Comparison of Various Similarity Measures for Average Image Hash in Mobile Phone Application

To cite this article: Sam Farisa Chaerul Haviana and Muhammad Taufik 2017 *IOP Conf. Ser.: Mater. Sci. Eng.* 190 012013

View the article online for updates and enhancements.
Comparison of Various Similarity Measures for Average Image Hash in Mobile Phone Application

Sam Farisa Chaerul Haviana¹ and Muhammad Taufik²

¹,² Sultan Agung Islamic University, Semarang, Central Java, Indonesia

¹ samfch@unissula.ac.id, ² mtaufik@unissula.ac.id

Abstract. One of the main issue in Content Based Image Retrieval (CIBR) is similarity measures for resulting image hashes. The main key challenge is to find the most benefits distance or similarity measures for calculating the similarity in term of speed and computing costs, specially under limited computing capabilities device like mobile phone. This study we utilize twelve most common and popular distance or similarity measures technique implemented in mobile phone application, to be compared and studied. The results show that all similarity measures implemented in this study was perform equally under mobile phone application. This gives more possibilities for method combinations to be implemented for image retrieval.

1. Introduction

One of the key issue in Content Based Image Retrieval (CIBR) is similarity measurement. After extracting image feature like image hash, next step is to calculate the distance or similarity of this hash with the others. Method like Average Hashing has consistent results in building the hashes. Those hashes can be represented as binary or hexadecimal. In its applications, those hashes could also be represented as string or character. This give possibilities to use more distance and similarity measurement method to find similarities between hashes. Numeric similarity measurement like Euclidean Distance, Manhattan Distance and Cosine Similarity or string similarity method like Damerau-Levenshtein are possible to be used as similarity measure in application. For binary representation or Boolean representation, method like Hamming distance is also suitable for this. Method like Q-gram [1], N-gram [2], Longest Common Subsequence (LCSS) [3] and Dynamic Time Warping (DTW) [4] were also give more possibilities for more method combination usage in application.

For certain purposes many distance or similarity measurement methods have been studied in terms of performance an accuracy. Like shown in [5], that studying some similarity measures for recommender system using Apache Mahout, Euclidean Distance Measure consistently performs well and produces better quality results compared to other similarity measures [5]. Another comparison has been studied in [6] that implement and evaluate 9 different similarity measures and their variant. The results, for small time series data DTW and LCSS were significantly more accurate than Euclidean [6]. Those studies was give the idea for this study to do more comparison of similarity measures, as similarity measures has grown into larger variations. Another purpose is to get more obvious implementation view of those similarity measures under limited computing capabilities device like mobile phone.

† Content from this work may be used under the terms of the Creative Commons Attribution 3.0 licence. Any further distribution of this work must maintain attribution to the author(s) and the title of the work, journal citation and DOI.

Published under licence by IOP Publishing Ltd
2. Implementation and Testing

For the purpose of this study, we design implementation and testing scenario to evaluate similarity measures. We compare each similarity measure in term of speed and memory cost. We rule out accuracy as accuracy will strongly related to the method to build hash that we utilize. Average hashing method has been chosen for this study as this method was one of the simplest method for building the image hashes. As it the one of the simplest method, the cost and speed of it will not take majority resources when application executed. By combining average hashing and similarity measures we try to achieve more obvious similarity measures performance in application.

For testing purpose we built an Android based application for perceptual image similarity. This application runs under testing device with specification of 2Gb of RAMs, Cortex-A7 Quad-core 1.6 GHz CPU, 13 MP primary back camera and 6.0.1 Marshmallow Android OS. This specification was common specification for mobile phone released the last few years. We use three category of images dataset. First category was online images we get from http://www.gettyimages.com and http://www.kapanlagi.com. We also tests the application using images found under DCIM folder as second category image dataset and images found under WhatsApp Media folder as the third category dataset. This categories was intended to test the application for various numbers and sizes of images. This three category were also considered as commonly used images in real usage of mobile phone. Each dataset category has different number of images. Online image category has 42 images with average size of image 145kb, image category under DCIM folder has 76 images with larger average size 2.656k (2.6Mb), taken with 13MP main device camera. The last category is WhatsApp Media images, has 721 images with average size of image is 132kb.

We test by choosing a random query image then run the application for hashin and similarity measures function for each category. The results were recorded and analyzed. All similarity measures used in this study will be tested under the same scenario. Figure 1 show the flow of application for testing purpose.

As shown in Figure 1, the process to find images was linear search the images under the specified path of image category folder. This means that we process images one by one for every image found. The process includes Average Hashing for building the hash of query image and every image found, then calculate the similarity of the hashes using specified similarity measures method. For every complete applications flows executed will output the running time and maximum allocated memory of single similarity measure. For every similarity measures method the application will be executed for five times then calculate the averages of running time and maximum allocated memory. The results will be analyzed for conclusion.

3. Results and Analysis
The first result we want to achieve is the comparison of running time of application for each similarity measures method. Figure 2 show the comparison of application running time in average for five times of testing for each similarity measures.

![Running Time Comparison of Similarity Measures](image)

**Figure 2.** Running Time Comparison of Similarity Measures

From Figure 2 we got application running time for all similarity measures in each category were almost the same. This mean that in terms of speed, all similarity measures have similar performance under mobile application implementation. Although there is very slight differences running time in all similarity measures, but in general the running time is equal. The very significant difference is running time between categories. Category with smallest number and average size of images, Online image category has the fastest running time. In more various size and number of images like in WhatsApp images, the running time increased significantly. In largest size images category, DCIM images, running time was the highest. This show us that running time was increased in line with the size and number of images. The very dominant factor affecting running time is the Average hashing process itself. Since larger images need more time creating the hash. More number of images hash also affecting running time. From this results we can conclude that there is no fastest or slowest similarity measures method under mobile application implementation.

Our second test was to get maximum allocated memory for every similarity measures running under mobile application. Since Android has a very complicated memory management we are unable to get detailed results of allocated memory from each similarity measures implemented. What we have obtain was only average of maximum allocated memory for all similarity measures in each category. This number of allocated memory was affected by another android process like UI thread and other processes. So the number does not represent allocated memory only for each similarity measures process. Online image category has average of maximum allocated memory of 16.225 Mb. DCIM images was 111.51 Mb and WhatsApp image was 61.51 Mb. Since this results did not represent the allocated memory for similarity process only, we did not conclude any particular conclusion related to similarity measures memory cost.

### 4. Conclusion

Under mobile phone application, all similarity measures methods perform equally in terms of speed. Running time has no significant differences in all methods, although there is a very slight differences, but in general they are able to perform equally. Running time was very affected by the size and number of images processed, it increased in line with size and number of images. Memory usage for each running process of similarity measures under Android mobile phone was difficult to analyze, but from the tests we did not found any out of memory problem when running all methods.
References

[1] E. Ukkonen, “Approximate string-matching with q-grams and maximal matches,” Theor. Comput. Sci., vol. 92, no. 1, pp. 191–211, Jan. 1992.

[2] G. Kondrak, “N-Gram Similarity and Distance,” SPIRE 2005, pp. 115–126, 2005.

[3] W. J. Hsu and M. W. Du, “Computing a longest common subsequence for a set of strings,” Computing, vol. 24, no. March 1983, 1984.

[4] E. Keogh and C. A. Ratanamahatana, “Exact indexing of dynamic time warping Eamonn,” Knowl. Inf. Syst., vol. 7, pp. 358–386, 2005.

[5] S. Bagchi, “Performance and Quality Assessment of Similarity Measures in Collaborative Filtering Using Mahout,” Procedia - Procedia Comput. Sci., vol. 50, pp. 229–234, 2015.

[6] X. Wang, A. Mueen, H. Ding, G. Trajcevski, P. Scheuermann, and E. Keogh, “Experimental comparison of representation methods and distance measures for time series data,” Data Min. Knowl. Discov., vol. 26, no. 2, pp. 275–309, 2013.