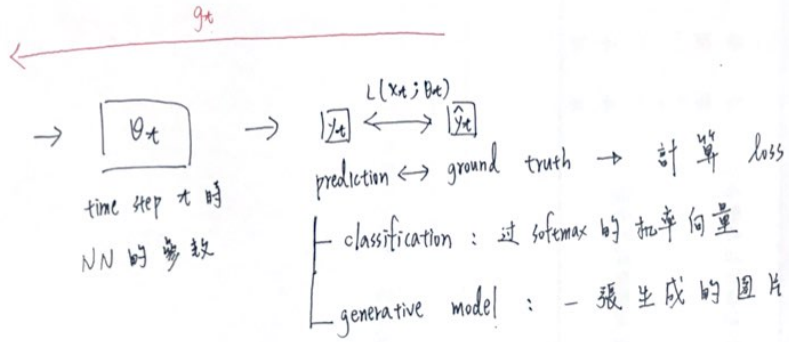
θ_t : model parameters → 此訓練想 optimize 的問題.

$\nabla L(\theta_t)$ / g_t : 計算某組 θ 在特定時候 loss 的 gradient → 依 θ_t optimize 得 θ_{t+1} 的 constant

m_{t+1} : momentum , 記錄之前的 gradient → $\theta_t \to \theta_{t+1}$ 的 gradient (会压縮，不会全概括)



Optimization

找一个 θ 使 trainning data 的 x 算出的 loss 最小

→ parameters 越貼近訓練资料越好，使 $y_n \approx \hat{y}_n$

On-line : 每个 time step 只有一个 x_t

Off-line : 每个 time step 可得的有的 x, y, ground truth

容易 → get all trainning data

※ 要這麼大的 memory

→ 一次只得一个 x

SGD ： 於初始的 θ_0 開始，計算 gradient 後往反方向移動

→ ∵ gradient 為 L 增加的方向，∴ 往反向才可 find 最小 L 的 θ　　$\theta_t = \theta_{t-1} - \eta g_{t-1}$

SGD with Momentum ： Movement $v^0 = 0$

(SGDM)　update → Movement　$v^1 = \lambda v^0 - \eta \nabla L(\theta^0)$　→ 如慣性，入市慣性影响力成正比

lambda　Eta

SGD 算出來要 update 的方向

v^i → 做時間的 weighted sum，过去所有 gradient 的總和

Adagrad ： 防止一開始 gradient 過大，導致移動過多而走向更差的位置

$$\theta_t = \theta_{t-1} - \frac{\eta}{\sqrt{\sum_{i=0}^{t-1} (g_i)^2}} g_{t-1}$$

time step gradient 的和，gradient 大 → 移动小，避免跳過最小

小 → 可放心移动

gradient 持續加總，若一開始 gradient 太大，learning rate 過小，可能走沒幾步便卡住

RMSProp ： 確保 V_t 不斷變化，gradient 避免持續變大，使 Optimizer 不会卡住

(以 Momentum)

$$\theta_t = \theta_{t-1} - \frac{\eta}{\sqrt{V_t}} g_{t-1}$$

$$V_1 = g_0^2$$

$$V_t = \alpha V_{t-1} + (1-\alpha)(g_{t-1})^2$$

v^i → sum of gradient

無法處理 gradient = 0，導致 Optimizer 卡住問題

Adam : SGDM → $\theta_t = \theta_{t-1} - \eta m_t$

$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_{t-1}$

+

RMSProp → $\theta_t = \theta_{t-1} - \frac{\eta}{\sqrt{v_t}} g_{t-1}$

$v_1 = g_0^2$

$v_t = \beta_2 v_{t-1} + (1-\beta_2)(g_{t-1})^2$

$$\theta_t = \theta_{t-1} - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \hat{m}_t$$

$\hat{m}_t$ → 確保 m_t 不會隨時間增加越來越大

∵ $\beta_1 < 1$ ∴ m 一開始較小，隨時間穩定

ϵ → 避免 $t=0$，$v_t = 0$，造成 η 無限大

epsilon

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t} \qquad \beta_1 = 0.9$$

$$\hat{v}_t = \frac{v_t}{1-\beta_2^t} \qquad \beta_2 = 0.999$$

$$\epsilon = 10^{-8}$$