

# Formulation

KwanHee Lee  
Graduation Project  
CSE@Konkuk Univ.  
24.02.21

# Contents

1. Detailed formulation
2. Code Implementation
3. Disuccusion

# Detailed formulation

Net with  $f_\theta(x) = y$ .

$\theta = \{w_1, \dots, w_L\}$   $w_i$  =  $i$ th layer weight matrix.

data  $D \sim (x, y)$   $x$ : input  
 $y$ : label.

Loss function:  $\mathcal{L}_{CE}(f_\theta(x), y)$

Optimizer: SGD:  $\theta_t \leftarrow \theta_t - \eta \nabla_{\theta} \mathcal{L}_{CE}(f_\theta(x), y)$

Adam:  $\theta_{t+1} \leftarrow \theta_t - \eta \frac{m_t}{(\sqrt{v_t} + \epsilon)}$

Adam uses ADAPTIVE LR

$$\begin{cases} m_t \leftarrow \beta_1 m_{t-1} + (1 - \beta_1) g_t \\ v_t \leftarrow \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \\ \hat{m}_t \leftarrow m_t / (1 - \beta_1^t) \\ \hat{v}_t \leftarrow v_t / (1 - \beta_2^t) \end{cases}$$

$$g_t \leftarrow \nabla_{\theta} \mathcal{L}_{CE}(f_\theta(x), y)$$

Formulation 1

Given neural net  $f_\theta$ , data  $D(x, y)$ , loss function  $\mathcal{L}_{CE}$ .

1. Generate adversarial sample  $x'$  where  $\|x - x'\|_2 \leq \epsilon$ . (via PGD attack)

$$x' = x + \delta, \|\delta\|_2 \leq \epsilon,$$

$$\delta = \epsilon \frac{\nabla_{\theta} \mathcal{L}_{CE}(f_\theta(x), y)}{\|\nabla_{\theta} \mathcal{L}_{CE}(f_\theta(x), y)\|_2}$$

2. Calculate Clean & Adv Loss.

$$\mathcal{L} = \mathcal{L}_{CE}(f_\theta(x), y) + \mathcal{L}_{CE}(f_\theta(x'), y).$$

(i) typical AT  $\Rightarrow \mathcal{L} = \mathcal{L}_{CE}(f_\theta(x'), y)$   
[TOWERS  $\Rightarrow \mathcal{L} = \mathcal{L}_{CE}(f_\theta(x), y) + \mathcal{L}_{CE}(f_\theta(x'), y)$ ]

24.02.27.

Kun

Idea:  
\* Based on  
SGD, but with  
adaptation of  $\eta$  &  $\beta$ s.

3. Update  $\theta_{t+1}$  using different optimizer.

$$\theta_{t+1} = \theta_t - \eta \nabla_{\theta} \mathcal{L}_{CE}(f_\theta(x), y)_{SGD}$$

$$- \eta \frac{m_t}{(\sqrt{v_t} + \epsilon)} g_t = \nabla_{\theta} \mathcal{L}_{CE}(f_\theta(x), y)_{ADAM}$$

\* where  $\eta \gg \epsilon$  (learning rate).

\* Use learning rate scheduler for  $\eta$ . (warning with large  $\eta$ !)

Q) Adam can explode?

Q) large learning rate for SGD, small for Adam.

to how  $\eta$ , in which scale?

# Code Implementation

## Optimization process

$$\theta_{t+1} \leftarrow \theta_t - \eta \nabla_{\theta} \mathcal{L}_{ce}(f_{\theta}(x), y) - \gamma \nabla_{\theta} \frac{m}{\sqrt{v_t} + \epsilon} \mathcal{L}_{ce}(f_{\theta}(x_{adv}), y)$$

Implementation using torch.autograd.grad()

```
loss_natural.backward() # maintains loss in computational graph
adv_gradients = torch.autograd.grad(loss_robust, model.parameters()) # gradients w.r.t adv loss
optimizer_sgd.step() # update natural loss

optimizer_adam.zero_grad() # zero gradient
for param, grad in zip(model.parameters(), adv_gradients): # update param.grad with previous gradients w.r.t adv loss
    param.grad = grad
optimizer_adam.step() # update adversarial loss

loss = loss_natural + beta*loss_robust # meaningless
```

- Autograd.grad() -> calculates gradient and return tensor(gradient)
- backward() -> calculates gradient and accumulates it into tensor.grad
- Optimizer.step() -> updates parameter w.r.t tensor.grad using optimizer
- Optimizer.zero\_grad() -> removes tensor.grad w.r.t given parameter

# Discussion

- Need to specify hyperparameters..
- Will it work out?