Unsupervised Feature Learning by Deep Sparse Coding

Yunlong He, Georgia Tech
Koray Kavukcuoglu, DeepMind Technologies
Yun Wang, Princeton University
Arthur Szlam, The City College of New York
Yanjun Qi, University of Virginia
Two lines of Previous Research

- Bag of Visual words
  - SIFT Descriptors

- Deep learning

Motivation: Can we combine the power of two methods?
Background: Bag of Visual words

- Image
  - Dense Code
  - Sparse Code
  - Pyramid Pooling
  - Classifier
Background: Bag of Visual words

Image → Dense Code → SIFT Descriptors
Background: Bag of Visual words

Example: 900 Patches
Background: Bag of Visual words

Image

Dense Code

SIFT filter

SIFT Descriptors for every patch
Background: Bag of Visual words

SIFT Descriptors
\[ x \in \mathbb{R}^{128}, \text{dense} \]

\[ s = \arg\min_s ||x - Bs||^2 + \lambda ||s||_1 \]

B \in \mathbb{R}^{128 \times 1024}

Sparse Codes
\[ s \in \mathbb{R}^{1024} \]
Background: Bag of Visual words

Sparse Codes $s \in \mathbb{R}^{1024}$

Max Pooling: $\max(s_1, s_2, ..., s_n)$

Concatenate Sparse Codes of 21 pooling regions

Pyramid Pooling

Sparse Code

21 pooling regions

Level 1

Level 0
Background: Bag of Visual Words

Can we learn multiple layers of sparse representations?

Image

Dense Code

Sparse Code

Pyramid Pooling

Classifier

Relatively shallow
Deep Sparse Coding

Unsupervised Deep Architecture!
Deep Sparse Coding

Image
  Dense Code
  Sparse Code
  Pyramid Pooling
  Relatively shallow
  Classifier

Image
  Dense Code
  Sparse Code
  Pyramid Pooling
  Classifier

Unsupervised Deep Architecture!
Deep Sparse Coding

Image

Dense Code

Sparse Code

Pyramid Pooling

Relatively shallow

Classifier

Unsupervised Deep Architecture!

Embedding methods!
Dimension expansion

Idea:
• Separation (for classification)
• Representation (bag of visual words)

Method:
• Sparse coding
Dimension reduction

Idea:
• Invariance
• Compositionality

Method:
• Locally spatial pooling + Low dimensional embedding
• Both unsupervised
Locally spatial pooling

Max Pooling:
\[ s = \max(s_1, s_2, \ldots, s_{16}) \]

Covers larger area
Low dimensional embedding

Sparse Code $s \in \mathbb{R}^{1024}$

$y = Ds, D \in \mathbb{R}^{128 \times 1024}$

Dense Code $y \in \mathbb{R}^{128}$

Unstable $\rightarrow$ Stable
Low dimensional embedding

Idea: embedding with the help of spatial information
Low dimensional embedding

Dimensionality Reduction by Learning an Invariant Mapping (DR. LIM)

\[ D = \arg\min_D \sum_{i,j} w_{ij} \|Ds_i - Ds_j\|^2 + \sum_{i,j} (1 - w_{ij}) \max(0, \alpha - \|Ds_i - Ds_j\|)^2 \]

- \( w_{ij} = 1 \), if \( s_i \) and \( s_j \) are overlapping neighbors
- \( w_{ij} = 0 \), if \( s_i \) and \( s_j \) are non-overlapping neighbors
Low dimensional embedding

Dimensionality Reduction by Learning an Invariant Mapping (DR. LIM)

\[ D = \arg\min_D \sum_{i,j} w_{ij} ||D_{s_i} - D_{s_j}||^2 + \sum_{i,j} (1 - w_{ij}) \max(0, \alpha - ||D_{s_i} - D_{s_j}||)^2 \]

- \( w_{ij} = 1 \), if \( s_i \) and \( s_j \) are overlapping neighbors
- \( w_{ij} = 0 \), if \( s_i \) and \( s_j \) are non-overlapping neighbors

\( y = Ds, D \in R^{128 \times 1024} \)

Sparse Code \( s \in R^{1024} \)

Dense Code \( y \in R^{128} \)
Deep Sparse Coding

Unsupervised Deep Architecture!
### Experimental Results

<table>
<thead>
<tr>
<th></th>
<th>Caltech-101</th>
<th>Caltech-256</th>
<th>15-Scene</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPM-SC</td>
<td>75.66 ± 0.59</td>
<td>43.04 ± 0.34</td>
<td>80.83 ± 0.59</td>
</tr>
<tr>
<td>DeepSC-2</td>
<td>77.41 ± 1.06</td>
<td>46.02 ± 0.57</td>
<td>82.57 ± 0.72</td>
</tr>
<tr>
<td>DeepSC-3</td>
<td>78.24 ± 0.76</td>
<td>47.00 ± 0.45</td>
<td>82.71 ± 0.68</td>
</tr>
</tbody>
</table>

- **Average per-class recognition accuracy** (shown as percentage) on three data sets using 1024 as dictionary size.
- The number of training images per class for the three data sets are 30 for Caltech-101, 60 for Caltech-256, and 100 for 15-Scene respectively.
- SPM-SC: the normal BoV pipeline with one layer of sparse coding plus spatial pyramid pooling.
Conclusion

✓ Combine sparse coding with deep learning
✓ Combine dimension expansion and dimension reduction

✓ A connecting function is learned by embedding method
✓ Combining multiple layers of sparse code achieves state-of-the-art performance on image classification tasks