Lecture 14: Neural Networks

Nisha Chandramoorthy

October 12, 2023

Last time

- A hypothesis class restricted to some C *shatters* C if any binary function on C is in the class

Last time

- A hypothesis class restricted to some C *shatters* C if any binary function on C is in the class
- VC dimension, $\mathrm{VCdim}(\mathcal{H})$: maximal size of $C \subseteq \mathcal{X}$ that is shattered by $\mathcal{H}$.

Last time

- A hypothesis class restricted to some C *shatters* C if any binary function on C is in the class
- VC dimension, $\mathrm{VCdim}(\mathcal{H})$: maximal size of $C \subseteq \mathcal{X}$ that is shattered by $\mathcal{H}$.
- Eg 1. VCdim of threshold functions on $\mathbb{R}$ is 1.

Last time

- A hypothesis class restricted to some C *shatters* C if any binary function on C is in the class
- VC dimension, $\mathrm{VCdim}(\mathcal{H})$: maximal size of $C \subseteq \mathcal{X}$ that is shattered by $\mathcal{H}$.
- Eg 1. VCdim of threshold functions on $\mathbb{R}$ is 1.
- Eg 2: VCdim of indicator functions on intervals of $\mathbb{R}$ is 2.

Last time

- A hypothesis class restricted to some C *shatters* C if any binary function on C is in the class
- VC dimension, $\mathrm{VCdim}(\mathcal{H})$: maximal size of $C \subseteq \mathcal{X}$ that is shattered by $\mathcal{H}$.
- Eg 1. VCdim of threshold functions on $\mathbb{R}$ is 1.
- Eg 2: VCdim of indicator functions on intervals of $\mathbb{R}$ is 2.
- Eg 3: VCdim of a finite class $\mathcal{H} \leqslant \log_2 |\mathcal{H}|$

Generalization bounds based on VC dimension

- $\mathcal{H} = \{h_\theta(x) = \sin(\theta x) : \theta \in \mathbb{R}\}$.

Generalization bounds based on VC dimension

- $\mathcal{H} = \{h_\theta(x) = \sin(\theta x) : \theta \in \mathbb{R}\}$.
- VCdim is ∞

Generalization bounds based on VC dimension

- $\mathcal{H} = \{h_\theta(x) = \sin(\theta x) : \theta \in \mathbb{R}\}$.
- VCdim is ∞
- Binary classification generalization for 0-1 loss over class $\mathcal{H}$ with VCdim = d: there exist constants $C_1, C_2 > 0$ such that

$$C_1 \frac{d + \log(1/\delta)}{\epsilon^2} \leqslant m_{\mathcal{H}}(\epsilon, \delta) \leqslant C_2 \frac{d + \log(1/\delta)}{\epsilon}$$

(partial) History - trace back from transformers (source:Wikipedia)

- Transformer architecture: 2017, Google Brain [Vaswani et al]
- Deep learning, unsupervised learning 2010s (e.g., GANs 2014)...
- ImageNet: 2009, Fei Fei Li
- Long-short term memory (LSTM) architecture: 1997, [Hochreiter and Schmidhuber]
- Convolutional NNs: (inspired from) 1979 work by [Fukushima]; Recurrent neural networks: 1982 [Hopfield]
- ...
- Automatic Differentiation: 1970 [Linnainmaa]
- ...
- First neural networks: 1950s [Minsky and others]

Fully connected Neural Networks

- Neuron: input $\sum_j w_j h_j$; output $\sigma(\sum_j w_j h_j)$
- Organized into layers of depth l and width n
- Graph: V, E, σ, w; weight function.

Fully connected Neural Networks

- Neuron: input $\sum_j w_j h_j$; output $\sigma(\sum_j w_j h_j)$
- Organized into layers of depth l and width n
- Graph: V, E, σ, w; weight function.
- VC dim of $\mathcal{H}_{V,E,\mathrm{sign}} \leqslant C|E|\log|E|$

Fully connected Neural Networks

- Neuron: input $\sum_j w_j h_j$; output $\sigma(\sum_j w_j h_j)$
- Organized into layers of depth l and width n
- Graph: V, E, σ, w; weight function.
- VC dim of $\mathcal{H}_{V,E,\mathrm{sign}} \leqslant C|E|\log|E|$
  - Proof: Growth function $\tau_{\mathcal{H}}(m) = \max_{C,|C|=m} |\mathcal{H}|_C|$

Fully connected Neural Networks

- Neuron: input $\sum_j w_j h_j$; output $\sigma(\sum_j w_j h_j)$
- Organized into layers of depth l and width n
- Graph: V, E, σ, w; weight function.
- VC dim of $\mathcal{H}_{V,E,\mathrm{sign}} \leqslant C|E|\log|E|$
  - Proof: Growth function $\tau_{\mathcal{H}}(m) = \max_{C,|C|=m} |\mathcal{H}|_C|$ Fact 1: Let $\mathcal{H} = \mathcal{H}_l \circ \cdots \circ \mathcal{H}_1$. Then, $\tau_{\mathcal{H}}(m) \leqslant \prod_{t=1}^{l} \tau_{\mathcal{H}_t}(m)$.

Fully connected Neural Networks

- Neuron: input $\sum_j w_j h_j$; output $\sigma(\sum_j w_j h_j)$
- Organized into layers of depth l and width n
- Graph: V, E, σ, w; weight function.
- VC dim of $\mathcal{H}_{V,E,\mathrm{sign}} \leqslant C|E| \log |E|$
  - Proof: Growth function $\tau_{\mathcal{H}}(m) = \max_{C,|C|=m} |\mathcal{H}|_C|$ Fact 1: Let $\mathcal{H} = \mathcal{H}_l \circ \cdots \circ \mathcal{H}_1$. Then, $\tau_{\mathcal{H}}(m) \leqslant \prod_{t=1}^{l} \tau_{\mathcal{H}_t}(m)$.
  - Fact 2: Let $\mathcal{H} = \mathcal{H}^{(1)} \cdots \circ \mathcal{H}^{(n)}$. Then, $\tau_{\mathcal{H}}(m) \leqslant \prod_{t=1}^{l} \tau_{\mathcal{H}^{(t)}}(m)$.

Fully connected Neural Networks

- Neuron: input $\sum_j w_j h_j$; output $\sigma(\sum_j w_j h_j)$
- Organized into layers of depth l and width n
- Graph: V, E, σ, w; weight function.
- VC dim of $\mathcal{H}_{V,E,\mathrm{sign}} \leqslant C|E|\log|E|$
  - Proof: Growth function $\tau_{\mathcal{H}}(m) = \max_{C,|C|=m} |\mathcal{H}|_C|$ Fact 1: Let $\mathcal{H} = \mathcal{H}_l \circ \cdots \circ \mathcal{H}_1$. Then, $\tau_{\mathcal{H}}(m) \leqslant \prod_{t=1}^{l} \tau_{\mathcal{H}_t}(m)$.
  - Fact 2: Let $\mathcal{H} = \mathcal{H}^{(1)} \cdots \circ \mathcal{H}^{(n)}$. Then, $\tau_{\mathcal{H}}(m) \leqslant \prod_{t=1}^{l} \tau_{\mathcal{H}^{(t)}}(m)$.
  - Fact 3: Sauer's Lemma: $\tau_{\mathcal{H}}(m) = (em/d)^d$, where $d \geqslant$ VCdim $(\mathcal{H})$

Universal approximation theorems

Theorem [Park et al 2020, ICLR] (Informal) For $f \in L^p(\mathbb{R}^n, \mathbb{R}^m)$, and any $\epsilon > 0$, there exists a fully connected ReLU network F of width exactly $d = \max\{n+1, m\}$ such that $\|f - F\|_p^p < \epsilon$.

Kolmogorov-Arnold-Sprecher representation theorem: Any continuous multivariate function $f : \mathbb{R}^n \to \mathbb{R}$ can be written as

$$f(x) = \sum_{i=0}^{2n} \Phi\left(\sum_{j=1}^{n} w_j \sigma(x_i + \eta i) + i\right),$$

where $\sigma : [0, 1] \to [0, 1]$.

Convolutional Neural Networks (source: cs231n.stanford.edu)

- Suitable for image recognition. Won the 2012 ImageNet competition and subsequent ones.
- Three types of layers: convolutional, FC, pooling
- Convolutional layer: accepts a volume of size $W_1 \times H_1 \times D_1$ and outputs a volume of size $W_2 \times H_2 \times D_2$ where $W_2 = (W_1 - F + 2P)/S + 1$ and $H_2 = (H_1 - F + 2P)/S + 1$ and $D_2 = K$.
- K is number of filters, F is filter size, S is stride, P is padding.
- Pooling layer: downsamples along width and height, and optionally along depth.
- FC layer: computes class scores, resulting in volume of size $1 \times 1 \times K$.