DEEP NEURAL DECISION FORESTS

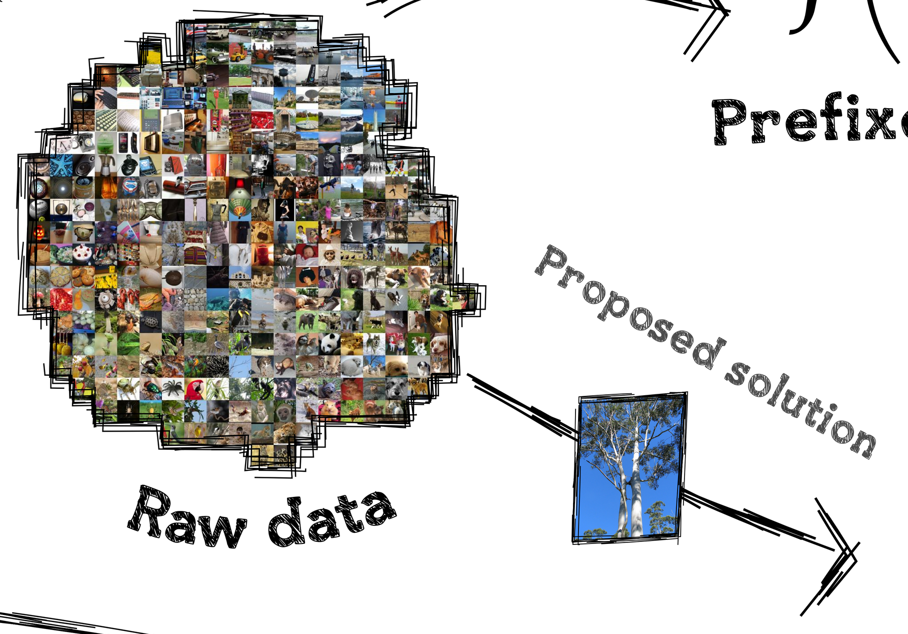
Peter Kontschieder[1] Madalina Fiterau[2] Antonio Criminisi[1] Samuel Rota Bulò[1,3]

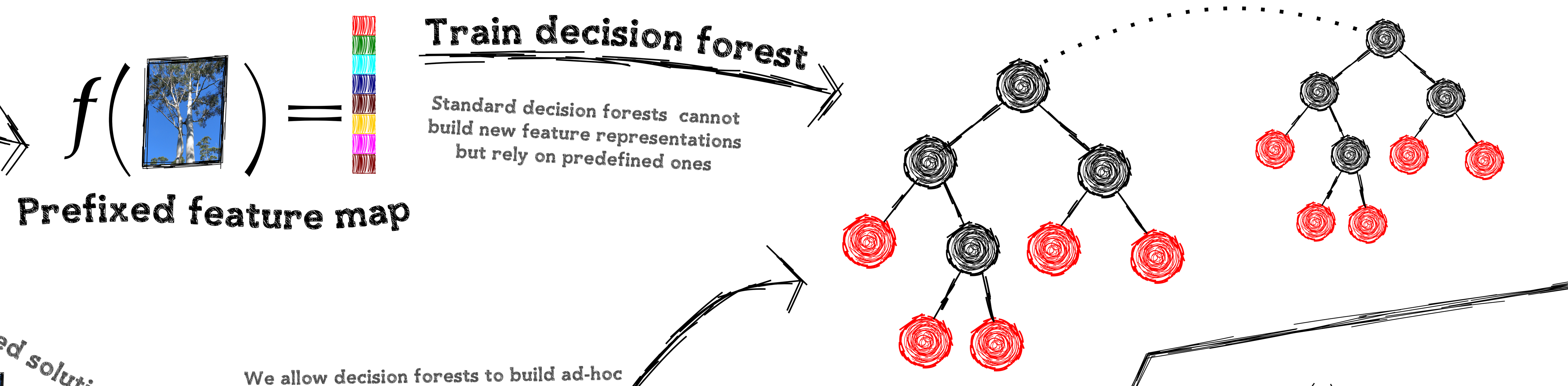
[1] Microsoft Research

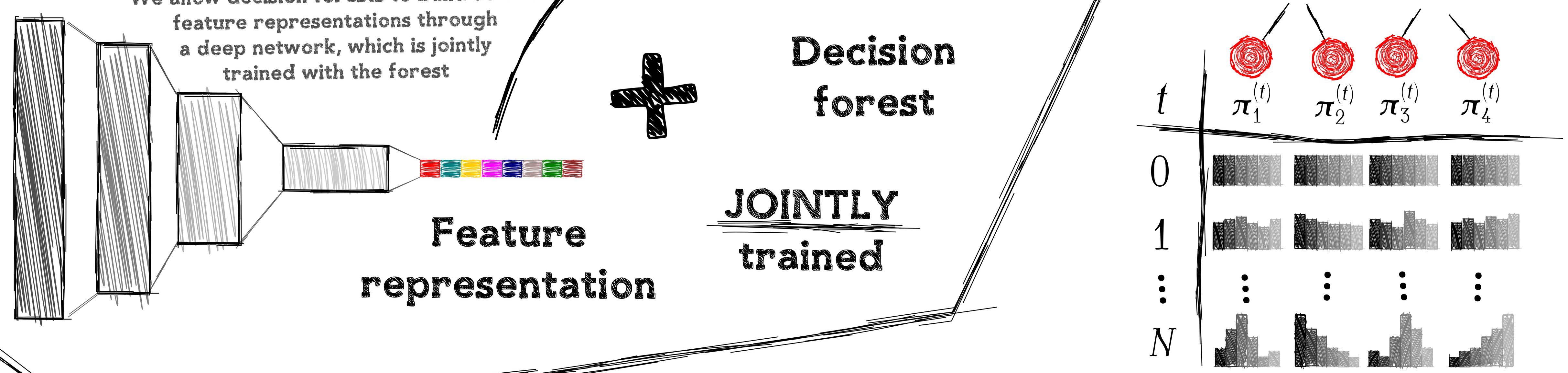
[2] Carnegie Mellon University

[3] FONDAZIONE BRUNO KESSLER

MOTIVATION





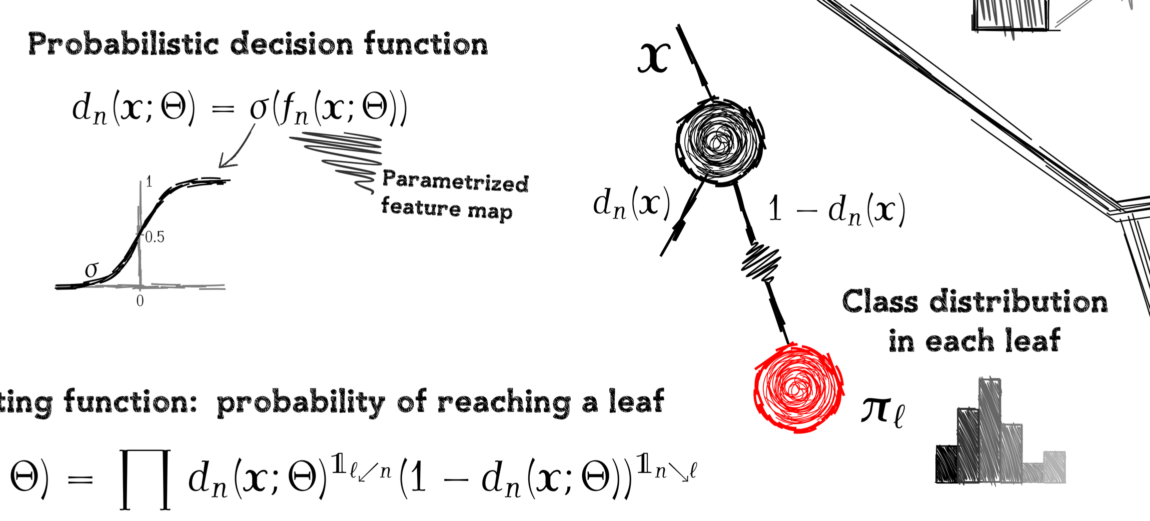


MODEL

Probabilistic decision function

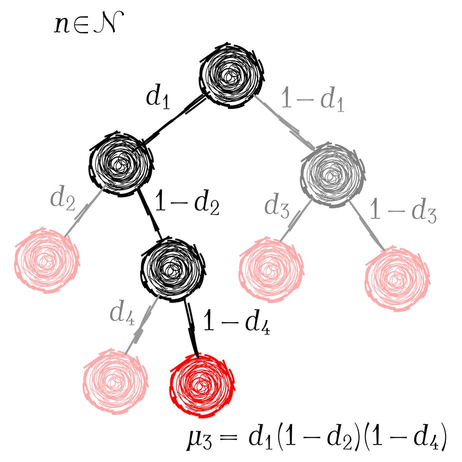
$d_n(x;\Theta) = \sigma(f_n(x;\Theta))$

Parametrized feature map



Routing function: probability of reaching a leaf

$$\mu_\ell(x;\Theta) = \prod_{n\in\mathcal{N}} d_n(x;\Theta)^{\mathbb{1}_{\ell\swarrow n}}\left(1-d_n(x;\Theta)\right)^{\mathbb{1}_{n\searrow\ell}}$$



Decision tree prediction

$$\mathbb{P}_T[y|x;\Theta,\pi] = \sum_{\ell\in\mathcal{L}} \pi_{\ell y}\mu_\ell(x;\Theta)$$

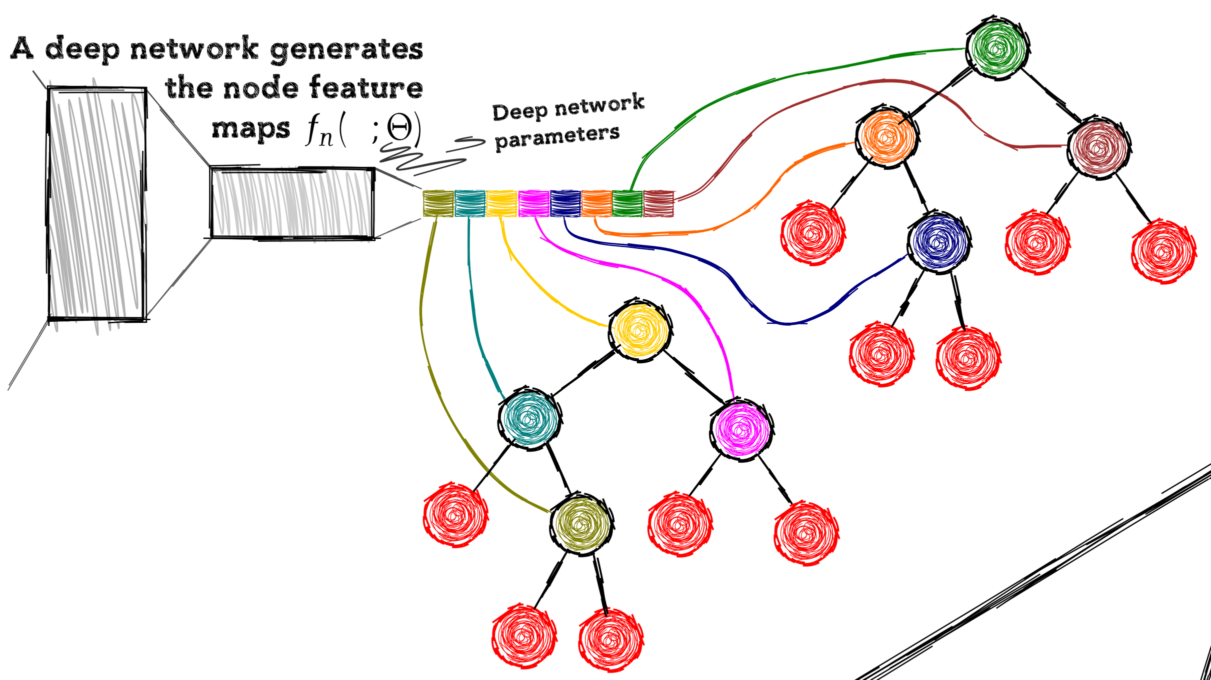
Decision forest prediction

$$\mathbb{P}_{\mathcal{F}}[y|x] = \frac{1}{k}\sum_{h=1}^{k} \mathbb{P}_{T_h}[y|x]$$

Trees in the forest

A deep network generates the node feature maps $f_n(\ \cdot\ ;\Theta)$

Deep network parameters



TRAINING

$\min_{\pi} R(\Theta,\pi;\mathcal{T})$ — Convex optimization problem

Empirical risk minimization

$$R(\Theta,\pi;\mathcal{T}) = \frac{1}{|\mathcal{T}|}\sum_{(x,y)\in\mathcal{T}} L(\Theta,\pi;x,y)$$

Training set; Log-loss: $-\log \mathbb{P}_T[y|x;\Theta,\pi]$

Independent updates per tree

$\min_{\Theta} R(\Theta,\pi;\mathcal{T})$

alternating optimization

Leaf nodes update

$$\pi_{\ell y}^{(t+1)} = \frac{1}{Z_\ell^{(t)}}\sum_{(x,y')\in\mathcal{T}} \frac{\mathbb{1}_{y=y'}\,\pi_{\ell y}^{(t)}\mu_\ell(x;\Theta)}{\mathbb{P}_T[y|x,\Theta,\pi^{(t)}]}$$

normalization factor

Step-size-free update rule

Converges to global optimum

Interleave leaf nodes updates and 1 epoch of decision nodes updates

alternatively per mini-batch leaf nodes updates

Decision nodes update

$$\Theta^{(t+1)} = \Theta^{(t)} - \eta\frac{\partial R}{\partial\Theta}(\Theta^{(t)},\pi;\mathcal{B})$$

Mini-batch

Stochastic Gradient Descent

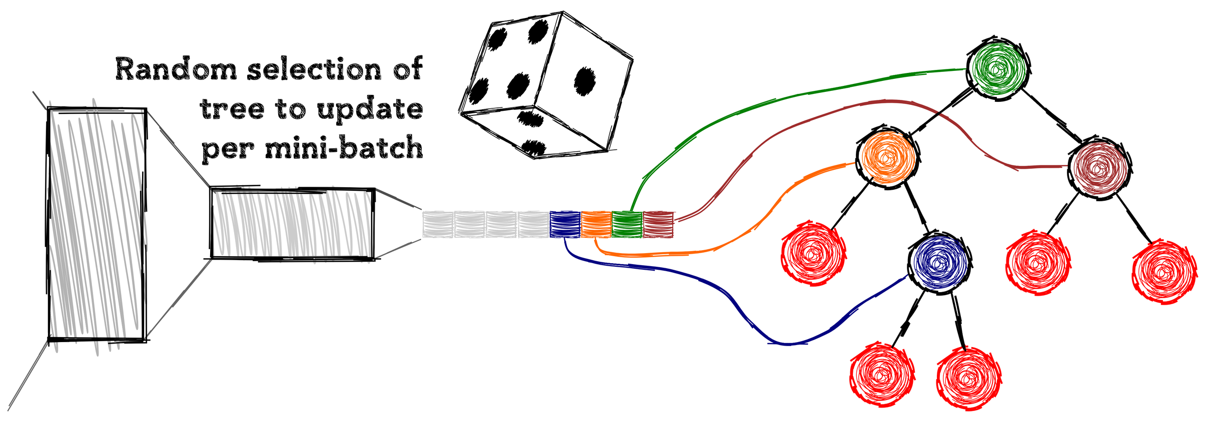
$$\frac{1}{|\mathcal{B}|}\sum_{(x,y)\in\mathcal{B}} \frac{\partial L}{\partial\Theta}(\Theta^{(t)},\pi;x,y)$$

$$\sum_{n\in\mathcal{N}} \frac{\partial L(\Theta,\pi;x,y)}{\partial f_n(x;\Theta)}\frac{\partial f_n(x;\Theta)}{\partial\Theta}$$

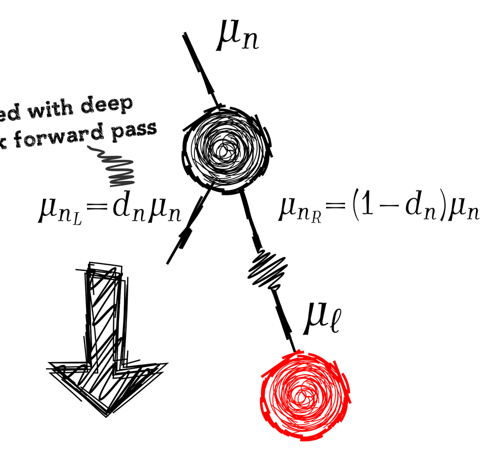
Back-propagation through deep network

$d_n(x;\Theta)A_{n_R} - (1-d_n(x;\Theta))A_{n_L}$

Random selection of tree to update per mini-batch

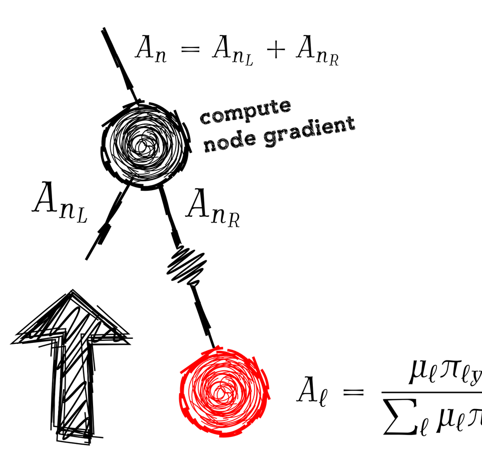


Forward pass over tree



μ_n; $\mu_{n_L} = d_n\mu_n$; $\mu_{n_R} = (1-d_n)\mu_n$; μ_ℓ

Backward pass over tree



$A_n = A_{n_L} + A_{n_R}$; A_{n_L}; A_{n_R}

$$A_\ell = \frac{\mu_\ell\pi_{\ell y}}{\sum_\ell \mu_\ell\pi_{\ell y}}$$

EXPERIMENTS

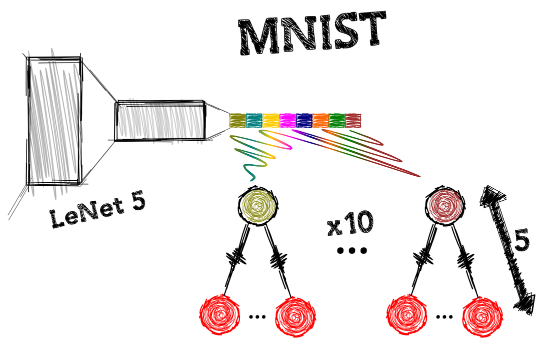
deep Neural Decision Forest (dNDF)

shallow Neural Decision Forest (sNDF)

MACHINE LEARNING DATASETS

	G50c	Letter	USPS	MNIST	Char74k
# Train Samples	50	16000	7291	60000	66707
# Test Samples	500	4000	2007	10000	7400
# Classes	2	26	10	10	62
# Input dimensions	50	16	256	784	64
Alternating Decision Forest (ADF)	18.71±1.27	3.52±0.17	5.59±0.16	2.71±0.10	16.67±0.21
Shallow Neural Decision Forest (sNDF)	17.4±1.52	**2.92±0.17**	**5.01±0.24**	2.8±0.12	**16.04±0.20**
Tree input features	10 (random)	8 (random)	10x10 patches	15x15 patches	10 (random)
Depth	5	10	10	10	12
Number of trees	50	70	100	80	200
Batch size	25	500	250	1000	1000

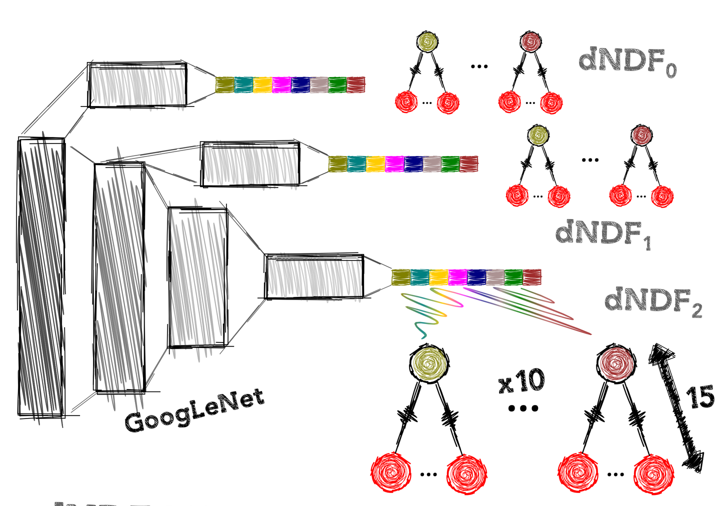
MNIST



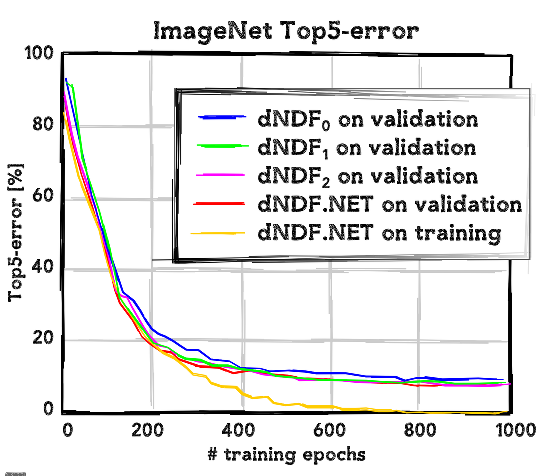
dNDF 0.7%

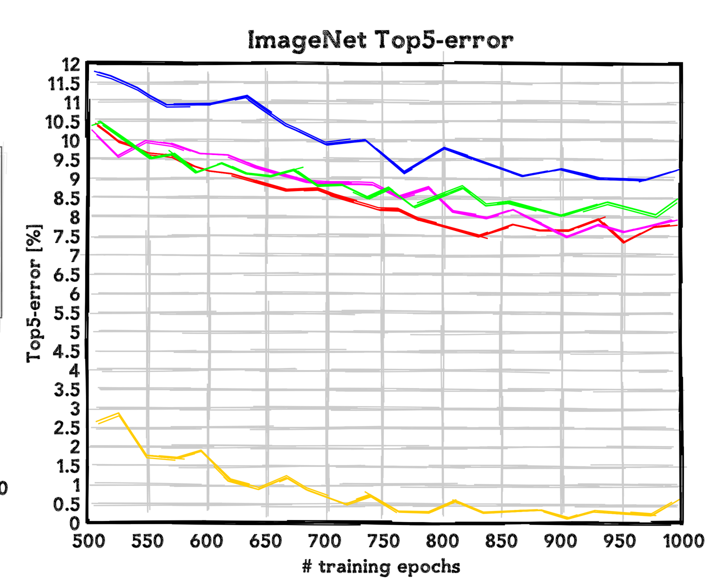
LeNet-5 0.9%

IMAGENET



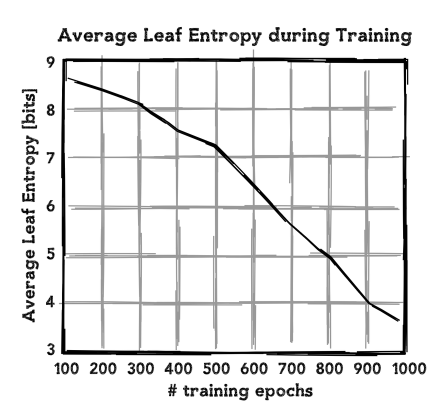
dNDF.NET = dNDF$_0$ + dNDF$_1$ + dNDF$_2$



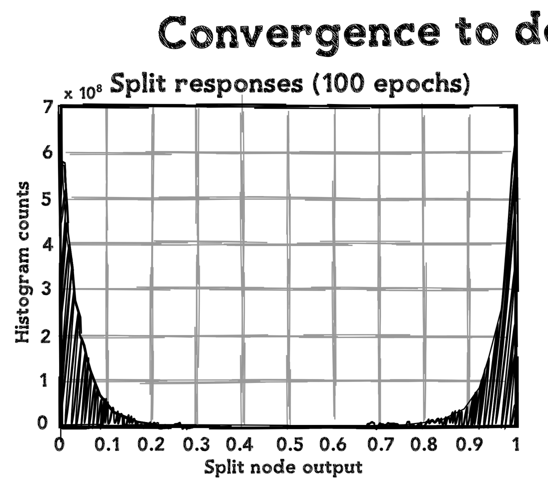


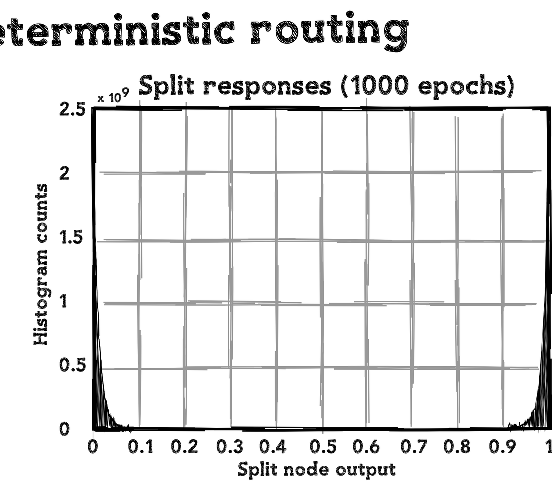
	# models	# crops	Top5-err.
GoogLeNet	1	1	10.07%
	1	10	9.15%
	1	144	7.89%
	7	1	8.09%
	7	10	7.62%
	7	144	6.67%
dNDF.NET	1	1	7.84%
	1	10	7.08%
	7	1	6.38%

Leaf entropy



Convergence to deterministic routing





ICCV15