CROSS-MEDIA RELEVANCE MINING FOR EVALUATING TEXT-BASED IMAGE SEARCH ENGINE

Zhongwen Xu[*], *Yi Yang*

ITEE, The University of Queensland
{z.xu3, yi.yang}@uq.edu.au

Ashraf Kassim, Shuicheng Yan

ECE, National University of Singapore
{ashraf, eleyans}@nus.edu.sg

ABSTRACT

Targeted at MSR-Bing Image Retrieval grand challenge, we accumulate the experience from the one in 2013, and the make further investigation into different models to solve the relevance assessment problem. Generally speaking, the assessment of relevance between text query and image list is very hard due to the semantic gap. It's not easy to find the "mapping" from text query into the visual world. Solutions from 2013 MSR-Bing grand challenge are discussed in this paper. Combining with our own observation, we give some insights into why some solutions work well, while others not. Our main solution is to combine the deep learning features with the wining solution of last year (average similarity measurement and page rank), since the deep learning features have more compact representation than the traditional BoWs features, and deep learning features are efficient (on a descent GPU) with very good performance. Our solution achieved the 1st place in MSR-Bing grand challenge 2014. Finally, we give the running time of our solution in the testing phase for the 2014 ICME testing set and development set, respectively.

***Index Terms*—** image retrieval, deep learning, semantic gap, page rank

1. INTRODUCTION

MSR-Bing Image retrieval [1] tries to tackle the following problem: how can we build a system to assess the relevance of a text query and its returned image list? This problem is important for choosing a better search engine which matches users' requirements. And it's essential for search engine itself, since it can help to re-rank its returned list to make a better relevance score. Furthermore, it helps to build up the link between short phrases/sentences and visual world. There are two sets of rules in the ICME competition. The first one is that we have to build a online system which receives query-image pairs from servers of Microsoft, and then response each of the query-image pairs with a float number to indicate the relevance in 12 seconds. And the whole testing phase lasts for a week, which means 1 week for 77,453 query-image pairs. However, to attract more teams to participate, the rule was changed into building an offline system and participants only have 1 day to run and submit their results. The testing set of MSR-Bing grand challenge this year is significantly larger than last year. Though the number of queries and the number of images with unique hash key keep almost the same (this year 999 queries with 80 thousands images, and last year 1,000 queries with 80 thousands images), the number of query-image pairs increases significantly, from 80 thousands pairs to 320 thousands pairs. With the change of the evaluation rule, the timing requirement of participants actually becomes tighter.

Here is the statistics of the training set, development set and test set (ACM MM 2013 and ICME 2014 respectively). In the development set, three labels with different levels are given, "Excellent", "Good", and "Bad", while in the training set, only the click count of one image under some queries is given, without the level labels. In Table 1, a triad indicates <query, image, click count> for training set, and <query, image, relevance level label> for development set. The inconsistency of representation of labels between training and development set increases the difficulties to investigate this problem significantly. We might need to learn a transformation from the click count to the relevance labels by some thresholds. And the click counts are given with <query, image> pairs, which means only the click counts of two images image_0 and image_1 are only comparable when they are under the same query. Otherwise, given two triads $<\text{query}_0, \text{image}_0, \text{click count}_0>$ and $<\text{query}_1, \text{image}_1, \text{click\_count}_1>$, even if $\text{click\_count}_0 > \text{click\_count}_1$, we cannot make any conclusion that query-image pair $<\text{query}_0, \text{image}_0>$ is more relevant than $<\text{query}_1, \text{image}_1>$. Some might think that normalizing the click count under the same query would help. However, the training set suffers from severely imbalanced condition, we can see from Table 1 that the training data has only 2 images under the same query on average. Some common and popular queries have a very long image list, e.g. 100, 500. Compared with the average length of image list, the long image list would decrease the importance of the click counts, however,

[*]This work was done when Zhongwen Xu was visiting National University of Singapore

the long image list should have much more information than the short one.

	Training	Dev	Test2014	Test2013
# of queries	11 M	1 k	1 k	1 k
# of images	1 M	79 k	71 k	77 k
# of triads	23 M	79 k	321 k	77 k
running time	N.A	N.A	one day	one week

Table 1. Statistics of training, development and testing set

From Table 1, we can see that the number of query-image pairs of the task in this year is 4 times as the one of last year, while the available running time drops from one week to one day.

The measurements of the MSR-Bing grand challenge are by measuring the top 25 returned images for each query. Given the scores returned by the system for the whole image list, scores are sorted and the top 25 images are used for measurements. The evaluation metric is called Discounted Cumulated Gain (DCG), the calculation of DCG@25 is shown as follows,

$$\mathrm{DCG}_{25} = 0.01757 \sum_{i=1}^{25} \frac{2^{\mathrm{rel}_i - 1}}{\log_2(i+1)}, \tag{1}$$

where $\mathrm{rel}_i = \{$ “Excellent”: 3, “Good”: 2, “Bad”: 0$\}$ is the judgement for the top ranked i-th image regarding to the query.

From Equation 1 we can see that DCG@25 appreciates predicting the “Excellent” labeled images as top-ranked, and gives penalty when predicting the “Bad” labeled images as top-ranked. With $\log_2(i+1)$ as denominator, the higher positions (e.g. 1st, 2nd place) have more influence on the final DCG@25 than the lower positions (e.g. 24th, 25th place). Noting that in development dataset, many queries have less than 25 “Excellent” images responding to the query, some queries even have less than 25 total images. So the optimal DCG@25 over the 1,000 queries in Dev set is not 1. According to our estimation, the optimal DCG@25 is about 0.68, while in the random guess, the DCG@25 is about 0.47. And the best performance for online evaluation last year is about 0.538. Though it is far higher than the result of random guess, it is still very far from the optimal bound due to the semantic gap, especially in the short sentences. This task requires perfect understanding of the semantic meaning of the natural languages, since the queries not only contain short phases, but also sentences. Some might be even questions.

2. SOLUTIONS

From the report of the champion in last year, they assume that the provided image list is from results of text-based image retrieval (TBIR), so that the image list forms some clusters with similar images. Based on this assumption, we only need to find the clusters with high similarity within the cluster, and assign a high relevance score to each image inside the clusters. To achieve this goal, [2] utilizes Page Rank method to learn the relevance score, where the transition matrix is formed by building a graph among images, with normalized similarity weights between images. To confirm this assumption, [2] gives the relevance score as the average similarity between testing image and the whole image list. This simple method achieves very good results (compared with the random guess), which indicates that using other images in the image list to represent the query, and then choosing the top most similar images to give a score is a reasonable way for relevance measure. The more similar to other images, the more relevant to the text query. With Page Rank method, it improves further than the average similarity method. And to compromise with the online evaluation rule, [2] claims that we cannot see the whole image list under online condition, so they utilize the image comes before the testing image to serve as the image list, which means for 10-th testing image in the online evaluation, 1st image to 9-th image are added into the image list in addition to the matched training images.

Similar with the observation of [2], [3] uses text clustering results to form sets image clusters. Then these image clusters serve as experts to score query-image pairs. When an expert scoring a query-image pair, it calculates the similarity between the test image and the top 5 images inside the image cluster. The key idea of Orange team’s solution is using some images to represent the query, then the relevance problem between text query and the testing image is transformed into the similarity assessment problem between testing image and some training images. As experiments from [3] have shown, CSIFT with a very large vocabulary (1 Million Bag-of-Words) comes with the best performance, while slowest. It takes 7 seconds for Orange’s system to response to a single query-test pair, since it takes a very long time to compare each CSIFT descriptor extracted in the testing image with the 1 Million vocabulary.

The above two teams both utilize the testing images which come before the current one. Compared with teams who did not apply this “trick”, the performance of top 2 teams has 3% absolute advantage than the team ranked in 3rd place. In this case, we can see the importance of utilizing other testing images to assess the relevance of query-image pairs. Our observation is that the most matched images to the specific query. Even though in the large training click logs we can find some matched queries, it is really hard to find exactly matched queries. The similar training queries we can find only share some words with the testing query. which results the found images in the training set are only partially matched to the testing query. What we want is utilizing the retrieved images in the training data to represent the text query, in this way, we build up the relation between text query and testing image through the retrieved images. This indicates that the quality of retrieved images have great impacts on the final performance. Obviously, the perfectly matched images to the text query are all in the testing image list.

Other solutions include training some concept classifiers from the training data, and in the testing phase, find the matched concepts and then apply the concept classifiers on the testing image. The prediction scores from the concept classifiers reflect the relevance between the text query and the testing images. This method transforms the relevance between query and image into the prediction from concept classifiers trained on the positive images and negative images. Some teams tried to build the relationship between text query and testing images in a way that we firstly find the most similar images in the training set, and then get the corresponding text queries of those images and then measure the similarity between the found training queries and the testing query. The weakness of this method is that the BoWs representation of short phases/sentences may be not accurate enough. The BoWs would be too sparse to calculate the similarity between two queries.

3. EXPERIMENTS UNDER ONLINE CONDITION

Since deep learning has achieved state-of-the-art performance on many different applications in computer vision and multimedia, we adapt the convolutional neural networks from ImageNet to the MSR-Bing grand challenge. Firstly, we test with some public pre-trained models. The first one is DeCaf [4], which has exactly the same structures as Alex's convnet, 5 convolutional layers and 3 fully connected layers. The following names of layers are indicated as follows, "pool5" indicates the features from the 5th convolutional layer, which contains the spatial information with 6 x 6 x 256 dimensions. "fc6" indicates the first fully connected layer (but the 6-th layer in the whole network), without the linear rectifier ReLU layer, and "fc7" indicates the second fully connected layer (but the 7-th layer in the whole network), also without the ReLU layer.

feature	DCG@25	Dimension
pool5	0.5372	6 x 6 x 256
fc6	**0.5379**	4,096
fc7	0.5314	4,096
VLAD	0.5313	98,304

Table 2. Performance of Decaf and VLAD on MSR-Bing Dev

We can see from Table 3 that features pool5 and fc6 have very similar performance, while feature fc7 has a bit lower performance than those two. Comparing pool5 and fc6, we can notice that the fc6 layer has more compact representation than pool5, in which the output of fc6 layer has only 4,096 dimensions, while pool5 has 9,216 dimensions. Worth noticing that from layer pool5 to layer, we have to store parameters of total number 9,216 x 4,096 = 37 Millions, which is much larger than the total parameter numbers in all the 5 convolutional layers. For comparison of deep learning features and the VLAD [5] feature from the traditional framework (Extracting Dense SIFT, clustering to form the coarse vocabulary, and then apply VLAD encoding to get the high dimensional representation). In this experiment, we utilize the color Dense SIFT version with different spatial pyramids. The number of coarse centers is 256. Thus the total dimension is 98,304. Compared with the features from fc6, the features from VLAD have much larger dimensions, which means much longer time to compute distances or to apply other operations. On the other hand, having a compact representation is very helpful for limiting the storage, even putting all the features of images into large memory to avoid reading from the hard disk. Image retrieval fights to fit as many images as possible into the memory with compact representation, some ways are product quantization or dimension reduction. However, as shown in Table 3, the compact representation is easily obtained from deep learning network with excellent performance.

We have tried OverFeat [6] by NYU as well, specifically, we apply the large model and get the feature by the OverFeat toolkit.

feature	DCG@25	Dimension
DeCaf	**0.5379**	4,096
Overfeat	0.5372	4,096

Table 3. Performance of Decaf and OverFeat on MSR-Bing Dev

In two different networks Decaf and OverFeat, when we extract the same layer inside the network, we get similar performance on the MSR-Bing dataset.

We have tried different ways to normalize the output from the deep learning layers. The above results are from L2 normalization of the output of neural networks. Next, we compare the L2 normalization results with SSR [7](signed squared root) normalization, which has been shown very effective for traditional features like SIFT descriptors and Bag-of Words, yet extremely simple. When we get the output of fc6, firstly apply SSR as follows on each dimension of feature x,

$$x_i = sgn(x_i)\sqrt{|x_i|}, \tag{2}$$

and then we apply L2 normalization on the feature after SSR.

feature	DCG@25	Dimension
DeCaf L2	**0.5379**	4,096
DeCaf SSR	0.5378	4,096
Overfeat L2	0.5372	4,096
Overfeat SSR	0.5368	4,096

Table 4. Performance of different normalization methods on MSR-Bing Dev

After fusing the results from Decaf and Overfeat, we got the result as 0.5437.

4. EXPERIMENTS UNDER OFFLINE CONDITION

Since the rule was changed into offline condition, which means we can see the whole image list when processing one testing image, we conduct another sets of experiments to see the performance under this setting. Noted that Decaf only has CPU version and it takes 2 seconds for Decaf to extract the feature of a single image. Overfeat takes almost the same time, might be a bit faster. Though it can match the timing requirement of this competition (it will take 44 minutes to extract features for 80 thousands images with 60 CPU cores), we are not satisfied with this timing considering the advantage in deep learning area. It is estimated that deep learning can extract 40 million images per day with a descent GPU. The toolkit we used is Caffe [8], an efficient implementation of deep convolutional neural network.

We apply the average similarity method firstly to see how much would it bring when it comes to offline condition, since the image list would be complete for each testing image. Similarly, we compare the performance of layer pool5 and fc6.

feature	DCG@25
pool5	0.5497
fc6	**0.5510**

Table 5. Performance of features from different layers of Caffe on MSR-Bing Dev

We can see from Table 4 and Table 3, the completeness of image list brings us huge benefits. The assessment between images becomes more accurate when it can see more images in the testing phase. This again confirms the assumption that the image list contains mutually similar images, and they can form some clusters under certain similarity metrics. As shown in [2], page rank works very well under this assumption, we apply page rank under the deep learning features. The relevance score r for the whole image list can be calculated as:

$$\mathbf{r} = (\alpha \mathbf{P} + (1-\alpha)\mathbf{1}^T)\mathbf{r} \tag{3}$$

We test the performance under different parameters α,

α	Dev	Test 2014
0.85	**0.5608**	0.56346
0.90	0.5605	0.56474
0.95	0.5602	N.A.
0.995	0.5603	**0.56608**

Table 6. Performance of page rank under different α on MSR-Bing Dev and Test 2014

As we can see, the performance is very stable under different α.

5. TIMING

Our implementation is very efficient to the MSR-Bing image retrieval task. We extracted features on-the-fly, did not read or write the disk for features. In the testing phase, it takes 21 min 27s to decode the base64 string into JPEG images and did some preprocessing before feeding into the DCNN. This part is done by 60 CPU cores using a very simple MapReduce framework. Then we simultaneously cooperate 4 GPUs to run together, each one processes one query at a time. It takes totally 2380.5s (39.7 mins) to process all the 320 thousands query-image pairs. For the development set, it takes almost the same time to preprocess the decoded images. And for the query-image pairs assessment, it takes about 10 mins, which is extremely fast.

6. REFERENCES

[1] Xian-Sheng Hua, Linjun Yang, Jingdong Wang, Jing Wang, Ming Ye, Kuansan Wang, Yong Rui, and Jin Li, "Clickage: Towards bridging semantic and intent gaps via mining click logs of search engines," in *Multimedia*. ACM, 2013.

[2] Chun-Che Wu, Kuan-Yu Chu, Yin-Hsi Kuo, Yan-Ying Chen, Wen-Yu Lee, and Winston H Hsu, "Search-based relevance association with auxiliary contextual cues," in *Multimedia*. ACM, 2013.

[3] Hongliang Bai Chong Huang Nan Zhao Bo Liu Yanchao Feng Yuan Dong Lezi Wang, Shusheng Cen, "France telecom orange labs (beijing) at msr-bing challenge on image retrieval 2013," in *MSR-Bing IRC Workshop*, 2013.

[4] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell, "Decaf: A deep convolutional activation feature for generic visual recognition," *arXiv:1310.1531*, 2013.

[5] Hervé Jégou, Florent Perronnin, Matthijs Douze, Jorge Sánchez, Patrick Pérez, and Cordelia Schmid, "Aggregating local image descriptors into compact codes," *TPAMI*, vol. 34, no. 9, pp. 1704–1716, 2012.

[6] Pierre Sermanet, David Eigen, Xiang Zhang, Michael Mathieu, Rob Fergus, and Yann LeCun, "Overfeat: Integrated recognition, localization and detection using convolutional networks," *arXiv:1312.6229*, 2013.

[7] Relja Arandjelovic and Andrew Zisserman, "Three things everyone should know to improve object retrieval," in *CVPR*. IEEE, 2012.

[8] Yangqing Jia, "Caffe: An open source convolutional architecture for fast feature embedding," 2013.