Fast RCNN and DPM As a Combination for Spatial Reranking


(1) National Institute of Informatics, Japan (NII)
(2) VNU-HCMC - University of Information Technology, Vietnam (UIT-HCM)
(3) VNU-HCMC - University of Science, Vietnam (HCMUS-HCM)
(4) Universitat Politecnica de Catalunya (UPC)
(5) Nagoya University, Japan (NU)
General Instance Search Framework (1)

Three things everyone should know to improve object retrieval, R. Arandjelović, A. Zisserman, CVPR 2012

Last Year (2014) Method

1. Query images
2. Build DPM Model
3. Retrieve Top K Shots Using BOW Model
4. Compute DPM Score and Bounding Box
5. Remove Outlier Shared Words Using RANSAC
6. Compute New Score
7. Sort Scores
8. Final Ranked List

DPM with denser feature (HOG) improves the performance in case of less featured object.

BOW are used to quickly filter out unrelated frames/shots.

Our main contribution last year system.
BOW is Good for Rich Featured Objects
But ... Not for Less Textured Objects

- Small objects

Query
Background Dominated Query Object

- Burstiness

Query
DPM-based Object Localizer

**Benefit:**
- Model query object as a shape structure.
- Work well with small and texture-less object.
- Augment bounding box information.

Visualization of DPM model for query 9109

Query 9109
DPM Is Good for Less Textured Objects

Wrong shared words case

No shared word case
DPM: The Good and The Bad

- DPM is based on gray scale feature
Re-Scoring Method
→ Our Main Contribution in 2014

\[ S_{\text{new}} = (1 + N_d)^2 (1 + N_{fg} - N_d) \log_2 (1 + N_{bg}) \left( w_1 S_{BOW}^* + w_2 S_{DPM}^* \right) \]

where:

\( N_d \) : number of shared words of foreground inside bounding box (green lines)
\( N_{fg} \) : number of shared word of foreground (both blue and green lines)
\( N_{bg} \) : number of shared word of background (black lines)
\( w_1 \) : weight of BOW score
\( w_2 \) : weight of DPM score
• How to weight score of BOW and DPM?
• How to handle more highly deformable and rich colored texture objects?

⇒ This year, we tried two methods.
Query Adaptive Fusion

- Instead of using average approach \((w1=w2)\), we proposed an adaptive way of fusion.
- A neural network is used to automatically estimate weights of combining the two scores of BOW and DPM.
Query Adaptive Fusion

- Input of the network are features derived from:
  - average ratio of object area to image area
  - average number of keypoints inside query mask
  - number of shared visual words between two query examples
- Output of the network is weight of BOW and DPM derived from last years dataset
- Adaptive fusion score \( (NII\_HITACHI\_UIT\_1) \):

\[
S_{\text{new}} = \left(1 + N_d \right)^2 \left(1 + N_{fg} - N_d \right) \log_2 \left(1 + N_{bg} \right) \left( w_1 S_{BOW}^* + w_2 S_{DPM}^* \right)
\]
Combination with RCNN Based Object Detector

- DPM are good, but it:
  - does not take into account color information
  - has not enough training data and hard negatives
  - still bad at too much deformable object (with occlusion)

- RCNN based object detector are current SOA
  - uses color information to compute similarity score
  - trained on a lot of data
  - retrained on specific query object
  - still not good at finding bounding box

⇒ We combine these methods together
Final Score Based on Fast RCNN and DPM

The final score of our proposed method is given as following (NII_HITACHI_UIT_3):

$$S_{new} = \left(1 + N_{d}^{DPM}\right)^2 \left(1 + N_{fg}^{DPM} - N_{d}^{DPM}\right) \log_2 \left(1 + N_{bg}^{DPM}\right) \left(w_1 S_{BOW}^* + w_2 S_{FRCNN}^*\right)$$

where,

- Bounding box is kept as last year (returned from DPM), 3 types of shared points are computed the same
- Normalized score of Fast RCNN are used to compute base score
# Experiments

<table>
<thead>
<tr>
<th>Run ID</th>
<th>Description</th>
<th>MAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>F_A_NII_Hitachi_UIT_1</td>
<td>Query adaptive fusion</td>
<td>40.11%</td>
</tr>
<tr>
<td>F_A_NII_Hitachi_UIT_2</td>
<td>Last year config with ( w_1 = w_2 = 0.5 )</td>
<td>41.76%</td>
</tr>
<tr>
<td>F_A_NII_Hitachi_UIT_3</td>
<td>Late fusion of DPM and Fast RCNN</td>
<td>42.42%</td>
</tr>
<tr>
<td>F_A_NII_Hitachi_UIT_4</td>
<td>Last year config with ( w_1 = 0.67, w_2 = 0.33 )</td>
<td>41.53%</td>
</tr>
</tbody>
</table>
We got max perf on 8/30 queries from our 4 submitted runs.

*Object query (9145 → this jukebox wall unit)*

*Object query (9146 → this change machine)*
Consistently good for logo query (2014 & 2015)
(9137 → a Ford script logo)
Results - Bad

- Small objects ($9129 \rightarrow this\ silver\ necklace$)
Results - Bad

- Texture, illumination \((9139 \rightarrow \text{this shaggy dog (Genghis)})\)

1. [shot194_1104-0.211400] 😞

2. [shot206_381-0.208600] 😞
Results - Bad

- Color information is important (9136 → this yellow VW beetle with roofrack)
Results - Bad

- Context (9155 → this dart board)

7. [shot6_111-0.593400]

8. [shot4_977-0.558200]
Conclusions

- The first time we use a RCNN in our system and it improves pretty much compared to two baselines (41.76% → 42.42%)
  - take into account pretrained network.
  - take advantage of color information.
- We tried to improve the adaptive weighting and it works on previous datasets, but unsuccessful in this year (40.11% vs 41.76%)
- There still have unsolved problems:
  - Too small objects (with no texture).
  - Too flexible query instances: persons, animals.
Best Run NII_Hitachi_UIT_3 (42.42%)