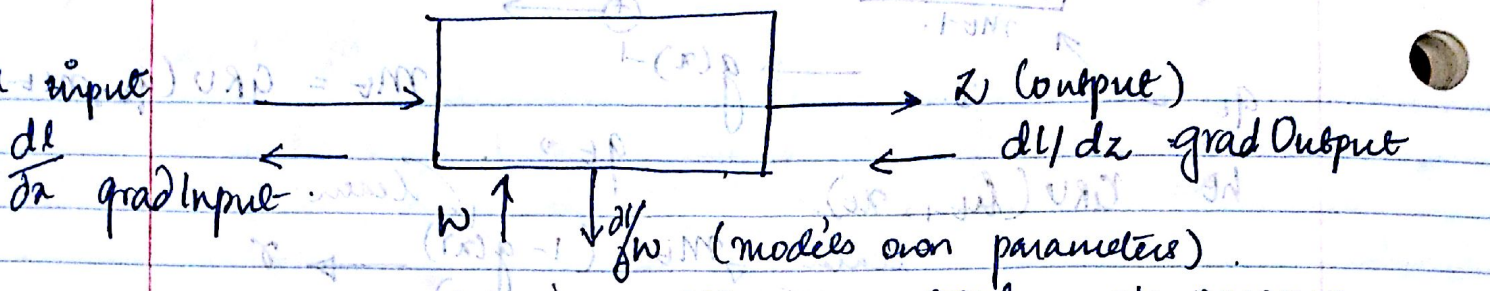
TORCH : understanding backward & forward pass.



z = model : forward (x). → calls updateOutput (x) internally.

$\frac{dl}{dx}$ = model : backward (x, $\partial l/\partial z$)

↳ updateGradInput (x, $\partial l/\partial z$) → the $\partial l/\partial x$.

↳ accGradParam = computes gradient w.r.t model's own params (w) "trainable weights"

$\frac{dl}{dw}$.

you can zero this acc zeroGradParams()

you can update $w^* = w - lr * grad\,w.r.t\ params$

$w^* = w - lr \times \partial l/\partial w$.

w, dl_dw = model : getParams() → learnable params.

feval (w_new).

if w ~= w_new // copy weights new

w : copy (w_new)

end

dl_dw : zero OR. 0 the $\partial l/\partial w$.

zeroGradParams()

x ← inp

y ← op new batch.

prediction = model : forward (x).

loss = criterion : forward (pred)

dl_dz = ~~model~~ criteria : backward ( )

model : backward → gives $\partial l/\partial x$, $\partial l/\partial w$

Returns loss, $\partial l/\partial w$