Sparse Variational Dropout and Variance Networks

Neklyudov Kirill

Agenda

- Variational Dropout
- Sparse Variational Dropout
- Overview of other compression methods
- Variance Networks

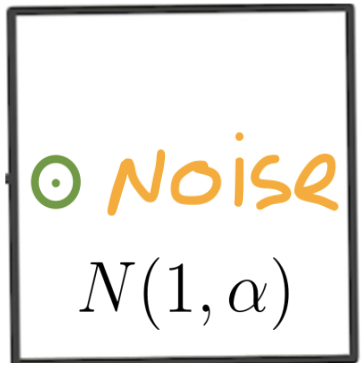
Agenda

- Variational Dropout
- Sparse Variational Dropout
- Overview of other compression methods
- Variance Networks

Gaussian Dropout

input

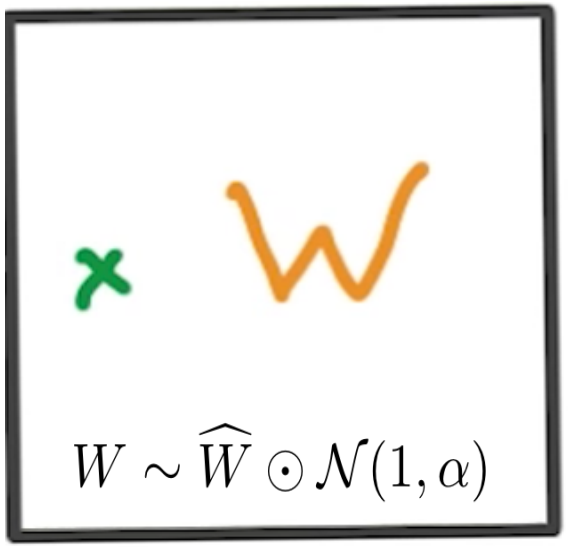
weights



noise

activations

nonlinearity



it does not change the mean value

adds multiplicative noise

$$W \sim \mathcal{N}(\widehat{W}, \alpha \odot \widehat{W}^2)$$

Variational Dropout

$$\mathbb{E}_{q(W\,|\,\phi)} \log p(y\,|\,x, W) - D_{KL}(q(W\,|\,\phi)\,||\,p(W)) \to \max_{\phi}$$

- Posterior distribution

Gaussian Dropout with noise $\sim \mathcal{N}(1, \alpha_{ij})$

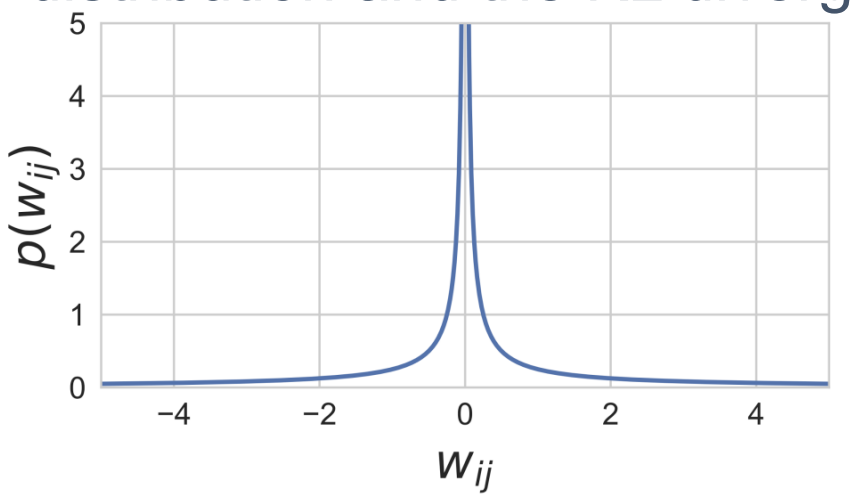
$$w_{ij} = \hat{w}_{ij} \cdot (1 + \sqrt{\alpha_{ij}} \cdot \varepsilon_{ij})$$

$$\varepsilon_{ij} \sim \mathcal{N}(0, 1)$$

$$q(w_{ij}\,|\,\phi_{ij}) = \mathcal{N}(w_{ij}\,|\,\hat{w}_{ij}, \alpha_{ij}\hat{w}_{ij}^2)$$

Prior distribution and the KL divergence term

$$p(w_{ij}) \propto \frac{1}{|w_{ij}|}$$



$$-D_{KL}(q(w_{ij}\,|\,\hat{w}_{ij}, \alpha_{ij})\,\|\,p(w_{ij})) =$$

$$= 0.5 \log \alpha_{ij} - \mathbb{E}_{\epsilon \sim \mathcal{N}(1, \alpha_{ij})} \log |\epsilon| + \mathrm{C}$$

Does not depend on w_{ij}

Diederik Kingma, Tim Salimans, and Max Welling, Variational dropout and the local reparameterization trick

Variational Dropout with the LRT

Local Reparameterization:

$$p(w_{ij}) = \mathcal{N}(w_{ij}|\mu_{ij}, \sigma_{ij}^2) \qquad b_i = w_i^T x$$

$$b_i = \sum_j (\mu_{ij} + \varepsilon_{ij}\sigma_{ij}) \cdot x_{ij} = \sum_j \mu_{ij} x_{ij} + \sum_j \varepsilon_{ij}\sigma_{ij} x_{ij}$$

$$b_i \sim \mathcal{N}\left(\sum_j \mu_{ij} x_{ij}, \sum_j \sigma_{ij}^2 x_{ij}^2 \right)$$

Variational Dropout

$$w_{ij} \sim \mathcal{N}(w_{ij}|\hat{w}_{ij}, \alpha_{ij}\hat{w}_{ij}^2)$$

$$\text{Output} = WX$$

$$\text{Output} = \text{ReLU}(\text{Output})$$

+ Local Reparameterization

$$\text{Output} \sim \mathcal{N}(WX, (A \odot W^2)X^2)$$

$$\text{Output} = \text{ReLU}(\text{Output})$$

Agenda

- Variational Dropout
- Sparse Variational Dropout
- Overview of other compression methods
- Variance Networks

Variance Reduction

The variance of the gradients goes out of control when α are large

$$\frac{\partial \mathcal{L}}{\partial \hat{w}_{ij}} = \frac{\partial \mathcal{L}}{\partial w_{ij}} \cdot \boxed{\frac{\partial w_{ij}}{\partial \hat{w}_{ij}}}$$

$$w_{ij} = \hat{w}_{ij} \cdot (1 + \sqrt{\alpha_{ij}} \cdot \varepsilon_{ij})$$

$$\frac{\partial w_{ij}}{\partial \hat{w}_{ij}} = 1 + \sqrt{\alpha_{ij}}\varepsilon_{ij}$$ **Very noisy!**

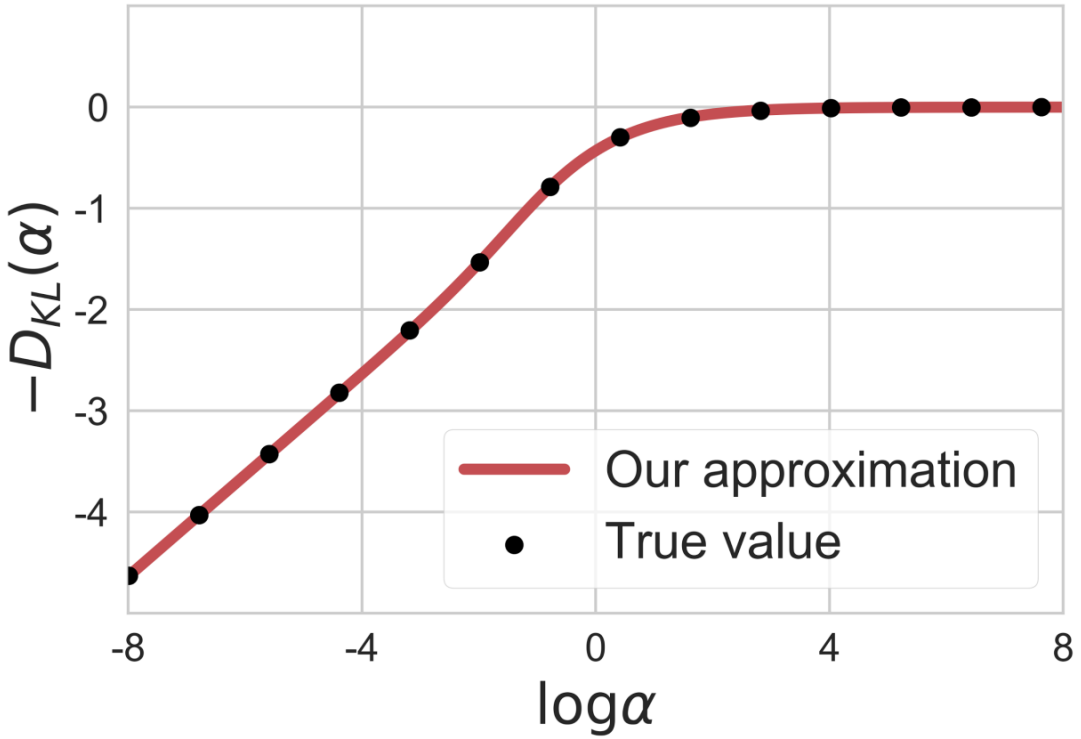
Solution: restrict $0 < \alpha < 1$ (Kingma, et. al.)

It prohibits to use large alphas!

Why do we need large alphas?

$$\mathbb{E}_{q(W|\widehat{W},\alpha)} \log p(t|X,W) - \mathrm{KL}(\alpha) \to \max_{\widehat{W},\alpha}$$

The KL term favors large dropout rates α



Large α_{ij} $(\alpha_{ij} \to +\infty)$ means:

- Infinitely large noise that corrupts the data term

$$w_{ij} = \hat{w}_{ij} \cdot (1 + \alpha_{ij} \cdot \varepsilon_{ij})$$
$$\Rightarrow \hat{w}_{ij} \to 0$$

- Equivalent binary dropout rate goes to 1

$$w_{ij} = \hat{w}_{ij}\theta_{ij} \qquad p_{ij} = \frac{\alpha_{ij}}{1+\alpha_{ij}} \to 1$$
$$\theta_{ij} \sim \mathrm{Bernoulli}(1 - p_{ij})$$

Variance Reduction

The variance of the gradients goes out of control when α are large

$$\frac{\partial \mathcal{L}}{\partial \hat{w}_{ij}} = \frac{\partial \mathcal{L}}{\partial w_{ij}} \cdot \boxed{\frac{\partial w_{ij}}{\partial \hat{w}_{ij}}}$$

Before $w_{ij} = \hat{w}_{ij} \cdot (1 + \sqrt{\alpha_{ij}} \cdot \varepsilon_{ij})$

$$\frac{\partial w_{ij}}{\partial \hat{w}_{ij}} = 1 + \sqrt{\alpha_{ij}}\varepsilon_{ij}$$ **Very noisy!**

Solution: restrict $0 < \alpha < 1$ (Kingma, et. al.) or …

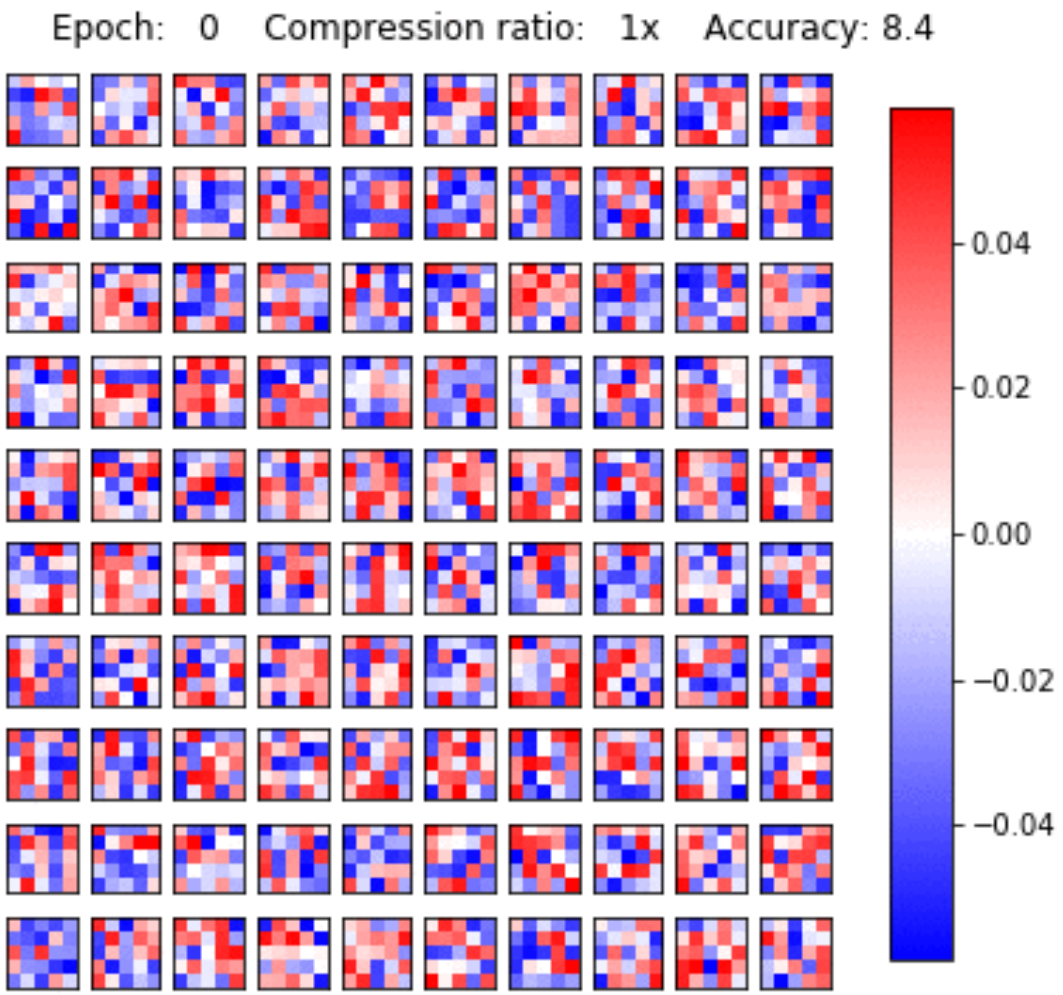
… use Additive Noise Parameterization!

After: $w_{ij} = \hat{w}_{ij} + \sigma_{ij}\varepsilon_{ij}$

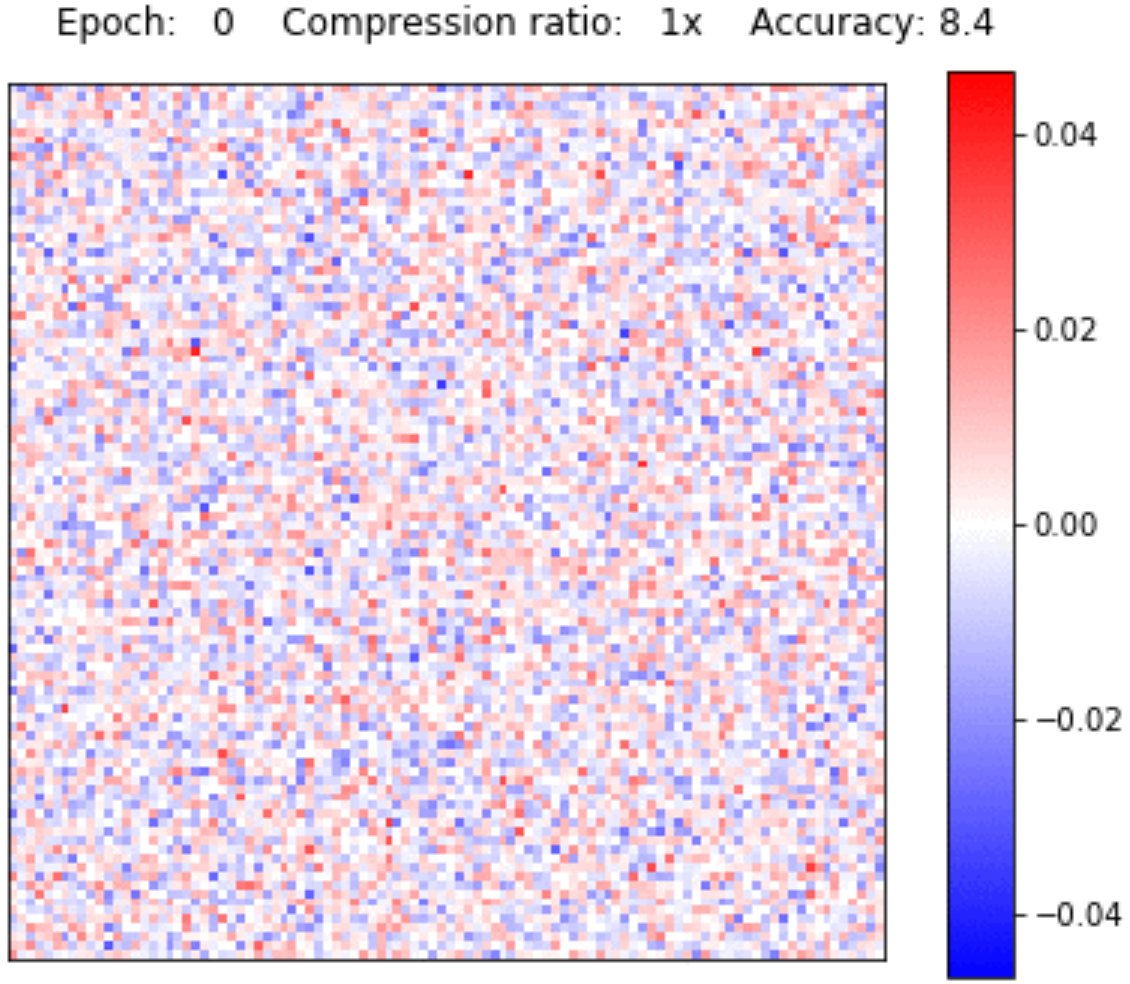
$$\frac{\partial \tilde{w}_{ij}}{\partial w_{ij}} = 1$$ **No noise!**

Optimize the VLB w.r.t. $(\hat{W}, \sigma)$ σ is a new independent variable

Visualization



LeNet-5: convolutional layer



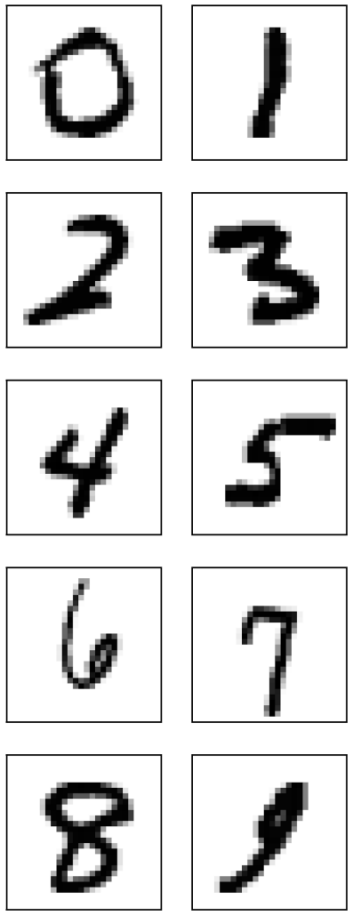
LeNet-5: fully-connected layer
(100 x 100 patch)

Lenet-5-Caffe and Lenet-300-100 on MNIST

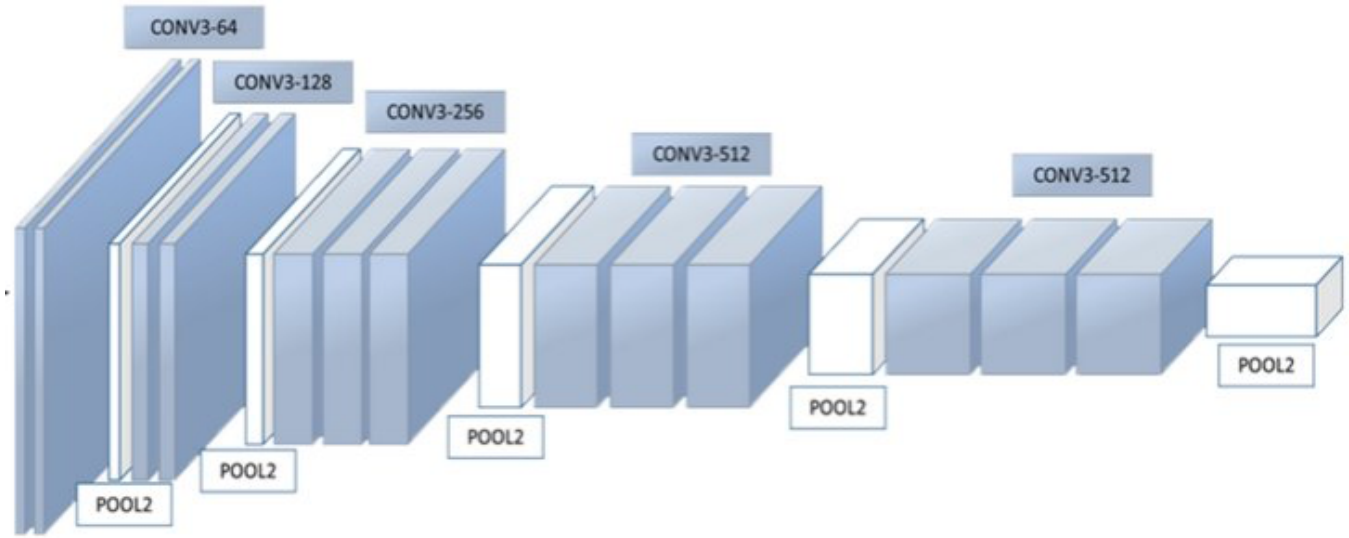
Fully Connected network: LeNet-300-100

Convolutional network: Lenet-5-Caffe

Network	Method	Error %	Sparsity per Layer %	$\frac{\lvert\mathbf{W}\rvert}{\lvert\mathbf{W}_{\neq 0}\rvert}$
	Original	1.64		1
	Pruning	1.59	92.0 – 91.0 – 74.0	12
LeNet-300-100	DNS	1.99	98.2 – 98.2 – 94.5	56
	SWS	1.94		23
(ours)	Sparse VD	1.92	98.9 – 97.2 – 62.0	**68**
	Original	0.80		1
	Pruning	0.77	34 – 88 – 92.0 – 81	12
LeNet-5-Caffe	DNS	0.91	86 – 97 – 99.3 – 96	111
	SWS	0.97		200
(ours)	Sparse VD	0.75	67 – 98 – 99.8 – 95	**280**



VGG-like on CIFAR-10

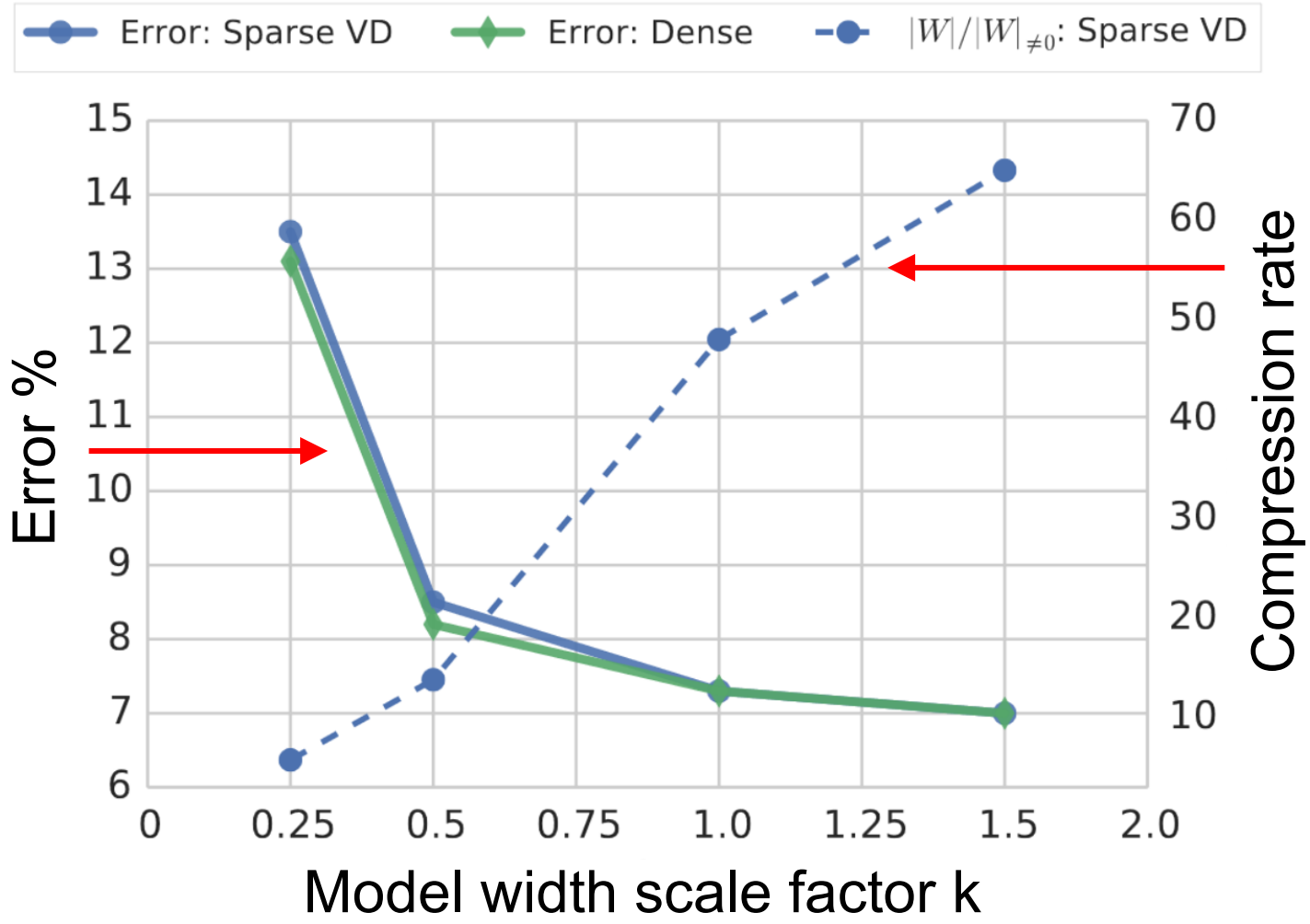


- 13 Convolutional layers and 2 Fully-Connected layers
- Pre-Activation Batch Norm and Binary Dropout after each layer

Official Torch Blog: Zagoruyko, http://torch.ch/blog/2015/07/30/cifar.html

VGG-like on CIFAR-10

Number of filters / neurons is linearly scaled by k (the width of the network)



Random Labeling

Dataset	Architecture	Train Acc.	Test Acc.	Sparsity
MNIST	FC + BD	100%	10%	—
MNIST	FC + Sparse VD	10%	10%	100%
CIFAR-10	VGG + BD	100%	10%	—
CIFAR-10	VGG + Sparse VD	10%	10%	100%

No dependency between data and labels $\Rightarrow$ Sparse VD yields an empty model where conventional models easily overfit.

Zhang, Chiyuan, et al. "Understanding deep learning requires rethinking generalization."

Agenda

- Variational Dropout
- Sparse Variational Dropout
- Overview of other compression methods
- Variance Networks

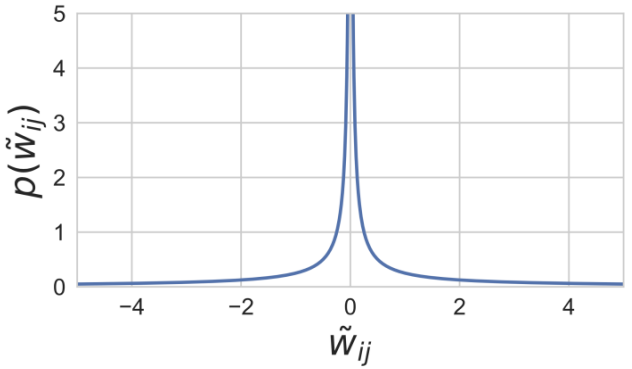
Bayesian Sparsification of Recurrent Neural Networks

$$h_t = f_h(x_t, h_{t-1}) = g_h(x_t W^x + h_{t-1} W^h + b_1)$$

$$q(w_{ij}^x) = N(w_{ij}^x | \hat{w}_{ij}^x, {\sigma_{ij}^x}^2) \qquad q(w_{ij}^h) = N(w_{ij}^h | \hat{w}_{ij}^h, {\sigma_{ij}^h}^2)$$

$$p(t|x, W) = \text{SoftMax}\Big(f_y\Big(f_h(x_T, f_h\left(\ldots, f_h(x_1, h_0)\right)\Big)\Big)$$

- Log-uniform prior
- The same sample W for all timestamps
- We cannot perform LRT



$$\mathbb{E}_{q(W\,|\,\phi)} \log p(y\,|\,x, W) - D_{KL}(\alpha_x) - D_{KL}(\alpha_h) \to \max_{\phi}$$

Bayesian Sparsification of RNNs: Sentiment Analysis

Method (init)	Test MSE	Sparsity x - h %
No dropout	0.1518	–
VBD	0.1488	–
SparseVD (random)	0.1526	99.91 – 99.90
SparseVD (no dropout)	0.1503	**99.95 – 99.92**
SparseVD (VBD)	**0.1475**	99.63 – 99.49

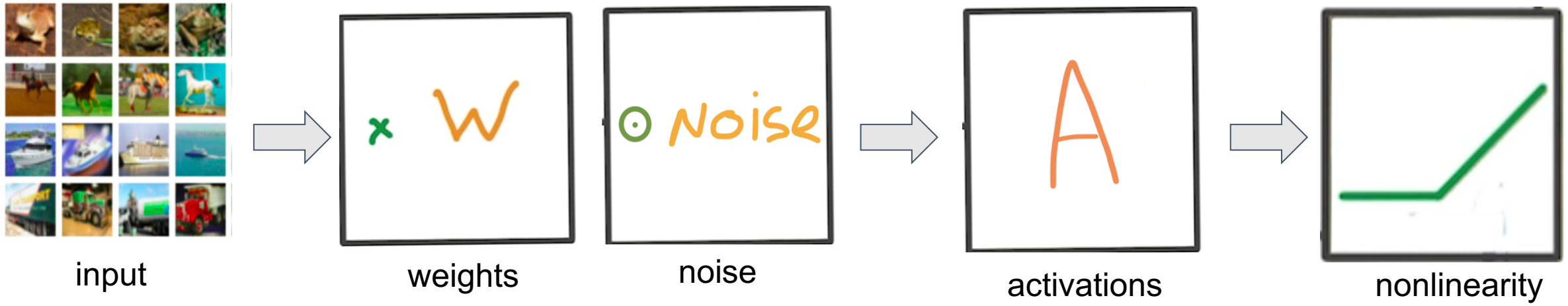
Bayesian Sparsification of RNNs: Language Modeling

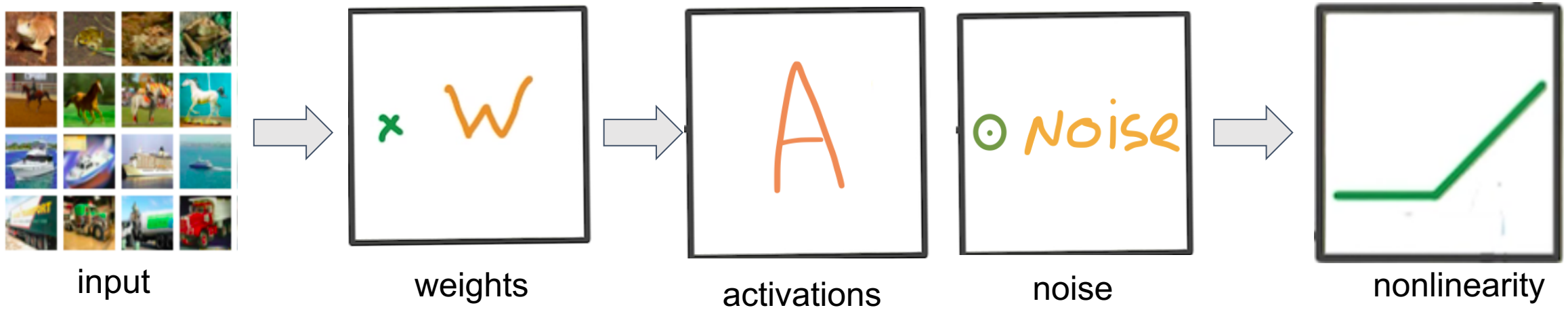
Method (init)	Valid	Test	Sparsity x - h - y %
No dropout	1.499	1.453	–
VBD for W^h	**1.394**	**1.358**	–
SparseVD (random)	1.506	1.461	**79.7 – 88.0 – 71.7**
SparseVD (no dropout)	1.454	1.414	61.9 – 60.0 – 43.9
SparseVD (VBD for W^h)	1.409	1.372	52.2 – 49.8 – 37.9

Group sparsity

- Sparse VD provides **general** sparsity with **no structure**
- Sparse VD can’t be efficiently generalized for group sparsity
- Structured sparsity is the key to DNN acceleration
- Structured sparsity allows more efficient compression

Activations sparsity





Distributions

Variational Dropout:

$$p(w_{ij}) \propto \frac{1}{|w_{ij}|}$$

$$q(w_{ij}|\phi_{ij}) = \mathcal{N}(w_{ij}|\hat{w}_{ij}, \alpha_{ij}\hat{w}_{ij}^2)$$

Log-Normal Dropout:

$$p(w_{ij}) = \mathrm{LogU}_{[a,b]}(w_{ij})$$

$$q(w_{ij}|\phi_{ij}) = \mathrm{LogN}_{[a,b]}(w_{ij}|\mu_{ij}, \sigma_{ij}^2)$$

Results on MNIST

Network	Method	Error %	Neurons per Layer %	CPU	GPU	FLOPs
LeNet-500-300 (ours)	Original	1.54	784 – 500 – 300 – 10	1.00×	1.00×	1.00×
	SparseVD	1.57	537 – 217 – 130 – 10	1.19×	1.03×	3.73×
	SSL	1.49	434 – 174 – 78 – 10	2.21×	1.04×	6.06×
	StructuredBP	1.55	245 – 160 – 55 – 10	2.33×	1.08×	11.23×
LeNet5-Caffe (ours)	Original	0.80	20 – 50 – 800 – 500	1.00×	1.00×	1.00×
	SparseVD	0.75	17 – 32 – 329 – 75	1.48×	1.41×	2.19×
	SSL	1.00	3 – 12 – 800 – 500	5.17×	1.80×	3.90×
	StructuredBP	0.86	3 – 18 – 284 – 283	5.41×	1.91×	10.49×

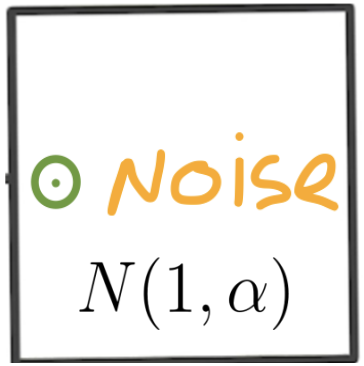
Agenda

- Variational Dropout
- Sparse Variational Dropout
- Overview of other compression methods
- Variance Networks

Gaussian Dropout

input

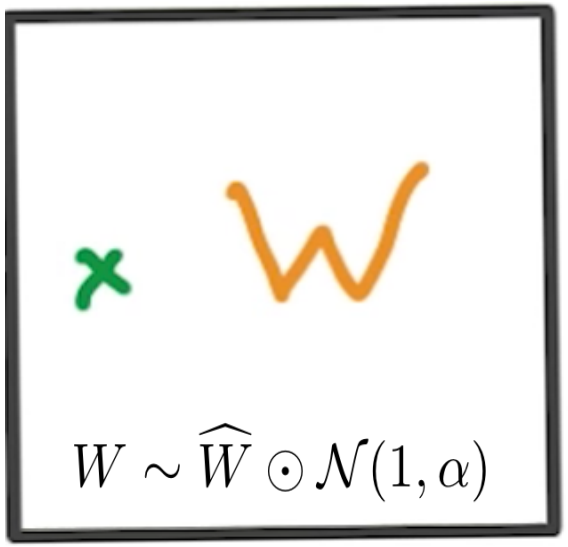
weights



noise

activations

nonlinearity



it does not change the mean value

adds multiplicative noise

$$W \sim \mathcal{N}(\widehat{W}, \alpha \odot \widehat{W}^2)$$

Variational Dropout

$$\mathbb{E}_{q(W\,|\,\phi)} \log p(y\,|\,x, W) - D_{KL}(q(W\,|\,\phi)\,||\,p(W)) \to \max_{\phi}$$

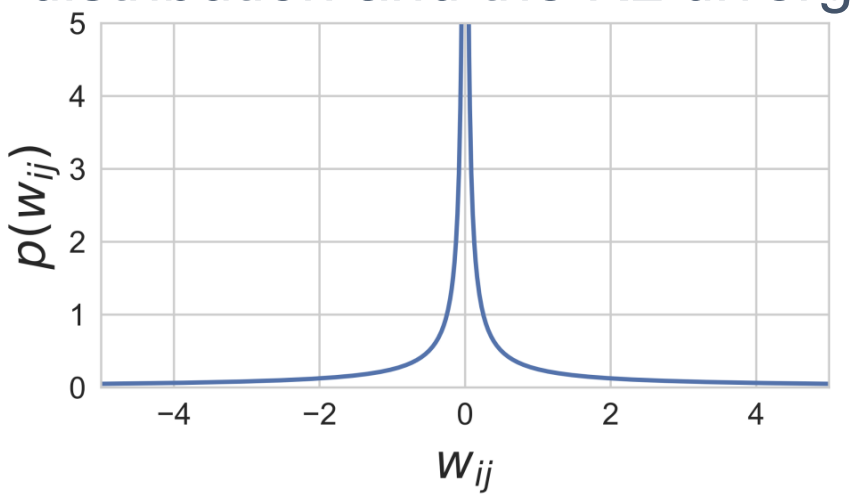
- Posterior distribution

$$w_{ij} = \hat{w}_{ij} \cdot (1 + \sqrt{\alpha} \cdot \varepsilon_{ij})$$

$$\varepsilon_{ij} \sim \mathcal{N}(0, 1)$$

$$q(w_{ij}\,|\,\phi_{ij}) = \mathcal{N}(w_{ij}\,|\,\hat{w}_{ij}, \alpha\hat{w}_{ij}^2)$$

Prior distribution and the KL divergence term



$$p(w_{ij}) \propto \frac{1}{|w_{ij}|}$$

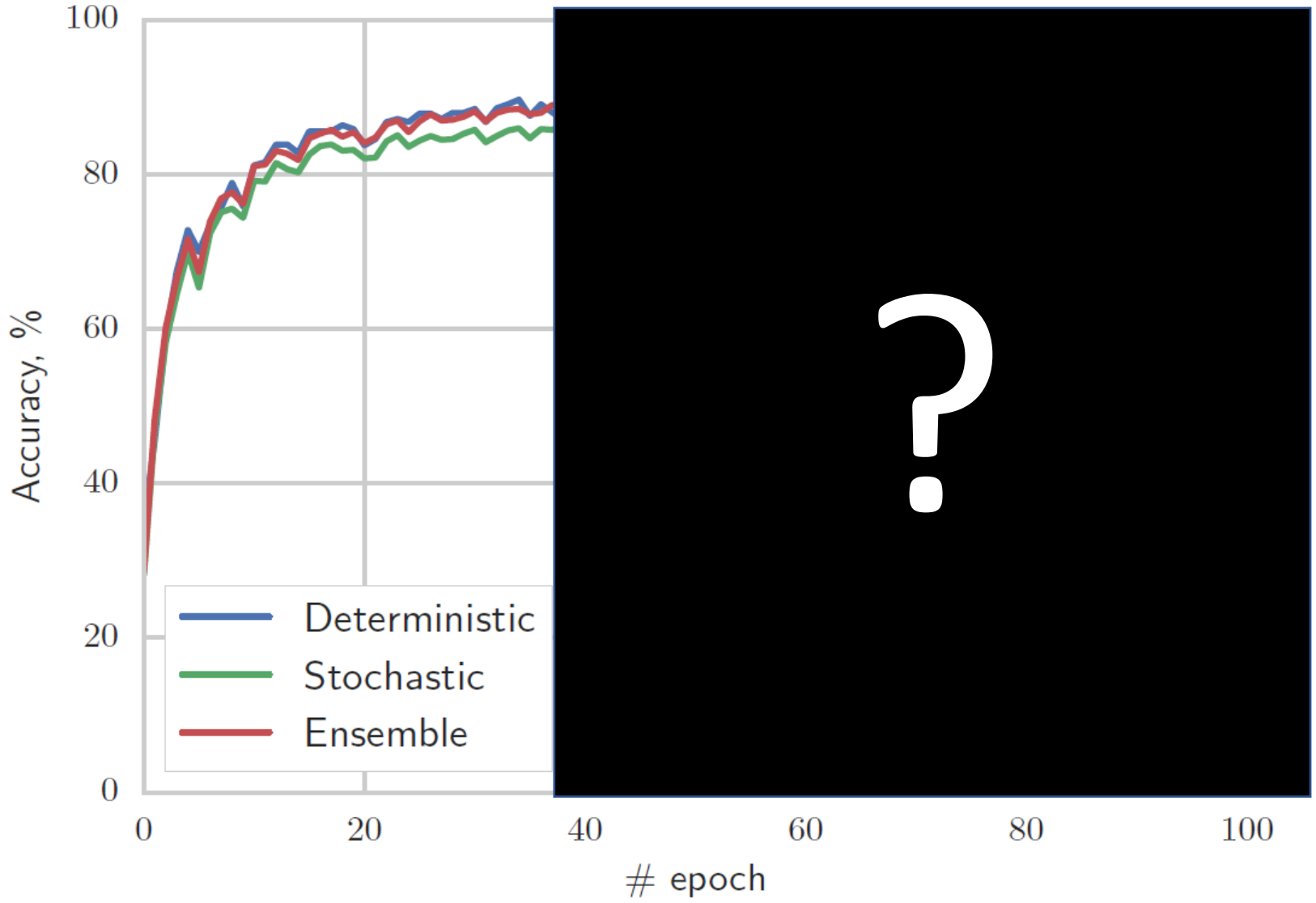
$$-D_{KL}(q(w_{ij}\,|\,\hat{w}_{ij}, \alpha)\,|\,p(w_{ij})) =$$

$$0.5 \log \alpha - \mathbb{E}_{\varepsilon \sim \mathcal{N}(1,\alpha)} \log |\epsilon| + \mathrm{C}$$

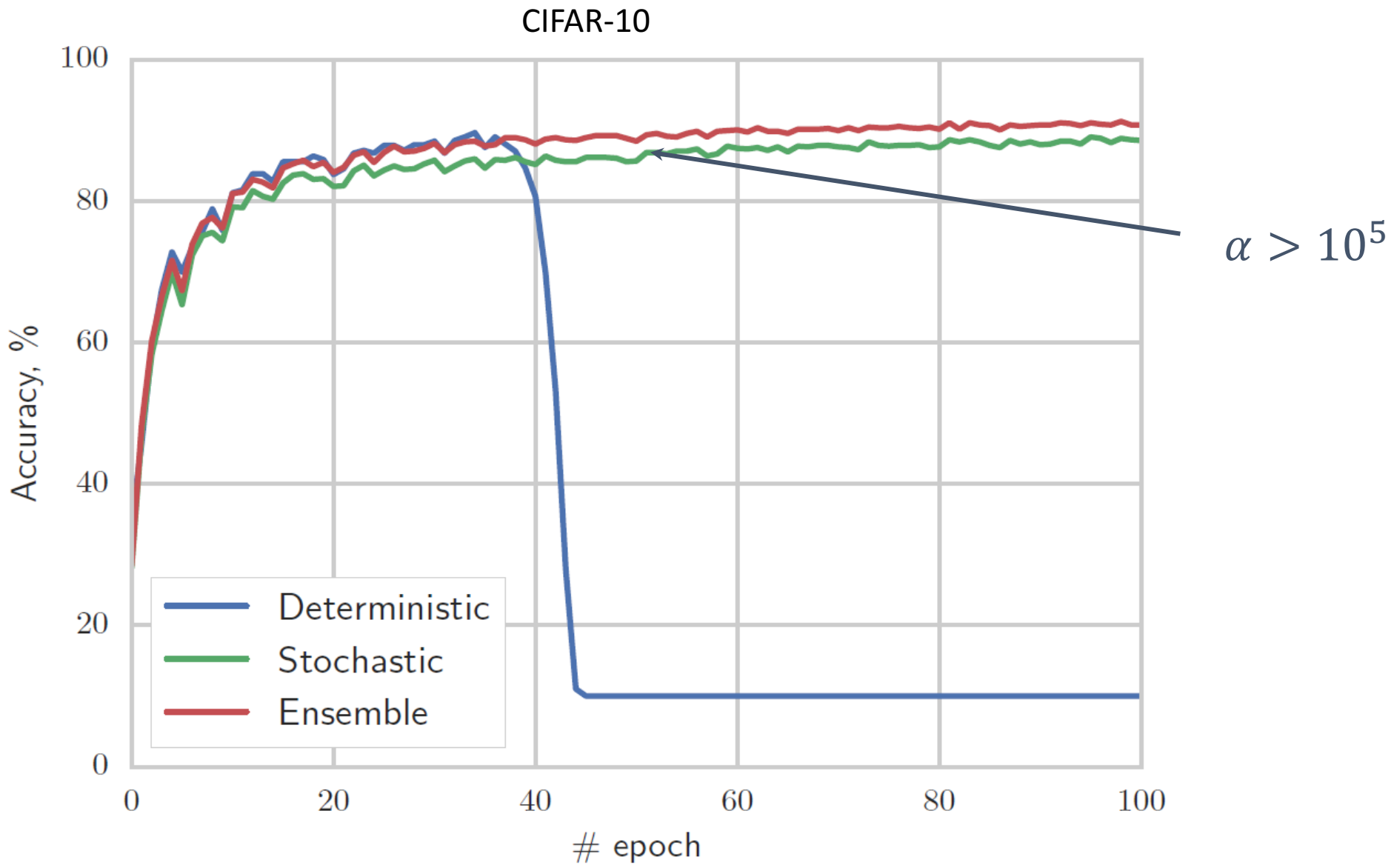
Diederik Kingma, Tim Salimans, and Max Welling, Variational dropout and the local reparameterization trick

What will happen?

CIFAR-10



Something weird is happening



Variational dropout converges to a variance network

Let's take a look at alpha $\sigma_{ij}^2 = \alpha \hat{w}_{ij}^2$

$$\frac{1}{\alpha} = \frac{\hat{w}_{ij}^2}{\sigma_{ij}^2} = \mathrm{SNR} = 10^{-5} \qquad \mathcal{N}(\hat{w}_{ij}, \alpha \hat{w}_{ij}^2) \to \mathcal{N}(0, \sigma_{ij}^2)$$

- We have almost zero Signal-to-Noise Ratio (SNR)
- Means are negligible compared to variances

Theorem

$$\mathrm{MMD}\left(\mathcal{N}(\hat{w}_{ij}, \alpha \hat{w}_{ij}^2) \middle|\middle| \mathcal{N}(0, \alpha \hat{w}_{ij}^2)\right) \to 0 \quad \text{as} \quad \alpha \to \infty$$

Variance Netrworks

$$\mathbb{E}_{q(W\,|\,\phi)} \log p(y\,|\,x, W) - D_{KL}(q(W\,|\,\phi)\,||\,p(W)) \to \max_{\phi}$$

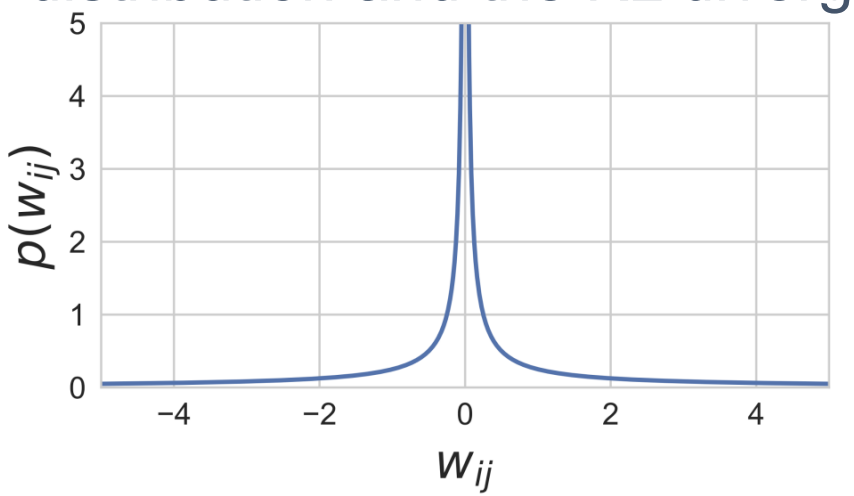
- Posterior distribution

$$w_{ij} = \sigma_{ij} \cdot \varepsilon_{ij}$$

$$\varepsilon_{ij} \sim \mathcal{N}(0, 1)$$

$$q(w_{ij}\,|\,\phi_{ij}) = \mathcal{N}(w_{ij}\,|\,0, \sigma_{ij}^2)$$

Prior distribution and the KL divergence term



$$p(w_{ij}) \propto \frac{1}{|w_{ij}|}$$

$$-D_{KL}(q(w_{ij}\,|\,0, \sigma_{ij}^2)\,|\,p(w_{ij})) = \text{const}$$

Different parameterizations

- Variance network is actually the best possible variational dropout network!
- Sparse Variational Dropout is just a poor local optimum 😔

	Layer	Neuron	Weight	Additive
ELBO	$\mathbf{-5.9 \cdot 10^2}$	$-7.7 \cdot 10^2$	$-6.4 \cdot 10^4$	$-2.3 \cdot 10^4$
Det. accuracy	11.3	11.3	81.3	96.3
Ens. accuracy	99.2	99.2	99.2	99.2

$$q(w_{ij}) = \mathcal{N}(w_{ij} \,|\, \mu_{ij}, \alpha \mu_{ij}^2) \qquad \text{layer-wise}$$

$$q(w_{ij}) = \mathcal{N}(w_{ij} \,|\, \mu_{ij}, \alpha_j \mu_{ij}^2) \qquad \text{neuron-wise}$$

$$q(w_{ij}) = \mathcal{N}(w_{ij} \,|\, \mu_{ij}, \alpha_{ij} \mu_{ij}^2) \qquad \text{weight-wise}$$

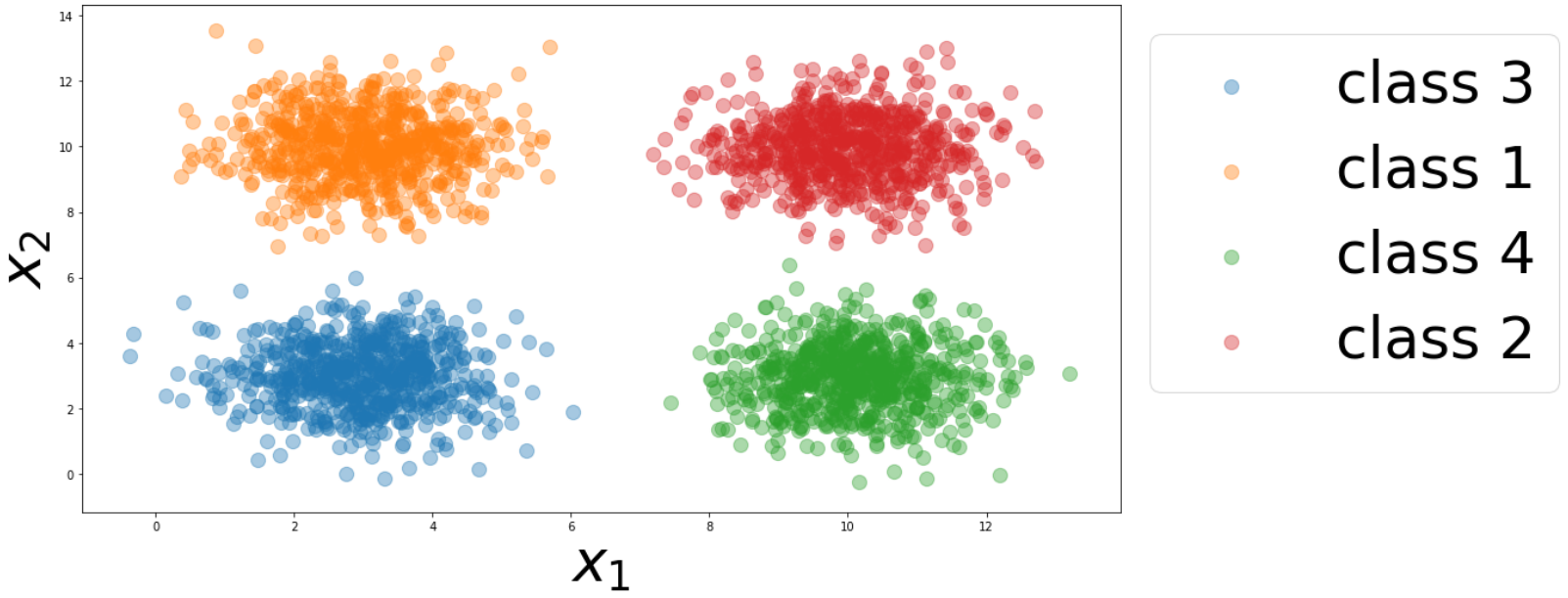
$$q(w_{ij}) = \mathcal{N}(w_{ij} \,|\, \mu_{ij}, \sigma_{ij}^2) \qquad \text{additive}$$

Experiments: classification

Architecture	Dataset	Network	Accuracy (%)		
			Stoch.	Det.	Ens.
LeNet5	MNIST	Dropout	99.1	99.4	99.4
		Variance	95.9	10.1	99.3
VGG-like	CIFAR10	Dropout	91.0	93.1	93.4
		Variance	91.3	10.0	93.4
VGG-like	CIFAR100	Dropout	77.5	79.8	81.7
		Variance	76.9	5.0	82.2

Intuition of Variance networks on toy problem

Dataset:

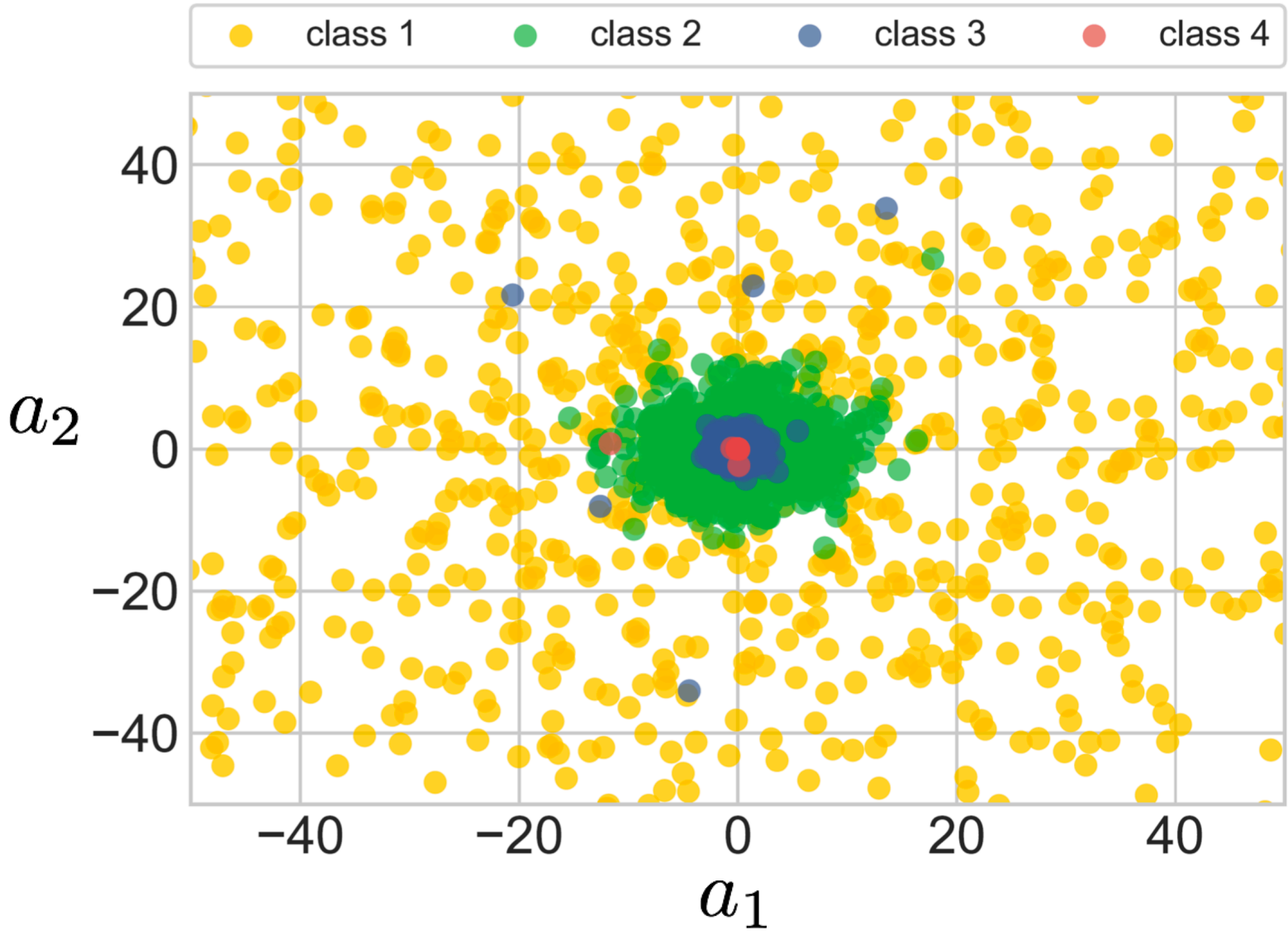


Model:

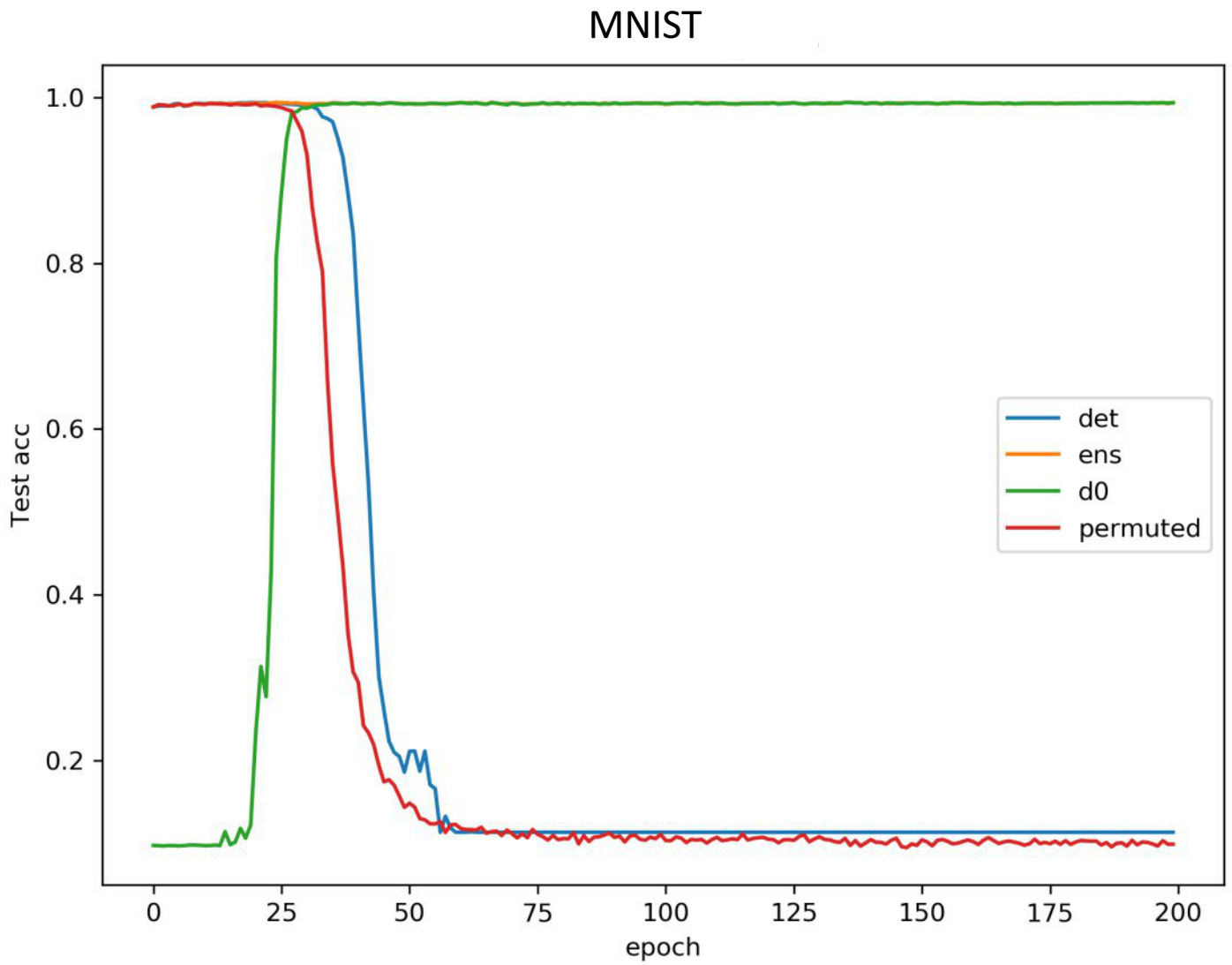
$$x \rightarrow \boxed{2 \times d_1} \rightarrow \boxed{d_1 \times d_2} \cdots \boxed{d_i \times 2} \rightarrow \boxed{2 \times 4} \rightarrow y$$

Variance layer with output $\begin{bmatrix} a_1 \\ a_2 \end{bmatrix}$

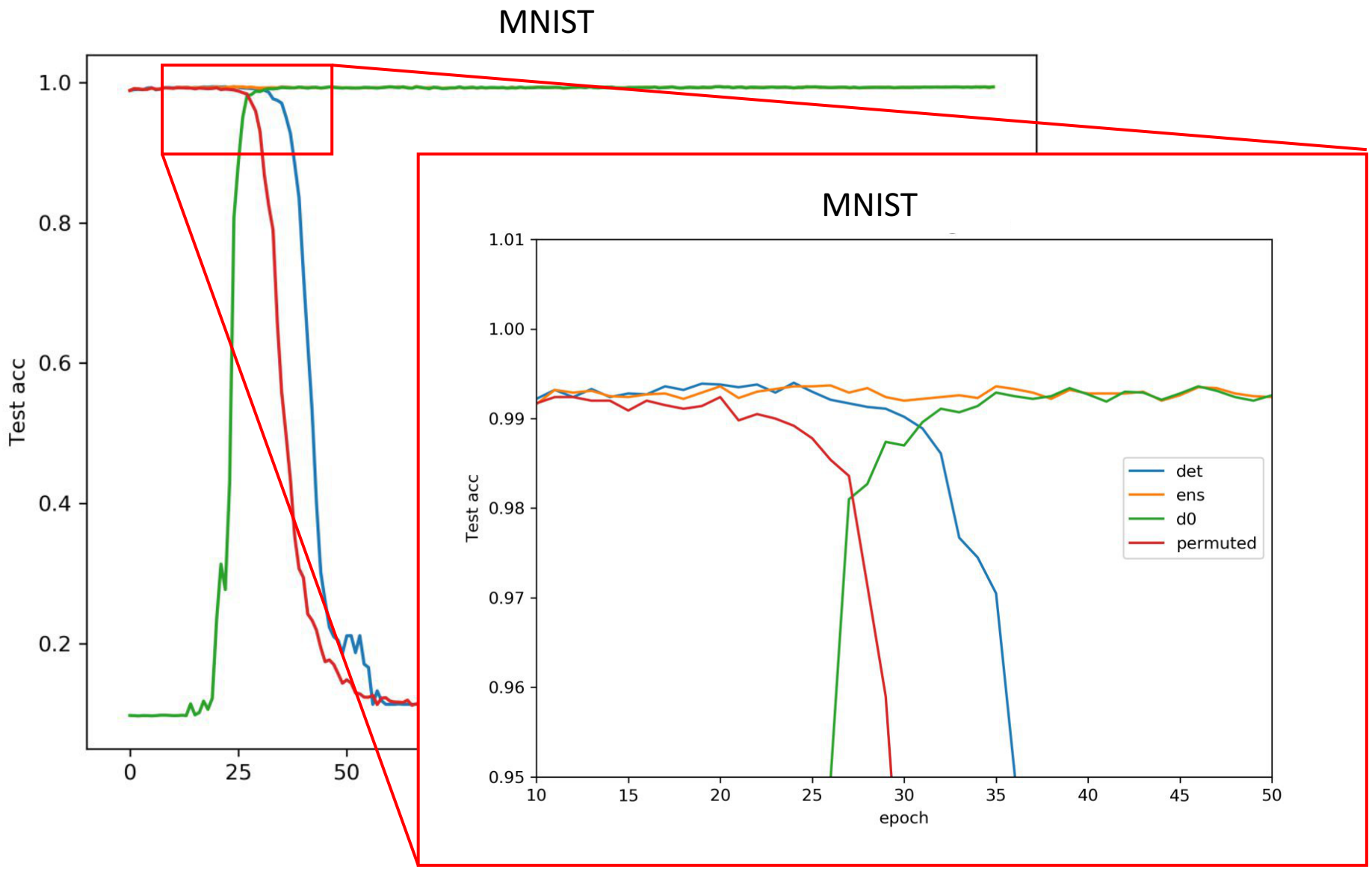
Intuition of Variance networks on toy problem



Information flow from means to variances



Information flow from means to variances



Conclusions

- Variational Dropout is one of the first examples of Bayesian Neural Networks
- A lot of different models with totally different properties
  appeared from Variational Dropout
- Variational Inference is a powerful framework
  for building Bayesian Neural Networks
- We can efficiently store information in the variances (if we remove means 🙃)
- Don’t forget about pitfalls of Weight Scaling Rule