Supervised Deep Sparse Coding Networks

Xiaoxia Sun†, Nasser M. Nasrabadi‡ and Trac D. Tran†

† *The Johns Hopkins University*
‡ *West Virginia University*

October 7, 2018

Motivations

- Motivated by convolutional neural networks.
- One-layer sparse coding model does not work well on large dataset.
- Can sparse coding be efficiently extend to deep architecture?

Introduce SparseNet

Deep Sparse Coding Networks (SparseNet)

- Clean and neat framework based on sparse coding.
- Less tweaking on network architecture.
- Competitive performance on image classification using small model.
- Better interpretation of deep networks.

SparseNet on Image Classification

Overwhelmingly outperforms previous sparse coding-based model.

- CIFAR-10: 94.19% accuracy compared to 81.40%. [1]
- CIFAR-100: 80.07% accuracy compared to 60.80%.
- STL-10: 83.11% accuracy compared to 67.90%.
- MNIST: 0.36% error rate compared to 0.54%.

1. Second best sparse coding-based approach until 2017 under fair comparison.

SparseNet on Image Classification

Exhibits competitive performance compared to deep neural networks (DNN).

- CIFAR-10: 94.19% accuracy compared to 96.42%. [2]
- CIFAR-100: 80.07% accuracy compared to 82.69%.
- STL-10: 83.11% accuracy compared to 76.29%.
- MNIST: 0.36% error rate compared to 0.21%.

2. Best reported result of deep neural network until 2017 under fair comparison.

Easily reproducible

Coded based on third-party deep learning toolbox (MatConvNet).

- Code available on GitHub: https://github.com/XiaoxiaSun/supervised-deep-sparse-coding-networks

Easy integration with deep learning schemes.

- Batch normalization
- Shortcut connection
- Dropout, Swapout and more...

Difference from previous approaches

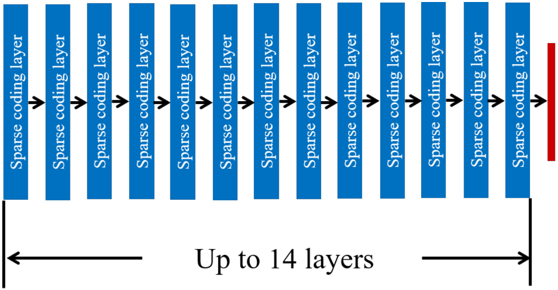


Figure: We extend sparse coding to a 14-layer deep architecture.

- Dictionary is trained using end-to-end supervised learning based on error backpropagation.
- Nonlinear dimension reduction is employed to reduce the redundancy of sparse codes.
- Regularization parameters are adaptive to the given task.
- Render state-of-the-art performance.

Inference with nonnegative sparse coding

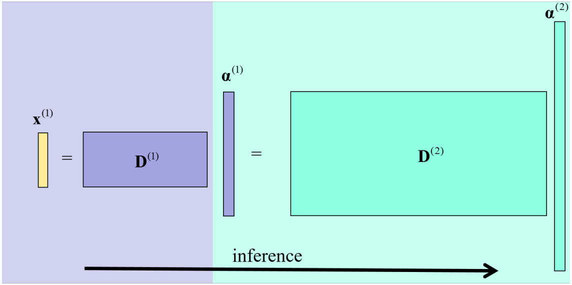
Enforce nonnegative constraint on sparse codes.

$$\boldsymbol{\alpha}^* = \arg\min_{\boldsymbol{\alpha} > \mathbf{0}} \frac{1}{2}\|\mathbf{x} - \mathbf{D}\boldsymbol{\alpha}\|_2^2 + \lambda_1 \|\boldsymbol{\alpha}\|_1 + \frac{\lambda_2}{2}\|\boldsymbol{\alpha}\|_2^2, \tag{1}$$

Advantages of nonnegative sparse coding.

- Fast convergence. Converges in 30-50 iterations in practice.
- Known clustering effect, similar to semi-nonnegative matrix factorization (semi-NMF).

Explosion of feature dimension

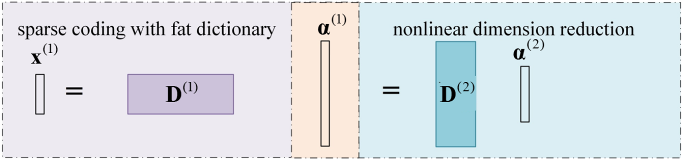


One-layer sparse coding cannot be naturally extended to multilayer architecture

- Employ wide dictionary each layer is computationally infeasible.
- Needs to reduce the dimensionality of sparse codes before passing to the deeper layer.

Bottleneck module

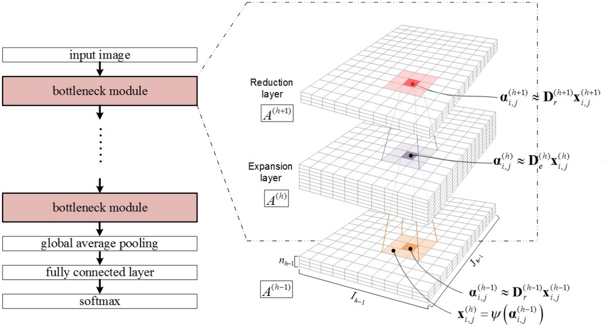
Reduce the dimensionality of sparse codes.



$$\boldsymbol{\alpha}^{(1)^*} = \arg\min_{\boldsymbol{\alpha}^{(1)}>0} \frac{1}{2}\left\|\mathbf{y} - \mathbf{D}^{(1)}\boldsymbol{\alpha}^{(1)}\right\|_2^2 + \lambda_1\left\|\boldsymbol{\alpha}^{(1)}\right\|_1 + \frac{\lambda_2}{2}\left\|\boldsymbol{\alpha}^{(1)}\right\|_2^2$$

$$\boldsymbol{\alpha}^{(2)^*} = \arg\min_{\boldsymbol{\alpha}^{(2)}>0} \frac{1}{2}\left\|\boldsymbol{\alpha}^{(1)} - \mathbf{D}^{(2)}\boldsymbol{\alpha}^{(2)}\right\|_2^2 + \lambda_1\left\|\boldsymbol{\alpha}^{(2)}\right\|_1 + \frac{\lambda_2}{2}\left\|\boldsymbol{\alpha}^{(2)}\right\|_2^2$$

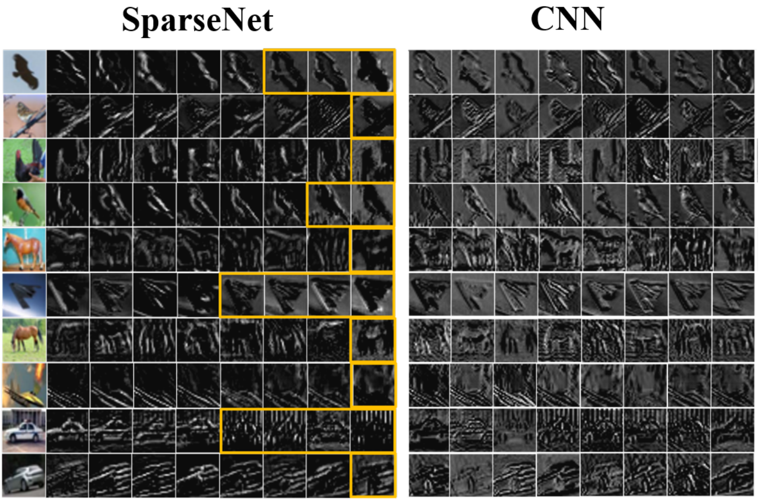
Architecture of SparseNet



(a) (b)

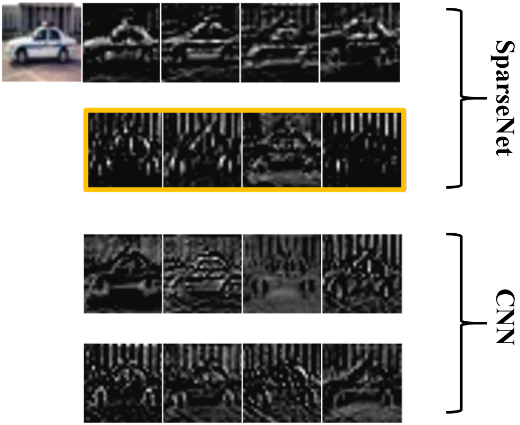
- The deep SparseNet is constructed by repeatedly stacking multiple bottleneck modules.
- Bottleneck module consists of one expansion layer and one reduction layer.
- Contains 14 sparse coding layers.

Dimension reduction leads to clustering effect



Dimension reduction leads to clustering effect

A closer look:



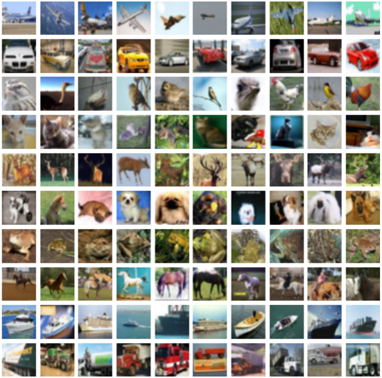
Supervised learning for SparseNet

Formulation with multilevel optimization:

$$
\begin{aligned}
&\min_{\theta} \frac{1}{S}\sum_{s=1}^{S} L(y_s, f(\mathcal{A}_s^{(h)}, \mathbf{w})) + \frac{\mu}{2}R(\theta),\\
&s.t. \quad \boldsymbol{\alpha}_s^{(H)^*} = \arg\min_{\boldsymbol{\alpha}_s^{(H)} \geq \mathbf{0}} F(\mathbf{D}^{(H)}, \lambda^{(H)}, \mathbf{x}^{(H)s}, \boldsymbol{\alpha}_s^{(H)}),\\
&\qquad \vdots\\
&\qquad s.t. \quad \boldsymbol{\alpha}_s^{(1)^*} = \arg\min_{\boldsymbol{\alpha}^{(1)} \geq \mathbf{0}} F(\mathbf{D}^{(1)}, \lambda^{(1)}, \mathbf{x}_s^{(1)}, \boldsymbol{\alpha}_s^{(1)}),\\
&\qquad\qquad s.t. \quad \lambda^{(h)} > 0,\ \mathbf{x}_s^{(h)} = \psi(\boldsymbol{\alpha}_s^{(h-1)^*}), \quad \forall h = 1, \dots, H,
\end{aligned}
\tag{2}
$$

where $\theta = \{\mathbf{D}^{(h)}, \lambda^{(h)}\}_{h=1}^{H}$.

CIFAR-10 and CIFAR-100



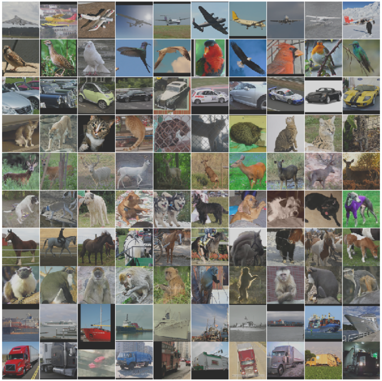
- 50, 000 training images, 10, 000 testing images.
- Evenly split into 10 (CIFAR-10) or 100 (CIFAR-100) classes.

Classification on CIFAR-10 and CIFAR-100

Table: Classification Error (%) on CIFAR-10 and CIFAR-100.

Method	# Params	# Layers	CIFAR-10	CIFAR-100
SCKN [1]	10.50M	10	10.20	-
OMP [2]	0.70M	2	18.50	-
PCANet [3]	0.28B	3	21.33	-
NOMP [4]	1.09B	4	18.60	39.92
NiN [5]	-	-	8.81	35.68
DSN [6]	1.34M	7	7.97	36.54
WRN [7]	36.5M	28	4.00	19.25
ResNet-110 [8]	0.85M	110	6.41	27.22
ResNet-1001 v2 [9]	10.2M	1001	4.92	27.21
ResNext-29 [10]	**68.10M**	**29**	**3.58**	**17.31**
SwapOut-20 [11]	1.10M	20	5.68	25.86
SwapOut-32 [11]	7.43M	32	4.76	22.72
SCN-1	0.17M	15	8.86	25.08
SCN-2	0.35M	15	7.18	22.17
SCN-4	**0.69M**	**15**	**5.81**	**19.93**

STL-10



- $5,000$ labeled training images, $8,000$ testing images.
- Evenly split into 10 classes.

Classification on STL-10

Table: Classification Accuracy (%) on STL-10.

Method	#Params	#Layers	Accuracy
SWWAE [12]	10.50M	10	74.33
Deep-TEN [13]	25.60M	50	76.29
SCN-4	**0.69M**	**15**	**83.11**

- Follow the supervised training protocol of [13].
- Training takes about 25 hours on a server with 4 Nvidia Tesla P40 GPUs.
- State-of-the-art performance with few labeled samples.

Classification on MNIST

Table: Classification Error (%) on MNIST.

Method	#Params	#Layers	Accuracy
CKN [14]		2	0.39
ScatNet [15]	-	3	0.43
PCANet [3]	-	3	0.62
S-SC [16]	-	1	0.84
TDDL [17]	-	1	0.54
SCN-4	**0.69M**	**15**	**0.36**

- Train with 25 epochs.
- Training takes about 4 hours on a server with 4 Nvidia Tesla P40 GPUs.
- Highest accuracy among sparse coding-based models.

Future works - Simplify backpropagation rule

- Dictionary update require matrix inversion:

$$\frac{\partial L}{\partial d_{jk}} = \left(\frac{\partial L}{\partial \boldsymbol{\alpha}}\right)_{\Lambda}^{\top} \cdot (\mathbf{D}_{\Lambda}^{\top}\mathbf{D}_{\Lambda} + \lambda_2 \mathbf{I}_{|\Lambda|})^{-1} (\frac{\partial \mathbf{D}_{\Lambda}^{\top}\mathbf{x}}{\partial d_{jk}} - \frac{\partial \mathbf{D}_{\Lambda}^{\top}\mathbf{D}_{\Lambda}}{\partial d_{jk}} \boldsymbol{\alpha}_{\Lambda}). \quad (3)$$

- Around 80% computation time are spent for matrix inversion.
- Find possible ways to avoid it.

Conclusion

- Dictionary learning can efficiently adapt features to the given dataset.
- Extending sparse coding to multilayer architecture is able to substantially improve the performance.
- Computational complexity is much higher than deep neural network during backpropagation.
- Large potentials of improving performance of SparseNet.

[1] J. Mairal. End-to-end kernel learning with supervised convolutional kernel networks. *CoRR*, abs/1605.06265, 2016.

[2] A. Coates and A. Y. Ng. The importance of encoding versus training with sparse coding and vector quantization. In *ICML*, 2011.

[3] T. H. Chan, K. Jia, S. Gao, J. Lu, Z. Zeng, and Y. Ma. PCANet: A simple deep learning baseline for image classification? *IEEE TIP*, 24(12):5017–5032, 2015.

[4] T. Lin and H. T. Kung. Stable and efficient representation learning with nonnegativity constraints. In *ICML*, 2014.

[5] M. Lin, Q. Chen, and S. Yan. Network in network. *CoRR*, abs/1312.4400, 2013.

[6] C. Lee, S. Xie, P. W. Gallagher, Z. Zhang, and Z. Tu. Deeply-supervised nets. In *AISTATS*, 2015.

[7] S. Zagoruyko and N. Komodakis. Wide residual networks. *CoRR*, abs/1605.07146, 2016.

[8] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[9] K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. *CoRR*, abs/1603.05027, 2016.

[10] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He. Aggregated residual transformations for deep neural networks. *arXiv preprint arXiv:1611.05431*, 2016.

[11] S. Singh, D. Hoiem, and D. A. Forsyth. Swapout: Learning an ensemble of deep architectures. In *NIPS*, 2016.

[12] J Zhao, M. Mathieu, R. Goroshin, and Y. LeCun. Stacked what-where

auto-encoders. *CoRR*, abs/1506.02351, 2015.

[13] H. Zhang, J. Xue, and K. Dana. Deep ten: Texture encoding network. *arXiv preprint arXiv:1612.02844*, 2016.

[14] J. Mairal, P. Koniusz, Z. Harchaoui, and C. Schmid. Convolutional kernel networks. In *NIPS*, 2014.

[15] J. Bruna and S. Mallat. Invariant scattering convolution networks. *IEEE TPAMI*, 35(8):1872–1886, 2013.

[16] J. Yang, K. Yu, and T. Huang. Supervised translation-invariant sparse coding. *in CVPR*, Jun. 2010.

[17] J. Mairal, F. Bach, and J. Ponce. Task-driven dictionary learning. *IEEE TPAMI*, 34(4):791–804, 2012.