Finding the gaps in ML/DL

$|\min 0\text{-}1 \text{ loss}| \leftarrow$ discrete function

intract optimisation

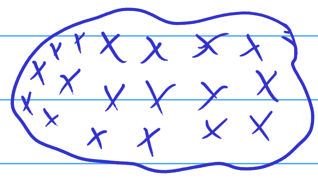
0-1 (not even integer) optimization problem

very difficult to solve

So they want the surrogate optimisation problems!!

$$\left[E_{(x,y)}\left(\mathcal{L}(f(x;\theta),y)\right)\right]$$

$(x,y) \sim P(data)$



☆ Lets assume that the dataset, and testing one coming from the same probability distribution

☆

→ We are never minimizing expectation before

$$\iint (p(x,y)) L(f(x;\theta),y)\, dx\, dy$$

→ We are never doing this before (we're never talking about gradient descent)

Instead, we are doing

$$\min \frac{1}{n} \sum_{i=1}^{n} L(f(x_i, \theta), y)$$

(not a KL divergence no joint distribution)

$$\min_{(x,y)\sim \hat{P}data} E(L(f(x_i, \theta), y))$$

$$\frac{\partial E}{\partial \theta_k} = \frac{1}{n} \sum_{i=1}^{n} \frac{\partial E_i}{\partial \theta_k}$$ ← $n = 10{,}00{,}00{,}000$ Too much

⇒ SGD / minibatch

{ we estimate $\frac{\partial E}{\partial \theta_k}$ by $\frac{\partial E_{100}}{\partial \theta_k}$ }

or

estimate $\frac{\partial}{\partial \theta}(E_2 + E_5 + \cdots + E_{75} + E_{2000})$

minibatch gradient descent

batch gradient descent typically means classical

¿¿ bigger batch size, better estimate?? (ig yes)

minibatch, stochastic, (online)

uses "most recent" stuff (samples) in timestamp or sth for example