Self-Supervised Image Re-Ranking based on Hypergraphs and Graph Convolutional Networks

Leonardo Tadeu Lopes, Lucas Pascotti Valem, Daniel Carlos Guimarães Pedronette
Department of Statistics, Applied Mathematics and Computing
São Paulo State University (UNESP), Rio Claro, Brazil
leonardo.lopes@unesp.br, lucas.valem@unesp.br, daniel.pedronette@unesp.br

***Abstract*—Image retrieval approaches typically involve two fundamental stages: visual content representation and similarity measurement. Traditional methods rely on pairwise dissimilarity metrics, such as Euclidean distance, which overlook the global structure of datasets. Aiming to address this limitation, various unsupervised post-processing approaches have been developed to redefine similarity measures. Diffusion processes and rank-based methods compute a more effective similarity by considering the relationships among images and the overall dataset structure. However, neither approach is capable of defining novel image representations. This paper aims to overcome this limitation by proposing a novel self-supervised image re-ranking method. The proposed method exploits a hypergraph model, clustering strategies, and Graph Convolutional Networks (GCNs). Initially, an unsupervised rank-based manifold learning method computes global similarities to define small and reliable clusters, which are used as soft labels for training a semi-supervised GCN model. This GCN undergoes a two-stage training process: an initial classification-focused stage followed by a retrieval-focused stage. The final GCN embeddings are employed for retrieval tasks using the cosine similarity. An experimental evaluation conducted on four public datasets with three different visual features indicates that the proposed approach outperforms traditional and recent rank-based methods.**

I. Introduction

With the significant advancements in multimedia acquisition, storage, and dissemination technologies, image retrieval approaches capable of considering the visual content to search for and retrieve pertinent multimedia data, have garnered substantial interest from both industry and academia [1]. In Content-Based Image Retrieval (CBIR) systems, the ranking process generally involves two fundamental stages: the representation of image content and the measurement of similarity between the query image and images in a collection. The representation phase involves encoding an image as a point within a high-dimensional feature space. Subsequently, the similarity measurement is concerned with determining the proximity of the feature space representations of database images to the query image. Traditionally, this is achieved by calculating the pairwise dissimilarity between feature vectors using metrics such as the Euclidean distance.

Once images (and other multimedia data) are often represented in spaces of much smaller dimensions than their respective feature vectors, exploiting the intrinsic structure of datasets becomes a central problem in retrieval, learning, and computational vision tasks [2]–[4]. Pairwise distance measures (as the Euclidean distance) define relationships only between pairs of images, the global structure of the dataset and the context wherein the query is computed are ignored. In general, this is the central point of re-ranking methods based on unsupervised similarity learning methods, whose objective is to compute more effective distances among images, capable of taking into account the relationships among images and the global structure of datasets.

In fact, various different unsupervised post-processing approaches have been proposed for retrieval tasks during the last decades [2]–[9]. Among them, two categories can be highlighted: diffusion processes [2], [3], [5] and rank-based [4], [6]–[9] approaches. Diffusion processes often rely on a graph and on spreading the affinities through that graph. The definition of a global measure describes the relationship between pairs of points in terms of their connectivity. Rank-based approaches employ various distinct techniques exploiting the ranking structure, which defines similarity relationships not only between pairs but among a set of images. In spite of the differences, both approaches (diffusion and rank-based) are based on the redefinition of similarity/dissimilarity measures among images.

However, both diffusion and rank-based approaches redefine the similarity among images without redefining their representation. On the other hand, Graph Convolutional Networks (GCNs) exploit multidimensional feature vectors and graph-based structures to learn more effective representations (embeddings) [10]. In opposite to Convolutional Neural Networks (CNNs), which often apply convolutions in the Euclidean space, GCNs allow convolution operations in non-Euclidean domains defined by graphs. Supported by such flexibility, various GCN models have been proposed [11], achieving impressive results in several tasks, especially semi-supervised classification.

Recently, GCN models have been exploited for a clustering approach [12]. The Self-Supervised Graph Convolutional Clustering (SGCC) exploits the strengths of different learning paradigms, combining unsupervised, semi-supervised, and self-supervised perspectives. Firstly, an unsupervised rank-based manifold learning uses a hypergraph model to compute a more global similarity and define reliable and small clusters. The small clusters are modeled as soft-labels for training a semi-supervised GCN, used for classification. Finally, SGCC uses the GCN embeddings to assign data items to clusters.

This paper proposes a novel image re-ranking method named *Self-Supervised GCN for Re-Ranking* (SGRR). The method uses visual features trained through transfer learning based on CNNs and recent Transformers models. Such features are taken as input by SGCC [12] for defining hypergraphs and the small clusters used as soft-labels. The soft-labels are subsequently used by a GCN model, which is trained through a two-stage procedure. A loss focused on classification is initially used, followed by a second stage based on the triplet loss, where the triplets are also defined by the soft-labels. Finally, the embeddings computed by GCNs are used for retrieval tasks, considering the cosine similarity for ranking.

An experimental evaluation was conducted on 4 public datasets considering 3 different visual features. The proposed approach achieved highly effective retrieval results compared

with the original results and recent rank-based approaches as baselines. The main contributions of the proposed method can be summarized as:

- A novel self-supervised image re-ranking is proposed. In contrast to diffusion and rank-based methods, the proposed approach learns a novel representation given by the GCN embeddings;
- A two-stage training procedure is proposed, considering both classification and retrieval-focused loss functions.

The remainder of the paper is organized as follows: Section II discussed a formal definition of the problem. Section III provides an overview of the proposal. Section IV discusses the SGCC approach and Section V defines the proposed re-ranking approach. Section VI presents the experimental evaluation and Section VII draws the conclusions.

II. Retrieval and Re-Ranking Formal Definition

In this section, we formally define the retrieval and ranking models considered for this work, mostly following [12]. First, let $\mathcal{C} = \{o_1, o_2, \ldots, o_n\}$ be a collection, where each object o_i denotes a data object. Second, let $\mathbf{x}_i$ be a feature vector defined in $\mathbb{R}^d$, which represents an $o_i \in \mathcal{C}$ element in a d-dimensional feature space, which can be used for retrieval and machine learning tasks, and is commonly supported on distance or similarity measures computed between pairs of objects.

Formally, let ρ: $\mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}^+$ be a function that, based on their feature vectors, computes the distance between two objects. Therefore, the distance between o_i and o_j can be defined by $\rho(\mathbf{x}_i, \mathbf{x}_j)$. The traditional Euclidean distance is often employed.

However, focusing solely on pairs of objects can overlook valuable information embedded in more complex relationships. In this context, rank-based techniques aim to represent and utilize rich contextual similarity data.

A ranked list τ_q, based on the distance function ρ, can be computed to identify the most similar objects to a given element o_q. Consequently, τ_q=$(o_1, o_2, \ldots, o_l)$ can be formally defined as a permutation of the collection $\mathcal{C}_l$, where l indicates the length of the ranked list and $\mathcal{C}_l \subset \mathcal{C}$ is a subset containing the l objects most similar to o_q.

Additionally, the permutation τ_q is a bijection from the set $\mathcal{C}l$ to the set $[L_g] = 1, 2, \ldots, l$. Additionally, $\tau_q(o_i)$ represents the position of the object o_i in the ranked list τ_q. If oi is ranked before o_j in o_q's ranked list, meaning $\tau_q(o_i) < \tau_q(o_j)$, then $\rho(\mathbf{x}_q, \mathbf{x}_i) \leq \rho(\mathbf{x}_q, \mathbf{x}_j)$. By computing a ranked list τ_i for each object $o_i \in \mathcal{C}$, we obtain the set $\mathcal{T}$ = $\tau_1, \tau_2, \ldots, \tau_n$ of ranked lists. This set encodes crucial similarity information, reflecting the dataset's structure. Rank-based manifold learning algorithms leverage the similarity data embedded in the set of ranked lists $\mathcal{T}$ to compute a new similarity measure, which can then be used to update the ranked lists. Formally, we can define an unsupervised manifold learning method as a function $m(\cdot)$, which computes a more effective set of ranked lists $\mathcal{T}' = m(\mathcal{T})$.

III. Overview of Proposed SGRR Approach

The main objective of the proposed SGRR method consists of exploiting contextual similarity information for image re-ranking. With this aim, two models are used to encode the contextual similarity information. Firstly, a hypergraph is used to encode first and second-order neighborhood similarity information (neighbors and neighbors of neighbors). The hypergraph is also used to define small and reliable clusters as soft-labels. Subsequently, a GCN is trained using the soft-labels and a graph representation of similarity computed by the hypergraph model. The soft-labels are used through a two-stage training procedure, with two different loss functions.

Figure 1 illustrates the steps of the proposed SGRR approach. The next sections describe the main steps in detail.

IV. Clustering by Manifold Learning based on Hypergraphs

A. Manifold Learning based on Hypergraphs

The *Log-based Hypergraph of Ranking References (LHRR)* [8] is an unsupervised manifold learning method that computes more effective similarities among data elements. The method is based on ranking information from the set of ranked lists $\mathcal{T}$ modeled in hypergraph structures. The algorithm can be broadly divided into three main steps, which are described in the following sections:

1) Rank Normalization: Firstly, LHRR computes a new similarity measure by using reciprocal rank positions. Using the computed new similarity, the top-l elements from the ranked lists are reordered using a stable sorting algorithm.

2) Hypergraph Construction: Hypergraphs are a powerful generalization of graphs, where hyperedges can connect any set of vertices. Let $G = (V, E, w)$ be a hypergraph consisting of a finite set of vertices V and a set of hyperedges E. Each item $o_i \in \mathcal{C}$ is associated with a vertex, $v_i \in V$, and the hyperedge set E is defined as a collection of subsets of V.

Following this definition, LHRR starts the creation of a hypergraph by defining a relevance weight function $w_p(o_i, o_z) = 1 - \log_k \tau_i(o_z)$ that computed the relevance of an element o_z to an element o_i based on the log value of its ranked list position.

Using $wp(\cdot)$, the function $r(e_i, v_j)$ is computed by multiplying the relevance between all elements that are neighbors from both o_i and o_j. Both relevance functions are applied to create an incidence matrix $\mathbf{H}$ of size $|E| \times |V|$, such that:

$$h(e_i, v_j) = \begin{cases} r(e_i, v_j) & \text{if } v_j \in e_i, \\ 0 & \text{otherwise.} \end{cases}$$

Additionally, the Hyperedge Weight $w(e_i)$ measures the confidence of the relationships between the objects in hyperedge e_i. The weight $w(e_i)$ is computed as the sum of $h(e_i, \cdot)$ for all k elements with the highest scores in the hyperedge e_i.

3) Hypergraph-Based Similarity: The Hypergraph-based similarity matrix $\mathbf{W} = \mathbf{Q} \circ \mathbf{S}_p$ is obtained by combining the hyperedge relationship, encoded in the matrix $\mathbf{S}_p$ and the vertices pairwise relationship, encoded in the matrix $\mathbf{Q}$. This final similarity matrix, which concentrates all similarity information extracted from the hypergraph, is used to compute a new set of ranked lists for the data collection.

Finally, by generating an improved set of ranked lists, LHRR can be executed sequentially over t iterations.

B. Clustering through Hypergraph Structures

The *Self-Supervised Graph Convolutional Clustering (SGCC)* [12] is an algorithm that exploits the similarity information encoded into the hypergraph structures, to separate the data into small and reliable clusters, which are later used as soft-labels for training a GCN in a semi-supervised manner. The creation of these soft-labels can be described in three main steps, presented below.

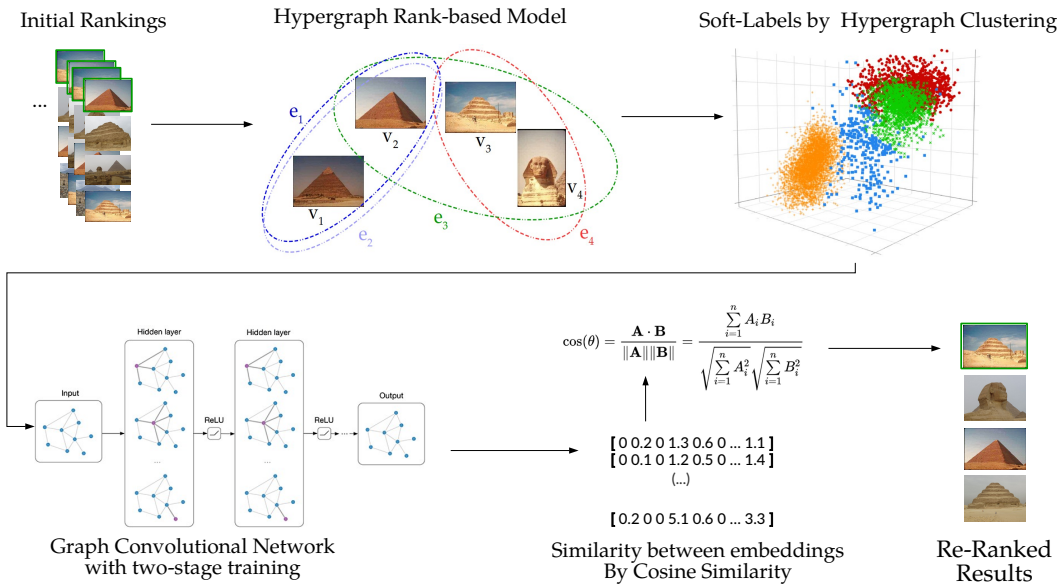


Fig. 1: Overview of the proposed Self-Supervised GCN for Re-Ranking (SGRR).

1) Hyperedge Self-Confidence Score: Using the hyperedge weight and the incidence of an element in its own hyperedge, SGCC defines a function $w_h(e_i) = h(e_i, v_i) \times w(e_i)$, which explores both estimations to compute a self-confidence score. The higher the score value, the more important an element o_i is for its neighbors.

Based on w_h, the ranked list τ_h=$(o_1,\ o_2,\ \ldots,\ o_n)$ is defined as a permutation of the collection $\mathcal{C}$ such that if o_i is ranked ahead of o_j, then $w_h(e_i) \geq w_h(e_j)$. The ranked list τ_h establishes the sequence in which the dataset items are processed by the proposed algorithm, ensuring that the more reliable items are selected and combined earlier.

2) Representatives Proxy Selection: Despite containing a reliable order of elements, τ_h can include various items similar to each other at top positions. To surpass this limitation, SGCC creates a selection of *representatives*, defined by $\mathcal{R} = (o_1, o_2, \ldots, o_c) \in \mathcal{C}$, where $|\mathcal{R}| = c$.

The selection criteria for representatives can be described as follows: prioritize candidates with a high self-confidence score (numerator) and minimal similarity to previously selected representatives (denominator). The set of representatives $\mathcal{R}$ starts with the first element in τ_h, which is the element with the highest self-confidence score in the collection. After that, $c-1$ iterations are conducted to select the remaining representatives.

Based on $\mathcal{R}$, the initial clusters set $\mathcal{S}$ can be defined, such that $|\mathcal{S}| = c$ and $\forall \mathcal{S}_i \in \mathcal{S},\ \mathcal{S}_i = \{r_i \in \mathcal{R}\}$, creating a unitary cluster for each representative object.

3) Reliable Clusters Set: The hyperedge is a powerful representation of the relationship between multiple data elements at once, which can be extended for the clusters created in IV-B2.

Therefore, a cluster assignment degree $h_s(\mathcal{S}_i, v_j)$ is defined by the sum of all similarity values between an object, represented by v_j, and all the elements contained in the cluster $\mathcal{S}_i$. Additionally, the function $n_c(o_i)$ encounters the most similar cluster to o_i, by using $h_s(\cdot)$ and the size of each cluster $\mathcal{S}_i \in \mathcal{S}$.

After a cluster is selected, o_i is agglomerated to it, with o_i's hyperedge being used to update the cluster's hyperedge as well. This agglomeration process is conducted until the q first elements of τ_h are allocated into clusters, where $q = \text{round}(n \times p)$, $n = |\mathcal{C}|$ and $p \in (0, 1)$ is a constant.

After the agglomeration step, SGCC recovers a highly reliable initial cluster configuration, which can be used as soft-labels for training a GCN in a semi-supervised classification task. In this work, we explore the soft-labels to train our GCN model, evolving the approach proposed in [12]. The soft-labels usage is described in the next section.

V. Self-Supervised Graph Convolutional Network for Image Re-Ranking

A. Graph Convolutional Network (GCN)

Substantial progress has been made in developing deep learning techniques specifically for graph data [13] in recent years. In [10], a two-layer GCN model for semi-supervised classification, utilizing a graph described by a symmetric adjacency matrix $\mathbf{A}$. The goal of Graph Convolutional Networks (GCNs) is to learn node representations (embeddings) by iteratively aggregating information from neighboring nodes, effectively capturing the graph's structure in a neural network model. Therefore, the resulting model can be expressed as a function of both the feature matrix $\mathbf{X}$ and the adjacency matrix $\mathbf{A}$: $\mathbf{Z} = f(\mathbf{X}, \mathbf{A})$.

In this context, $\mathbf{Z}$ denotes an embedding matrix, where $\mathbf{Z} = [\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_n]^T \in \mathbb{R}^{n \times d}$. Each $\mathbf{z}_i$ represents a d-dimensional embedded vector for the node v_i. Firstly, the degree matrices are computed during a preprocessing step, which involves defining $\hat{\mathbf{A}} = \tilde{\mathbf{D}}^{-1/2}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-1/2}$, where $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$ and $\tilde{\mathbf{D}}$ is the degree matrix corresponding to $\tilde{\mathbf{A}}$. Afterward, the matrix $\mathbf{Z}$ for a two-layer GCN model, can be obtained by the function $f(\cdot)$:

$$\mathbf{Z} = log(softmax(\hat{\mathbf{A}} ReLU(\hat{\mathbf{A}}\mathbf{X}\mathbf{W}^{(0)})\mathbf{W}^{(1)})) \qquad (1)$$

Where $\mathbf{W}^{(0)}, \mathbf{W}^{(1)} \in \mathbb{R}^{d \times H}$ are the network weights for the input-to-hidden and the hidden-to-output layers, respectively, and H represents the number of feature maps. Both matrices are optimized using gradient descent on the cross-entropy error calculated over the labeled node set $\mathcal{V}_L$. Following the embedding process, each node's embedded representation $\mathbf{z}_i$ undergoes a softmax activation function row-wise, resulting in a probability distribution over d class labels.

The label assignment for each node v_i is determined by selecting the class with the highest log probability from $\mathbf{z}_i$.

In this work, based on recent research applications [14] and on results obtained in [12], the *Simple Graph Convolution (SGC)* [15], which is a simplified GCN obtained by the collapse of weight matrices between consecutive layers and the removal of nonlinearities, was selected as the GCN model.

B. Two-stage GCN Training and Loss Functions

In a re-ranking scenario, the embeddings learned through a classification loss can fail to encode the intrinsic relationship between images. Therefore, we extend the training approach from [12] by adding a triplet-based loss function, alongside the classification focused *Negative Log-Likelihood (NLL)* loss. The configuration of our GCN training procedure can be defined in three main steps:

1) Soft-labels creation: As described in Section IV-B3, we explore the hypergraph structure to construct a set of reliable clusters $\mathcal{S}$, which contains c clusters and classifies q images from the dataset into soft-labels that can be used for semi-supervised training. For this work, we set $p = 0.5$ as default, resulting in half of the dataset being classified into soft-labels.

2) Triplet creation: The groups created within the soft-label clusters are reliable associations, extracted from the hypergraph manifold learning algorithm. Therefore, we can explore these groups to formulate triplet-based examples for training our GCN model. We follow the triplet pattern described in [19], where a triplet is a set (a, p, n) containing three elements: the anchor (*a*), a positive example that should be set approximated to the anchor (*p*), and a negative example that should be separated from both the anchor and the positive example (*n*).

For each created cluster, we extract a set of triplet examples by conducting three steps. First, let $\mathcal{P}_i$ be a set with all possible pairs of elements from a cluster $\mathcal{S}_i$:

$$\mathcal{P}_i = \{(o_a, o_b) \mid o_a, o_b \in \mathcal{S}_i \text{ and } a \neq b\} \tag{2}$$

Next, we need to define a negative element for each obtained pair. Therefore, we define a set of possible negative candidates, $\mathcal{PM}_i$ as the difference between our image collection $\mathcal{C}$ and the cluster $\mathcal{S}_i$:

$$\mathcal{PM}_i = \mathcal{C} \setminus \mathcal{S}_i \tag{3}$$

From this set of possible negative examples, we randomly extract j elements, where $|\mathcal{P}_i| = j$:

$$\mathcal{M}_i = \{o_1, o_2, \ldots, o_j\}, \quad o_i \in_R PM_i, \quad |\mathcal{M}_i| = j \tag{4}$$

After obtaining both positive and negative examples, we can combine $\mathcal{P}_i$ and $\mathcal{M}_i$ to create a set of triplets $\mathcal{J}_i$, which contains all training examples for the cluster $\mathcal{S}_i$:

$$\mathcal{J}_i = \{(o_a, o_b, o_c) \mid o_a, o_b \in \mathcal{P}_i, o_c \in \mathcal{M}_i\} \tag{5}$$

Finally, by executing these steps for all clusters obtained from our algorithm, we can define a set of all triplets available for training our GCN model. Therefore, let the function $tf()$ be the sequential application of Equations 2, 3, 4, and 5 for a given cluster, the set of training triplets $\mathcal{J}$ can be defined as:

$$\mathcal{J} = \bigcup_{\mathcal{S}_i \in \mathcal{S}} tf(\mathcal{S}_i) \tag{6}$$

C. Two-Stage Training

As mentioned in this section, we extend the training procedure from [12] by adding a second stage using a triplet-based loss. In this scenario, our GCN model is trained with the NLL loss and the Triplet Margin Loss [19] in a semi-supervised approach, using the soft-label and the triplet examples, respectively.

Therefore, we define a custom loss function that combines both approaches:

$$\mathcal{L} = \alpha \times \mathcal{L}_{NLL}(\mathcal{X}, \mathcal{S}) + (1 - \alpha) \times \mathcal{L}_{TRIPLET}(\mathcal{X}, \mathcal{J}), \tag{7}$$

where $\mathcal{Z}$ is the set of feature embeddings obtained from the GCN model, $\mathcal{S}$ is the set of soft-label clusters, $\mathcal{J}$ is the set of triplet examples, and $\alpha \in 0, 1$ defines which loss is being used in the current training epoch.

Finally, our training is conducted by running a defined number of epochs using $\alpha = 1$, training the model in the classification task, and afterward running another defined number of epochs with $\alpha = 0$, fine-tuning the model for the retrieval task.

D. Image Re-Ranking based on GCN Embeddings

After training, a final inference is executed for the complete collection, retrieving a set of embeddings $\mathcal{Z}$:

$$\mathcal{Z} = \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_n \mid \mathbf{z}_i \in \mathbb{R}^c\} \tag{8}$$

Based on the extracted set of embeddings, let $\rho(o_i, o_j)$ represent the cosine similarity calculation between images i and j, being defined as:

$$\rho(o_i, o_j) = \frac{\mathbf{z}_i \cdot \mathbf{z}_j}{\|\mathbf{z}_i\|_2 \, \|\mathbf{z}_j\|_2} \tag{9}$$

By computing the cosine similarity between all elements from the image collection, we can retrieve an improved set of ranked lists $\mathcal{T}_f$ which are used to perform the retrieval operations.

VI. Experimental Evaluation

A. Datasets and Experimental Protocol

In the experimental analysis, we considered four diverse image datasets. Three of them are general-purpose datasets containing from 17 to 200 classes: (*i*) Flowers [20], 1360 images, 17 classes; (*ii*) Corel5k [21], 5000 images, 50 classes; and (*iii*) CUB200 [22], 11788 images, 200 classes. The last one is a person re-identification (Re-ID) dataset, which encompasses the challenge of identifying the same individual in different camera views: (*iv*) CUHK03 [23], 14,097 images, 1,467 individuals (classes). The Re-ID protocol and the detected dataset version [23] were used.

For the general-purpose datasets, our experiments used three feature vector sets extracted from a ResNet CNN [16], a ViT [17], and a Swin [18] Transformers networks, all pre-trained on the ImageNet dataset. For Re-ID, since it is a different domain, ResNet [16] and two OSNet [24] variants were used, both trained on the MSMT17 dataset. The input ranked lists used in LHRR and the *Original* metrics were obtained using the Euclidean distance in the input features. The k-NN graph used during GCN training was constructed based on the final ranked lists obtained from LHRR.

TABLE I: Mean Average Precision (MAP) results on the Flowers dataset for k = 80.

Method	Features					
	RESNET [16]	**Rel. Gain**	**VIT-B16 [17]**	**Rel. Gain**	**SWIN-TF [18]**	**Rel. Gain**
Original	49.32%	-	86.98%	-	92.59%	-
RFE [4]	71.73%	+45.44%	**97.16%**	**+11.70%**	99.19%	+07.13%
RDPAC [6]	72.37%	+46.73%	92.21%	+06.01%	97.31%	+05.09%
BFSTREE [7]	64.87%	+31.52%	90.61%	+04.17%	95.74%	+03.40%
LHRR [8]	70.81%	+43.57%	96.19%	+10.58%	99.39%	+07.34%
CPRR [9]	63.64%	+29.03%	90.93%	+04.54%	95.75%	+03.41%
Ours	**73.10 ± 00.23**	**+48.19%**	96.84% ± 00.09	+11.33%	**99.62% ± 00.002**	**+07.59%**

TABLE II: Mean Average Precision (MAP) results on the Corel5k dataset for k = 80.

Method	Features					
	RESNET [16]	**Rel. Gain**	**VIT-B16 [17]**	**Rel. Gain**	**SWIN-TF [18]**	**Rel. Gain**
Original	62.93%	-	73.76%	-	72.93%	-
RFE [4]	86.08%	+36.78%	91.11%	+23.52%	94.63%	+29.75%
RDPAC [6]	79.73%	+26.69%	86.07%	+16.68%	84.20%	+15.45%
BFSTREE [7]	75.10%	+19.33%	82.41%	+11.97%	80.27%	+10.06%
LHRR [8]	86.85%	+38.01%	**91.40%**	**+23.91%**	95.93%	+31.53%
CPRR [9]	76.07%	+20.88%	83.05%	+12.59%	80.58%	+10.48%
Ours	**89.05% ± 00.12**	**+41.50%**	89.76% ± 00.07	+21.69%	**97.60% ± 00.34**	**+33.82%**

TABLE III: Mean Average Precision (MAP) results on the CUB200 dataset for k = 50.

Method	Features					
	RESNET [16]	**Rel. Gain**	**VIT-B16 [17]**	**Rel. Gain**	**SWIN-TF [18]**	**Rel. Gain**
Original	20.55%	-	59.00%	-	56.54%	-
RFE [4]	34.20%	+66.42%	66.37%	+12.49%	66.24%	+17.15%
RDPAC [6]	30.45%	+48.17%	68.07%	+15.37%	70.09%	+23.96%
BFSTREE [7]	27.30%	+32.86%	65.78%	+11.49%	66.31%	+17.27%
LHRR [8]	34.88%	+69.73%	69.64%	+18.03%	70.73%	+25.09%
CPRR [9]	28.37%	+38.05%	66.31%	+12.38%	67.31%	+19.04%
Ours	**38.44% ± 00.11**	**+87.05%**	**70.09% ± 00.06**	**+18.79%**	**76.61% ± 00.11**	**+35.49%**

TABLE IV: Mean Average Precision (MAP) results on the CUHK03 dataset, following the person Re-ID protocol, for k = 10.

Method	Features					
	RESNET	**Rel. Gain**	**OSNET-IBN**	**Rel. Gain**	**OSNET-AIN**	**Rel. Gain**
Original	13.08%	-	20.78%	-	27.00%	-
RFE [4]	16.88%	+29.05%	28.11%	+35.27%	35.74%	+32.37%
RDPAC [6]	19.03%	+45.49%	**30.30%**	**+45.81%**	**37.39%**	**+38.48%**
BFSTREE [7]	16.68%	+27.52%	26.86%	+29.26%	34.21%	+26.70%
LHRR [8]	16.64%	+27.22%	29.03%	+39.70%	34.69%	+28.52%
CPRR [9]	16.05%	+22.71%	26.22%	+26.18%	32.76%	+21.37%
Ours	**19.57% ± 00.07**	**+49.62%**	28.74% ± 00.17	+38.31%	31.81% ± 00.08	+17.81%

B. Implementation Details

Regarding method parameters, we follow the best results from [12], setting $T = 2$, $p = 0.5$ and the number of classes and clusters, c, as the number of real classes for all datasets. Additionally, L is defined as $4 * k$ with a minimum default value of 100, in order to better extract relationships when exploring smaller neighborhoods, e.g. $k <= 25$. Moreover, a value of k was selected, and used in all methods, for each dataset. We used $k = 80$ for the Flowers and Corel5k datasets, $k = 50$ for the CUB200 dataset, and $k = 10$ for CUHK03.

For the GCN training, we used 32 hidden layers for all experiments and a learning rate of 10^-3, using ADAM as our optimizer. Furthermore, we executed 300 epochs of training in the classification task ($\alpha = 1$) and other 300 epochs in the retrieval task ($\alpha = 0$). All experiments for our proposed method were executed 10 times, with averages and standard deviations being reported.

C. Retrieval Results

In our retrieval experiments, we compared the proposed SGRR with recent ranking and diffusion-based manifold learning algorithms: CPRR [9] (2018), LHRR [8] (2019), RDPAC [6] (2021), BFSTREE [7] (2021), and RFE [4] (2023).

Table I presents the results on the Flowers dataset. The proposed approach obtained the best results in two of the three explored features. In a similar result, for the Corel5K dataset, presented in Table II, SGRR obtained the best values in both RESNET and Swin-TF features. In the CUB200 dataset, presented in Table III, our approach obtained the best results in all features. Additionally, on relative gain, the proposed algorithm obtained a strong 87.05% increase in MAP performance for the RESNET feature. As we can observe, among the three general retrieval datasets, the proposed method achieved the best results for all features on CUB200, which is the biggest and most challenging dataset.

Table IV presents the results of the Re-ID experiment on the CUHK03 dataset. Our method obtained the best result for the RESNET feature while presenting competing results on the other two features. Additionally, it is interesting to see that, not only the LHRR results were improved by our approach in almost all scenarios, but also we achieved the highest improvement for the RESNET features in all datasets. This is an indication that our proposed framework is capable of improving re-ranking results while also being more reliable when handling noisy data, obtained from weaker feature extractors.

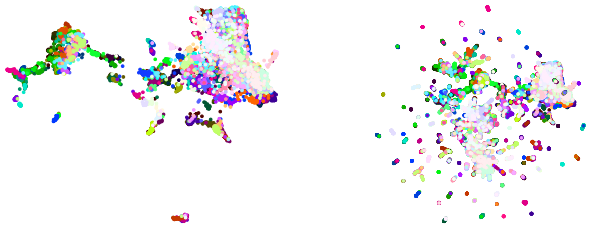
D. Visual Results

As one of the most important contributions from this work, SGRR is capable of generating embedding vectors with fewer dimensions, while improving performance in retrieval tasks. For instance, the embedding vector for the Flowers dataset was reduced from 2048 dimensions (from the RESNET output) to only 17 (the number of classes in the dataset) while the MAP performance was improved in 48.19%.

The results obtained in VI-C, demonstrate that the manifold learning processes, combined with our GCN training approach, were able to extract better representations with a lower number of dimensions.

To illustrate this improvement in separability, we apply UMAP [25], using the default parameters, to reduce both the input and SGRR features to two dimensions, allowing the creation of a visualization image.

Figure 2 presents the results comparing RESNET and SGRR features for the CUB200 dataset. The SGRR features broke the two main clusters observed on the RESNET image, creating smaller and more separated groups. This example resulted in an impressive 87.05% increase in MAP.



(a) RESNET features (b) SGRR features

Fig. 2: Feature visualization for RESNET and SGRR.

VII. Conclusion

This paper proposes a novel re-ranking algorithm by exploring hypergraph-based manifold learning, clustering, and GCN models to generate a new set of image representations. Additionally, we propose a novel two-stage approach for training GCNs, combining semi-supervised losses for both classification and retrieval tasks. The results in retrieval experiments show that our approach can better encode similarity information based on query images in most considered scenarios. The computed embeddings are also superior to the hypergraph used in isolation (LHRR results), indicating that the hypergraph structures can be further explored by GCNs.

Additionally, our novel approach is capable of generating new embedding representations for retrieval with lower dimensions, drastically reducing storage space while improving effectiveness. In future work, we intend to explore more complex training approaches, triplet creation methods based on the hypergraph manifold, and an extension of this work to function on inference with new query images, allowing us to classify and retrieve similar data for never-seen images.

Acknowledgment

The authors are grateful to São Paulo Research Foundation - FAPESP (grant #2018/15597-6), Brazilian National Council for Scientific and Technological Development - CNPq (grants #313193/2023-1, and #422667/2021-8), and Petrobras (grant #2023/00095-3) for financial support.

References

[1] W. Chen, Y. Liu, W. Wang, E. M. Bakker, T. Georgiou, P. W. Fieguth, L. Liu, and M. S. Lew, "Deep image retrieval: A survey," *CoRR*, vol. abs/2101.11282, 2021.

[2] J. Jiang, B. Wang, and Z. Tu, "Unsupervised metric learning by self-smoothing operator," in *ICCV*, 2011, pp. 794–801.

[3] S. Bai, X. Bai, Q. Tian, and L. J. Latecki, "Regularized diffusion process on bidirectional context for object retrieval," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 5, pp. 1213–1226, 2019.

[4] L. P. Valem, D. C. G. Pedronette, and L. J. Latecki, "Rank flow embedding for unsupervised and semi-supervised manifold learning," *IEEE Transactions on Image Processing*, vol. 32, pp. 2811–2826, 2023.

[5] F. Yang, R. Hinami, Y. Matsui, S. Ly, and S. Satoh, "Efficient image retrieval via decoupling diffusion into online and offline processing," in *Conference on Artificial Intelligence, AAAI 2019*. AAAI Press, 2019, pp. 9087–9094.

[6] D. C. G. Pedronette, L. P. Valem, and L. J. Latecki, "Efficient rank-based diffusion process with assured convergence," *Journal of Imaging*, vol. 7, no. 3, 2021. [Online]. Available: https://www.mdpi.com/2313-433X/7/3/49

[7] D. C. G. Pedronette, L. P. Valem, and R. da S. Torres, "A bfs-tree of ranking references for unsupervised manifold learning," *Pattern Recognition*, vol. 111, p. 107666, 2021.

[8] D. C. G. Pedronette, L. P. Valem, J. Almeida, and R. da S. Torres, "Multimedia retrieval through unsupervised hypergraph-based manifold ranking," *IEEE Transactions on Image Processing*, vol. 28, no. 12, pp. 5824–5838, 2019.

[9] L. P. Valem, D. C. G. Pedronette, and J. Almeida, "Unsupervised similarity learning through cartesian product of ranking references," *Pattern Recognition Letters*, vol. 114, pp. 41–52, 2018.

[10] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.

[11] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and P. S. Yu, "A comprehensive survey on graph neural networks," *IEEE Trans. on Neural Networks and Learning Systems*, vol. 32, no. 1, pp. 4–24, 2021.

[12] L. T. Lopes and D. C. G. a. Pedronette, "Self-supervised clustering based on manifold learning and graph convolutional networks," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, January 2023, pp. 5634–5643.

[13] H. Cai, V. W. Zheng, and K. C. Chang, "A comprehensive survey of graph embedding: Problems, techniques, and applications," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 9, pp. 1616–1637, 2018.

[14] D. Pedronette and L. J. Latecki, "Rank-based self-training for graph convolutional networks," *Information Processing & Management*, vol. 58, p. 102443, 03 2021.

[15] F. Wu, A. Souza, T. Zhang, C. Fifty, T. Yu, and K. Weinberger, "Simplifying graph convolutional networks," in *International Conference on Machine Learning (ICML)*, vol. 97, 2019, pp. 6861–6871.

[16] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.

[17] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations*, 2021. [Online]. Available: https://openreview.net/forum?id=YicbFdNTTy

[18] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," *CoRR*, vol. abs/2103.14030, 2021.

[19] V. Balntas, E. Riba, D. Ponsa, and K. Mikolajczyk, "Learning local feature descriptors with triplets and shallow convolutional neural networks," 01 2016, pp. 119.1–119.11.

[20] M.-E. Nilsback and A. Zisserman, "A visual vocabulary for flower classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, 2006, pp. 1447–1454.

[21] G.-H. Liu and J.-Y. Yang, "Content-based image retrieval using color difference histogram," *Pattern Recognition*, vol. 46, no. 1, pp. 188 – 198, 2013.

[22] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, "The Caltech-UCSD Birds-200-2011 Dataset," California Institute of Technology, Tech. Rep. CNS-TR-2011-001, 2011.

[23] Z. Zhong, L. Zheng, D. Cao, and S. Li, "Re-ranking person re-identification with k-reciprocal encoding," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, July 2017, pp. 3652–3661.

[24] K. Zhou, Y. Yang, A. Cavallaro, and T. Xiang, "Learning generalisable omni-scale representations for person re-identification," *IEEE transactions on pattern analysis and machine intelligence*, vol. PP, 03 2021.

[25] L. McInnes, J. Healy, and J. Melville, "Umap: Uniform manifold approximation and projection for dimension reduction," 2020. [Online]. Available: https://arxiv.org/abs/1802.03426