A Review on Sparse Neural Network

2017.5.12

Sparse Neural Network

Sparse Stimuli $\Longleftrightarrow$ Activated Neural

hierarchical High-level abstraction

Sparse Neural Network

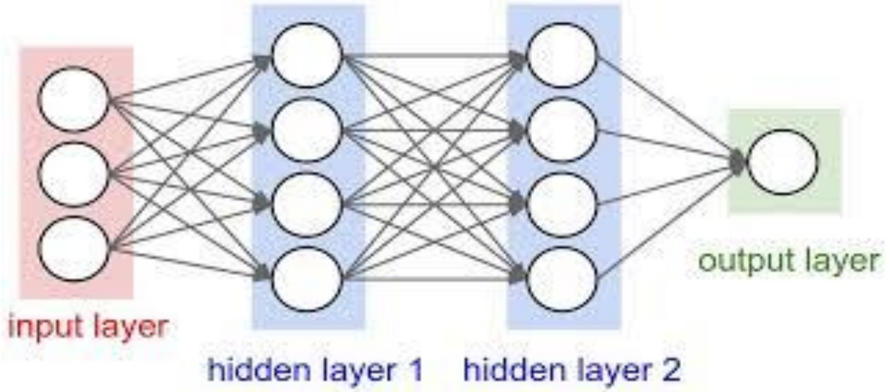
Sparse Deep Model

Deep Sparse Model

Sparse Deep Model

Overall Framework

sparse regularization ⟹



Deep Neural Network

Sparse Deep Model

Sparse Regularization

Combine

Weight Sparseness ⟺ Neural Sparseness

Norm

Zero out Wij(k)

Matrix Decomposition

Cluster & Quantization

Huffman

Norm

Activation Function

Sparse Regularization

Combine

Weight Sparseness ⟺ Neural Sparseness

Weight Sparseness	Neural Sparseness
Norm	Norm
Zero out Wij(k)	Activation Function
Matrix Decomposition	
Cluster & Quantization	
Huffman	

Sparse Deep Model

Sparse Regularization

Combine

Weight Sparseness $\Longleftrightarrow$ **Neural Sparseness**

Weight Sparseness	Neural Sparseness
Norm	Norm
Zero out Wij(k)	Activation Function
Matrix Decomposition	
Cluster & Quantization	
Huffman	

Weight Sparseness

Norm (Weight Decay)

训练准则公式

$$J(W,b;S) \quad \Longrightarrow \quad \ddot{J}(W,b;S) = J(W,b;S) + \lambda R(W)$$

Sparse Deep Model

Sparse Regularization

Combine

Weight Sparseness ⟺ Neural Sparseness

Weight Sparseness	Neural Sparseness
Norm	Norm
Zero out Wij(k)	Activation Function
Matrix Decomposition	
Cluster & Quantization	
Huffman	

Zero Out Wij(k)

Simplest: if (W_{ij} < threshold) then $W_{ij} = 0$

Choose a criteria: OBD (Optimal Brain Damage)

Weight Sparseness

Zero Out Wij(k)

OBD

(Optimal Brain Damage)

$\Delta E_i = ?$

$$\Delta E = E(w + \Delta w) - E(w)$$

$$\Downarrow \quad E(w + \Delta w) = E(w) + \frac{\partial E}{\partial w} \Delta w + \frac{1}{2} \Delta w^T H \Delta w$$

$$\Delta E = \frac{\partial E}{\partial w} \Delta w + \frac{1}{2} \Delta w^T H \Delta w$$

其中

$$H = \begin{pmatrix} \frac{\partial E^2}{\partial w_1 \partial w_1} & \cdots & \frac{\partial E^2}{\partial w_1 \partial w_K} \\ \cdots & \cdots & \cdots \\ \frac{\partial E^2}{\partial w_K \partial w_1} & \cdots & \frac{\partial E^2}{\partial w_K \partial w_K} \end{pmatrix}$$

Weight Sparseness

Zero Out Wij(k)

OBD

(Optimal Brain Damage)

$\Delta E_i = ?$

$$\Delta E = E(w + \Delta w) - E(w)$$

$$\Downarrow \quad E(w + \Delta w) = E(w) + \frac{\partial E}{\partial w}\Delta w + \frac{1}{2}\Delta w^T H \Delta w$$

$$\Delta E = \frac{\partial E}{\partial w}\Delta w + \frac{1}{2}\Delta w^T H \Delta w$$

$$\Downarrow \quad \textit{well trained:} \ \frac{\partial E}{\partial w} = 0$$

$$\Delta E \approx \frac{1}{2}\sum_{i=1}^{K} h_{i,i}\Delta w_i^2$$

$$\Downarrow$$

$$\Delta E_i = \frac{1}{2} h_{i,i}\Delta w_i^2$$

Weight Sparseness

Zero Out Wij(k)

OBD

(Optimal Brain Damage)

$h_{k,k}$ =?

Back Propagation

$$\frac{\partial^2 E}{\partial (y_i^m)^2} = f'(y_i^m)^2 \sum_l w_{l,i}^2 \frac{\partial^2 E}{\partial (y_l^{m+1})^2} + f''(y_i^m) \frac{\partial E}{\partial a_i^m}$$

由 误差函数 & 输出层的激活函数 得到

$$\frac{\partial^2 E}{\partial (y_i^M)^2}$$

代入

$$h_{k,k} = \frac{\partial^2 E}{\partial W_{i,j}^{(m)2}} = \frac{\partial^2 E}{\partial (y_i^m)^2} (a_j^m)^2$$

其中 $a_i^m = f(y_i^m) \quad y_i^m = \sum_j w_{i,j}^m a_j^{m-1}$

Weight Sparseness

Zero Out Wij(k)

OBD

(Optimal Brain Damage)

$h_{\mathrm{k,k}}$

$\Downarrow$

$$\Delta E_i = \frac{1}{2} h_{i,i} \Delta w_i^2$$

Weight Sparseness

Zero Out Wij(k)

OBD

(Optimal Brain Damage)

① train NN until convergence

② 把 ΔE_i 作为 neural node 孰优孰劣的标准

Sparse Deep Model

Sparse Regularization

Combine

Weight Sparseness $\Longleftrightarrow$ Neural Sparseness

Weight Sparseness	Neural Sparseness
Norm	Norm
Zero out Wij(k)	Activation Function
Matrix Decomposition	
Cluster & Quantization	
Huffman	

Weight Sparseness

Matrix Decomposition

矩阵分解： $A_{m\times n} = B_{m\times r} \times C_{r\times n}$

存储空间改变： m*n $\Rightarrow$ m*r + r*n

网络结构改变： $W = W^{(1)} \times W^{(2)}$

Weight Sparseness

Matrix Decomposition

Low-Rank Matrix

A_{m*n}
（r << m,n）

分解低秩矩阵A_{m*n}

$$A_{m\times n} = B_{m\times r} \times C_{r\times n}$$

节省存储空间

$$m*n > m*r + r*n$$

Weight Sparseness

Matrix Decomposition

General Sparse Matrix

A_{m*n}
($k \ll m,n$)

$$A_{m\times n} = U_{m\times n} \times \Sigma_{n\times n} \times V_{n\times n}^{T}$$

Sparse: 很多奇异值较小

取$\Sigma_{n\times n}$中前k个奇异值

$$A_{m\times n} = U_{m\times k} \times \Sigma_{k\times k} \times V_{k\times n}^{T}$$

$$= U_{m\times k} \times W_{k\times n}$$

Sparse Deep Model

Sparse Regularization

Combine

Weight Sparseness $\Longleftrightarrow$ Neural Sparseness

Norm

Zero out Wij(k)

Matrix Decomposition

Cluster & Quantization

Huffman

Norm

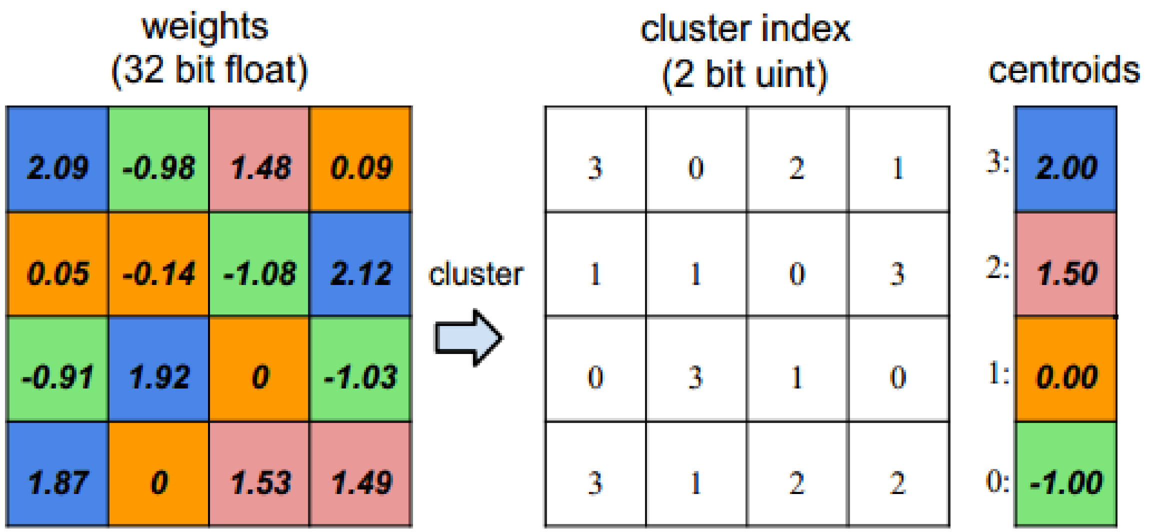
Activation Function

Weight Sparseness

Cluster (Weight Sharing)

存储空间

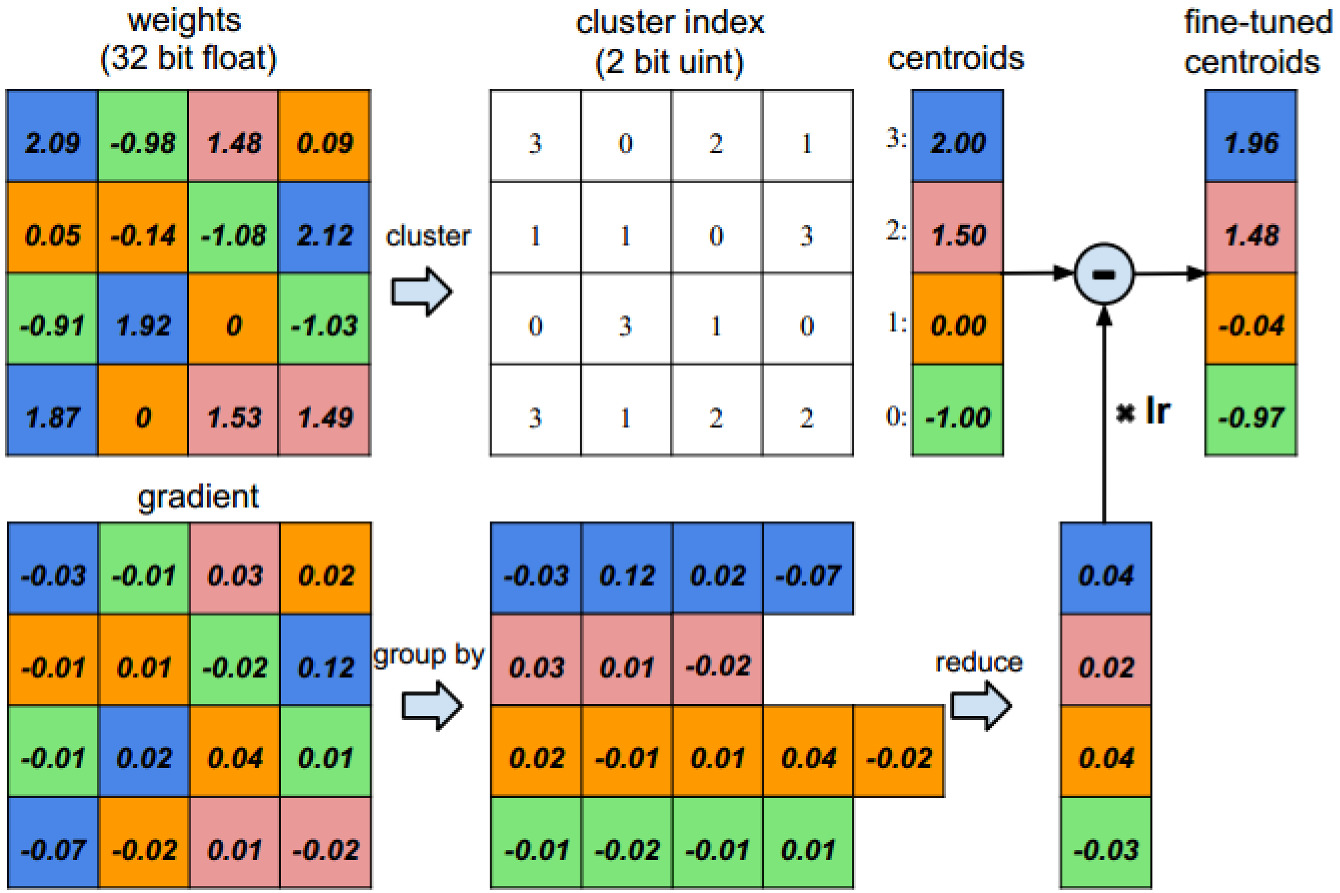
n*b ⟹ n*log(k) + k*b



Weight Sparseness

Cluster (Weight Sharing)

Cluster整体减gradient



Sparse Deep Model

Sparse Regularization

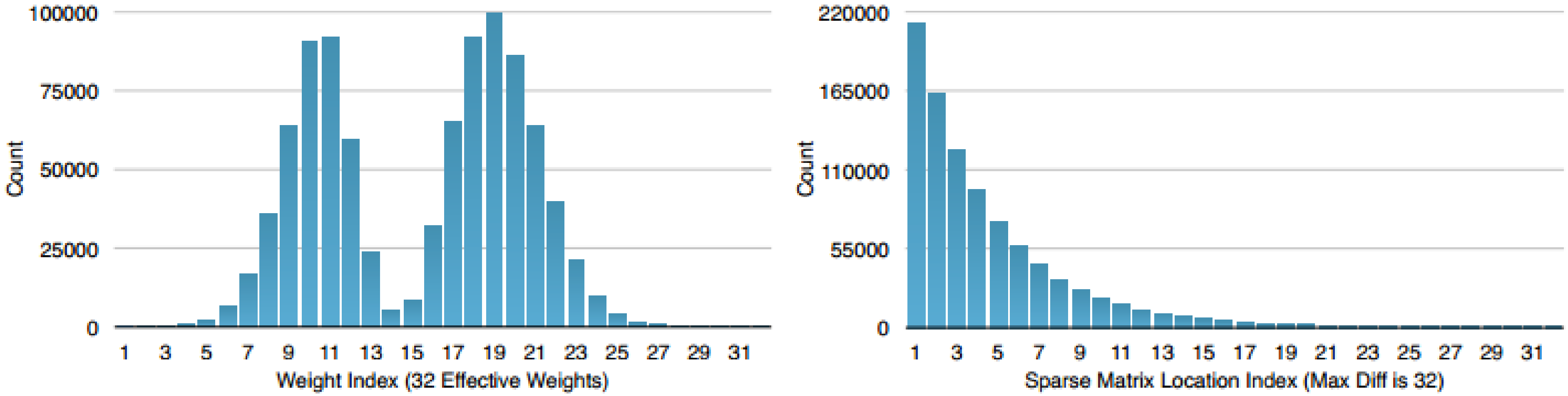
Combine

Weight Sparseness ⟺ Neural Sparseness

Weight Sparseness	Neural Sparseness
Norm	Norm
Zero out Wij(k)	Activation Function
Matrix Decomposition	
Cluster & Quantization	
Huffman	

Weight Sparseness

Huffman



Distribution for weight (Left) and index (Right). The distribution is biased.

Biased Distribution ⇨ Huffman coding

Sparse Deep Model

Sparse Regularization

Combine

Weight Sparseness ⟺ Neural Sparseness

Weight Sparseness	Neural Sparseness
Norm	Norm
Zero out Wij(k)	Activation Function
Matrix Decomposition	
Cluster & Quantization	
Huffman	

Neural Sparseness

Norm

在损失函数中加入正则项

$$\sum_{i=1}^{M} \log(1+z_i)^2$$

随 M增加 而 增加

Sparse Deep Model

Sparse Regularization

Weight Sparseness	Combine ⟺	Neural Sparseness
Norm		Norm
Zero out Wij(k)		Activation Function
Matrix Decomposition		
Cluster & Quantization		
Huffman		

Neural Sparseness

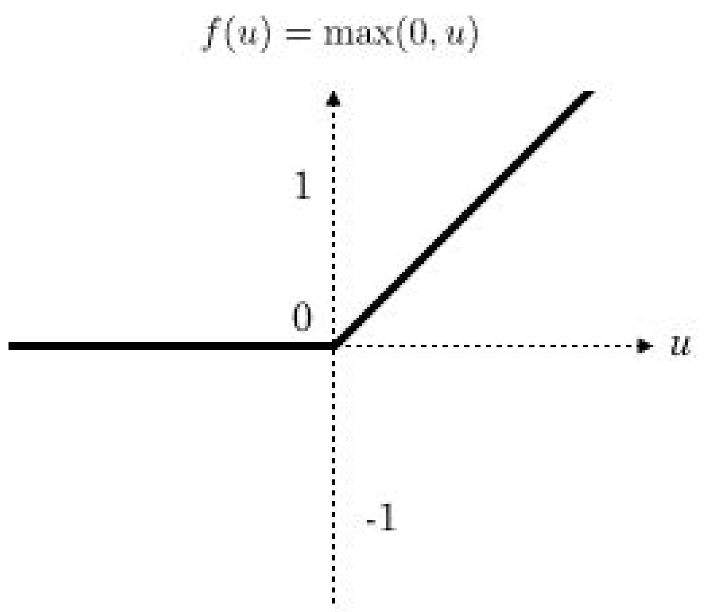
Activation Function

$$y = \sigma(x) \quad \Longrightarrow \quad y = Tr[\sigma(x)]$$

Neural Sparseness

Activation Function

$$y = \mathrm{Re}Lu(x)$$



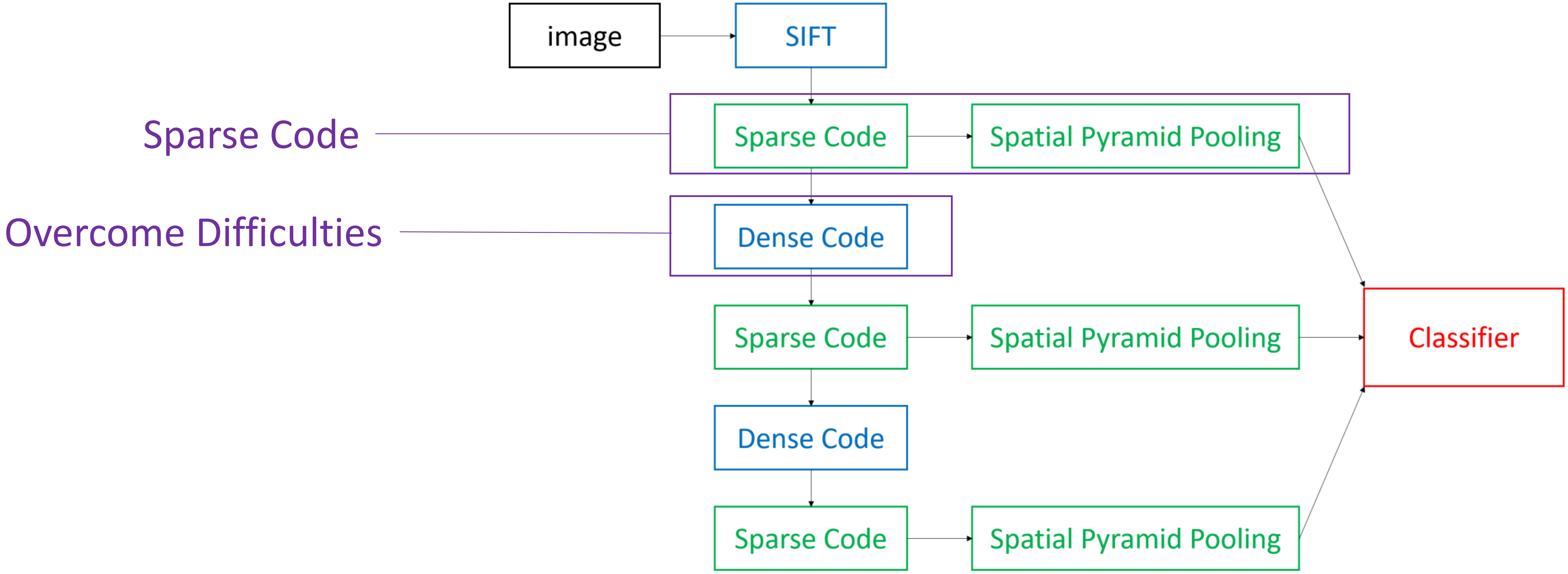
Sparse Neural Network

Sparse Deep Model

Deep Sparse Model

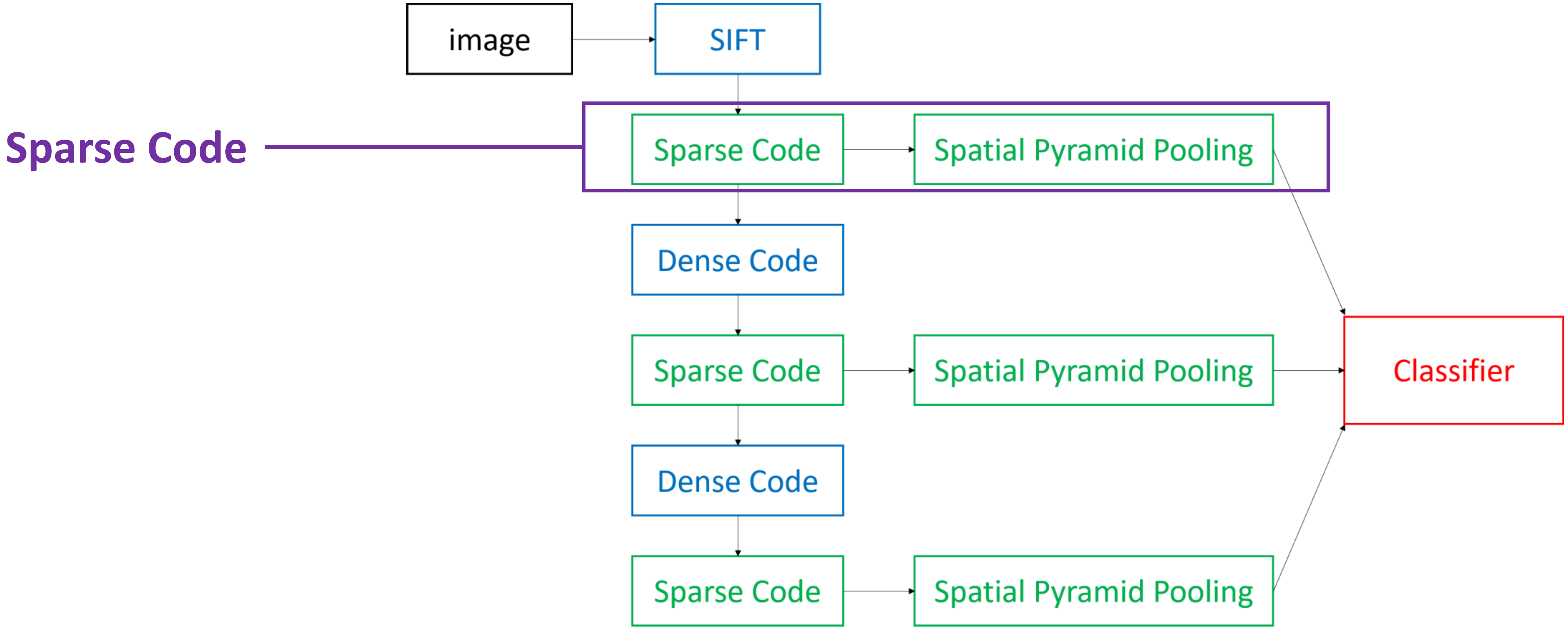
Deep Sparse Model

Overall Framework



Deep Sparse Coding

Sparse Coding



Deep Sparse Coding

Sparse Coding

Sparse Code
(Bag of Visual Words Pipeline)

Feature Extraction

$$X = [X^{(1)}, X^{(2)}, X^{(3)} \dots]$$

其中 $X^{(i)} = [X_1^{(i)}, X_2^{(i)} \dots X_{M_i}^{(i)}]$

Learning

$$[V, Y] = \underset{V,Y}{\operatorname{argmin}} ||X - VY||^2 + \alpha ||Y||$$

其中 Dict $V = [v_1, v_2, v_3 \dots v_K]$

Sparse Code $Y = [y_1, y_2, y_3 \dots y_M]$

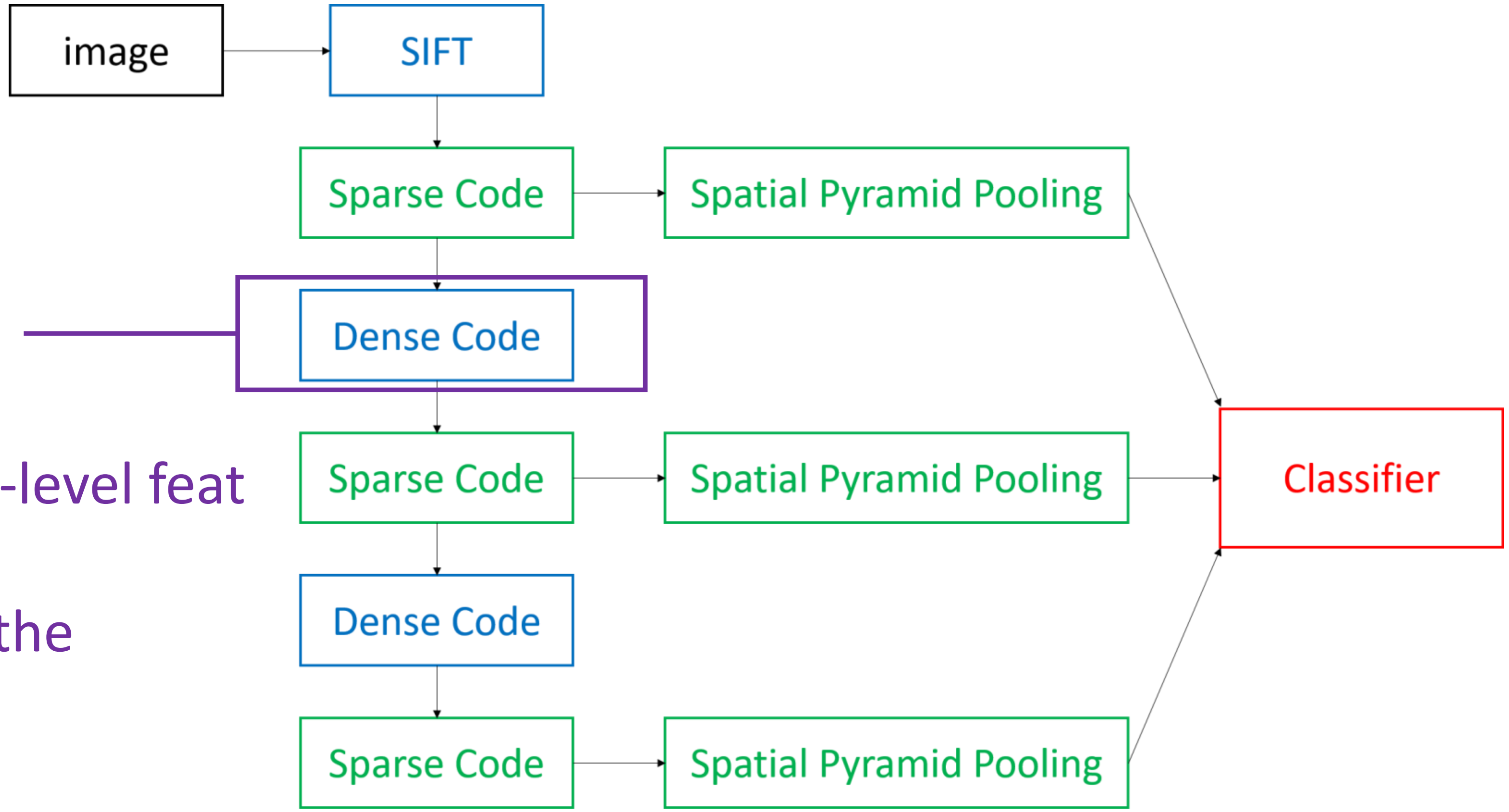
Pooling

$$y = op_{\max}(y_1, y_2, y_3 \dots y_n)$$

Deep Sparse Coding

Dense Coding

Overcome Difficulties



1. Gain spatial info and higher-level feat

2. Dimension increase makes the model not smooth

Deep Sparse Coding

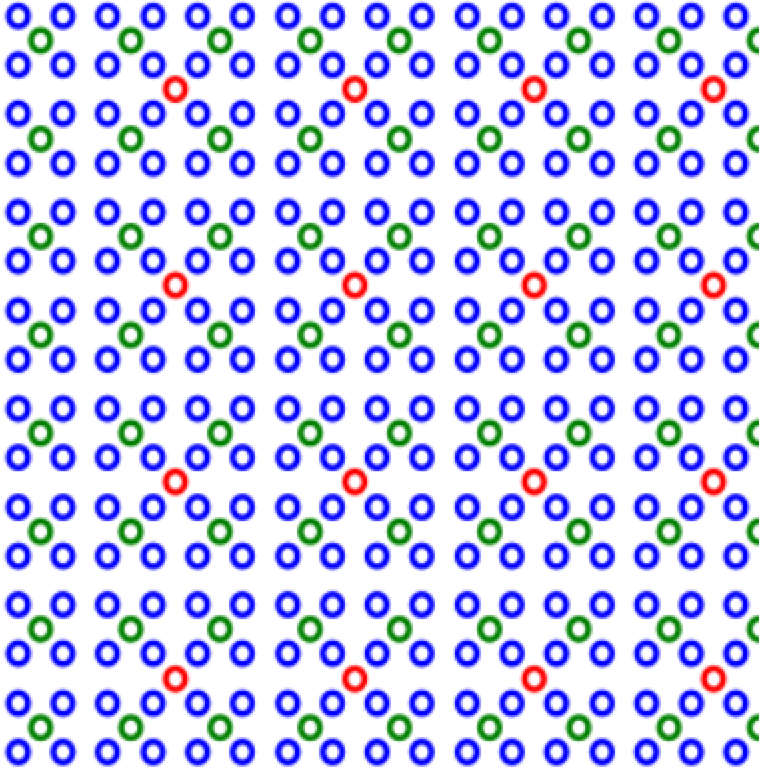
Dense Coding

Dense Code

(Overcome Difficulties)

1. Gain spatial info and higher-level feat

1. Local Spatial Pooling



$$f: (Y, G) \rightarrow (Z, G')$$

Deep Sparse Coding

Dense Coding

Dense Code

(Overcome Difficulties)

1. Gain spatial info and higher-level feat

1. Local Spatial Pooling

$$f: (Y, G) \rightarrow (Z, G')$$

其中 $Y = [y_1, y_2, y_3 \ldots]$

$$Z = [z_1, z_2, z_3 \ldots]$$

Lower-level Features → Higher-level Features

Exhibit Larger Scopes

Deep Sparse Coding

Dense Coding

Dense Code

(Overcome Difficulties)

1. Gain spatial info and higher-level feat

1. Local Spatial Pooling

实现 $f: (Y, G) \rightarrow (Z, G')$

① 确定新的点域G’

② pooling

$$\bar{y_i} = op_{\max}(y_{i1}, y_{i2}, y_{i3} \cdots y_{i16})$$

Deep Sparse Coding

Dense Coding

Dense Code

(Overcome Difficulties)

2. Dimension increase makes the model not smooth

2. Dimensionality Reduction

$$\bar{y_i} = op_{\max}(y_{i1}, y_{i2}, y_{i3} \ldots y_{i16})$$

$$A(\bar{y_i}) = W\bar{y_i}$$

$$W = \operatorname{argmin} L_{ij}(W)$$

其中

$$L_{ij}(W) = (1 - l_{ij})\frac{1}{2}||W\bar{y_i} - W\bar{y_j}||^2 +$$

$$l_{ij}\max(0, \beta - ||W\bar{y_i} - W\bar{y_j}||^2)$$

Dense Coding

Dense Code

(Overcome Difficulties)

2. Dimension increase makes the model not smooth

2. Dimensionality Reduction

$$A(\bar{y_i}) = W\bar{y_i}$$

$$W = \text{argmin}\, L_{ij}(W)$$

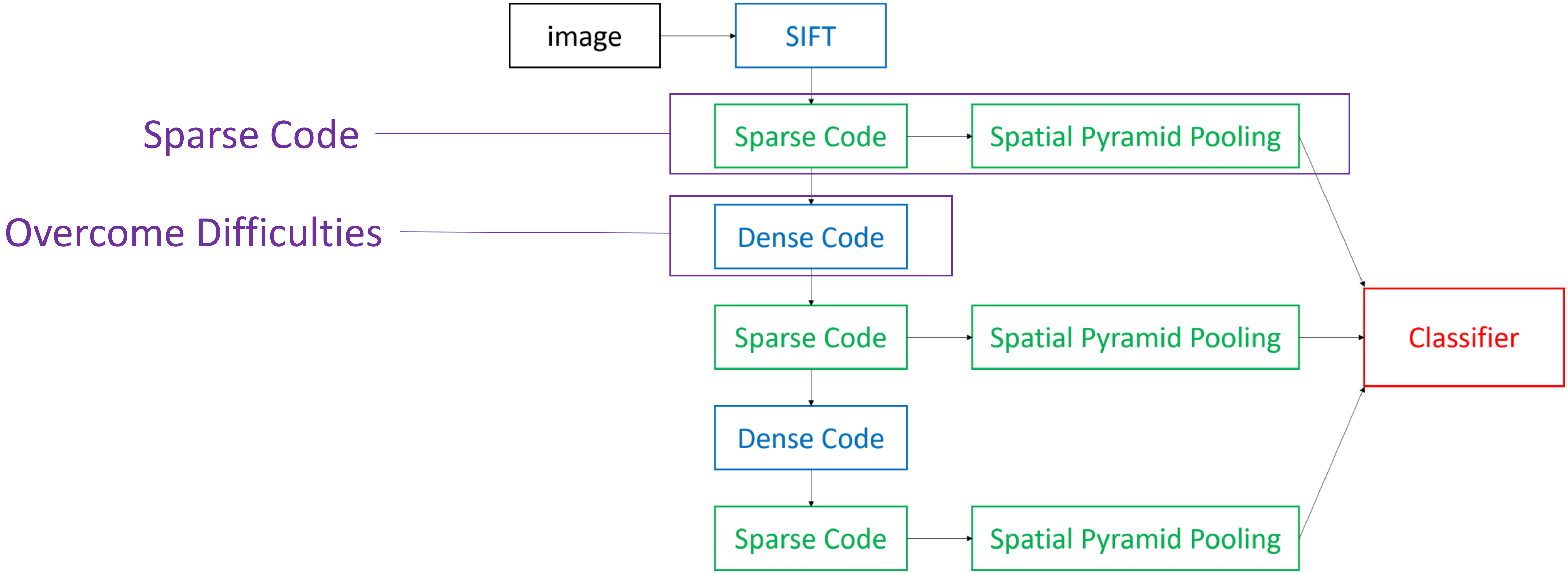
Dimensionality Reduction

$\Downarrow$

More Smooth

Deep Sparse Model

Overall Framework



Sparse Neural Network

Sparse Deep Model

Deep Sparse Model

Sparse Neural Network

谢谢！

2017.5.12