Training algorithm

Created	@October 15, 2021 8:46 AM
Materials	
Type	

α : learning rate

μ : momentum (thường chon $\mu = 0.9$)

Vanilla SGD

Sử dụng mini-batch cho mỗi lần train, update bình thường

$$W -= \alpha \times dW$$

SGD - Momentum

Thay đổi một chút cách update, ý tưởng dựa trên vật lý

$$v = \mu \times v - \alpha \times dW$$
$$W += v$$

SGD - Nesterov Momentum

Tương tự như momentum nhưng thay vì update v theo dW thì update v theo $d(W + \mu \times v)$

$$v_{pred} = v$$
$$v = \mu \times v - \alpha \times dW$$
$$W += -\mu \times v_{pred} + (1 + \mu) \times v$$

Adagrad

Ý tưởng là làm cho những trọng số có gradient cao thì sẽ có learning rate nhỏ lại, còn những trọng số có gradient thấp thì learning rate sẽ cao. Làm như vậy để giảm việc update quá nhanh theo 1 hướng và quá chậm theo 1 hướng khác.

Note: ϵ nhỏ cỡ $1e-7$ để không bị chia cho 0.

$$cache += (dW)^2$$
$$W += -\alpha \times dW / (\sqrt{cache} + \epsilon)$$

Vấn đề với Adagrad là cache sẽ cứ tằng dần và đến một lúc nào đấy sẽ khiến cho model không update được nữa.

RMSprop

Để giả quyết vấn đề của Aragrad, thay vì cứ để $cache$ tiếp tục tăng mạnh thì ta sẽ làm nó tăng chậm lại một chút.

$$cache = decay \times cache + (1 - decay) \times (dW)^2$$
$$W = W - \alpha \times dW / (\sqrt{cache} + \epsilon)$$

$decay$ thường chọn là $0.9, 0.99$