Review of Lecture 6

- **Backpropagation training example**

- **Forward pass** to compute $x^{(l)}$ for $l = 1, \ldots, L$ and
  \[ s^{(l)} = (W^{(l)})^T x^{(l-1)}. \]

- **Backward pass** to recursively compute
  \[ \delta^{(l)}_j = \theta'(s^{(l)}_j) \sum_{k=1}^{\delta^{(l+1)}_k} W^{(l+1)}_{jk} \delta^{(l+1)}_k. \]
Review of Lecture 6

• The backpropagation algorithm

1. Initialize all weights $w_{ij}^{(l)}$ at random
2. for $t = 0, 1, 2, \ldots$ do
3. Pick $n \in \{1, 2, \ldots, N\}$
4. Forward: Compute all $x_j^{(l)}$
5. Backward: Compute all $\delta_j^{(l)}$
6. Update the weights: $w_{ij}^{(l)} \leftarrow w_{ij}^{(l)} - \eta \cdot x_i^{(l-1)} \cdot \delta_j^{(l)}$
7. Iterate to the next step until it is time to stop
8. Return the final weights $w_{ij}^{(l)}$

• Update options:
  - Gradient descent: Compute gradient for ALL points in dataset, then update weights
  - Stochastic gradient descent (SGD): Compute gradient for ONE point, then update weights
  - SGD (with mini-batch): Compute the gradient for SOME points (small number), then update weights

• A tighter bound on generalization error?

$$E_{out}(g) \leq E_{in}(g) + \sqrt{\frac{2|H|}{2N} \ln \frac{2|H|}{\delta}}$$

$$E_{out}(g) \leq E_{in}(g) + \sqrt{\frac{8}{N} \ln \frac{4m_H}{\delta}}$$

The new bound will be applicable to infinite $|H|$.
Today’s Lecture

• Theory of generalization (continued)

• Generalization and the VC-dimension
Dichotomies: Mini-hypotheses

A hypothesis \( h : X \rightarrow \{-1, +1\} \)

A dichotomy \( h : \{x_1, x_2, \ldots, x_N\} \rightarrow \{-1, +1\} \)

The number of hypotheses \(|H|\) can be infinite.

The number of dichotomies \(|H(x_1, x_2, \ldots, x_N)|\) is at most \(2^N\).

This measure is a candidate for replacing \(|H|\) when bounding generalization error.

\[
E_{out}(g) \leq E_{in}(g) + \sqrt{\frac{1}{2N} \ln \frac{2|H|}{\delta}}
\]
The growth function

The growth function counts the most dichotomies on any $N$ points.

$$m_H(N) = \max_{x_1, \ldots, x_N \in X} |H(x_1, \ldots, x_N)|$$

The growth function satisfies:

$$m_H(N) \leq 2^N$$
Example: 2-D perceptron

Cannot implement.  
Can implement all 8.  
Can implement at most 14.

\[ m_H(3) = 8 = 2^3. \]

\[ m_H(4) = 14 < 2^4. \]

What about \( m_H(5) \)?
Example: Convex sets

$H$ is a set of $h : \mathbb{R}^2 \rightarrow \{-1, +1\}$

$h(x) = +1$ is convex

$m_H(N) = 2^N$

The points are ‘shattered’ by convex sets.
Shattering

If $H$ is capable of generating ALL possible dichotomies on $x_1, \ldots, x_N$, then:

$$H(x_1, \ldots, x_N) = \{-1, +1\}^N$$

and we say that $H$ can shatter $x_1, \ldots, x_N$.

$H$ is as diverse as can be on this particular sample.
Break point of $H$

Definition:

If no dataset of size $k$ can be shattered by $H$, then $k$ is a break point for $H$.

$$m_H(k) < 2^k$$

For perceptrons, $k = 4$.

(larger datasets cannot be shattered either.)

Break points are generally easier to find than the full growth function.
3 example break points

- **Positive rays** \( m_H(N) = N + 1 \)
  - break point \( k = 2 \)

- **Positive intervals** \( m_H(N) = \frac{1}{2}N^2 + \frac{1}{2}N + 1 \)
  - break point \( k = 3 \)

- **Convex sets** \( m_H(N) = 2^N \)
  - break point \( k \to \infty \)
Main result

No break point $\implies m_H(N) = 2^N$

Any break point $\implies m_H(N)$ is polynomial in $N$
The polynomial relationship

**Theorem:** If \( m_H(k) < 2^k \) for some value \( k \), then

\[
m_H(N) \leq \sum_{i=0}^{k-1} \binom{N}{i}
\]

for all \( N \). The RHS is polynomial in \( N \) of degree \( k - 1 \).

(Proof for the bound of the growth function in LFD, Ch. 2.1.)

If \( H \) has a break point, we have what we want to ensure good generalization.
The VC dimension is the single parameter that characterizes the growth function.

**Definition:** The Vapnik-Chervonenkis dimension of a hypothesis set $H$, denoted by $d_{vc}(H)$ or simply $d_{vc}$, is the largest value of $N$ for which $m_H(N) = 2^N$. If $m_H(N) = 2^N$ for all $N$, then $d_{vc}(H) = \infty$.

\[
m_H(N) \leq \sum_{i=0}^{k-1} \binom{N}{i} = \sum_{i=0}^{d_{vc}} \binom{N}{i} \tag{max power is $N^{d_{vc}}$}
\]

Can prove by induction, $m_H(N) \leq N^{d_{vc}} + 1$
What are the implications of the VC dim?

\[ E_{out} \leq E_{in} + \sqrt{\frac{1}{2N} \ln \frac{2m_H(N)}{\delta}} \]  
(recall: error bound from Hoeffding’s and if we replace \(|H|\) with \(m_H(N)\))

1. Unless \(d_{vc}(H) = \infty\), \(m_H(N)\) is bound by a polynomial in \(N\);

2. \(m_H(N)\) grows logarithmically in \(N\) regardless of the order of the polynomial;

3. It will be crushed by the \(\frac{1}{N}\) factor.

4. Therefore, for any fixed tolerance, \(\delta\), the bound on \(E_{out}\) will be arbitrarily close to \(E_{in}\) for sufficiently large \(N\).

5. The smaller the \(d_{vc}\) is, the faster the convergence to zero.

**Only if** \(d_{vc}(H) = \infty\) **will this argument fail** (growth function is exponential).
The VC Generalization Bound

\[ \Pr[|E_{\text{in}}(g) - E_{\text{out}}(g)| > \epsilon] \leq 2e^{-2\epsilon^2N}, \ \epsilon > 0 \]

\[ \Pr[|E_{\text{in}}(g) - E_{\text{out}}(g)| > \epsilon] \leq 4m_H(2N)e^{-\epsilon^2N/8}, \ \epsilon > 0 \]

\[ E_{\text{out}}(g) \leq E_{\text{in}}(g) + \sqrt{\frac{1}{2N} \log \frac{2|H|}{\delta}} \]

w.p. at least \( 1 - \delta \)

\[ E_{\text{out}}(g) \leq E_{\text{in}}(g) + \sqrt{\frac{8}{N} \log \frac{4m_H(2N)}{\delta}} \]

w.p. at least \( 1 - \delta \)
The VC dimension and learning

\[ d_{vc}(H) \] is finite \[ \implies g \in H \] will generalize.

- Independent of the learning algorithm
- Independent of the input distribution
- Independent of the target function
VC dimension of perceptrons

For $d = 2$, $d_{vc} = 3$

In general, $d_{vc} = d + 1$

We will prove in two directions:

$$d_{vc} \leq d + 1$$

$$d_{vc} \geq d + 1$$
The first direction

A set of $N = d + 1$ points in $\mathbb{R}^d$ shattered by the perceptron:

$$X = \begin{bmatrix} x_1^T \\ x_2^T \\ \vdots \\ x_{d+1}^T \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & \ldots & 0 \\ 1 & 1 & 0 & \ldots & 0 \\ 1 & 0 & 1 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & 0 & \ldots & 0 & 1 \end{bmatrix}$$

( $X$ is invertible)
Can we shatter this dataset?

\[ y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{d+1} \end{bmatrix} = \begin{bmatrix} \pm 1 \\ \pm 1 \\ \vdots \\ \pm 1 \end{bmatrix}, \quad \text{can we find a vector } w \text{ satisfying } \text{sign}(Xw) = y? \]

Easy! Just make \( Xw = y \)

which means \( w = X^{-1}y \)
We can shatter these $d + 1$ points

This implies what?

[a] $d_{vc} = d + 1$
[b] $d_{vc} \geq d + 1$  
[c] $d_{vc} \leq d + 1$
[d] No conclusion
Now, to show that $d_{vc} \leq d + 1$

We need to show that:

[a] There are $d + 1$ points we cannot shatter

[b] There are $d + 2$ points we cannot shatter

[c] We cannot shatter any set of $d + 1$ points

[d] We cannot shatter any set of $d + 2$ points ✅
Take any $d + 2$ points

For any $d + 2$ points,

$$x_1, \ldots, x_{d+1}, x_{d+2}$$

More points than dimensions $\implies$ we must have

$$x_j = \sum_{i \neq j} a_i x_i$$

where not all the $a_i$'s are zeros.
So?

\[ x_j = \sum_{i \neq j} a_i x_i \]

Consider the following dichotomy:

\( x_i \)'s with non-zero \( a_i \) get \( y_i = \text{sign}(a_i) \)

and \( x_j \) gets \( y_j = -1 \)

No perceptron can implement such a dichotomy!
Why?

\[ x_j = \sum_{i \neq j} a_i x_i \implies w^T x_j = \sum_{i \neq j} a_i w^T x_i \]

If \( y_i = \text{sign}(w^T x_i) = \text{sign}(a_i) \), then \( a_i w^T x_i > 0 \)

This forces \( w^T x_j = \sum_{i \neq j} a_i w^T x_i > 0 \)

Therefore, \( y_j = \text{sign}(w^T x_j) = +1 \)
Putting it together

We proved $d_{vc} \leq d + 1$ and $d_{vc} \geq d + 1$

\[ d_{vc} = d + 1 \]

What is the $d + 1$ in the perceptron?

It is the number of parameters $w_0, w_1, \ldots, w_d$
Interpreting the VC dimension

Parameters create degrees of freedom

# of parameters: **analog** degrees of freedom

$d_{vc}$ : equivalent ‘**binary**’ degrees of freedom
The usual suspects

Positive rays ($d_{vc} = 1$):

Positive intervals ($d_{vc} = 2$):
Not just parameters

Parameters may not contribute degrees of freedom:

\[ d_{vc} \] measures the effective number of parameters.
Amount of data needed?

Set the error bar at \( e \).

\[
e = \sqrt{\frac{8}{N} \cdot \frac{4((2N)^{d_{vc}} + 1)}{\delta}}
\]

Solve for \( N \):

\[
N = \frac{8}{e^2 \ln \frac{4((2N)^{d_{vc}} + 1)}{\delta}} = O(d_{vc} \ln N)
\]

**Example.** \( d_{vc} = 3 \); error bar \( e = 0.1 \); confidence 90\% (\( \delta = 0.1 \)).

A simple iterative method works well. Trying \( N = 1000 \) we get

\[
N \approx \frac{8}{0.1^2 \ln \frac{4(2000)^3 + 4}{0.1}} = O(d_{vc} \ln N) \approx 21192
\]

We continue iteratively, and converge to \( N \approx 30000 \).

If \( d_{vc} = 4, N \approx 40000 \); for \( d_{vc} = 5, N \approx 50000 \).

\( N \propto d_{vc}, \) but grossly overestimates!

**Practical rule-of-thumb:** \( N \geq 10d_{vc} \)
Open questions for deep neural networks

1. Why/how does optimization find decent solutions? The surfaces are highly non-convex.

2. Why do deep, large nets generalize well even with little training data? E.g., VGG19 on CIFAR10: 6M+ variables; 50K samples.

3. Expressiveness/interpretability: What are the nodes expressing? How useful is depth?

(Sanjeev Arora, DeepMath 2018)
Further reading


