Regularization for Deep Learning: The Dropout Method

Discussant: Javier Zapata

Updated: 2018/03/02

Today's plan

1. Quick Detour: Regularization in Stats
2. Some mathematical notation
3. Dropout definition & training with SGD
4. Review of popular Image Datasets in Deep Learning
5. Dropout: Experimental Results
6. Dropconnect definition & comparison with Dropout
7. Experimental results comparing Dropout and Dropconnect
8. Computational challenges of using Dropconnect

Let's quickly remember how we regularize in Stats...

Regularization in Statistics

Some **canonical regularization for least squares**:

Lagrangian form:

$$\min_{\beta \in \mathbb{R}^p} \|y - X\beta\|_2^2 + \underbrace{\lambda}_{\text{regularization parameter}} \|\beta\|_q$$

Constrained form:

$$\min_{\beta \in \mathbb{R}^p} \|y - X\beta\|_2^2 \quad \text{subject to } \|\beta\|_q \leq \underbrace{t}_{\text{tuning parameter}}$$

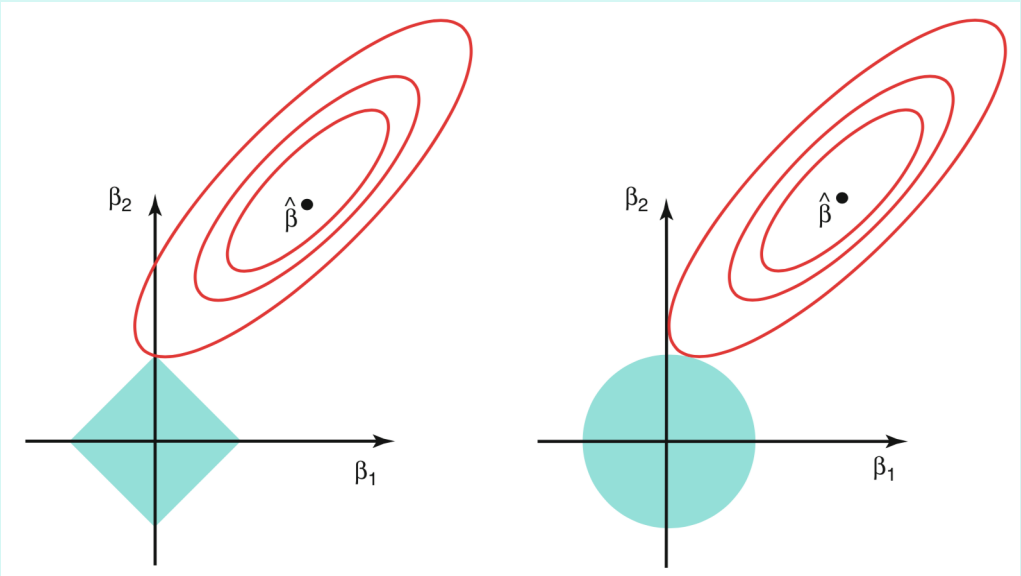


Figure: *(Left)* **LASSO** $(q = 1)$ *(Right)* **Ridge** $(q = 2)$ ▶

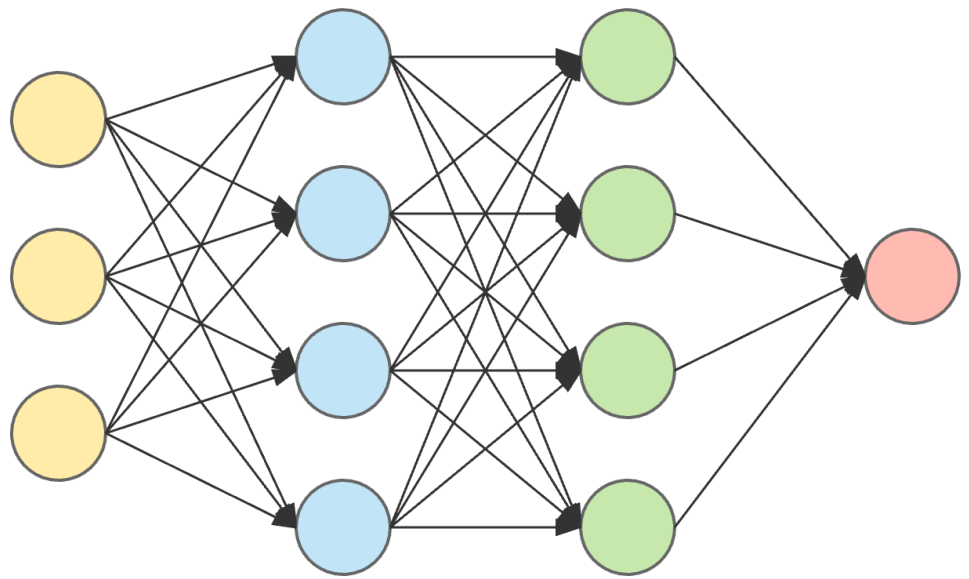
Long story short: the choice of λ (or t) defines which $\beta_i = 0$

Back to Deep Learning...

Some notation first ☺

Neural Net (NN) & Deep (Feedforward) NN

- ◎ $L :=$ # of **layers**, $x :=$ **input layer**, $y :=$ **output layer**
- ◎ $\sigma(\cdot) : \mathbb{R} \to \mathbb{R} :=$ the **activation function** (componentwise ☺)
- ◎ $W_\ell(\cdot) := \mathbf{A}^\ell(\cdot) + b^\ell$, i.e., an **affine transformation** $\mathbb{R}^{N_{\ell-1}} \to \mathbb{R}^{N_\ell}$
- ◎ $F := W_L \circ F_{L-1} \circ \cdots \circ F_2 \circ F_1$ is a **feed forward Deep NN**
  with $F_1, \ldots, F_{L-1}$ the **hidden units** defined as:
  $F_\ell := \sigma \circ W_\ell(F_{\ell-1}) = \sigma\left(\mathbf{A}^\ell F_{\ell-1} + b^\ell\right)$ for $\ell = 1, \ldots, L-1$



What's new with Dropout? (Srivastava et. al. ,2014)

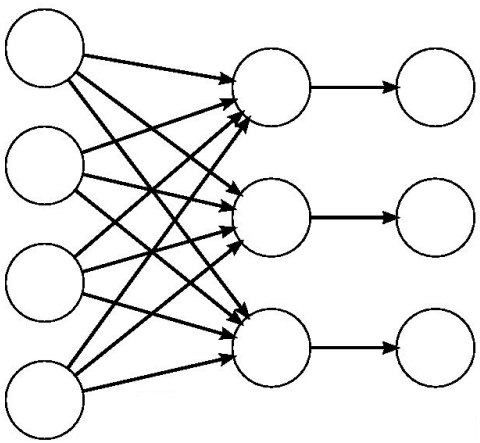
At every iteration of the stochastic gradient descent, we set some hidden units to zero. In other words, we sample a thinned network by dropping out units.

Dropout: regularizes a layer before the activation takes place

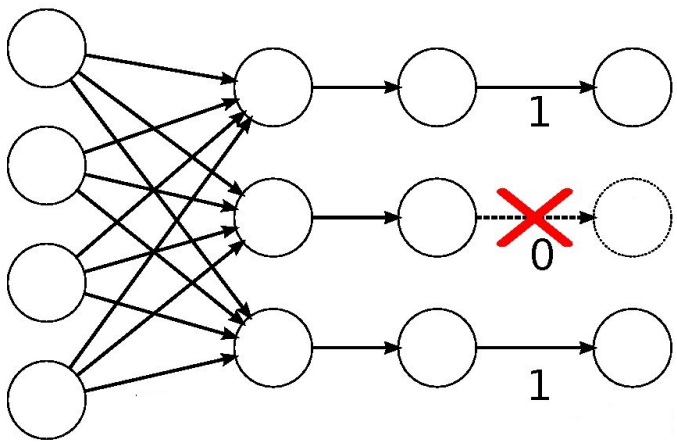
Let layer ℓ have size n, then the hidden unit F_ℓ is:

- without dropout: $F_\ell := \sigma\left(\mathbf{A}^\ell F_{\ell-1} + b^\ell\right) \in \mathbb{R}^n$
- with dropout: $F_\ell := (r_1, \ldots, r_n) \odot \sigma\left(\mathbf{A}^\ell F_{\ell-1} + b^\ell\right) \in \mathbb{R}^n$

  with $r_i \overset{iid}{\sim} Be(p)$ for $i = 1, .., n$ and p:= **dropout rate**

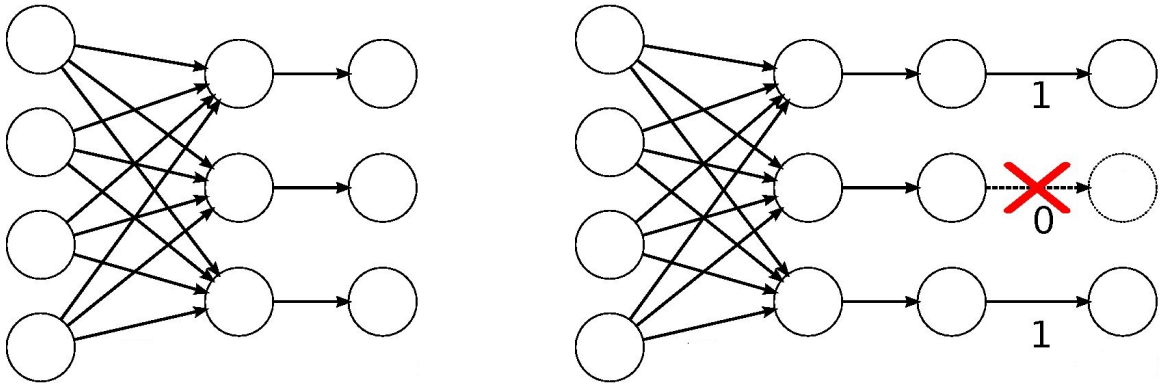


no dropout



with dropout

Dropout: Backpropagation



no dropout with dropout

- Forward and backpropagation for that training case are done only on the thinned network.
- For learning, the derivatives of the loss function are backpropagated through the thinned network.
- The gradients for each parameter are averaged over the training cases in each mini-batch. Any training case which does not use a parameter contributes a gradient of zero for that parameter.
- At test time, the weights are scaled: $A^{\ell}_{\text{test}} = pA^{\ell} = E_{\text{dropout}}[A^{\ell}]$

Experimental Results on Image Data Sets

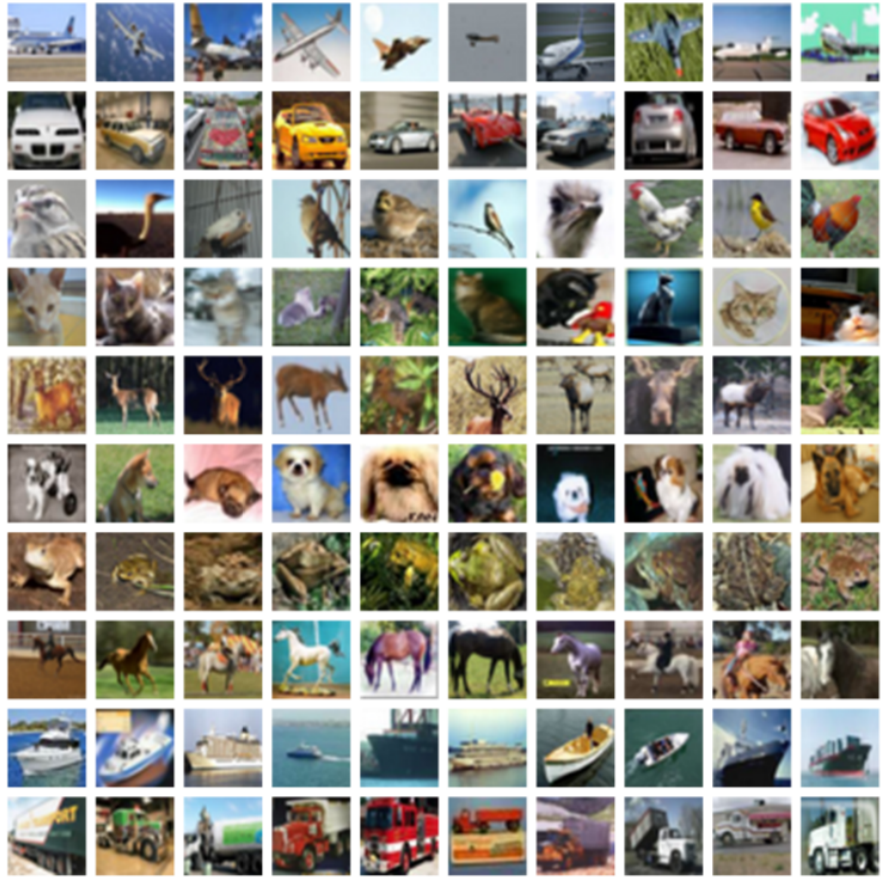
They obtain successful results with popular classification image datasets:

- **MNIST**: A standard toy data set of handwritten digits.
- **TIMIT**: A standard speech benchmark for clean speech recognition.
- **CIFAR-10** and **CIFAR-100**: Tiny natural images.
- **Street View House Numbers data set (SVHN)**: Images of house numbers collected by Google Street View.
- **ImageNet** : A large collection of natural images.
- **Reuters-RCV1**: A collection of Reuters newswire articles.
- **NORB** a collection of stereo images of 3D models.

more about Image Data Sets ►

Experimental Results on Image Data Sets

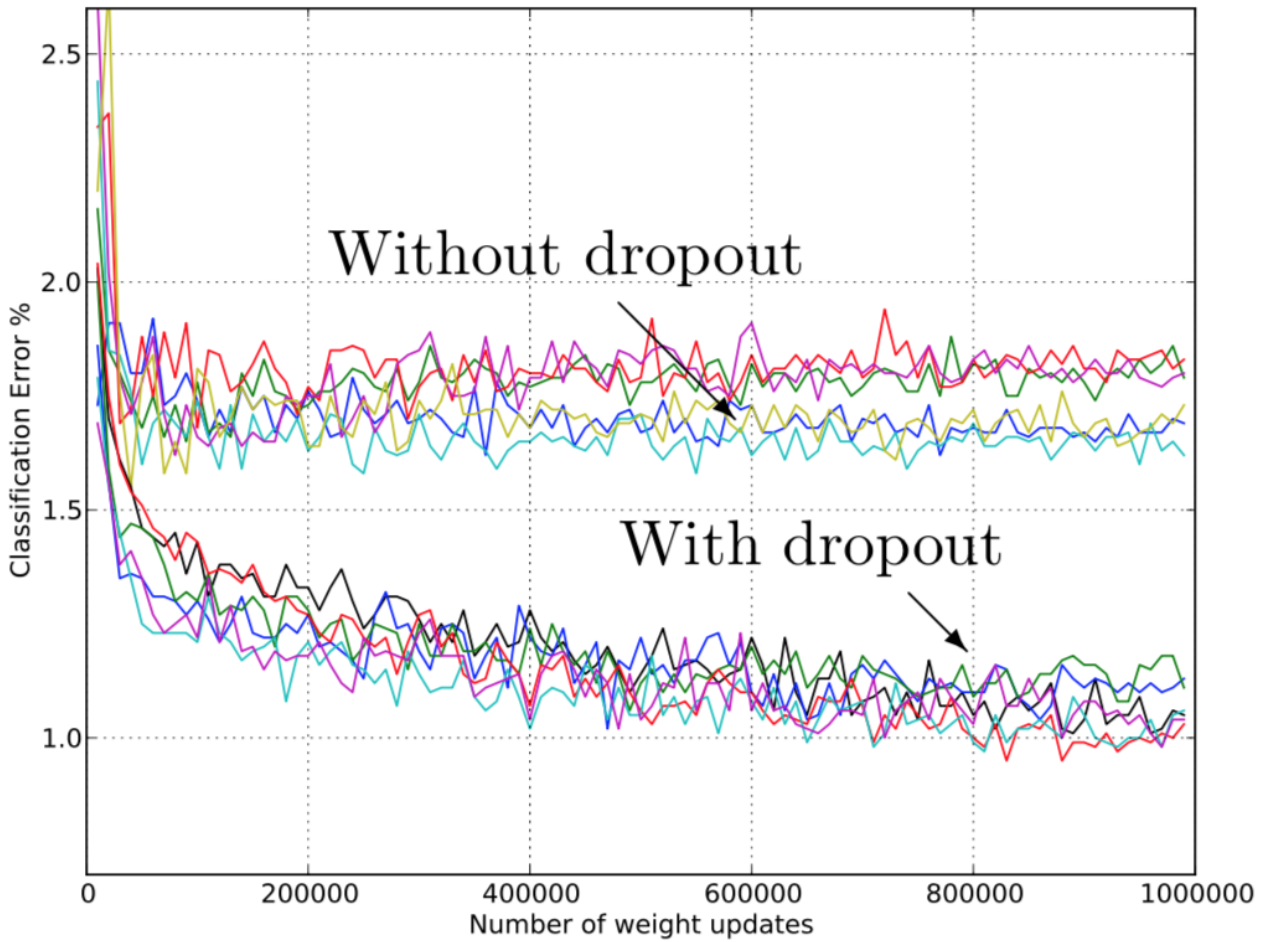
(a) Street View House Numbers (SVHN)



(b) CIFAR-10

Results on Image Data Set (MNIST)

The **MNIST** data set consists of 28 x 28 pixel handwritten digit images. The task is to classify the images into 10 digit classes.



"Dropout gives a huge improvement across all architectures, without using hyperparameters that were tuned specifically for each architecture"

Results on Image Data Set (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**

"...using dropout along with max-norm regularization... provides a significant boost over just using dropout."

"A possible justification is that constraining weight vectors to lie inside a ball of fixed radius makes it possible to use a huge learning rate without the possibility of weights blowing up. The noise provided by dropout then allows the optimization process to explore different regions of the weight space that would have otherwise been difficult to reach."

Dropout vs Standard Regularizers (MNIST)

NN trained using SGD with different regularizations.

- **Network architecture:** 784-1024-1024-2048-10
- **Activation function:** ReLU ($\sigma(x) = max\{0, x\}$)

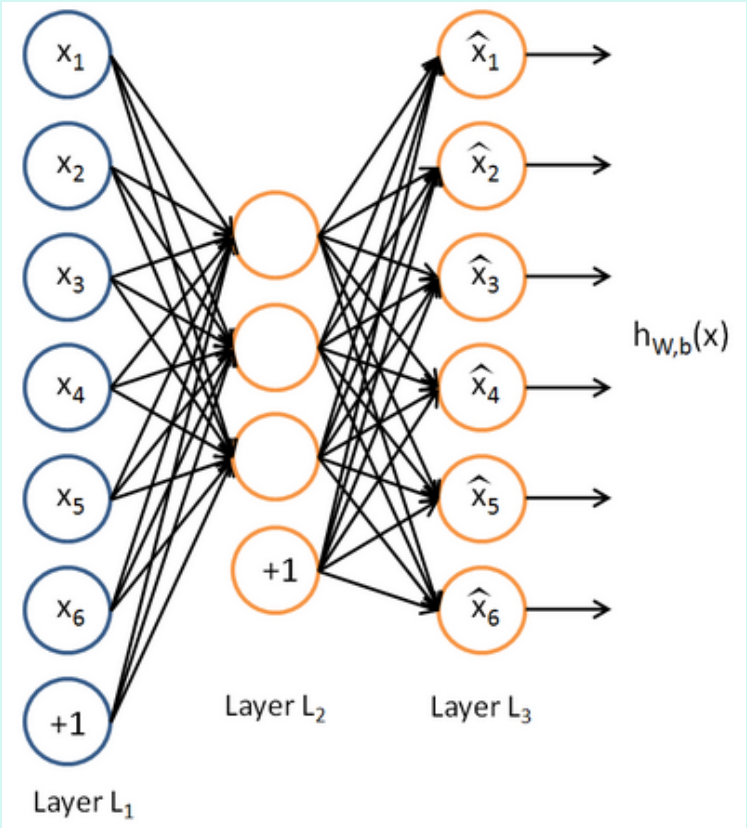
Method	**Test Classification error %**
L2	1.62
L2 + L1 applied towards the end of training	1.60
L2 + KL-sparsity	1.55
Max-norm	1.35
Dropout + L2	1.25
Dropout + Max-norm	**1.05**

Let's quickly define an **Autoencoder NN**

Autoencoder NN

An **Autoencoder NN** is a NN that sets the output layer to be equal to the input layer ($y := x$), and tries to learn an approximation to the identity function.

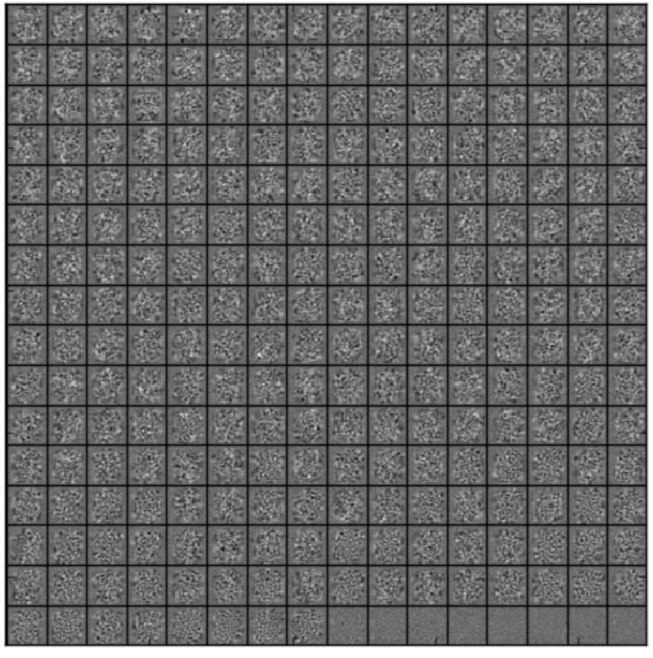
By placing constraints on the network, we can discover interesting structure about the data.



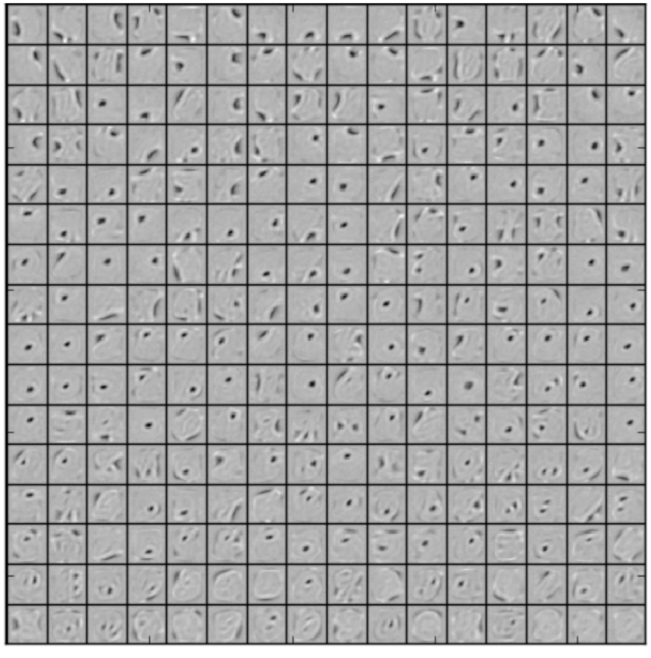
Source:
http://ufldl.stanford.edu/wiki/index.php/Autoencoders_and_Sparsity

Salient Features of Dropout: Breaks co-adaptation

- **co-adaptations**:=hidden units may change in a way that they fix up the mistakes of the other units.
- *"...for each hidden unit, dropout prevents co-adaptation by making the presence of other hidden units unreliable. Therefore, a hidden unit ... must perform well in a wide variety of different contexts provided by the other hidden units"*
- Figure shows features learned by an autoencoder on MNIST with a single hidden layer of 256 rectified linear units without dropout



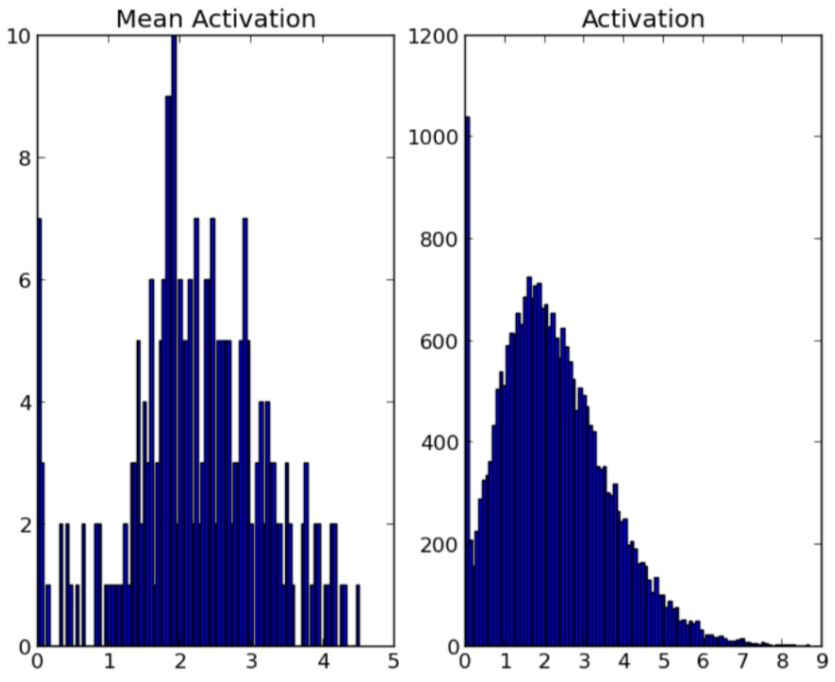
(a) Without dropout



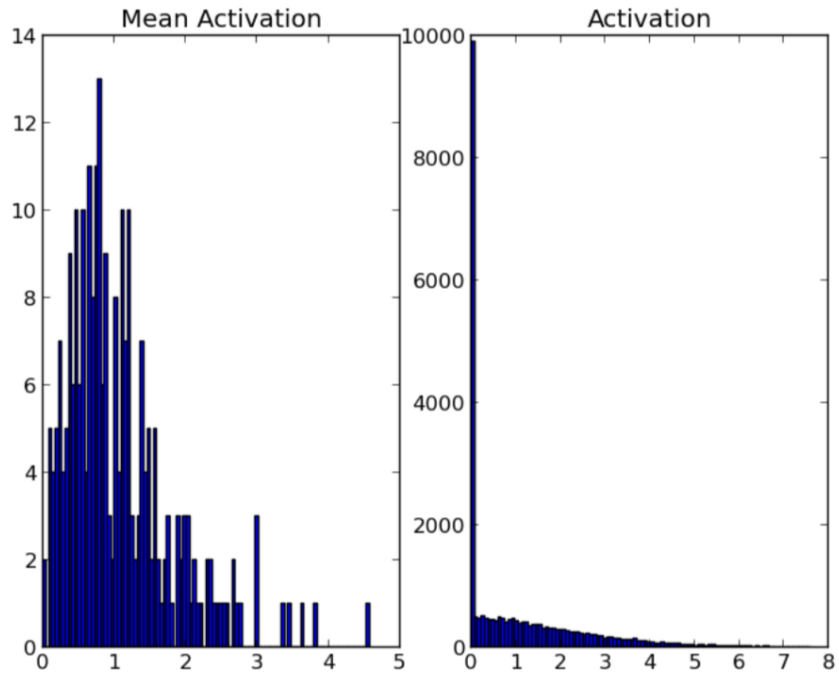
(b) Dropout with $p = 0.5$.

Salient Features of Dropout: Sparsity

- *"In a good sparse model, there should only be a few highly activated units for any data case. Moreover, the average activation of any unit across data cases should be low."*
- For each model, the histogram on the left shows the distribution of mean activations of hidden units across the minibatch. The histogram on the right shows the distribution of activations of the hidden units.



(a) Without dropout



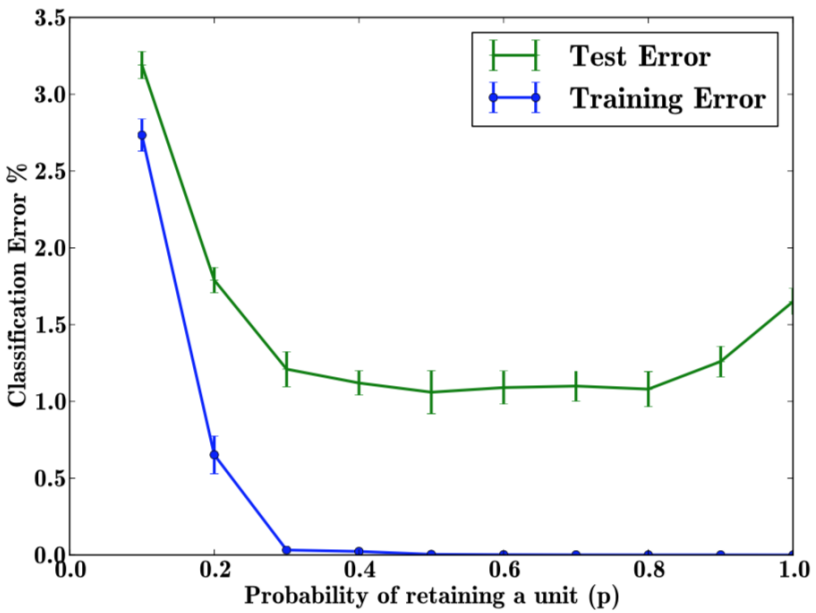
(b) Dropout with $p = 0.5$.

Effect of Dropout Rate and # of Hidden Units

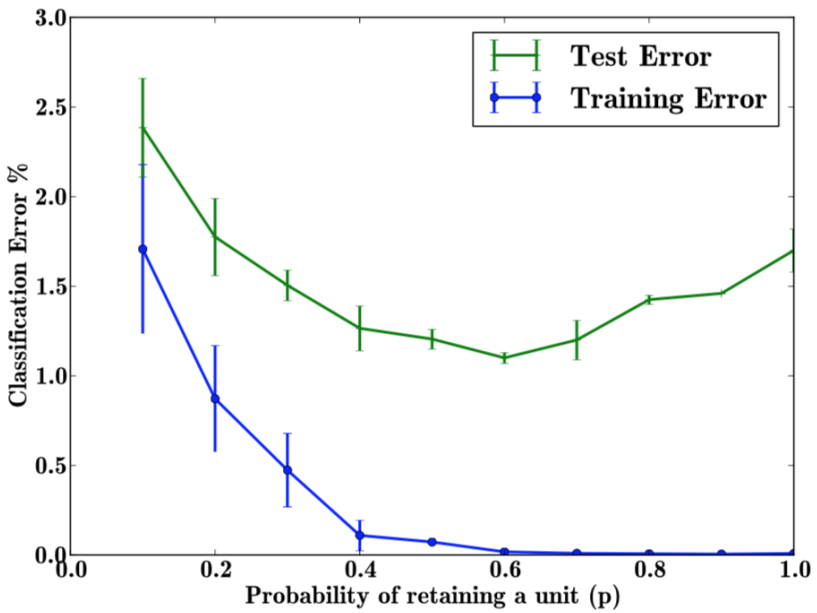
With dropout: $F_\ell := (r_1, \ldots, r_n) \odot \sigma\left(\mathbf{A}^\ell F_{\ell-1} + b^\ell\right) \in \mathbb{R}^n$ with $r_i \overset{iid}{\sim} Be(p)$ for $i = 1, .., n$ and p:= **dropout rate**

Architecture: 784-2048-2048-2048-10

- Case a: $n = (\#$ of hidden units$)$ is fixed
- Case b: $E[\#$ of hidden units$] = p \cdot n$ is fixed



(a) Keeping n fixed.

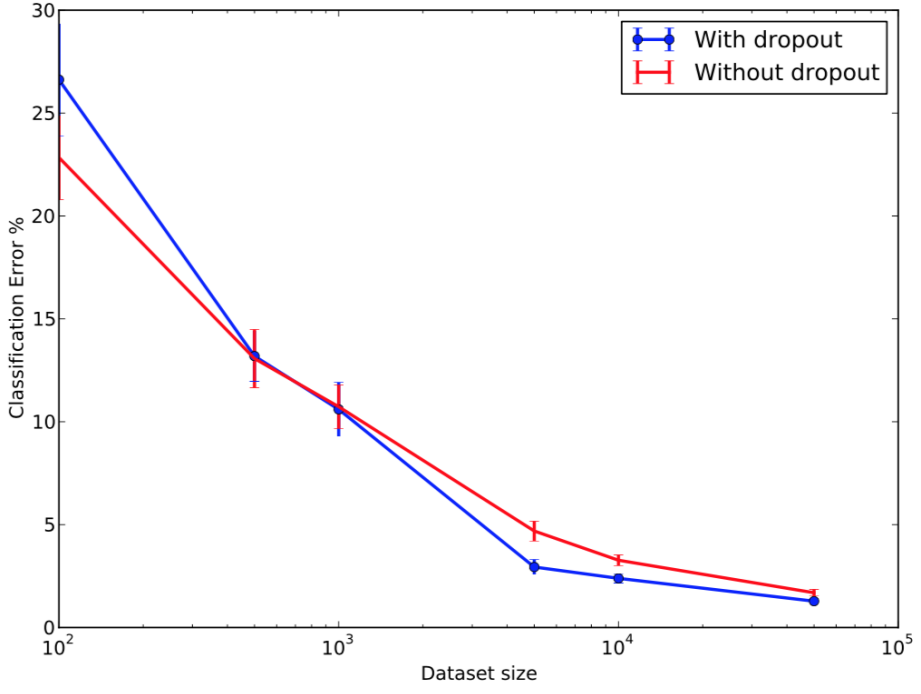


(b) Keeping pn fixed.

We see that as p increases, the error goes down. It becomes flat when $0.4 \leq p \leq 0.8$ and then increases as p becomes close to 1

Dropout: Effect of Data Set Size

- As the size of the data set is increased, the gain from doing dropout increases up to a point and then declines.
- For any given architecture and dropout rate p, there is a "*sweet spot*": some amount of data that is large enough to not be memorized in spite of the noise but not so large that overfitting is not a problem anyways.



Marginalizing Dropout: A deterministic version

Definition: mask

A matrix $\mathbf{R}$ s.t. $[\mathbf{R}]_{ij} \overset{iid}{\sim} Be(p)$ and $p \in (0, 1)$

- Add a mask to least squares: $\min_{\beta} \quad \mathbb{E}_{\mathbf{R}}\left[\|y - \mathbf{R} \odot \mathbf{X}\beta\|^2\right]$
- Equivalently: $\min_{\beta} \quad \|y - p\mathbf{X}\beta\|^2 + p(1-p)\|\Gamma\beta\|^2$

  with $\Gamma = (\text{diag}(\mathbf{X}'\mathbf{X}))^{1/2}$
- Absorb p into β: $\min_{\tilde{\beta}} \quad \|y - \mathbf{X}\tilde{\beta}\|^2 + \frac{(1-p)}{p}\|\Gamma\tilde{\beta}\|^2$
- *"Therefore, dropout with linear regression is equivalent, in expectation, to ridge regression with a particular form for* Γ*"*

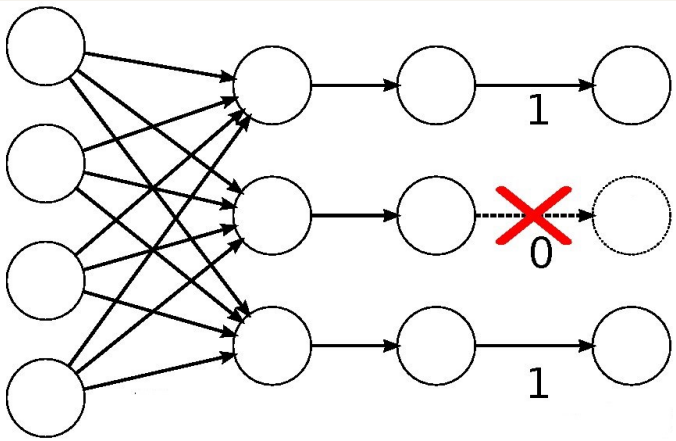
Dropconnect: A generalization of Dropout

Dropout vs Dropconnect (Wan et. al., 2013)

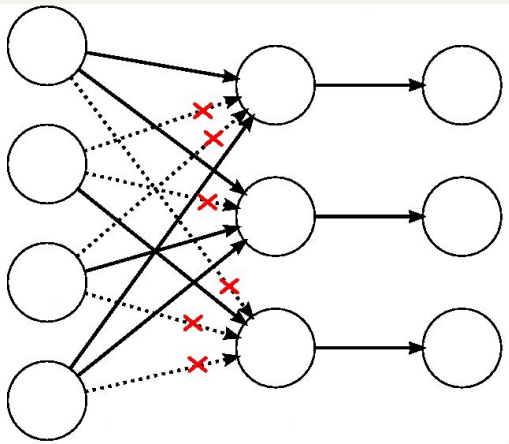
Dropout regularizes layer ℓ after the activation $\sigma(\cdot)$

Dropconnect regularizes layer ℓ before the activation $\sigma(\cdot)$

- **mask**:= matrix $\mathbf{R}$ s.t. $[\mathbf{R}]_{ij} \overset{iid}{\sim} Be(p)$ and $p \in (0, 1)$
- **Dropout**: $F_\ell := (r_1, \ldots, r_n) \odot \sigma\left\{\mathbf{A}^\ell F_{\ell-1} + b^\ell\right\} \in \mathbb{R}^n$, $r_i \overset{iid}{\sim} Be(p)$
- **Dropconnect**: $F_\ell := \sigma\left\{\mathbf{R} \odot \left[\mathbf{A}^\ell; b^\ell\right] \cdot (F_{\ell-1}; 1)^T\right\} \in \mathbb{R}^n$



dropout



dropconnect

Dropconnect vs Dropout: Results on MNIST

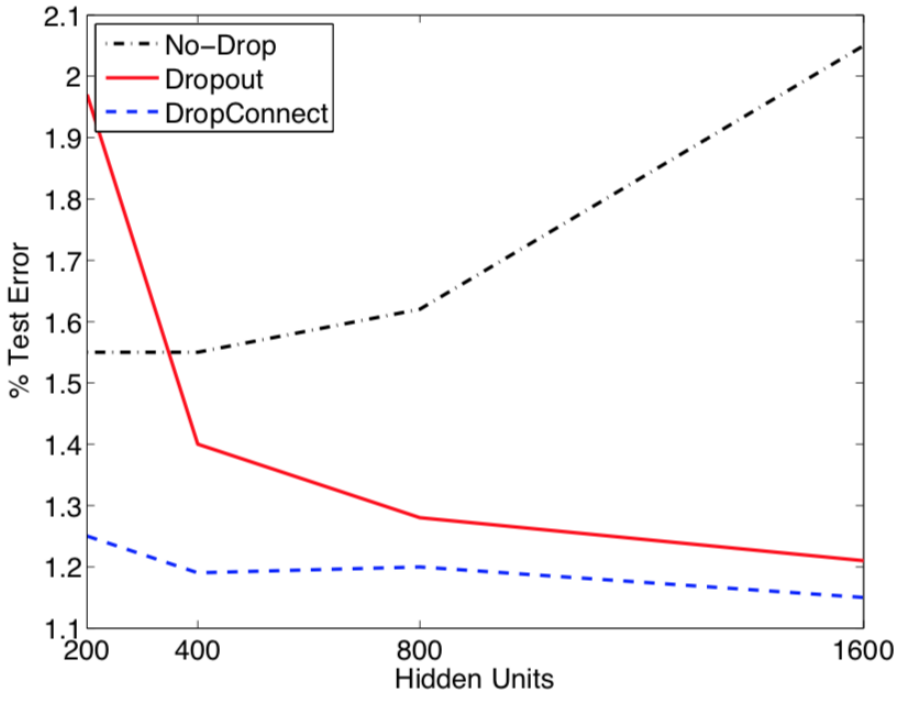
The **MNIST** data set consists of 28 x 28 pixel handwritten digit images. The task is to classify the images into 10 digit classes.

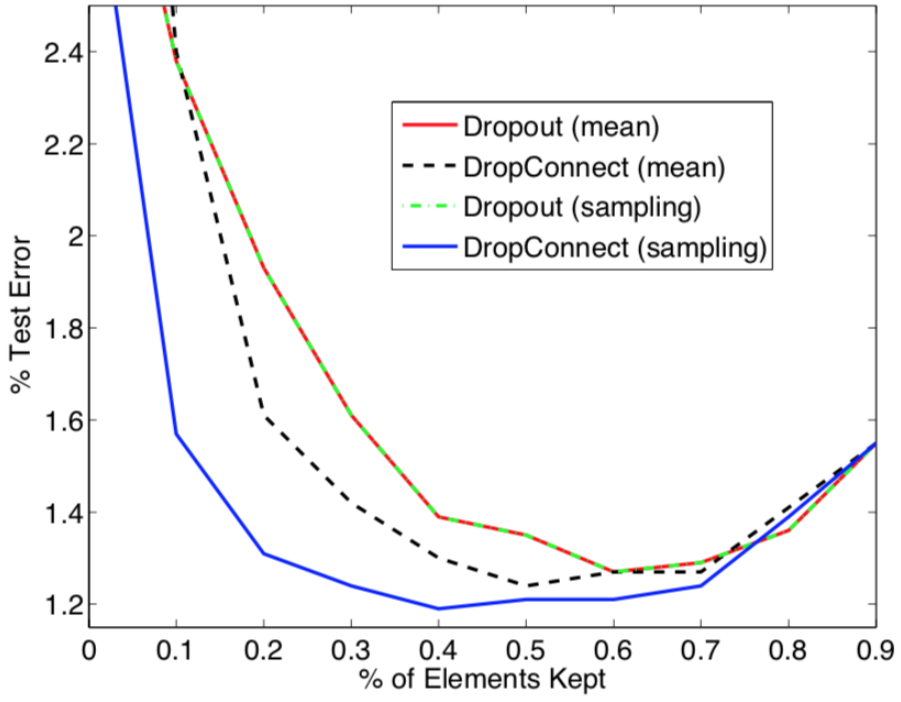
neuron	model	error(%) 5 network	voting error(%)
relu	No-Drop	1.62 ± 0.037	1.40
	Dropout	1.28 ± 0.040	1.20
	DropConnect	1.20 ± 0.034	**1.12**
sigmoid	No-Drop	1.78 ± 0.037	1.74
	Dropout	1.38 ± 0.039	**1.36**
	DropConnect	1.55 ± 0.046	1.48
tanh	No-Drop	1.65 ± 0.026	1.49
	Dropout	1.58 ± 0.053	1.55
	DropConnect	1.36 ± 0.054	**1.35**

Dropconnect vs Dropout: Results on MNIST

The **MNIST** data set consists of 28 x 28 pixel handwritten digit images. The task is to classify the images into 10 digit classes.
Figures below:

- (Left): the ability of Dropout and Dropconnect to prevent overfitting as the size of the 2 fully connected layers increase.
- (Right): Varying the drop-rate in a 400-400 network shows near optimal performance around the $p = 0.5$





Dropconnect vs Dropout: Results on popular Image Datasets

Image Classification Error(%) of DropConnect v.s. Dropout

DataSet	DropConnect	Dropout	Previous best result(2013)
MNIST	**0.21**	0.27	0.23
CIFAR-10	**9.32**	9.83	9.55
SVHN	**1.94**	1.96	2.80
NORB-full-2fold	3.23	**3.03**	3.36

Dropconnect: Computational burdens during Training

Each training example in a **batch** uses a different **mask**. A single mask does not regularize the model enough in practice.
These different masks cause technical difficulties:

1. For a 4096 x 4096 fully connected layer with batch size 128, the matrix would be too large to fit into GPU memory (8 GB)
   Soln: mask matrix elements stored as a single bit to encode the connectivity information rather than as a float.
2. It's not easy to access all the mask elements required during the matrix multiplications so as to maximize performance
   Soln: using an efficient memory access pattern using 2D texture aligned memory..

THE END

BIBLIOGRAPHY

- Gareth,J., Witten, D., Hastie, T., & Tibshirani, R. (2013). An introduction to statistical learning: With applications in R. Springer: Chicago.
- Hastie, T., Tibshirani, R. , & Friedman, J. . (2011) The Elements of Statistical Learning: Data Mining, Inference, and Prediction, 2nd ed., Springer: New York
- Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Journal of Machine Learning Research, 15, 19291958.
- Wan, L., Zeiler, M., Zhang, S., LeCun, Y., & Fergus, R. (2013). Regularization of neural networks using dropconnect. ICML, (1), 109111.
- Zaremba, W., Sutskever, I., & Vinyals, O. (2014). Recurrent Neural Network Regularization, (2013), 18.

Note: all the images and comments in this presentation were also obtained from these sources.

Appendix: Regularization in Statistics

Some canonical **regularization for least squares**:

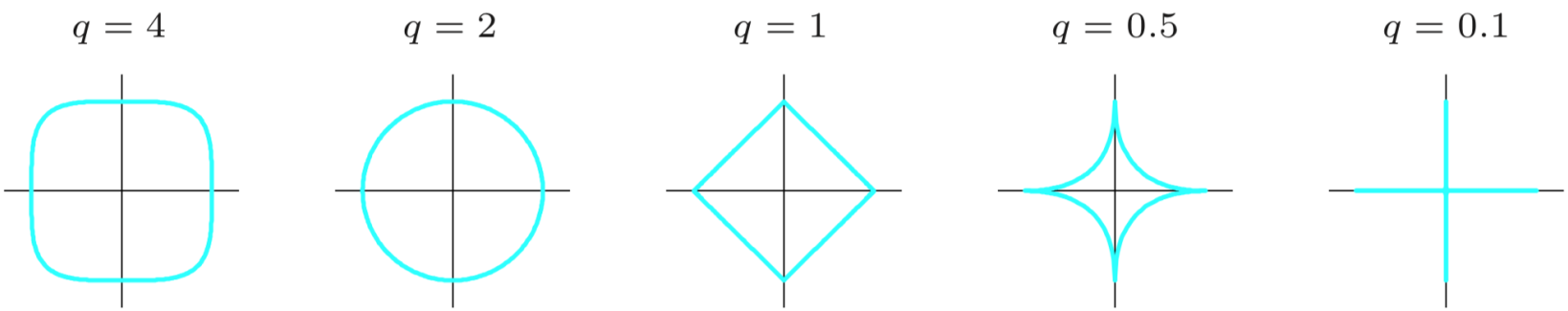
Lagrangian form:

$$\min_{\beta \in \mathbb{R}^p} \frac{1}{2}\|y - X\beta\|_2^2 + \underbrace{\lambda}_{\text{regularization parameter}} \|\beta\|_q$$

Constrained form:

$$\min_{\beta \in \mathbb{R}^p} \frac{1}{2}\|y - X\beta\|_2^2 \quad \text{subject to } \|\beta\|_q \leq \underbrace{t}_{\text{tuning parameter}}$$

- $q = 2 \rightarrow$ Ridge regression
- $q = 1 \rightarrow$ Lasso regression
- $q = 0 \rightarrow$ # of non-zero elements of β (non-convex "norm")



◀go back.

APPENDIX: Image Data Sets Details

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

◀go back

APPENDIX: Regularization in Statistics

Lagrangian form (L):

$$\min_{\beta \in \mathbb{R}^p} \frac{1}{2}\|y - X\beta\|_2^2 + \underbrace{\lambda}_{\text{regularization parameter}} \|\beta\|_q$$

Constrained form (C):

$$\min_{\beta \in \mathbb{R}^p} \frac{1}{2}\|y - X\beta\|_2^2 \quad \text{subject to } \|\beta\|_q \leq \underbrace{t}_{\text{tuning parameter}}$$

(L) is equivalent to (C): This occurs because by Lagrangian duality there is a 1-to-1 correspondence between (C) and (L) formulations. Indeed, since the only value that leads to a feasible but not strictly feasible constraint solution set in (C) is $t = 0$, then:

$$\bigcup_{\lambda \geq 0} \{\text{solutions in (L)}\} = \bigcup_{t > 0} \{\text{solutions in (C)}\}$$