Filter Response Normalization Layer: Eliminating Batch Dependence in the Training of Deep Neural Networks[1]

September 23, 2020

[1] "Filter Response Normalization Layer: Eliminating Batch Dependence in the Training of Deep Neural Networks" by Singh and Krishnan, 2020

Overview

- BatchNorm performance degrades heavily when batch size is small
- A lot of other normalization techniques have been proposed (LayerNorm, GroupNorm, etc), but they are insufficient
- Authors propose two things:
  - *Filter Response Normalization*:

$$y_i = \gamma \frac{x_i}{\sqrt{\nu^2 + \epsilon}} + \beta, \qquad \text{where } \nu^2 = \sum_i x_i^2 / N \tag{1}$$

  - *Thresholded Linear Unit*

$$z_i = \max(y_i, \tau) \tag{2}$$

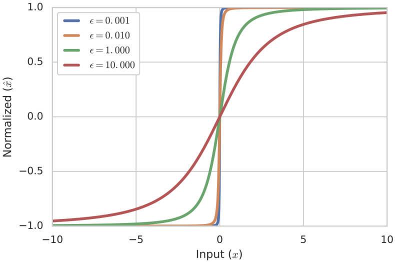
- These two techniques work the best when used in combination
- Authors demonstrate superior performance for different batch sizes (from small to large ones) on ImageNet classification and MS-COCO object detection over other normalization methods

Filter Response Normalization (FRN)

We compute the norm across spatial locations and normalize the input with it:

$$y_i = \gamma \frac{x_i}{\sqrt{\nu^2 + \epsilon}} + \beta, \qquad \text{where } \nu^2 = \sum_i x_i^2 / N \tag{3}$$

A problem with FRN occurs when $N = H \times W$ is too small: in this case small values of ϵ diverge the procedure into a sign function:



That's why authors initialize $\epsilon = 0.0001$ for $N = 1$ and learn it.

Thresholded Linear Unit (TLU)

- One of the main differences between FRN and BN is that FRN does not keep activations zero-centered (by subtracting the mean)
- This may make them deviate arbitrarily far away from zero
- And this consequently pushes ReLU into bad zones (all-zeros or all-linear)
- That's why authors turn ReLU activation into TLU:

$$\boldsymbol{z} = \max(\boldsymbol{y}, \tau) \tag{4}$$

  where τ is a learnable bias.
- In practice, it showed to perform well

Classification results on ImageNet

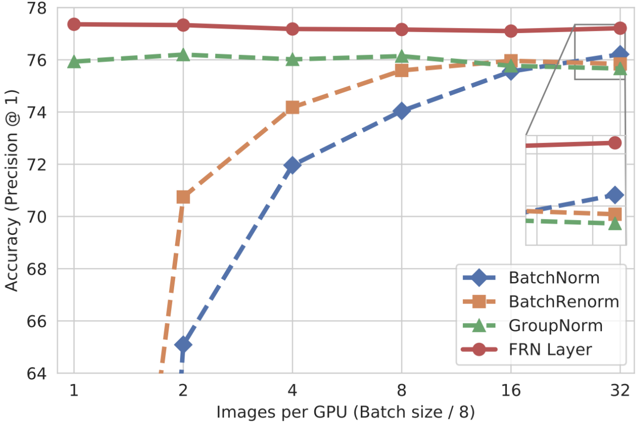


Figure: ImageNet results for ResNetV2-50

MS-COCO object detection results

Method	AP			AP^{50}			AP^{75}		
imgs/gpu	8	4	2	8	4	2	8	4	2
BN*	38.3	37.1	32.9	57.2	55.4	49.1	41.5	40.4	35.9
BN	38.7	37.9	30.2	56.6	55.2	44.5	42.1	41.4	32.5
GN	39.3	39.0	38.7	57.8	57.5	56.9	42.6	42.3	41.8
FRN	**39.6**	**39.5**	**39.1**	**58.5**	**58.4**	**57.5**	**43.1**	**43.3**	**42.3**

Figure: RetinaNet with Resnet101 FPN backbone

FRN/TLU ablation

Method	P@1	R@5
BN + $\max(x, 0)$ (ReLU)	76.21	92.98
BN + $\max(x, \tau)$ (TLU)	76.03	92.94
FRN + $\max(x, 0)$ (ReLU)	75.24	92.65
$\max(x, \kappa x)$ (PReLU) [12]	76.43	93.30
$\max(x, \kappa x + \tau)$ (Affine-TLU)	76.71	93.32
$\max(x, \tau)$ (TLU)	**77.21**	**93.57**

Figure: Results for difference normalization/activation setups for ImageNet classification

Final thoughts

- Other normalization techniques (LN, GN, etc) work better than BN on small batch sizes, but worse on large ones
- Authors managed to “take the best of the both worlds”
- Authors argue that LN/GN perform poorly because they create additional correlations between channels
  - That’s an interesting perspective
- It’s interesting to see that parameters ϵ and τ are learnt to meaningful values
  - Usually such kind of parameters are not optimized well