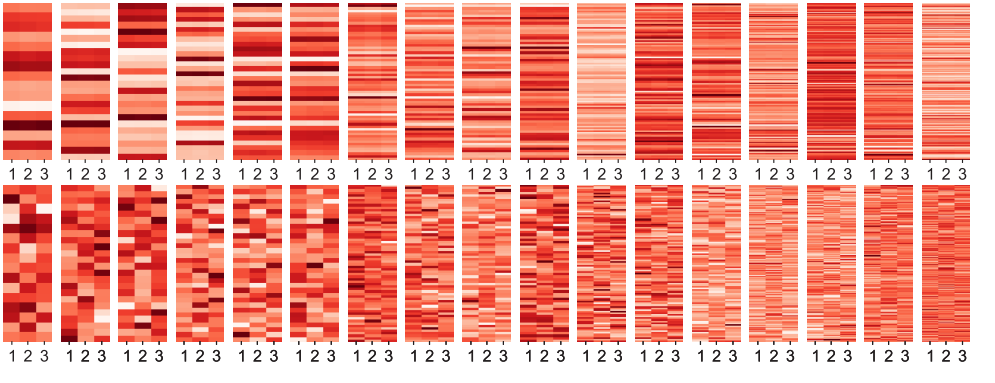

Figure 5: Visualization of the 17 SepBN modules with $K = 3$ sets of mapping parameters (up: scale parameters; down: shift parameters) applied in MobileNetV2 trained on WFLW.

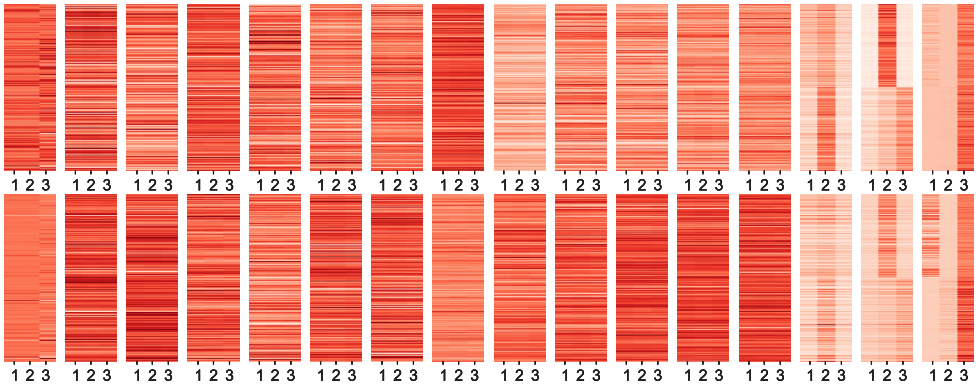

Figure 6: Visualization of the 16 SepBN modules with $K = 3$ sets of mapping parameters (up: scale parameters; down: shift parameters) applied in ResNeXt50 trained on WFLW.

IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 11030–11039, 2020.

[2] Zhen-Hua Feng, Josef Kittler, and Xiao-Jun Wu. Mining hard augmented samples for robust facial landmark localization with cnns. *IEEE Signal Processing Letters*, 26(3): 450–454, 2019.

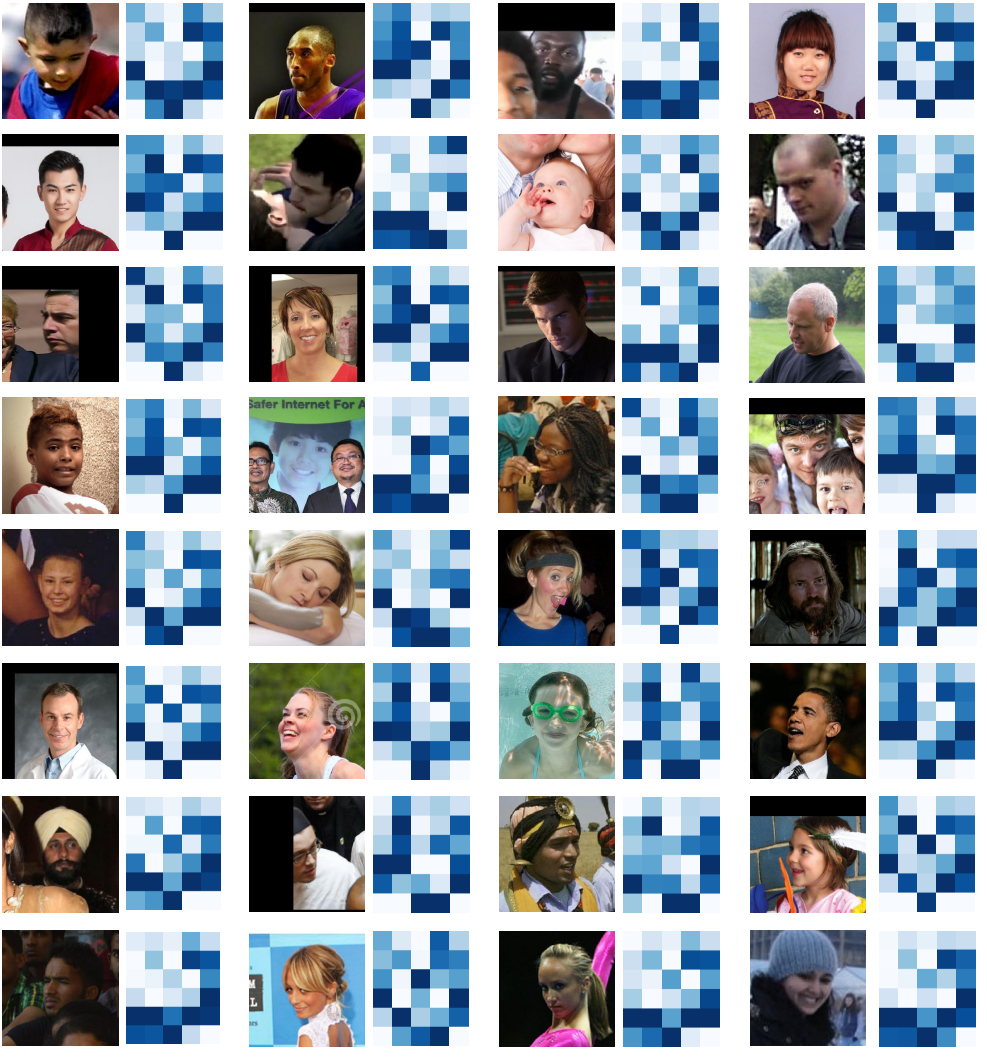

Figure 7: Visualization of the attention weights generated by the 5 SepBN modules (column by column, 5 columns in total) in the Vanilla CNN network trained on WFLW.