Neural Networks and Deep Learning: Advanced Optimization

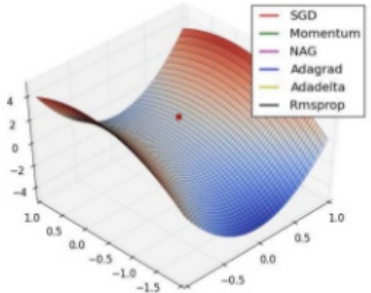
Nicolas Thome

Conservatoire National des Arts et Métiers (Cnam)
Département Informatique

Learning Rate Adaptation

Motivation:

- **Adapting updates to each individual parameter** w_i
  $\Rightarrow$ larger/smaller updates depending on the landscape cost function
- **Context:** sparse data, scaling parameter variation in deep networks (*cf* batch norm)
- Family of algorithms with adaptive learning rates:
  - AdaGrad
  - AdaDelta
  - RMSProp
  - Adam



Adaptive Gradient (Adagrad) [Duchi et al., 2011]

- Adaptative update rule for each parameter of the form: $\boxed{w_i^{t+1} = w_i^t - \Delta_i^{t+1}}$
- **Definition:** $g_i^t = \frac{\partial f}{\partial w_i}(w_i^t)$, *i.e.* gradient dimension
- $\mathbf{G_i^t} = \sum_{\mathbf{i=1}}^{\mathbf{t}} (\mathbf{g_i^t})^\mathbf{2} \Rightarrow$ memory of squared gradients over time
  - $\sqrt{G_i^t}$: ℓ_2 norm of the gradient for $t = \{1; t\}$
  - $G_i^t \leftrightarrow E[g_i^2] : 2^{nd}$ moment, uncentered variance
- **Adagrad** ($\varepsilon \sim 10^{-8}$, $\eta \sim 10^{-2}$):

$$\Delta_i^{t+1} = \frac{\eta}{\sqrt{G_i^t + \varepsilon}} g_i^t = \eta' g_i^t$$

  - **Intuition:** large $G_i^t \Rightarrow \downarrow \eta'$, small $G_i^t \Rightarrow \uparrow \eta'$

$\Rightarrow$ **Supporting very predictive but comparatively rare features**

Adaptive Gradient (Adagrad) [Duchi et al., 2011]

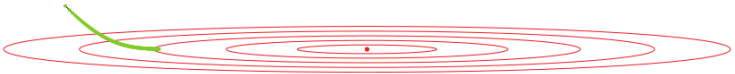
- Vectorial form: $\mathbf{w}^{t+1} = \mathbf{w}^t - \Delta^{t+1}$

$$\Delta^{t+1} = \frac{\eta}{\sqrt{G^t + \varepsilon}} \odot g^t$$

- **SHORTCOMING:** very aggressive learning rate decay
- $(g_i^t)^2 \geq 0 \Rightarrow G_i^t$ monotonically increasing

$$\Rightarrow \eta' = \frac{\eta}{\sqrt{G_i^t + \varepsilon}} \to 0$$

- $\mathbf{w}$ updates stop



Root Mean Square Propagation [Tieleman and Hinton, 2012]

- **RMSProp Idea:** compute average of $(g_i^t)^2$ only using a recent time window:

$$\tilde{G}_i^t = \rho G_i^{t-1} + (1-\rho)(g_i^t)^2 \qquad \rho \sim 0.9$$

$$\Delta_i^{t+1} = \frac{\eta}{\sqrt{\tilde{G}_i^t + \varepsilon}} g_i^t = \eta' g_i^t$$

- **RMSProp:** Same update rule as Adagrad ($\tilde{G}_i^t \leftrightarrow G_i^t$)
  $\Rightarrow$ $\tilde{G}_i^t$ **NOT monotonically increasing**
  $\Rightarrow$ Lessen aggressive learning rate decay
- Final parameter update:

$$w_i^{t+1} = w_i^t - \Delta_i^{t+1}$$

AdaDelta (AdaDelta) [Zeiler, 2012]

- **Essentially: AdaDelta = RMSProp + momentum**
- Compute g_i^t and $\tilde{G}_i^t \sim$ RMSProp: local average of previous $(g_i^t)^2$

$$\tilde{G}_i^t = \rho G_i^{t-1} + (1-\rho)(g_i^t)^2 \qquad \rho \sim 0.9$$

- **New AdaDelta update vector:** $\mathbf{u_i^{t+1}} = \frac{\sqrt{\mathbf{U_i^t}+\varepsilon}}{\sqrt{\tilde{\mathbf{G}}_\mathbf{i}^\mathbf{t}+\varepsilon}}\mathbf{g_i^t} = \eta' \mathbf{g_i^t}$
- **New term U_i^t in numerator compared to RMSProp:**
  - $\sim$ Momentum (acceleration term) accumulating prior updates

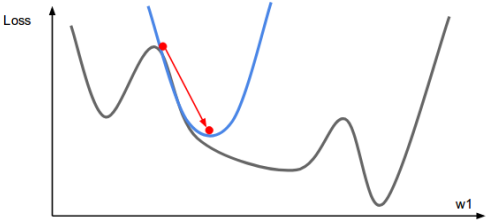
$$U_i^{t+1} = \rho U_i^t + (1-\rho)(u_i^t)^2$$

- Final parameter update:

$$w_i^{t+1} = w_i^t - u_i^{t+1}$$

AdaDelta (AdaDelta) [Zeiler, 2012]

$$u_i^{t+1} = \frac{\sqrt{U_i^t + \varepsilon}}{\sqrt{\tilde{G}_i^t + \varepsilon}}\, g_i^t = \frac{RMS(updates)}{RMS(gradients)}\, g_i^t$$



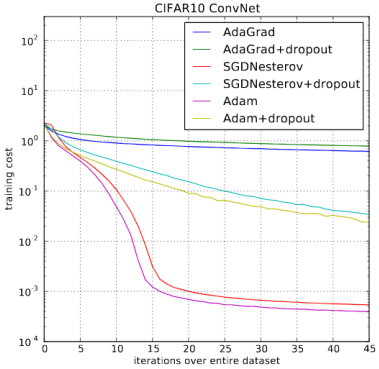
- Approximate second order (Hessian H diagonal)
  $\mathbf{w}^{t+1} = \mathbf{w}^t - \mathbf{H}^{-1}\mathbf{g} \Rightarrow w_i^{t+1} \propto w_i^t - \frac{f'}{f''}$
  - Approximate $\frac{1}{f''}$ by $\frac{RMS(updates)}{RMS(gradients)}$
  - Homogeneous dimension update: $u_i^{t+1} \sim \mathbf{w}$!
    $\neq$ SGD, momentum, RMSProp!
- Even no learning rate!
  - $\oplus$ no parameter
  - $\ominus$ no way to design variable update speeds (*e.g.* fine-tuning)

Adaptive Moment Estimation (Adam) [Kingma and Ba, 2015]

- Adam: 1^{st} and 2^{st} gradient moment estimation
- **Strong similarities to Adadelta: 2^{nd} gradient moment + momentum**
  - 2^{nd} gradient moment ~ Adadelta/RMSProp with local accumulation:
    $$v_i^t = \beta_2 v_i^{t-1} + (1-\beta_2)(g_i^t)^2$$
  - 1^{st} **gradient moment (mean)** ***vs*** **squared activation Adadelta**
    $$m_i^t = \beta_1 m_i^{t-1} + (1-\beta_1)(g_i^t)$$
  - **Overcome Adadelta limitations on biased moment estimate (init)**
    $$\hat{m}_i^t = \frac{m_i^t}{1-\beta_1} \qquad \hat{v}_i^t = \frac{v_i^t}{1-\beta_2}$$
  - Final update: $w_i^{t+1} = w_i^t - \eta \frac{\hat{m}_i^t}{\sqrt{\hat{v}_i^t + \varepsilon}}$

Learning Rate Adaptation: Conclusion

- Per-dimension update: important in case of sparse data
- Adagrad, RMSProp, Adadelta and Adam: all use 2^{nd} gradient moment (var) on denominator
- Adadelta, Adam: use $\sim$ momentum on numerator
- Adam good default choice in many cases
- **Implementation Issues?**
  $\Rightarrow$ **following!**



References I

Duchi, J., Hazan, E., and Singer, Y. (2011).
Adaptive subgradient methods for online learning and stochastic optimization.
J. Mach. Learn. Res., 12:2121–2159.

Kingma, D. P. and Ba, J. (2015).
Adam: A method for stochastic optimization.
In ICLR, volume abs/1412.6980.

Tieleman, T. and Hinton, G. (2012).
RMSprop Gradient Optimization.

Zeiler, M. D. (2012).
ADADELTA: an adaptive learning rate method.
CoRR, abs/1212.5701.