YFCC100M HybridNet fc6 Deep Features for Content-Based Image Retrieval

Giuseppe Amato
ISTI-CNR
via G. Moruzzi, 1
56124 Pisa, Italy
giuseppe.amato@isti.cnr.it

Fabrizio Falchi
ISTI-CNR
via G. Moruzzi, 1
56124 Pisa, Italy
fabrizio.falchi@isti.cnr.it

Claudio Gennaro
ISTI-CNR
via G. Moruzzi, 1
56124 Pisa, Italy
claudio.gennaro@isti.cnr.it

Fausto Rabitti
ISTI-CNR
via G. Moruzzi, 1
56124 Pisa, Italy
fausto.rabitti@isti.cnr.it

ABSTRACT

This paper presents a corpus of deep features extracted from the YFCC100M images considering the *fc6* hidden layer activation of the HybridNet deep convolutional neural network. For a set of random selected queries we made available k-NN results obtained sequentially scanning the entire set features comparing both using the Euclidean and Hamming Distance on a binarized version of the features. This set of results is ground truth for evaluating Content-Based Image Retrieval (CBIR) systems that use approximate similarity search methods for efficient and scalable indexing. Moreover, we present experimental results obtained indexing this corpus with two distinct approaches: the Metric Inverted File and the Lucene Quantization. These two CBIR systems are public available online allowing real-time search using both internal and external queries.

Keywords

YFCC100M; Deep Features; Content-Based Image Retrieval; Multimedia Information Retrieval

1. INTRODUCTION

Deep learning methods are "representation-learning methods with multiple levels of representation, obtained by composing simple but non-linear modules that each transform the representation at one level (starting with the raw input) into a representation at a higher, slightly more abstract level" [20]. Starting from 2012 [19], Deep Convolutional Neural Networks (DCCNs) have attracted enormous interest within the Computer Vision community because of the state-of-the-art results achieved in image classification tasks. The relevance of the internal representation learned by the neural network during training have been proved by recent works. In particular, the activation produced by an image within the intermediate layers of a DCNN can be used as a high-level descriptor of the image visual content [26, 10, 12, 24].

The Yahoo Flickr Creative Commons 100M (YFCC100M) [28] dataset was created in 2014 as part of the Yahoo Webscope program[1]. The dataset consists of approximately 99.2 million photos and 0.8 million videos, all uploaded to Flickr between 2004 and 2014 and published under a Creative Commons commercial or non-commercial license. Metadata are publicly available through the Yahoo! Webscope.

The importance of having a very large dataset of publicly available features has been proven by the Content-based Photo Image Retrieval (CoPhIR) [11] released on 2009, which has been used by many scientists working in the field of very large scale similarity search algorithms [16, 22, 17, 6]. The CoPhIR dataset consists of MPEG-7 features extracted from about 107M Flickr images. Considering that the extracted deep features are very high dimensional, consisting of vectors having 4,096 dimensions, and that the total size of the dataset is close to one hundred million, building an index that is able to interactively respond to similarity queries using limited computing and storage resources is a significant challenge.

In this paper, we present: the deep features extracted from the YFCC100M images [1] considering the *fc6* hidden layer activation of the HybridNet DCNN [29]; two sets of ground truth k-NN results using the Euclidean distance and a simple but effective binarization approach [4] that allows compact representation and fast comparison using the Hamming distance; two online Content-Based Image Retrieval (CBIR) systems indexing the whole corpus [8, 2].

The two CBIR systems use two very different approach: the Metric Inverted File (MI-File) (Section 4.1.1) and the Lucene Quantization (Section 4.1.2). The MI-File is a permutation based method that uses inverted files for fast approximate similarity search. The Lucene Quantization (LuQ) exploits the sparsity of the deep features using a quantization approach to allow text encoding. In Section 4 we report experimental results on both. The results show that MI-File

MMCommons'16, October 16 2016, Amsterdam, Netherlands

ISBN 978-1-4503-4515-6/16/10...$15.00

DOI: http://dx.doi.org/10.1145/2983554.2983557

[1] https://webscope.sandbox.yahoo.com/

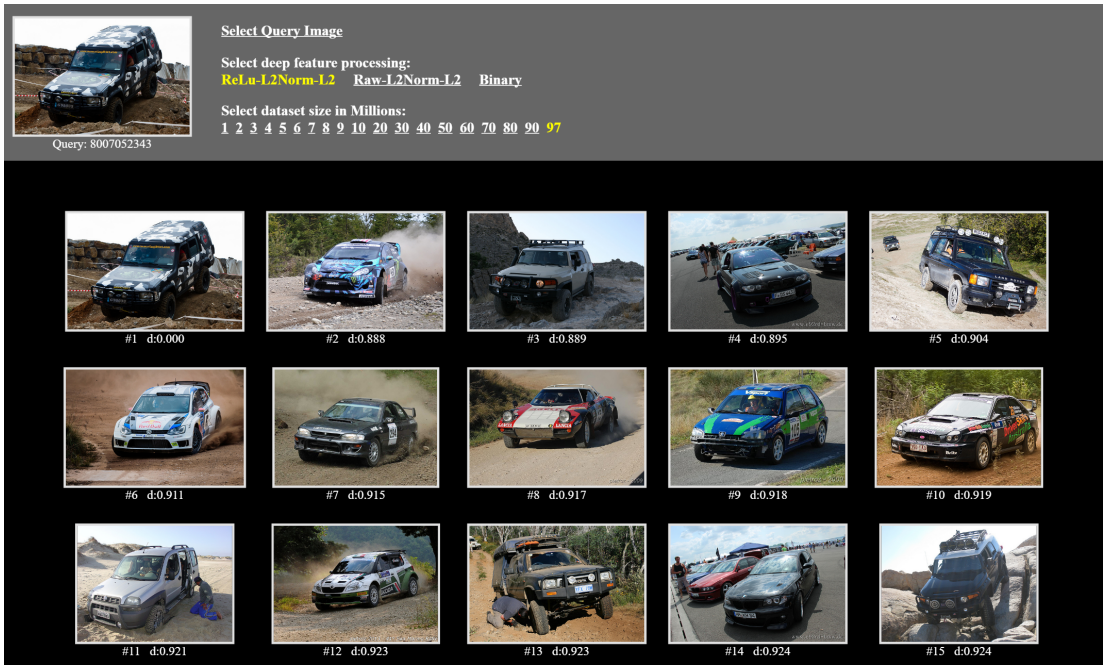


Figure 1: Content-Based Image Retrieval Example using the presented HNfc6 features.

is more effective and efficient. However, the simplicity of the LuQ and the use of a standard text search engine (i.e., Lucene) could made it preferable in many scenarios. Moreover, LuQ allows text (over the metadata) and/or content-based image searchers.

2. RELATED WORK

Deep Convolutional Neural Networks (DCNNs) have recently become state-of-the-art approach for many computer vision task such as image classification [19, 27],image retrieval [15, 10, 24, 19, 27]and object recognition [15]. The use of the activation of intermediate layers as a high-level descriptor of the image visual content has been also proven effective by many recent works [26, 10, 12, 24].

Rectified Linear Unit (ReLU) is part of almost all of the DCNN models and is typically applied also for extracting deep features from images [15, 12]. However, there are works in which the ReLU was omitted [26, 10, 24]. The L2 Normalization of the feature in order to and compare using the Euclidean distance is a standard de-facto for deep features [26, 12]. It is worth to mention, that the resulting ranking of similarity search is equivalent to the cosine similarity. Principal Component Analysis has been successfully used in [25, 10].

The Multimedia Commons initiative is an effort to develop and share sets of computed features and ground truth annotations for the Yahoo Flickr Creative Commons 100 Million dataset (YFCC100M), which contains around 99.2 million images and nearly 800,000 videos from Flickr, all shared under Creative Commons licenses. An in depth presentation of the dataset is given in [28].

When we started the research work reported in this paper, the deep features extracted from YFCC100M presented in [23] were not yet available through the Multimedia Commons Initiative [3]. Thus, in this paper we don't make a comparison. We believe that having two distinct sets of features and the two online CBIR systems we presents in this paper will allow comparison.

3. THE HNFC6 DEEP FEATURES

HybridNet is essentially the AlexNet [19] DCNN trained on both the ImageNet subset used for the ILSVRC competition, and the MIT Places Database [29] commonly used for Scene Recognition. The training set of HybridNet consists of 3.5 million images from 1,183 categories. We extracted the features using the trained model public available for the popular Caffe framework [18]. Many deep neural network models and in particular trained models are available for this framework at[2].

We chose the HybridNet for several reasons: first, its architecture is the same as the famous AlexNet [19]; second, the HybridNet has been trained on the ImageNet subset used for ILSVRC competitions (as many others) and the Places Database [29]; last, but not least, experiments conducted on various datasets demonstrate the good transferability of the learning [29, 12, 9]. Originally proposed in [29], HybridNet has been used in [29, 12, 9]. The results reported in [12] show that deep features extracted from the HybridNet outperforms various architectures trained only on ImageNet, on both InriaHolidays and OxforBuilding benchmarks. On the UKBench and better performance were obtained by the VGGNet [27] while AlexNet was preferable on the Graphics [13] benchmarks. We believe that overall the HybridNet is a good choice for deep features because of the larger set and diversity of training set.

We decided to use the activation of the first fully connected layer (i.e., *fc6*) given the results reported in [15, 10, 12]. It is worth to mention that the activation of the second fully connected layer (i.e., *fc7*) can be obtained from the *fc6*

[2]https://github.com/BVLC/caffe/wiki/Model-Zoo

activation with a simple matrix operation using the weights and biases that we made public available at [1]. In the rest of the paper we call these features HNfc6.

Rectified Linear Unit (ReLU) is part of almost all of the DCNN models and is typically applied also for extracting deep features from images [15, 12]. However, there are works in which the ReLU was omitted [26, 10, 24]. The L2 Normalization of the features in order to compare by using the Euclidean distance is a standard de-facto for deep features [26, 12]. It is worth to mention, that the resulting ranking of similarity search is equivalent to the cosine similarity. Principal Component Analysis has been successfully used in [25, 10].

The deep features we present are 4,096 dimensional L2 Normalized vectors corresponding to the activation of the neurons of the HybridNet *fc6* layer after the ReLU. This activation function, which is part of the HybridNet Convolutional Neural Network, simply sets to zero all the elements of the vectors that are negative. The distance used to compare is the Euclidean (aka L2 distance). The deep features made available through the Multimedia Commons Initiative have been extracted after the ReLU given that this is the standard approach. However, through our website, we also give the values obtained without the ReLU.

3.1 Binary Features

In [4], a simple binarization of deep features was shown to lead to a negligible performance drop for both classification and detection. In particular, PASCAL-CLS performance was nearly identical before and after binarization for both fc6 and fc7 fully-connected layers. This binarization consists in encoding the positive values of the activation as 1s, while zeros and negative (in case values have been extracted before ReLU) values as 0s. We evaluated k-NN results also for these features using the Hamming distance.

We conducted our own experiments on INRIA Holidays using both the deep features L2 Normalized and their binarization. For this specific task the binary features performed even better than their float counterpart obtaining a mAP of 0.76 against the 0.75 obtained by the original features.

3.2 k-NN ground truth results

On [1], we report the results obtained for k-NN queries on 1,000 randomly selected images with $k = 10,001$. An example of the results available on our site is given in Figure 1. The webpage in the figure reports the results obtained comparing the HybridNet *fc6* Deep Features after the ReLU and using the Euclidean Distance. From the website, it is also possible to see the results comparing the features before the ReLU activation function and also considering the binarization described before.

3.3 Statistics

In this Section, we report some statistical information for the deep features we extracted. We first show some statistics about the sparsity of the proposed deep features. In Figure 2, we report the cumulative distribution function and probability density function for the number of positive values. Please note that we made available features before and after the ReLU and that these statistics are the very same for both. In particular, the amount of positives has: min=221, $mode$=919, $median$=972, $mean$=972 and max=2,201. The results show that on average, each image has about 25% of positive elements between the 4,096 values of the feature vector. Thus, the *ReLU-L2Norm* vectors are quite sparse in the population sparsity sense [21]. The sparsity of the features vector is relevant for indexing using inverted files. As an example, the LuQ approach presented in Section 4.1.2, leverage on this. Moreover, for the binarized features (see Section 3.1 these statistics report the distribution of one and zeros.

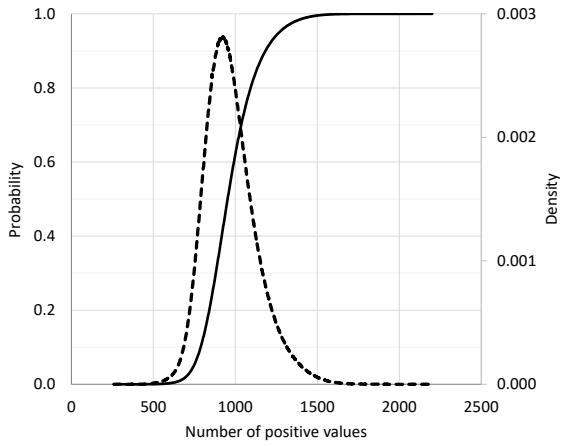


Figure 2: Cumulative distribution function (probability density function as dotted line with secondary vertical axes) of positive values per deep features.

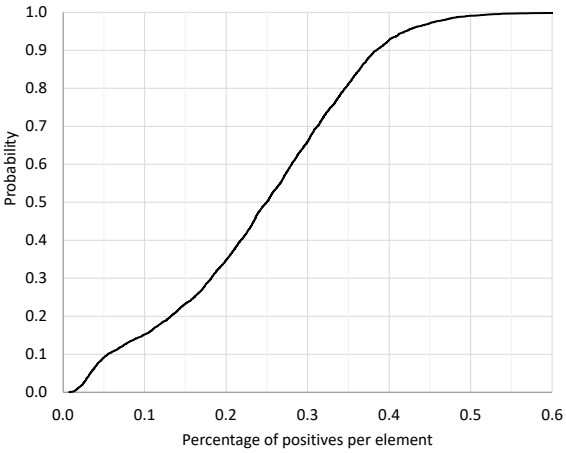
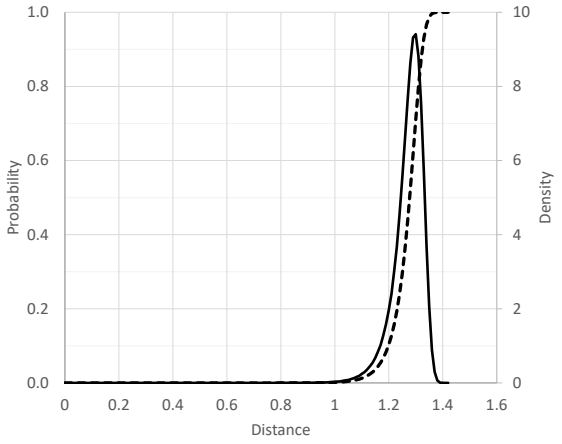
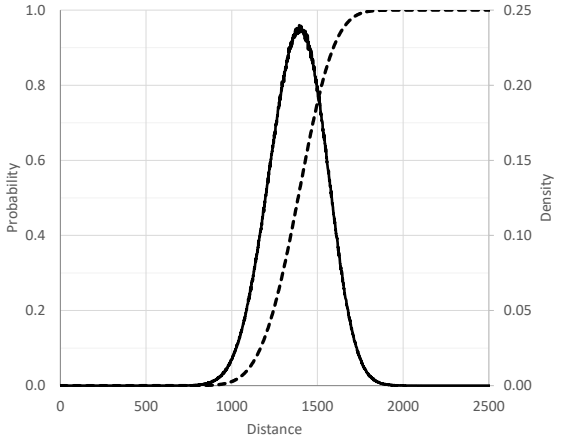


Figure 3: Cumulative distribution function of percentage of positives per element (single dimension) of the 4,096 dimensional presented deep features.

In Figure 3, we consider each vector component (i.e., each neuron in the *fc6* layer) individually. We report the cumulative distribution related to the percentage of positives for an element all over the YFCC100M dataset. The graph shows that 10% of the elements have positive values for only 5% of the images, while 10% of elements are positive in at least 40% of the images. Thus, there are 10% of neurons that are activated only on 5% of the images while a distinct 10% is active in more than 40% of the images. In other words, while population sparsity holds (has seen before), lifetime sparsity and high dispersal [21] can't be considered properties of this feature.



(a) Euclidean



(b) Hamming

Figure 4: Cumulative distribution function (probability density function as dotted line with secondary vertical axes) for both Euclidean and Hamming distances

In Figure 4, we report the cumulative and probability density functions for both the Euclidean distance applied to the HNfc6 features (a) and the Hamming distance applied to their binarization (b) as reported in Section 3.1. The Figures show near Gaussian Distribution for both distances with the Hamming even more Gaussian.

In Table 1, we report the metric space intrinsic dimensionality [14] defined as $\mu^2/(2\sigma^2)$, and other information also related to these distributions. The statistics show, as expected, a lower intrinsic dimensionality for the binarized features that should then be easier to index for similarity search.

The *ReLU-L2Norm* features in conjunction with the Euclidean distance appear to be very hard to index. The curse of dimensionality is revealed in the graph and confirmed by the high *intrinsic dimensionality*. On the contrary, the *Binary* features combined with the Hamming distance reveal an intrinsic dimensionality of only 35 and the distribution is very similar to a Gaussian.

In Figure 5, we analyze the amount of intersection between the results of the 1,000 k-NN queries we performed

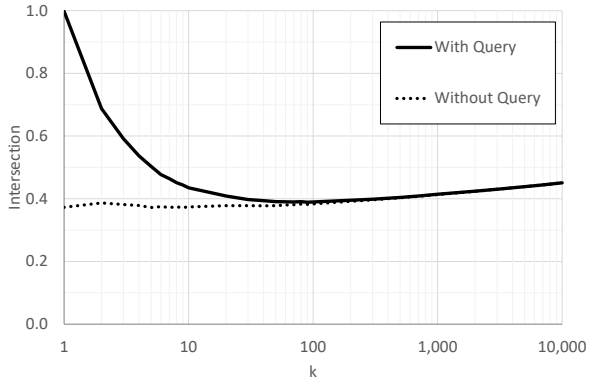


Figure 5: Intersection between Euclidean and Hamming k-NN results varying k used for the k-NN search (with and without the query in the dataset).

by sequentially scanning the dataset for the aim of creating the ground truths that we made public available on our website. We compare the results obtained with the *ReLU-L2Norm* and *Binary* features reporting the average intersection varying k. We also considered the cases in which the query is between the results or is removed. The most interesting curve is the one in which we do not consider the query itself in the results. We obtained an intersection of about 40% for k between 1 and 1,000.

Table 1: Distance-Related Statistics

	Euclidean	Hamming
Mean (μ)	1.27	1383
Standard Deviation (σ)	0.054	164.5
Intrinsic Dim. ($\mu^2/(2\sigma^2)$)	276	35
Mode	1.28	1388
Variance	0.0029	27057

4. ONLINE CBIR SYSTEMS

We made public available online [8, 2] two distinct systems that allow CBIR using the proposed features.

The first CBIR system is based on the Metric Inverted File (MI-File) technique [7]. MI-File uses an inverted file to store relationships between permutations, and many approximations and optimizations to improve both efficiency and effectiveness. The basic idea is that entries (the lexicon) of the inverted file are the set of permutants (or pivots) P. The posting list associated with an entry $p_i \in P$ is a list of pairs $(o, \Pi_o^{-1}(i))$, $o \in C$, i.e. a list where each object o of the dataset C is associated with the position of the pivot p_i in Π_o.

The second CBIR system (*LuQ*) [5] is based on quantization of the features. *LuQ* represents each deep feature as a text document and uses a NoSQL database (Apache Lucene) for efficiently indexing and searching purposes. The whole Lucene 5.5 archive of LuQ approach is also available for download from the deep features website [1]. The advantage of this representation is that can be directly queried with Lucene by simply extracting the term vectors from the archive.

4.1 Performance Evaluation of the indexes

The two CBIR systems use approximate similarity search indexes, i.e., indexes that introduce some errors in the search results, to provide very high query execution efficiency. In the following, we evaluate the introduced approximation using the ground truth k-NN results described in Section 3.2.

4.1.1 Metric Inverted File: MI-File

MI-File [7] is a permutation based method that uses inverted files to perform fast approximate execution of k-NN queries. MI-File offers the following parameters to trade efficiency with accuracy:

- Amplification factor amp: when searching for the k-NN the MI-File retrieves a candidate set of $k' = amp \cdot k$ objects, reorders it according to the original distance function, and returns the top-k objects. The larger amp, the higher the search cost, and the higher the accuracy.
- data object permutation length k_i: the permutation representing a data object is obtained using the k_i closest reference objects out of the total set of reference objects. The value of k_i determines the number of posting lists containing a reference to the object being inserted.
- Query permutation length k_s: the permutation representing the query is obtained using the k_s closest reference objects out of the total set of reference objects. The value of k_s determines the number of posting lists accessed during a query execution.
- Maximum position difference mpd: posting lists are scanned considering entries referring objects whose reference objects position difference in their permutation, with respect to the query permutation, is at most mpd. The higher mpd, the more entries are retrieved from the posting lists.

Please see [7] for further details on the MI-File and its parameters usage.

Each HNfc6 feature consists of 4,096 floats. If floats are represented with 4 bytes, an uncompressed database of 100M features requires roughly 1.5 TB of storage space. In order to reduce the size of the database, we binarized the features, as described in Section 3.1. In this case, given that 4,096 bits are stored in 512 bytes, an uncompressed database of 100M features requires roughly just 46 GB of storage space. We used the Hamming distance to estimate similarity between binarized features.

In our experiments, we indexed the entire binarized HNfc6 dataset, using $k_i = 100$. The total number of reference objects for building permutations is 20,000. The total index size is roughly 36 GB.

The queries were executed with amp ranging from 1 to 70. The values used for k_s ranged from 1 to 50 and $mpd = k_s$. We executed k-NN queries with $k = 100$, using the 1,000 queries of the ground truth, and performance measures were obtained as average of the measures computed for all query. Results are shown in Figure 6.

The upper graph shows the relationships between the number of disk blocks accessed and the quality of results. Disk block size is 4K bytes. Every plot corresponds to different

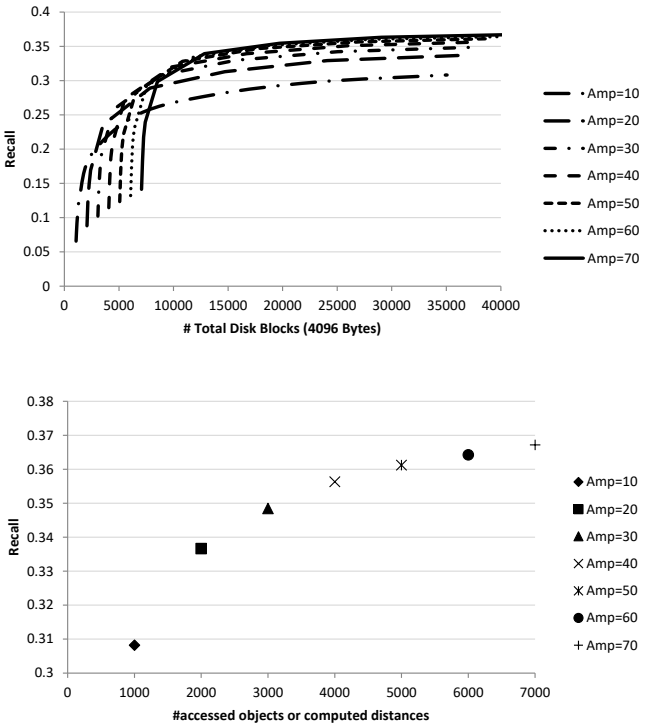


Figure 6: Performance of the MI-File with the YFCC100M-HNfc6 binarized features using Euclidean distance ground truth.

setting for amp, and the amount of disk blocks accessed was tuned by setting the k_s parameter. When indexing the binarized YFCC100M-HNfc6 features, MI-File reaches a recall of 37%, with respect to the ground truth, with a number of disk block accesses around 30,000.

The bottom graph shows the relationships between the number of database objects accessed and the quality of the results. Here, the index access cost is not taken into account. In this case, 7,000 objects out of 100M total objects have to be accessed to have a recall of almost 37%.

However, as observed in Section 3.1, some works in literature [4], and experiments carried out by ourselves, have shown that the binarized features are comparable and sometimes more effective than float deep features, in multimedia information retrieval and classification tasks. The evaluation above was obtained by comparing the results of the approximate similarity search algorithms on the binarized features against the ground truth created using the float features and the Euclidean distance. To have a more objective estimation of the performance of the MI-File, we compared the obtained results with a new ground truth obtained using directly the binary features and the Hamming distance. The process for creating this ground truth, is the same that described above for the float feature ground truth, with the difference that binary features and Hamming distance are used. In particular, the 1,000 queries are the same in both cases. Results of this additional evaluation are reported in Figure 7. We can see that, in this case, the recall arrives up to 75%, with the same query execution cost than before. The approximation introduced by the index, is therefore negligi-

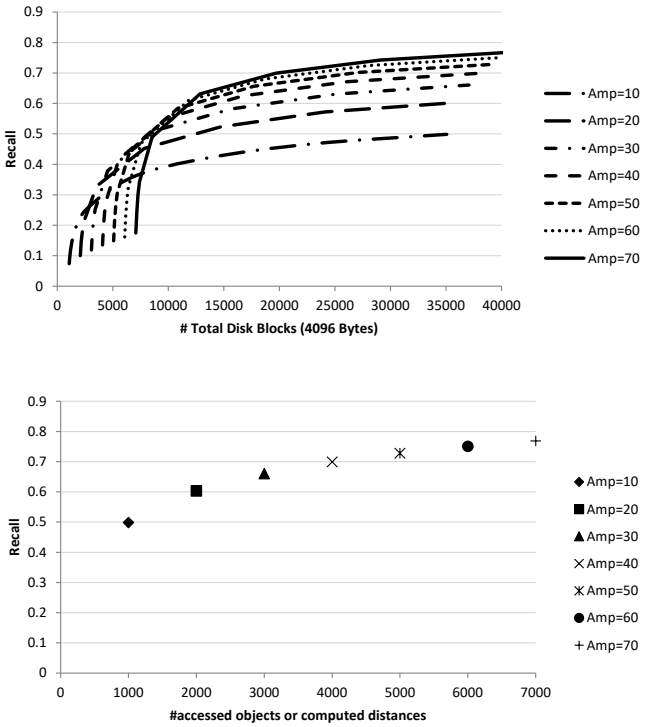


Figure 7: Performance of the MI-File with the YFCC100M-HNfc6 binarized features, using the Hamming distance over binarized features ground truth.

ble and most of the recall drop was due to the fact that comparison was made against the float ground truth.

It is also worth mentioning that, in this evaluation, precision is always equal to recall. In fact the denominator, in the precision and recall definitions, is equal to the total number of retrieved objects, which is $k = 100$ both for approximate and exact search, and the numerator is always the number of correct objects retrieved.

4.1.2 Lucene Quantization: LuQ

A convenient way of representing the HNfc6 features to encode them in text form and use a text retrieval engine to perform image similarity search. This approach, called *LuQ*, has been presented in [5]. It exploits the quantization of the vector components of the deep features, in which each real-valued vector component x_i is transformed in a natural number n_i given by $\lfloor Qx_i \rfloor$; where $\lfloor \rfloor$ denotes the floor function and Q is a multiplication factor > 1 that works as a *quantization factor*. n_i are then used as term frequencies for the "term-components" of the text document representing the feature vectors.

To employ this idea, in LuQ, we provide a text encoding for the DCNN feature vectors that guarantees the direct proportionality between the feature components and the term frequencies. Let $\mathbf{w} = (w_1, \ldots, w_m)$ denote the L2-normalized DCNN vector of m dimensions. Firstly, we associated each of its component w_i with a unique alphanumeric term τ_i (for instance, the prefix 'f' followed by the numeric values corresponding to the index i). The text encoding $doc(\mathbf{w})$ corresponding to the vector $\mathbf{w}$ is given by:

$$doc(\mathbf{w}) = \bigcup_{i=1}^{m} \bigcup_{j=1}^{\lfloor Qw_i \rfloor} \tau_i$$

Where $\lfloor \rfloor$ denotes the floor function and Q is a multiplication factor > 1 that works as a *quantization factor*[3].

Therefore, we form the text encoding of w_i by repeating the term τ_i for the non-zero components a number of times directly proportional to w_i. This process introduces a quantization error due to the representation of float components in integers. However, as we will see, this error does not affect the retrieval effectiveness. The accuracy of this approximation depends on the factor Q, used to transform the vector $\mathbf{w}$. For instance, if we fix $Q = 2$, for $w_i < 0.5$, $\lfloor Qw_i \rfloor = 0$, while for $w_i \geq 0.5$, $\lfloor Qw_i \rfloor \geq 1$. In contrast, the smaller we set Q the smaller the inverted index will be. This is because the floor function will set to zero more entries of the posting lists. Hence, we have to find a good trade-off between the effectiveness of the retrieval system and its space occupation.

For instance, if we fix $Q = 2$, for $w_i < 0.5$, $\lfloor Qw_i \rfloor = 0$, while for $w_i \geq 0.5$, $\lfloor Qw_i \rfloor \geq 1$. In contrast, the smaller we set Q the smaller the inverted index will be. This is because the floor function will set to zero more entries of the posting lists. Hence, we have to find a good trade-off between the effectiveness of the retrieval system and its space occupation.

For example, if we set $Q = 30$ and we have for instance a feature vector with just three components $\mathbf{w} = (.01, .15, .09)$ the corresponding integer-representation of the vector will be $(0, 4, 2)$ and its text encoding will be: $doc(\mathbf{w})$ ="f2 f2 f2 f2 f3 f3".

Since on average the 25% of the DCNN features are non-zero (in our specific case the fc6 layer), the size of their corresponding text encoding will have a small fraction of the unique terms present in the whole dictionary (composed of 4,096 terms). In our case, on average a document contains about 275 unique terms, which is about 6.7% of the dictionary because of quantization that set to zero the feature components smaller than $1/Q$.

When we have to process similarity search, therefore the search engine has to treat query of that size. These unusual long queries, however, can affect the response time if the inverted index contains millions of items.

A quite intuitive way to overcome this issue is to reduce the size of the query by exploiting the knowledge of the *tf*idf* (i.e., term frequency * inverse document frequency) statistic of the text encoding, which comes for free in standard full-text retrieval engines. We can retain the elements of the query that exhibit greater values of *tf*idf* and eliminate the others. For instance, for a query of about 275 unique term on average, we can take the first ten terms that exhibits the highest *tf*idf*, we obtain a query time reduction of about 96%.

This query reduction comes, however, with a price: it decreases the precision of results. To attenuate this problem, for a top-k query, we reorder the results using the cosine similarity between the original query (i.e., the one without reduction) and the first $C_r \times k$ candidate documents retrieved. Where C_r is the amplification factor introduced above for MI-File.

[3]By abuse of notation, we denote the space-separated concatenation of keywords with the union operator $\cup$.

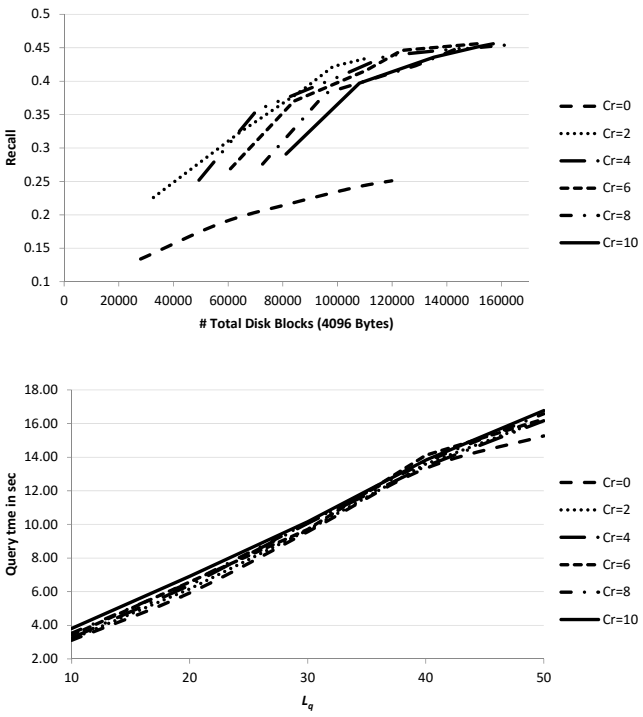


Figure 8: Performance of LuQ with the YFCC100M-HNfc6 binarized features.

In order to calculate the cosine similarity of the original query and the $C_r \times k$ candidates, we have to reconstruct the quantized features by accessing to the posting list of the document returned by the search engine. This approach does not affect significantly the efficiency of the query but can offer great improvements in terms of effectiveness.

In Figure 8, we have reported the relationships between the number of disk blocks accessed and the quality of results as recall and the average search time of 100 queries of the ground truth for k=100. In both experiments, we test the impact of the query length L_q (ranging from 10 to 50 terms) and the amplification factor C_r (ranging from 0 to 10) while Q was set to 30. When the implication factor assumes the conventional value $C_r = 0$, we mean that no reordering was used. The recall grows rapidly as the query length L_q increases and reaches about 45% independently to C_r. Obviously, L_q also affects the performance in terms of block disks and consequently the query time, which is practically not influenced by C_r as the second graph show. The query times have been obtained in an Intel Core i7 computer equipped with a SSD disk using Lucene 5.5.0.

4.2 Discussion

The two CBIR systems that we made available are very different. While MI-File is a permutation based method that uses inverted files, LuQ exploits the sparsity of the deep features using a quantization approach to allow text encoding. Considering that both methods are approximate methods, we have to compare the trade-off between effectiveness and efficiency of the two approaches. The overall best performance have been obtained with MI-File. MI-File was able to achieve better approximation (more accurate results) for the same number of total disk blocks accessed with respect to LuQ. It also was able to achieve the same accuracy of LuQ if we compare the two approaches fixing the number of accessed blocks.

However, LuQ is simpler and use of standard text search engine (i.e., Lucene). The overall goal of the systems we made public available is allowing online searching of the YFCC100M images using content-base image retrieval. The LuQ CBIR system also allows searching using text in the metadata and combining text and deep features. It is worth to mention that the offline indexing is cheaper for LuQ. In fact, the quantization is trivial, while MI-File has to compare each image feature against all the permutants during the indexing phase.

The effectiveness of the search was evaluated comparing the results obtained by the two systems with the ones we obtained sequentially scanning the entire corpus. In particular, we compared with the ground truth obtained with both the binarized features (used for efficiency) and the ReLU L2 Normalized ones that is considered the standard approach for comparing deep features. No users were involved in the evaluation of the results. Thus, while the quality measures we reported are useful to understand the approximation introduced by the indexing methods, they can't be considered the overall quality perceived by the users.

5. CONCLUSIONS

In this paper, we presented the deep features extracted from the YFCC100M images [1] considering the *fc6* hidden layer activation of the HybridNet DCNN [29]. Ground truth k-NN results using the Euclidean distance and a simple but effective binarization approach were also given. Moreover, we evaluated different implementation strategies to index deep features for large-scale CBIR datasets as YFCC100M.

The first approach, MI-File, is a native metric access methods based on permutations, which exhibits excellent performance. The second approach, LuQ, exploits quantization of the sparse deep feature vector for text encoding. Then, an off-the-shelf IR engine (Lucene) is used for indexing. LuQ has the advantage of providing multi-field search that can be used for multimodal retrieval combining image similarity and text (comments, tags, etc.). The two CBIR systems are public available online at [8] and [2]. The experimental results show that MI-File outperforms LuQ. However, the simplicity of the LuQ approach and the fact that it constructs textual representation for the deep features allows the use of standard text search engines and multimodal (text plus images) searching.

As future work, we plan to index also the HNfc7 and VGG features allowing direct comparison.

6. ACKNOWLEDGMENTS

This work was partially founded by: EAGLE, Europeana network of Ancient Greek and Latin Epigraphy, co-founded by the European Commision, CIP-ICT-PSP.2012.2.1 - Europeana and creativity, Grant Agreement n. 325122; and Smart News, Social sensing for breakingnews, co-founded by the Tuscany region under the FAR-FAS 2014 program, CUP CIPE D58C15000270008.

We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Tesla K40 GPU used for this research.

7. REFERENCES

[1] Deep features. http://www.deepfeatures.org. Last accessed: 2016-07-10.

[2] Melisandre. http://melisandre.deepfeatures.org/. Last accessed: 2016-07-10.

[3] The multimedia commons initiative. https://multimediacommons.wordpress.com/.

[4] P. Agrawal, R. Girshick, and J. Malik. Analyzing the performance of multilayer neural networks for object recognition. In *European Conference on Computer Vision*, pages 329–344. Springer, 2014.

[5] G. Amato, F. Debole, F. Falchi, C. Gennaro, and F. Rabitti. Large scale indexing and searching deep convolutional neural network features. In *International Conference on Big Data Analytics and Knowledge Discovery*, pages 213–224. Springer, 2016.

[6] G. Amato, A. Esuli, and F. Falchi. A comparison of pivot selection techniques for permutation-based indexing. *Information Systems*, 52:176–188, 2015.

[7] G. Amato, C. Gennaro, and P. Savino. MI-File: using inverted files for scalable approximate similarity search. *Multimedia Tools and Applications*, 71(3):1333–1362, 2014.

[8] G. Amato and P. Savino. Approximate similarity search in metric spaces using inverted files. In *Proceedings of the 3rd international conference on Scalable information systems*, InfoScale '08, pages 28:1–28:10, ICST, Brussels, Belgium, Belgium, 2008. ICST.

[9] H. Azizpour, A. Razavian, J. Sullivan, A. Maki, and S. Carlsson. From generic to specific deep representations for visual recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 36–45, 2015.

[10] A. Babenko, A. Slesarev, A. Chigorin, and V. Lempitsky. Neural codes for image retrieval. In *Computer Vision–ECCV 2014*, pages 584–599. Springer, 2014.

[11] P. Bolettieri, A. Esuli, F. Falchi, C. Lucchese, R. Perego, T. Piccioli, and F. Rabitti. CoPhIR: a test collection for content-based image retrieval. *CoRR*, abs/0905.4627v2, 2009. http://cophir.isti.cnr.it.

[12] V. Chandrasekhar, J. Lin, O. Morère, H. Goh, and A. Veillard. A practical guide to cnns and fisher vectors for image instance retrieval. *arXiv preprint arXiv:1508.02496*, 2015.

[13] V. R. Chandrasekhar, D. M. Chen, S. S. Tsai, N.-M. Cheung, H. Chen, G. Takacs, Y. Reznik, R. Vedantham, R. Grzeszczuk, J. Bach, et al. The stanford mobile visual search data set. In *Proceedings of the second annual ACM conference on Multimedia systems*, pages 117–122. ACM, 2011.

[14] E. Chávez, G. Navarro, R. Baeza-Yates, and J. L. Marroquín. Searching in metric spaces. *ACM computing surveys (CSUR)*, 33(3):273–321, 2001.

[15] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. *arXiv preprint arXiv:1310.1531*, 2013.

[16] A. Esuli. Pp-index: Using permutation prefixes for efficient and scalable approximate similarity search. *Proceedings of LSDS-IR*, 2009, 2009.

[17] F. Falchi, C. Lucchese, S. Orlando, R. Perego, and F. Rabitti. Similarity caching in large-scale image retrieval. *Information Processing & Management*, 48(5):803–818, 2012.

[18] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

[19] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[20] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[21] J. Ngiam, Z. Chen, S. A. Bhaskar, P. W. Koh, and A. Y. Ng. Sparse filtering. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 1125–1133. Curran Associates, Inc., 2011.

[22] D. Novak, M. Batko, and P. Zezula. Metric index: An efficient and scalable solution for precise and approximate similarity search. *Information Systems*, 36(4):721–733, 2011.

[23] A. Popescu, E. Spyromitros-Xioufis, S. Papadopoulos, H. Le Borgne, and I. Kompatsiaris. Toward an automatic evaluation of retrieval performance with large scale image collections. In *Proceedings of the 2015 Workshop on Community-Organized Multimodal Mining: Opportunities for Novel Solutions*, MMCommons '15, pages 7–12, New York, NY, USA, 2015. ACM.

[24] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. CNN features off-the-shelf: an astounding baseline for recognition. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2014 IEEE Conference on*, pages 512–519. IEEE, 2014.

[25] A. S. Razavian, J. Sullivan, A. Maki, and S. Carlsson. A baseline for visual instance retrieval with deep convolutional networks. *arXiv preprint arXiv:1412.6574*, 2014.

[26] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229*, 2013.

[27] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[28] B. Thomee, B. Elizalde, D. A. Shamma, K. Ni, G. Friedland, D. Poland, D. Borth, and L.-J. Li. Yfcc100m: The new data in multimedia research. *Communications of the ACM*, 59(2):64–73, 2016.

[29] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. Learning deep features for scene recognition using places database. In *Advances in neural information processing systems*, pages 487–495, 2014.