Multi-modal image retrieval with random walk on multi-layer graphs

Renata Khasanova*, Xiaowen Dong† and Pascal Frossard*
*Signal Processing Laboratory (LTS4),
EPFL, Lausanne, Switzerland
Email: renata.khasanova@epfl.ch, pascal.frossard@epfl.ch
†Media Lab, MIT, Cambridge, USA
Email: xdong@mit.edu

***Abstract*—The analysis of large collections of image data is still a challenging problem due to the difficulty of capturing the true concepts in visual data. The similarity between images could be computed using different and possibly multimodal features such as color or edge information or even text labels. This motivates the design of image analysis solutions that are able to effectively integrate the multi-view information provided by different feature sets. We therefore propose an algorithm that is able to sort images through a random walk on a multi-layer graph, where each layer corresponds to a different type of information about the image data. We propose an effective method to select the edge weights for the multi-layer graph, such that the image ranking scores are optimised. Our experiments show that the proposed algorithm surpasses state-of-the-art solutions due to a more meaningful image similarity computation.**

***Keywords*-Image retrieval, multi-modal data analysis, multi-layer graphs.**

I. Introduction

Image collections, stored on the Internet, are rapidly growing every day. In the same time, retrieving relevant information in these huge data collections is an increasingly important challenge. Image retrieval systems can be constructed in different ways, where the user can provide the system with either a query image, text or characteristic for the search. We are interested in the case where the user provides a query image, and the task is to reorder the images in a multi-view dataset according to their relevance or similarity with the query image. We therefore develop a new query-based image retrieval algorithm, which has access to image labels for a part of the dataset.

The image similarity is usually estimated through the comparison of different features that characterize the images. However, extracting features that can effectively describe every image in the database is a challenging problem. This is often referred to as the semantic gap problem, as numerical features that we can extract from images cannot really describe the semantic information of the images. To tackle this, we propose to use multi-modal data gathered from different sources and to combine these multi-view features to compute the similarity between images. For example, textual and visual features can both describe an image and characterize diverse properties that are complementary to each other.

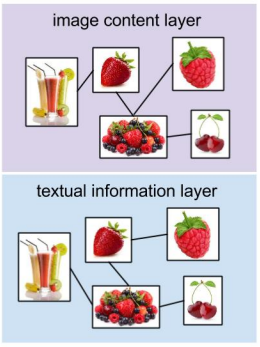

Figure 1. Example of a multi-layer image graph. Each image is a node, and the edges in each layer depend on the similarity between images for a given type of feature.

However, these features can be fundamentally different: some features measured with real numbers (e.g., gradient histogram of an image [1]) others are binary (e.g., indicator that shows that an image contains a particular object [2]). The effective combination of these different features for image retrieval is therefore a challenging problem.

We propose in this paper a new algorithm based on multi-layer graphs to rank images in combining heterogeneous features. First, to work with big data collections, we store relationship pairs between features of all images in a flexible graph structure (see Fig.1) where each type of modality forms a different layer. Using this structure, we preserve important information about each type of information. Second, we develop a random walk process on the graph to retrieve similarity information between images. This permits to cope with possibly incomplete data in some modalities. Our algorithm makes transitions between layers based on the categories' distribution in the neighborhood of the current node. It forms a flexible framework where labeled data can be used to give node-specific weights to different layers. Extensive experiments show that the proposed algorithm outperforms baseline and state-of-the-art methods in multimodal image retrieval.

In summary, our contribution is three-fold:

1) we develop a new image retrieval algorithm that works with possibly incomplete multi-modal features;

2) we propose a generalization of a random walk algorithm to multi-layer graphs;
3) we introduce node-specific weights to effectively combine features on graphs.

The paper is organized as follows. In Section II, we describe the related work. The proposed image retrieval algorithm and an approach to find a transition probability matrix across layers are given in Section III. Experiments are discussed in Section IV and we conclude in Section V.

II. Related Work

An image retrieval algorithm obtains a ranking result using features that can distinguish images from each other. In many cases, the query image is described as a combination of features of different nature, thus, multi-modal fusion of the features is needed to respond to user queries [3]. The algorithms that use multi-modal data, can be divided into early and late fusion algorithms. In early fusion models, multi-modal data is combined at the feature level, while in late fusion they are rather combined at the output level.

A lot of the research work has been dedicated to early fusion methods. In [4], the authors assume that multi-modal data lies on manifolds that are embedded in high-dimentional spaces. They construct multiple graphs using different features. Afterwards, they propose to find a common Laplacian matrix for the graphs. Other researchers use joint matrix factorization to build a unified optimization algorithm [5]. In [6], the authors formulate a nonnegative matrix factorization constraint for clustering tasks that penalizes solutions without consensus between different features. Another method uses canonical correlation analysis [7] and projects the multi-modal data into a low-dimensional space to eventually work with the projected data. In late fusion methods, such as [8] and [9], different ranking results are obtained for different features, and effective rules are implemented to aggregate them. It is however challenging to fuse ranks that are obtained using different features, because top results in different modalities can have only a few common images or even empty intersections.

At the same time, there is nowadays a growing interest in graph-based algorithms in supervised, semi-supervised and unsupervised image clustering, retrieval and classification tasks [10]. Clustering could be efficiently performed on graphs using spectral clustering, for example [11]. Traditionally, image retrieval is based on a search of pairwise distances of all the images, which translates into finding the most similar neighbours on graphs. However, in this case, some important information about the distribution of the features of all images in the dataset can be lost. Context properties or data models can help to preserve this global information. The authors of [12], for example, propose to improve the result of unsupervised image retrieval using a manifold structure. A random walk model is used in [10], which looks for the combination of the initialization of the graph, the type of the transition matrix, and the definition of the diffusion process, which gives the best retrieval result. The method in [13] finally reorders images, using content features the images that are initially ranked based of textual information. The authors propose to learn a graph for every feature individually based on query images. After that, these graphs are combined into a single graph structure and the images are reordered accordingly. Graphs also play an important role in classification. The main assumption in classification tasks is that similar objects tend to belong to the same class. A lot of works based on regularization theory search for the smoothest graph signals [14], [15] for proper classification. For example, the authors in [14] interpret the labels as a signal on graphs and classification is performed by computing a smooth graph signal. However, all these methods are designed for a single data modality; on the other hand, we propose an algorithm that effectively combines data from different sources.

In summary, there are a lot of methods that try to solve the image retrieval problem. However, the challenging semantic gap problem is still not fully solved [3]. To tackle the problem, we propose to use a flexible, sparse multi-layer graph structure (Fig. 1). The graphs capture information about the distribution of the images in the database, and the proper combination of multi-modal data addresses the semantic gap problem. In particular, we develop a new framework that permits to effectively handle data that can be incomplete in some modalities.

III. Random walk for image retrieval

A. Multi-layer graphs

Images can be compared with each other using similarity measures, which can be conveniently represented by sparse graph structures. A graph (V, E, W) is defined by a set of nodes V, edges E and edge weights W. Each node is associated with one image and each edge represents the relationship between two images. The weight of the edge expresses the image similarity that is measured with respect to some particular features.

Data around us can typically be represented by multi-modal information, where different kinds of information complete each other. We therefore propose to use a multi-layer graph to combine data from different sources into one single structure. For example, we can construct a multi-layer graph using textual and image content information. Assume that the system contains images of a raspberry and a berry smoothie (Fig. 1). These images are not connected on textual and content layer because they do not have common tags and look different. However, these images are related to each other. A multi-layer graph structure helps to find this relationship. Let the database contains an image with a basket of berries connected to a raspberry image at the content layer (they can have common local-level features) and to the smoothie image at the textual layer (they have in

common the tag "berry"). In this case, the multi-layer graph is able to establish a relationship between a raspberry and a smoothie, which is reasonable from a semantic viewpoint.

We extract features of a different nature – content features, features based on tags, meta-data features and so on – and use these features to construct different layers. Each layer l of multi-layer graph $(V, E_1, \ldots, E_L, W_1, \ldots, W_L)$ is a single graph (V, E_l, W_l). All layers have the same set of nodes V but with different edges and edge weights (E_l, W_l) in each layer. For example, in Fig. 1, a two-layer graph for an image dataset is constructed. The images are the same for all layers, but the relationships between these nodes represent similarities in terms of different features, namely, textual and visual content of an image.

B. Random walk on a multi-layer graph

The image retrieval problem can be solved via a random walk process on the graph [16]. A random walk consists in a succession of random steps driven by transition probabilities that depend on edge weights. The most visited nodes get high rank values in the retrieval result. If the graph is properly constructed, similar images are connected by strong edges. This increases the probability of the corresponding nodes to be visited by the random walk.

We create a random walk process on a multi-layer graph that is constructed using multimodal information as indicated above. We suppose that similar images are connected to each other in one or more layers. For example, sea and lake images are connected at the texture layer, and images of food and restaurants are connected at the text layer. Then, we start a random walk process on the query image. On every step, the algorithm can walk within a layer of the graph or can make a transition to another layer of the graph based on the relative importance of the layers (Fig. 2). This transition between layers extends the classical random walk process to multi-layer graphs, in order to benefit from the availability of multi-modal information.

More formally, we start the algorithm with a vector of ranking values for all the images nodes $r^{(0)} = \pi$, where $\pi = \{\pi_1, \pi_2, \ldots, \pi_M\}$ is a fixed distribution with $\pi_q = 1$ for the query node and 0 for other nodes and where M is the number of images in the dataset. We then perform a random walk on the multi-layer graph. At each time step, we consider two sub-steps. On the first sub-step, we choose the layer l for the node i to perform the random walk with the probability α_{li}. Afterwards, we choose the neighbor node j to perform a random walk step according to the transition probabilities in layer l. Accordingly, we iteratively update the rank vector till convergence in the following way:

$$r^{(t)} = (1-\eta)\pi + \eta(P_1^T\Lambda_1 + P_2^T\Lambda_2 + \cdots + P_L^T\Lambda_L)r^{(t-1)}, \quad (1)$$

where $r^{(t)}$ is a ranking value on iteration t, $1-\eta$ is the probability of jumping back to the query vertex, P_l^T is a transition matrix for layer l and Λ_l is a node-specific matrix that represents the probability to choose the layer l in the random walk. The transition probability between node i and node j for layer l is defined as

$$p_{i,j}(l) = \frac{w_{i,j}(l)}{\sum_{j \in N_i(l)} w_{i,j}(l)}, \quad (2)$$

where $w_{i,j}(l)$ is the weight of the edge between nodes i and j for layer l, $N_i(l)$ is a set of the vertices that are the neighbors of vertex i in layer l. The weights $w_{i,j}(l)$ simply represent the similarity between images i and j based on the features considered in layer l. These transitional probabilities $p_{ij}(l)$ that form the matrix P_l^T can be calculated in a similar way for all the layers.

Then, the layer transition probability matrices Λ_l for every layer l are diagonal matrices $\Lambda_l = diag(\alpha_{l1}, \ldots, \alpha_{lM})$, where α_{li} is a node-specific probability for jumping to layer l at node i. Note that the sum of each row across different layer's matrices is equal to 1:

$$\sum_{1}^{L} \alpha_{li} = 1. \quad (3)$$

We propose a new method to compute these matrices in the next section.

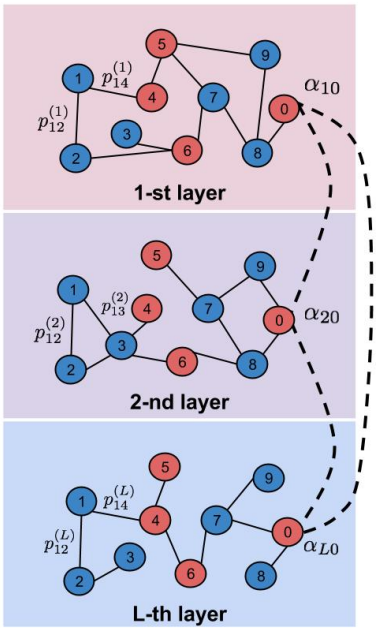

Figure 2. Example of L-layer graph structure with labeled and unlabeled nodes in red and blue, respectively. On node 0, we show the transition probability α_{l0} to choose layer l to continue the random walk, and, on node 1, we show the transition probability $p_{1j}(l)$ to make a random walk step toward the neighbor node j on layer l (best seen in color).

C. Layer transition probabilities

Choosing a layer for the random walk in a multi-layer graph is a process that has not been studied well. To the best of our knowledge, a method that uses node-specific probabilities to combine layers does not exist yet.

We thus propose a method for computing the layer transition probability Λ_l. First, we observe that the layer transition probabilities could be different for different query images. Thus, we suggest learning node-specific transition probabilities Λ_l with respect to a query image.

To calculate these probabilities, we assume that we know part of the labels in the image dataset. This means that we know the categories, which some of the images belong to. Then, the idea is to favor a walk in the layer where most of the neighbors of the current node belong to the same category. For this purpose, we consider the labeled nodes in the neighborhood of a particular node i within a radius d_l. The neighborhood of the node i is a set of nodes, which are strongly connected to i. Nodes i and j are strongly connected if the weights $D_l(i,j) \geq d_l$, where $D_l(i,j)$ is defined as:

$$D_l(i,j)_{i,i_1,i_2\ldots,j} = w_{i,i_1}(l)w_{i_1,i_2}(l)\ldots w_{i_J,j}(l), \quad (4)$$

with $\{i, i_1, i_2, \ldots, i_J, j\}$ a path in a graph's layer. If there are several paths $k \in K$ between nodes i and j we choose the path that gives maximum weight value:

$$D_l(i,j) = \max_{k\in K}(D_l(i,j)_k). \quad (5)$$

For example, consider that a graph has strong edges $e(i,i_1), e(i_1,i_2)$ and a weak edge $e(i_2,i_3)$. Then for the node i the algorithm includes the nodes i_1 and i_2 in a neighborhood set, and stops to look for deeper neighbors for the node i. Thus, the neighborhood of each node contains only relevant neighbors for this node.

To calculate the layer transition probabilities, we then compute the number of categories in this neighborhood and the maximum fraction of a certain category:

$$n(l,i) = \max_{c\in C}\frac{\#v_{li}(c)}{\#v_{li}}, \quad (6)$$

where C is a set of categories, $\#v_{li}(c)$ is the total number of labeled neighbors for the node i in the layer l that belongs to the category c, and $\#v_{li}$ is the number of labeled neighbors.

After that, since we want to find the images that are similar to the query node we prefer walking in a layer that is important for the query node with more probability than for the other layers. Therefore, the label distribution around the query image q should also affect the transition probabilities. Thus, we combine probabilities $n(l,i)$ and $n(l,q)$ and normalize the result so that the transition probabilities for a given vertex sum up to one. We finally walk with the node-specific probability α:

$$\alpha_{li} = \frac{z(l,i)z(l,q)}{\sum_l \big(z(l,q)z(l,i)\big)}, \quad (7)$$

where z is a sigmoid function:

$$z(l,i) = \frac{1}{1+e^{-a(n(l,i)-n^*)}}. \quad (8)$$

The sigmoid function gives higher priority to the probabilities that are larger than a threshold n^*, and lower priority to others. The coefficient a permits to change the sigmoid function's slope. The probability in Eq. (7) is influenced by both the neighborhood of the node i and the query q, through $z(l,i)$ and $z(l,q)$ respectively.

To sum up, we propose to calculate the layer transition probability for each node based on labeled nodes in its neighborhood. The neighborhood contains only nodes that have strong connections with each other and can vary from layer to layer. The algorithm gives more priority to a layer which contains many nodes from the same category, because it is an indicator that the features in the layer properly represent this category.

IV. Experiments

We now develop extensive experiments to evaluate the performance of the proposed algorithm. First of all, we present the evaluation metrics, the details of the graph construction and the datasets under consideration. Then we provide comparative results with state-of-the-art retrieval algorithms.

A. Experimental settings

1) Evaluation metrics: The objective of our retrieval algorithm is to obtain a ranking of images, where all images in the first positions should have a similar category as the query node. Assume that we know the ground truth image categories for all our dataset. We can thus measure the quality of our result using the mean Average Precision function (mAP) that estimates the quality of ranking for different queries. The function calculates the average precision (APr) for all queries from a dataset. More formally, let M denote the number of images that are relevant to the query image in a database of N images. Let $I(k)$ be an indicator function, which is equal to 1 if the item at position k in the image ranking is a relevant image, and zero otherwise. Let further $Pr(k)$ be the precision of the top k-rank values. We can then define APr and mAP as:

$$APr(q) = \frac{\sum_{k=1}^{N} Pr(k)I(k)}{M}, mAP = \frac{\sum_{q\in N} APr(q)}{|N|}. \quad (9)$$

However, when only a few examples are available (e.g., in the dataset [17]) the N-S score is used. It represents the average number of correct images among top M retrieved images, where M is the size of the ground truth data set.

2) Graph construction: For all our experiments we construct undirected graphs where the edge weights are computed using Gaussian kernels to emphasize larger similarity values. In each layer, we define the edge weight between the corresponding feature vector x_i of image i and the respective feature vector x_j of image j as

$$w_{i,j} = \exp\left(\frac{-||x_i - x_j||^2}{\sigma^2}\right), \tag{10}$$

where σ is used to adjust the degree of similarity between nodes. In order to construct sparse layers, we connect a node only with its $k = 5$ nearest neighbors (in terms of Euclidean distance). We run a 5-Fold cross validation to select the parameters of our method, and as a result we set the parameters of the sigmoid function to be $a = 10$ and $n^* = 0.5$, radius parameter $\beta = 0.5$, return probability $\eta = 0.9$ in our further experiments.

3) Datasets: To evaluate the effectiveness of our method we consider Holidays [18] and Ukbench [17] datasets. Each of them has a ground-truth annotation.

The Holidays dataset contains 1491 images where 500 of them are query images. For every query image there is a groundtruth list of corresponding relevant images. Then, the Ukbench dataset is released with 10200 images that are grouped into 2550 clusters. For every image, three corresponding images are known and form the groundtruth information.

For the Holidays and Ukbench datasets we use HSV Histogram, Convolutional Neural Network, GIST and Random Projection features from [19]. Each of these features is used to construct a layer in our multi-view graph.

B. Comparison with other image retrieval methods

In this section, we compare our method and state-of-the-art algorithms namely [19], which tackles multi-modal image retrieval problems using a late fusion strategy, and [20], which uses an early fusion strategy on the Holiday and Ukbench datasets. In [20], the authors aggregate the layers of a multi-layer graph into one layer. Notice that our results can be slightly different from the ones actually reported in [19], [20], since we do not use the exact same set of features as these papers due to the high extraction complexity. In our experiments, we however use the same set of features for all algorithms under comparison.

We also compare our algorithm with the following baseline algorithms:

- **Baseline 1.** Random walk with the equal transition probabilities $\alpha_l = \frac{1}{L}$ for all graph's nodes and graph layers, where L is a number of layers.
- **Baseline 2.** To justify the node specific probabilities we compare our method with a random walk with equal transition probabilities α_{lq} for all graph's nodes but the choice of this probability is individual for every query and layer. These probabilities are calculated in the same way according to Eq. (7) but only for the query node.
- **Baseline 3.** We combine all features into one single vector and sort images according to their distances to the query image.

We evaluate the performance in a similar way as in [19]: for the Holiday dataset we use the mean Average Precision (mAP) value. For the Ukbench dataset we use the N-S score because it contains only 4 correct images for each query. Table I shows the results of our experiments. Our method outperforms all baseline algorithms and the algorithm in [20]. The result of the proposed method is further comparable with [19]. However, we use only image features from the datasets to run our algorithm, unlike the state-of-the-art method [19], which uses a large Flickr dataset to train the feature distributions. Also, it is worth noting that both datasets under consideration contain only a few ground truth examples for every query. It gives further advantages to the algorithm in [19], which calculates feature weights to get a final result based on information about a specific query image.

For the Holiday dataset, our method outperforms Baseline 1, Baseline 3 and [20]. It shows that we can achieve improvement using labeled data. Also, our method gives better result than Baseline 2, which has access to labeled data. It shows that the combination of different layers using the neighborhood of every node is more effective than using information about the query node alone. The algorithm in [19], which is an unsupervised method, works slightly better, however they use information about the feature distribution from a large Flickr collection. The algorithm in [19] combines features with the same weight for all nodes in one layer. This strategy is similar to the Baseline 2.

For the Ukbench dataset we use $k = 3$ to construct our kNN graph, because we know that every image in the dataset has only four corresponding images. Our method outperforms the baseline and the state-of-the-art methods. Baseline 1 and Baseline 2 have the same N-S score, which is better than Baseline 3. It happens because the actual distribution of the features is important for this dataset. The graph methods give an opportunity to capture this distribution.

In summary Table I shows that our method and [19] achieve top results and outperform [20] on Holiday dataset. For Ukbench dataset our method produces the best result with respect to the state-of-the-art methods. The results further show the influence of the neighbors nodes to the query image in choosing the right transition probabilities or equivalently in properly modeling the feature distribution. A more extensive experimental evaluation along with retrieval examples can be found in [21], where we show that our method achieves a better ranking than competitors [22].

Retrieval algorithm	Holidays,mAP%	Ukbench,N-S score
Baseline 1	0.7151	3.5835
Baseline 2	0.7214	3.5835
Baseline 3	0.7193	3.5485
[19]	**0.7580**	3.5864
[20]	0.6593	3.2625
Proposed method	0.7435	**3.5899**

Table I
COMPARISON OF OUR AND STATE-OF-THE-ART ALGORITHMS ON THE HOLIDAY AND THE UKBENCH DATASETS.

V. CONCLUSION

This work is dedicated to the timely but challenging problem of image retrieval. It tries to mitigate the issues induced by the so-called semantic gap by properly combining multimodal features for image ranking. Currently, researchers use very complicated techniques to solve this problem in image retrieval. We rather show in this paper that combining features of different modalities in a proper way with a multi-layer graph permits to achieve effective retrieval with a simple random walk algorithm. In particular, the proposed solution achieves good image retrieval performance compared to the state-of-the-art methods.

We firmly believe that flexible structures like graphs offer promising solutions to capture the underlying geometry of multi-view data. This is confirmed by the performance of our algorithm.

REFERENCES

[1] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Conference on Computer Vision and Pattern Recognition*, 2005, pp. 886–893.

[2] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "Brief: Binary robust independent elementary features," in *European Conference on Computer Vision*, 2010, pp. 778–792.

[3] R. Datta, D. Joshi, J. Li, and J. Wang, "Image retrieval: ideas, influences, and trends of the new age," *ACM Computing Surveys*, 2008.

[4] D. Eynard, K. Glashoff, M. Bronstein, and A. Bronstein, "Multimodal diffusion geometry by joint diagonalization of laplacians," *arXiv Preprint*, 2012.

[5] X. Dong, P. Frossard, P. Vandergheynst, and N. Nefedov, "Clustering with multi-layer graphs: A spectral perspective," *IEEE Transactions on Signal Processing*, vol. 60, no. 11, pp. 5820–5831, 2012.

[6] J. Liu, C. Wang, J. Gao, and J. Han, "Multi-view clustering via joint nonnegative matrix factorization," in *SIAM Data Mining Conference*, 2013, pp. 252–260.

[7] K. Chaudhuri, S. Kakade, K. Livescu, and K. Sridharan, "Multi-view clustering via canonical correlation analysis," in *International Conference on Machine Learning*. ACM, 2009, pp. 129–136.

[8] C. Dwork, R. Kumar, M. Naor, and D. Sivakumar, "Rank aggregation methods for the web," in *International Conference on World Wide Web*. ACM, 2001, pp. 613–622.

[9] R. Fagin, R. Kumar, and D. Sivakumar, "Efficient similarity search and classification via rank aggregation," in *ACM SIGMOD International Conference on Management of Data*, 2003, pp. 301–312.

[10] M. Donoser and H. Bischof, "Diffusion processes for retrieval revisited," in *Conference on Computer Vision and Pattern Recognition*, 2013, pp. 1320–1327.

[11] U. V. Luxburg, "A tutorial on spectral clustering," *Statistics and computing*, vol. 17, no. 4, pp. 395–416, 2007.

[12] D. Zhou, J. Weston, A. Gretton, O. Bousquet, and B. Scholkopf, "Ranking on data manifolds," in *Advances in Neural Information Processing Systems*, 2004.

[13] C. Deng, R. Ji, W. Liu, D. Tao, and X. Gao, "Visual reranking through weakly supervised multi-graph learning," in *International Conference on Computer Vision*, 2013, pp. 2600–2607.

[14] A. Sandryhaila and J. Moura, "Classification via regularization on graphs," in *IEEE Global Conference on Signal and Information Processing*, 2013, pp. 495–498.

[15] M. Long, J. Wang, G. Ding, D. Shen, and Q. Yang, "Transfer learning with graph co-regularization," *Knowledge and Data Engineering*, vol. 26, no. 7, pp. 1805–1818, 2014.

[16] L. Lovász, "Random walks on graphs," *Combinatorics, Paul erdos is eighty*, vol. 2, pp. 1–46, 1993.

[17] D. Nister and H. Stewenius, "Scalable recognition with a vocabulary tree," in *Conference on Computer Vision and Pattern Recognition*, vol. 2. IEEE, 2006, pp. 2161–2168.

[18] H. Jégou, M. Douze, and C. Schmid, "Hamming embedding and weak geometry consistency for large scale image search-extended version," in *European Conference on Computer Vision*, 2008, pp. 304–317.

[19] L. Zheng, S. Wang, L. Tian, F. He, Z. Liu, and Q. Tian, "Query-adaptive late fusion for image search and person re-identification," in *Conference on Computer Vision and Pattern Recognition*, vol. 1, 2015, pp. 1741–1750.

[20] S. Zhang, M. Yang, T. Cour, K. Yu, and D. Metaxas, "Query specific fusion for image retrieval." in *European Conference on Computer Vision*, vol. 7573, 2012, pp. 660–673.

[21] R. Khasanova, X. Dong, and P. Frossard, "Multi-modal image retrieval with random walk on multi-layer graphs," *arXiv Preprint*, vol. abs/1607.03406, 2016.

[22] M. Wang, H. Li, D. Tao, K. Lu, and X. Wu, "Multimodal graph-based reranking for web image search," *IEEE Transactions on Image Processing*, vol. 21, no. 11, pp. 4649–4661, 2012.