So far, we have introduced...

- Mallat’s invariant scattering transform networks
  - The deeper the network, the more invariant (translation, local deformation, scaling and rotation)
- Poggio et al. local (sparse), hierarchical, compositional functions
  - Avoid the curse of dimensionality
- Panyan et al. sparse cascaded convolutional dictionary learning
  - Uniqueness and stability guarantees of sparse recovery
Let’s continue on …

- **Harmonic Analysis**: What are the optimal (transferrable) representations of functions as input signals (sounds, images, …)?

- **Approximation Theory**: When and why are deep networks better than shallow networks?

- **Optimization**: What is the landscape of the empirical risk and how to minimize it efficiently?

- **Statistics**: How can deep learning generalize well without overfitting the noise?
Generalization of Supervised Learning

Collect data: $\{(x_i, y_i) | i = 1, 2, ..., n\}$

Learn a model: $f: \mathcal{X} \rightarrow \mathcal{Y}, f \in \mathcal{H}$

Predict new data: $x \rightarrow f(x)$

All $(x, y) \sim \mathcal{D}$, where $\mathcal{D}$ is unknown

A common approach to learn: ERM (Empirical Risk Minimization)

$$\min R_n(w) = \frac{1}{n} \sum_{i} l(w; x_i, y_i)$$

$w$: model parameters

$l(w; x, y)$: loss function w.r.t. data

Population Risk: $R(w) := E[l(w; x, y)]$
Generalization Error

- We consider the standard ML setup:

\[
\hat{E}(\Theta) = \mathbb{E}_{(X,Y) \sim \hat{P}} \ell(\Phi(X; \Theta), Y) + R(\Theta)
\]

\[
E(\Theta) = \mathbb{E}_{(X,Y) \sim P} \ell(\Phi(X; \Theta), Y).
\]

\[
\hat{P} = \frac{1}{n} \sum_{i \leq n} \delta_{(x_i, y_i)}
\]

\[\ell(z)\] convex

\[R(\Theta):\] regularization

- Population loss decomposition (aka "fundamental theorem of ML"):

\[
E(\Theta^*) = \underbrace{\hat{E}(\Theta^*)}_{\text{training error}} + \underbrace{E(\Theta^*) - \hat{E}(\Theta^*)}_{\text{generalization gap}}.
\]

- Long history of techniques to provably control generalization error via appropriate regularization.
- Generalization error and optimization are entangled [Bottou & Bousquet]
Bias-Variance Tradeoff?

Models where $p>20n$ are common
Why big models generalize well?

CIFAR10

What happens when I turn off the regularizers?

<table>
<thead>
<tr>
<th>Model</th>
<th>parameters</th>
<th>p/n</th>
<th>Train loss</th>
<th>Test error</th>
</tr>
</thead>
<tbody>
<tr>
<td>CudaConvNet</td>
<td>145,578</td>
<td>2.9</td>
<td>0</td>
<td>23%</td>
</tr>
<tr>
<td>CudaConvNet (with regularization)</td>
<td>145,578</td>
<td>2.9</td>
<td>0.34</td>
<td>18%</td>
</tr>
<tr>
<td>MicroInception</td>
<td>1,649,402</td>
<td>33</td>
<td>0</td>
<td>14%</td>
</tr>
<tr>
<td>ResNet</td>
<td>2,401,440</td>
<td>48</td>
<td>0</td>
<td>13%</td>
</tr>
</tbody>
</table>

n=50,000  
d=3,072     
k=10

Ben Recht FoCM 2017
How to control generalization error?

- However, when $\Phi(X; \Theta)$ is a large, deep network, current best mechanism to control generalization gap has two key ingredients:
  - **Stochastic Optimization**
    - "During training, it adds the sampling noise that corresponds to empirical-population mismatch" [Léon Bottou].
  - Make the model *as large as possible*.
    - see e.g. "Understanding Deep Learning Requires Rethinking Generalization", [Ch. Zhang et al, ICLR’17].
Traditional Learning Theory

Common form of generalization bound (in expectation or high probability)

\[ R(w) \leq R_n(w) + \sqrt{\frac{\text{Capacity Measure}}{n}} \]

<table>
<thead>
<tr>
<th>Capacity Measurement</th>
<th>Complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>VC-dimension</td>
<td>( VC \leq O(</td>
</tr>
<tr>
<td>( \varepsilon )-Covering number</td>
<td>( \log_2 N_{l_1}(\mathcal{F}, \varepsilon, m) \leq O \left( \frac{(AL\phi)^{L(L+1)}}{\varepsilon^{2L}} \right) )</td>
</tr>
<tr>
<td>Rademacher Average</td>
<td>( R_m(\mathcal{F}) \leq O(\mu^L) )</td>
</tr>
</tbody>
</table>

\(|E|\): # of edges

\( L \): # of layers

Big model should fail!
Training Algorithms for Deep Learning

Commonly Used Algorithms

Non-adaptive
- SGD
- SGD with momentum
- ......

Adaptive
- Adam
- AdaGrad
- AdaDelta
- ......

Easy to implement

Learn the curvature adaptively

Automaticall y tune parameters
Stochastic Gradient Descent

Objective loss function:

\[
\min \ R_n(w) = \frac{1}{n} \sum_i l(w; x_i, y_i)
\]

where \((x_i, y_i)\) is the data, \(w\) is the parameter vector.

Gradient Descent:

\[
w_{t+1} = w_t - \frac{\eta}{n} \sum l(w_t; x_i, y_i)
\]

SGD:

\[
w_{t+1} = w_t - \eta \nabla l(w_t; x_{i_t}, y_{i_t}), \text{ where } i_t \text{ is uniform in } \{1, \ldots, n\}
\]

Extensions: mini-batch SGD

\(O(1)\) time complexity

\(O(n)\) time complexity
SGD with early stopping regularizes

![Graph showing excess risk and Lipschitz constants during training.](image)

Bartlett et al. 2017. Generalization error of AlexNet in Cifar10
Margin and Network Lipschitz based Generalization error bound (Bartlett et al. 2017)

\textbf{Theorem 1.1.} Let nonlinearities \((\sigma_1, \ldots, \sigma_L)\) and reference matrices \((M_1, \ldots, M_L)\) be given as above (i.e., \(\sigma_i\) is \(\rho_i\)-Lipschitz and \(\sigma_i(0) = 0\)). Then for \((x, y), (x_1, y_1), \ldots, (x_n, y_n)\) drawn iid from any probability distribution over \(\mathbb{R}^d \times \{1, \ldots, k\}\), with probability at least \(1 - \delta\) over \(((x_i, y_i))_{i=1}^n\), every margin \(\gamma > 0\) and network \(F_A : \mathbb{R}^d \to \mathbb{R}^k\) with weight matrices \(A = (A_1, \ldots, A_L)\) satisfy

\[
\Pr \left[ \arg \max_j F_A(x)_j \neq y \right] \leq \hat{\mathcal{R}}_{\gamma}(F_A) + \tilde{O}\left( \frac{\|X\|_2 R_A \ln(W)}{\gamma n} \ln(1/\delta) \right),
\]

where \(\hat{\mathcal{R}}_{\gamma}(f) \leq n^{-1} \sum_i 1 \left[ f(x_i)_y \leq \gamma + \max_{j \neq y_i} f(x_i)_j \right]\) and \(\|X\|_2 = \sqrt{\sum_i \|x_i\|^2_2}\).

The spectral complexity \(R_{F_A} = R_A\) of a network \(F_A\) with weights \(A\) is the defined as

\[
R_A := \left( \prod_{i=1}^L \rho_i \|A_i\|_{\sigma} \right) \left( \sum_{i=1}^L \frac{\|A_i^\top - M_i^\top\|_{2,1}^{2/3}}{\|A_i\|_{\sigma}^{2/3}} \right)^{3/2}.
\]
Stochastic Gradient/Discrete Langevin Dynamics (SGLD)

SGLD is a variant of SGD:

\[ w_{t+1} = w_t - \eta \nabla l(w_t; x_t, y_t) + \sqrt{\frac{2\eta}{\beta}} z_t, \text{ where } z_t \sim \mathcal{N}(0, I_d) \]

Injection of Gaussian noise makes SGLD completely different with SGD.

For small enough step size \( \eta_t \), Gaussian noise will dominate the stochastic gradient.
Distinctions of SGLD

Intuitively, injected isotropic Gaussian noise helps escape saddle points or local minimum.

SGLD is the discretization of following SDE

\[ dW_t = -\nabla F(W_t) dt + \sqrt{\frac{2}{\beta}} dB_t \]

where \( F(\cdot) \) is the empirical loss function, \( B(t) \) is the standard Brownian motion.

Its distribution converges to Gibbs distribution \( \propto \exp(-\beta F(w)) \)

Large \( \beta \) will concentrate on the global minimizer of \( F(w) \).
From the view of stability theory:

Under mild conditions of (surrogate) loss function, the generalization error of SGLD at $N$-th round satisfies

$$E[l(w_S, z)] - E_S[l(w_S, z)] \leq O\left(\frac{1}{n} \left( k_0 + L \beta \sum_{k=k_0+1}^{N} \eta_k \right)^\frac{1}{2}\right)$$

where $L$ is the Lipschitz constant, and $k_0 := \min \{ k : \eta_k \beta L^2 < 1 \}$

If consider high probability form, there is an additional $\tilde{O}(\sqrt{1/n})$ term
From the view of PAC-Bayesian theory:

For regularized ERM with \( R(w) = \lambda \|w\|^2 / 2 \). Under mild conditions, with high probability, the generalization error of SGLD at \( N\)-th round satisfies

\[
E[E[l(w_S, z)]] - E_S[E[l(w_S, z)]] \leq O \left( \beta \sum_{k=1}^{N} \eta_k e^{-\lambda (T_N - T_k)/2} E[\|g_k\|^2] \right)
\]

where \( T_k = \sum_{j=1}^{k} \eta_k \), \( g_k \) is the stochastic gradient in each round.
Comparison Two Results

Both bounds suggest “train faster, generalize better”, which explain the random label experiments in ICLR17

- In expectation, stability bound has a faster $O\left(\frac{1}{n}\right)$ rate.
- PAC-Bayes bound is data dependent, and doesn’t rely on Lipschitz condition.
- Effect of step sizes in PAC-Bayes exponentially decay with time.
The Landscape of Risks

• However, when $\Phi(X; \Theta)$ is a large, deep network, current best mechanism to control generalization gap has two key ingredients:
  – Stochastic Optimization
    - "during training, it adds the sampling noise that corresponds to empirical-population mismatch" [Léon Bottou].
    - Make the model as large as possible.
  - see e.g. "Understanding Deep Learning Requires Rethinking Generalization", [Ch. Zhang et al, ICLR'17].

• We first address how overparametrization affects the energy landscapes $E(\Theta), \hat{E}(\Theta)$. 
A `Deep' Dream:  
All Critical Point/local optima = Global Optima?

- Choromanska-LeCun-Ben Arous'15: most of critical values are concentrated in a narrow bind of global optima, using random Morse theory on sphere (spin class models)
- Haeffele et al.'15,16: overparameterized tensor factorization models, every local optima are global optima
- Kawaguchi'16: linear networks have no poor local optima
- Bruna et al.'16,17: simple sublevel set topology of multilinear regression, with group symmetry, and some nonlinear networks
- Chaudhari et al’17: Moreau envelope of empirical risk
- Pennington & Bahri’17: Hessian Analysis using Random Matrix Theory
A Dream: All Critical Point = Global Optima?

- Models from Statistical physics have been considered as possible approximations [Dauphin et al.'14, Choromska et al.'15, Segun et al.'15]
- Tensor factorization models capture some of the non convexity essence [Anandukar et al.'15, Cohen et al. '15, Haeffele et al.'15]
- [Soudry’16, Song et al.’16] study Empirical Risk Minimization in two-layer ReLU networks, also in the over-parametrized regime.
- [Tian’17] studies learning dynamics in a gaussian generative setting.
- [Chaudhari et al’17]: Studies local smoothing of energy landscape using the local entropy method from statistical physics.
- [Pennington & Bahri’17]: Hessian Analysis using Random Matrix Th.
- [Soltanolkotabi, Javanmard & Lee’17]: layer-wise quadratic NNs.
Nonconvexity vs. Gradient Descent

• We can perturb any convex function in such a way it is no longer convex, but such that gradient descent still converges.
• E.g. quasi-convex functions.
Symmetry and Group Invariance

\[ F(\theta) = F(g.\theta), \ g \in G \text{ compact.} \]

- We can perturb any convex function in such a way it is no longer convex, but such that gradient descent still converges.
- E.g. quasi-convex functions.
- In particular, deep models have internal symmetries.
Linear Networks

- Some authors have considered linear “deep” models as a first step towards understanding nonlinear deep models:

\[
E(W_1, \ldots, W_K) = \mathbb{E}_{(X,Y) \sim P} \|W_K \cdots W_1 X - Y\|^2.
\]

\[X \in \mathbb{R}^n, \ Y \in \mathbb{R}^m, \ W_k \in \mathbb{R}^{n_k \times n_{k-1}}.\]

**Theorem: [Kawaguchi’16]** If \( \Sigma = \mathbb{E}(XX^T) \) and \( \mathbb{E}(XY^T) \) are full-rank and \( \Sigma \) has distinct eigenvalues, then \( E(\Theta) \) has no poor local minima.

- studying critical points.
- later generalized in [Hardt & Ma’16, Lu & Kawaguchi’17]
Given loss $E(\theta), \theta \in \mathbb{R}^d$, we consider its representation in terms of level sets:

$$E(\theta) = \int_0^\infty 1(\theta \in \Omega_u) du, \quad \Omega_u = \{y \in \mathbb{R}^d : E(y) \leq u\}.$$ 

A first notion we address is about the topology of the level sets.

In particular, we ask how connected they are, i.e. how many connected components $N_u$ at each energy level $u$?
Simple Topology

- A first notion we address is about the topology of the level sets. In particular, we ask how connected they are, i.e. how many connected components $N_u$ at each energy level $u$?

- This is directly related to the question of global minima:

**Proposition:** If $N_u = 1$ for all $u$ then $E$ has no poor local minima.

(i.e. no local minima $y^*$ s.t. $E(y^*) > \min_y E(y)$)

- We say $E$ is simple in that case.
- The converse is clearly not true.
Simple Topology of Linear Networks
[Bruna-Freeman’16]

\[ E(W_1, \ldots, W_K) = \mathbb{E}_{(X,Y) \sim P} \| W_K \ldots W_1 X - Y \|^2. \]

**Proposition: [BF’16]**

1. If \( n_k > \min(n, m) \), \( 0 < k < K \), then \( N_u = 1 \) for all \( u \).

2. (2-layer case, ridge regression)
   \[
   E(W_1, W_2) = \mathbb{E}_{(X,Y) \sim P} \| W_2 W_1 X - Y \|^2 + \lambda (\|W_1\|^2 + \|W_2\|^2)
   \]
   satisfies \( N_u = 1 \ \forall \ u \) if \( n_1 > \min(n, m) \).

- We pay extra redundancy price to get simple topology.
Group Symmetries [Bruna-Venturi-Bandiera’17]

• Q: How much extra redundancy are we paying to achieve $N_u = 1$ instead of simply no poor-local minima?

- In the multilinear case, we don’t need $n_k > \min(n, m)$
  - We do the same analysis in the quotient space defined by the equivalence relationship $W \sim \tilde{W} \iff W = \tilde{W}U$, $U \in GL(\mathbb{R}^n)$.

**Corollary [LBB’17]:** The Multilinear regression $E_{(X,Y) \sim P} \|W_1 \ldots W_k X - Y\|^2$ has no poor local minima.

- Construct paths on the Grassmanian manifold of subspaces.
- Generalizes best known results for multilinear case (no assumptions on data covariance).
Nonlinear ReLU network

- Good behavior is recovered with nonlinear ReLU networks, provided they are sufficiently overparametrized:

- Setup: two-layer ReLU network:
\[ \Phi(X; \Theta) = W_2 \rho(W_1 X), \quad \rho(z) = \max(0, z) \]
\[ W_1 \in \mathbb{R}^{m \times n}, \quad W_2 \in \mathbb{R}^{m} \]

**Theorem [BF’16]**: For any \( \Theta^A, \Theta^B \in \mathbb{R}^{m \times n}, \mathbb{R}^{m} \)
with \( E(\Theta^{\{A,B\}}) \leq \lambda \), there exists path \( \gamma(t) \)
from \( \Theta^A \) and \( \Theta^B \) such that
\[ \forall t, E(\gamma(t)) \leq \max(\lambda, \epsilon) \] and \( \epsilon \sim m^{-\frac{1}{n}} \).

- Overparametrisation "wipes-out" local minima (and group symmetries).
- The bound is cursed by dimensionality, i.e. exponential in \( n \).
- Open question: polynomial rate using Taylor decomp of \( \rho(z) \)?
Better Optimization Algorithms?

- Backpropagation Algorithm (made popular by Rumelhart-Hinton-Williams’1986) as stochastic gradient descent is equivalent to Lagrange Multiplier method with gradient descent on weights (prox-linear)
  - Used in control theory (dynamic programming) in 1960s
  - It suffers from vanishing of gradients due to the chain rule of gradient map

- Experiment Four: Saturation?

![Figure: BP on MNIST[2]: Two hidden layers, speed of learning](image-url)
Multi-Layer Perceptron (MLP)
Forward Pass

- Cascade of repeated [linear operation followed by coordinatewise nonlinearity]'s
- Nonlinearities: sigmoid, hyperbolic tangent, (recently) ReLU.

**Algorithm 1** Forward pass

**Input:** $x_0$

**Output:** $x_L$

1: for $\ell = 1$ to $L$ do
2: $x_\ell = f_\ell(W_\ell x_{\ell-1} + b_\ell)$
3: end for
Stochastic Gradient Descent Training

- Training examples \( \{x^i_0\}_{i=1}^n \) and labels \( \{y^i\}_{i=1}^n \)
- Output of the network \( \{x^i_L\}_{i=1}^m \)
- Objective

\[
J(\{W_l\}, \{b_l\}) = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{2} \|y^i - x^i_L\|_2^2
\]  

(1)

- Gradient descent

\[
W_l = W_l - \eta \frac{\partial J}{\partial W_l}
\]

\[
b_l = b_l - \eta \frac{\partial J}{\partial b_l}
\]

In practice: use Stochastic Gradient Descent (SGD)
Backward Propagation as Lagrangian Multiplier (LeCun’88)

Given $n$ training examples $(I_i, y_i) \equiv \text{(input,target)}$ and $L$ layers

- **Constrained optimization**
  
  $$
  \min_{W,x} \sum_{i=1}^{n} \|x_i(L) - y_i\|_2
  $$

  subject to
  
  $$
  x_i(\ell) = f_\ell \left[ W_\ell x_i(\ell - 1) \right],
  \quad i = 1, \ldots, n, \quad \ell = 1, \ldots, L, \quad x_i(0) = I_i
  $$

- **Lagrangian formulation (Unconstrained)**
  
  $$
  \min_{W,x,B} \mathcal{L}(W, x, B)
  $$

  $$
  \mathcal{L}(W, x, B) = \sum_{i=1}^{n} \left\{ \|x_i(L) - y_i\|_2^2 + 
  \left( \sum_{\ell=1}^{L} B_i(\ell)^T (x_i(\ell) - f_\ell \left[ W_\ell x_i(\ell - 1) \right]) \right) \right\}
  $$

**BP derivation**

<table>
<thead>
<tr>
<th><strong>Forward pass</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_i(\ell) = f_\ell\left[ W_\ell x_i(\ell - 1) \right]_{A_i(\ell)}$  $\ell = 1, \ldots, L, \quad i = 1, \ldots, n$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Backward (adjoint) pass</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>$z(L) = 2\nabla f_L \left[ A_i(L) \right] (y_i - x_i(L))$</td>
</tr>
<tr>
<td>$z_i(\ell) = \nabla f_\ell \left[ A_i(\ell) \right] W_{\ell+1}^T z_i(\ell + 1)$  $\ell = 0, \ldots, L - 1$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Weight update</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>$W \leftarrow W + \lambda \frac{\partial C}{\partial W}$</td>
</tr>
<tr>
<td>$W_\ell \leftarrow W_\ell + \lambda \sum_{i=1}^{n} z_i(\ell)x_i^T(\ell - 1)$</td>
</tr>
</tbody>
</table>
Batch Normalization

**Algorithm 2** Batch normalization [Ioffe and Szegedy, 2015]

**Input:** Values of \( x \) over minibatch \( x_1 \ldots x_B \), where \( x \) is a certain channel in a certain feature vector

**Output:** Normalized, scaled and shifted values \( y_1 \ldots y_B \)

1: \( \mu = \frac{1}{B} \sum_{b=1}^{B} x_b \)
2: \( \sigma^2 = \frac{1}{B} \sum_{b=1}^{B} (x_b - \mu)^2 \)
3: \( \hat{x}_b = \frac{x_b - \mu}{\sqrt{\sigma^2 + \epsilon}} \)
4: \( y_b = \gamma \hat{x}_b + \beta \)

- Accelerates training and makes initialization less sensitive
- Zero mean and unit variance feature vectors
Alternative: (Augmented) Lagrangian Multiplier with Block Coordinate Descent

- ADMM-type: Taylor et al. ICML 2016
- Proximal Propagation, to appear in ICLR 2018
- BCD with zero Lagrangian multiplier: Zhang et al. NIPS 2017
- Discrete EMSA of PMP: Qianxiao Li et al 2017, talk on Monday in IAS workshop
  - No-vanshing gradients and parallelizable
- Some convergence theory: preliminary results on ADMM+BCD with Jinshan Zeng, Shaobo Lin, and Tsz Kit Lau et al.
Thank you!