Как жить без batch-norm?

Иванов Семен

Что такое batch-norm?

Слой batch-norm

1. В слой вводятся обучаемые параметры μ и σ^2
2. Получаем на вход батч $B = \{x_1, \cdots, x_n\}$
3. Стандартизуем батч. Получаем $\hat{B} = \{\hat{x}_1, \cdots, \hat{x}_n\}$
4. Перешкалируем и возвращаем $y_i = \sigma\hat{x}_i + \mu$

- Заметим, что качество ухудшиться не должно, так как слой допускает тождественное преобразование

Плюсы

- Можно повышать lr без риска разойтись → ускорение сходимости
- Решается проблема с нестабильными градиентами
- Качество лучше, скорость выше
- Сеть меньше зависит от удачной инициализации
- Может заменить dropout

  сдвиг и смещение на случайные для сэмпла значения выступают в роли регуляризации

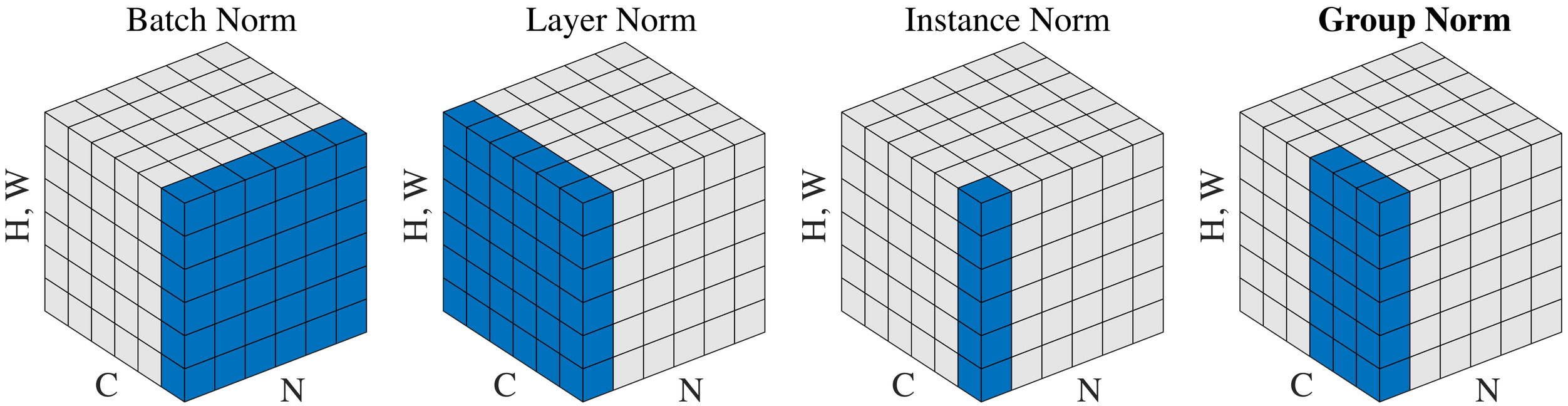
Internal covariate shift

~~Коварный~~ Ковариантный сдвиг?

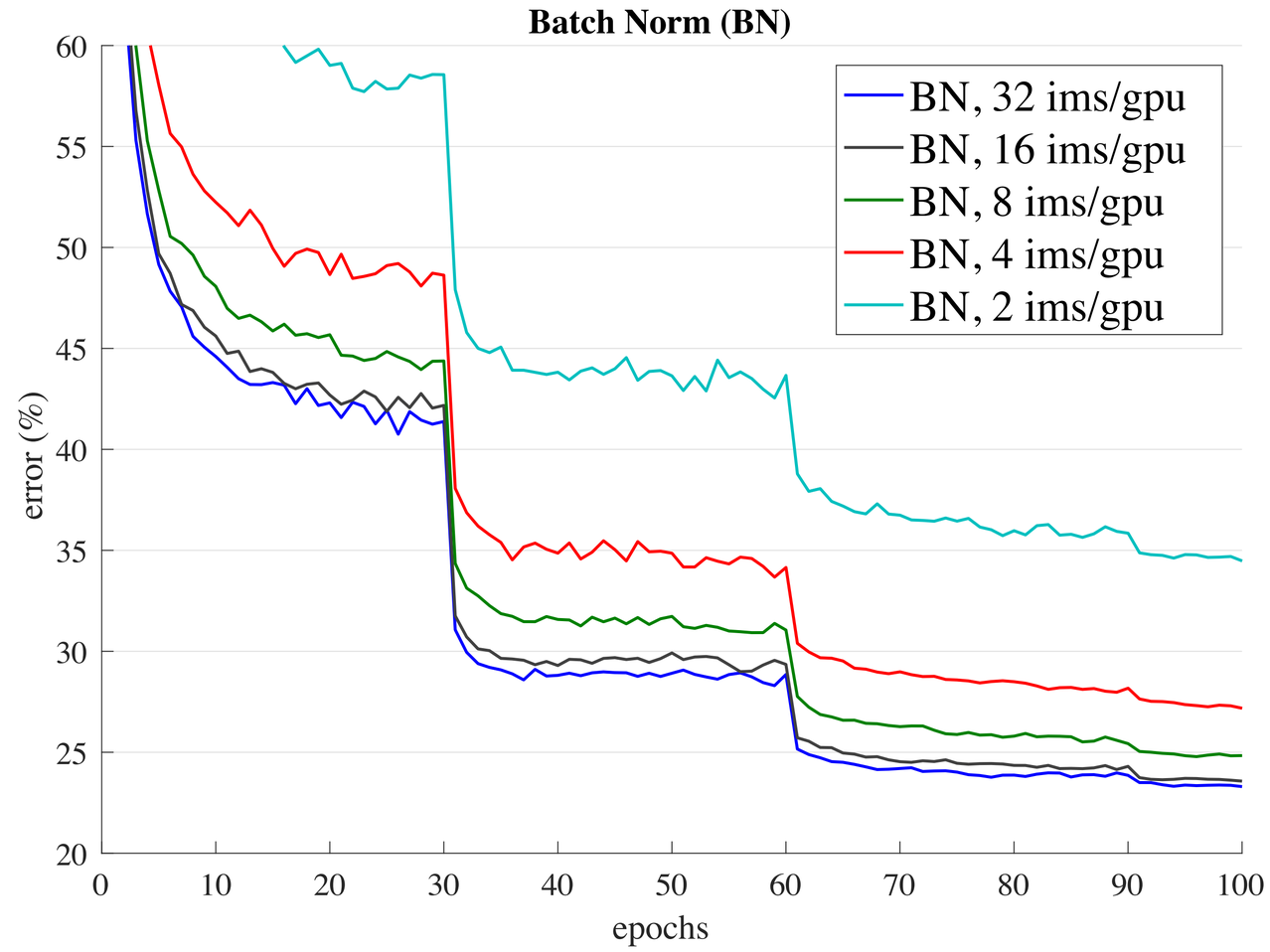
- Утверждалось, что BN помогает избавиться от сдвига в распределении входов для внутренних слоев
- Однако оказалось, что это не совсем так
- Иногда он может быть даже больше, чем без BN
- Эмпирически выяснилось, что BN делает функцию потерь и ее градиент “более” Липшицевой

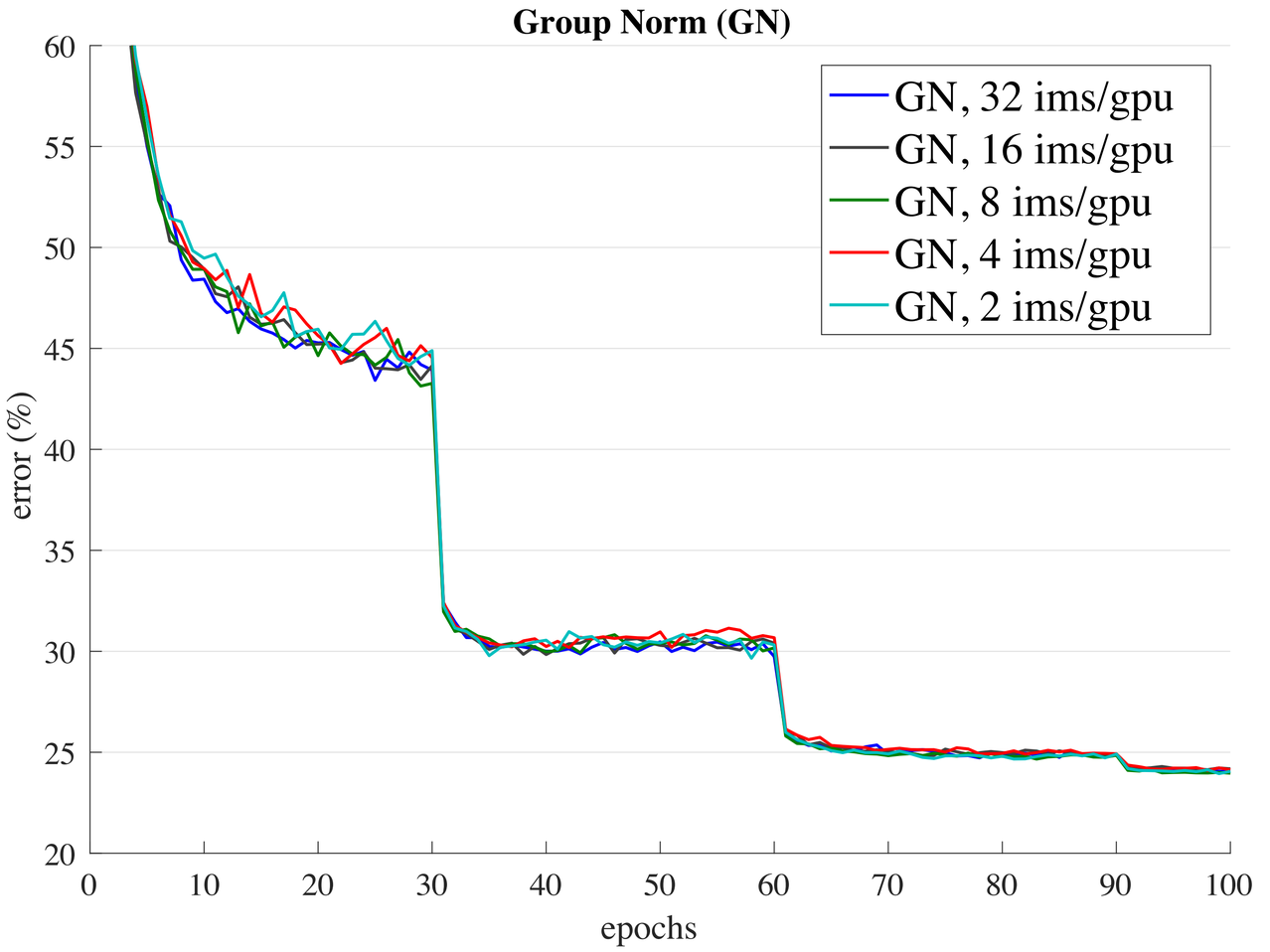
Минусы

- Маленькие батчи – слишком шумно, а большие – иногда ограничивает сложность других компонент
- Проблемнее обучать объемные модели
- Зависимость внутри батча
- На больших батчах будут проблемы с распределенными вычислениями

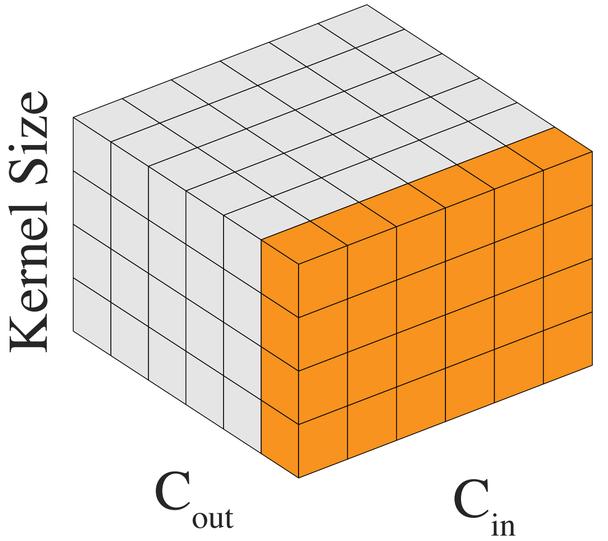


- Нормализуем, группируя по нескольким каналам для одного примера
- Теперь не зависим от размера батча
- Если брать маленькие батчи, то GN обходит BN
- Просто перейти от BN к GN не получится. Настроенный BN с правильным размером батча все равно лучше

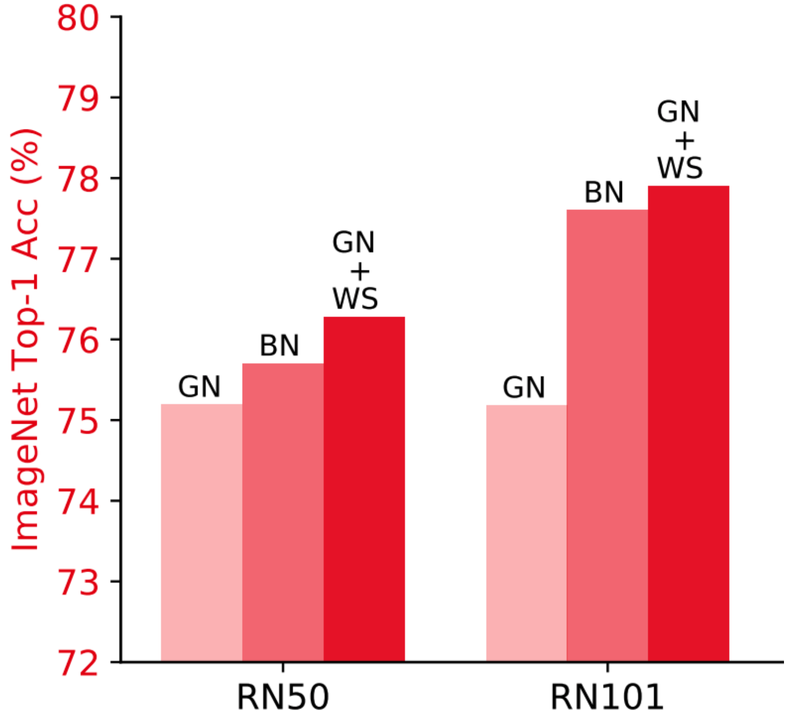

Batch Norm (BN)
BN, 32 ims/gpu
BN, 16 ims/gpu
BN, 8 ims/gpu
BN, 4 ims/gpu
BN, 2 ims/gpu
error (%)
epochs
60
55
50
45
40
35
30
25
20
0
10
20
30
40
50
60
70
80
90
100


Group Norm (GN)
GN, 32 ims/gpu
GN, 16 ims/gpu
GN, 8 ims/gpu
GN, 4 ims/gpu
GN, 2 ims/gpu
error (%)
epochs
60
55
50
45
40
35
30
25
20
0
10
20
30
40
50
60
70
80
90
100

Weight Standardization + GN



- **Weight Standardization**: стандартизуем веса слоя по каждому выходному каналу (то есть по сверткам)
- Комбинация WS+GN может обходить BN
- WS уменьшает Липшицеву константу
- $W \in \mathbb{R}^{C_{out} * C_{in} K^2}$, чтобы $y = Wx$ давал вектор значений в точке по каждому выходному каналу



$$\widehat{W}_{ij} = \gamma \frac{W_{ij} - \mu_{W_i}}{\sigma_{W_i}}$$

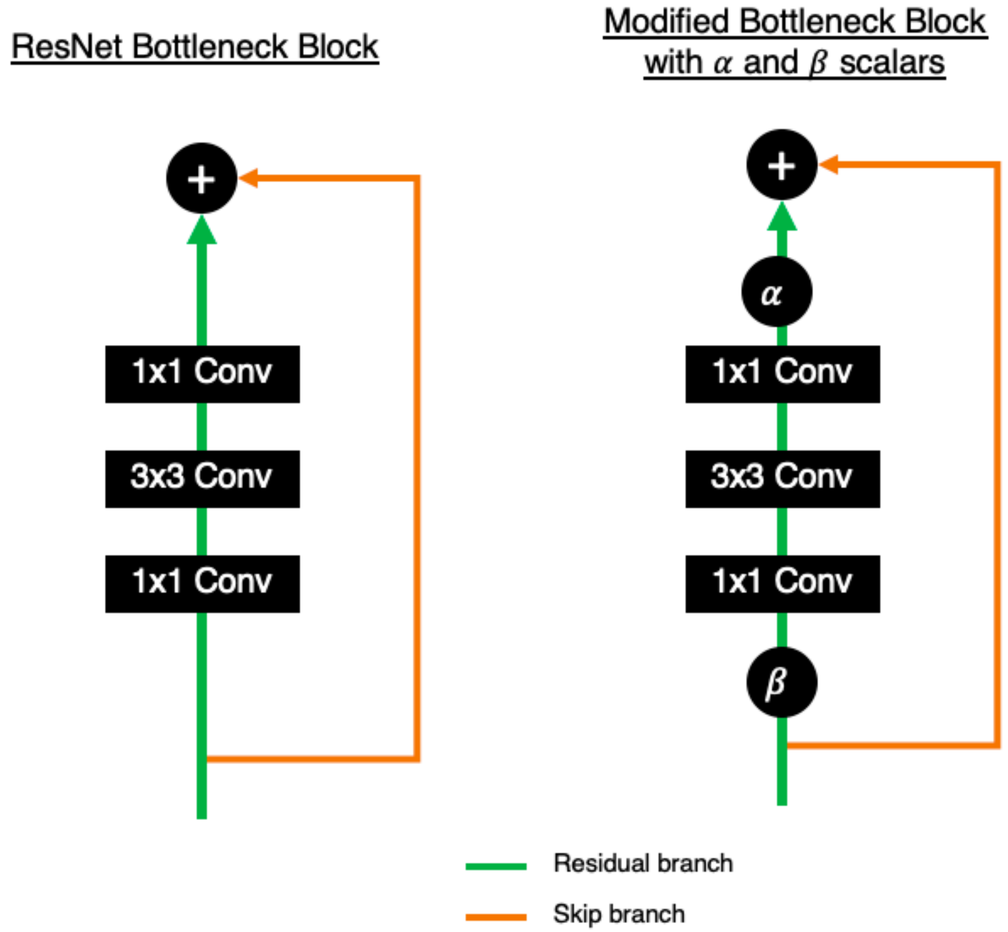
Normalizer-Free ResNet networks

новая SOTA на ImageNet

- Контролируемая остаточная сеть
- Scaled Weight Standartizarion
- Adaptive gradient clipping

Контролируемая остаточная сеть

- Вместо $x_{n+1} = x_n + f_n\left(x_n\right)$ возвращаем $x_n + \alpha \cdot f_n\left(x_n / \beta_n\right)$
- $\beta_n = \sqrt{\text{Var}\left(x_n\right)}$, $\text{Var}\left(x_0\right) = 1$
- Хотим $\text{Var}\left(f_n(x)\right) = \text{Var}\left(x\right)$
- Тогда $\text{Var}\left(x_{n+1}\right) = 1 + \alpha^2$
- Полезно иногда сбрасывать разброс и шкалировать для всех на выходе из блока
- Контролируем рост разброса в блоке
- Как получить блок с таким свойством?



Scaled Weight Standartizarion

- Перешкалируем веса слоев по сверткам: $\widehat{W}_{ij} = \gamma \frac{W_{ij} - \mu_{W_i}}{\sigma_{W_i}\sqrt{N}}$
- Между слоями применяем активацию $g(x)$ с дисперсией σ_g^2 и средним μ_g для $N(0,1)$
- Теперь после свертки (и между ними):

$$y = f(g(x)) = Wg(x)$$

$$y_i = \sum_{j=1}^{N} W_{ij} g(x_j)$$

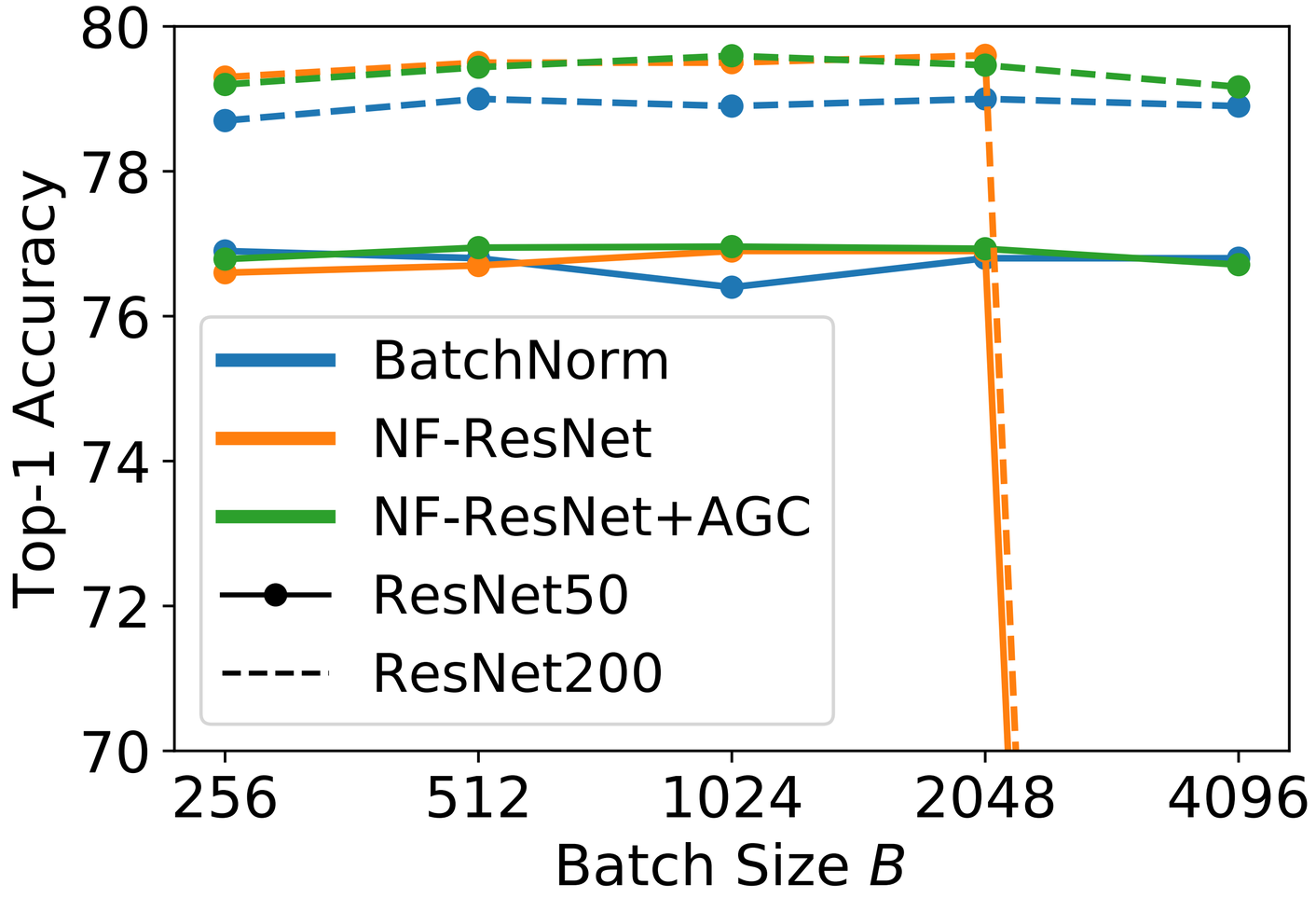
$$\mathbb{E} y_i = N\mu_g \mu_{W_i} \qquad \text{Var}(y_i) = N\sigma_g^2(\sigma_{W_i}^2 + \mu_{W_i}^2)$$

- Подставив перешкалированные веса, получим, что наша хотелка выполнилась
- γ – параметр, который борется с распределением активации. $\gamma = \frac{1}{\sigma_g}$

Adaptive gradient clipping

- Следующим образом изменяем градиент для i-той компоненты в ℓ-том блоке
- Ограничиваем шаг, если свертка меняется слишком сильно

$$G_i^\ell \to \begin{cases} \lambda \dfrac{\|W_i^\ell\|_F^\star}{\|G_i^\ell\|_F} G_i^\ell & \text{if } \dfrac{\|G_i^\ell\|_F}{\|W_i^\ell\|_F^\star} > \lambda, \\ G_i^\ell & \text{otherwise.} \end{cases}$$



- Также модель сравнима по скорости с BN
- На ImageNet удалось достичь 86.5% точности, что является SOTA для моделей без дополнительных данных

	224px	320px	384px
BN-ResNet-50	78.1	79.6	79.9
NF-ResNet-50	**79.5**	**80.9**	**81.1**
BN-ResNet-101	80.8	82.2	82.5
NF-ResNet-101	**81.4**	**82.7**	**83.2**
BN-ResNet-152	81.8	83.1	83.4
NF-ResNet-152	**82.7**	**83.6**	**84.0**
BN-ResNet-200	81.8	83.1	83.5
NF-ResNet-200	**82.9**	**84.1**	**84.3**

Ссылки

1. https://arxiv.org/pdf/2102.06171.pdf
2. https://paperswithcode.com/method/weight-standardization
3. https://github.com/joe-siyuan-qiao/WeightStandardization
4. https://www.kaggle.com/residentmario/batch-normalization-and-its-successors
5. https://arxiv.org/pdf/1803.08494.pdf
6. https://towardsdatascience.com/nfnets-explained-deepminds-new-state-of-the-art-image-classifier-10430c8599ee
7. https://arxiv.org/pdf/2101.08692.pdf