Neural Codes for Image Retrieval

David Stutz

July 22, 2015

Table of Contents

1. Introduction
2. Image Retrieval
   - Bag of Visual Words
   - Vector of Locally Aggregated Descriptors
   - Sparse-Coded Features
   - Compression and Nearest-Neighbor Search
3. Convolutional Neural Networks
   - Multi-layer Perceptrons
   - Convolutional Neural Networks
   - Architectures
   - Training
4. Neural Codes for Image Retrieval
5. Experiments
6. Summary

Table of Contents

1. Introduction
2. Image Retrieval
   - Bag of Visual Words
   - Vector of Locally Aggregated Descriptors
   - Sparse-Coded Features
   - Compression and Nearest-Neighbor Search
3. Convolutional Neural Networks
   - Multi-layer Perceptrons
   - Convolutional Neural Networks
   - Architectures
   - Training
4. Neural Codes for Image Retrieval
5. Experiments
6. Summary

1. Introduction

Image retrieval:

Problem. Given a large database of images and a query image, find images showing the same object or scene.

1. Introduction

Image retrieval:

Problem. Given a large database of images and a query image, find images showing the same object or scene.

advantage: supports activities, emotions, ...

Originally:

- Text-based retrieval systems based on manual annotations;
- unpractical for large collections of images.

1. Introduction

Image retrieval:

Problem. Given a large database of images and a query image, find images showing the same object or scene.

advantage: supports activities, emotions, ...

Originally:

- Text-based retrieval systems based on manual annotations;
- unpractical for large collections of images.

Today, content-based image retrieval:

- Techniques based on the Bag of Visual Words [SZ03] model.

Table of Contents

1 Introduction

2 Image Retrieval
- Bag of Visual Words
- Vector of Locally Aggregated Descriptors
- Sparse-Coded Features
- Compression and Nearest-Neighbor Search

3 Convolutional Neural Networks
- Multi-layer Perceptrons
- Convolutional Neural Networks
- Architectures
- Training

4 Neural Codes for Image Retrieval

5 Experiments

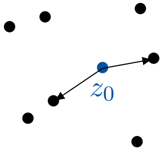
6 Summary

2. Image Retrieval

Formalization of content-based image retrieval:

Problem. Find K-nearest-neighbors of query z_0 in a (large) database $X = \{x_1, \ldots, x_N\}$ of image representations.

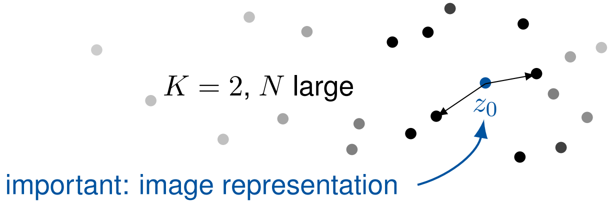
$K = 2, N = 7$



2. Image Retrieval

Formalization of content-based image retrieval:

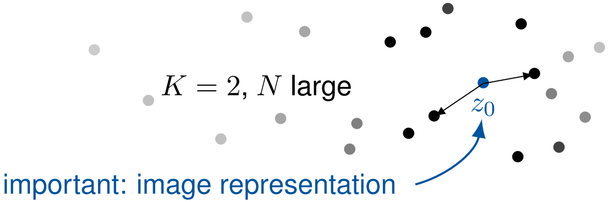
Problem. Find K-nearest-neighbors of query z_0 in a (large) database $X = \{x_1, \ldots, x_N\}$ of image representations.



2. Image Retrieval

Formalization of content-based image retrieval:

Problem. Find K-nearest-neighbors of query z_0 in a (large) database $X = \{x_1, \ldots, x_N\}$ of image representations.

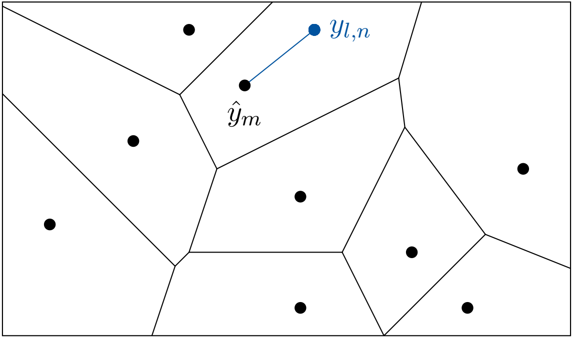


Examples for image representations from the “Computer Vision” lecture:

- Histograms;
- Bag of Visual Words [SZ03].

2.1. Bag of Visual Words

Intuition: assign local descriptors $y_{l,n}$ of image x_n to visual words $\hat{y}_1, \ldots, \hat{y}_M$ previously obtained using clustering.



2.1. Bag of Visual Words

1. Extract local descriptors Y_n for each image x_n.
2. Cluster all local descriptors $Y = \bigcup_{n=1}^{N} Y_n$ to obtain visual words

$$\hat{Y} = \{\hat{y}_1, \ldots, \hat{y}_M\}.$$

2.1. Bag of Visual Words

1. Extract local descriptors Y_n for each image x_n.
2. Cluster all local descriptors $Y = \bigcup_{n=1}^{N} Y_n$ to obtain visual words

$$\hat{Y} = \{\hat{y}_1, \ldots, \hat{y}_M\}.$$

3. Assign each $y_{l,n} \in Y_n$ to nearest visual word (embedding step):

$$f(y_{l,n}) = \left(\delta(\text{NN}_{\hat{Y}}(y_{l,n}) = \hat{y}_1), \ldots\right).$$

2.1. Bag of Visual Words

1. Extract local descriptors Y_n for each image x_n.
2. Cluster all local descriptors $Y = \bigcup_{n=1}^{N} Y_n$ to obtain visual words

$$\hat{Y} = \{\hat{y}_1, \ldots, \hat{y}_M\}.$$

3. Assign each $y_{l,n} \in Y_n$ to nearest visual word (embedding step):

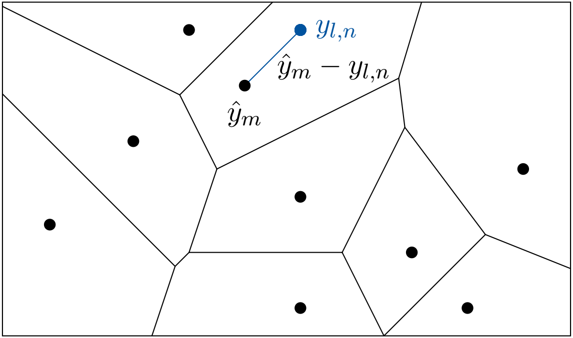
$$f(y_{l,n}) = \left(\delta(\text{NN}_{\hat{Y}}(y_{l,n}) = \hat{y}_1), \ldots\right).$$

4. Count visual word occurrences (aggregation step):

$$F(Y_n) = \sum_{l=1}^{L} f(y_{l,n}).$$

2.2. Vector of Locally Aggregated Descriptors

Intuition: consider the residuals $y_{l,n} - \hat{y}_m$ instead of counting visual words.



2.2. Vector of Locally Aggregated Descriptors

1. Extract and cluster local descriptors.

2.2. Vector of Locally Aggregated Descriptors

1. Extract and cluster local descriptors.
2. Compute residuals of local descriptors visual words (embedding step):

$$f(y_{l,n}) = \left(\delta(\text{NN}_{\hat{Y}}(y_{l,n}) = \hat{y}_1)(y_{l,n} - \hat{y}_1), \ldots\right).$$

2.2. Vector of Locally Aggregated Descriptors

1. Extract and cluster local descriptors.
2. Compute residuals of local descriptors visual words (embedding step):

$$f(y_{l,n}) = \left(\delta(\text{NN}_{\hat{Y}}(y_{l,n}) = \hat{y}_1)(y_{l,n} - \hat{y}_1), \ldots\right).$$

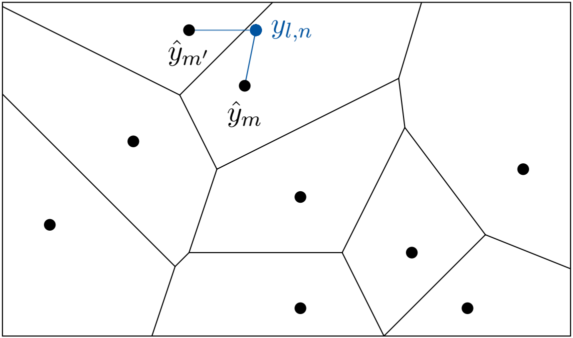
3. Aggregate residuals (aggregation step):

$$F(Y_n) = \sum_{l=1}^{L} f(y_{l,n}).$$

4. L_2-normalize $F(Y_n)$.

2.3. Sparse-Coded Features

Intuition: soft-assign local descriptors to visual words.



2.3. Sparse-Coded Features

1. Extract and cluster local descriptors.

2.3. Sparse-Coded Features

1. Extract and cluster local descriptors.
2. Compute sparse codes (embedding step):

$$f(y_{l,n}) = \underset{r_l}{\text{argmin}} \|y_{l,n} - \hat{Y} r_l\|_2^2 + \lambda \|r_l\|_1.$$

contains $\hat{y}_m$ as columns

2.3. Sparse-Coded Features

1. Extract and cluster local descriptors.
2. Compute sparse codes (embedding step):

$$f(y_{l,n}) = \underset{r_l}{\text{argmin}} \|y_{l,n} - \hat{Y} r_l\|_2^2 + \lambda \|r_l\|_1.$$

contains $\hat{y}_m$ as columns

3. Pool sparse codes (aggregation step):

$$F(Y_n) = \left(\max_{1 \leq l \leq L} \{f_1(y_{l,n})\}, \ldots\right)$$

first component of $f(y_{l,n})$

2.4. Compression, Nearest-Neighbor Search

Until now: image representation.

Additional aspects of image retrieval:

- compression of image representations;
- efficient indexing and nearest-neighbor search [JDS11];
- query expansion [CPS$^+$07] and spatial verification [PCI$^+$07].

2.4. Compression, Nearest-Neighbor Search

Until now: image representation.

Additional aspects of image retrieval:

- compression of image representations;
- efficient indexing and nearest-neighbor search [JDS11];
- query expansion [CPS+07] and spatial verification [PCI+07].

For example, compression can be accomplished using:

- Unsupervised methods, e.g. Principal Component Analysis (PCA);
- or discriminate methods, e.g. Joint Subspace and Classifier Learning [GRPV12] or Large Margin Dimensionality Reduction [SPVZ13].

discussed later ...

Table of Contents

1 Introduction

2 Image Retrieval

Bag of Visual Words

Vector of Locally Aggregated Descriptors

Sparse-Coded Features

Compression and Nearest-Neighbor Search

3 Convolutional Neural Networks

Multi-layer Perceptrons

Convolutional Neural Networks

Architectures

Training

4 Neural Codes for Image Retrieval

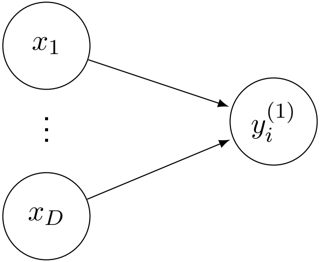
5 Experiments

6 Summary

3.1. Multi-layer Perceptrons

The prototypical neural network is the L-layer perceptron.

Given input $x \in \mathbb{R}^D$, layer $l = 1$ computes for $1 \leq i \leq m^{(l)}$:



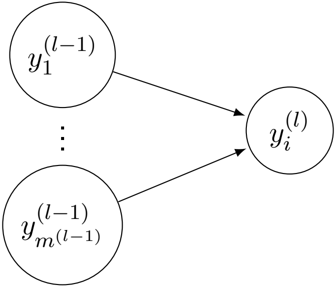
$$= f\left(\sum_{j=1}^{D} w_{i,j}^{(1)} x_j + w_{i,0}^{(1)}\right)$$

e.g. $f(z) = \frac{1}{1+\exp(-z)}$

3.1. Multi-layer Perceptrons

The prototypical neural network is the L-layer perceptron.

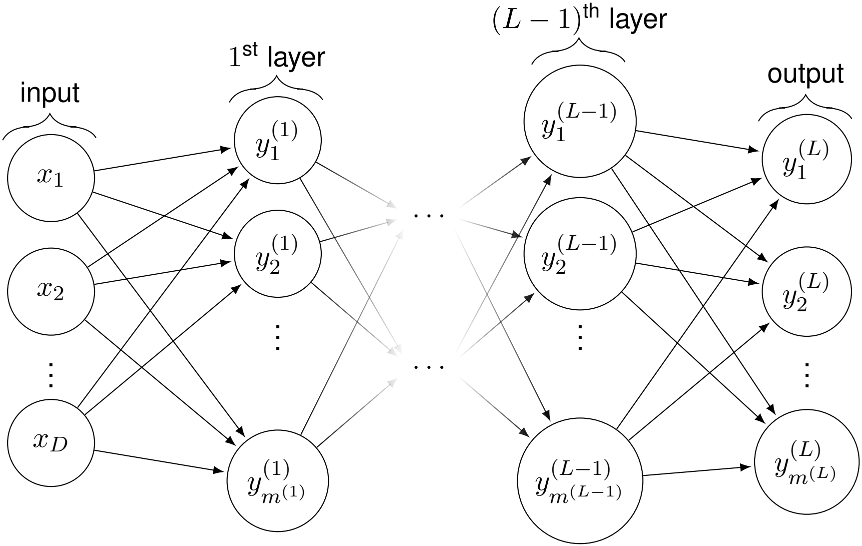
Given input $y^{(0)} := x \in \mathbb{R}^{m^{(0)}}$, layer l computes for $1 \leq i \leq m^{(l)}$:



$$y_i^{(l)} = f\left(\sum_{j=1}^{m^{(l-1)}} w_{i,j}^{(l)} y_j^{(l-1)} + w_{i,0}^{(l)}\right)$$

e.g. $f(z) = \frac{1}{1+\exp(-z)}$

3.1. Multi-layer Perceptrons



3.2. Convolutional Neural Networks

Motivation:

- Multi-layer perceptrons do not naturally accept images as input;
- however, spatial information is important.

3.2. Convolutional Neural Networks

Motivation:

- Multi-layer perceptrons do not naturally accept images as input;
- however, spatial information is important.

Solution: convolutional neural networks.

Intuition: apply learned filters on the input image to compute a set of feature maps.

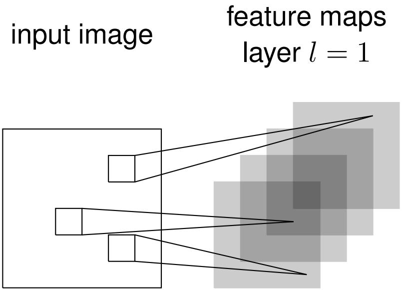
Repeat: normalize and pool feature maps before applying another set of learned filters.

Apply a multi-layer perceptron on the obtained (small) feature maps.

3.2. Convolutional Layer

General architecture:

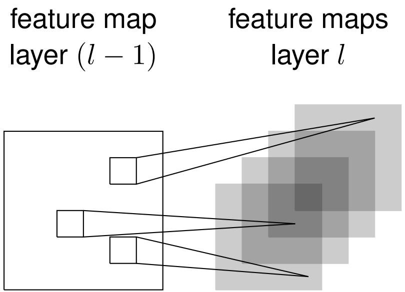
convolutional layer – contrast normalization layer – pooling layer



3.2. Convolutional Layer

General architecture:

convolutional layer – contrast normalization layer – pooling layer



3.2. Convolutional Layer

General architecture:

convolutional layer – contrast normalization layer – pooling layer

Given $m_1^{(l-1)}$ feature maps $Y_j^{(l-1)}$, layer l computes

$$Y_i^{(l)} = f\left(B_i^{(l)} + \sum_{j=1}^{m_1^{(l-1)}} W_{i,j}^{(l)} * Y_j^{(l-1)}\right), \quad 1 \leq i \leq m_1^{(l)}$$

discrete convolution

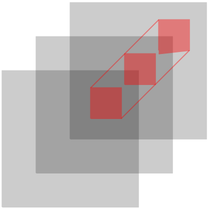
where $B_i^{(1)}$ are bias matrices and $W_{i,j}^{(1)}$ are filters.

3.2. Local Contrast Normalization Layer

General architecture:

convolutional layer – **contrast normalization layer** – pooling layer

feature maps
layer $(l-1)$



ensure that values
are comparable

3.2. Local Contrast Normalization Layer

General architecture:

convolutional layer – **contrast normalization layer** – pooling layer

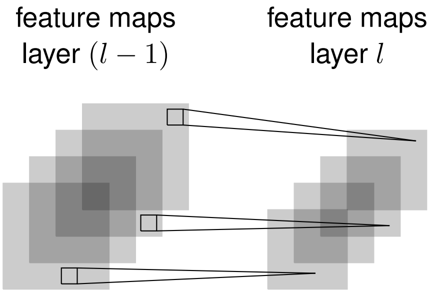
Given $m_1^{(l-1)}$ feature maps $Y_j^{(l-1)}$, brightness normalization [KSH12] computes

$$\left(Y_i^{(l)}\right)_{r,s} = \frac{\left(Y_i^{(l-1)}\right)_{r,s}}{1 + \sum_{j=1}^{m_1^{(l-1)}} \left(Y_j^{(l-1)}\right)_{r,s}^2}, \quad 1 \leq i \leq m_1^{(l)} = m_1^{(l-1)}.$$

3.2. Pooling Layer

General architecture:

convolutional layer – contrast normalization layer – **pooling layer**



3.2. Pooling Layer

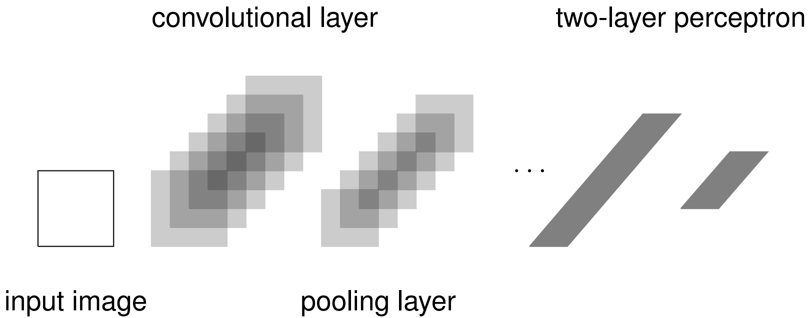
General architecture:

convolutional layer – contrast normalization layer – **pooling layer**

Given feature maps $Y_j^{(l-1)}$ of size $m_2^{(l-1)} \times m_3^{(l-1)}$, it computes feature maps $Y_i^{(l)}$ of reduced size by

- computing the average value within (non-overlapping) windows (average pooling);
- or keeping the maximum value of (non-overlapping) windows (max pooling).

3.3. Schematic Architecture



3.3. ImageNet Architecture "AlexNet"

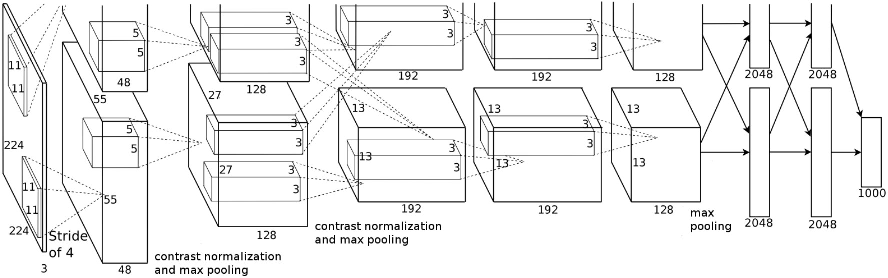


Figure : Architecture used by Krizhevsky et al. [KSH12], $L = 13$.

3.4. Training

For classification, use softmax activation function in layer L:

interpreted as posteriors

$$f(z_i^{(L)}) = \frac{\exp\left(z_i^{(L)}\right)}{\sum_{j=1}^{m^{(L)}} \exp\left(z_j^{(L)}\right)}.$$

3.4. Training

For classification, use softmax activation function in layer L:

interpreted as posteriors

$$f(z_i^{(L)}) = \frac{\exp\left(z_i^{(L)}\right)}{\sum_{j=1}^{m^{(L)}} \exp\left(z_j^{(L)}\right)}.$$

Given a training set $\{(x_n, t_n)\}$ with $t_n = i$ iff x_n belongs to class i, minimize multinomial loss

all weights of the network

$$E(W) = -\frac{1}{m^{(L)}} \sum_{n=1}^{N} \log\left(y_{t_n}^{(L)}\right)$$

using gradient descent.

Table of Contents

1. Introduction
2. Image Retrieval
    - Bag of Visual Words
    - Vector of Locally Aggregated Descriptors
    - Sparse-Coded Features
    - Compression and Nearest-Neighbor Search
3. Convolutional Neural Networks
    - Multi-layer Perceptrons
    - Convolutional Neural Networks
    - Architectures
    - Training
4. Neural Codes for Image Retrieval
5. Experiments
6. Summary

4. Neural Codes – Motivation

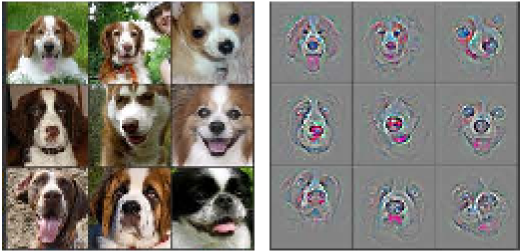


Figure : Back-projection of a single feature activation in the fourth convolutional layer [ZF14].

4. Neural Codes for Image Retrieval

Motivation: Intermediate feature activations are rich representations of image content.

For application in image retrieval, Babenko et al. [BSCL14] use

- layer $l = 10$: last convolutional layer, including subsequent max pooling;
- layer $l = 11$ and $l = 12$: first and second layer of the three-layer perceptron.

4. Neural Codes for Image Retrieval

Motivation: Intermediate feature activations are rich representations of image content.

For application in image retrieval, Babenko et al. [BSCL14] use

- layer $l = 10$: last convolutional layer, including subsequent max pooling;
- layer $l = 11$ and $l = 12$: first and second layer of the three-layer perceptron.

Two models:

- pre-trained on ImageNet[1] (~ 3.2 million images, > 1000 classes);
- and re-trained on the Landmark dataset ($213,678$ images of 672 popular landmarks).

[1] Available at `http://www.image-net.org/`.

4. Neural Codes for Image Retrieval

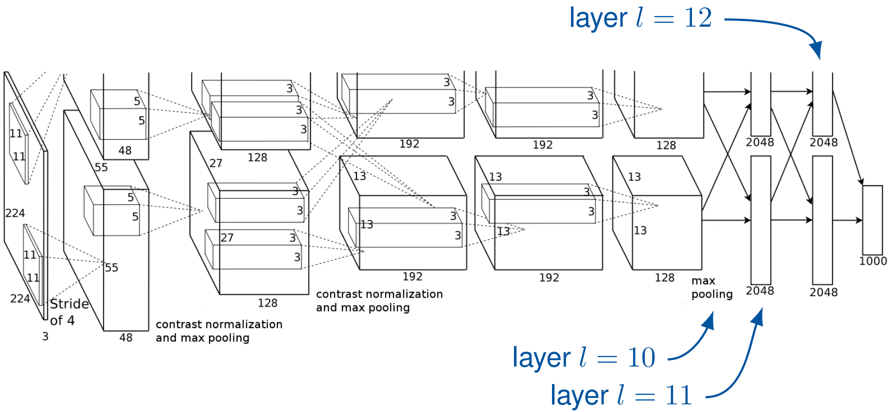


Figure : Architecture used by Krizhevsky et al. [KSH12], $L = 13$.

4. Compressed Neural Codes

Compression using PCA and Large Margin Dimensionality Reduction.

4. Compressed Neural Codes

Compression using PCA and Large Margin Dimensionality Reduction.

Large Margin Dimensionality Reduction:

1. Match images such that $t_{n,n'} = 1$ iff images x_n and $x_{n'}$ are related.
2. Compute linear dimensionality reduction $P \in \mathbb{R}^{C' \times C}$ by minimizing

$$E(P) = \sum_{n,n'}^{N} \max\{0, 1 - t_{n,n'} \left(b - (x_n - x_{n'})^T P^T P (x_n - x_{n'})\right)\}$$

large margin condition

using gradient descent.

Table of Contents

1 Introduction

2 Image Retrieval

Bag of Visual Words

Vector of Locally Aggregated Descriptors

Sparse-Coded Features

Compression and Nearest-Neighbor Search

3 Convolutional Neural Networks

Multi-layer Perceptrons

Convolutional Neural Networks

Architectures

Training

4 Neural Codes for Image Retrieval

5 Experiments

6 Summary

5. Datasets and Metric

Datasets:

- **Oxford 5k** [PCI$^+$07]: $5,062$ images of eleven different landmarks in Oxford; 5 queries with ground truth per landmark.
- INRIA Holidays [JDS08]: $1,491$ holiday images with 500 distinct queries including ground truth.

Figure : Example images from the Oxford 5k dataset showing the All Souls College of the University of Oxford.

5. Datasets and Metric

Datasets:

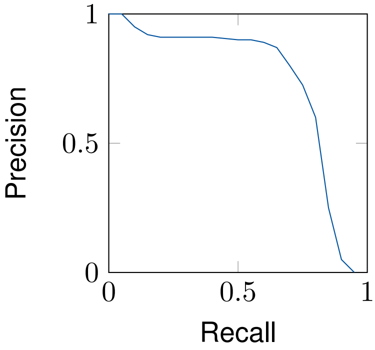
- Oxford 5k [PCI^{+}07]: $5,062$ images of eleven different landmarks in Oxford; 5 queries with ground truth per landmark.
- **INRIA Holidays** [JDS08]: $1,491$ holiday images with 500 distinct queries including ground truth.

Figure : Example images from the INRIA Holidays dataset.

5. Precision-Recall Framework

Precision-Recall curves:

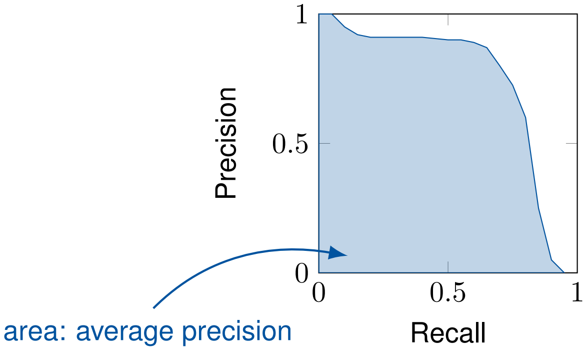
- Recall: ratio of true positives to all related images;
- Precision: ratio of true positives to number of retrieved images.



5. Precision-Recall Framework

Precision-Recall curves:

- Recall: ratio of true positives to all related images;
- Precision: ratio of true positives to number of retrieved images.



5. Experiments

	Oxford 5k	Holidays
Fisher Vectors [GRPV12]	–	0.774
Vector of Locally Aggregated Descriptors [AZ13]	0.555	0.646
Sparse-Coded Features [GKS13]	–	0.767
Triangulation Embedding [JZ14]	0.676	0.771
Pre-Trained on ImageNet		
$l = 10$	0.389	0.69
$l = 11$	0.435	0.749
$l = 12$	0.430	0.736
Re-Trained		
$l = 10$	0.387	0.674
$l = 11$	0.545	0.793
$l = 12$	0.538	0.764

Table : Mean average precision for the Oxford 5k dataset and the Holidays dataset.

5. Experiments

	Oxford 5k	Holidays
Fisher Vectors [GRPV12]	–	0.774
Vector of Locally Aggregated Descriptors [AZ13]	0.555	0.646
Sparse-Coded Features [GKS13]	–	0.767
Triangulation Embedding [JZ14]	**0.676**	0.771
Pre-Trained on ImageNet		
$l = 10$	0.389	0.69
$l = 11$	0.435	0.749
$l = 12$	0.430	0.736
Re-Trained		
$l = 10$	0.387	0.674
$l = 11$	0.545	0.793
$l = 12$	0.538	0.764

Table : Mean average precision for the Oxford 5k dataset and the Holidays dataset.

5. Experiments

	Oxford 5k	Holidays
Fisher Vectors [GRPV12]	–	0.774
Vector of Locally Aggregated Descriptors [AZ13]	0.555	0.646
Sparse-Coded Features [GKS13]	–	0.767
Triangulation Embedding [JZ14]	0.676	0.771
Pre-Trained on ImageNet		
$l = 10$	0.389	0.69
$l = 11$	0.435	0.749
$l = 12$	0.430	0.736
Re-Trained		
$l = 10$	0.387	0.674
$l = 11$	0.545	**0.793**
$l = 12$	0.538	0.764

Table : Mean average precision for the Oxford 5k dataset and the Holidays dataset.

5. Experiments with Compression

	Oxford 5k	Holidays
Fisher Vectors [GRPV12]	–	0.723
Fisher Vectors* [GRPV12]	–	0.764
Vector of Locally Aggregated Descriptors [AZ13]	0.448	0.625
Sparse-Coded Features [GKS13]	–	0.727
Triangulation Embedding [JZ14]	0.433	0.617
Pre-Trained on ImageNet		
$l = 11$ (PCA)	0.433	0.747
$l = 11$ (Large-Margin)	0.439	–
Re-Trained		
$l = 11$ (PCA)	0.557	0.789

Table : Mean average precision for the Oxford 5k dataset and the Holidays dataset using 128 dimensional image representations.

5. Experiments with Compression

	Oxford 5k	Holidays
Fisher Vectors [GRPV12]	–	0.723
Fisher Vectors* [GRPV12]	–	0.764
Vector of Locally Aggregated Descriptors [AZ13]	0.448	0.625
Sparse-Coded Features [GKS13]	–	0.727
Triangulation Embedding [JZ14]	0.433	0.617
Pre-Trained on ImageNet		
$l = 11$ (PCA)	0.433	0.747
$l = 11$ (Large-Margin)	0.439	–
Re-Trained		
$l = 11$ (PCA)	**0.557**	0.789

Table : Mean average precision for the Oxford 5k dataset and the Holidays dataset using 128 dimensional image representations.

5. Experiments with Compression

	Oxford 5k	Holidays
Fisher Vectors [GRPV12]	–	0.723
Fisher Vectors* [GRPV12]	–	0.764
Vector of Locally Aggregated Descriptors [AZ13]	0.448	0.625
Sparse-Coded Features [GKS13]	–	0.727
Triangulation Embedding [JZ14]	0.433	0.617
Pre-Trained on ImageNet		
$l = 11$ (PCA)	0.433	0.747
$l = 11$ (Large-Margin)	0.439	–
Re-Trained		
$l = 11$ (PCA)	0.557	**0.789**

Table : Mean average precision for the Oxford 5k dataset and the Holidays dataset using 128 dimensional image representations.

5. Experiments – Examples

Figure : Qualitative examples provided by Babenko et al. [BSCL14]: left-most image is the query; correctly retrieved images are marked.

5. Experiments – Conclusion

Notes on Experiments:

- no experiments using Large Margin Dimensionality Reduction on the re-trained model;
- and the results for state-of-the-art approaches are taken from the corresponding publications.

5. Experiments – Conclusion

Notes on Experiments:

- no experiments using Large Margin Dimensionality Reduction on the re-trained model;
- and the results for state-of-the-art approaches are taken from the corresponding publications.

Conclusion:

- fully learned features are interesting alternative to hand-crafted features;
- and convolutional neural networks may be explicitly trained for the image retrieval task.

Table of Contents

1 Introduction
2 Image Retrieval
  - Bag of Visual Words
  - Vector of Locally Aggregated Descriptors
  - Sparse-Coded Features
  - Compression and Nearest-Neighbor Search
3 Convolutional Neural Networks
  - Multi-layer Perceptrons
  - Convolutional Neural Networks
  - Architectures
  - Training
4 Neural Codes for Image Retrieval
5 Experiments
6 Summary

6. Summary

Summary and takeaways:

1. State-of-the-art image retrieval techniques aggregate local descriptors:
    - Bag of Visual Words [SZ03];
    - Vector of Locally Aggregated Gradients [AZ13];
    - Sparse-Coded Features [GKS13].

6. Summary

Summary and takeaways:

1. State-of-the-art image retrieval techniques aggregate local descriptors:
   - Bag of Visual Words [SZ03];
   - Vector of Locally Aggregated Gradients [AZ13];
   - Sparse-Coded Features [GKS13].
2. Convolutional neural networks are powerful, but complex models for classification.
   - Excellent performance on ImageNet;
   - but difficult to train or implement.

6. Summary

Summary and takeaways:

1. State-of-the-art image retrieval techniques aggregate local descriptors:
   - Bag of Visual Words [SZ03];
   - Vector of Locally Aggregated Gradients [AZ13];
   - Sparse-Coded Features [GKS13].
2. Convolutional neural networks are powerful, but complex models for classification.
   - Excellent performance on ImageNet;
   - but difficult to train or implement.
3. Intermediate feature activations of convolutional neural networks offer rich representations.

A.1. Bag of Visual Words – Discussion

For large Y, k-means clustering may be infeasible:

- hierarchical k-means [NS06];
- or approximate k-means [PCI$^+$07].

Burstiness, that is single large components can strongly affect performance [AZ13]:

- term frequency, inverse document frequency weighting;
- or component-wise square root and L_2 normalization.

A.2. Vector of Locally Aggregated Descriptors

Remember, embedding step:

$$f(y_{l,n}) = \left(\delta(\text{NN}_{\hat{Y}}(y_{l,n}) = \hat{y}_1)(y_{l,n} - \hat{y}_1), \ldots\right),$$

and aggregation step:

$$F(Y_n) = \sum_{l=1}^{L} f(y_{l,n}).$$

Further normalization techniques:

- power-law normalization (usually, $\alpha = 0.5$):

$$F_m(Y_n) = \text{sign}\left(F_m(Y_n)\right)\left|F_m(Y_n)\right|^\alpha;$$

- intra-normalization: L_2-normalize sum of residuals for each visual word independently.

B. Training in Practice

Training with gradient descent, in iteration $[t+1]$ compute

$$W[t+1] = W[t] - \gamma \nabla E(W[t])$$

with learning rate γ.

In practice:

- Compute $\nabla E(W[t])$ in $\mathcal{O}(|W|)$ using Error Backpropagation.
- Add a regularizer of the form

$$\hat{E}(W) = E(W) + \lambda \|W\|_1.$$

- Use dropout [HSK+12] and stochastic gradient descent.

C. Neural Codes – Motivation

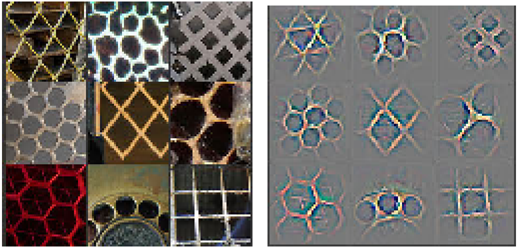


Figure : Back-projection of a single feature activation in layer $l = 3$ [ZF14][2].

[2]Note that the architecture used by Zeiler et al. [ZF14] does not exactly match the architecture presented previously.

C. Neural Codes – Motivation

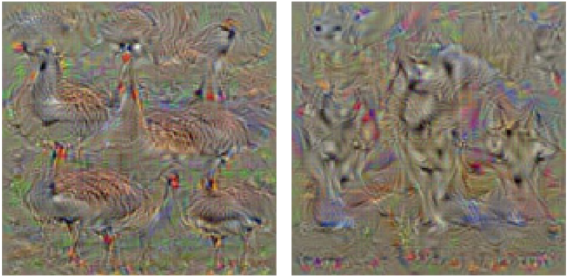


Figure : Computed image to maximize posterior for classes “goose” (left) and “husky” (right) [SVZ13].

D. Try it out ...

Unfortunately, Babenko et al. do not provide source code to reproduce their experiments.

However, you can try other state-of-the-art approaches:

- Oxford 5k dataset (including evaluation script): `http://www.robots.ox.ac.uk/~vgg/data/oxbuildings/`;
- SIFT, Vector of Locally Aggregated Descriptors and Fisher Vectors [PD07] are implemented in the VLFeat library: `http://www.vlfeat.org/overview/encodings.html`;

... or try to use convolutional neural networks, for example using

- Caffe: `http://caffe.berkeleyvision.org/`.

Relja Arandjelović and Andrew Zisserman.
All about VLAD.
In Computer Vision and Pattern Recognition, Conference on, pages 1578–1585, Portland, Oregon, June 2013.

Artem Babenko, Anton Slesarev, Alexander Chigorin, and Victor S. Lempitsky.
Neural codes for image retrieval.
In Computer Vision, European Conference on, volume 8689 of Lecture Notes in Computer Science, pages 584–599, Zurich, Switzerland, September 2014. Springer.

Ondrej Chum, James Philbin, Josef Sivic, Michael Isard, and Andrew Zisserman.
Total recall: Automatic query expansion with a generative feature model for object retrieval.
In Computer Vision, International Conference on, pages 1–8, Rio de Janeiro, Brazil, October 2007.

Tiezheng Ge, Qifa Ke, and Jian Sun.

Sparse-coded features for image retrieval.
In British Machine Vision Conference, Bristol, United Kingdom, September 2013.

Albert Gordo, José A. Rodríguez-Serrano, Florent Perronnin, and Ernest Valveny.
Leveraging category-level labels for instance-level image retrieval.
In Computer Vision and Pattern Recognition, Conference on, pages 3045–3052, Providence, Rhode Island, June 2012.

Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov.
Improving neural networks by preventing co-adaptation of feature detectors.
CoRR, abs/1207.0580, 2012.

Hervé Jégou, Matthijs Douze, and Cordelia Schmid.
Hamming embedding and weak geometric consistency for large scale image search.

In Computer Vision, European Conference on, volume 5302 of Lecture Notes in Computer Science, pages 304–317, Marseille, France, October 2008. Springer.

Hervé Jégou, Matthijs Douze, and Cordelia Schmid.
Product quantization for nearest neighbor search.
IEEE Transactions on Pattern Analysis and Machine Intelligence, 33(1):117–128, 2011.

Hervé Jégou and Andrew Zisserman.
Triangulation embedding and democratic aggregation for image search.
In Computer Vision and Pattern Recognition, Conference on, pages 3310–3317, Columbus, June 2014.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton.
ImageNet classification with deep convolutional neural networks.
In Advances in Neural Information Processing Systems, pages 1106–1114, Lake Tahoe, Nevada, December 2012.

David Nistér and Henrik Stewénius.

Scalable recognition with a vocabulary tree.
In Computer Vision and Pattern Recognition, Conference on, pages 2161–2168, New York, NY, June 2006.

- James Philbin, Ondrej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman.
  Object retrieval with large vocabularies and fast spatial matching.
  In Computer Vision and Pattern, Conference on, pages 1–8, Minneapolis, Minnesota, June 2007.

- Florent Perronnin and Christopher R. Dance.
  Fisher kernels on visual vocabularies for image categorization.
  In Computer Vision and Pattern Recognition, Conference on, pages 1–8, Minneapolis, Minnesota, June 2007.

- Karen Simonyan, Omkar M. Parkhi, Andrea Vedaldi, and Andrew Zisserman.
  Fisher vector faces in the wild.
  In British Machine Vision Conference, Bristol, United Kingdom, September 2013.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman.
Deep inside convolutional networks: Visualising image classification models and saliency maps.
CoRR, abs/1312.6034, 2013.

Josef Sivic and Andrew Zisserman.
Video google: A text retrieval approach to object matching in videos.
In Computer Vision, International Conference on, pages 1470–1477, Nice, France, October 2003.

Matthew D. Zeiler and Rob Fergus.
Visualizing and understanding convolutional networks.
In Computer Vision, European Conference on, volume 8689 of Lecture Notes in Computer Science, pages 818–833, Zurich, Switzerland, September 2014. Springer.