# Stat 212b: Topics in Deep Learning
# Lecture 19

Joan Bruna
UC Berkeley

# Planning of remaining lectures

- Lecture 19
  - Optimization, Estimation and Approximation
  - Stochastic optimization
  - First order methods, Nesterov Momentum
- Lecture 20: Guest Speaker: **Soumith Chintala** (FAIR)
- Lecture 21
  - Fighting Overfitting: Dropout
  - Fighting Covariance Shift: Batch Normalization
- Lecture 22
  - Tensor Methods, SDPs and Optimization.
  - Statistical Physics
- Lecture 23: Guest Speaker: **Yann Dauphin** (FAIR)
- Lectures 24-25: Oral Presentations/ Open Problems.

# From unsupervised to *self-supervised* learning

# From unsupervised to *self-supervised* learning

- So far, we have seen models that attempt to estimate a density of the input domain $x \in \mathbb{R}^n$

$$p(x) = \int p(h)p(x|h)dh \ , \ p(x|h) = \exp(\langle \theta_h, \Phi(x) \rangle - A(\theta_h))$$

$$p(x) = p_0(\Phi(x)) \cdot |\det \nabla \Phi(x)|^{-1}$$

# From unsupervised to *self-supervised* learning

- So far, we have seen models that attempt to estimate a density of the input domain $x \in \mathbb{R}^n$

$$p(x) = \int p(h)p(x|h)dh \ , \ p(x|h) = \exp(\langle \theta_h, \Phi(x) \rangle - A(\theta_h))$$

$$p(x) = p_0(\Phi(x)) \cdot |\det \nabla \Phi(x)|^{-1}$$

- Chained Bayes Rule: for any ordering $(x_{\sigma(1)}, \ldots, x_{\sigma(n)})$ of the coordinates we have

$$p(x) = \prod_{i \leq n} p(x_{\sigma(i)} | x_{\sigma(1)} \ldots x_{\sigma(i-1)})$$

# From unsupervised to *self-supervised* learning

- So far, we have seen models that attempt to estimate a density of the input domain $x \in \mathbb{R}^n$

$$p(x) = \int p(h)p(x|h)dh \ , \ p(x|h) = \exp(\langle \theta_h, \Phi(x) \rangle - A(\theta_h))$$

$$p(x) = p_0(\Phi(x)) \cdot |\det \nabla \Phi(x)|^{-1}$$

- Chained Bayes Rule: for any ordering $(x_{\sigma(1)}, \ldots, x_{\sigma(n)})$ of the coordinates we have

$$p(x) = \prod_{i \leq n} p(x_{\sigma(i)} | x_{\sigma(1)} \ldots x_{\sigma(i-1)})$$

- Q: In which situations is it better to use the factorized?

# From unsupervised to *self-supervised* learning

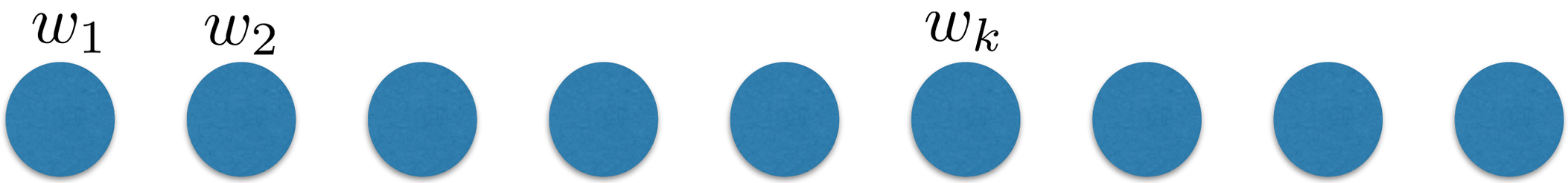
- Temporally ordered data
  - Speech, Music
  - Video
  - Language
  - Other time series (Weather, Finance, …)
- Spatially ordered data, Multi-Resolution data
  - Images

- Learning is thus reduced to the problem of conditional prediction.

$$p(x) \rightarrow \{p(x_i|x_{N(i)})\}_i$$

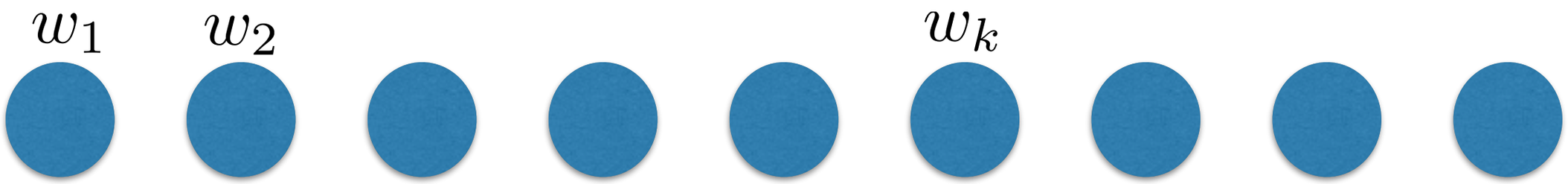
# Word2vec [Mikolov et al.'13].

- Unsupervised learning "success story".



# Word2vec [Mikolov et al.'13].

- Unsupervised learning "success story".

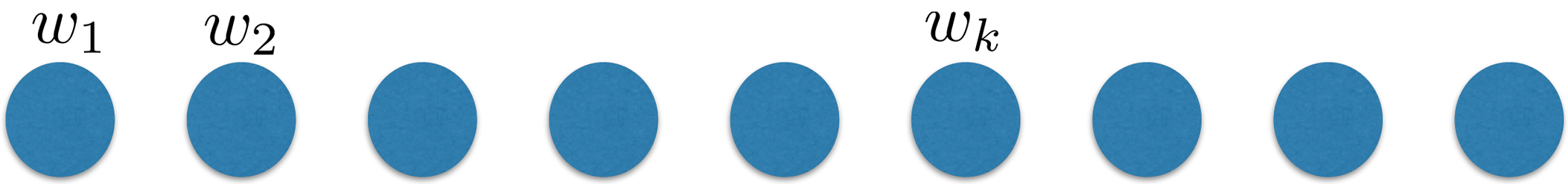



- Language creates a notion of similarity between words:

words $w_1$, $w_2$ are similar if they are "exchangeable"
i.e., they appear often within the same context.

# Word2vec [Mikolov et al.'13].

- Unsupervised learning "success story".




- Language creates a notion of similarity between words:

  words $w_1$, $w_2$ are similar if they are "exchangeable" i.e., they appear often within the same context.

- *Goal*: find a word representation $\Phi(w_i) \in \mathbb{R}^d$ that expresses this similarity as a dot product

$$\mathrm{sim}(w_i, w_j) \approx \langle \Phi(w_i), \Phi(w_j) \rangle \ .$$

# Word2vec [Mikolov et al.'13].

- Main idea: Skip-gram with negative sampling.
- Construct a "training set"
  - *positive pairs* $\mathcal{D} = \{(w_k, c_k)\}_k$ of (words, contexts) appearing in a huge language corpus.
  - *negative pairs* $\mathcal{D}' = \{(w_{k'}, c_{k'})\}_{k'}$ of (words, contexts) not appearing in the corpus.

# Word2vec [Mikolov et al.'13].

- Main idea: Skip-gram with negative sampling.
- Construct a "training set"
  - *positive pairs* $\mathcal{D} = \{(w_k, c_k)\}_k$ of (words, contexts) appearing in a huge language corpus.
  - *negative pairs* $\mathcal{D}' = \{(w_{k'}, c_{k'})\}_{k'}$ of (words, contexts) not appearing in the corpus.
- Model the probability of a pair $(w, c)$ being positive as

$$p(D = 1|c, w) = \sigma(\langle v_w, v_c \rangle) \ , \ v_w, v_c \in \mathbb{R}^d$$

.

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

# Word2vec [Mikolov et al.'13].

- Main idea: Skip-gram with negative sampling.
- Construct a "training set"
  - *positive pairs* $\mathcal{D} = \{(w_k, c_k)\}_k$ of (words, contexts) appearing in a huge language corpus.
  - *negative pairs* $\mathcal{D}' = \{(w_{k'}, c_{k'})\}_{k'}$ of (words, contexts) not appearing in the corpus.
- Model the probability of a pair $(w, c)$ being positive as

$$p(D = 1|c, w) = \sigma(\langle v_w, v_c \rangle) \ , \ v_w, v_c \in \mathbb{R}^d \quad . \qquad \sigma(x) = \frac{1}{1 + e^{-x}}$$

- Training with Maximum Likelihood:

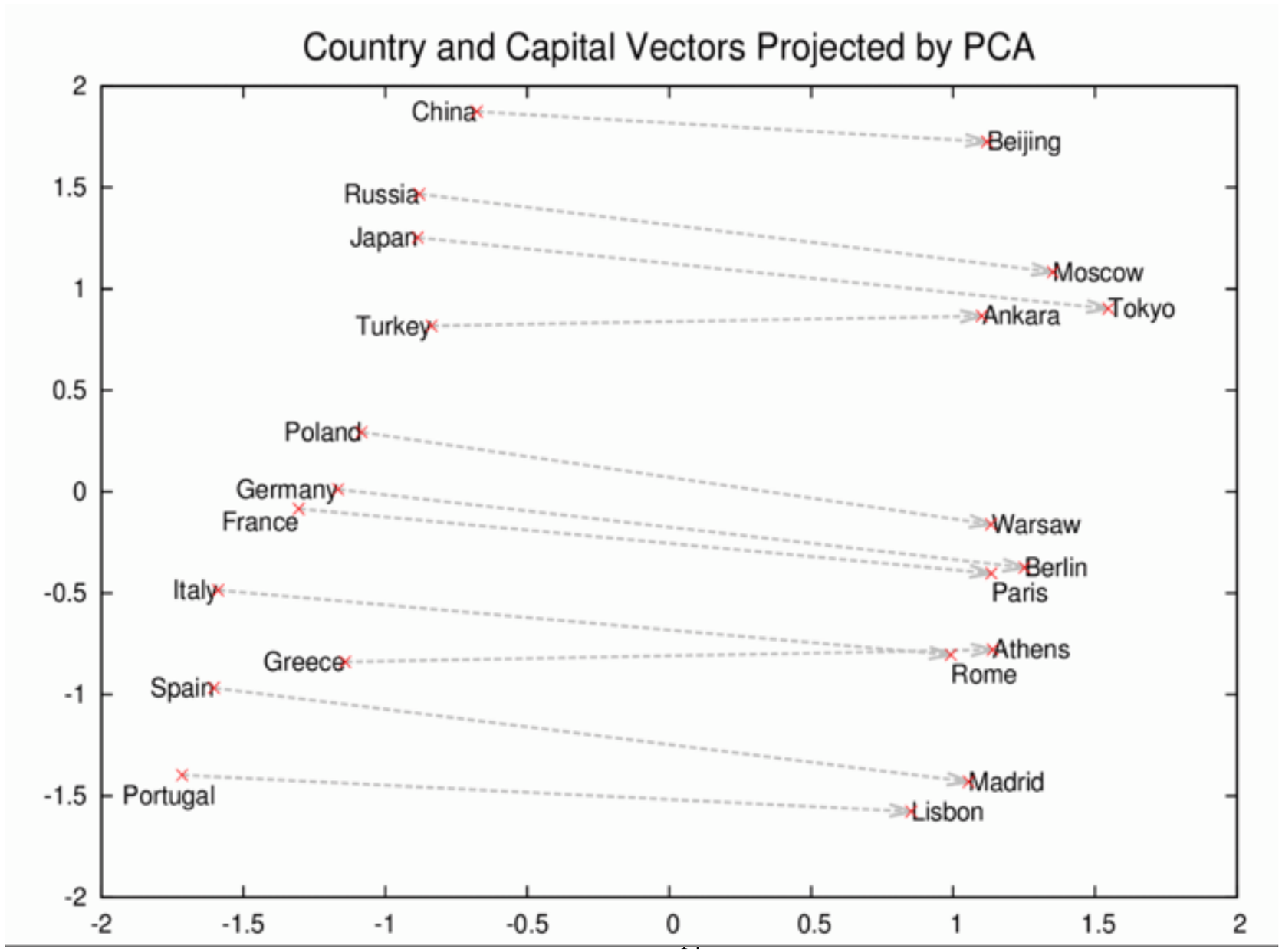
$$\arg\max_\theta \prod_{(w,c)\sim\mathcal{D}} p(D = 1|c, w, \theta) \prod_{(w,c)\sim\mathcal{D}'} p(D = 0|c, w, \theta)$$

$$\arg\max_\theta \sum_{(w,c)\sim\mathcal{D}} \log \sigma(\langle v_w, v_c \rangle) + \sum_{(w,c)\sim\mathcal{D}'} \log \sigma(-\langle v_w, v_c \rangle)$$

$\mathcal{D}$ : positive contexts $\quad$ $\mathcal{D}'$ : negative contexts

# Word2vec [Mikolov et al.'13].

- Can be seen as an implicit matrix factorization using a mutual information criteria [Yoav & Goldberg,'14].
- Huge impact on Google's business bottom-line.




*[google]*

# Video Prediction

- Rather than modeling the density of natural images

$$p(x) \ , \ x \in \mathbb{R}^d$$

- we may be also interested in modeling the conditional distributions $p(x_{t+1}|x_1, \ldots, x_t)$ where $(x_t)_t$ is temporally ordered data.

# Video Prediction

- Rather than modeling the density of natural images

$$p(x) \ , \ x \in \mathbb{R}^d$$

- we may be also interested in modeling the conditional distributions $p(x_{t+1}|x_1, \ldots, x_t)$ where $(x_t)_t$ is temporally ordered data.

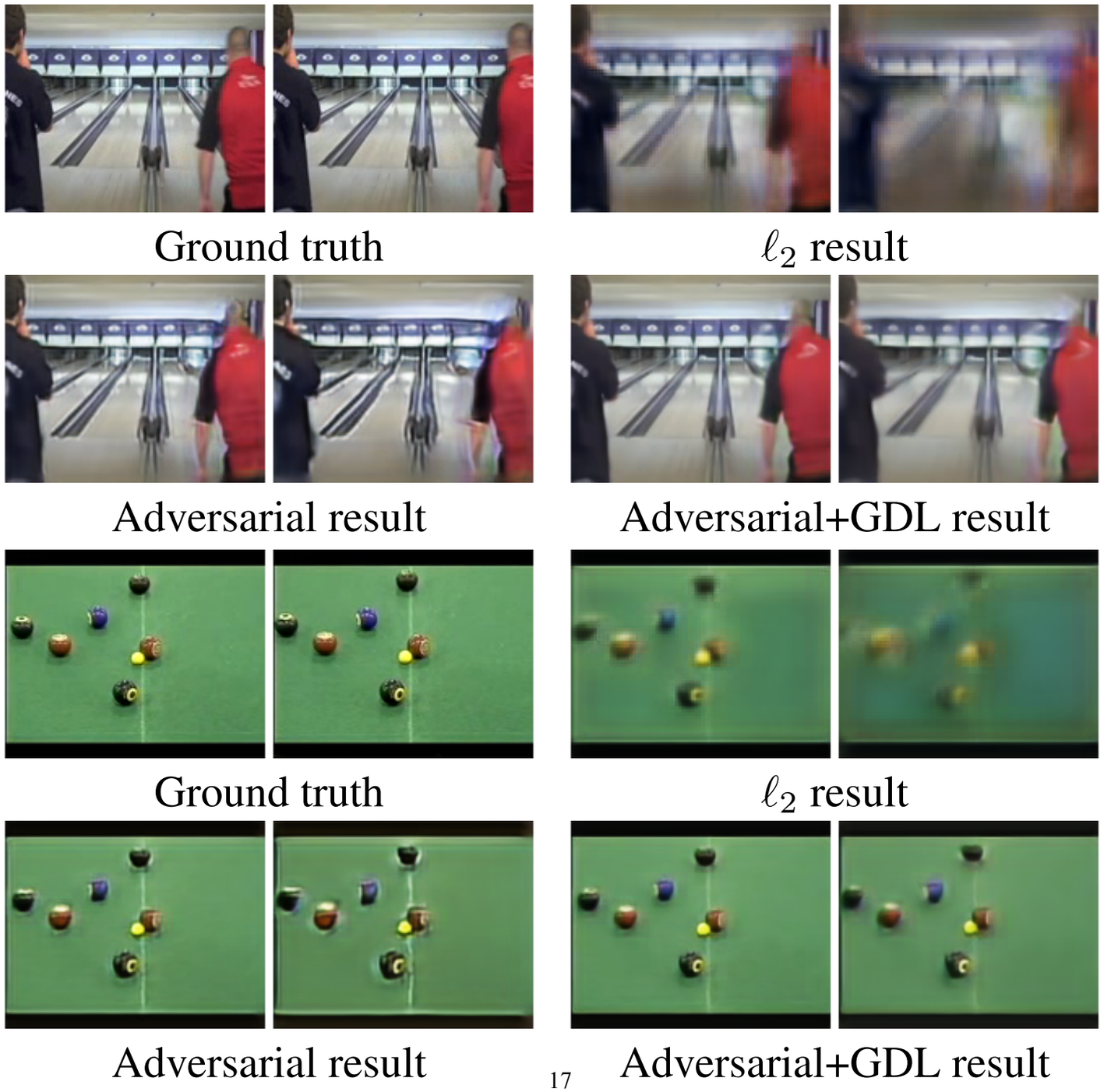
- Similarly, can we find a signal representation $\Phi(x_t)$ that is consistent with the "video language" metric? i.e.

$$\langle \Phi(x_t), \Phi(x_s) \rangle \approx h(|t-s|)$$

- This is the objective of Slow Feature Analysis [Sejnowski et al'02, Cadieu& Olshausen'10 and many others].

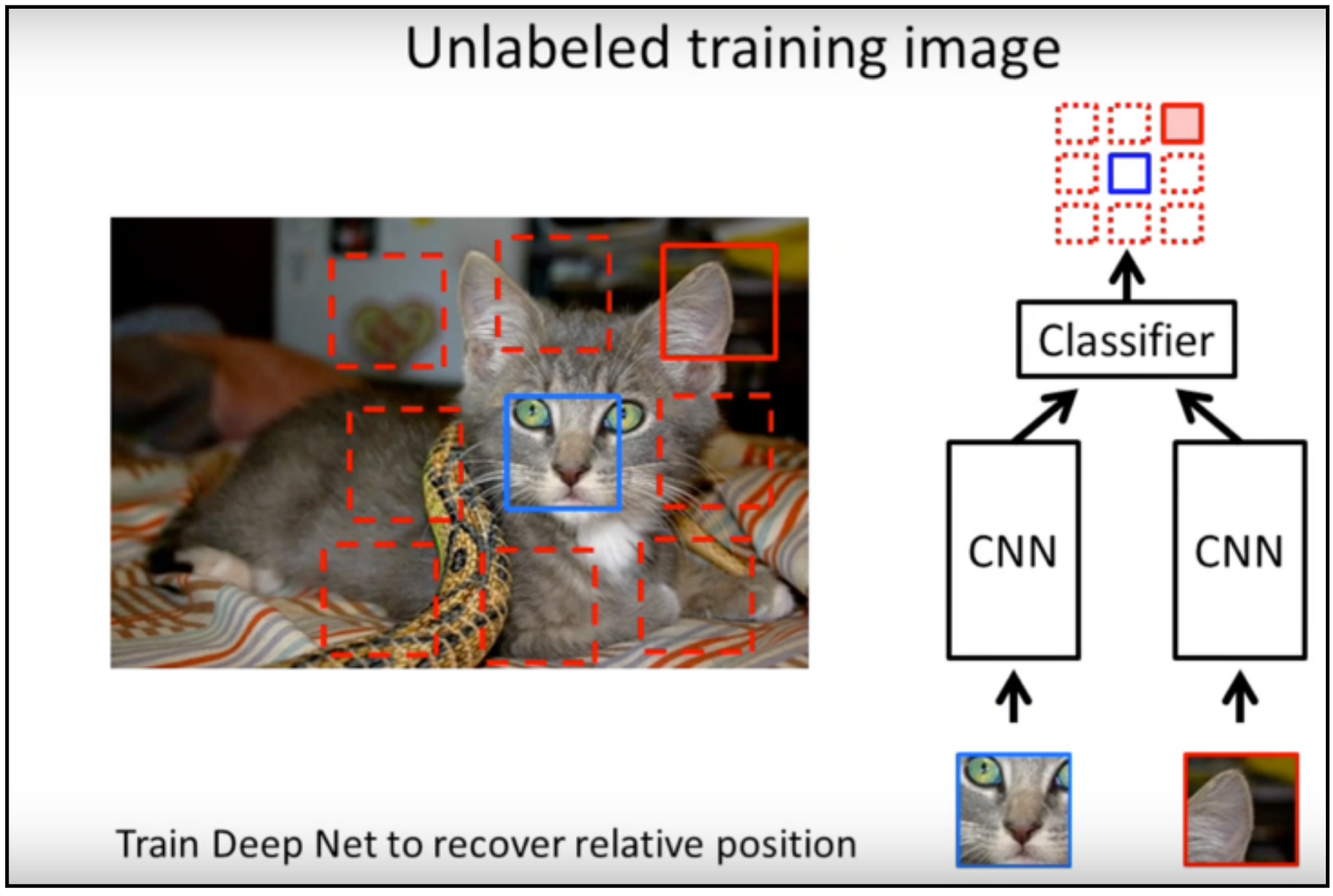
# Video Prediction

- [Mathieu, Couprie, LeCun,'16]: Conditional video prediction using CNNs and an adversarial cost:

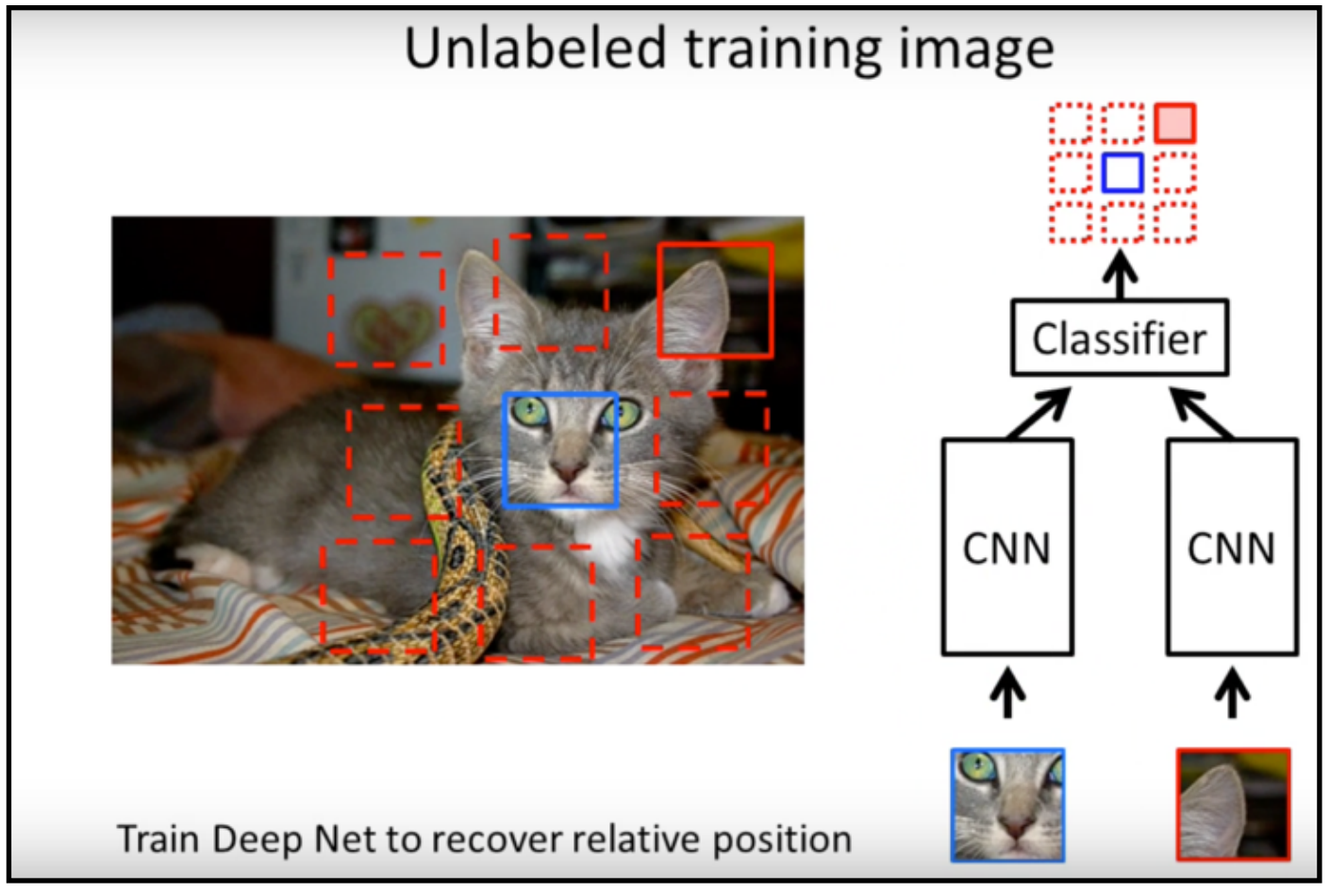


# Patch Relative Configuration [Doerch et al.'15]

- Generalize the idea of positive, negative pairs to a multi-class classification problem about spatial configurations.



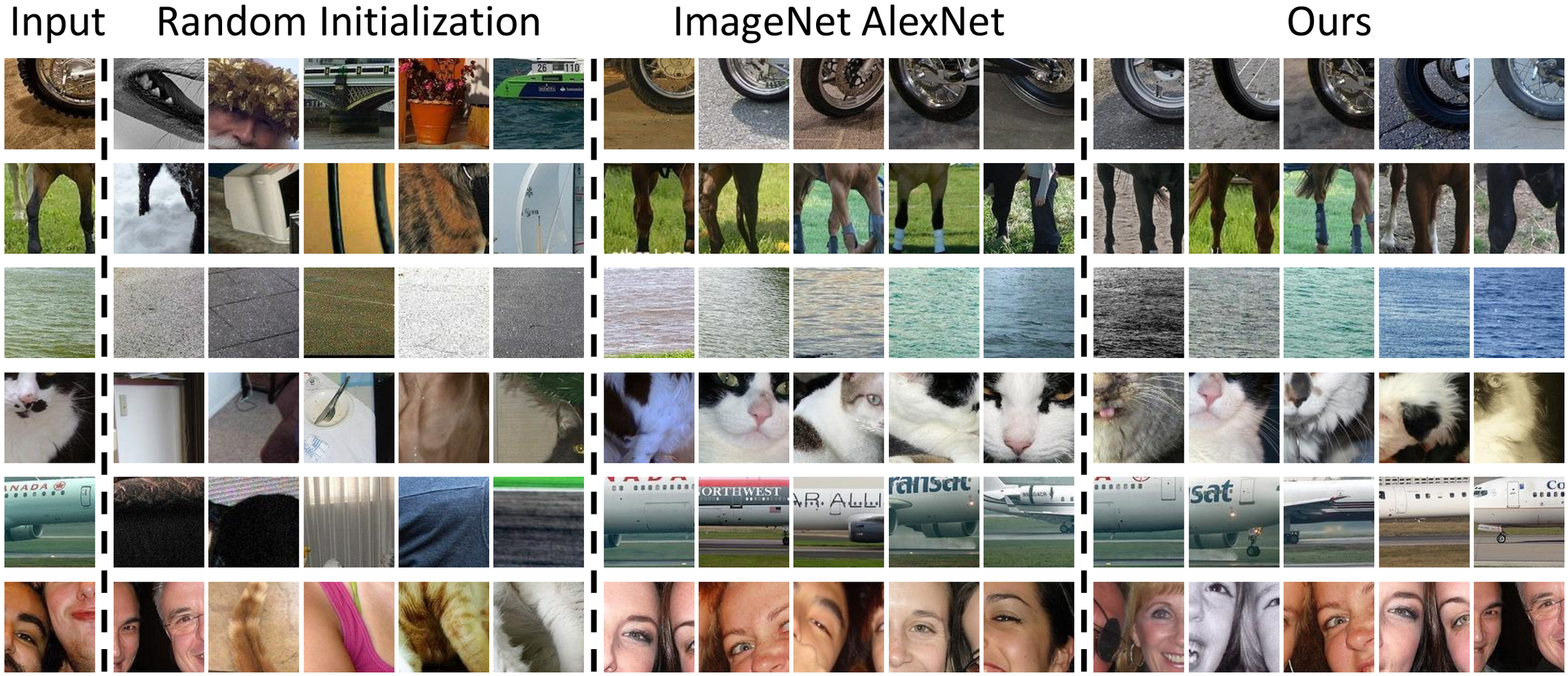
# Patch Relative Configuration [Doerch et al.'15]




- Premise: A patch representation $\Phi(x)$ that does well in this task indirectly builds object priors.
- The criterion is not generative, but it retains enough information to generalize to other tasks.

# Patch Relative Configuration [Doerch et al.'15]

- Retrieval tasks:




- The representation captures visual similarity, leveraged in object detection, retrieval, etc.

# Pixel Recurrent Networks

- Prediction tasks of the form $\hat{x_{t+1}} = F(x_1, \ldots, x_t)$ require a loss or an associated likelihood

$$e.g.\ \|\hat{x}_{t+1} - x_{t+1}\|^2 \ \Leftrightarrow\ p(x_{t+1}|x_1, \ldots, x_t) = \mathcal{N}(F(x_1, \ldots, x_t), I)$$

# Pixel Recurrent Networks

- Prediction tasks of the form $\hat{x_{t+1}} = F(x_1, \ldots, x_t)$ require a loss or an associated likelihood

$$e.g.\ \|\hat{x}_{t+1} - x_{t+1}\|^2 \ \Leftrightarrow\ p(x_{t+1}|x_1, \ldots, x_t) = \mathcal{N}(F(x_1, \ldots, x_t), I)$$

- In discrete domains we simply use a multinomial loss, in continuous domains there is no principled choice.

- How about images?

# Pixel Recurrent Networks

- Prediction tasks of the form $\hat{x_{t+1}} = F(x_1, \ldots, x_t)$ require a loss or an associated likelihood

$$e.g.\ \|\hat{x}_{t+1} - x_{t+1}\|^2 \ \Leftrightarrow\ p(x_{t+1}|x_1, \ldots, x_t) = \mathcal{N}(F(x_1, \ldots, x_t), I)$$

- In discrete domains we simply use a multinomial loss, in continuous domains there is no principled choice.

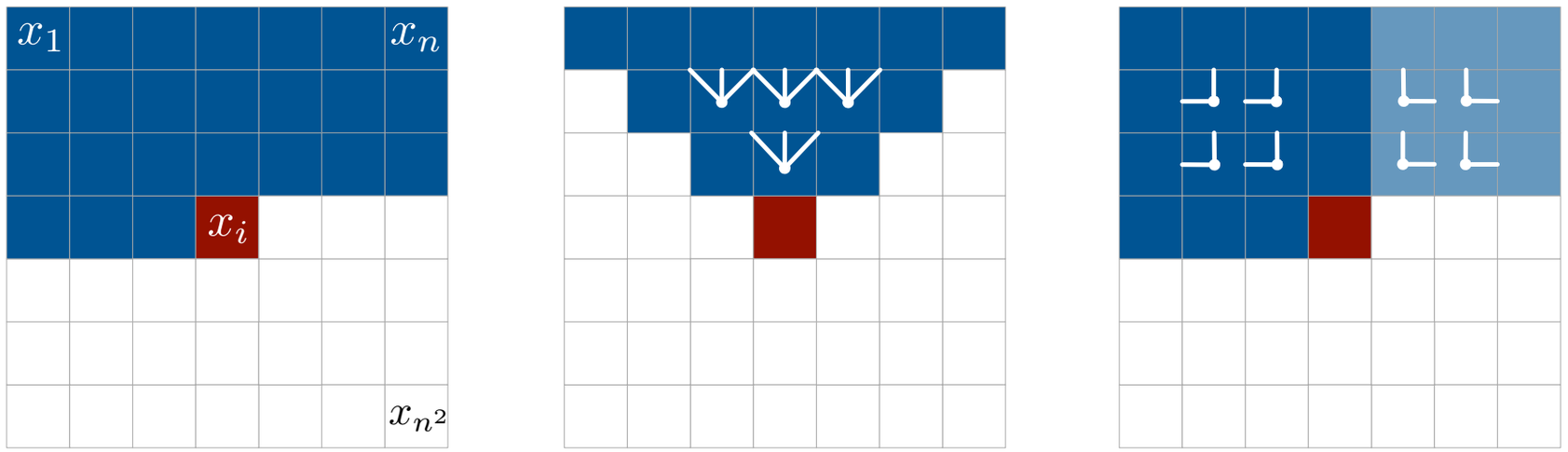
- How about images?
  - We can treat them as discrete two-dimensional grids

    $x(u) \in \{0, 255\}$
  - Model each pixel from its "past" context:

$$p(x(u)|x(v); v \in \Omega(u)) = \mathrm{softmax}\left(\Phi(x, \Omega(u))\right)$$

# Pixel Recurrent Networks [v.d.Oord et al'16]

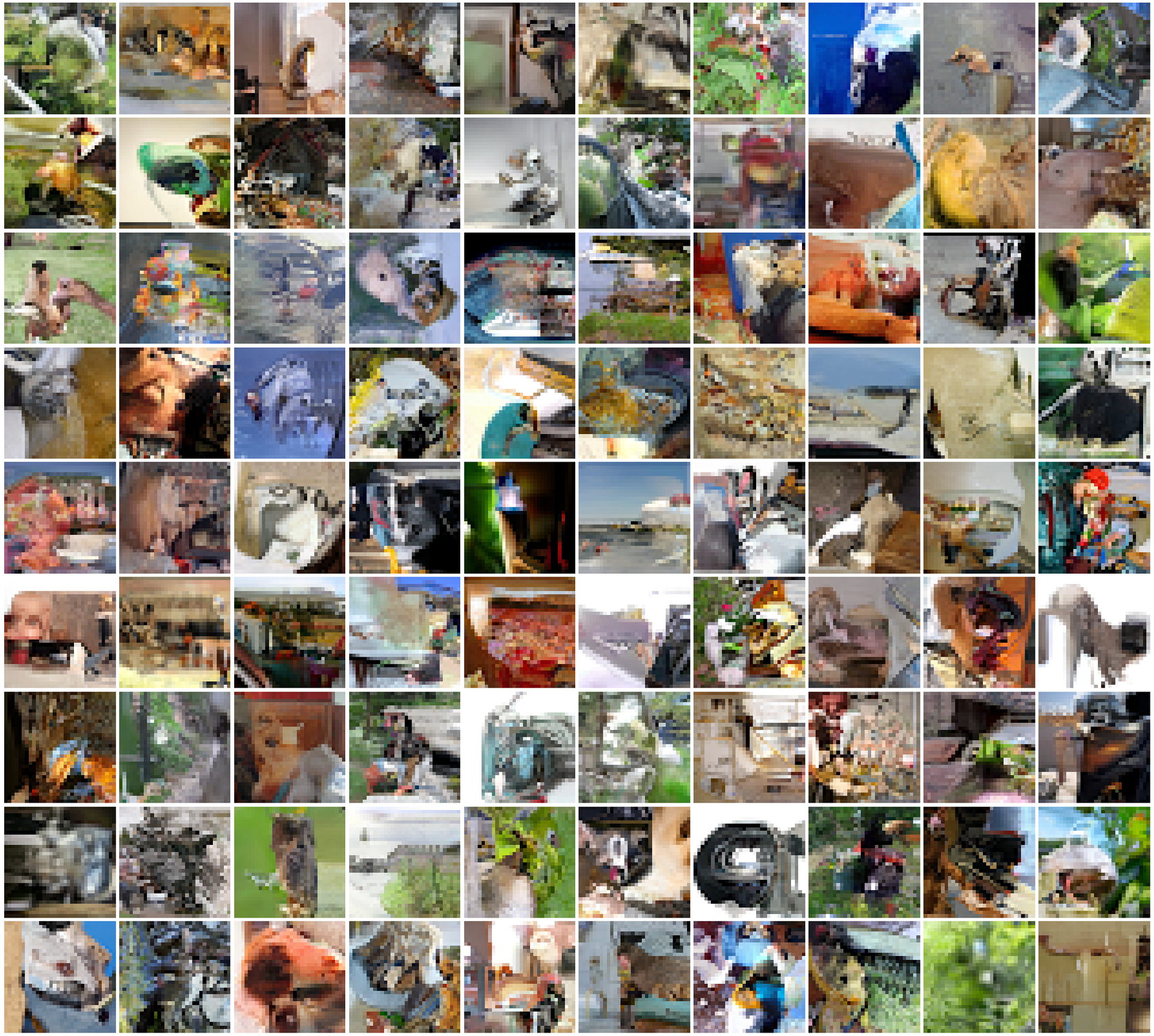
- Contexts are modeled using "diagonal BiLSTMS".




- Multi-Scale architecture conditions generations upon low-resolution samples (similarly as in LAPGANS).
- Very deep Recurrent Networks (> 10 layers).

# Pixel Recurrent Networks [v.d.Oord et al'16]

- State-of-the-art image generation and modeling.



# Pixel Recurrent Networks [v.d.Oord et al'16]

occluded completions original

occluded completions original

# Pixel Recurrent Networks [v.d.Oord et al'16]

- MNIST and Cifar-10 log-likelihoods:

| Model | NLL Test |
|---|---|
| DBM 2hl [1]: | $\approx$ 84.62 |
| DBN 2hl [2]: | $\approx$ 84.55 |
| NADE [3]: | 88.33 |
| EoNADE 2hl (128 orderings) [3]: | 85.10 |
| EoNADE-5 2hl (128 orderings) [4]: | 84.68 |
| DLGM [5]: | $\approx$ 86.60 |
| DLGM 8 leapfrog steps [6]: | $\approx$ 85.51 |
| DARN 1hl [7]: | $\approx$ 84.13 |
| MADE 2hl (32 masks) [8]: | 86.64 |
| DRAW [9]: | $\leq$ 80.97 |
| Diagonal BiLSTM (1 layer, $h = 32$): | **80.75** |
| Diagonal BiLSTM (7 layers, $h = 16$): | **79.20** |

*Table 4.* Test set performance of different models on MNIST in *nats* (negative log-likelihood). Prior results taken from [1] (Salakhutdinov & Hinton, 2009), [2] (Murray & Salakhutdinov, 2009), [3] (Uria et al., 2014), [4] (Raiko et al., 2014), [5] (Rezende et al., 2014), [6] (Salimans et al., 2015), [7] (Gregor et al., 2014), [8] (Germain et al., 2015), [9] (Gregor et al., 2015).

| Model | NLL Test (Train) |
|---|---|
| Uniform Distribution: | 8.00 |
| Multivariate Gaussian: | 4.70 |
| NICE [1]: | 4.48 |
| Deep Diffusion [2]: | 4.20 |
| Deep GMMs [3]: | 4.00 |
| RIDE [4]: | 3.47 |
| PixelCNN: | 3.14 (3.08) |
| Row LSTM: | 3.07 (3.00) |
| Diagonal BiLSTM: | **3.00** (2.93) |

*Table 5.* Test set performance of different models on CIFAR-10 in *bits/dim*. For our models we give training performance in brackets. [1] (Dinh et al., 2014), [2] (Sohl-Dickstein et al., 2015), [3] (van den Oord & Schrauwen, 2014a), [4] personal communication (Theis & Bethge, 2015).

# Self-supervised vs Unsupervised

- Two views on the same underlying problem: How to measure errors in our models?
  - Self-supervision finds useful proxies that ensure enough discriminative information is kept.
  - Unconditional probabilistic models define a similarity kernel via the Fisher kernel.

- Depending on the application/ dataset, the indirect route might be more effective than the direct one.

# Optimization in Deep Learning

# Optimization Set-up

- Our general problem is of the form

$$\min_{\Phi} \ \mathbb{E}_{z\sim\pi} f(z;\Phi) := F(\Phi) \ .$$

Supervised Learning:
$z = (x, y)$

$f(x, y; \Phi) = \log p(y|x; \Phi) = \ell(y, \Phi(x))$

$\pi$ : joint data/labels distribution

Unsupervised Learning:
$z = x$

$f(x; \Phi) = \log \Phi(x)$

$\pi$ : data distribution

# Optimization Set-up

$$\min_{\Phi} \ \mathbb{E}_{z\sim\pi} f(z;\Phi) := F(\Phi) \ .$$

- Challenges:
  - *Statistical*: The function $F$ to be optimized is unknown: only access to an estimator

$$\hat{F}_n(\Phi) = \frac{1}{n}\sum_{i<n} f(z_i,\Phi) \ , \ \{z_i\}_{i\leq n} : \text{training set.}$$

# Optimization Set-up

$$\min_{\Phi} \ \mathbb{E}_{z\sim\pi} f(z;\Phi) := F(\Phi) \ .$$

- Challenges:
  - *Statistical*: The function $F$ to be optimized is unknown: only access to an estimator

$$\hat{F}_n(\Phi) = \frac{1}{n}\sum_{i<n} f(z_i,\Phi) \ , \ \{z_i\}_{i\leq n} : \text{training set.}$$

  - *Analytical*: In practice, we search within a parametric functional class

$$\mathcal{F} = \{\Phi = \Phi(\,\cdot\,;\Theta)\,;\ \Theta \in \mathcal{X}\}$$

# Optimization Set-up

$$\min_{\Phi} \ \mathbb{E}_{z\sim\pi} f(z;\Phi) := F(\Phi) \ .$$

- Challenges:
  - *Statistical*: The function $F$ to be optimized is unknown: only access to an estimator

$$\hat{F}_n(\Phi) = \frac{1}{n}\sum_{i<n} f(z_i,\Phi) \ , \ \{z_i\}_{i\leq n} : \text{training set.}$$

  - *Analytical*: In practice, we search within a parametric functional class

$$\mathcal{F} = \{\Phi = \Phi(\,\cdot\,;\Theta)\,;\ \Theta \in \mathcal{X}\}$$

  - *Numerical*: Algorithms to optimize $\hat{F}_n$.
    - What can we say when $\hat{F}_n$ is non-convex?
    - What is the convergence rate of iterative solutions to stationary points? $\|\Theta^{(k)} - \Theta^*\| = O(h(k)) \ .$

# Decomposition of Error

[Bottou, Bousquet '08]

- Define

$$\Phi^* = \arg\min_{\Phi} F(\Phi) \ , \ \text{optimal model} \ ,$$

$$\Phi^*_{\mathcal{F}} = \arg\min_{\Phi \in \mathcal{F}} F(\Phi) \ , \ \text{optimal achievable model in } \mathcal{F} \ ,$$

$$\Phi_{\mathcal{F},n} = \arg\min_{\Phi \in \mathcal{F}} \hat{F}_n(\Phi) \ , \ \text{optimal empirical model in } \mathcal{F} \ ,$$

$$\widetilde{\Phi}_{\mathcal{F},n} = \ \text{solution of our optimization of } \min_{\Phi \in \mathcal{F}} \hat{F}_n(\Phi) \ ,$$

# Decomposition of Error

[Bottou, Bousquet '08]

- Define

$$\Phi^* = \arg\min_{\Phi} F(\Phi) \ , \ \text{optimal model} \ ,$$

$$\Phi^*_{\mathcal{F}} = \arg\min_{\Phi \in \mathcal{F}} F(\Phi) \ , \ \text{optimal achievable model in } \mathcal{F} \ ,$$

$$\Phi_{\mathcal{F},n} = \arg\min_{\Phi \in \mathcal{F}} \hat{F}_n(\Phi) \ , \ \text{optimal empirical model in } \mathcal{F} \ ,$$

$$\widetilde{\Phi}_{\mathcal{F},n} = \ \text{solution of our optimization of } \min_{\Phi \in \mathcal{F}} \hat{F}_n(\Phi) \ ,$$

- *Remark*: we can also modify empirical risk minimization with a regularizer (structured risk minimization):

$$\hat{F}_n(\Theta) = \frac{1}{n} \sum_{i \leq n} f(z_i, \Phi(\Theta)) + \lambda \mathcal{R}(\Theta) \ ,$$

# Decomposition of Error

[Bottou, Bousquet '08]

- Define

$$\Phi^* = \arg\min_{\Phi} F(\Phi) \ , \ \text{optimal model} \ ,$$

$$\Phi^*_{\mathcal{F}} = \arg\min_{\Phi \in \mathcal{F}} F(\Phi) \ , \ \text{optimal achievable model in } \mathcal{F} \ ,$$

$$\Phi_{\mathcal{F},n} = \arg\min_{\Phi \in \mathcal{F}} \hat{F}_n(\Phi) \ , \ \text{optimal empirical model in } \mathcal{F} \ ,$$

$$\widetilde{\Phi}_{\mathcal{F},n} = \ \text{solution of our optimization of } \min_{\Phi \in \mathcal{F}} \hat{F}_n(\Phi) \ ,$$

- Regret is decomposed as

$$\begin{aligned} F(\widetilde{\Phi}_{\mathcal{F},n}) - F(\Phi^*) = {} & F(\Phi^*_{\mathcal{F}}) - F(\Phi^*) && \text{(approximation error)} \\ & + F(\Phi_{\mathcal{F},n}) - F(\Phi^*_{\mathcal{F}}) && \text{(estimation error)} \\ & + F(\widetilde{\Phi}_{\mathcal{F},n}) - F(\Phi_{\mathcal{F},n}) \ . && \text{(optimization error)} \end{aligned}$$

# Constrained Approximation, Estimation and Optimization

[Bottou, Bousquet '08]

- Our goal is thus to minimize regret with respect to

$$\text{model } \mathcal{F}, \text{ optimization tolerance } \rho, \text{ number of examples } n$$

$$\text{subject to } n \leq n_{max}, \text{ compute time } T \leq T_{max}$$

# Constrained Approximation, Estimation and Optimization

[Bottou, Bousquet '08]

- Our goal is thus to minimize regret with respect to

$$\text{model } \mathcal{F}, \text{ optimization tolerance } \rho, \text{ number of examples } n$$

$$\text{subject to } n \leq n_{max}, \text{ compute time } T \leq T_{max}$$

- Constrained optimization trade-offs:

Approximation error decreases as $\mathcal{F}$ gets larger

Estimation error decreases as $n$ gets larger.
Estimation error increases as $\mathcal{F}$ gets larger.

Optimization error increases as $\rho$ gets larger.

# Two learning regimes

[Bottou, Bousquet '08]

- *Small-scale learning problems*: the active constraint is the number of examples: $n = n_{max}$ .

- *Large-scale learning problems*: the active constraint is the maximum computation time: $T = T_{max}$ .

# Two learning regimes

[Bottou, Bousquet '08]

- *Small-scale learning problems*: the active constraint is the number of examples: $n = n_{max}$ .

- *Large-scale learning problems*: the active constraint is the maximum computation time: $T = T_{max}$ .

- Statistical Learning Theory is mostly concerned with small-scale learning problems.
- Whereas small-scale learning problems can neglect the role of the optimization, large-scale problems cannot: generally it is not a good choice to fully optimize the empirical objective function.

# Empirical Risk Minimization

- Q: How to control the estimation error?

# Empirical Risk Minimization

- Q: How to control the estimation error?

consider first a bounded loss $\ell(x, y) \leq b$.

For a given $\Phi \in \mathcal{F}$, we consider

$$F(\Phi) - F_n(\Phi) = \mathbb{E}(f_\Phi(Z)) - \frac{1}{n}\sum_i f_\Phi(Z_i)$$

# Empirical Risk Minimization

- Q: How to control the estimation error?

consider first a bounded loss $\ell(x, y) \leq b$.

For a given $\Phi \in \mathcal{F}$, we consider

$$F(\Phi) - F_n(\Phi) = \mathbb{E}(f_\Phi(Z)) - \frac{1}{n}\sum_i f_\Phi(Z_i)$$

A first idea is to use Hoeffding's inequality:

$$P\left(\left|\mathbb{E}(f(Z)) - \frac{1}{n}\sum_{i\leq n} f(Z_i)\right| > \epsilon\right) \leq 2\exp\left(-\frac{2n\epsilon^2}{b^2}\right) \text{ , for } f(Z) \in [0, b] \text{ .}$$

It results that for any $\Phi$ and $\delta > 0$, with probability at least $1 - \delta$,

$$F(\Phi) \leq F_n(\Phi) + b\sqrt{\frac{\log 2/\delta}{2n}} \text{ .}$$

# Empirical Risk Minimization

- Q: How to control the estimation error?

consider first a bounded loss $\ell(x, y) \leq b$.

For a given $\Phi \in \mathcal{F}$, we consider

$$F(\Phi) - F_n(\Phi) = \mathbb{E}(f_\Phi(Z)) - \frac{1}{n}\sum_i f_\Phi(Z_i)$$

A first idea is to use Hoeffding's inequality:

$$P\left(\left|\mathbb{E}(f(Z)) - \frac{1}{n}\sum_{i\leq n} f(Z_i)\right| > \epsilon\right) \leq 2\exp\left(-\frac{2n\epsilon^2}{b^2}\right) \text{ , for } f(Z) \in [0, b] \text{ .}$$

It results that for any $\Phi$ and $\delta > 0$, with probability at least $1 - \delta$,

$$F(\Phi) \leq F_n(\Phi) + b\sqrt{\frac{\log 2/\delta}{2n}} \text{ .}$$

Is this useful?

# Empirical Risk Minimization

- We need to control the uniform deviations:

$$\sup_{\Phi \in \mathcal{F}} |F(\Phi) - F_n(\Phi)|$$

# Empirical Risk Minimization

- We need to control the uniform deviations:

$$\sup_{\Phi \in \mathcal{F}} |F(\Phi) - F_n(\Phi)|$$

Suppose first that $\mathcal{F}$ is finite: $\mathcal{F} = \{\Phi_1, \ldots, \Phi_N\}$.

For each member $j$ of the family, Hoeffding says that the bad samples

$$C_j = \{z_i; i \leq n; |F(\Phi_j) - F_n(\Phi_j)| \geq \epsilon\} \ .$$

have low probability: $P(C_j) \leq \delta$ for all $j$.

# Empirical Risk Minimization

- We need to control the uniform deviations:

$$\sup_{\Phi \in \mathcal{F}} |F(\Phi) - F_n(\Phi)|$$

Suppose first that $\mathcal{F}$ is finite: $\mathcal{F} = \{\Phi_1, \ldots, \Phi_N\}$.

For each member $j$ of the family, Hoeffding says that the bad samples

$$C_j = \{z_i; i \leq n; |F(\Phi_j) - F_n(\Phi_j)| \geq \epsilon\} \ .$$

have low probability: $P(C_j) \leq \delta$ for all $j$.

Union bound: $P(\cup_{j \leq N} C_j) \leq \sum_j P(C_j) \leq N\delta$ .

# Empirical Risk Minimization

- We need to control the uniform deviations:

$$\sup_{\Phi \in \mathcal{F}} |F(\Phi) - F_n(\Phi)|$$

Suppose first that $\mathcal{F}$ is finite: $\mathcal{F} = \{\Phi_1, \ldots, \Phi_N\}$.

For each member $j$ of the family, Hoeffding says that the bad samples

$$C_j = \{z_i; i \leq n; |F(\Phi_j) - F_n(\Phi_j)| \geq \epsilon\} \ .$$

have low probability: $P(C_j) \leq \delta$ for all $j$.

Union bound: $P(\cup_{j \leq N} C_j) \leq \sum_j P(C_j) \leq N\delta$ .

It results that for all $\delta > 0$ whp $1 - \delta$

$$\forall \ \Phi \in \mathcal{F} \ , |F(\Phi) - F_n(\Phi)| \leq \sqrt{\frac{\log N - \log \delta}{2n}} \ .$$

# Empirical Risk Minimization

- Q: What if the family is infinite/ uncountable?

# Empirical Risk Minimization

- Q: What if the family is infinite/ uncountable?
- A: Vapnik-Chervonekis Theory projects an uncountable function class into the finite sample.
- We recover the previous bound via the so-called VC-dimension of the class. In the binary classification setting:

The VC dimension of a class $\mathcal{F}$ is the size of the largest set $\{z_1, \ldots, z_n\}$ such that $\mathcal{F}$ can generate any classification result.

# Empirical Risk Minimization

- Q: What if the family is infinite/ uncountable?
- A: Vapnik-Chervonekis Theory projects an uncountable function class into the finite sample.
- We recover the previous bound via the so-called VC-dimension of the class. In the binary classification setting:

The VC dimension of a class $\mathcal{F}$ is the size of the largest set $\{z_1, \ldots, z_n\}$ such that $\mathcal{F}$ can generate any classification result.

- For example, half-spaces in *d* dimensions have VC dimension *d*+*1*.

# Empirical Risk Minimization

- Q: What if the family is infinite/ uncountable?
- A: Vapnik-Chervonekis Theory projects an uncountable function class into the finite sample.
- We recover the previous bound via the so-called VC-dimension of the class. In the binary classification setting:

The VC dimension of a class $\mathcal{F}$ is the size of the largest set $\{z_1, \ldots, z_n\}$ such that $\mathcal{F}$ can generate any classification result.

If $\mathcal{F}$ has VC-dim $d$, with probability $1 - \delta$

$$\sup_{\Phi \in \mathcal{F}} |F(\Phi) - F_n(\Phi)| \lesssim \sqrt{\frac{d \log n}{n}} \ .$$

# Back to Estimation Error

- Using the previous bound we thus have

$$\begin{aligned} F(\Phi_{\mathcal{F},n}) - F(\Phi^*_{\mathcal{F}}) &= (F(\Phi_{\mathcal{F},n}) - F_n(\Phi_{\mathcal{F},n})) + (F_n(\Phi_{\mathcal{F},n}) - F_n(\Phi^*_{\mathcal{F}})) + (F_n(\Phi^*_{\mathcal{F}}) - F(\Phi^*_{\mathcal{F}})) \\ &\leq 2 \sup_{\Phi \in \mathcal{F}} |F(\Phi) - F_n(\Phi)| \\ &\lesssim \sqrt{\frac{d}{n}} \, . \end{aligned}$$

# Back to Estimation Error

- Using the previous bound we thus have

$$\begin{aligned} F(\Phi_{\mathcal{F},n}) - F(\Phi^*_{\mathcal{F}}) &= (F(\Phi_{\mathcal{F},n}) - F_n(\Phi_{\mathcal{F},n})) + (F_n(\Phi_{\mathcal{F},n}) - F_n(\Phi^*_{\mathcal{F}})) + (F_n(\Phi^*_{\mathcal{F}}) - F(\Phi^*_{\mathcal{F}})) \\ &\leq 2 \sup_{\Phi \in \mathcal{F}} |F(\Phi) - F_n(\Phi)| \\ &\lesssim \sqrt{\frac{d}{n}} \, . \end{aligned}$$

- This bound is pessimistic.
  - Faster rates are available e.g. $O\left(\left(\frac{d}{n} \log \frac{n}{d}\right)^{\alpha}\right)$ for $\alpha \in [1/2, 1]$.

# Back to Estimation Error

- Using the previous bound we thus have

$$\begin{aligned} F(\Phi_{\mathcal{F},n}) - F(\Phi^*_{\mathcal{F}}) &= (F(\Phi_{\mathcal{F},n}) - F_n(\Phi_{\mathcal{F},n})) + (F_n(\Phi_{\mathcal{F},n}) - F_n(\Phi^*_{\mathcal{F}})) + (F_n(\Phi^*_{\mathcal{F}}) - F(\Phi^*_{\mathcal{F}})) \\ &\leq 2 \sup_{\Phi \in \mathcal{F}} |F(\Phi) - F_n(\Phi)| \\ &\lesssim \sqrt{\frac{d}{n}} \, . \end{aligned}$$

- This bound is pessimistic.
  - Faster rates are available e.g. $O\left(\left(\frac{d}{n}\log\frac{n}{d}\right)^{\alpha}\right)$ for $\alpha \in [1/2, 1]$.

- Joint Estimation and Optimization error:

$$\mathcal{E}_{\text{est}} + \mathcal{E}_{\text{opt}} \simeq O\left(\left(\frac{d}{n}\log\frac{n}{d}\right)^{\alpha}\right) + \rho \, .$$

  - so in practice we should choose $\rho \simeq O\left(\left(\frac{d}{n}\log\frac{n}{d}\right)^{\alpha}\right)$

# Influence of the Approximation Class

- For large datasets, we may want to use larger, richer approximation classes (e.g. huge Convnets).
  - Complexity bounds are intrinsically pessimistic, not realistic.

- Regularization (structured risk minimization) entangles further the optimization and capacity of the signal class.

- Q: For a fixed signal class, how to make a principled choice of optimization algorithm?

# Iterative Optimization Algorithms

- [Bottou & Bousquet '08] study four main iterative algorithms in the large-scale learning regime:
  - Gradient Descent
  - Second Order Gradient Descent (i.e. Newton method)
  - Stochastic Gradient Descent (SGD)
  - Second Order Gradient Descent.

- Assumptions:
  - Signal class $\mathcal{F}$ is fixed,
  - linearly parametrized by $w \in \mathbb{R}^d$: $\Phi_w(x) = \langle \Phi(x), w \rangle$.

    loss functions $w \mapsto \ell(\Phi_w(x), y)$ convex and twice differentiable.

# Iterative Optimization

- Let $H$ and $G$ be respectively the Hessian and gradient covariance matrices at the empirical optimum $w_n = \arg\min_w F_n(\Phi_w)$:

$$H = \frac{\partial^2 F_n}{\partial w^2}(\Phi_{w_n}) = \frac{1}{n}\sum_i \frac{\partial^2 \ell(\Phi_w(x_i), y_i)}{\partial w^2}$$

$$G = \frac{1}{n}\sum_i \left(\frac{\partial \ell(\Phi_w(x_i), y_i)}{\partial w}\right)\left(\frac{\partial \ell(\Phi_w(x_i), y_i)}{\partial w}\right)^T .$$

# Iterative Optimization

- Let $H$ and $G$ be respectively the Hessian and gradient covariance matrices at the empirical optimum $w_n = \arg\min_w F_n(\Phi_w)$:

$$H = \frac{\partial^2 F_n}{\partial w^2}(\Phi_{w_n}) = \frac{1}{n}\sum_i \frac{\partial^2 \ell(\Phi_w(x_i), y_i)}{\partial w^2}$$

$$G = \frac{1}{n}\sum_i \left(\frac{\partial \ell(\Phi_w(x_i), y_i)}{\partial w}\right)\left(\frac{\partial \ell(\Phi_w(x_i), y_i)}{\partial w}\right)^T .$$

- Suppose that

$$\lambda(H) \subset [\lambda_{min}, \lambda_{max}] \ , \ \text{with } \lambda_{min} > 0$$

$$tr(GH^{-1}) \leq \nu \ .$$

- Condition number: $\kappa = \lambda_{max}/\lambda_{min}$.

# Gradient Descent (GD)

$$w_{t+1} = w_t - \eta \nabla_w F_n(\Phi_{w_t}) \ .$$

- When step size $\eta = \lambda_{max}^{-1}$, $O(\kappa \log(\rho^{-1}))$ iterations to reach accuracy $\rho$ (linear convergence).

| | Cost per iteration | Iterations to reach $\rho$ | Time to reach accuracy $\rho$ | Time to reach $F(\tilde{\Phi}_n) - F(\Phi^*_{\mathcal{F}} < \epsilon$ |
|---|---|---|---|---|
| GD | $O(nd)$ | $O(\kappa \log \rho^{-1})$ | $O(nd\kappa \log \rho^{-1})$ | $O\left(d^2 \kappa \epsilon^{-1/\alpha} \log^2(\epsilon^{-1})\right)$ |

# Second Order Gradient Descent

$$w_{t+1} = w_t - H^{-1}\nabla_w F_n(\Phi_{w_t}) \ , \ H^{-1} \text{ known in advance.}$$

| | Cost per iteration | Iterations to reach $\rho$ | Time to reach accuracy $\rho$ | Time to reach $F(\tilde{\Phi}_n) - F(\Phi^*_{\mathcal{F}} < \epsilon$ |
|---|---|---|---|---|
| GD | $O(nd)$ | $O(\kappa \log \rho^{-1})$ | $O(nd\kappa \log \rho^{-1})$ | $O\left(d^2\kappa\epsilon^{-1/\alpha}\log^2(\epsilon^{-1})\right)$ |
| 2GD | $O((n+d)d)$ | $O(\log\log \rho^{-1})$ | $O((n+d)d \log\log \rho^{-1})$ | $O\left(d^2\epsilon^{-1/\alpha}\log\log(\epsilon^{-1})\log(\epsilon^{-1})\right)$ |

- Optimization speed is much faster
- The problem does not depend on condition number.

# Stochastic Gradient Descent (SGD)

- At each $t$, we draw random $z_t$ from training set.

$$w_{t+1} = w_t - \frac{\eta}{t} \nabla_w f(\Phi_w(z_t)) \ .$$

- With $\eta = \lambda_{min}^{-1}$, we have $\|w_t - w_n\| = O(1/\sqrt{t})$.

| | Cost per iteration | Iterations to reach $\rho$ | Time to reach accuracy $\rho$ | Time to reach $F(\tilde{\Phi}_n) - F(\Phi^*_{\mathcal{F}}) < \epsilon$ |
|---|---|---|---|---|
| GD | $O(nd)$ | $O(\kappa \log \rho^{-1})$ | $O(nd\kappa \log \rho^{-1})$ | $O\left(d^2 \kappa \epsilon^{-1/\alpha} \log^2(\epsilon^{-1})\right)$ |
| 2GD | $O((n+d)d)$ | $O(\log \log \rho^{-1})$ | $O((n+d)d \log \log \rho^{-1})$ | $O\left(d^2 \epsilon^{-1/\alpha} \log \log(\epsilon^{-1}) \log(\epsilon^{-1})\right)$ |
| SGD | $O(d)$ | $\nu\kappa^2\rho^{-1} + o(\rho^{-1})$ | $O(\frac{d\nu\kappa^2}{\rho})$ | $O(\frac{d\nu\kappa^2}{\epsilon})$ |

- Optimization speed is much worse than GD.
- However, learning speed is better.

# Second Order Stochastic Gradient Descent (2SGD)

- At each $t$, we draw random $z_t$ from training set.

$$w_{t+1} = w_t - \frac{H^{-1}}{t} \nabla_w f(\Phi_w(z_t)) \ .$$

| | Cost per iteration | Iterations to reach $\rho$ | Time to reach accuracy $\rho$ | Time to reach $F(\tilde{\Phi}_n) - F(\Phi^*_{\mathcal{F}}) < \epsilon$ |
|---|---|---|---|---|
| GD | $O(nd)$ | $O(\kappa \log \rho^{-1})$ | $O(nd\kappa \log \rho^{-1})$ | $O\left(d^2 \kappa \epsilon^{-1/\alpha} \log^2(\epsilon^{-1})\right)$ |
| 2GD | $O((n+d)d)$ | $O(\log \log \rho^{-1})$ | $O((n+d)d \log \log \rho^{-1})$ | $O\left(d^2 \epsilon^{-1/\alpha} \log \log(\epsilon^{-1}) \log(\epsilon^{-1})\right)$ |
| SGD | $O(d)$ | $\nu\kappa^2\rho^{-1} + o(\rho^{-1})$ | $O(\frac{d\nu\kappa^2}{\rho})$ | $O(\frac{d\nu\kappa^2}{\epsilon})$ |
| 2SGD | $O(d^2)$ | $\nu\rho^{-1} + o(\rho^{-1})$ | $O(\frac{d^2\nu}{\rho})$ | $O(\frac{d^2\nu}{\epsilon})$ |

- Iteration is more expensive, but less iterations.
- Constants are affected.

# Accelerated Gradient Descent

- Gradient descent has rate 1/T after T steps. Lower bound is 1/T^2 amongst first order methods.
- Q: How to improve using a first order method?

# Accelerated Gradient Descent

- Gradient descent has rate 1/T after T steps. Lower bound is 1/T^2 amongst first order methods.
- Q: How to improve using a first order method?
- Use a momentum term (Nesterov,'83):

$$\lambda_0 = 0 \ , \ \lambda_t = \frac{1 + \sqrt{1 + 4\lambda_{t-1}^2}}{2} \ , \ \gamma_t = \frac{1 - \lambda_t}{\lambda_{t+1}} \ .$$

$$y_{t+1} = x_t - \frac{1}{\beta} \nabla f(x_t) \ ,$$

$$x_{t+1} = (1 - \gamma_t) y_{t+1} + \gamma_t y_t \ .$$

# Geometric Interpretation

# Geometric Interpretation