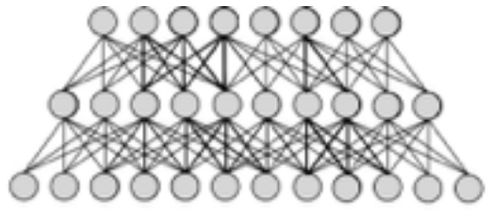
Kernels for Deep Learning - With and without tricks

Amir Globerson (Tel Aviv University)
Uri Heinemann (HUJI), Roi Livni (HUJI, Princeton), Gal Elidan (HUJI, Google), Elad Eban (Google), Yoav Wald (HUJI)

Deep Learning

- A class of prediction models.



- Excellent empirical performance in some domains
- Training is non convex

- Sometimes a problem, sometimes not
- Great opportunity for theory!

This Talk

- Consider a larger class of models
- Show that it corresponds to SVM learning with a particular kernel
- Learn by:
  - Closed form expression for kernel
  - Online sampling
- Multifocal attention
- Robust conditional probability estimation.

Binary Classification

- Map feature vectors **x** to binary label y
- Consider a class of functions $f(\boldsymbol{x}, \boldsymbol{w})$
- Classify using: $y = \Theta\left[f(\boldsymbol{x}, \boldsymbol{w})\right]$
- Learn $\boldsymbol{w}$ from labeled data $\{\boldsymbol{x}_i, y_i\}_{i=1}^{M}$
- Minimize loss: $\frac{1}{M}\sum_i \ell(\boldsymbol{x}_i, y_i, \boldsymbol{w}) + C\|\boldsymbol{w}\|_2^2$
- Where ℓ is a loss function (e.g., hinge, logistic)

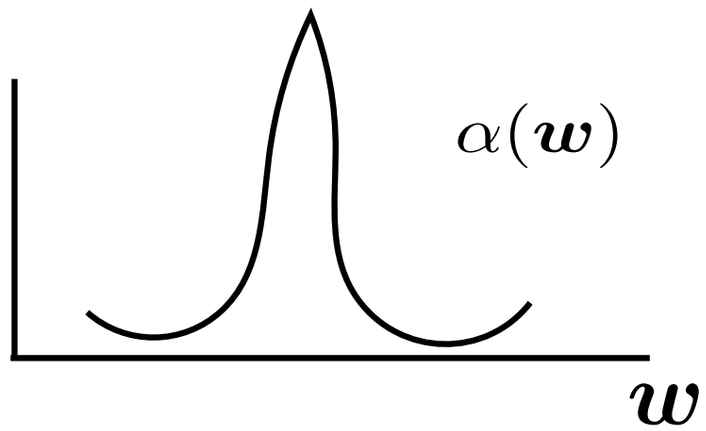
Optimization Challenges

- Want to minimize: $\frac{1}{M}\sum_i \ell(\boldsymbol{x}_i, y_i, \boldsymbol{w}) + C\|\boldsymbol{w}\|_2^2$
- What if it's a non-convex function of $\boldsymbol{w}$?
- Can apply SGD, sometimes with good results.
- But would be nice to have something with more guarantees.

Improper Learning

- Consider an extended hypothesis class, parameterised by a function $\alpha(\boldsymbol{w})$

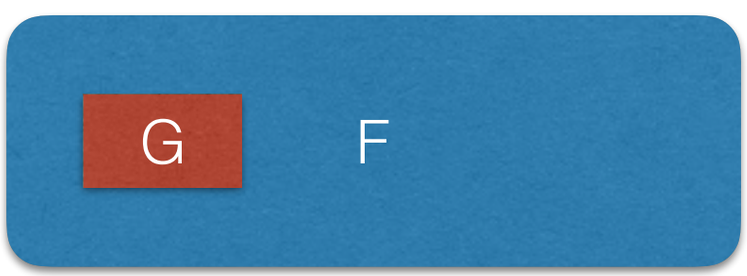
$$g(\boldsymbol{x}, \alpha) = \int \alpha(\boldsymbol{w}) f(\boldsymbol{x}, \boldsymbol{w}) d\boldsymbol{w}$$



- If $\alpha(\boldsymbol{w}) = \delta(\boldsymbol{w} - \boldsymbol{w}_0)$ then:

$$g(\boldsymbol{x}, \alpha) = \int \alpha(\boldsymbol{w} - \boldsymbol{w}_0) f(\boldsymbol{x}, \boldsymbol{w}) d\boldsymbol{w} = f(\boldsymbol{x}, \boldsymbol{w}_0)$$

- We have a larger class



A linear classifier

- Our classifier is a linear function of $\alpha(\boldsymbol{w})$

$$g(\boldsymbol{x}, \alpha) = \int \alpha(\boldsymbol{w}) f(\boldsymbol{x}, \boldsymbol{w}) d\boldsymbol{w} = \langle \alpha, f(\boldsymbol{x}, \cdot) \rangle$$

- $\alpha(\boldsymbol{w})$ is a function. How do we optimize over it?

- Recall the objective: $\sum_i [1 - y_i \langle \alpha, f(\boldsymbol{x}_i, \cdot) \rangle]_+ + C\|\alpha\|_2^2$

- The subgradient is: $\dfrac{\partial \ell}{\alpha(\boldsymbol{w})} = -y_i f(x_i, \boldsymbol{w}) + C\alpha(\boldsymbol{w})$

- So SGD will lead to: $\alpha(\boldsymbol{w}) = \sum_i \gamma_i f(\boldsymbol{x}_i, \boldsymbol{w})$

The Kernel Trick

- $\alpha(\boldsymbol{w})$ can be expressed as: $\alpha(\boldsymbol{w}) = \sum_i \gamma_i f(\boldsymbol{x}_i, \boldsymbol{w})$
- Still a function of **w**, but what we care about is:

$$g(\boldsymbol{x}, \alpha) = \langle \alpha, f(\boldsymbol{x}, \cdot) \rangle = \sum_i \gamma_i \langle f(\boldsymbol{x}_i, \cdot), f(\boldsymbol{x}, \cdot) \rangle$$

- The kernel:

$$K(\boldsymbol{x}, \boldsymbol{x}') = \langle f(\boldsymbol{x}, \cdot), f(\boldsymbol{x}', \cdot) \rangle = \int f(\boldsymbol{x}', \boldsymbol{w}) f(\boldsymbol{x}, \boldsymbol{w}) d\boldsymbol{w}$$

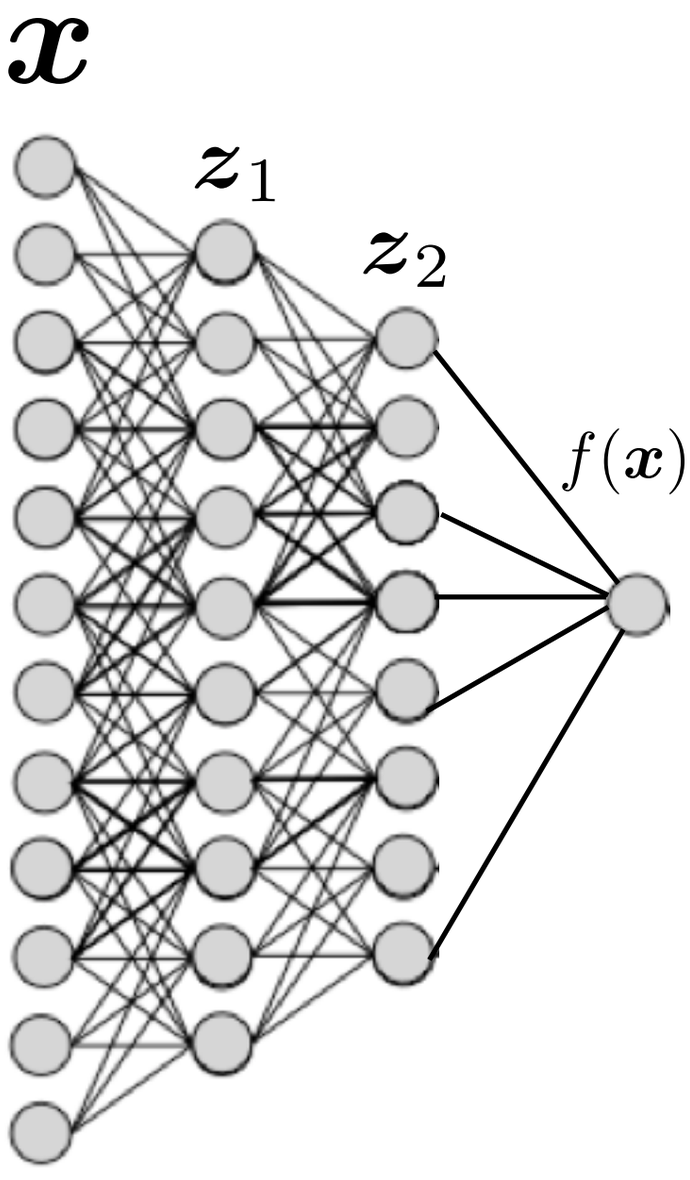
- **If we can calculate it, learning becomes tractable and convex!**

A Kernel For Neural Nets

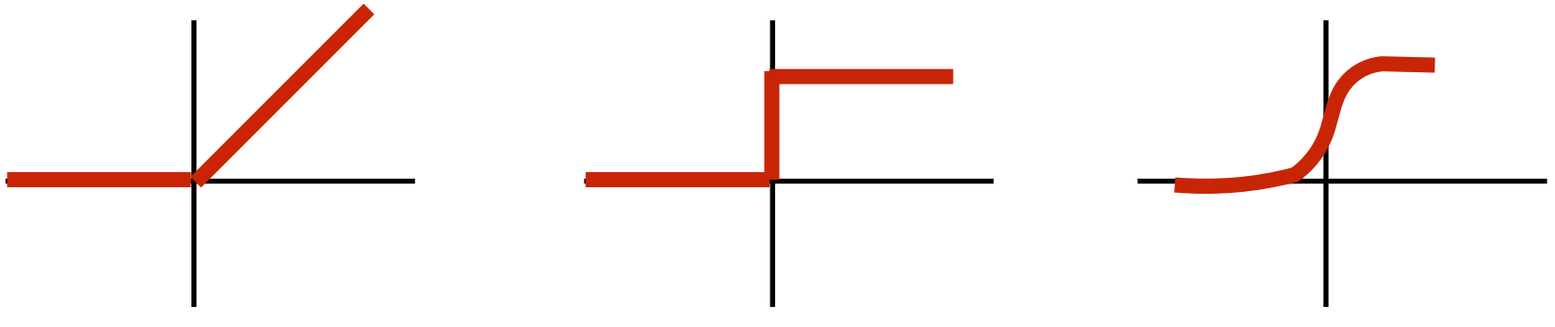
- A function $f(\boldsymbol{x}, \boldsymbol{w})$
- Defined recursively:

$$\boldsymbol{z}_k = h(W_k \boldsymbol{z}_{k-1})$$
$$\boldsymbol{z}_0 = \boldsymbol{x}$$

$$f(\boldsymbol{x}, \boldsymbol{w}) = \boldsymbol{w}_L \cdot \boldsymbol{z}_{L-1}$$



- h is an activation function



Calculating the Kernel

- We want:

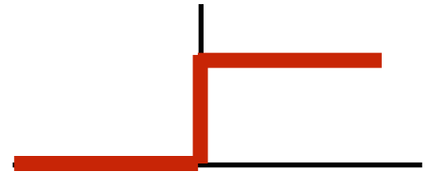
$$K(\boldsymbol{x}, \boldsymbol{x}') = \langle f(\boldsymbol{x}, \cdot), f(\boldsymbol{x}', \cdot) \rangle = \int f(\boldsymbol{x}', \boldsymbol{w}) f(\boldsymbol{x}, \boldsymbol{w}) d\boldsymbol{w}$$

- Need to integrate over all weights

- Surprisingly, it's possible under some conditions:
  - Activation function is the threshold. 
  - All to all connections

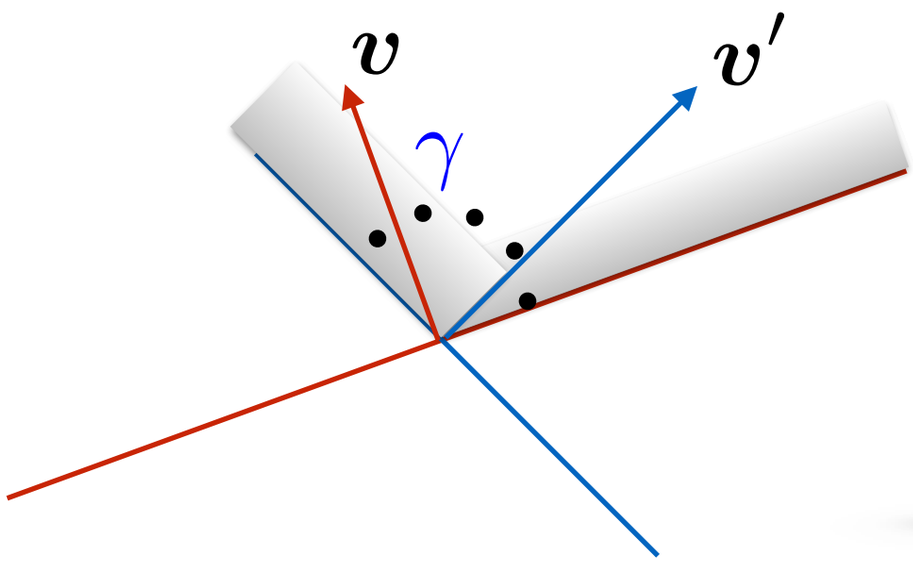
A key integral

- Denote the threshold function by $\Theta(\boldsymbol{x})$



- We'll make repeated use of the integral:

$$H(\boldsymbol{v},\boldsymbol{v}') = \int \Theta(\boldsymbol{w}\cdot\boldsymbol{v})\Theta(\boldsymbol{w}\cdot\boldsymbol{v}')d\boldsymbol{w} = \frac{1}{2} - \frac{1}{2\pi}\arccos\frac{\boldsymbol{v}\cdot\boldsymbol{v}'}{\|\boldsymbol{v}\|_2\|\boldsymbol{v}'\|_2}$$



$$= \frac{\gamma}{2\pi}$$

Activation turns into volume!

[Cho and Saul, NIPS 2009]

A Key Integral

- H depends only on norms and dot product

$$H(\boldsymbol{v}, \boldsymbol{v}') = \int \Theta(\boldsymbol{w} \cdot \boldsymbol{v}) \Theta(\boldsymbol{w} \cdot \boldsymbol{v}') d\boldsymbol{w} = \frac{1}{2} - \frac{1}{2\pi} \arccos \frac{\boldsymbol{v} \cdot \boldsymbol{v}'}{\|\boldsymbol{v}\|_2 \|\boldsymbol{v}'\|}$$

- For integral $\boldsymbol{v}, \boldsymbol{v}'$ the dot products and norms will also be integral.

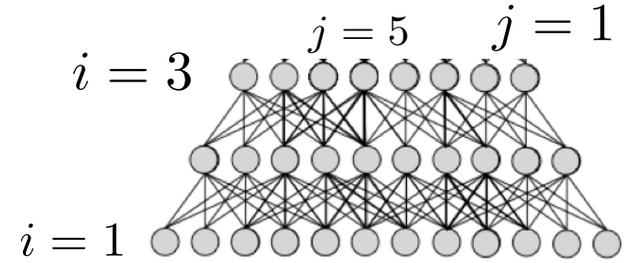
- Define: $J(k, l, m) = \frac{1}{2} - \frac{1}{2\pi} \arccos \frac{m}{\sqrt{kl}}$

Deep Improper Kernel

- Our integral is (for three layers):

$$K(\boldsymbol{x}, \boldsymbol{x}') = \int \left(\boldsymbol{w}_3^T \Theta(W_2 \Theta(W_1 \boldsymbol{x})\right) \left(\boldsymbol{w}_3^T \Theta(W_2 \Theta(W_1 \boldsymbol{x}')\right) d\boldsymbol{w}_3 dW_2 dW_1$$

- Recursively calculate: $V_{i,j}(\boldsymbol{x}, \boldsymbol{x}')$



$$V_{1,q}(\boldsymbol{x}, \boldsymbol{x}') = H^{N_1 - q}(\boldsymbol{x}, \boldsymbol{x}')(0.5 - H(\boldsymbol{x}, \boldsymbol{x}'))^q.$$

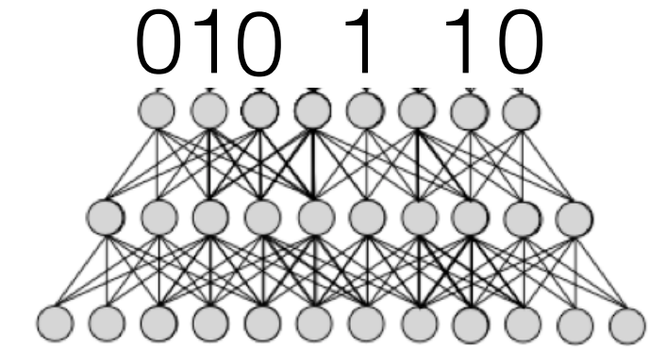
$$V_{n,q}(\boldsymbol{x}, \boldsymbol{x}') = \sum_{s=1}^{N_{n-1}} \sum_{s'=1}^{N_{n-1}} \sum_{k=[s+s'-N_{n-1}]_+}^{\min\{s,s'\}} r(s, s', k) V_{n-1, s+s'-2k}(\boldsymbol{x}, \boldsymbol{x}')$$

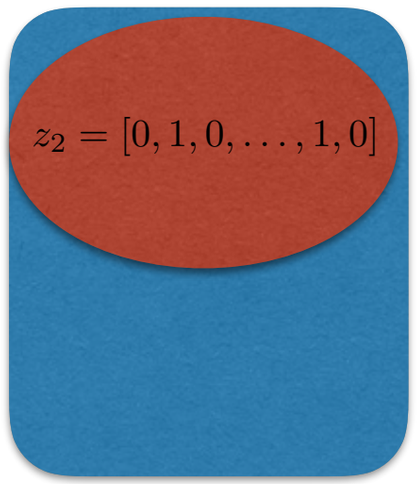
$$K(\boldsymbol{x}, \boldsymbol{x}') = V_{M,0}(\boldsymbol{x}, \boldsymbol{x}')$$

See paper for definition

Key Ideas

- Discrete outputs
- Integrate over weights that product these outputs
- Volume integral that looks a lot like $H(\boldsymbol{v}, \boldsymbol{v}') = \int \Theta(\boldsymbol{w} \cdot \boldsymbol{v}) \Theta(\boldsymbol{w} \cdot \boldsymbol{v}') d\boldsymbol{w}$
- Use many symmetries, as a result of independence of weight distribution.

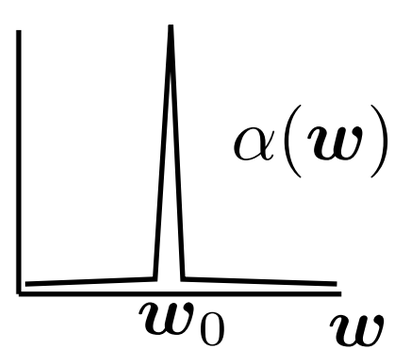




W

Generalization Bounds

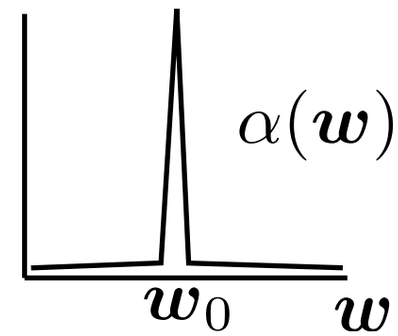
- Large hypothesis space. Generalization?
- Generalization function of $\|\alpha\|_2 \leq B$
- Suppose data generated from network $\boldsymbol{w}_0$
- Can be achieved with $\alpha = \delta(\boldsymbol{w} - \boldsymbol{w}_0)$

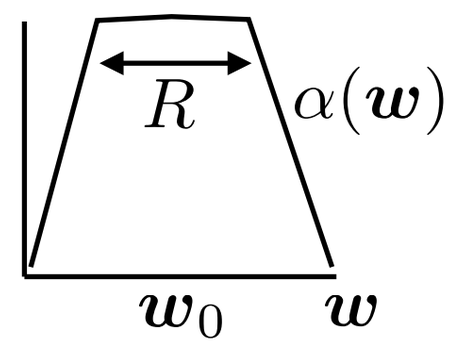


Unbounded Norm!

Stable Solutions

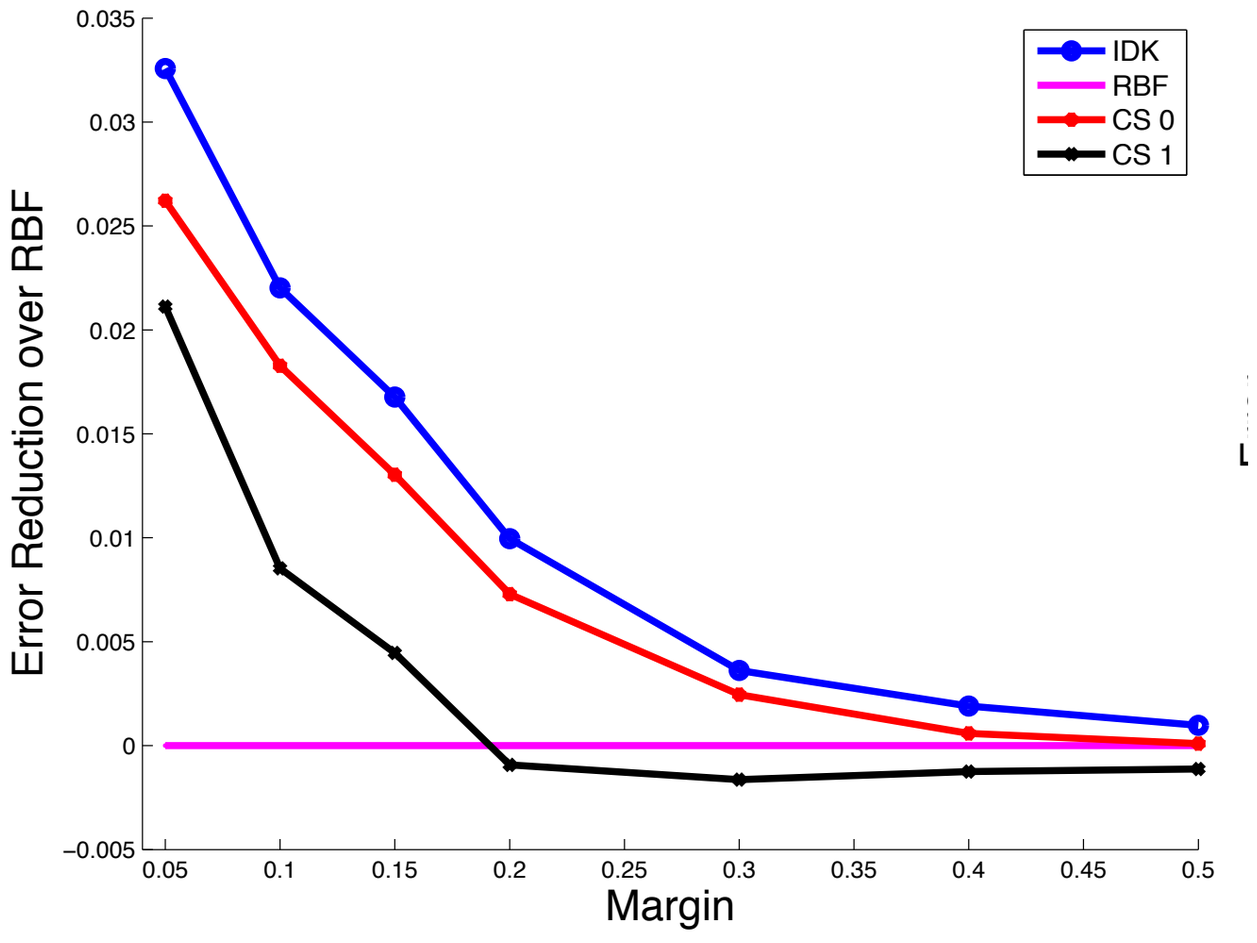
- Often many networks that implement the same rule.
- Sample complexity is then: $O\left(\frac{R^{-d}}{\epsilon^2}\right)$
- Can also give dimension independent bounds using margin.
- Exponential sample complexity expected, since problem is NP hard (Daniely et al., 2015)





Toy Experiments

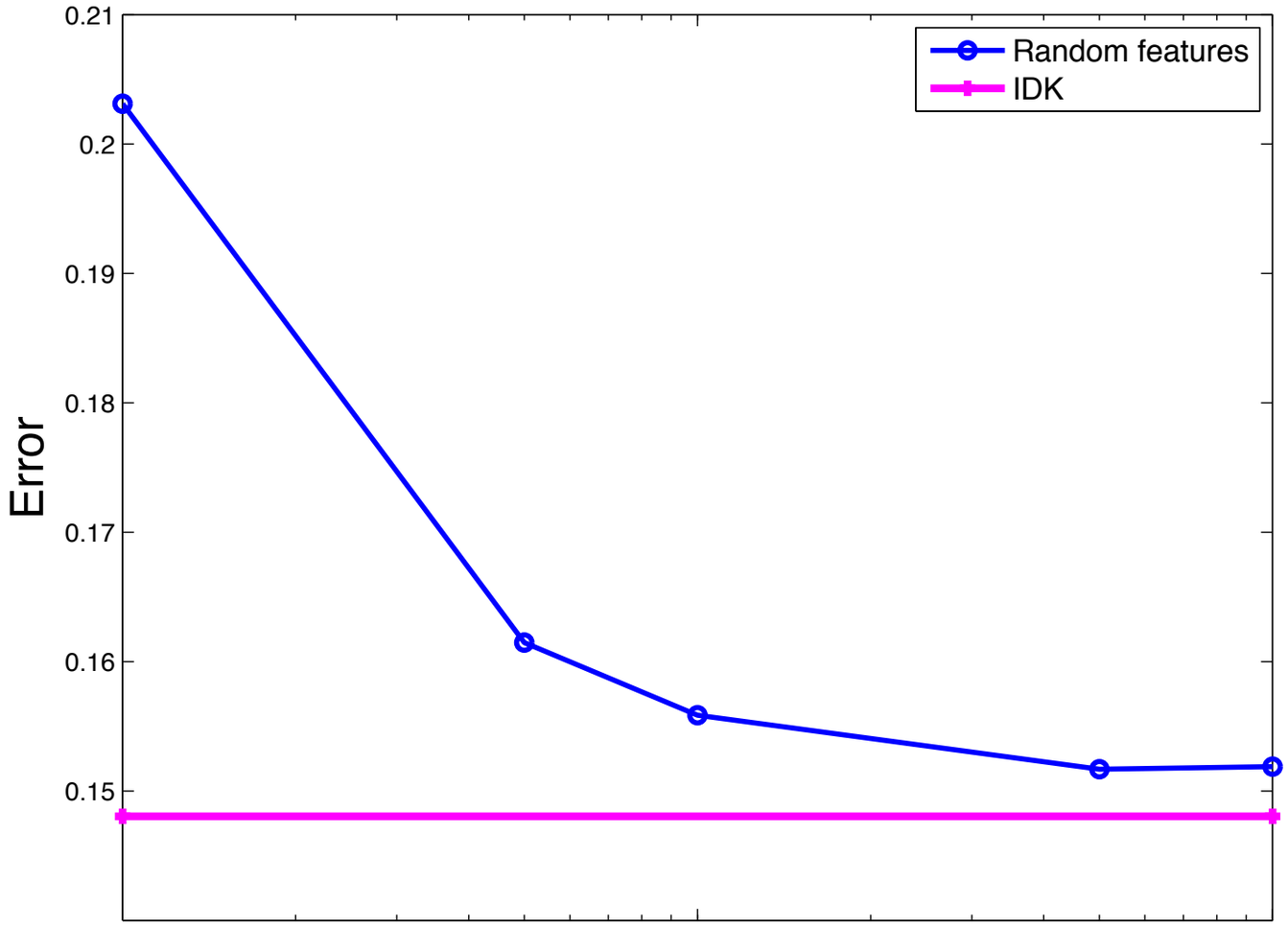
- Compare our “improper deep kernel” to RBF kernels, and two “infinite layer” kernels.



Random Features

- What if we just sample random models

$$K(\boldsymbol{x}, \boldsymbol{x}') = \int f(\boldsymbol{x}, \boldsymbol{w}) f(\boldsymbol{x}', \boldsymbol{w}) d\boldsymbol{w} \approx \frac{1}{M} \sum_{i=1}^{M} f(\boldsymbol{x}, \boldsymbol{w}_i) f(\boldsymbol{x}', \boldsymbol{w}_i)$$



Vision Experiments

- Evaluate on two image datasets: CIFAR and STL-10
- Compared several kernel and non-kernel based methods

	IDK	RBF	CS0	CS1	SPN	CKN
CIFAR-10	81.8	81.8	81.63	82.49	83.96	82.18
STL-10	62.6	61.7	62.3	52	62.3	62.32

A Bayesian Interpretation

- Recall: $K(\boldsymbol{x}, \boldsymbol{x}') = \int f(\boldsymbol{x}, \boldsymbol{w}) f(\boldsymbol{x}', \boldsymbol{w}) \mu(\boldsymbol{w}) d\boldsymbol{w}$

- Assume f is a conditional density:

$$K(\boldsymbol{x}, \boldsymbol{x}') = \int f(\boldsymbol{x}|\boldsymbol{w}) f(\boldsymbol{x}'|\boldsymbol{w}) \mu(\boldsymbol{w}) d\boldsymbol{w}$$

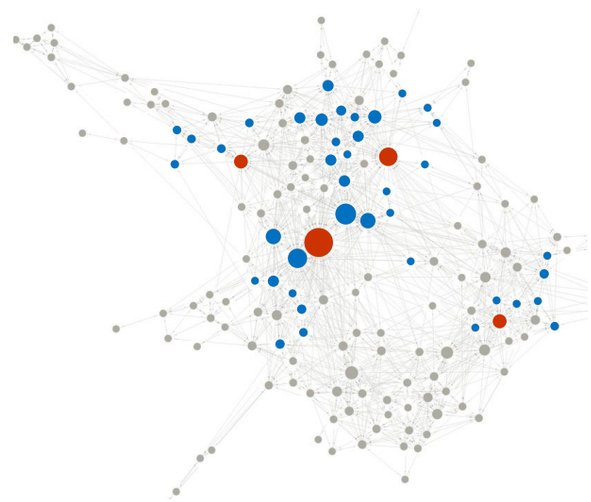
- Then K is the probability that $\boldsymbol{x}$ and $\boldsymbol{x}'$ were generated assuming they are IID given $\boldsymbol{w}$.

Kernels in Deep Learning

- Early work by Neal on Bayesian neural networks
- Continuous Neural Networks (Le Roux and Bengio, 2007),Kernels for Deep Learning (Cho and Saul, 2009), Convex NN (Bach 2015): consider continuum of hidden units.
- Recursively constructed kernels (Daniely, Frostig and Singer, 2016). Relations to initialization.
- Kernels and invariance - Mairal et al. Learn neural nets to approximate an invariant kernel.

Diffusion Estimation

- Consider the problem of predicting the spread of activity in a network



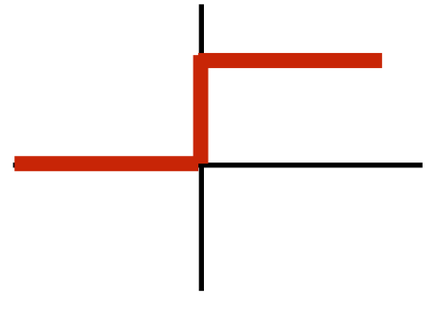
- Many generative models exist (independent cascade, threshold etc).

- Can use an improper approach to integrate over these.

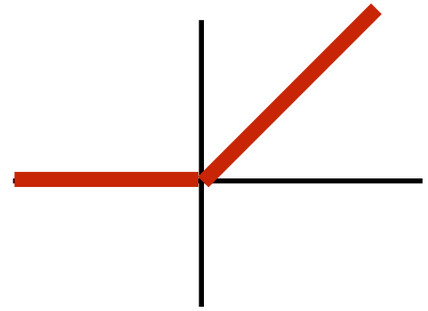
- Empirically successful (Rosenfeld, Nitzan, G., WSDM 16).

No tricks…

- Our closed form assumed: threshold activation, full connectivity, and no parameter sharing



- Real networks use: Relu activation convolutions and pooling



- We can do some of these in some restricted cases, but general case seems hard

- What can we do?

Sampling

- Recall: $K(\boldsymbol{x}, \boldsymbol{x}') = \int f(\boldsymbol{x}', \boldsymbol{w}) f(\boldsymbol{x}, \boldsymbol{w}) \mu(\boldsymbol{w}) d\boldsymbol{w}$
- Recast as: $K(\boldsymbol{x}, \boldsymbol{x}') = \mathbb{E}\left[f(\boldsymbol{x}, W) f(\boldsymbol{x}', W)\right]$
- 1st try: sample as many W as you need to approximate the kernel
- But need to do this every sample.
- Our end goal is learning. How many samples do we need for that?

Optimal Sampling

- Can we achieve the same guarantees of closed form kernel, with sampling?

- Yes (for quadratic loss)!

- With closed form kernel need $O\left(\frac{1}{\epsilon^2}\right)$ examples to reach ϵ accurate classifier

- We achieve the same sample complexity by sampling $O\left(\frac{1}{\epsilon^4}\right)$ weights.

Key Ideas

- Objective is: $$\sum \left(y_i - \int_{\boldsymbol{w}} f(\boldsymbol{x}_i, \boldsymbol{w}) \alpha(\boldsymbol{w}) \mu(\boldsymbol{w}) d\boldsymbol{w} \right)^2$$
- Gradient is: $$\left(y_i - \int_{\boldsymbol{w}} f(\boldsymbol{x}_i, \boldsymbol{w}) \alpha(\boldsymbol{w}) \mu(\boldsymbol{w}) d\boldsymbol{w} \right) f(\boldsymbol{x}_i, \boldsymbol{w})$$

- Assume at time t: $$\alpha(\boldsymbol{w}) = \sum_{j=0}^{t} \gamma_j f(\boldsymbol{x}_j, \boldsymbol{w})$$
- Then: $$\int f(\boldsymbol{x}_i, \boldsymbol{w}) \alpha(\boldsymbol{w}) \mu(\boldsymbol{w}) d\boldsymbol{w} = \sum_{j=0}^{t} \gamma_j \int f(\boldsymbol{x}_j, \boldsymbol{w}) f(\boldsymbol{x}_i, \boldsymbol{w}) \alpha(\boldsymbol{w}) \mu(\boldsymbol{w}) d\boldsymbol{w}$$

sample

Unbiased!

Double Sampling

$$\int f(\boldsymbol{x}_i, \boldsymbol{w})\alpha(\boldsymbol{w})\mu(\boldsymbol{w})d\boldsymbol{w} = \sum_{j=0}^{t} \gamma_j \int f(\boldsymbol{x}_j, \boldsymbol{w}) f(\boldsymbol{x}_i, \boldsymbol{w})\alpha(\boldsymbol{w})\mu(\boldsymbol{w})d\boldsymbol{w}$$

- Sample $j_1, \ldots, j_m$ proportional to $|\gamma_j|$
- Sample $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_m$ from $\mu(\boldsymbol{w})$
- Estimate: $\frac{1}{m} \sum_{k=1}^{m} \gamma_{j_k} f(\boldsymbol{x}_{j_k}, \boldsymbol{w}_k) f(\boldsymbol{x}_i, \boldsymbol{w}_k)$

- Unbiased estimate.

Shrinking Gradient Algorithm

- Similar to SGD, but with:
  - An unbiased estimate of the gradient as above
  - Shrinking the weights γ if estimate gets too large
- Detailed analysis (using online stochastic optimization tools) provides the sample complexity bound.

Related Work

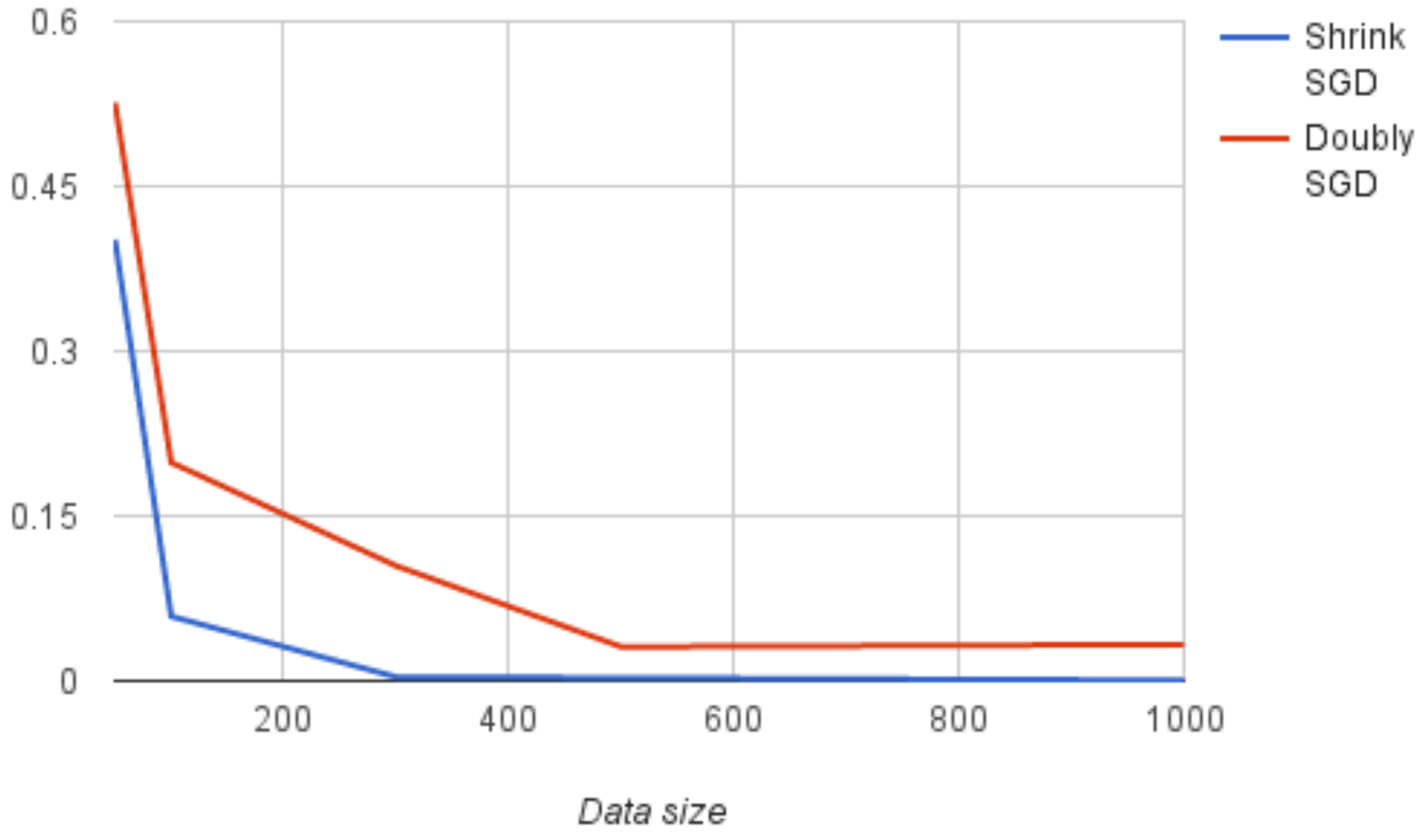
- Rahimi and Recht (2007,2008) showed that random features could approximate a kernel.

- They obtain sample complexity bounds for a more restricted hypothesis class than ours.

- Dai et al. (2014) introduce a doubly stochastic algorithm with $O(\frac{1}{\epsilon^4})$ sample complexity.

Example

- On synthetic data from Dai et al.



Multi Focal Attention

- Attention models have become widespread in machine vision

- Commonly one focus of attention

- Easy to optimize with soft-max

- What is the multi-focal extension?

- How is soft-max extended?

Entity Linking

Caroline was the last, best hope for the family, which has had members in the Senate since **Jack** was elected in 1952. Following him: **RFK** and **Ted** joined the Senate, both of them with presidential dreams that didn't materialize (because of **Bobby's** assassination and Teddy's being not as beloved as Bobby and Jack).

Multi-Focal Attention

- In ACL 16, we provide a soft multi-focal attention mechanism.

- Nicely generalizes soft-max.

Sentence with mention	Entity	Attn. focus mentions
Caroline has dropped her name from consideration for the seat that Hillary has left vacant.	base: Caroline (given name) attn: Caroline Kennedy	Democratic Party New York Robert Kennedy
Chris Johnson had just 13 tackles last season, and the Raiders currently have have 11 defensive backs on their roster.	base: Chris Johnson (running back) attn: Chris Johnson (cornerback)	Oakland Raiders Oakland Raiders Oakland Raiders

Reasoning About Conditional Probabilities

- Given features X, what is $p(Y|X_1, \ldots, X_n)$
- Typically hard to estimate directly
- Models (e.g., logistic regression) often used, but hard to relate to real probabilities
- Given statistics as we have, we can reason about range of values for conditionals

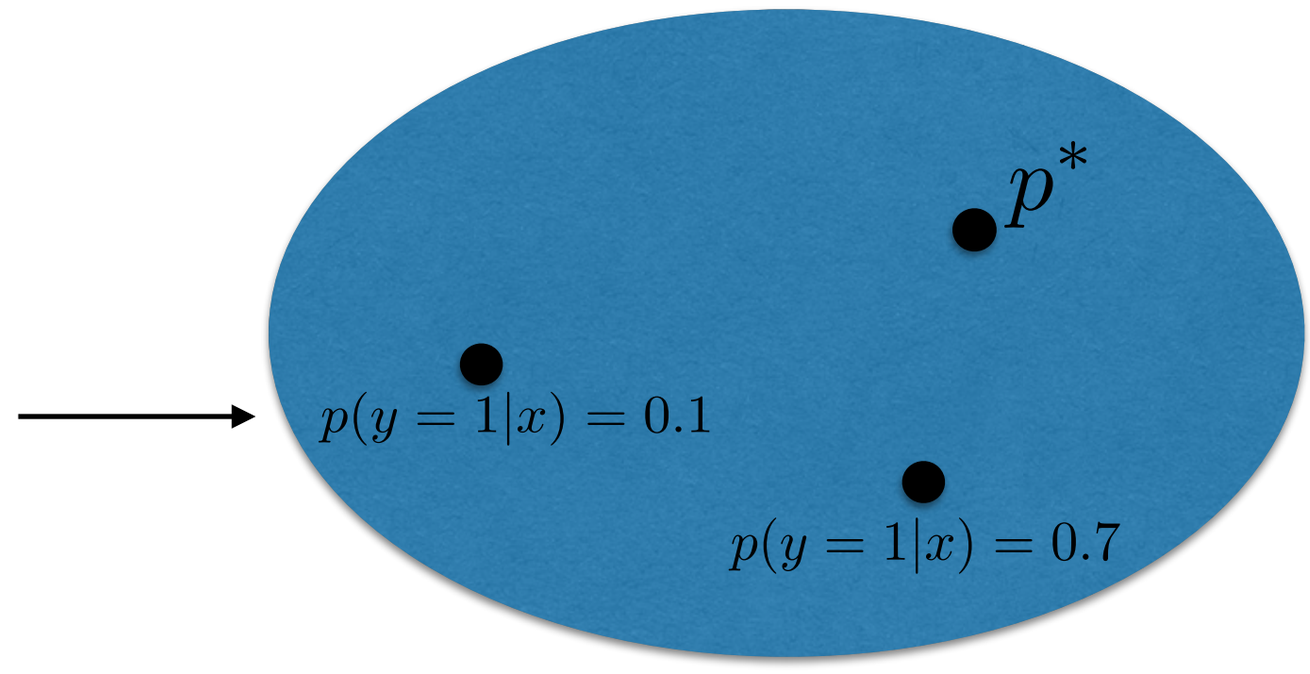
Marginals as Constraints

- Assume we know the following marginals:

$$\mu_{ij}(x_i, x_j, y) = Pr(X_i = x_i, X_j = x_j, Y = y)$$

- These inform us about the real distribution p^*.

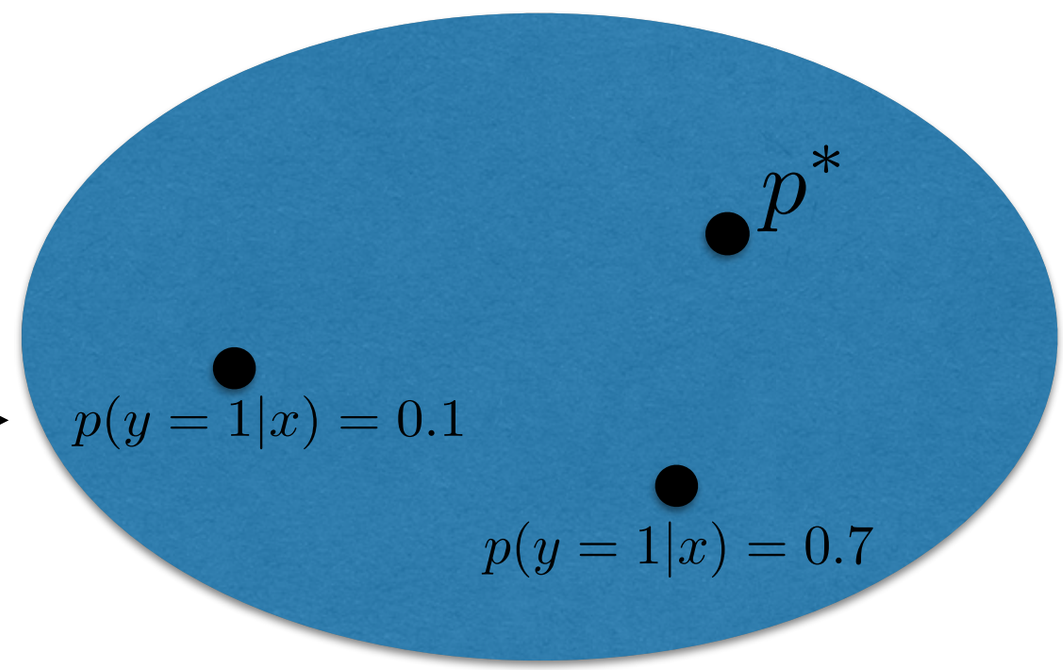
Distributions that agree with μ



Bounds on Conditionals

- So what can we say about $p(y = 1|x)$?
- It has a minimum and maximum value
- When evidence for y=1 is strong, we expect a large minimum
- Can we calculate it?

Distributions that agree with μ



Bounds on Conditionals

- We study bounds on probabilities given:

$$\mu_{ij}(x_i, x_j, y) = Pr(X_i = x_i, X_j = x_j, Y = y)$$

- We can give results for: $\min p(x, y), \max p(x, y), \min p(y|x)$

- Useful for semi-supervised learning, where goal is to label unlabelled data with high confidence.

Min Conditional

- Define:

$$f(x,y) = \min p(x,y) = \sum_i (1-d_i)\mu_i(x_i,y) + \sum_{ij} \mu_{ij}(x_i,x_j,y)$$

- Then we can show that if (i,j) define a tree then:

$$\min p(y|x) = \frac{f(x,y)}{f(x,y) + \sum_y \min_{ij} \mu_{ij}(x_i,x_j,y)}$$

- As expected: Large for y with close to deterministic relations to x, and other y less so.

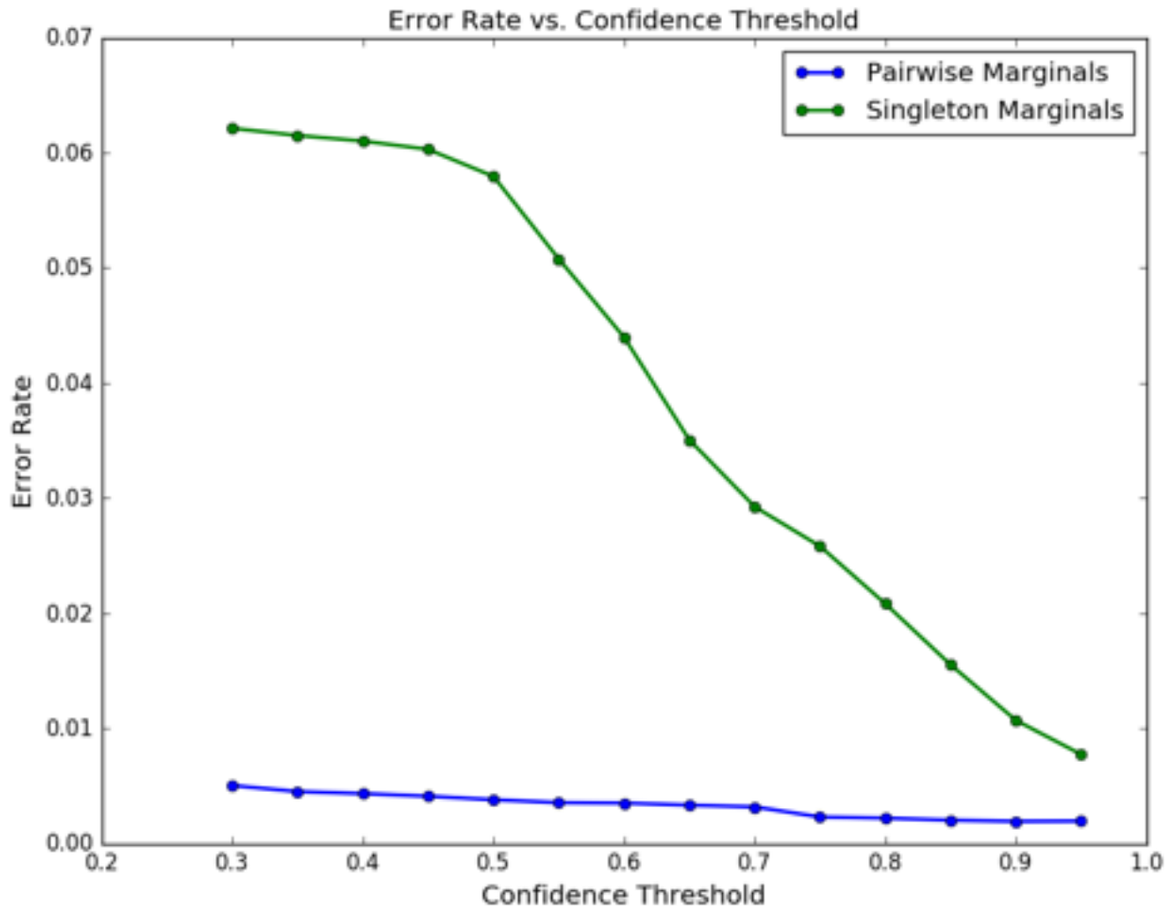
Part of Speech Tagging

- Assign POS tags to sentence:

N V D N **x**

John hit the ball **y**

- Use pairwise statistics on words and word-tag pairs.



Summary

- Discussed:
  - Improper learning with kernels
  - Closed and non-closed form approaches
  - Multi Focal Attention
  - Robust reasoning about conditionals