Clustering for Photo Retrieval at Image CLEF 2008

Diana Inkpen, Marc Stogaitis, François DeGuire, and Muath Alzghool

School of Information Technology and Engineering
University of Ottawa
diana@site.uottawa.ca, mstog024@uottawa.ca,
fdegu079@uottawa.ca, alzghool@site.uottawa.ca

Abstract This paper presents the first participation of the University of Ottawa group in the Photo Retrieval task at Image CLEF 2008. Our system uses the following components: Lucene for text indexing and LIRE for image indexing. We experiment with several clustering methods in order to retrieve images from diverse clusters. The clustering methods are: k-means clustering, hierarchical clustering, and our own method based on WordNet. We present results for thirteen submitted runs, in order to compare retrieval based on text description, to image-only retrieval, and to merged retrieval, and to compare results for the different clustering methods.

Keywords Information retrieval, image retrieval, photographs, text retrieval, k-means clustering, agglomerative clustering, WordNet-based clustering.

1 Introduction

This paper presents the first participation of the University of Ottawa group in the photo retrieval track at Image CLEF 2008. This year's task focused on clustering images in order to retrieve images from different clusters. We present our system, followed by results for the submitted runs. We worked only with the English part of the collection (English captions and queries). The research questions that we are investigating include: what happens if we index only the text captions, only the images, or the captions and the images; what is the performance of the system with and without clustering. We investigate different types of clustering. First, the k-means clustering algorithm, then hierarchical clustering in three variants: based on average link similarity, complete link, and single link. Then we try our own clustering method, based on searching words from the query and from the text caption in the WordNet lexical knowledge base [1]. We present four versions of this algorithm.

2 System Description

The University of Ottawa's Image Retrieval system was built with off-the-shelf components. For text retrieval we used Lucene[1] and for image retrieval we use LIRE[2].

Lucene is an open-source text search engine that we have used in order to match the text captions of the queries with text description of the images from the collection. This tool provided us with a wide variety of options that we were able to use to quickly create our index and, later on, to perform the appropriate queries on it. It also provided us with parsers that we used in order to automatically remove the stopwords from the provided queries. We added our own parsing component that removed phrases from the narrative fields such as: "the relevant images should not include …".

LIRE is the tool that we have used in order to perform the content-based image search. The main advantage of this tool for us is that it is compatible with Lucene, thus providing a good solution for the problem at hand. Lucene and LIRE were designed to work together. Furthermore, these tools are also fairly simple to use, also providing straightforward methods for indexing and searching.

We have used a data fusion technique to merge the text retrieval with the image retrieval based on the method proposed in [3]. Their method, called combMNZ, sums up all the scores of a document multiplied by the number of non-zero scores of the document, as in formula 1:

$$combMNZ = \sum_{i \in IR\ schemes} score_i * n \qquad (1)$$

where $score_i$ is the similarity score of the document for the indexing technique used to retrieve this document (text or image indexing), and n is the number of non-zero scores of the document.

Since there are there are two indexing techniques based on text and image retrieval from different systems, these systems will generate different ranges of similarity scores, so it is necessary to normalize the similarity scores of the document. Lee [2] proposed a normalization method by utilizing the maximum and minimum scores for each weighting scheme as defined by formula 2.

$$NormalizedScore = \frac{score - MinScore}{MaxScore - MinScore} \qquad (2)$$

We have adapted combMNZ to carry a weight for each indexing technique. Our data fusion model uses a fusion formula that we call WCombMNZ represented by formula 3.

[1] Lucene text search engine library, http://lucene.apache.org/java/docs/
[2] LIRE open source java content-based image retrieval library, http://www.semanticmetadata.net/lire/

$$WCombMNZ = \sum_{i \in \text{indexing schemes}} W_i * NormalizedScore_i * n \quad (3)$$

where W_i is a predefinded weight associated with each indexing technique's results, n is the number of non-zero scores of the document, and the $NormalizedScore_i$ is calculated by formula 2 as described before.

We have tried different weights (W_i) for each indexing scheme, as shown in the next section, usually giving more weight to the text retrieval.

We added a **clustering component** that clusters the text captions. Our text clustering component was implemented with the use of the Dragon[3] Toolkit. Our text clustering works as follows: using the Dragon Toolkit, a second index of all image annotations, indexed by the doc number field, was created. When performing a clustering operation during a search, the program goes through the Lucene results, extracts their doc number, and uses them to generate a list of Dragon Toolkit documents. The Dragon Toolkit algorithm is then called, which clusters the data based on its own index (the second index described above).

The way we integrated the results of the clustering component into the system was by post processing the search results for the test queries. After the clusters were formed, the results were re-ranked so that as many as possible distinct clusters were retrieved in the first 20 results, keeping only the first image from each cluster. By the first image from a cluster we mean the image with the highest similarity score. The other images from the same cluster were re-arranged below the limit of first 20, in the order of their scores.

Our clustering component used the k-means clustering algorithm, a hierarchical agglomerative clustering algorithm, and a new clustering method that we designed based on WordNet search.

The WordNet-based runs work as follows. The algorithm takes as input the ranked results list that was created by our standard text/image search. It then cycles through each word of the first document, looking for words that match the current clustering criteria. For example, if the query indicates that we should cluster based on animals, then the algorithm will try to spot a word in the document that it recognizes as an animal (for example, the word "dog"). This is accomplished by performing a recursive search up the WordNet semantic relation called hypernym ("is a" relation). We also included the "is an instance of" relation together with the hypermyms. Here is an example of this process: suppose we have the sentence "Run fast fox and don't look back". Also assume that our clustering criterion is "animal". The algorithm would first take the word "run" and, for each of the possible word senses, recursively look through its hypernym ancestry to see if it can find the word "animal". In this case, it would not find it. It would then move on to the next word, "fast". After not finding it for this word, it would move on to the word "fox". In this case, it would find "animal" somewhere in its hypernym tree and would therefore be able to return the word "fox" as being related to the cluster. Once the algorithm has identified the

[3] Dragon Toolkit http://www.dragontoolkit.org/textcluster.asp

relevant word which describes the current document in relation to the cluster, it then checks if we have already entered a result in the top 20 results with that word. If we have not, it adds this result to the current list of top results. If it is already in the list, it holds on to this result and will eventually append it lower in the list (after the first 20 results). The performance of the algorithm seems pretty good. It is particularly good at detecting locations, countries, states etc. To help enhance this capability, the algorithm looks at the <location> field of the documents first.

3 Experimental Results

Table 1 shows the results of the thirteen submitted results on the test topics/queries. The evaluation measure we report are standard measures computed with the trec_eval script: MAP (Mean Average Precision) and the precisions at 20 documents. In addition, we report the number of distinct clusters included in the first 20 documents, which was the main focus of this year's task.

Next, we describe the first 9 runs.

Run1 is a text only search followed by re-arranging using the k-means clustering algorithm.

Run2 merged the results of four different retrieval runs, one with text only and one for each of the three sample images (similarity with that image). The results were merged with the following weights: 70% for the text retrieval, and 10% for each image retrieval.

Run3 is the same as Run2, but with the following weights: 85% for the text retrieval, and 5% for each image retrieval.

Run4 is the same as Run3 but with the following weights: 55% for the text retrieval, and 15% for each image retrieval.

Run5 was an image only run. It was done by assigning the following weights: 0% for the text retrieval, and 33% for each image retrieval.

Run6 is a run in which the clustering step was disabled so that the order that we present is the same as from a text & image retrieval only run. The weights that were used are: 55% for the text retrieval, and 15% for each image retrieval.

Run7 uses the Hierarchical Clustering algorithm instead of the K-Means. It uses Average Linkage to measure similarity. The weights that were used are: 55% for the text retrieval, and 15% for each image retrieval.

Run8 uses the Hierarchical Clustering algorithm instead of the K-Means. It uses Complete Linkage to measure similarity. The weights that were used are: 55% for the text retrieval, and 15% for each image retrieval.

Run9 uses the Hierarchical Clustering algorithm instead of the K-Means. It uses Single Linkage to measure similarity. The weights that were used are: 55% for the text retrieval, 15% for each image retrieval.

Three more runs use a different way of re-arranging the data that is based on WordNet, as explained at the end of the previous section. Below is an explanation of the next three runs that we performed using our WordNet-based clustering algorithm.

Run10 was a basic WordNet run where the algorithm functions as mentioned above.

Run11 was the same as Run10, except that instead of using the clustering field from the query directly, we process it a bit (basically, we only take the first word of the clustering criterion). This helped fix a few exceptions that were found in the clustering queries, such as a cluster called “vehicle type”, which would not be recognized by WordNet. WordNet will however recognize vehicle and should be able to cluster properly with that term.

Run12 is the same as Run11 with the added constraint that we only take the first 3 definitions of a word sense when looking for hypernym links. This helped fixed problems with things like the word “blue” being recognized as an animal since its 7th word sense according to WordNet is a “butterfly”.

In the previous algorithm, there were cases when none of the terms of a document were found to match the category, according to WordNet. In the previous runs, we just kept the result where it was. However, another option was to only put results that had matches in WordNet in the 20 first documents. Run 13 tries this second option.

Therefore, Run13 is similar to Run12 except that we now make sure that the first 20 results returned contain words that are found to match the clustering criterion according to WordNet.

Table 1. Results of the thirteen submitted runs. All of them are for English queries and English captions of the images (EN-EN). They are all automatic, no manual feedback was involved (AUTO). TXT indicates that the image annotation text was used during the search. IMG indicates that the images themselves were used during the search. The names of the runs also contain a code for the clustering algorithm (KM for k-means clustering; HA – hierarchical clustering, average linkage; HC – hierarchical clustering, complete linkage; HS – hierarchical clustering, single linkage; WN – WordNet-based clustering; NO – no clustering).

Run name	CR@20	P@20	MAP
UOt01_EN_EN_AUTO_TXT_KM	0.3684	0.3333	0.2314
UOt02_EN_EN_AUTO_TXTIMG_KM	0.3767	0.3474	0.2429
UOt03_EN_EN_AUTO_TXTIMG_KM	0.3716	0.3436	0.2384
UOt04_EN_EN_AUTO_TXTIMG_KM	0.3970	0.3615	0.2479
UOt05_EN_EN_AUTO_IMG_KM	0.2694	0.1590	0.0693
UOt06_EN_EN_AUTO_TXTIMG_NO	0.3970	**0.3654**	**0.2490**
UOt07_EN_EN_AUTO_TXTIMG_HA	0.1488	0.0705	0.1665
UOt08_EN_EN_AUTO_TXTIMG_HC	0.3149	0.1333	0.1869
UOt09_EN_EN_AUTO_TXTIMG_HS	0.1204	0.0590	0.1625
UOt10_EN_EN_AUTO_TXTIMG_WN	**0.4102**	0.2756	0.1382
UOt11_EN_EN_AUTO_TXTIMG_WN	0.4038	0.2679	0.1391
UOt12_EN_EN_AUTO_TXTIMG_WN	0.4027	0.2705	0.1372
UOt13_EN_EN_AUTO_TXTIMG_WN	0.4069	0.2026	0.1965

Discussion of the results

The first five runs used the k-means clustering algorithm, Run1 used only the text, and Run5 used only the images. We can see that using only the images the performance is the lowest. Text-only retrieval was pretty good, and using a combination of text and image retrieval brings a small improvement over text-only. Among the first 5 runs, the best weights for merging text and image retrieval results were for Run4, namely 55% for the text retrieval, and 15% for each image retrieval. These weights for merging results were kept for the remaining runs, with good results.

Run6, with no clustering achieved the best P@20 and MAP core, and a score of 0.3970 from CR@20. Adding clustering we reduces P@20 and MAP score, but some of the methods improved the number of distinct clusters retrieved in the first 20.

The four WordNet-based clustering methods worked best in terms of retrieving many relevant clusters, especially the version from Run 10, which archived 0.4102 for CR@20. The hierarchical clustering was the worst, especially the complete link and single link.

4 Conclusion

Our system used merged results from Lucene for text retrieval and Lire for images search. We incorporated a clustering component. We experimented with the k-means algorithm, agglomerative clustering, and a WordNet-based clustering algorithm.
Our experiments showed that text retrieval works well, and adding image similarity brings a bit of improvement. In terms of retrieving many different clusters, our WordNet-based algorithm worked best.

References

1. Fellbaum, Christiane (ed.): WordNet, An Electronic Lexical Database, The MIT Press, 1998
2. Lee, J.H. 1995. Combining multiple evidence from different properties of weighting schemes. In Proceedings of the Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval (Seattle, Washington, United States, 1995). ACM, 1995.
3. Shaw, J.A. and Fox, E.A. 1994. Combination of Multiple Searches. National Institute of Standards and Technology Special Publication, 1994.