Winner-Take-All Autoencoders

Alireza Makhzani, Brendan Frey
Probabilistic and Statistical Inference Group, University of Toronto

NIPS 2015

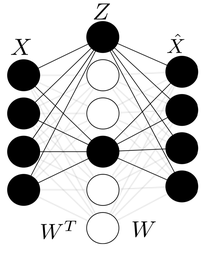
Objectives

- Winner-Take-All autoencoders use a sparsity enforcing operator and backpropagation to learn sparse hierarchical feature maps.
- WTA autoencoders achieve competitive error rate for unsupervised MNIST (0.48% error), and perform better than several complicated models on SVHN and CIFAR-10.
- WTA autoencoders are faster than EM-like algorithms (convolutional PSD, deconvnets) and contrastive divergence (convolutional RBM).

Fully-Connected WTA Autoencoders

Training:
After performing the feedforward phase, keep the $k\%$ largest activations within the mini-batch and set the rest to zero.

Sparse Encoding:
Turn off the sparsity and compute the features using ReLU activation function.



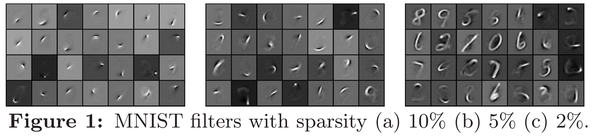


Figure 1: MNIST filters with sparsity (a) 10% (b) 5% (c) 2%.

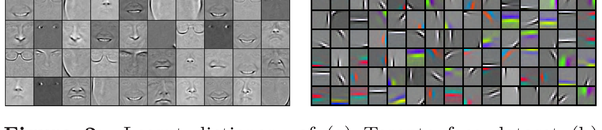


Figure 2: Learnt dictionary of (a) Toronto face dataset (b) CIFAR-10.

Winner-Take-All RBMs

In the positive phase of the contrastive divergence, we first keep the $k\%$ largest $P(h_i|\mathbf{v})$ for each h_i across the mini-batch dimension and set the rest to zero, and then sample h_i according to the sparsified $P(h_i|\mathbf{v})$.

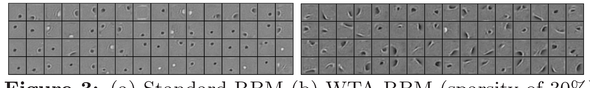


Figure 3: (a) Standard RBM (b) WTA-RBM (sparsity of 30%)

Convolutional Autoencoders

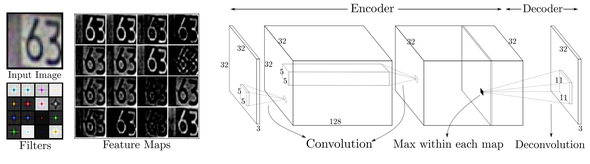


Figure 4: (a) Filters and feature maps of a denoising/dropout convolutional autoencoder. (b) Proposed architecture.

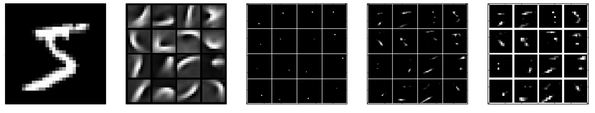
Convolutional WTA Autoencoders

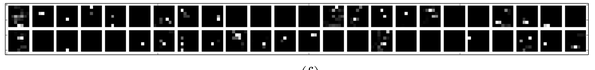
Training (unsupervised):
After performing the feedforward phase, find the largest activation within each feature map and set the rest of the hidden units in that feature map to zero. Then compute the output and the error using the sparsified maps and backpropagate the error only through the largest activations.

Sparse Encoding:
Turn off the sparsity constraint and compute the features using the ReLU activation function. Pool the maps using overlapping max-pooling to find the final representation.

Deep Winner-Take-All Autoencoders:
Train a WTA autoencoder and find the first layer feature maps as explained above. Fix the feature maps and train another WTA autoencoder to obtain the deep feature maps.

Dictionary Visualization





(f)

Figure 5: The CONV-WTA autoencoder with 16 first layer filters and 128 second layer filters trained on MNIST: (a) Input image. (b) Learnt dictionary. (c) 16 feature maps while training. (d) 16 feature maps after sparsity turned off. (e) 16 feature maps of the first layer after max-pooling. (f) final representation.

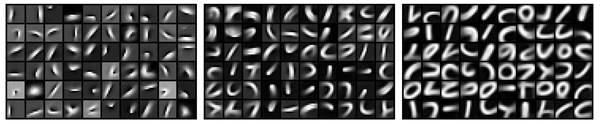


Figure 6: MNIST: (a) Only spatial sparsity (b) Spatial + lifetime sparsity 20% (c) Spatial + lifetime sparsity 5%

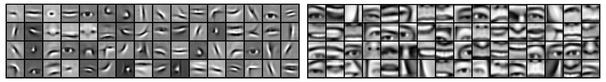


Figure 7: Toronto Face Dataset: (a) Only spatial sparsity (b) Spatial + lifetime sparsity 10%

Figure 8: ImageNet: (a) Only spatial sparsity (b) Spatial + lifetime sparsity 10%

Figure 9: Street View House Numbers.

Classification Results

We evaluate the quality of unsupervised features of WTA autoencoders by training a naive linear classifier (*i.e.*, SVM) on top them with no fine-tuning.

	Error
Deep Deconvolutional Network	0.84%
Convolutional Deep Belief Network	0.82%
Scattering Convolution Network	0.43%
Convolutional Kernel Network	0.39%
CONV-WTA Autoencoder, 16 maps	1.02%
CONV-WTA Autoencoder, 128 maps	0.64%
Stacked CONV-WTA, 128 & 2048 maps	0.48%

Table 1: MNIST: Unsupervised convolutional features + SVM

N	Convnet	CKN	Scattering Net	**CONV-WTA**
300	7.18%	4.15%	4.70%	**3.47%**
600	5.28%	-	-	**2.37%**
1K	3.21%	2.05%	2.30%	**1.92%**
2K	2.53%	1.51%	**1.30%**	1.45%
5K	1.52%	1.21%	**1.03%**	1.07%
10K	**0.85%**	0.88%	0.88 %	0.91%
60K	0.53%	**0.39%**	0.43%	0.48%

Table 2: Semi-Supervised MNIST: Unsupervised features + SVM trained on N labels.

	Accuracy
Convolutional Triangle k-means	90.6%
CONV-WTA Autoencoder	88.5%
Stacked CONV-WTA Autoencoder	93.1%
Deep VAE (non-convolutional, N=1000)	63.9%
Stacked CONV-WTA Autoencoder	76.2%

Table 3: Unsupervised and Semi-Supervised SVHN

Acknowledgments

We would like to thank Ruslan Salakhutdinov and Andrew Delong for the valuable comments.