Assembled Image Reverse Search

Chang Liu, Yiwen Yang *

Abstract

Image reverse searching technology is now widely used to find the source of images or similar images. Meanwhile, with decreasing difficulty of adapting existing images, combining multiple source images becomes a popular way to creation. Current reverse search engines tend to return globally matching images, while original images of each assembled part are more expected. To solve this problem, this project proposes a method that treats the query image as a combination of multiple source images and returns all possible original images that composes the query image. This method is implemented and evaluated through experimental steps using real-life images, and the robustness is proved through tests.

1. Introduction

Reverse image search technology has well developed and widely applied. Some reverse image search engine like Yandex and Google can already return images which are very similar to the input. Nevertheless, those famous reverse image search engine all focus more on the global similarity. If they are fed with a image created by assembling multiple images together, they cannot find the exactly same image in their database. In this case, they will return images with highest global similarity, which is likely to be other than all the sources of the assembled images. However, when users input assembled images, they are probably more interested in where each element in the image comes from. For example, user desires to search an assembled image shown in Figure 1. This figure is in fact composed of four different images, as shown in Figure 2. When this query is sent to reverse search engine, since the image can be assembled by any users and is very likely not included in search engine database, the result will not return what user wants. Yandex provides users with the feature to crop the image manually, but cropping cannot solve image incompleteness when multiple images overlap with each other, also when the assembled image only contains a fraction of the original image. To improve user experience under this circumstance, our project aims at resolving partial matching when global matching fails. In other words, this project proposes a method of searching images with highly similar segment with the input.

Figure 1. Sample Query Image

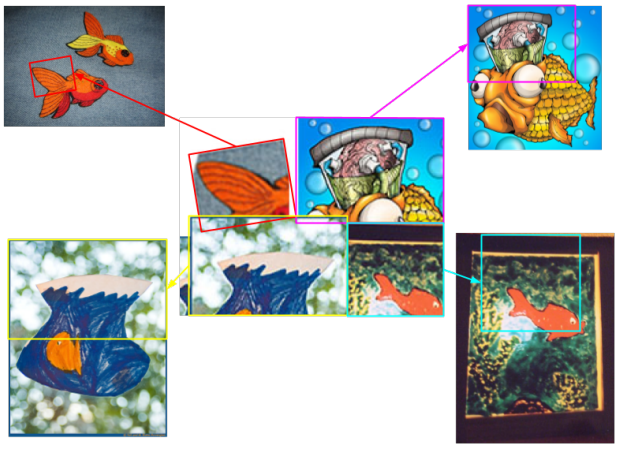


Figure 2. Components of Sample Image

2. Literature Review

A significant feature of assembled images is that they usually have a intersection with its source images in which they have very high similarity. Such local similarity is suitable for template matching. However, conventional template matching can hardly handle scale change, rotation and other transforms. In [1], a scale-invariant and rotation-invariant

*This submission is the project proposal for course ECE4880J in 2022SU. This course is held by UM-SJTU Joint Institute, Shanghai Jiao Tong University. The mentor of this course is Dr. Siheng Chen.

variation of template-matching is proposed. This method can take the most advantage of the intersection with high similarity and gives a high score to source images. Nevertheless, this method can only be applied when input is a subset of its target image. If the input contains part not belonging to the target, template matching cannot work well.

[2] proposed another local matching algorithm, the main idea of which is dividing both the input and images in database into patches and the run matching between patches. However, in this way, the patch size is fixed, causing low accuracy when resolution of input significantly varies from the target. Besides, calculating similarity for all patches in the database is also time-consuming.

In current stage, feature points matching is considered suitable in preprocessing, because feature points in source images are highly possible to be inherited by assembled images. By doing feature points matching, irrelevant images can be quickly excluded. Besides, the matching result also provides positional information which helps subsequent verification. Due to enormous number of feature points in database, a efficient feature matching algorithm is necessary. In [3], a method based on kd-tree is proposed. It narrows the range of possible matches in an efficient way, enabling searching matches among huge number of candidates.

Deep learning can also be applied for improvement in this project. As is proposed by [4], ordinary CNN can be modifies so as to inspect both global and local features. This paper use global features to do a primary filtration with global feature first, and then conduct more precise matching within candidates. The second step is helpful to our project while the primary filtration has to be substituted to fit our objective.

3. Method

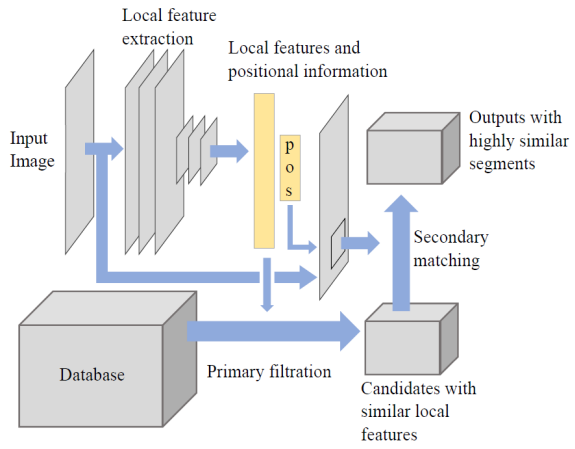


Figure 3. The Overall Structure of Local Matching Method

The core idea of this method is to make used of the spatial relations between feature points, so it's mainly based on SIFT detection, matching and calculation of perspective transformation. However, the difference of brightness and contrast between patches strongly influence feature point detection and many mismatches make find perspective transformation difficult. Therefore, besides the core steps, preprocessing like local contrast normalization and match refining are combined.

As the very first to all the images in databases and the query image, local contrast normalization is applied to get rid of the influence of change in brightness and contrast. For each pixel, its local mean and local standard deviation are calculated in its surrounding region. In this method, 50×50 square windows is used, so

$$\mu_{local} = \frac{1}{n} \sum_{i \in \text{window}} p_i$$
$$\mu'_{local} = \frac{1}{n} \sum_{i \in \text{window}} p_i^2$$
$$\sigma_{local} = \sqrt{\mu'_{local} - \mu^2_{local}}$$
$$p'_i = (p_i - \mu_{local})/\sigma_{local} \times 80 + 128$$

where p_i is the original value of the pixel at position i and p'_i is the normalized value of it. then it is restricted to the the range $[0, 255]$ and be represented by *uint8* type.

Before the first query, the database is needed to be constructed. To do this, the local contrast normalization is applied, and then SIFT feature points detection is calculated for all the images in database and the information about both the coordinate and feature description are stored together with images.

In each query, the same normalization is applied first and then SIFT feature points are detected and corresponding feature descriptions are calculated. Then these set of feature descriptions are matched with the feature description of each image in the database using flann-based matcher.

To improve efficiency, a small part of feature points in the query image are used to do first-round matching. These feature points are expected to be sparse while cover all patches. Therefore, the query image is divided into 9 parts and SIFT detection is applied separately. Only 10 strongest feature points in each part are preserved, so that at most 90 feature points evenly distributing over the image are extracted. These points are matched with all feature points in database. If one targe image has match with distance lower than 150, it's regarded as candidate and enter the next step. Otherwise, it's discarded.

For those candidates, a second-round matching involving all feature points is calculated, and the result is used to find perspective transformation. However, before any processing to the match results, there are a lot of matches between query

image and each image in database, most of which are weak and cannot really indicate local similarity. To get rid of them, the distance rate between the first two matches can be check. Weak matches usually happen when two points are not on the same object. In this case, both the first two matches are mismatch, so they have close distance. Therefore, if the distance of the first two matches are denoted by d_1, d_2, only when $d_1 < 0.7d_2$ and $d_1 < 200$ are considered as valid match and is subsequently used in the following step.

Further, because flann-based matcher only assure that one query point is matched with single destination point, multiple query points can be matched with single destination point. Such result should not happen if two images are really locally the same, because it should be a bijective mapping relation. To solve this, brute force match with cross validation can be applied to the result of flann-based matchers. As a result, one destination point will only be matched with single query points, and due to the small number of points in primary match result, efficiency is barely influenced. After all these filtration, query image and destination image are considered possible to have the same patch only if more than 10 matches can be found.

However, even with such filtration, lots of common patterns like black 90-degree corner on white background does exist in many images and lead to high quality matches. To cut out theses matches, the spacial relation between multiple feature points have to be taken into consideration. This can be achieved by find the perspective transformation between two sets of feature points. If two images have the same patch, the relative spacial relation of feature points in destination image should be maintained in the query image, so a high quality perspective transformation can be found. Moreover, this method is robust to common transformations like resizing, changing ratio, flipping and rotation. To be more specific, a transformation matrix

$$H = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix}$$

such that the back-projection error is minimized

$$E = \|X' - HX\|_2^2$$

where X' is the coordinates of destination points and X is the coordinates of query points, which are in the form of

$$X = \begin{pmatrix} x_1 & x_2 & \cdots & x_n \\ y_1 & y_2 & \cdots & y_n \\ 1 & 1 & \cdots & 1 \end{pmatrix}$$

$$X' = \begin{pmatrix} x'_1 & x'_2 & \cdots & x'_n \\ y'_1 & y'_2 & \cdots & y'_n \\ 1 & 1 & \cdots & 1 \end{pmatrix}$$

Here, n is the number of query points, which is exactly the number of qualified matches. The error can also be written in expected form.

$$E = \sum_i \left(x'_i - \frac{h_{11}x_i + h_{12}y_i + h_{13}}{h_{31}x_i + h_{32}y_i + h_{33}} \right)^2 + \left(y'_i - \frac{h_{21}x_i + h_{22}y_i + h_{23}}{h_{31}x_i + h_{32}y_i + h_{33}} \right)^2$$

Due to the existence of mismatches, which is very common, RANSAC method is used, which discards outliers completely. In this case, the least squares method takes outliers caused by mismatches into consideration, which gives less reliable result.

Eventually, the local similarity between query image and one destination image is evaluated by the back-projection error. Basically, the error is calculated in the same way in which E is calculated to find the perspective transformation. However, to get rid of the influence of mismatches, only part of the data is used.

Since $X' - HX$ is in the form of

$$X' - HX = \begin{pmatrix} \Delta x_1 & \Delta x_2 & \cdots & \Delta x_n \\ \Delta y_1 & \Delta y_2 & \cdots & \Delta y_n \\ 0 & 0 & \cdots & 0 \end{pmatrix}$$

Distance of each pair is calculated and form a array

$$D = \begin{pmatrix} \Delta x_1^2 + \Delta y_1^2 & \Delta x_2^2 + \Delta y_2^2 & \cdots & \Delta x_n^2 + \Delta y_n^2 \end{pmatrix}$$

which is then sorted and the mean of its smallest m elements is calculated as the finally error. Here, m is selected such that

$$m = \max\{11, \lfloor n \times 0.75 \rfloor\}$$

In other word, if the index of m smallest elements in D is contain the array L, the final score S indicating the local similarity between two images is

$$S = \sum_{i=0}^{m-1} D_{L[i]}$$

Lower S means higher probability that query image and destination image contain the same patch and destination images with $S < 1$ are considered to be candidate for the second stage.

4. Experiments

Our dataset comes from ImageNet-R, which contains art, cartoons, deviantart, graffiti, embroidery, graphics, origami, paintings, patterns, plastic objects, plush objects, sculptures, sketches, tattoos, toys, and video game renditions of ImageNet classes [5]. ImageNet-R is a subset of ImageNet and consists of real-life and art images rather than highly

similar ones oriented for image classification. In daily life, people often tend to assemble those interesting artworks and illustrations to create new ones, so ImageNet-R is a suitable dataset for this project.

For the experiments, we randomly select 4 images from the dataset and assemble them into one with transformations including scaling, cropping, rotating and down-sampling. The components index are recorded for experiment result verification and accuracy evaluation. The images are placed in grids to prevent one image from completely overlaying others. We choose the query number to be 4 for the simplicity of the experiment part, but this number can be scaled to any integers. In fact, our model does not need the image count as an input, and unknown number of candidates will be returned. So the experiment precision does not change with 6, 10 or more images that are assembled in one, as long as they locate reasonably in a grid-like frame.

Before starting the experiment, we first pre-compute feature points of all images in the dataset and store the feature space data in a well-organized local database, so that we do not need to calculate again during the image query (reverse image searching) process. Next, we use randomly generated query images according to the image assembling process described above and store them all in one folder. During the experiment, we simply iteratively apply assembled image reverse search for each query image.

The main assembled image reverse search function returns the original image candidates sorted with their errors. The error of a candidate image is defined as the mean distance error compared to the exact nearest neighbours using Fast Library for Approximate Nearest Neighbors (FLANN) [6] on feature points. Candidates with error value lower than 1 are filtered out to be the most possible original images of the query image. To evaluate the candidates, we define two evaluation metrics: accuracy and score. Suppose the assembled image has k children image ($k = 4$), the query returns n candidates $c_0, c_1, \ldots, c_n$ and m of them are correct. Then accuracy of a search is defined as

$$Acc = \frac{m}{k} \times 100\%$$

And the score of a search is defined as

$$S = 100 - \sum_{i=1}^{n} (\frac{100}{nk} R(arg(c[i]) - k + 1) + e_i)$$

where e is the error, and $R(x)$ is the ramp function

$$R(x) = \begin{cases} x & x > 0 \\ 0 & x \leq 0 \end{cases}$$

With higher accuracy means that more correct source images are found, and with higher scores means that confidence for all candidates to be correct is higher. Meanwhile, with closer $100Acc$ and S means to have better robustness.

5. Result and Discussion

In our experiment, 100 random assembled images are generated and tested as query images. As a result, we find mean of accuracy to be 78.50% with standard deviation 0.21, and mean of score to be 75.12 with standard deviation 21.46. The box plot for score is shown in Figure 4. The test result indicates that our method works well with relatively high accuracy. To compare with existing searching engine, we found that neither Google or Yandex can return even one correct source image of our assembled query images. In addition, the standard deviation is very low, which means that this method is stable on the test dataset. To test the robustness, we also calculated the consistency of accuracy and score, shown in Figure 5. By looking at the difference between the accuracy and score, we may conclude that the majority of test data points falls very close to x-axis, i.e. accuracy and score are consistent.

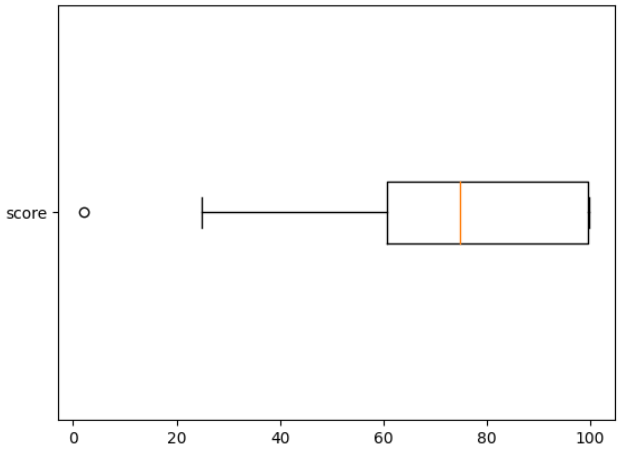


Figure 4. Score for 100 query

To better illustrate the outcome, we use the sample query image (Figure 1) for a single result. The matching result is shown in Figure 6, with error of each notated in the caption.

From this result, we may observe that first four candidates (Figure 6a, Figure 6b, Figure 6c, Figure 6d) all correspond to the correct original images, error within the threshold, so the accuracy achieves 100%. We notice that the fifth candidate (Figure 6e) also contributes a low error to the results. This is because that this candidate image is an variation of the target image, and it is reasonable that partially duplicated images should also become candidates for a reverse image query. However, it falls out of the component count range that user desires, so the overall score is 98.8264.

We also looked into source images that our method did not correctly figured out. These images share the common characteristics that they are hand-drawings with very few lines, and in our feature database they contributed very limited number of feature points. This leads to the result that

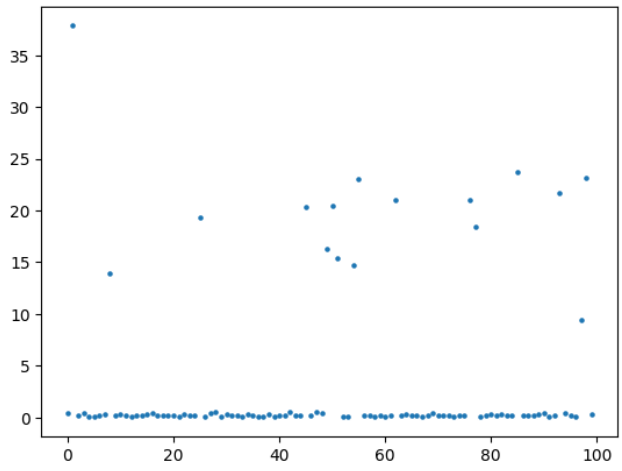
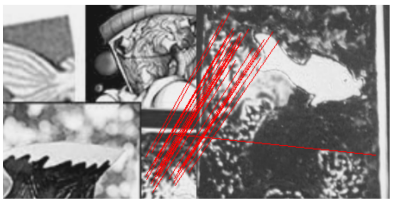
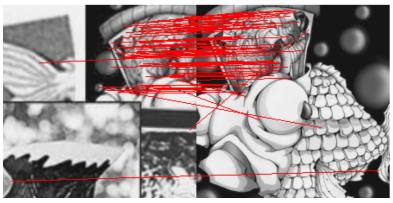


Figure 5. Consistency of Acc and S

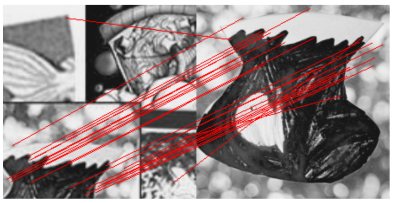
even we can find matchings between the query image and the source image, the matching count is still below the lowest threshold that we can use to distinguish whether it is the source or not. In fact, matching of this image category still remain a hard issue, and there are few related works that efficiently solve the problem. Although these images contribute to a few outliers, the overall performance of our method still overwhelms existing search engines.



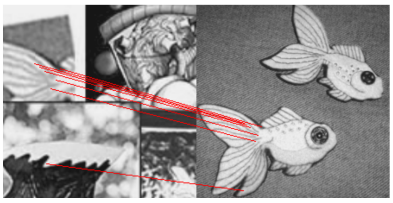
(a) Candidate 0, Error: 0.0659



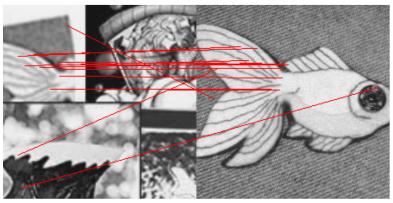
(b) Candidate 1, Error: 0.1166



(c) Candidate 2, Error: 0.1668



(d) Candidate 3, Error: 0.2478



(e) Candidate 4, Error: 0.5765

Figure 6. Result of Sample Query Image

References

[1] H. Y. Kim and S. A. de Araújo, *Grayscale Template-Matching Invariant to Rotation, Scale, Translation, Brightness and Contrast*. Berlin, Heidelberg: Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 100–113. 1

[2] A. Kimura, T. Kawanishi, and K. Kashino, "Similarity-based partial image retrieval guaranteeing same accuracy as exhaustive matching," vol. 3. Piscataway NJ: IEEE, Conference Proceedings, pp. 1895–1898 Vol.3. 2

[3] K. Wang, N. Zhu, Y. Cheng, R. Li, T. Zhou, and X. Long, "Fast feature matching based on r-nearest k-means searching," *CAAI Transactions on Intelligence Technology*, vol. 3, no. 4, pp. 198–207, 2018. 2

[4] Z. Zhou, K. Lin, Y. Cao, C.-N. Yang, and Y. Liu, "Near-duplicate image detection system using coarse-to-fine matching scheme based on global and local cnn features," *Mathematics (Basel)*, vol. 8, no. 4, p. 644, 2020. 2

[5] D. Hendrycks, S. Basart, N. Mu, S. Kadavath, F. Wang, E. Dorundo, R. Desai, T. Zhu, S. Parajuli, M. Guo, D. Song, J. Steinhardt, and J. Gilmer, "The many faces of robustness: A critical analysis of out-of-distribution generalization," *ICCV*, 2021. 3

[6] M. Muja and D. G. Lowe, "Fast approximate nearest neighbors with automatic algorithm configuration," in *VISAPP*, 2009. 4