TECHNION
Israel Institute
of Technology

The Henry and Marilyn Taub
Faculty of Computer Science

Coin Flipping Neural Networks

Yuval Sieradzki, Nitzan Hodos, Gal Yehuda
Advised by Assaf Schuster

Technion Israel Institute of Technology

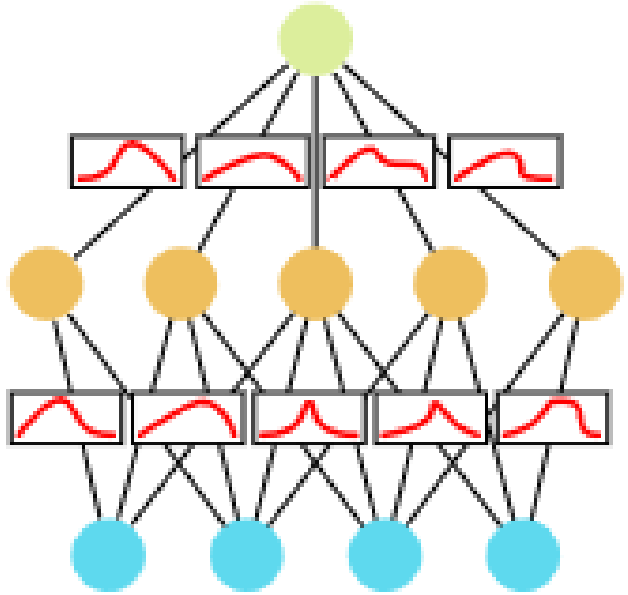
Definition

Coin-Flipping NNs are ***stochastic NNs***,

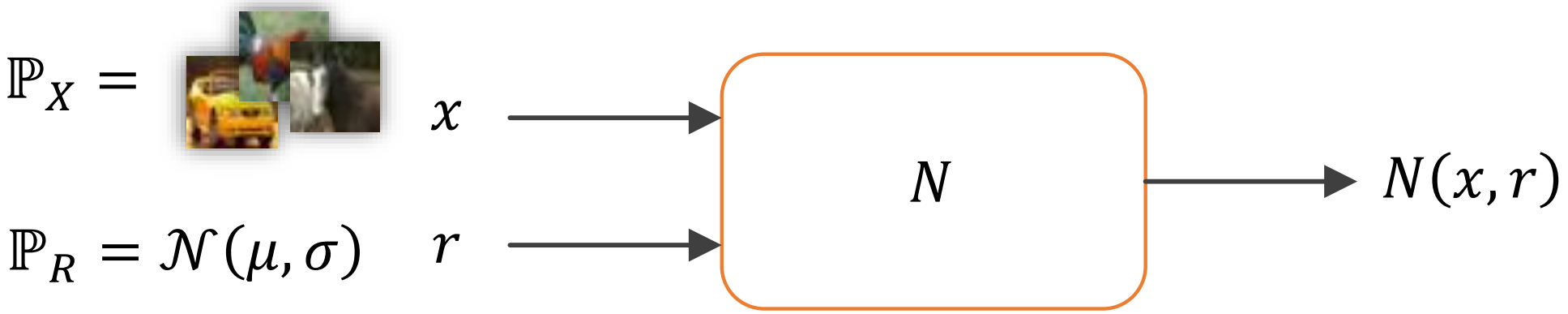
where we are allowed to ***amplify*** their outputs during inference.

Stochastic NNs

- Output is a Random Variable
  - Random weights, activations…
- Bayesian NN, Variational AE, etc.



- In general: $N(x, r), x \sim \mathbb{P}_X, r \sim \mathbb{P}_R$

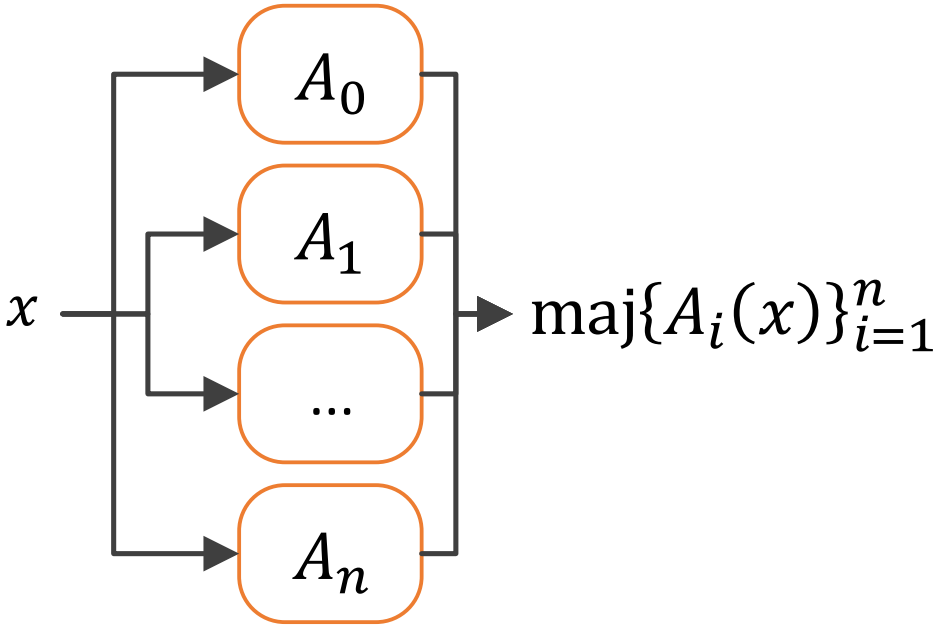


Amplification

- Randomized A solves decision problem f
  - $\Pr[A(x) = f(x)] = \frac{1}{2} + \varepsilon$

- Take multiple samples from $A(x)$
- Calculate majority

- $\Pr[\text{maj}\{A_i(x)\}_{i=1}^{n} = f(x)] \geq 1 - e^{-2\varepsilon^2 n}$



- Improved complexity in practice
  - Convex volume estimation, polynomial equality test…

CFNN

- *Can amplification add power to NNs? –* ***yes!***
- Reduce size, improve performance
- Our focus – classification

- Given $f(x): X \to \{1,2,\dots,C\}$, define:

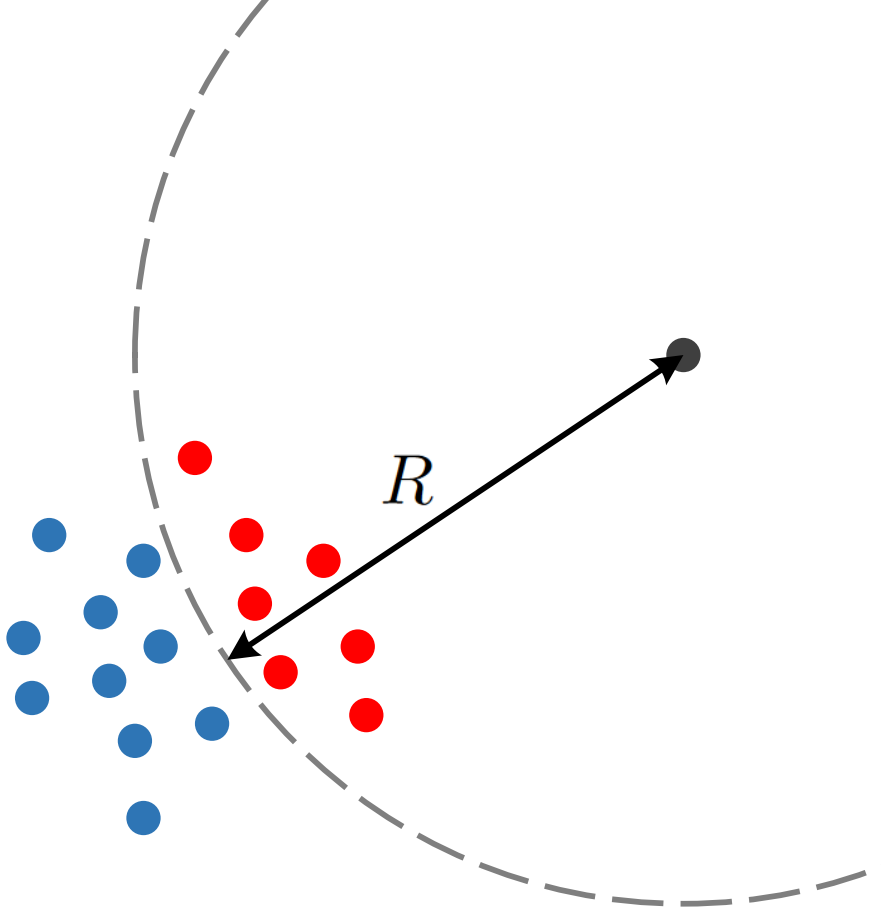
$$\textit{Random Accuracy} = \Pr\left[\lim_{n\to\infty} \text{maj}\{N(x,r_i)\}_{i=1}^{n} = f(x)\right]$$

- Probability that taking the maj of ∞ samples will give the right answer

- Lets see an example…

Ball Problem Example

- Given $x \in \mathbb{R}^d$, determine if $x \in B_R$



- Deterministic, 2-Layer NN[1]:
  - $\Omega(e^d)$ neurons
  - Otherwise, $MSE > \frac{c}{d^4}$

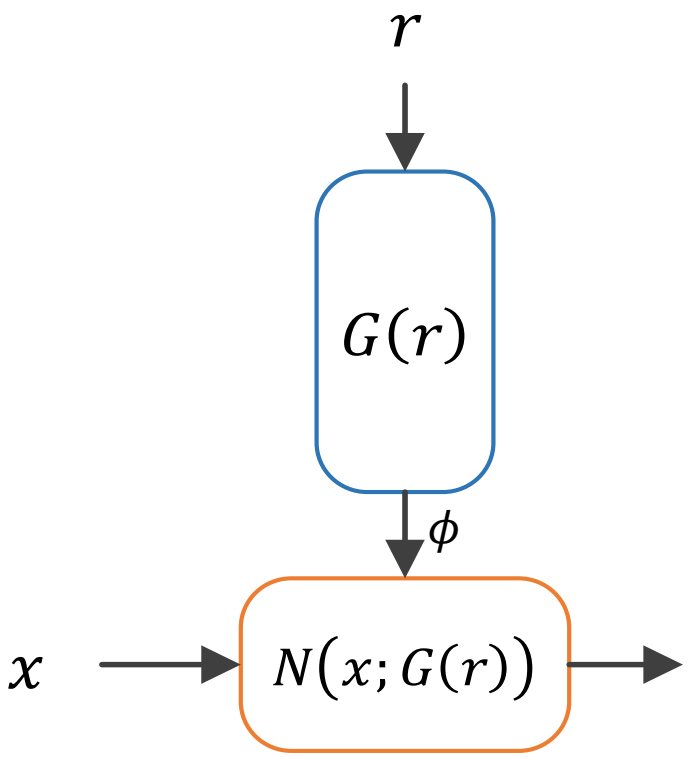
- We show that 2-layer CFNN only require $O(d)$ neurons!
  - Achieve 100% *Random Accuracy*
- Total complexity: $O(poly(d))$, an **exponential improvement!**

[1] Safran & Shamir (2016)

Hypernetwork CFNN

- Generator $\phi = G(r)$ of parameters for $N(x; \phi)$
- If N is sufficiently deep, $G(r) = Ar$ is enough to approx. any function[2]
- "Transposed" theorem:

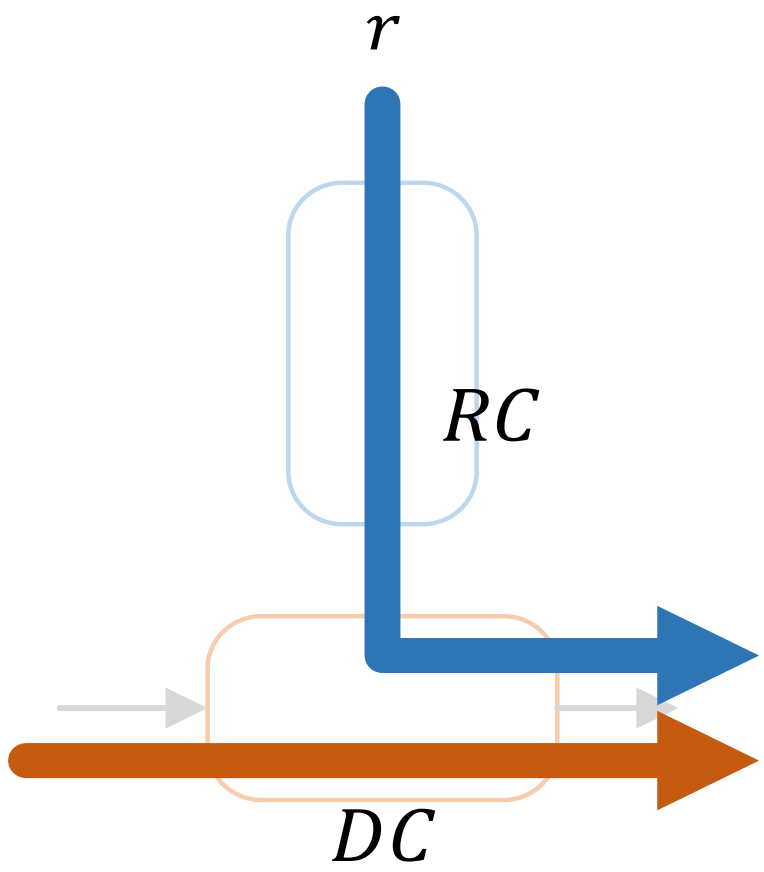
If G is sufficiently deep, $N(x; \phi) = \mathrm{sgn}(\phi x + b)$ is enough to approx. any function $f: \mathbb{R} \to \mathbb{R}$



[2] Dwaracherla et. al. (2020)

DC vs. RC tradeoff

- We suggest a new tradeoff:
- Data Complexity (DC)
  # neurons from x to output
- Random Complexity (RC)
  # neurons from r to output

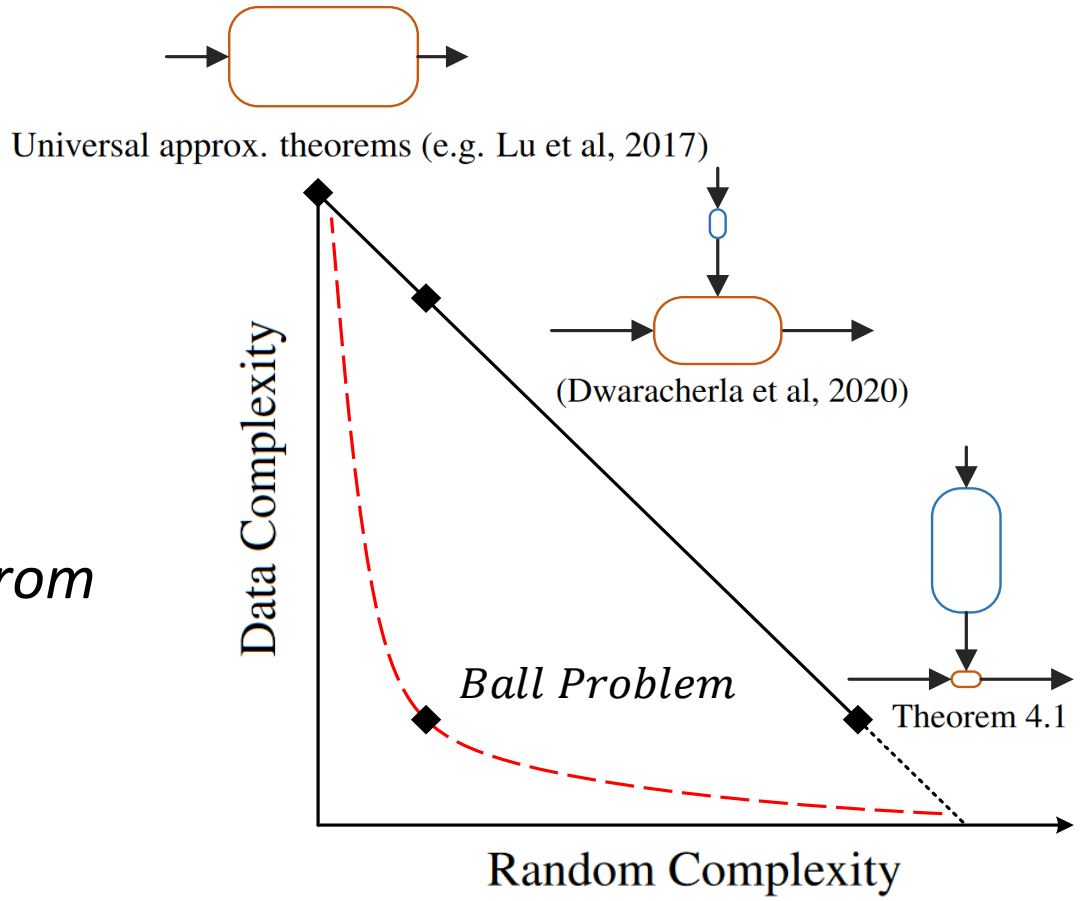


DC vs. RC tradeoff

- *Exact* complexity tradeoff between DC and RC
- Some problems show significant improvement
  - E.g. our Ball Problem
- Conjecture:

Most classification problems will benefit from CFNN

- e.g. functions easy to sample but hard to compute



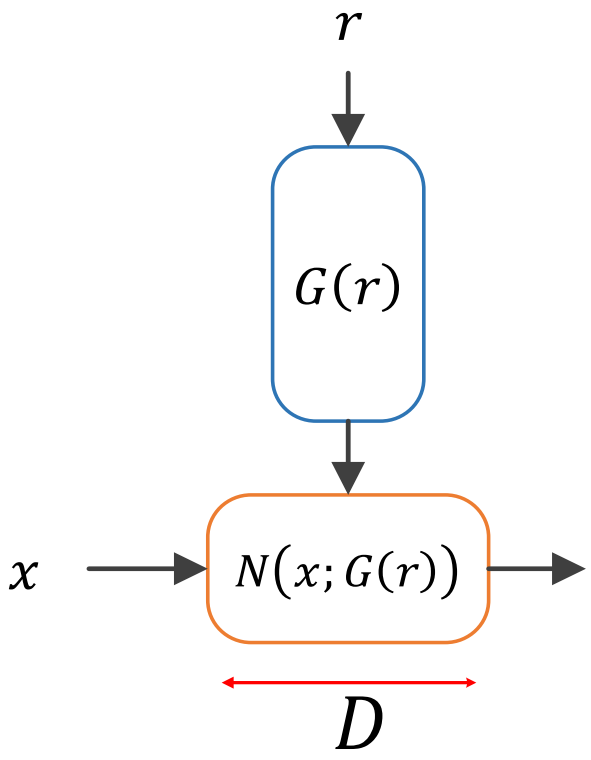
Empirical Study

- Two architectures
- Hypernetwork CFNN
- Resnet20 + Dropout as CFNN

ARCH.	DET.	CFNN	Δ
	CIFAR10		
HYPERNET	79.71 ± 0.5	81.45 ± 0.2	$+1.74 \pm 0.6$
RESNET20	87.82 ± 0.6	90.49 ± 0.1	$\mathbf{+2.67 \pm 0.7}$
	CIFAR100		
HYPERNET	58.09 ± 0.2	61.24 ± 0.2	$+3.15 \pm 0.3$
RESNET20	50.90 ± 0.1	60.16 ± 0.6	$\mathbf{+9.25 \pm 0.7}$

Testing the Tradeoff

D	CFNN	CIFAR10 Det.	Δ
2	44.05	42.14	+1.91
6	**81.45 ± 0.24**	**79.71 ± 0.52**	**+1.74± 0.57**
10	89.37	88.41	+0.96
14	91.81	91.42	+0.39
20	92.37	92.89	-0.52
		CIFAR100	
6	45.29	44.23	+1.06
10	**61.24± 0.41**	**58.09 ±0.22**	**+3.15 ±0.29**
14	66.48	65.16	+1.32
20	69.26	67.68	+1.58
56	72.00	72.23	-0.23



Comparing to MCDropout

METHOD	CIFAR10	CIFAR100
STANDARD TRAIN, STANDARD TEST	87.82 ± 0.65	50.90 ± 0.03
STANDARD TRAIN, MEAN IN TEST (MCDROPOUT)	88.61 ± 0.10	53.66 ± 0.24
STANDARD TRAIN, MAJ IN TEST	88.37 ± 0.18	53.31 ± 0.16
MAJ TRAIN, STANDARD TEST	89.70 ± 0.39	47.92 ± 1.78
MAJ TRAIN, MEAN IN TEST	$\mathbf{90.49 \pm 0.12}$	56.49 ± 0.57
MAJ TRAIN, MAJ TEST (CFNN)	90.20 ± 0.14	$\mathbf{60.16 \pm 0.67}$

Summary

- CFNNs: *Stochastic NNs + Amplification*
- Ball problem – exponential complexity improvement!
- New DC vs. RC tradeoff for hypernetworks
- Empirically verified

Amplification in Inference helps!

TECHNION
Israel Institute
of Technology

The Henry and Marilyn Taub
Faculty of Computer Science

Thank you!

Hope you enjoyed the talk