2016

MultiVCRank with applications to image retrieval

Xutao Li
Harbin Institute of Technology

Yunming Ye
Harbin Institute of Technology

Michael K. Ng
Hong Kong Baptist University

This document is the authors' final version of the published article.
Link to published article: https://dx.doi.org/10.1109/TIP.2016.2522298

Recommended Citation
MultiVCRank with Applications to Image Retrieval

Xutao Li  
Yunming Ye  
Michael K. Ng

Abstract—In this paper, we propose and develop a multi-visual-concept ranking (MultiVCRank) scheme for image retrieval. The key idea is that an image can be represented by several visual concepts, and a hypergraph is built based on visual concepts as hyperedges where each edge contains images as vertices to share a specific visual concept. In the constructed hypergraph, the weight between two vertices in a hyperedge is incorporated, and it can be measured by their affinity in the corresponding visual concept. A ranking scheme is designed to compute the association scores of images and the relevance scores of visual concepts by employing input query vectors to handle image retrieval. In the scheme, the association and relevance scores are determined by an iterative method to solve limiting probabilities of a multi-dimensional Markov chain arising from the constructed hypergraph. The convergence analysis of the iterative method is studied and analyzed. Moreover, a learning algorithm is also proposed to set the parameters in the scheme, which makes it simple to use. Experimental results on the MSRC, Corel and Caltech256 datasets have demonstrated the effectiveness of the proposed method. In the comparison, we find that retrieval performance of MultiVCRank is substantially better than those of HypergraphRank, ManifoldRank, TPHITS and RankSVM.

I. INTRODUCTION

Content-based image retrieval (CBIR) is a challenging problem in image database management systems [28], [7], [34], [2], which exploits the visual information instead of keywords to search relevant images. Based on the level of visual information used, CBIR can be generally categorized into two types [40]: low-level feature based search and region based search. Low-level feature based search makes use of color layout or histogram from pixel level to represent images so as to accomplish the retrieval task. Due to the “semantic gap” between these features to human perception system, the performance of such methods is still far from users’ expectations [14]. Although recent studies on deep convolution neural network may shed light on bridging this gap [24], [36], [10], training such a deep learning model is usually very time-consuming. Different from low-level feature based search, region based search extracts features from region level for an image, where the regions correspond to “objects” or “parts of objects” in this image. In such case, each image is naturally considered as a combination of multiple visual concepts, and thus the images are finally ranked based on their similarities to query image computed from this perspective. In the literature, many region based search methods have been proposed, see [4], [18], [27] for instance. A detailed review can be found in [14].

Although region-based retrievals may alleviate the problem of “semantic gap”, their performance is highly dependent on the quality of regions. In real applications, the image database is usually very large and contains a huge number of images. It is infeasible to obtain the image regions of perfect quality by human’s labor. Usually, they need to be automatically identified by algorithms, e.g., image segmentation [9], [33] or salient object detection algorithms [17], [13], [5] etc. However, the regions output by these algorithms may contain useful visual information as well as noisy information, which will hurt the performance of retrieval methods.

Inspired by link analysis, which aims at addressing the noisy information for webpage retrieval, we propose in this paper, MultiVCRank, a graph-based ranking scheme that is able to exploit unclean image regions generated by salient object detection or segmentation algorithms for CBIR. In this scheme, we first cluster the regions to form a set of visual concepts, and then build a hypergraph to model the affinity between images, where each image is a vertex and each concept is a hyperedge. In this hypergraph, the weight between two vertices in a hyperedge is incorporated, and it indicates their affinity in the corresponding visual concept. Following link analysis, we consider the association scores of images and the relevance scores of visual concepts to denote their importance with respect to a particular query image. Our objective is thus to design a ranking scheme (MultiVCRank) to compute these scores to handle image retrieval.

According to the constructed hypergraph, we derive a multi-dimensional Markov chain [30], [25] to determine the association scores of images and relevance scores of visual concepts. A multivariate polynomial system is required to solve for such association scores and relevance scores. An iterative method is developed to solve the multivariate polynomial system, and the analysis of the iterative method is shown to guarantee the convergence of the proposed method, and the effectiveness of the method. In addition, a learning algorithm is also proposed for setting the parameter of this method. We conduct experiments on MSRC, Corel and Caltech256 image datasets. The experimental
results will show the effectiveness of the proposed MultiVCRank scheme, and it outperforms the other retrieval methods such as MainfoldRank [11], HypergraphRank [16], TOPHITS [23] and RankSVM [20].

The rest of this paper is organized as follows. In Section II, we review related work. In Section III, we present the proposed MultiVCRank scheme and the analysis of the proposed solver. Experimental results are presented in Section IV. Finally, the concluding remarks are given in Section V.

II. RELATED WORK

In this section, we first give a review on graph based-ranking methods, from Section II-A to Section II-D, and then briefly discuss some recent progress of image processing with deep learning in Section II-E.

A. PageRank

PageRank is a well-known graph-based ranking algorithm for evaluating the importance of webpages. In [19], Jing et al. applied PageRank for large-scale web image search. In their method, the PageRank is performed on a graph computed from the visual information of image lists that are obtained by keywords query to refine the ranking result. Ambai et al. [1] extended and improved this method by considering image lists where multi-class images are included. In their method, images obtained by keywords query are first divided into several categories based on their visual features, and then PageRank is applied to give rankings within each category.

B. ManifoldRank

In [11], a transductive learning framework, ManifoldRank, based on pairwise-image similarity graph was proposed to rank images in terms of query example. This method calculates a weighted graph according to a similarity metric. The weight between the two nodes indicates the similarity between the corresponding two images under the metric. With an initial labeling score vector constructed from query images, it computes the relevance scores of other images by transductive propagations along the graph structure. Since it explores the relationship of all images, this method usually provides better relevance scores compared to traditional methods. The generalization of this method is also given in [12].

C. HypergraphRank

Recently, Huang et al. proposed, HypergraphRank, a probabilistic hypergraph model [16]. In this model, a hypergraph is constructed by considering each image as a “centroid” and connecting its k-nearest neighbors with a probabilistic hyperedge. With a database composed of m images, a hypergraph is built with m node and m hyperedges, where the i-th hyperedge indicates the similarities of images from the grouping view of i-th image. Hence this method indeed exploits multiple similarity metrics based on the construction of image groups. Similar to ManifoldRank, HypergraphRank method computes the relevance scores of images via transductive propagations in terms of the whole hypergraph as well.

D. MultiRank and HAR

In [30], MultiRank was proposed to develop a co-ranking scheme for objects and relations in multi-relational data. The main idea is to determine the importance of both objects and relations simultaneously based on a probability distribution computed from multi-relational data. Extensive experiments on real-world data have shown that MultiRank is able to provide a co-ranking scheme for objects and relations successfully. In [25], Li et al. considered a framework HAR to study the hub and authority scores of objects, and the relevance scores of relations in multi-relational data for query search. Their basic idea of the framework is to consider a random walk in multi-relational data, and study in such random walk, limiting probabilities of relations for relevance scores, and of objects for hub scores and authority scores. These scores can be used to obtain relevant searching results.

In this paper, we make use of MultiRank and HAR to design a ranking scheme (MultiVCRank) to compute the association scores of images and the relevance scores of visual concepts to handle image retrieval in the constructed hypergraph. Although both ManifoldRank and HypergraphRank incorporate information of visual concepts in the graph construction, they do not distinguish the usefulness of visual concepts in process of image retrieval. For instance, HypergraphRank just uses the simple average of the similarities from different visual concepts together to check the difference between two images. The proposed ranking scheme is more distinguished and effective than ManifoldRank and HypergraphRank in image retrieval as the association scores of images and relevance scores of visual concepts are mutually-reinforcing in our ranking scheme, which are computed dynamically based on the our hypergraph. In Section IV, experimental results will be presented to demonstrate the effectiveness of the proposed ranking scheme.

E. Deep Learning

Recently, deep learning techniques have shed some light on alleviating the “semantic gap” between human perception systems and image representation. Due to its powerful representation ability, the techniques have been widely applied to various applications, for example, image classification [24], [36], [6], scene labeling [8], image similarity calculation [41], and image semantic hashing [43], [45], etc. In [24], Krizhevsky et al. proposed to classify images with a deep convolutional neural network (CNN), and obtained considerably better results than previous methods. In [8], Farabet et al. trained a multiscale CNN to extract features for scene labeling problem. Wang et al. developed a CNN model to learn proximity metric for similarities among images [41]. By utilizing the idea of ranking, Zhao el al. [45] recently learned a semantic hash function for multi-label images based on deep learning model. Although this method exploits the idea of multi visual concept ranking, image labels must be used and leveraged as visual concepts (supervised information) to design a ranking objective function. In contrast, the proposed method does not have
aims to generate a ranking for the number of features to be used. Given a query image $Q$, suppose that we have a region-based feature representation of an image or a region, where $d$ amount of computations. It is very time consuming and it is very hard to develop ranking methods on this hypergraph to search image similarity in region level with a hypergraph, and respect to the input query.

Our main idea of solving this problem is to model the training process requires a huge amount of information and effectively remove noisy information from the salient detection step.

On the other hand, the main issue of deep learning techniques is that the training process requires a huge amount of computations. It is very time consuming and GPUs are needed to accelerate the training process.

III. THE MULTIVCRANK SCHEME

A. Problem Definition

Assume we have an image database composed of $m$ images $I = \{I_1, I_2, \cdots, I_m\}$. After salient object detections or segmentations for these images, suppose we obtain $p$ regions $S = \{S_1, S_2, \cdots, S_p\}$. In this paper, we generate the salient regions with the algorithm in [5], due to both its efficiency and good performance. Let a $p$-by-$m$ matrix $M$ represent the membership between the regions and the images, where $[M]_{i,j} = 1$ if the region $S_i$ is from the image $I_j$, otherwise $[M]_{i,j} = 0$. Let $f(\cdot) \in \mathbb{R}^d$ indicate the feature representation of an image or a region, where $d$ is the number of features to be used. Given a query image $Q$ with $q$ salient regions $\{R_1, R_2, ..., R_q\}$, the MultiVCRank aims to generate a ranking for $I = \{I_1, I_2, \cdots, I_m\}$ with respect to the input query.

Our main idea of solving this problem is to model the image similarity in region level with a hypergraph, and then develop ranking methods on this hypergraph to search and retrieve images. The idea can be illustrated with the example in Figure 1. Suppose that we have a region-based image collection containing six images (a), (b), (c), (d), (e), (f) in Figure 1. There are eighteen regions $S_1, S_2, \cdots, S_{18}$. We consider these regions into a feature space and cluster them into seven visual concepts, namely, \{S_2, S_5, S_8\}, \{S_1, S_4, S_14\}, \{S_6, S_9\}, \{S_5, S_7\}, \{S_{13}, S_{18}\}, \{S_{11}, S_{12}, S_{15}, S_{17}\}, \{S_{10}, S_{16}\}, see Step (1) in Figure 1. Indeed, these visual concepts associate with seven objects “tiger”, “grass”, “river”, “ground”, “sky”, “lion” and “hay” in the six images. Now a hypergraph is built to model the relations among images based on these seven visual concepts, see Step (2) in Figure 1. For example, three images (a), (b) and (c) as vertices belong to the same hyperedge as they share the same visual concept “tiger”. Also the images (a), (b) and (e) belong to the same hyperedge of “grass”. The affinity between the two vertices in a hyperedge is measured by the similarity of their regions corresponding to the visual concept. In the hyperedge of “tiger”, the affinities between images (a) and (b), images (a) and (c) and images (b) and (c) are 0.7, 0.9 and 0.6 respectively. In the hyperedge of “grass”, the affinities between images (a) and (b), images (a) and (e) and images (b) and (e) are 0.3, 0.4 and 0.3 respectively. Given a query image, we calculate the scores of images based on the constructed hypergraph, and then output their rankings, see Steps (3) and (4) in Figure 1.

B. Construction of Hypergraph

Hypergraph refers to a generalization of graph where each edge can be connected to any number of nodes. It gains popularity because of the flexibility to model complex interactions between entities in real world. The usefulness of hypergraph has been shown in many applications, for example, music recommendation [3], image and video retrieval [26], [37], image categorization [15] etc. In this paper, we construct a hypergraph to model the region level similarity between images for retrieval purpose.

The first step of the construction of a hypergraph $G = (\mathcal{I}, \mathcal{E})$ is to construct visual concepts or set up hyperedges $\mathcal{E}$. We cluster $p$ regions $\{S_1, S_2, \cdots, S_p\}$ into $n$ clusters based on their features. Each cluster refers to a visual concept. For simplicity, we employ a $p$-by-$n$ matrix $C$ to indicate the clustering result: $[C]_{i,j} = 1$ if the region $S_i$ belongs to the $j$-th cluster, otherwise $[C]_{i,j} = 0$. For each hyperedge (visual concept or cluster) $E \in \mathcal{E}$, we have a set of vertices (i.e., a non-empty subset of $\mathcal{I}$) belonging to it. Note that we have $\cup_{E \in \mathcal{E}} = \mathcal{I}$.

One of the main difference between our method and the other hypergraph methods for image retrieval is that our constructed hypergraph contains more than binary information, i.e., we do not just indicate whether an image contains a visual concept (or belongs to a hyperedge). We incorporate affinities among images within a visual concept. Here we measure similarity between two vertices (images) in an hyperedge (a visual concept). In particular, we construct a tensor $T = [t_{i,j,k}]$ of size $m$-by-$m$-by-$n$ to represent the weight between two vertices in a hyperedge:

$$t_{i,j,k} = \begin{cases} \exp \left( -\frac{\|f(S_i) - f(S_j)\|^2}{2\sigma^2} \right), & I_i, I_j \in E_k \\ 0, & \text{otherwise} \end{cases} \quad (1)$$
where
\[ g(i, k) = \frac{\sum_{f=1}^{n} f(S_i)[M_{i, c_i}C_{c_i, k}]}{\sum_{f=1}^{n} [M_{i, c_i}C_{c_i, k}]}, \]
and \( g(i, k) \) is indeed the average feature representation of the \( k \)-th visual concept in an image \( i \). Here we employ the intrinsic manifold structure of data (Gaussian kernel function used to yield the non-linear version of affinity) [32] in Eq. (1) and \( \sigma \) is a positive number to control the linkage in the manifold. We remark that \( T \) is a non-negative tensor as each entry is non-negative.

C. Calculation of Association and Relevance Scores

Similar to PageRank and HITS algorithms [31], [22] for determining the importance of vertices in a graph, we would like to rank images according to the constructed hypergraph with Eq. (1). In particular, we calculate the association scores of images and relevance scores of visual concepts for a given query search. These scores have the association scores of images and relevance scores of visual concepts for a given query search. These scores have the following mutually-reinforcing relationships. An image that connects to images with high association scores through visual concepts of high relevance scores, receives a high association score. A visual concept that is connected in between images with high association scores, receives a high relevance score. By considering a random walk in the hypergraph \( G \) or in the tensor \( T \), and studying the limiting probabilities of visiting images and visual concepts, a multi-dimensional Markov chain [30], [25] is derived. The multi-dimensional Markov chain is based on three transition probability tensors \( T^{(1)} \), \( T^{(2)} \) and \( T^{(3)} \) arising from \( T \):

\[
\begin{align*}
T^{(1)}_{i,j,k} &= t^{(1)}_{i,j,k} = \frac{t_{i,j,k}}{\sum_{j=1}^{n} t_{i,j,k}}, \quad i = 1, 2, \cdots, m, \\
T^{(2)}_{i,j,k} &= t^{(2)}_{i,j,k} = \frac{t_{i,j,k}}{\sum_{j=1}^{n} t_{i,j,k}}, \quad j = 1, 2, \cdots, m, \\
T^{(3)}_{i,j,k} &= t^{(3)}_{i,j,k} = \frac{n_{i,k}}{n_{i,k}}, \quad k = 1, 2, \cdots, n.
\end{align*}
\]

We note that if \( \sum_{j=1}^{n} t_{i,j,k} = 0 \), this is called the dangling node [31], and the values of \( t^{(1)}_{i,j,k} \) can be set to \( 1/m \) for \( 1 \leq j \leq m \). The dangling nodes for \( T^{(2)} \) and \( T^{(3)} \) can be handled similarly. The method of construction is similar to that given in PageRank algorithm.

We incorporate input image query vector \( o \) and visual concept query vector \( r \) to handle query-specific search [30], [25]. In particular, when we consider the setting similar to the PageRank algorithm, we compute \( x \) (association scores) and \( z \) (relevance scores) that satisfy the following tensor equations:

\[
x = (1 - \alpha )T^{(1)}xz + \alpha o, \\
z = (1 - \beta )T^{(3)}xx + \beta r,
\]

and \( 0 < \alpha , \gamma < 1 \). Both \( \alpha \) and \( \gamma \) are used to control the restart probabilities in the random walk process of the multi-dimensional Markov chain.

When we consider the setting similar to the HITS algorithm, we compute \( x \) (association authority scores), \( y \) (association hub scores) and \( z \) (relevance scores) that satisfy the following tensor equations:

\[
x = (1 - \alpha )T^{(1)}yz + \alpha o, \\
y = (1 - \beta )T^{(2)}xz + \beta o, \\
z = (1 - \gamma )T^{(3)}xy + \gamma r,
\]

where

\[
\begin{align*}
T^{(1)}yz_{i} &= \sum_{j=1}^{n} \sum_{k=1}^{m} t^{(1)}_{i,j,k} x_{j} z_{k}, \quad 1 \leq i \leq m, \\
T^{(2)}xz_{j} &= \sum_{i=1}^{m} \sum_{k=1}^{m} t^{(2)}_{i,j,k} x_{i} z_{k}, \quad 1 \leq j \leq m, \\
T^{(3)}xy_{k} &= \sum_{i=1}^{m} \sum_{j=1}^{m} t^{(3)}_{i,j,k} x_{j} y_{j}, \quad 1 \leq k \leq n.
\end{align*}
\]

Similarly, \( \alpha , \beta \) and \( \gamma \) are used to control the restart probabilities in the random walk process of the multi-dimensional Markov chain, and their values should be in between 0 and 1.

The next task is to construct the “prior” distributions \( o \) and \( r \) for the input query vectors. Given a query image \( Q \) with \( q \) regions \( \{ R_1, R_2, \ldots, R_q \} \), we set the “prior” probabilities of images by measuring their normalized similarities to the query image as follows:

\[
o_j = \frac{\exp \left( -\frac{||f(Q) - f(Lj)||^2}{2\sigma^2} \right)}{\sum_{i=1}^{m} \exp \left( -\frac{||f(Q) - f(Li)||^2}{2\sigma^2} \right)}
\]

We note that \( \sum_{j=1}^{q} o_j = 1 \) and therefore \( o \) is a probability distribution vector.

To assign the “prior” probabilities for visual concepts, we calculate the similarity of all test regions \( \{ R_1, R_2, \ldots, R_q \} \) to a particular visual concept, and then aggregate them as the “prior” importance of this visual concept. Specifically, we set \( r_\ell \) as follows:

\[
r_\ell = \sum_{j=1}^{q} \exp \left( -\frac{||f(R_j) - f(\ell)||^2}{2\sigma^2} \right)
\]

where

\[
g(\ell) = \frac{\sum_{i=1}^{m} f(S_i)[C]_{i, \ell}}{\sum_{i=1}^{m} [C]_{i, \ell}}
\]

Here \( g(\ell) \) is the average feature representation of the \( \ell \)-th cluster (visual concept). Finally, we normalize the vector \( r \) to be a probability distribution vector. It can be seen that we use the global low-level features of images to compute the “prior” probability distribution vector \( o \), while use the region-based features to compute the “prior” probability distribution vector \( r \).

**Theorem 1:** If the non-negative tensor \( T \) is irreducible, then positive solutions of Eqs. (2) and (3) exist. When
1/2 < α, β < 1, the solution of Eq. (2) is unique. Also when 1/2 < α, β, γ < 1, the solution of Eq. (3) is unique. The proof can be found in Appendix A.

Algorithm 1 MultiVCRankI (or MultiVCRankII) Method

**Input:** a region-based image database I, parameters α and β (or α, β and γ), and a query image Q.

**Output:** ranking result with respect to query image Q.

**Procedure:**

1: Build a hypergraph G with I and the tensor T;
2: Compute T(1) and T(3) (or T(1), T(2) and T(3));
3: Compute α and r as Eqs. (4) and (5), respectively;
4: Randomly initialize x0 and z0 (or x0, y0 and z0), and set t = 1;
5: Iteratively update as follows:
   \[ x_t = (1 - \alpha)T(1)x_{t-1} + ox, \]
   \[ z_t = (1 - \beta)T(3)x_{t-1} + \beta r, \]
   (or as follows for MultiVCRankII:
   \[ x_t = (1 - \alpha)T(1)y_{t-1} + ox, \]
   \[ y_t = (1 - \beta)T(2)x_{t-1} + \beta o, \]
   \[ z_t = (1 - \gamma)T(3)x_{t-1}y_{t-1} + \gamma r, \]
   until it converges;
6: Rank images in I according to their association scores in x, and return the result.

**D. The Proposed Algorithms**

The proposed MultiVCRankI and MultiVCRankII methods are summarized in Algorithm 1. For both methods, the first two steps execute offline when an image database are given and the other four steps execute online when a query image is input.

**Complexity Analysis.** For the offline part, the major computational costs are from building the hypergraph G and tensor T. Assuming the image database I consists of p regions and each region is represented as d-dimensional features, grouping the regions into n clusters with K-means takes \( O(pdnl_kmeans) \) arithmetic operations \( t_k\text{means} \) (the number of iterations for clustering). Suppose that each cluster have the same number of regions, i.e., \( \lceil \frac{p}{n} \rceil \). According to Eq. (1), constructing tensor T thus costs \( O(\lceil \frac{p}{n} \rceil d) \) operations. The main computational complexities of online query lie at solving the tensor equations, which takes \( O(\lceil \frac{p}{n} \rceil \lceil \frac{r}{l} \rceil) \) arithmetic operations \( \lceil \frac{r}{l} \rceil \) represents the number of nonzero entries in tensor T and t is the number of iterations). Note that t is often less than 20 for both MultiVCRank models. Hence, as we will see in the experiments, the online query processes of MultiVCRank methods are very efficient.

**Theorem 2:** Suppose the non-negative tensor T is irreducible. If 1/2 < α, β < 1, then the MultiVCRankI method converges to the unique solution of (2). If 1/2 < α, β, γ < 1, then the MultiVCRankII method converges to the unique solution of Eq. (3) for any initial probability distribution vectors.

The proof can be found in Appendix B.

**E. Parameter Learning**

In MultiVCRank, we can see that the parameters α and β (or α, β and γ) balance the information from global features α, region features β and the hypergraph constructed in producing the final rankings. Their importance are data dependent, and might also rely on the qualities of features and regions. Without prior knowledge, it is hard to determine them. In this subsection, we present an algorithm to learn them with a small number of training samples. For simplicity, assume that we have \( \ell \) training queries and each query has a set of target images, denoted by \( R(\ell) \). As our objective is to produce the best ranking for images, we follow the idea of weighted approximation-rank pairwise loss function [42], and propose the following error function to minimize:

\[
err = \sum_{i=1}^{\ell} \sum_{j \in R(\ell)} L(\text{rank}(x^{(i)}_j)),
\]

where \( x^{(i)}_j \) denotes the score vector produced by MultiVCRank for the i-th query, and \( \text{rank}(x^{(i)}_j) \) indicates the rank of image j:

\[
\text{rank}(x^{(i)}_j) = \sum_{j \in R(\ell)} \theta(x^{(i)}_j) > x^{(i)}_j,
\]

where \( \theta \) is the indicator function, and \( L(\cdot) \) changes this rank into a loss:

\[
L(r) = \frac{1}{r}. \]

Clearly, the objective function in Eq. (6) considers the pairwise violations between target and non-target images by using different weights in terms of their positions in the ranked list, which leads to a focus on the top of the list and tends to rank target images higher.

Following [42], we consider stochastic gradient descend method for optimization. In other words, each time we randomly draw one query i and calculate the gradient on the draw to update. The expected loss for each query i can be rewritten as:

\[
err_i = \sum_{j \in R(\ell), j \notin R(\ell)} L(\text{rank}(x^{(i)}_j)) \frac{|x^{(i)}_j - x^{(i)}_j|_{++}}{\text{rank}(x^{(i)}_j)} \quad (7)
\]

where \( |x^{(i)}_j - x^{(i)}_j|_{++} = \max(0, x^{(i)}_j - x^{(i)}_j) \). For the sake of efficiency, Eq. (7) can be approximated by sampling. Specifically, given \( j \in R(\ell) \), we draw image \( j \notin R(\ell) \) randomly until we obtain one sample such that \( x^{(i)}_j > x^{(i)}_j \), and the \( \text{rank}(x^{(i)}_j) \) is approximated as:

\[
\text{rank}(x^{(i)}_j) = \left\lfloor \frac{m - 1}{N} \right\rfloor \quad (8)
\]

where N is the number of sampling trials. The Eq. (7) is then approximated as:

\[
err_i = \sum_{j \in R(\ell)} L(\text{rank}(x^{(i)}_j)) |x^{(i)}_j - x^{(i)}_j|_{++} \quad (9)
\]

Here the denominator disappears because the probability to draw \( j \) is \( 1/\text{rank}(x^{(i)}_j) \).
Considering MultiVCRankI, we can compute the gradients with respect to $\alpha$ and $\beta$ as:

$$\frac{\partial x}{\partial \alpha} = -T^{(1)} x z \tag{10}$$

$$\frac{\partial x}{\partial \beta} = (1 - \alpha)T^{(1)} \frac{\partial z}{\partial \beta} \tag{11}$$

As a result, given the $i$-th query, we can efficiently calculate the gradients with respect to $\alpha$ and $\beta$:

$$\frac{\partial err_1}{\partial \alpha} = \sum_{j \in R^{(i)}} L(rank(x_j)) \left( \frac{\partial x_j^i}{\partial \alpha} - \frac{\partial x_j^i}{\partial \alpha} \right) \tag{12}$$

$$\frac{\partial err_1}{\partial \beta} = \sum_{j \in R^{(i)}} L(rank(x_j)) \left( \frac{\partial x_j^i}{\partial \beta} - \frac{\partial x_j^i}{\partial \beta} \right) \tag{13}$$

The gradients for MultiVCRankII method can be derived similarly. With them, we summarize our learning algorithm in Algorithm 2. Note $\eta$ is a learning rate in the algorithm.

Algorithm 2 Parameter Learning

**Input:** $\ell$ queries with their corresponding target image sets \{R\}$_{i=1}^{\ell}$

**Output:** Optimal parameters $\alpha$, $\beta$ ($\gamma$ as well for MultiRankII)

**Procedure:**
1. Randomly initialize $\alpha$, $\beta$ (or $\gamma$ as well);
2. repeat
3. Randomly choose one query $i$ from $\ell$ queries;
4. Calculate $x^{(i)}$ by Algorithm 1;
5. For each $j \in R^{(i)}$, sample $\bar{j} \in R^{(i)}$ such that $x_j^{(i)} > x_j^{(i)}$ and estimate $rank(x_j^{(i)})$ as Eq. (8);
6. Update $\alpha = \alpha - \eta \frac{\partial err_1}{\partial \alpha}$, $\beta = \beta - \eta \frac{\partial err_1}{\partial \beta}$ (or $\gamma = \gamma - \eta \frac{\partial err_1}{\partial \gamma}$ as well);
7. Project $\alpha$, $\beta$ (or $\gamma$ as well) into [0,1] if they are not in this range;
8. until convergence
9. Return $\alpha$, $\beta$ (or $\gamma$ as well).

F. Overall Execution Procedure of Our Ranking Scheme

To better illustrate the execution procedure of our MultiVCRank framework, we depict the system block diagram in Figure 2. We can see that the system consists of two main components: offline part and online part. In the offline part, given an image database, we first extract features for each image and perform salient object detection to generate interesting regions. Subsequently, a hypergraph is built by using the method introduced in Section III-B. Then, we apply the parameter learning algorithm introduced in Section III-E to find the optimal parameters $\alpha$, $\beta$ (or $\gamma$ as well) for MultiVCRank model. We note only a small number of training samples are required to learn the parameters. In the online part, given a query image, we first run the salient object detection algorithm to identify interesting regions in the query image. After that, the query vectors $o$ and $r$ are constructed as Eqs. (4) and (5), respectively. Finally, the MultiVCRank algorithm is carried out to compute the association scores for all the images in database (as defined in Eq. (2) or Eq. (3)), and then produces a ranking result. All the blocks are implemented with MATLAB, except for the feature extraction and salient region detection, where we utilize existing softwares in [39] and [5].

IV. EXPERIMENTAL RESULTS

We conduct three experiments on different image datasets to demonstrate the effectiveness of the proposed MultiVCRank scheme. As for the comparison, we show the results of HypergraphRank [16], ManifoldRank [11], RankSVM [20], TOPHITS [23] and similarity based ranking algorithm in the three experiments.

A. Datasets, Image Features and Evaluation Metrics

**Datasets.** In our first two experiments, we utilize two image datasets. The first is the benchmark MSRC V2 image database, which contains 591 images of 20 categories. The second is a subset of images selected from Corel database, which contains 5000 images of 50 categories and each category has 100 images. The 50 categories are car, tractor, train, bus, race car, airplane, dinosaur, bear, elephant, wolf, lion, tiger, monkey, rhino, duck, horse, goat, deer, dog, fish, sailboat, Indians, bikini, fighters, tennis player, ship, swimmer, waves, lady model, ceremony, flower, leaf, mushroom, pot plant, castle, fireworks, drinks, vegetable, cloud, beach, waterfall, ballon, tower, 4 scene categories and 2 bird categories. For both datasets, we apply the salient region detection algorithm in [5] to generate the regions. As a result, we obtain 712 regions and 6198 regions for MSRC and Corel datasets, respectively.

In the last experiment, we test the robustness and query efficiency of our proposed MultiVCRank scheme by using a larger and noisier image database. Specifically, Caltech256 database is utilized, which includes 30608 web images of 257 categories. After salient region detection, 35506 image regions are obtained.

**Image Features.** To represent the images and image regions, the following five descriptors are employed: SIFT [29], C-SIFT, rgSIFT, OpponentSIFT and RGB-SIFT [39]. We use these features because according to the evaluation results on various image datasets in [39] combining the five
features produces the best performance. The sparse features for the five descriptors are extracted by using the Harris-Laplace point detectors. For each feature descriptor, we construct a codebook with 1024 vocabularies/bins by using K-means. As a result, the feature space in our experiments is $f(\cdot) \in \mathbb{R}^{1024}$, i.e., given an image $I_i$ (or a region $S_j$), we have $f(I_i) \in \mathbb{R}^{1024}$ (or $f(S_j) \in \mathbb{R}^{1024}$) referring its image feature representation.

Evaluation Metrics. Following [16], we report the precision vs. recall curve to evaluate the performance of different algorithms. Note that more areas under the curve denote better performance of the algorithm. With a set of queries, we report the average precision vs. recall curve for evaluation. In addition, we also adopt the precision at top-K position (denoted by precision@K) and Normalized Discounted Cumulative Gain at top-K position (denoted by NDCG@K) for evaluation.

B. Experiment I

In the first experiment, we test the retrieval performance of the proposed MultiVCRank scheme by utilizing MSRC and Corel datasets. For both datasets, we divide randomly the image collection into ten folds, and perform ten-fold cross validation. Specifically, each time we use eight-fold images as training set to construct the hypergraph, one-fold images as validation set to learn and turn parameters for different algorithms, and the remaining one-fold as query images for evaluation. We report the average results of the ten-fold test as evaluation. For both datasets, K-means clustering is performed to group the regions into 50, 100, 150 and 200 visual concepts. We find that the best performance is produced when 50 and 100 visual concepts are used for MSRC and Corel datasets, respectively, and thus use them in our experiments.

Table I shows the time costs for building and using a MultiVCRank model on the two data sets. The result reported is for MultiVCRankI model, and the time costs for MultiVCRankII are quite similar. We can see from the table that the model can be built in a reasonable amount of time, which are done offline; and since the constructed hypergraph contains very sparse edges, the query process can be very efficient as well.

1) Settings of Comparison Algorithms: In the following, we present the detailed settings of comparison algorithms.

- HypergraphRank_VC and ManifoldRank_VC: For these two algorithms, they both need an image-image similarity matrix for ranking purpose. To exploit the visual concept information, we aggregate the tensor $T$ into a similarity matrix $A$ by using a weighted sum: $[A]_{i,j} = \sum_{k=1}^n t_{i,j,k} r_k$, where vector $r$ is computed in the same way as in MultiVCRankI and MultiVCRankII for each query input. Both HypergraphRank algorithm and ManifoldRank algorithms are then performed based on this matrix $A$, with vector $o$ as initial labeling input for them. We tune the learning parameter $\mu$ in the interval $[1, 15]$ between the matrix and the initial label based on validation data.

- HypergraphRank and ManifoldRank: In this case, we compute the image-image similarity matrix $A$ without using region information. For images $i$ and $j$, $[A]_{i,j} = \exp\left(-\frac{||f(I_i)-f(I_j)||_2}{2\sigma^2}\right)$. We set $2\sigma^2 = 0.05$ whenever we use the Gaussian kernel function in the experiments. Other settings are similar to HypergraphRank_VC and ManifoldRank_VC, respectively.

- RankSVM: We use the authors’ implementation [20], and the validation data set is employed to learn the model.

- TOPHITS: For the TOPHITS algorithm, we perform it based on the tensor $T$ using 100-rank, 500-rank, and 1000-rank decomposition respectively, and report the best results. The input $r$ is used to perform query as in [23] to rank images.

- SimilarityRank: In this algorithm, the images are ranked merely based on their similarities to the query image. Specifically, we employed the vector $o$ to rank images.

2) Test of Convergence: First, we use a query example from Corel data to demonstrate the convergence of MultiVCRankI and MultiVCRankII. Figures 3(a), 3(b), and 3(c) show successive differences of probability vectors of $x$, $y$, $z$ computed in both algorithms, respectively. We see from these figures that the successive differences of probability vectors decrease monotonically, resulting in quite small differences (smaller than $1.0 \times 10^{-13}$) for different parameter settings within 20 iterations. We remark that even though $\alpha$, $\beta$ and $\gamma$ may not satisfy the requirement in Theorem 2, the two algorithms still converge.

3) Parameters learned: When learning the parameters for our proposed methods, we set the learning rate to be 0.01, and terminate the process when the difference between two successive updates is smaller than $1.0 \times 10^{-6}$. The learned parameters for ten fold cross validation are shown in Table II. The learned values for the parameters are consistent with the results when we perform grid search.

4) Performance Evaluation: Figures 4(a) and 4(b) show the precision vs. recall curves of all the algorithms on MSRC and Corel datasets, respectively. We observe that MultiVCRankI and MultiVCRankII always perform the best. The performance of HypergraphRank_VC and ManifoldRank_VC are also very promising, compared to HypergraphRank and ManifoldRank, because they both utilize the visual concept information from regions. The performance of RankSVM and TOPHITS are very poor, and are even significantly inferior to the similarity based rank method. For RankSVM, this is because the number of our training
Fig. 3. Convergence performance of MultiVCRankI and MultiVCRankII.

(a) MSRC Data.

(b) Corel Data.

Fig. 4. The retrieval performance of all algorithms on MSRC and Corel datasets, respectively. (Best viewed in colors.)

Fig. 5. The top ten results of different algorithms on MSRC data for a query image. The red lines mark the salient regions produced by the algorithm in [5]. The first row is the query image, and the second, third, fourth, fifth, sixth, seventh and eighth rows show the results of MultiVCRankI, MultiVCRankII, HypergraphRank_VC, ManifoldRank_VC, RankSVM, TOPHITS, and SimilarityRank, respectively.
Fig. 6. The top ten results of different algorithms on Corel data for a query image. The red lines mark the salient regions produced by the algorithm in [5]. The first row is the query image, and the second, third, fourth, fifth, sixth, seventh and eighth rows show the results of MultiVCRankI, MultiVCRankII, HypergraphRank_VC, ManifoldRank_VC, RankSVM, TOPHITS, and SimilarityRank, respectively.

TABLE II

MUL**I**VCRAn**K**I AND MULLI**VCR**ANKII IN TEN FOLD CROSS VALIDATION.

<table>
<thead>
<tr>
<th></th>
<th>MSRC</th>
<th>Corel</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>α</td>
<td>β</td>
</tr>
<tr>
<td>MultiRankI</td>
<td>0.84</td>
<td>0.71</td>
</tr>
<tr>
<td>MultiRankII</td>
<td>0.80</td>
<td>0.72</td>
</tr>
<tr>
<td></td>
<td>γ</td>
<td></td>
</tr>
<tr>
<td>MultiRankI</td>
<td>0.73</td>
<td>0.65</td>
</tr>
<tr>
<td>MultiRankII</td>
<td>0.70</td>
<td>0.64</td>
</tr>
</tbody>
</table>

TABLE III

THE PERFORMANCE COMPARISON IN TERMS OF PREC**IS**ION@K AND NDCG@K ON MSRC AND COREL DATASETS.

<table>
<thead>
<tr>
<th></th>
<th>MSRC</th>
<th>Corel</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>P@5</td>
<td>P@10</td>
</tr>
<tr>
<td>MultiVCRankI</td>
<td>0.4867</td>
<td>0.4506</td>
</tr>
<tr>
<td>MultiVCRankII</td>
<td>0.5000</td>
<td>0.4683</td>
</tr>
<tr>
<td>HypergraphRank_VC</td>
<td>0.3521</td>
<td>0.3196</td>
</tr>
<tr>
<td>HypergraphRank</td>
<td>0.2007</td>
<td>0.1833</td>
</tr>
<tr>
<td>ManifoldRank_VC</td>
<td>0.3267</td>
<td>0.2983</td>
</tr>
<tr>
<td>ManifoldRank</td>
<td>0.1521</td>
<td>0.1018</td>
</tr>
<tr>
<td>SVMRank</td>
<td>0.1433</td>
<td>0.1183</td>
</tr>
<tr>
<td>TOPHITS</td>
<td>0.1301</td>
<td>0.1083</td>
</tr>
<tr>
<td>SimilarityRank</td>
<td>0.2633</td>
<td>0.2367</td>
</tr>
</tbody>
</table>

samples is too small compared to that of feature dimensionality, e.g., we have around 60 training samples for each fold while have 5120 features in MSRC data, which makes learned model less effective. For TOPHITS, this is because the decomposition components are in different scales, and some components are too dominant to produce effective performance. For further evaluation, we also show the precision and NDCG at top-K position in Table III. We can see that MultiVCRankI and MultiVCRankII consistently outperform baseline methods.

Figures 5 and 6 give an example to show the top ten results obtained by different algorithms on MSRC and Corel datasets, respectively. We observe that although the salient regions detected may not be very accurate and clean, the proposed methods can still produce excellent performance, which benefit from the mutual-reinforcing relationship between images and the visual concept regions in our ranking scheme.

C. Experiment 2

Relevance feedback (RF) techniques are quite useful in boosting the performance of CBIR systems and algorithms.
by exploiting online information generated by users [38], [44], [16], [35]. It can be considered as a procedure that positive and negative examples are gradually provided. In this experiment, we aim to test the performance of the proposed MultiVCRank scheme in the RF setting. MSRC and Corel datasets are utilized.

To simulate the RF procedure and evaluate the performance, we randomly select 5 positive examples and 5 negative examples in the first round. In the second round, another 5 positive and negative examples are provided as training examples, respectively. As a result, we have 10 and 20 examples as training set, respectively, after each of the two round feedbacks. The other images of the same category as positive examples are considered target images to be retrieved for evaluation.

1) Settings of Different Algorithms:
- MultiVCRankI and MultiVCRankII: With positive and negative image examples, we calculate \( o^+ \) and \( r^+ \) by taking the average of \( o \) and \( r \) for the positive examples; similarly, \( o^- \) and \( r^- \) could be computed for negative examples. Using \( o^+ \) and \( r^+ \) in step 3 of Algorithm 1, \( x^+ \) are obtained. Also, we calculate \( x^- \). Finally, MultiVCRankI and MultiVCRankII use \( x^{(c)} = x^+ - x^- \) to rank images.
- HypergraphRank_VC and ManifoldRank_VC: For the initial labeling vector, we set the entries corresponding to positive examples to be 1, the entries corresponding to negative examples to be -1, and the other entries to be 0, which are standard RF settings for both algorithms, see [16], [11]. Other settings are the same as in Experiment 1.
- HypergraphRank and ManifoldRank: The entries of initial labeling vector are set to be 1, -1 or 0 similar to HypergraphRank_VC and ManifoldRank_VC. Other settings are the as in Experiment 1.
- RankSVM: We learn a model for each query by using the corresponding positive and negative examples as training set.
- TOPHITS: We use \( r^+ \) and \( r^- \) for input query to obtain the \( x^+ \) and \( x^- \) respectively. Images are ranked based on \( x^{(c)} = x^+ - x^- \).
- SimilarityRank: This method ranks images based on \( x = o^+ - o^- \).

2) Performance Evaluation: Figures 7 and 8 show the performance of all algorithms on MSRC and Corel datasets, respectively. Compared to the performance in experiment, we observe that RF techniques indeed boost the performance of all algorithms in the first round feedback. The improvement can be further enhanced in the second round. Again, we find that the performance of MultiVCRankI
and MultiVCRankII are better than those of the other algorithms, which demonstrates the effectiveness of the proposed ranking scheme.

D. Experiment 3

In this experiment, we utilize a large web image database, Caltech256, to test the robustness and query efficiency of the developed MultiVCRank scheme. There are 257 categories in the database, and we randomly select 5 images for each category as test queries. The remaining images are used to develop the retrieval models. For MultiVCRank, we group the identified image regions into 300 clusters and then build the MultiVCRankI and MultiVCRankII models. It takes around 9.5 hours to build each of the two models, including salient region detections, hypergraph construction and parameter learning. Although the running time is much longer than the ones used in previous experiments (see Table I), we note this process is done offline and hence the time is acceptable. Our main aim in the experiment is to examine the retrieval performance, in terms of both accuracy and query efficiency, when a large and noisy image database is presented.

Table IV presents the retrieval accuracy comparisons of MultiVCRank models and baseline methods. We can see that the MultiVCRankI and MultiVCRankII models perform substantially better than all the baseline methods, in terms of both precision and NDCG metrics. The result suggests that our MultiVCRank scheme is robust enough to handle retrieval problems on large and noisy image database. Figure 9 reports the response time of all the methods to answer a query. We observe that RankSVM, TOPHITS and SimilarityRank are the most efficient (less than 1000 ms); however, their retrieval accuracies are not good (see Table IV). The remaining approaches, including MultiVCRank models, take a little bit longer time (around 2500 ms or so), because they are all graph-based method and need to compute rankings in an iterative manner. By considering the retrieval accuracy, the effectiveness and efficiency of MultiVCRank models are still competitive.

Finally, we test the scalability of the proposed ranking schemes in both the training and testing phases. To this end, we utilize different ratios of images from the Caltech256 database, which are 1/16, 1/8, 1/4, 1/2 and 1.0, and record the time costs in the two phases when these ratios are used. Figure 10(a) presents growths of training time costs against the ratios for MultiVCRankI and MultiVCRankII. We observe that training costs of both methods grow linearly with respect to the number of images. This suggests our proposed methods are quite good in scalability. Similarly, Figure 10(b) reports the time costs in testing phase. We find that both methods scale linearly in the phase, and MultiVCRankI costs less time than MultiVCRankII as the number of images increases. The reason is that MultiVCRankII has one more set of tensor equations than MultiVCRankI, and thus it takes more time.

V. CONCLUDING REMARKS

In this paper, we have proposed the MultiVCRank scheme by constructing a hypergraph to exploit multiple visual concepts and construct affinities among images in these visual concepts. In the scheme, we define two mutually-reinforcing scores to indicate how images and visual concepts are associated to an input query example. The MultiVCRankI and MultiVCRankII algorithms are employed to compute them simultaneously. We have shown the convergence of both algorithms under suitable
conditions. Extensive experiments are conducted on various image databases to test the performance of the proposed MultiVCRank scheme, and the results demonstrated MultiVCRank outperforms existing ranking methods substantially.

In this paper, we utilize several handcrafted SIFT-features for representing images and salient regions. As recent studies have shown that deep learning representations outperform SIFT-features in many image processing applications, it thus would be very interesting to generate deep learning features to represent images and regions, and then apply our MultiVCRank model. We would like to explore this and examine the performance in the future.

VI. ACKNOWLEDGEMENT

This research was supported in part by NSFC under Grant No.61272538 and No. 61572158, Shenzhen Science and Technology Program under Grant No. JCYJ20140417172417128 and No. JSGG20141017150830428, and by HKRGC GRF Grant Nos. 202013 and 12302715, and HKBU FRG2/14-15/087.

APPENDIX A

PROOF OF THEOREM 1

The existence results for Eqs. (2) and (3) can be found in [30], [25]. Since the proof of the uniqueness result for Eq. (2) is similar to that for Eq.(3). Here we only give the proof of the uniqueness result for Eq. (3). We define the mapping \( \Phi : \Omega \rightarrow \Omega \) as follows

\[
\Phi([x, y, z]^T) = \begin{bmatrix}
(1 - \alpha)T^{(1)}yx + \alpha_0, \\
(1 - \beta)T^{(2)}xh + \beta_0, \\
(1 - \gamma)T^{(3)}xy + \gamma_0,
\end{bmatrix}
\]

where \( \Omega = \Gamma_m \times \Gamma_m \times \Gamma_n \) and \( \Gamma_m = \{ u | u = [u_1, u_2, \ldots, u_n] \in \mathbb{R}^n, u_i \geq 0, 1 \leq i \leq s, \sum u_i = 1 \} \).

The Jacobian matrix of \( \Phi \) is given by

\[
\Phi'(w) = \begin{bmatrix}
(1 - \alpha)P_{1,2}(w) & (1 - \alpha)P_{1,3}(w) & 0 \\
(1 - \beta)P_{2,1}(w) & (1 - \beta)P_{2,3}(w) & 0 \\
(1 - \gamma)P_{3,1}(w) & (1 - \gamma)P_{3,2}(w) & 0
\end{bmatrix}
\]

where \( w = [x, y, z]^T \in \Omega \) and \( \Phi'(w) \) is a \((2m + n)\times(2m + n)\) matrix. The main diagonal blocks of \( \Phi'(w) \) are zero matrices. \( P_{1,2}(w) \) and \( P_{2,1}(w) \) are \( m \times m \) non-negative matrices. \( P_{1,3}(w) \) and \( P_{2,3}(w) \) are \( m \times n \) non-negative matrices. \( P_{3,1}(w) \) and \( P_{3,2}(w) \) are \( n \times n \) non-negative matrices. The \((i, j)\)-entry in \( P_{1,2}(w) \) and \( P_{1,3}(w) \) are given by

\[
[P_{1,2}(w)]_{i,j} = \sum_{k=1}^{m} f_{i,j,k}^1 z_k,
\]

respectively. The \((i, j)\)-entry in \( P_{2,1}(w) \) and \( P_{2,3}(w) \) are given by

\[
[P_{2,1}(w)]_{i,j} = \sum_{k=1}^{m} f_{i,j,k}^2 z_k,
\]

respectively. The \((i, j)\)-entry in \( P_{3,1}(w) \) and \( P_{3,2}(w) \) are given by

\[
[P_{3,1}(w)]_{i,j} = \sum_{k=1}^{n} f_{i,j,k}^3 x_k,
\]

respectively.

Next we will show the spectral radius of \( \Phi'(w) \) is less than 1 for any \( w \in \Omega \). Let us define the following vector-norm

\[
||u|| = \max\{||u_1||, ||u_2||, ||u_3||, u_1, u_2 \in \mathbb{R}^m, u_3 \in \mathbb{R}^n \}.
\]

Here \( || \cdot || \) is the 1-norm for a vector. Clearly, \( || \cdot || \) is a vector-norm on \( \mathbb{R}^{2m+n} \). Then we can define the following matrix norm

\[
||U|| := \sup\{||Uu|| : ||u|| = 1\}.
\]

Since \( P_{k,l}(w) \) is a non-negative matrix and its column sum is 1, we have

\[
||P_{k,l}(w)||u|| \leq ||u|| \leq 1, \forall u \in \mathbb{R}^m \text{ or } \mathbb{R}^n \text{ with } ||u|| \leq 1.
\]

It follows that if \( ||u|| = 1 \) (i.e., \( ||u|| \leq 1 \)), we have

\[
||((1 - \alpha)P_{1,2}(w)u_2 + (1 - \beta)P_{1,3}(w)u_3) || \leq 2(1 - \alpha)||u|| = 2(1 - \alpha),
\]

\[
||((1 - \beta)P_{2,1}(w)u_1 + (1 - \beta)P_{2,3}(w)u_3) || \leq 2(1 - \beta)||u|| = 2(1 - \beta),
\]

\[
||((1 - \gamma)P_{3,1}(w)u_1 + (1 - \gamma)P_{3,2}(w)u_2) || \leq 2(1 - \gamma)||u|| = 2(1 - \gamma).
\]

It follows that

\[
||\Phi'(w)|| \leq \max\{2(1 - \alpha), 2(1 - \beta), 2(1 - \gamma)\} < 1.
\]

In [21], it has been given a general condition which guarantees the uniqueness of the fixed point in the Brouwer Fixed Point Theorem, namely, (i) 1 is not an eigenvalue of the Jacobian matrix of the mapping, and (ii) for each point in the boundary of the domain of the mapping, it is not a fixed point. In the first part, it has been mentioned that solution vectors are positive when \( T \) is irreducible, i.e., they do not lie on the boundary \( \partial \Omega \) of \( \Omega \). Therefore, the positive solution of Eq. (3) is unique.

APPENDIX B

PROOF OF THEOREM 2

Since the proof of the convergence of MultiVCRankII is similar to that of MultiVCRankII. Here we only give the proof of the convergence of MultiVCRankII.

Without loss of generality, we consider the proof for \( x \). Note that

\[
x_{t+1} - x_t = (1 - \alpha)T^{(1)}(y_t, z_t, y_{t-1}, z_{t-1})
\]

\[
= (1 - \alpha)T^{(1)}(y_t - y_{t-1}, z_t - z_{t-1})
\]

\[
= (1 - \alpha) (S^{(1)}(y_t - y_{t-1}) + S^{(2)}(z_t - z_{t-1}))
\]

where \( S^{(1)} = T^{(1)}z_t \) is an \( m \times m \) matrix with its entries given by

\[
[S^{(1)}]_{i,j} = \sum_{k=1}^{m} f_{i,j,k}^1 z_k,
\]

and \( S^{(2)} = T^{(1)}y_t \) is an \( m \times m \) matrix with its entries given by

\[
[S^{(2)}]_{i,j} = \sum_{k=1}^{n} f_{i,j,k}^2 y_k.
\]

We note that

\[
\sum_{i=1}^{m} [S^{(1)}]_{i,j} = \sum_{i=1}^{m} \sum_{k=1}^{n} f_{i,j,k}^1 z_k
\]

\[
= \sum_{k=1}^{n} z_k \sum_{i=1}^{m} f_{i,j,k}^1
\]

\[
= \sum_{k=1}^{n} z_k = 1
\]

therefore the column sum of \( S^{(1)} \) is equal to 1. Similarly, the column sum of \( S^{(2)} \) is equal to 1. By using the fact \( S^{(1)} \) and \( S^{(2)} \) are non-negative matrices, we know that the entries \([S^{(1)}]_{i,j} \) and \([S^{(2)}]_{i,j} \) are in between \([0, 1]\). It follows that

\[
||x_{t+1} - x_t||_1 \leq (1 - \alpha) ((||y_t - y_{t-1}||_1 + ||z_t - z_{t-1}||_1)) + ||x_t - z_{t-1}||_1.
\]
By similar arguments, we also have

\[ \|y_{t+1} - y_t\| \leq (1 - \beta)(\|x_t - x_{t-1}\| + \|z_t - z_{t-1}\|). \]

and

\[ \|z_{t+1} - z_t\| \leq (1 - \gamma)(\|x_t - x_{t-1}\| + \|y_t - y_{t-1}\|). \]

To sum them together, we obtain

\[ \|x_{t+1} - x_t\| + \|y_{t+1} - y_t\| + \|z_{t+1} - z_t\| \leq (2 - \beta - \gamma)(\|x_t - x_{t-1}\| + (2 - \alpha - \gamma)(\|y_t - y_{t-1}\| + (2 - \alpha - \beta)(\|z_t - z_{t-1}\|).

\[ \leq c(\|x_t - x_{t-1}\| + \|y_t - y_{t-1}\| + \|z_t - z_{t-1}\|), \]

where
\[ c = \max(2 - \beta - \gamma, 2 - \alpha - \gamma, 2 - \alpha - \beta) < 1. \]

Hence
\[ \lim_{t \to \infty} (\|x_t - x_{t-1}\| + \|y_t - y_{t-1}\| + \|z_t - z_{t-1}\|) = 0, \]

i.e., \( x_t, y_t, z_t \) generated by the algorithm, converge. By considering \( x_{t-1} - x^* \) and using the similar trick, we can show that
\[ \lim_{t \to \infty} x_t = x^*, \quad \lim_{t \to \infty} y_t = y^* \quad \text{and} \quad \lim_{t \to \infty} z_t = z^*, \]

where \( x^*, y^* \) and \( z^* \) are the unique solutions of Eq. (3) stated in Theorem 1.

REFERENCES


Xutao Li is now an associate professor in the Shenzhen Graduate School, Harbin Institute of Technology. He received the PhD and Master degrees in Computer Science from Harbin Institute of Technology in 2013 and 2009, and the Bachelor from Lanzhou University of Technology in 2007. His research interests include data mining, machine learning, graph mining and social network analysis, especially tensor based learning and mining algorithms.

Yunming Ye is a Professor in the Shenzhen Graduate School, Harbin Institute of Technology. He received the Ph.D. degree in Computer Science from Shanghai Jiao Tong University. His research interests include data mining, machine learning, graph mining and social network analysis, and ensemble learning algorithms.

Michael K. Ng is a Chair Professor in the Department of Mathematics at the Hong Kong Baptist University. He obtained his B.Sc. degree in 1990 and M.Phil. degree in 1992 at the University of Hong Kong, and Ph.D. degree in 1995 at Chinese University of Hong Kong. His research interests include bioinformatics, image processing, scientific computing and data mining, and he serves on the editorial boards of international journals, see http://www.math.hkbu.edu.hk/~mng.