Artificial Neural Networks Deep Learning Comparisons +1

What are differences between update rules like AdaDelta, RMSProp, AdaGrad, and AdaM?

This question previously had details. They are now in a comment.

3 Answers

Rajarshee Mitra, Machine Learning @ Microsoft
Answered Feb 28, 2016 · Upvoted by Ramon Viñas, MSc Machine Learning, University College London (2018)

I will try to give a not-so-detailed but very straightforward answer. My assumption is that you already know how Stochastic Gradient Descent works.

Overview : The main difference is actually how they treat the learning rate.

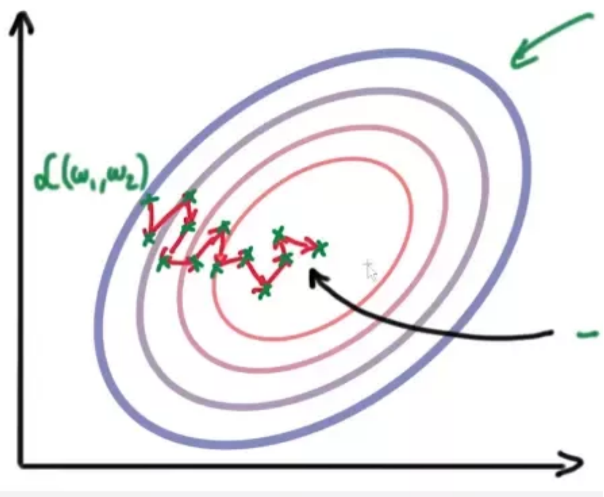
Stochastic Gradient Descent:

$$\theta_{t+1} = \theta_t - \alpha \delta L(\theta_t)$$

Theta (weights) is getting changed according to the gradient of the loss with respect to theta.

alpha is the learning rate. If it is very small, convergence will be very slow. On the other hand, large alpha will lead to divergence.

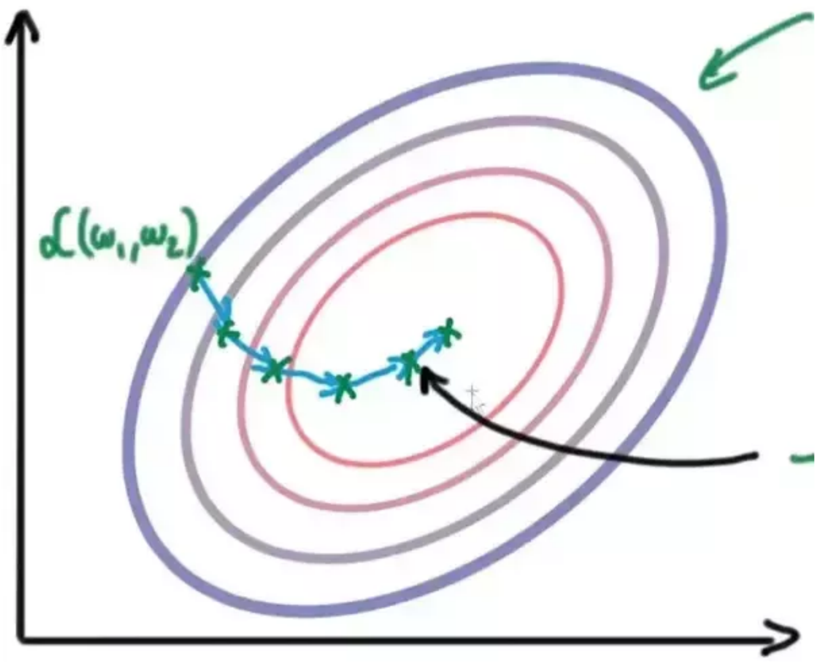
Now, the gradient of the loss (L) changes quickly after each iteration due to the diversity of each training example. Have a look at the convergence below. We are taking small steps but they are quite zig-zag (even though we slowly reach to a loss minima).



To overcome this, we introduce momentum. Basically taking knowledge from previous steps about where we should be heading. We are introducing a new hyperparameter μ

$$v_{t+1} = \mu v_t - \alpha \delta L(\theta_t)$$

We will use the concept of momentum again later. (Don't confuse it with moment
which is also use



This is the image of SGD equipped with momentum.

Adagrad:

Adagrad scales alpha for each parameter according to the history of gradients (previous steps) for that parameter which is basically done by dividing current gradient in update rule by the sum of previous gradients. As a result, what happens is that when the gradient is very large, alpha is reduced and vice-versa.

$$g_{t+1} = g_t + \delta L(\theta_t)^2$$

$$\theta_{t+1} = \theta_t - \frac{\alpha \delta L(\theta)^2}{\sqrt{g_{t+1}} + \epsilon}$$

RMSProp:

The only difference RMSProp has with Adagrad is that the g_t term is calculated by exponentially decaying average and not the sum of gradients.

$$g_{t+1} = \gamma g_t + (1 - \gamma)\delta L(\theta)^2$$

Here g_t is called the **second order moment** of δL . Additionally, a first order moment m_t can also be introduced.

$$m_{t+1} = \gamma m_t + (1 - \gamma)\delta L(\theta)$$

$$g_{t+1} = \gamma g_t + (1 - \gamma)\delta L(\theta)^2$$

Adding momentum as in the first case,

$$v_{t+1} = \mu v_t - \frac{\alpha \delta L(\theta)}{\sqrt{g_{t+1} - m_{t+1}^2 + \epsilon}}$$

And finally collecting new theta as we have done in the first example,

$$\theta_{t+1} = \theta_t + v_{t+1}$$

AdaDelta:

AdaDelta also uses exponentially decaying average of g_t which was our 2nd moment of gradient. But without using alpha that we were traditionally using as learning rate, it introduces x_t which is the 2nd moment of v_t.

$$g_{t+1} = \gamma g_t + (1 - \gamma)\nabla \mathcal{L}(\theta)^2$$

$$v_{t+1} = -\frac{\cdots}{\sqrt{g_{t+}}}$$

$$\theta_{t+1} = \theta_t + v_{t+1}$$

Adam:

It uses both first order moment m_t and 2nd order moment g_t but they are both decayed over time. Step size is approximately $\pm\alpha$. Step size will decrease, as it approaches minimum.

$$m_{t+1} = \gamma_1 m_t + (1 - \gamma_1)\nabla\mathcal{L}(\theta_t)$$

$$g_{t+1} = \gamma_2 g_t + (1 - \gamma_2)\nabla\mathcal{L}(\theta_t)^2$$

$$\hat{m}_{t+1} = \frac{m_{t+1}}{1 - \gamma_1^{t+1}}$$

$$\hat{g}_{t+1} = \frac{g_{t+1}}{1 - \gamma_2^{t+1}}$$

$$\theta_{t+1} = \theta_t - \frac{\alpha\hat{m}_{t+1}}{\sqrt{\hat{g}_{t+1}} + \epsilon}$$

Chandrakant Khandelwal, Machine Learning enthusiast
Answered Nov 16, 2015

I don't think there is any single rule of thumb for selecting an update rule, go through the following link for a detailed explanation of different update rules.
CS231n Convolutional Neural Networks for Visual Recognition
There are also some graphs in it which would help you to understand the concept better.
I would also suggest to go through this lecture: Page on toronto.edu
It talks in general about optimization methods for neural nets and their pros and cons.

Matthew Lai, Research Engineer @ Google DeepMind
Answered Aug 24, 2015

AdaDelta is a slight improvement over AdaGrad that fixes a few things. See the AdaDelta paper for more details.

RMSProp is a new thing that tries to adapt resilient prop (rprop), which only works for batch training, to stochastic gradient descent.

I'm not too familiar with Adam, but it seems to be similar in concept to AdaDelta according to the paper (adapting AdaGrad for problems with non-stationary objective). Curiously, they didn't compare it to AdaDelta, even though AdaDelta has been around for quite a while by the time they published Adam.

Add a com

Please refresh this page to receive new updates.

1 Answer Collapsed (Why?)

Top Stories from Your Feed