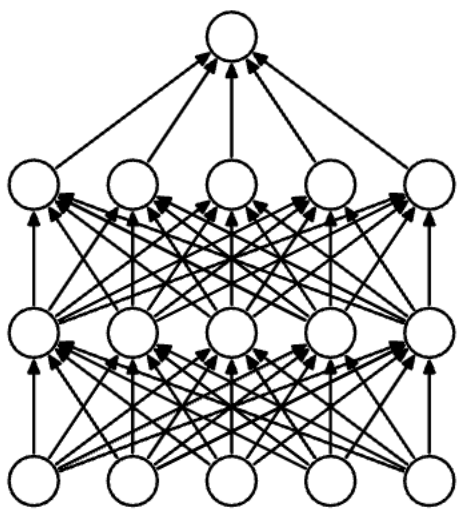
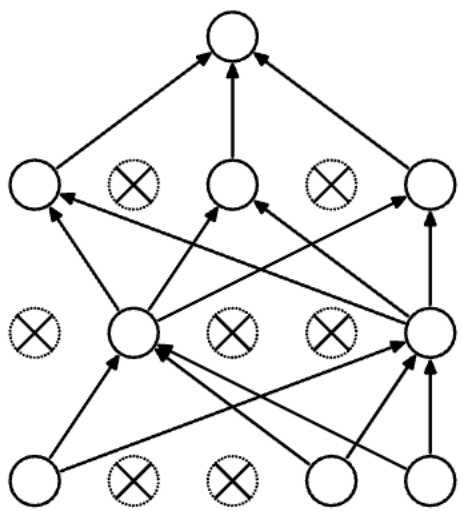
Dropout: A Simple Way to Prevent Neural Networks from Overfitting

Nitish Srivastava NITISH@CS.TORONTO.EDU
Geoffrey Hinton HINTON@CS.TORONTO.EDU
Alex Krizhevsky KRIZ@CS.TORONTO.EDU
Ilya Sutskever ILYA@CS.TORONTO.EDU
Ruslan Salakhutdinov RSALAKHU@CS.TORONTO.EDU
Department of Computer Science
University of Toronto
10 Kings College Road, Rm 3302
Toronto, Ontario, M5S 3G4, Canada.



(a) Standard Neural Net



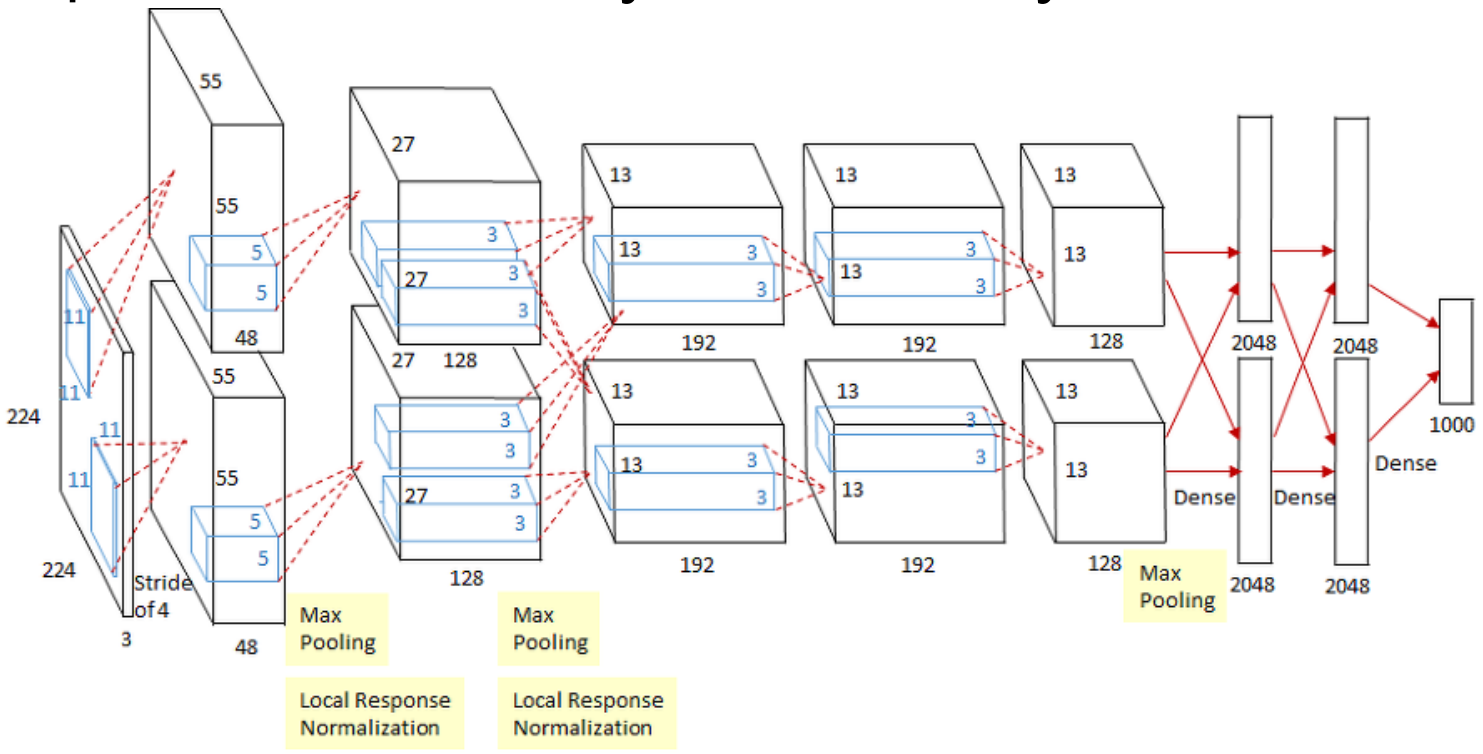
(b) After applying dropout.

Contents

- Brief History
- *“Dropout: A Simple Way to Prevent Neural Networks from Overfitting (2014)”*
  - Methods/Results
  - Discussion
- Mathematical Interpretations
- Dropout on various architectures
- Further Readings

Brief History

- AlexNet (2012) *- Alex Krizhevsky, Ilya Sutskever,* ***Geoffrey E. Hinton***
  - Dropout, p = 0.5, first two **fully-connected layers**

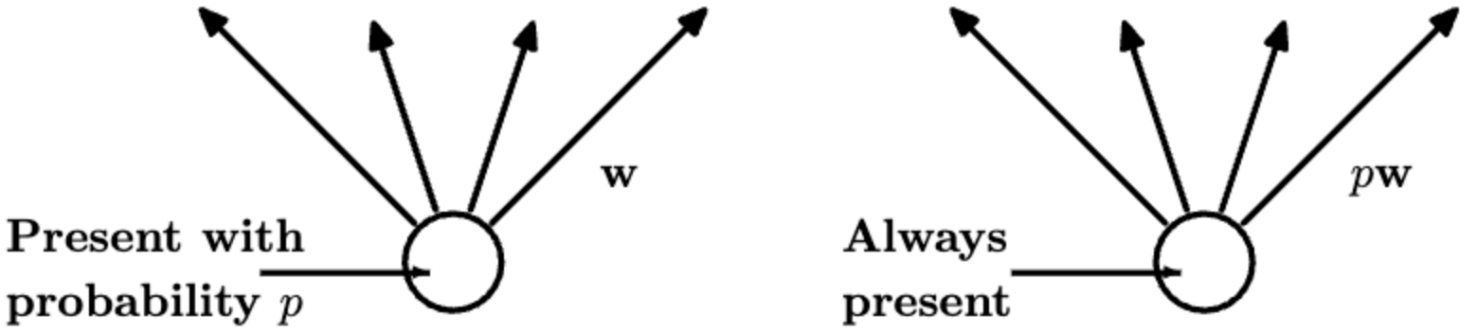


Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." *Advances in neural information processing systems* 25 (2012).

- Improving neural networks by preventing co-adaptation of feature detectors (2012)
- ImageNet Classification with Deep Convolutional Neural Networks (2012) – **AlexNet**
- Understanding Dropout (2013)
- **Dropout: A Simple Way to Prevent Neural Networks from Overfitting (2014)**
- The dropout learning algorithm (2014)

Dropout

- Regularization technique



(a) At training time (b) At test time

Figure 2: **Left**: A unit at training time that is present with probability p and is connected to units in the next layer with weights $\mathbf{w}$. **Right**: At test time, the unit is always present and the weights are multiplied by p. The output at test time is same as the expected output at training time.

Intuition on Dropout

- At training… (as regularization method)
  - Reduce complex **co-adaptations (dependency) between neurons**
  - Regularize (minimal) functional units to include least neurons

- At inference time...
  - “Averaged” model
  - Deterministic form of inference while marginalizing dropout noise
  - Ensemble model, Bagging

Validation methods

- Datasets

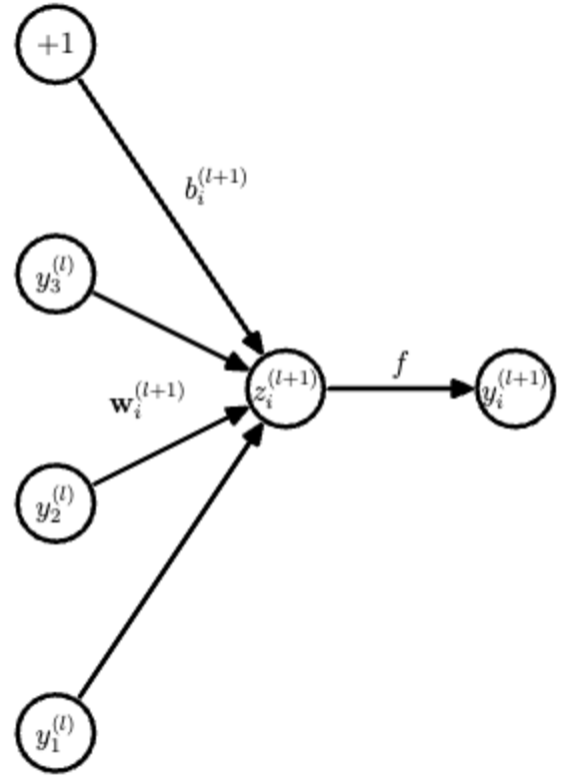
Data Set	Domain	Dimensionality	Training Set	Test Set
MNIST	Vision	784 (28 × 28 grayscale)	60K	10K
SVHN	Vision	3072 (32 × 32 color)	600K	26K
CIFAR-10/100	Vision	3072 (32 × 32 color)	60K	10K
ImageNet (ILSVRC-2012)	Vision	65536 (256 × 256 color)	1.2M	150K
TIMIT	Speech	2520 (120-dim, 21 frames)	1.1M frames	58K frames
Reuters-RCV1	Text	2000	200K	200K
Alternative Splicing	Genetics	1014	2932	733

- Architectures
  - NN
  - CNNs
  - RBM models after self-supervised learning
- Comparison to Bayesian NNs, Weight regularizations

Model description

- Dropout, p = 0.5
  - Standard propagation / backpropagation
  - Halve weights at test time, $W_{test} = pW$
- Max-norm weight matrix regularization
  - $||w||_2 \leq c$, for any weight matrix of hidden unit
  - Normalize if L2 exceeds c
- Large LR with high momentum and LR decay

$$r_j^{(l)} \sim \text{Bernoulli}(p),$$
$$\tilde{\mathbf{y}}^{(l)} = \mathbf{r}^{(l)} * \mathbf{y}^{(l)},$$



MNIST, FCN

6.1.1 MNIST

Method	Unit Type	Architecture	Error %
Standard Neural Net (Simard et al., 2003)	Logistic	2 layers, 800 units	1.60
SVM Gaussian kernel	NA	NA	1.40
Dropout NN	Logistic	3 layers, 1024 units	1.35
Dropout NN	ReLU	3 layers, 1024 units	1.25
Dropout NN + max-norm constraint	ReLU	3 layers, 1024 units	1.06
Dropout NN + max-norm constraint	ReLU	3 layers, 2048 units	1.04
Dropout NN + max-norm constraint	ReLU	2 layers, 4096 units	1.01
Dropout NN + max-norm constraint	ReLU	2 layers, 8192 units	0.95
Dropout NN + max-norm constraint (Goodfellow et al., 2013)	Maxout	2 layers, (5 × 240) units	0.94
DBN + finetuning (Hinton and Salakhutdinov, 2006)	Logistic	500-500-2000	1.18
DBM + finetuning (Salakhutdinov and Hinton, 2009)	Logistic	500-500-2000	0.96
DBN + dropout finetuning	Logistic	500-500-2000	0.92
DBM + dropout finetuning	Logistic	500-500-2000	**0.79**

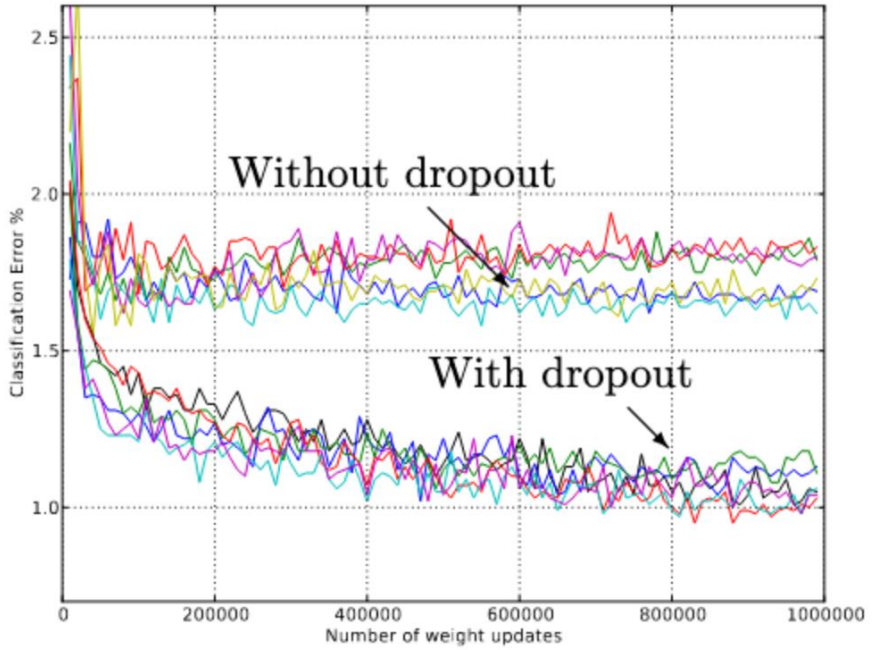


Figure 4: Test error for different architectures with and without dropout. The networks have 2 to 4 hidden layers each with 1024 to 2048 units.

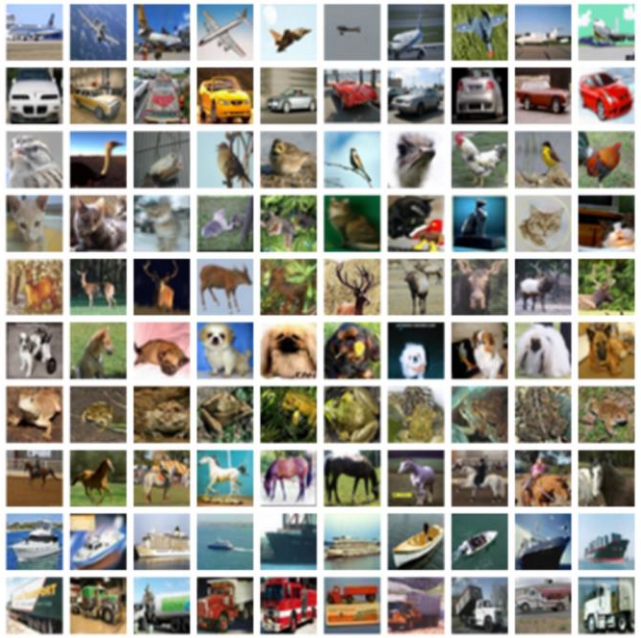
SVHN, CNN

- Dropout to **all** layers : 1-p = (0.9, 0.75, 0.75, 0.5, 0.5, 0.5)

(a) Street View House Numbers (SVHN)

Method	Error %
Binary Features (WDCH) (Netzer et al., 2011)	36.7
HOG (Netzer et al., 2011)	15.0
Stacked Sparse Autoencoders (Netzer et al., 2011)	10.3
KMeans (Netzer et al., 2011)	9.4
Multi-stage Conv Net with average pooling (Sermanet et al., 2012)	9.06
Multi-stage Conv Net + L2 pooling (Sermanet et al., 2012)	5.36
Multi-stage Conv Net + L4 pooling + padding (Sermanet et al., 2012)	4.90
Conv Net + max-pooling	3.95
Conv Net + max pooling + dropout in fully connected layers	3.02
Conv Net + stochastic pooling (Zeiler and Fergus, 2013)	2.80
Conv Net + max pooling + dropout in all layers	2.55
Conv Net + maxout (Goodfellow et al., 2013)	**2.47**
Human Performance	2.0

CIFAR, CNN



(b) CIFAR-10

Method	CIFAR-10	CIFAR-100
Conv Net + max pooling (hand tuned)	15.60	43.48
Conv Net + stochastic pooling (Zeiler and Fergus, 2013)	15.13	42.51
Conv Net + max pooling (Snoek et al., 2012)	14.98	-
Conv Net + max pooling + dropout fully connected layers	14.32	41.26
Conv Net + max pooling + dropout in all layers	12.61	**37.20**
Conv Net + maxout (Goodfellow et al., 2013)	**11.68**	38.57

Table 4: Error rates on CIFAR-10 and CIFAR-100.

ImageNet-2010, CNN(AlexNet)

Model	Top-1	Top-5
Sparse Coding (Lin et al., 2010)	47.1	28.2
SIFT + Fisher Vectors (Sanchez and Perronnin, 2011)	45.7	25.7
Conv Net + dropout (Krizhevsky et al., 2012)	37.5	17.0

Table 5: Results on the ILSVRC-2010 test set.

Model	Top-1 (val)	Top-5 (val)	Top-5 (test)
SVM on Fisher Vectors of Dense SIFT and Color Statistics	-	-	27.3
Avg of classifiers over FVs of SIFT, LBP, GIST and CSIFT	-	-	26.2
Conv Net + dropout (Krizhevsky et al., 2012)	40.7	18.2	-
Avg of 5 Conv Nets + dropout (Krizhevsky et al., 2012)	38.1	16.4	16.4

Table 6: Results on the ILSVRC-2012 validation/test set.

Regularization methods, MNIST

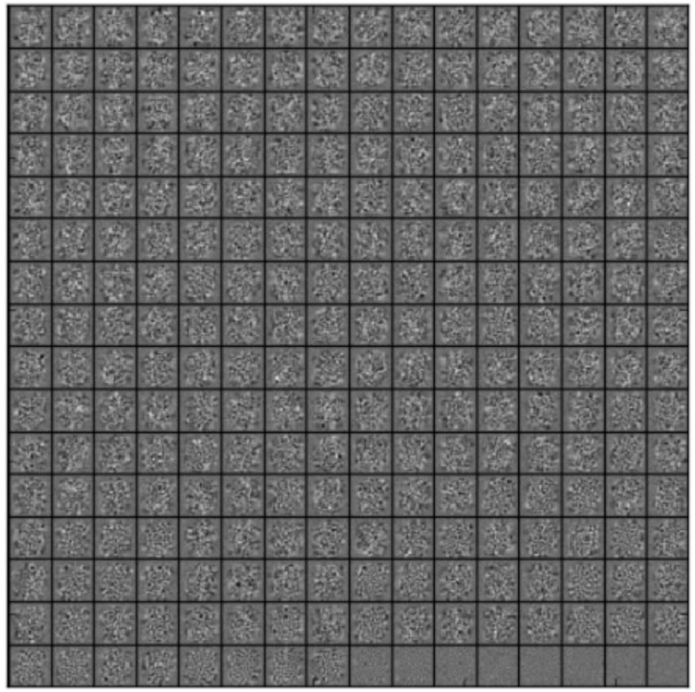
- FCN of dimensions; 784 – 1024 – 1024 – 2048 – 10
- Dropout v.s. L2, L1, KL-sparsity, Max-norm

Method	Test Classification error %
L2	1.62
L2 + L1 applied towards the end of training	1.60
L2 + KL-sparsity	1.55
Max-norm	1.35
Dropout + L2	1.25
Dropout + Max-norm	**1.05**

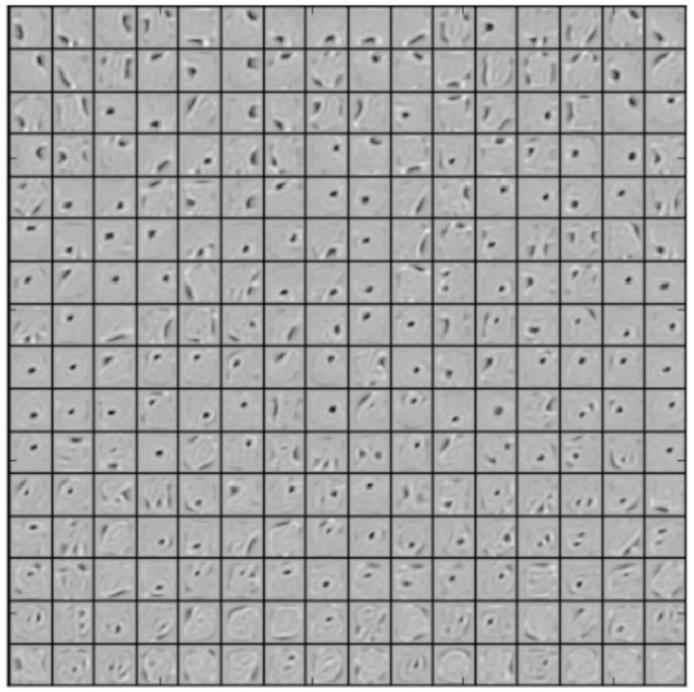
Table 9: Comparison of different regularization methods on MNIST.

Saliency Analysis

- MNIST, Autoencoder (2-layer, 256 dim)
  - Co-adaptations: units may change in a way that they fix up the mistakes of the other units



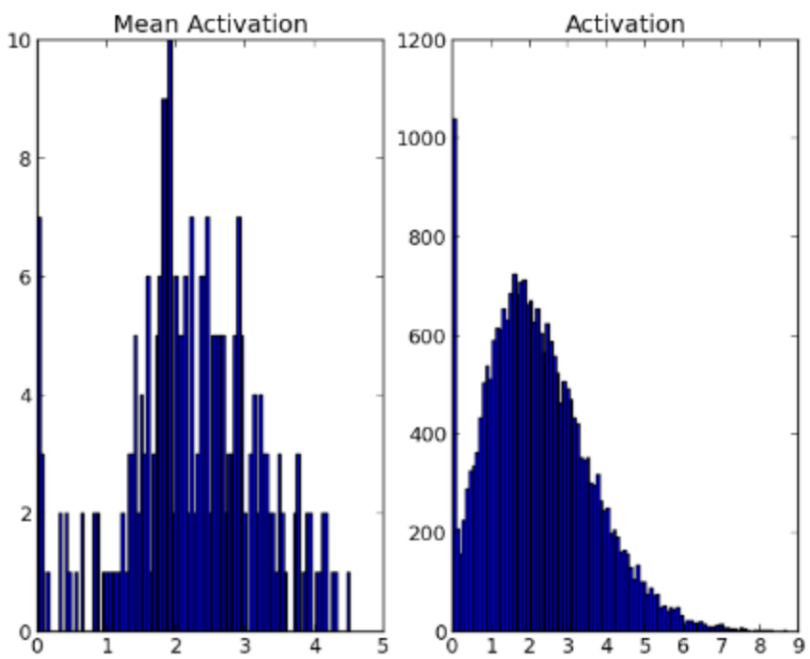
(a) Without dropout



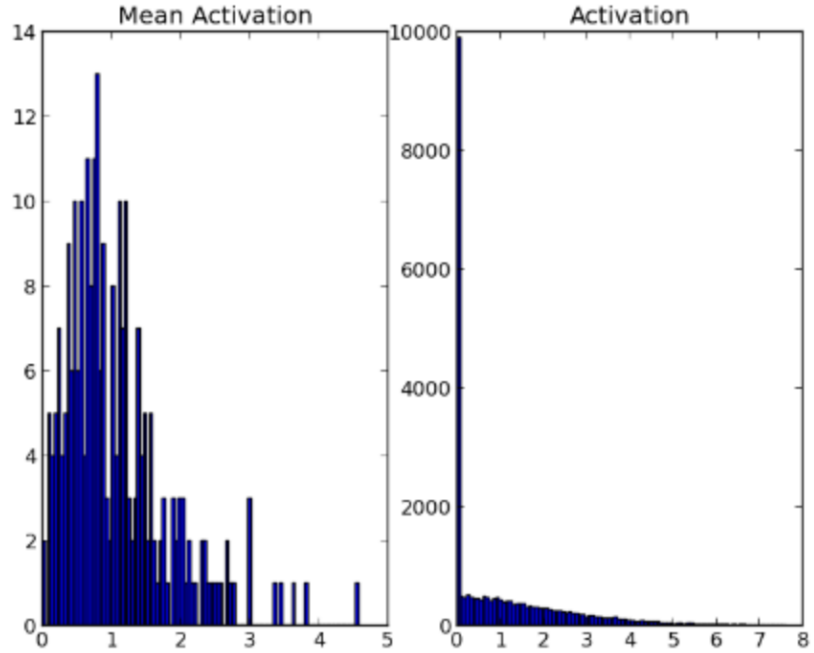
(b) Dropout with $p = 0.5$.

Saliency Analysis

- MNIST, Autoencoder (2-layer, 256 dim)
  - Activation of hidden units (across minibatch, across whole data)



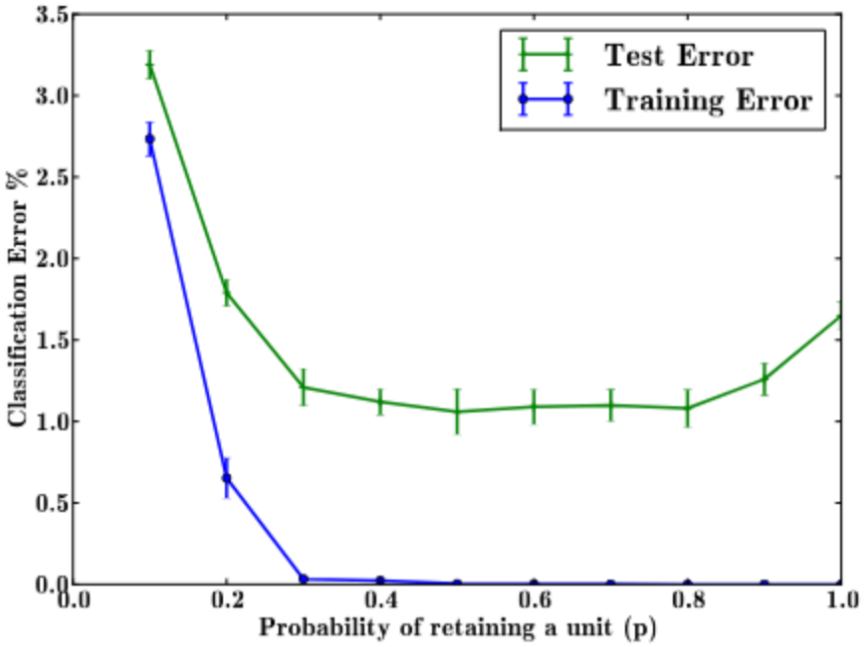
(a) Without dropout



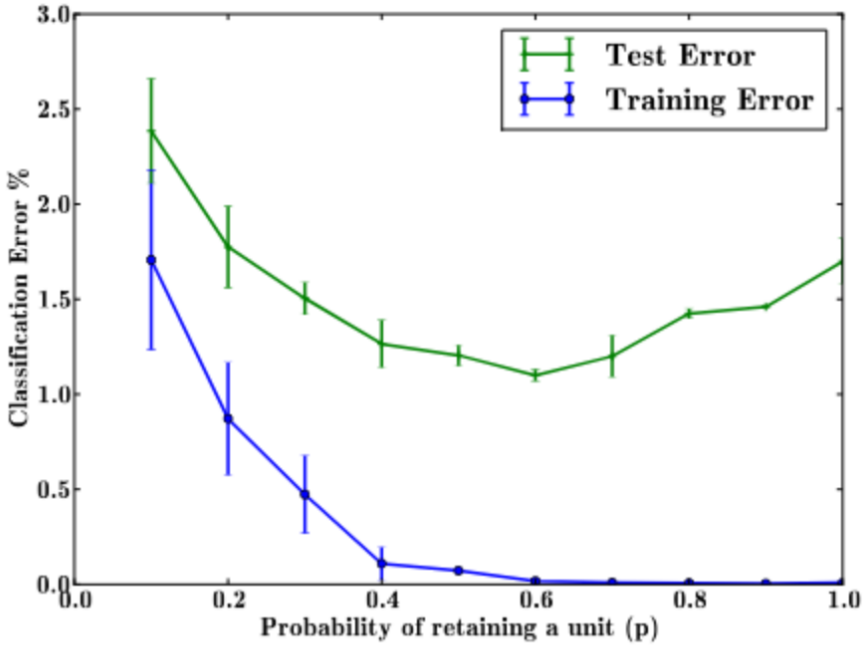
(b) Dropout with $p = 0.5$.

Optimizing Dropout Rate

- MNIST, FCN of dimensions; 784 – 2048 – 2048 – 2048 – 10
  - Fix $pn = 256$ for hidden layer 1, 2 and $pn = 512$ for layer 3



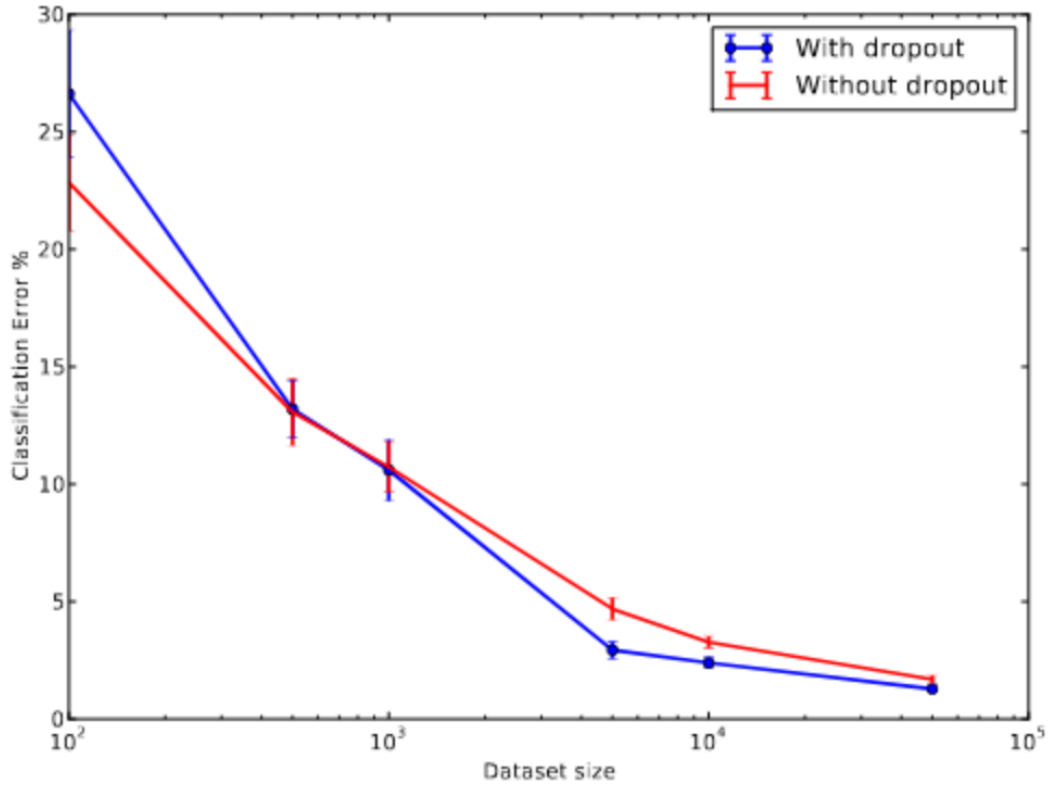
(a) Keeping n fixed.



(b) Keeping pn fixed.

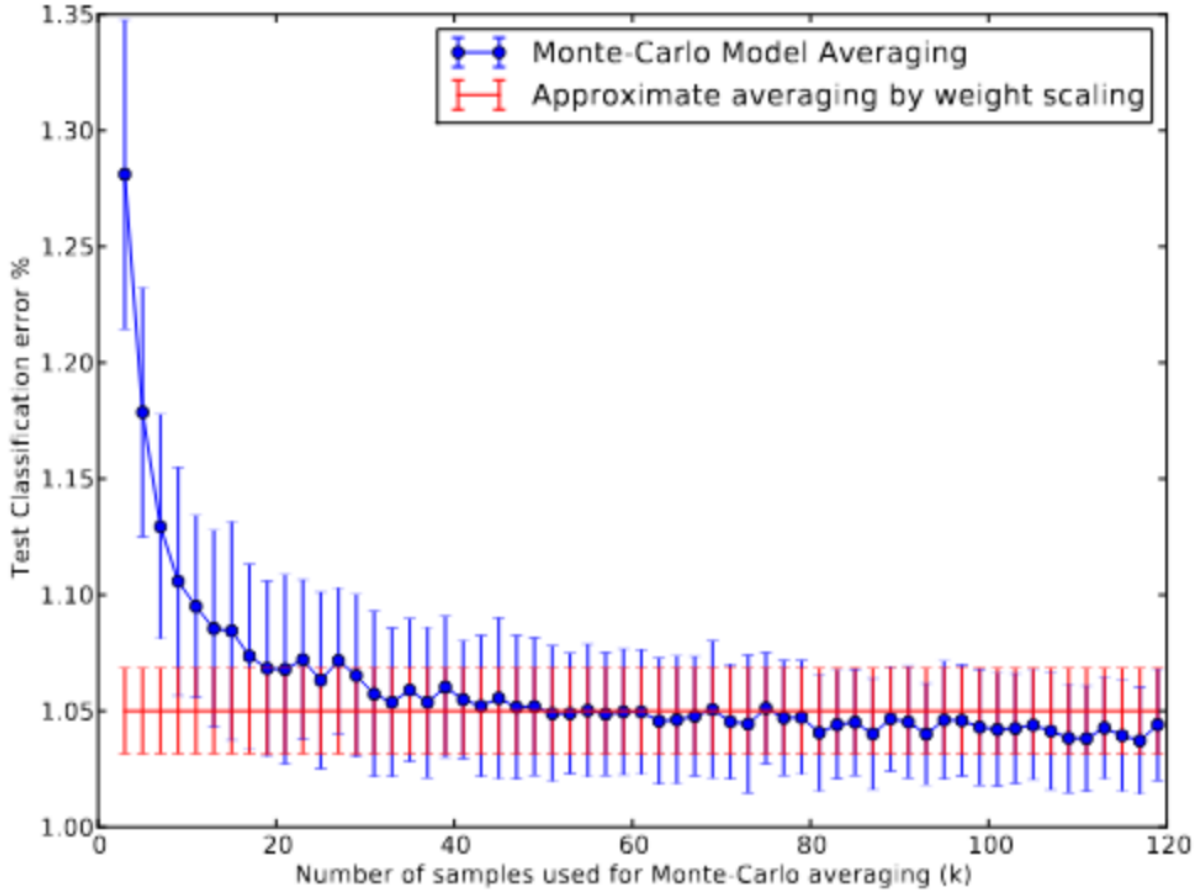
Scalability on Dataset size

- MNIST, training set size : 100, 500, 1K, 5K, 10K, 50K
- FCN, 784 – 2048 – 2048 – 2048 – 10, Dropout 1-p = 0.5



Monte-Carlo Dropout

- Weight scaling vs. averaging outputs (on MNIST)



Generalization to multiplicative noise

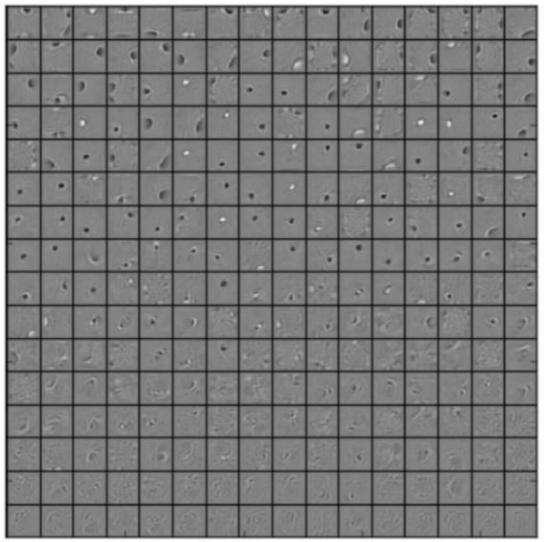
- Dropout = multiplying h_i by Bernoulli noise r
  - Distribution of r can be generalized to other distributions, $e.g.\ r \sim N(1, \sigma^2)$

Data Set	Architecture	Bernoulli dropout	Gaussian dropout
MNIST	2 layers, 1024 units each	1.08 ± 0.04	0.95 ± 0.04
CIFAR-10	3 conv + 2 fully connected layers	12.6 ± 0.1	12.5 ± 0.1

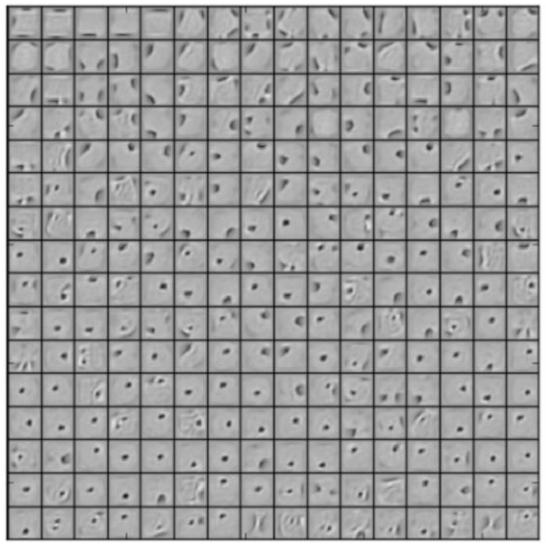
Table 10: Comparison of classification error % with Bernoulli and Gaussian dropout. For MNIST, the Bernoulli model uses $p = 0.5$ for the hidden units and $p = 0.8$ for the input units. For CIFAR-10, we use $p = (0.9, 0.75, 0.75, 0.5, 0.5, 0.5)$ going from the input layer to the top. The value of σ for the Gaussian dropout models was set to be $\sqrt{\frac{1-p}{p}}$. Results were averaged over 10 different random seeds.

Dropout in RBMs

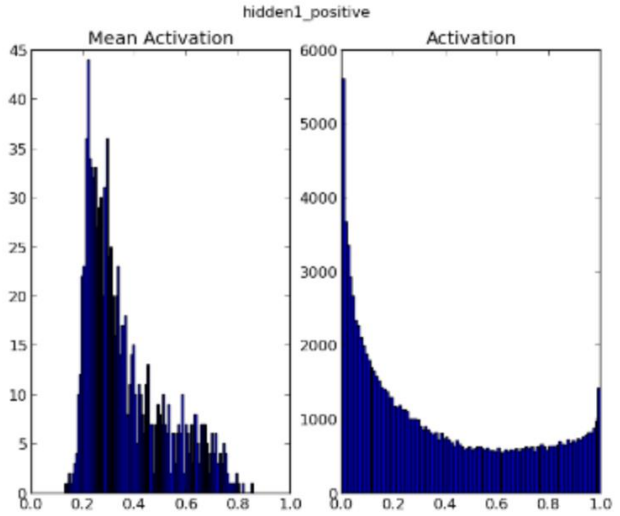
- Restricted Boltzmann Machine
  - Dropout p=0.5 in hidden units



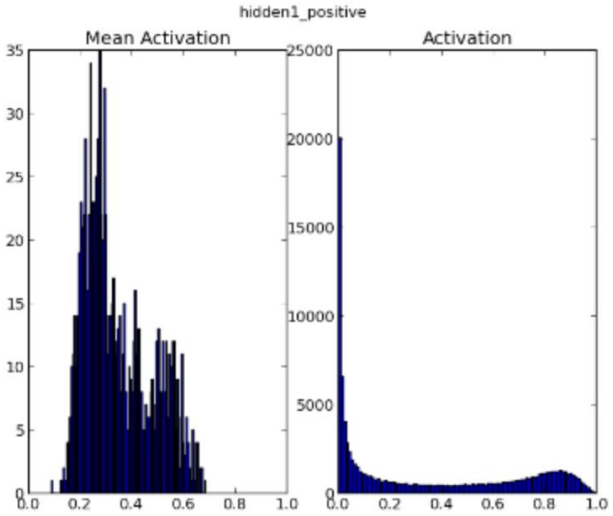
(a) Without dropout



(b) Dropout with $p = 0.5$.



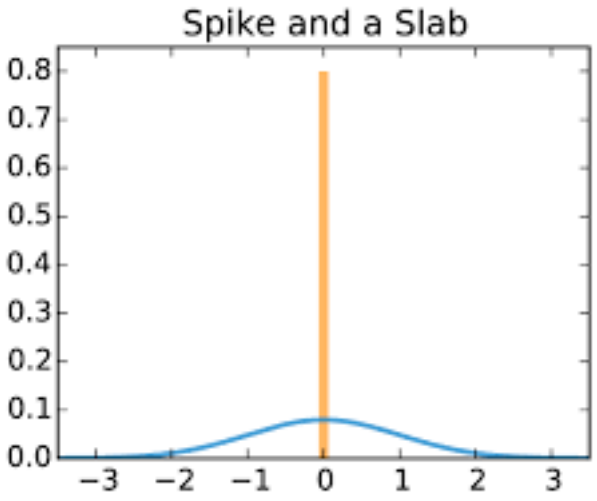
(a) Without dropout

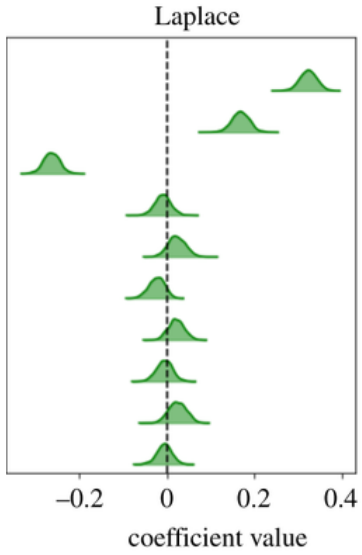


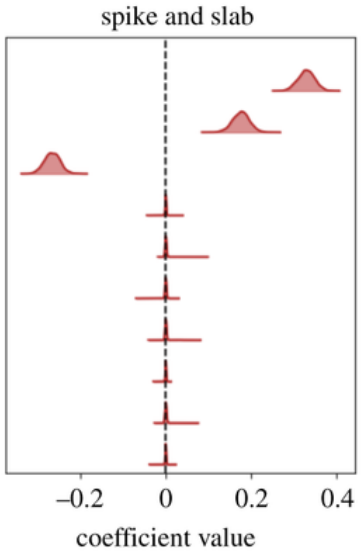
(b) Dropout with $p = 0.5$.

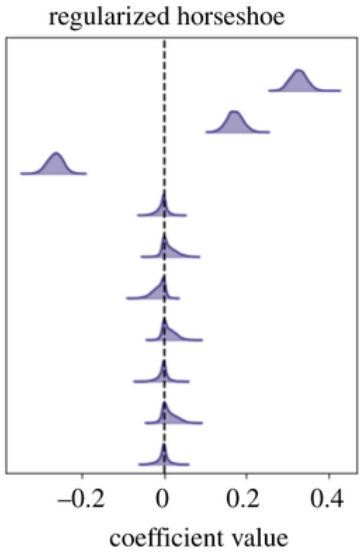
Interpretations

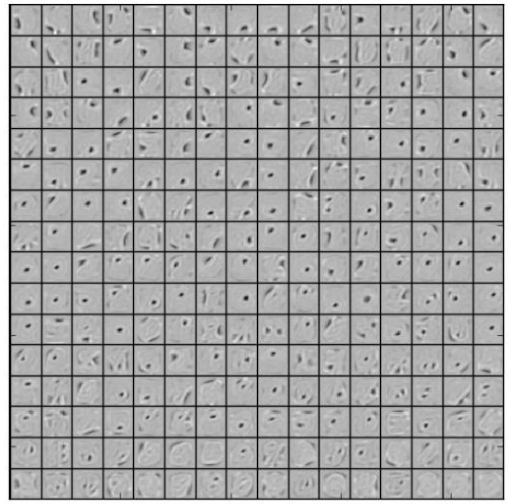
- Bayesian prior on weights
  - Gaussian scale mixture prior with Spike-and-Slab hyperprior
  - “automatic relevance determination”, sparsity constraint











(b) Dropout with $p = 0.5$.

Nalisnick, Eric, José Miguel Hernández-Lobato, and Padhraic Smyth. "Dropout as a structured shrinkage prior." *International Conference on Machine Learning*. PMLR, 2019.

In CNNs

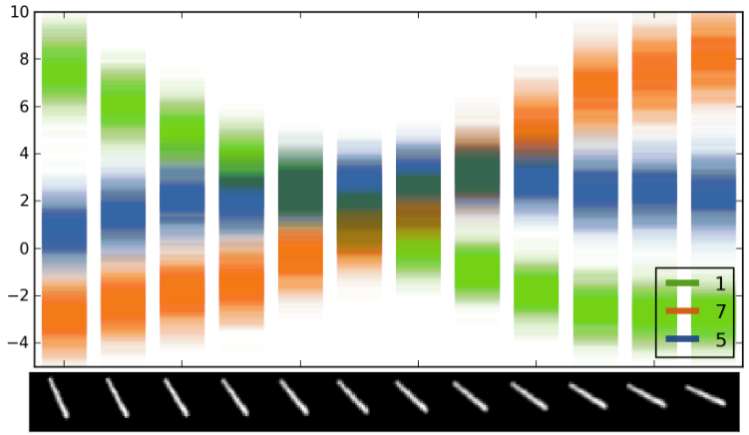
- Dropout is preferred in various architectures…
- Caveats in CNNs
  - Application to Convolution Kernels are questionable
    - Dropout *usually* adopted in FCN layers (VGGNet, AlexNet)
  - Concurrent use with BatchNorm
    - *“Understanding the Disharmony between Dropout and Batch Normalization by Variance Shift”*
- Considerations
  - Total parameters
  - Locality of neurons, Interference to inducted bias
- DropBlock, SpatialDrop …

Li, Xiang, et al. "Understanding the disharmony between dropout and batch normalization by variance shift." *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019.
Ghiasi, Golnaz, Tsung-Yi Lin, and Quoc V. Le. "Dropblock: A regularization method for convolutional networks." *Advances in neural information processing systems* 31 (2018).
Tompson, Jonathan, et al. "Efficient object localization using convolutional networks." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015.

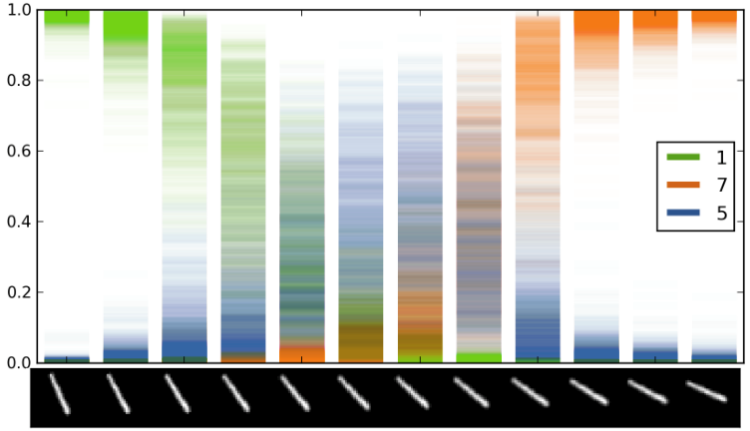
Further Readings

- Variational Interpretation / Model uncertainty
  - **Monte-Carlo Dropout**
  - Bayesian approximation of Gaussian Process

$$p(\boldsymbol{y}|\boldsymbol{x},\mathcal{D}) \approx \frac{1}{S}\sum_{s=1}^{S} p(\boldsymbol{y}|\boldsymbol{x},\hat{\mathbf{W}}\epsilon^s + \hat{\boldsymbol{b}})$$



(a) Softmax *input* scatter



(b) Softmax *output* scatter

Gal, Yarin, and Zoubin Ghahramani. "Dropout as a bayesian approximation: Representing model uncertainty in deep learning." *international conference on machine learning*. PMLR, 2016.

Further Readings

- Dropout at early iterations prevent underfitting

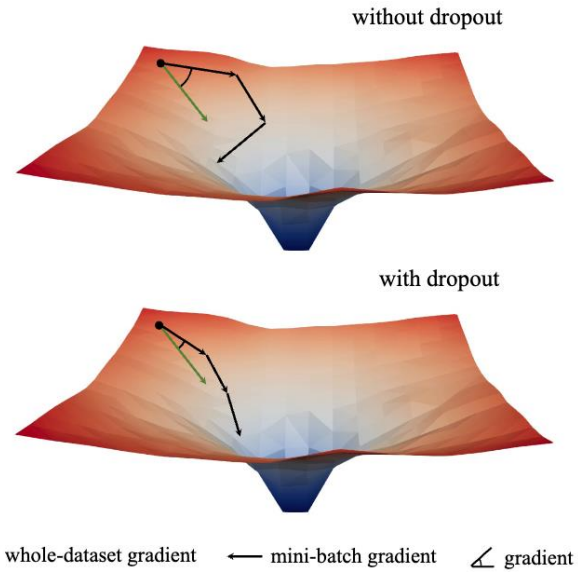


Figure 1. **Dropout in early training** helps the model produce mini-batch gradient directions that are more consistent and aligned with the overall gradient of the entire dataset.

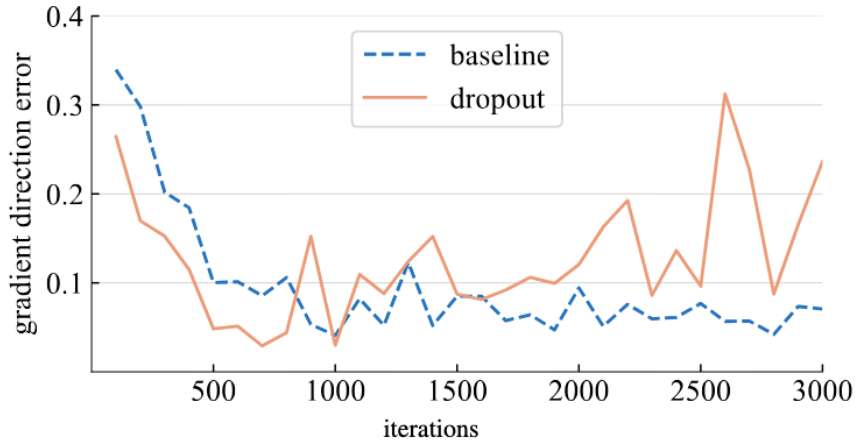


Figure 8. **Gradient direction error**. Dropout leads to mini-batch gradients that are more aligned with the gradient of the entire dataset at the beginning of training.

model	top-1 acc.	change	train loss	change
ViT-T†	72.8	-	-	-
ViT-T‡	75.5	-	-	-
ViT-T	76.3	-	3.033	-
+ standard dropout	71.5	↓4.8	3.437	↑0.404
+ standard s.d.	75.6	↓0.7	3.243	↑0.210
+ early dropout	**76.7**	↑0.4	2.991	↓0.042
+ early s.d.	**76.7**	↑0.4	3.022	↓0.011
ConvNeXt-F‡	77.5	-	-	-
ConvNeXt-F	77.5	-	3.011	-
+ standard s.d.	77.4	↓0.1	3.177	↑0.166
+ early s.d.	**77.7**	↑0.2	2.990	↓0.021

Liu, Zhuang, et al. "Dropout reduces underfitting." *International Conference on Machine Learning*. PMLR, 2023.