Learning Fine-grained Image Similarity with Deep Ranking

Jiang Wang¹ Yang Song² Thomas Leung² Chuck Rosenberg² Jingbin Wang² James Philbin² Bo Chen³ Ying Wu¹
¹Northwestern University  ²Google Inc.  ³California Institute of Technology

Gayoung Lee

(C이가영)
Contents

1. Background knowledge
2. Proposed method
3. Experimental Result
4. Conclusion
5. Quiz
Background knowledge

This paper use deep learning method to learn fine-grained images.

- What is Deep Learning?
  - Convolutional network
Common ways to learn

Processing...

2. Codewords dictionary formation

Calculate similarity

Result

Apple
Cat
Dog
Human
Bike
Duck
Ball
Common ways to learn

Processing...

Use Known Filter

Result

Apple
Cat
Dog
Human
Bike
Duck
Ball
Common ways to learn

Problem: We don’t know what processing is the best.

- Gabor filter, SIFT, HOG...
- Gabor: Really good for human face, but limited.
- SIFT: Can use generally, but it is hard to use when there are not much distinctive features
What we want to do

Processing...

Learn what to use.

Result

Apple
Cat
Dog
Human
Bike
Duck
Ball
What we want to do

Learn what to use.
Deep learning

Input (raw pixel)
Deep learning

Neuron(Module)

e.g.

$X' = WX$

$X' = \tanh((X) + b)$

Evaluation
Deep learning

Initial: \( W = (1, 3, 9) \)

\[ W = (1, 7, 9) \]

\[ \ldots \]

Update module using Feedback from evaluation layer
Convolutional Networks

Image from www.cs.nyu.edu/~yann
Convolutional Networks

Image from www.cs.nyu.edu/~yann

“Simple cells”

“Complex cells”

Multiple convolutions

pooling subsampling

Deal module as filter
Which can use every local block
Convolutional Networks: Result

Processing...

Result

Apple
Cat
Dog
Human
Bike
Duck
Duck
Ball

Learn what to use.
Convolutional Networks: Result

Image from www.cs.nyu.edu/~yann

Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]
Proposed methods

- They improve ConvNet for Fine-grained Image Similarity
  - They use triplet (ranking between same class)
  - They use low-resolution images
- They find a way of Triplet Sampling for ranking training
ConvNet

- Good Convolutional neural network model for image classification.

Label : Cat  Label : Airplane  Label : Cake
ConvNet

- However, it needs more improvement in fine-grained image searching.
- We can not give label for fine-grained image searching.

Label : White Cat?

Label : Stripe Cat?

Label : Stripe Cat?

Label : Stripe Cat?

White Cat?
Proposed methods vs. ConvNet

- Use triplets

\[ t = (P, P^+, P^-) \]

- \( P^+ \): More similar one
- \( P^- \): Not so much similar one
Proposed methods vs. ConvNet

1. Imply learning function (whole modules) to each \( P, P-, P+ \)

2. Calculate Goodness of function (Ranking Layer)

\[
l(p_i, p_i^+, p_i^-) = \max\{0, g + D(f(p_i), f(p_i^+)) - D(f(p_i), f(p_i^-))\}
\]

3. Give feedback to learning function
Proposed methods vs. ConvNet

Low-resolution Shallow part

Function f
Proposed methods vs. ConvNet

ConvNet: Strong invariance & capture the visual appearance

Two part: less invariance & capture the visual appearance
Triplet Sampling

- To train deep neural network, we need many train sets.
- So, for this paper, we need many triplet.
- Our dataset : 12 million images
  - All possible triplets: \((1.2 \times 10^7)^3 = 1.7 \times 10^{21}\)
- Suggest sampling algorithm which is good enough to train network within 24 million triplet samples.
Triplet Sampling

- Relevance score between training data.
  - Hand-made dataset
  - Collect image from google search (use query to generate relevance)
- Desired relationship in Triplet

\[ r_{i,i_+} - r_{i,i_-} \geq T_r, \forall t_i = (p_i, p_i^+, p_i^-) \]

Difference with sample of negative and positive image should be bigger than threshold
Triplet Sampling

We make buffer table to save image. If we have new image, we calculate value of image.

\[ k_j = u_j^{(1/r_j)} \]

- \( u_j \): uniformly sampled number (0,1)
- \( r_j \): Total relevance score
Triplet Sampling

If buffer is not full, we put the image. If full, we replace existed image whose value is lower than current image.

\[ k_j = u_j^{(1/r_j)} \]
Triplet Sampling

After putting sample, we can generate pair images.

Image in Different Buffer: **Out-class negative image**
Image in Same Buffer but below threshold: **In-class negative image**
Image in Same Buffer and upper threshold: **In-class positive image**

\[ r_{i,i^+} - r_{i,i^-} \geq T_r, \forall t_i = (p_i, p_i^+, p_i^-) \]
Experimental result

The learned filters of the first level convolutional layers of the multi-scale deep ranking model.
Experimental result

<table>
<thead>
<tr>
<th>Method</th>
<th>Precision</th>
<th>Score-30</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConvNet</td>
<td>82.8%</td>
<td>5772</td>
</tr>
<tr>
<td>Single-scale Ranking</td>
<td>84.6%</td>
<td>6245</td>
</tr>
<tr>
<td>OASIS on Single-scale Ranking</td>
<td>82.5%</td>
<td>6263</td>
</tr>
<tr>
<td>Single-Scale &amp; Visual Feature</td>
<td>84.1%</td>
<td>6765</td>
</tr>
<tr>
<td>DeepRanking</td>
<td>85.7%</td>
<td>7004</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>Precision</th>
<th>Score-30</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wavelet [9]</td>
<td>62.2%</td>
<td>2735</td>
</tr>
<tr>
<td>Color</td>
<td>62.3%</td>
<td>2935</td>
</tr>
<tr>
<td>SIFT-like [17]</td>
<td>65.5%</td>
<td>2863</td>
</tr>
<tr>
<td>Fisher [20]</td>
<td>67.2%</td>
<td>3064</td>
</tr>
<tr>
<td>HOG [4]</td>
<td>68.4%</td>
<td>3099</td>
</tr>
<tr>
<td>SPMKtexton1024max [16]</td>
<td>66.5%</td>
<td>3556</td>
</tr>
<tr>
<td>L1HashKPCA [14]</td>
<td>76.2%</td>
<td>6356</td>
</tr>
<tr>
<td>OASIS [3]</td>
<td>79.2%</td>
<td>6813</td>
</tr>
<tr>
<td>Golden Features</td>
<td>80.3%</td>
<td>7165</td>
</tr>
<tr>
<td>DeepRanking</td>
<td>85.7%</td>
<td>7004</td>
</tr>
</tbody>
</table>

Table 1. Similarity precision (Precision) and score-at-top-30 (Score-30) for different features.
Experimental result
Conclusion

- The proposed methods improve ConvNet for Fine-grained Image Searching
  - They use ranking approach using triplets.
  - They use low resolution shallow learning to prevent too much invariance.

- The paper suggests new triplet sampling algorithm.
  - It is more efficient. (Need much less triplet to train)
  - It overcomes memory limitation.
Thank you!

- Any Question?