Theoretical and empirical investigation of several common practices in deep learning

Daniel Soudry

Electrical Engineering
Why “overfit”? 

Daniel Soudry, Elad Hoffer, Nati Srebro
*The Implicit Bias of Gradient Descent on Separable Data* (ICLR 2018)

Suriya Gunasekar, Jason Lee, Daniel Soudry, and Nati Srebro.
*Characterizing implicit bias in terms of optimization geometry* (ICML 2018).

Mor Shpigel-Nacson, Jason Lee, Suriya Gunasekar, Nathan Srebro, Daniel Soudry
*Convergence of Gradient Descent on Separable Data* (ArXiv, 2018)

Suriya Gunasekar, Jason D. Lee, Daniel Soudry, Nathan Srebro
*Implicit Bias of Gradient Descent on Linear Convolutional Networks*

Mor Shpigel Nacson, Nathan Srebro, Daniel Soudry
*Stochastic Gradient Descent on Separable Data: Exact Convergence with a Fixed Learning Rate* (ArXiv, 2018)
"Overfitting" is good for generalization?

Dataset: CIFAR10, Architecture: Resnet44, Training: SGD + momentum + gradient clipping
Gradient descent on logistic loss: $\Delta w = -\eta \nabla L(w)$

**Theorem 1:** $w(t) = \hat{w} \log t + \rho(t)$, 
$\hat{w}$ is the (L2) max margin vector 
and $\|\rho(t)\| = O(\log \log t)$

Therefore: $\frac{w(t)}{\|w(t)\|} \rightarrow \frac{\hat{w}}{\|\hat{w}\|}$

Also holds for:
1) Stochastic gradient descent, **with fixed learning rate**
2) Multiclass
Logarithmically slow convergence to max-margin

\[ \mathbf{w}(t) = \hat{\mathbf{w}} \log t + \rho(t) \]
... While test loss increases

\[ w(t) = \hat{w} \log t + \rho(t) \]

Test loss \[ = \Omega(\log(t)) \]
“Overfitting” Explained?

Dataset: CIFAR10, Architecture: Resnet44, Training: SGD + momentum + gradient clipping
Why Adam has worse generalization?

• Adaptive rate methods (e.g. AdaGrad, Adam)
  → Solution can depend on initial conditions
Why Exponential tail?

• Similar results for exp-tailed losses

• Other loss functions?

→ Loss with exponential tail has optimal rate

→ Power low tail (or heavier), no longer converges to max margin
Improve convergence speed?

- GD has slow converge rate $\frac{1}{\log t}$
- Normalized GD:

$$\Delta w = - \frac{1}{\sqrt{t}} \frac{\nabla L(w)}{\|\nabla L(w)\|}$$

$\Rightarrow$ Convergence rate improves to $\frac{\log^2 t}{\sqrt{t}}$.

Potentially improves for deep networks:
Batch-norm and regularization

“Norm matters: efficient and accurate normalization schemes in deep networks”
Elad Hoffer, Ron Banner, Itay Golan, Daniel Soudry (Arxiv 2018)

“Scalable Methods for 8-bit Training of Neural Networks”
Ron Banner, Itay Hubara, Elad Hoffer, Daniel Soudry (Arxiv 2018)
Batch normalization

- Batch-norm (Ioffe, 15’) widely used:

  $$BN(x) = \frac{x - \langle x \rangle}{\sqrt{\frac{1}{n} \sum_{i=1}^{n} (x_i - \langle x \rangle)^2}}$$

- Shortcomings:
  - Assumes independence between samples (problem when modeling time-series, RL, GANs, metric-learning etc.)
  - Requires high-precision operations ($\sqrt{\sum x^2}$), numerically unstable.
  - Significant computational and memory impact, with data-bound operations – makes up to 25% of computation time in current models (Gitman, 17’)
  - Why it works? Interaction with other regularization
Batch-norm Leads to norm invariance

The key observation:

• Given input $x$, weight vector $w$, its direction $\hat{w} = \frac{w}{\|w\|}$

• Batch-norm is norm invariant: $BN(\|w\|\hat{w}x) = BN(\hat{w}x)$

• Weight norm only affects effective learning rate, e.g. in SGD:

\[
\Delta \hat{w} = \frac{\eta}{\|w\|^2} (I - \hat{w}\hat{w}^\top) \nabla L(\hat{w}) + O(\eta^2)
\]
Weight decay before BN is redundant

- Weight-decay equivalent to learning-rate scaling
Replacing Batch-norm – switching norms

• Batch-normalization – just scaled $L^2$ normalization
• More numerically stable norms:

$$
\|x\|_1 = \sum_i |x_i| \quad \|x\|_\infty = \max_i \{|x_i|\}
$$

We use additional scaling constants so that the norm will behave similarly to $L^2$, by assuming that neural input is Gaussian, e.g.:

$$
\frac{1}{\sqrt{n}} E \|x - Ex\|_2 = \sqrt{\frac{\pi}{2}} \cdot \frac{1}{n} E \|x - Ex\|_1
$$
$L^1$ Batch-norm (Imagenet, Resnet)
Low precision batch-norm

Using $L^1$ batch-norm alleviates some of the low-precision difficulties of batch-norm. We can now train ResNet50 without issues on FP16:
With a few more tricks...

- Can now train ResNet ImageNet with bottleneck operations in int8:
Thank you for your time! Questions?