# Deep learning theory.

Prereqs:

- Basic ML
- Proof-based linear algebra
- Probability
- Python programming.

All course info will be on my website.
You will turn in HW & projects on Canvas.

# Lecture 1: Approximation, convexity, ...

Basic, 2-layer shallow network: $x \in \mathbb{R}^d$

$$\mathbb{R}^d \ni x \longmapsto \sum_{j=1}^{m} a_j \varphi(\langle w_j, x \rangle + b_j) \in \mathbb{R}.$$

- $m$ neurons; width of network is $m$.
- $w_j \in \mathbb{R}^d$ = first layer weights, $b_j \in \mathbb{R}$ = bias for $j^{\text{th}}$ neuron
- $a_j \in \mathbb{R}$ = second layer weights.
- $\varphi : \mathbb{R} \to \mathbb{R}$ is activation function, eg $\max(0,x)$, $\exp(x)$, ....
- $\{a_j, w_j, b_j\}_{j=1}^{m}$ are trainable parameters.

  Sometimes we will freeze the $\{a_j\}$ at initialization, or assume $b_j = 0$, ....

$x \mapsto \sum_{j=1}^{m} a_j \varphi(\langle w_j, x\rangle + b_j)$.

$=: f(x; a, W, b)$ where $W \in \mathbb{R}^{m \times d}$ has rows $w_j$, $a \in \mathbb{R}^m$, $b \in \mathbb{R}^m$.

Sometimes will concatenate all params into single vector $\theta$.

Also sometimes call first layer <u>hidden activations</u> $\varphi(Wx+b)$, where $\varphi$ is applied component-wise ($[\varphi(Wx+b)]_j = \varphi(\langle w_j, x\rangle + b_j)$).

Then $f(x;\theta) = a^T \varphi(Wx+b)$.

<u>Deep network</u>: $\theta = (W_1, b_1, \dots, W_L, b_L)$,

$f(x;\theta) = \varphi_L(W_L \varphi_{L-1}(\dots W_2 \varphi_1(W_1 x + b_1) + b_2 \dots) + b_L)$

$\varphi_i$ are (possibly different) activation functions.

We will mostly deal with two-layer nets.

Def A class of functions $\mathcal{F} := \{ f : [0,1]^d \to \mathbb{R} \}$ a universal approximator if, for every continuous fcn $g : [0,1]^d \to \mathbb{R}$, and $\forall \varepsilon > 0$, $\exists f \in \mathcal{F}$ st

$$\sup_{x \in [0,1]^d} |f(x) - g(x)| = \|f - g\|_\infty < \varepsilon.$$

- Can generalize from $[0,1]^d$ to compact sets in topological space.

---

Class of functions we will consider:

$$\mathcal{F}_{\varphi,m,d} := \{ x \mapsto \sum_{j=1}^m a_j \varphi(\langle w_j, x \rangle + b_j) : a_j, b_j \in \mathbb{R}, w_j \in \mathbb{R}^d \}.$$

$$\mathcal{F}_{\varphi,d} := \bigcup_{m \geq 0} \underbrace{\mathcal{F}_{\varphi,m,d}}_{\text{width } m \text{ 2 layer nets}}$$

2-layer nets.

**Theorem** (Leshno et al '93, informal)

$\mathcal{F}_{\varphi,d}$ is a universal approximator $\iff$ $\varphi$ is not a polynomial.

- Two layer nets are universal appx typically.
- Result is not quantitative: no answer for how wide a neural net must be to appx a cts fcn; just that wide enough suffices.
- No claim that you can find this net w/ an algorithm.

We'll prove a simplified version of this thm for $\varphi = \exp$.

**Theorem** (Stone-Weierstrass; see Folland, Thm 4.45):
Suppose $\mathcal{F}$ is st: ① every $f \in \mathcal{F}$ is continuous; ② $\forall x$, $\exists f \in \mathcal{F}$ st $f(x) \neq 0$; ③ $\forall x \neq x'$, $\exists f \in \mathcal{F}$ st $f(x) \neq f(x')$; ④ $\mathcal{F}$ is closed under mult. & vector space ops. Then $\mathcal{F}$ is universal appx.

- "Closed under multiplication & vector space ops" means:
  - $f, g \in \mathcal{F} \Rightarrow f \cdot g \in \mathcal{F}$
  - $\alpha, \beta \in \mathbb{R},\ f, g \in \mathcal{F} \Rightarrow \alpha f + \beta g \in \mathcal{F}$
- Again, no quantitative bounds here, just existence.

<u>Theorem</u> $\mathcal{F}_{\exp, d}$ is universal.

<u>Pf.</u> ① Clearly every $f \in \mathcal{F}_{\exp, d}$ is continuous: linear combo of continuous functions.

② $\forall x,\ \exp(0^T x) = 1 \neq 0$.

③ For $x \neq x'$, let $f(z) := \exp\left(\frac{\langle z - x', x - x' \rangle}{\|x - x'\|^2}\right)$, then

$$f(x) = \exp(1), \quad f(x') = \exp(0).$$

④ Clear that $\mathcal{F}$ is closed under vector space ops. For products,

$$\left(\sum_{j=1}^{m} a_j \varphi(\langle w_j, x \rangle + b_j)\right) \cdot \left(\sum_{j=1}^{m'} a_j' \varphi(\langle w_j', x \rangle + b_j')\right) =$$

$$= \left(\sum_{j=1}^{m} a_j \exp(\langle w_j, x\rangle + b_j)\right)\left(\sum_{i=1}^{m'} a_i' \exp(\langle w_i', x\rangle + b_i')\right)$$

$$= \sum_{j=1}^{m} \sum_{i=1}^{m'} a_j a_i' \exp(\langle w_j, x\rangle + b_j) \exp(\langle w_i', x\rangle + b_i')$$

$$= \sum_{j=1}^{m} \sum_{i=1}^{m'} a_j a_i' \exp(\langle w_j + w_i', x\rangle + (b_j + b_i')).$$

$\boxtimes$

We won't discuss approximation results all that much.

Some research directions on approximation:

- Exact quantitative bounds on how wide (# neurons per layer) and deep (# layers) a network must be to approximate continuous functions, Sobolev space functions, ...
- Width/depth tradeoffs: $\exists$ functions which require $\exp(d)$ width to appx. using 2-layer nets but $\mathrm{poly}(d)$ width with $O(1)$ layers [Telgarsky '16; Safran-Shamir '17]