

Graph Convolutional Networks based on Manifold Learning for Semi-Supervised Image Classification

Lucas Pascotti Valem^a, Daniel Carlos Guimarães Pedronette^a, Longin Jan Latecki^b

^aDepartment of Statistics, Applied Mathematics and Computing, State University of São Paulo (UNESP), Av. 24-A, 1515, Rio Claro, SP 13506-900, Brazil

^bDepartment of Computer and Information Sciences, Temple University, North 12th Street, 1925, Philadelphia, PA 19122, USA

ABSTRACT

Due to a huge volume of information in many domains, the need for classification methods is imperious. In spite of many advances, most of the approaches require a large amount of labeled data, which is often not available, due to costs and difficulties of manual labeling processes. In this scenario, unsupervised and semi-supervised approaches have been gaining increasing attention. The GCNs (Graph Convolutional Neural Networks) represent a promising solution since they encode the neighborhood information and have achieved state-of-the-art results on scenarios with limited labeled data. However, since GCNs require graph-structured data, their use for semi-supervised image classification is still scarce in the literature. In this work, we propose a novel approach, the Manifold-GCN, based on GCNs for semi-supervised image classification. The main hypothesis of this paper is that the use of manifold learning to model the graph structure can further improve the GCN classification. To the best of our knowledge, this is the first framework that allows the combination of GCNs with different types of manifold learning approaches for image classification. All manifold learning algorithms employed are completely unsupervised, which is especially useful for scenarios where the availability of labeled data is a concern. A broad experimental evaluation was conducted considering 5 GCN models, 3 manifold learning approaches, 3 image datasets, and 5 deep features. The results reveal that our approach presents better accuracy than traditional and recent state-of-the-art methods with very efficient run times for both training and testing.

© 2023 Elsevier Ltd. All rights reserved.

1. Introduction

Over the last years, the fast development of data acquisition technologies and the huge growth of multimedia collections (e.g. image, video, music, and others) have made the use of classification systems indispensable (Datta et al., 2008). There is a wide range of different applications, including person re-identification (Karanam et al., 2019), diagnosis of diseases (Agarwal and Mostafa, 2011), facial recognition (Sultana and Gavrilova, 2013), remote sensing (Wang and Song, 2013), object identification (Schober et al., 2004), and various others. However, despite the significant recent advances in feature extraction methods, effectively retrieving multimedia data still remains a challenge in various scenarios. Such complexity is mainly associated to the diverse aspects involved in human visual perception, which usually can not be encoded by a single visual feature (Kherfi and Ziou, 2006; Piras and Giacinto, 2017).

Due to the huge success of deep learning, especially based on Convolutional Neural Networks (CNNs), multiple models applied to both image and video content, have been proposed (Chen et al., 2017; Deng, 2014; He et al., 2016; Sandler et al., 2018). Despite the remarkable results mainly supported by CNN models (Chen et al., 2017; Deng, 2014; He et al., 2016; Krizhevsky et al., 2012; Ye et al., 2020), most methods demand a high amount of data to be trained (Sener and Savarese, 2018; Ye et al., 2020). The availability of supervised training data is often a challenge due to the need for labeling a lot of information, which is expensive and time-consuming (Hino, 2020). With the objective of easing up the process of labeling data and not requiring manual intervention for this task, there are researches that propose to assist the process of labeling with automatic stages (Sener and Savarese, 2018). However, in spite of these possibilities, the process of obtaining labeled data remains a challenge for multiple tasks (Hino, 2020), since the

amount of multimedia data available increases much faster than the amount of labeled data that can be obtained for it (Li et al., 2014).

In this scenario, various learning paradigms have been attracting increasing attention in order to deal with the scarcity of labeled data. Unsupervised (Guimarães Pedronette et al., 2021; Li et al., 2019; Pedronette et al., 2019; Valem et al., 2017) and semi-supervised (Kipf and Welling, 2017; Mandal et al., 2020; Wang et al., 2020) strategies often offer attractive solutions. While the unsupervised approaches require no labeled data at all, the semi-supervised ones require a small set of labeled data. A recent trend is given by weakly supervised learning (Zhou, 2017), which is a broad taxonomy that covers different strategies often divided into three main categories: (i) incomplete supervision: where only a subset of the training data is labeled, a part of the other subset can be labeled considering active learning or semi-supervised learning; (ii) inexact supervision: the available labels are used to create rules and constraints (heuristics) on the training data; and (iii) inaccurate supervision: there are wrong or low-quality labels and the idea is to identify the potential mislabeled instances and to correct or remove them.

In general, incomplete supervision often depicts scenarios very close to real-world applications, modeled by Semi-Supervised Learning (SSL) which considers reliable but limited labeled data. Over the last years, SSL approaches have also witnessed huge advances, mainly supported by Graph Convolutional Networks (GCNs) (Kipf and Welling, 2017). Different from traditional CNN models, which generally operate through convolutions in the Euclidean space, the GCN models allow convolution operations in non-Euclidean domains defined by graph-based structures (Kipf and Welling, 2017; Wu et al., 2021). Although very effective, CNN models often ignore contextual information such as neighborhood references and the relationship between the elements in the dataset. Furthermore, CNNs are often applied to 2D and 3D data (e.g., images and videos) and are generally not easily applicable to 1D feature vectors, unless some data processing or conversion is done in the original data (Sharma et al., 2019).

The GCNs exploit multidimensional feature vectors and graph-based neighborhood structures to learn more effective representations. Due to these aspects, the GCNs have been recently applied for graph-based data on semi-supervised learning tasks, achieving state-of-the-art results. Several GCN variations have been proposed with relevant results (Bianchi et al., 2021; Klicpera et al., 2019; Veličković et al., 2018; Wu et al., 2019). The use of GCN has many different applications. There are some recent works that exploit graph learning for question and answer systems (Nie et al., 2020), including conversational image search (Nie et al., 2021).

While it offers an effective contextual representation learning strategy, GCN models require graph-structured data. The graph data is inherently available in some domains, but needs to be inferred or constructed in others (Franceschi et al., 2019). Consequently, several methods have been proposed for graph-structured data as citation datasets (Bai et al., 2019; Bianchi et al., 2021; Chen et al., 2020a; Klicpera et al., 2019; Li et al., 2018; Veličković et al., 2018; Wu et al., 2019), but only a

few approaches have been proposed for image and multimedia data (Chaudhuri et al., 2019; Liu et al., 2019; Yang et al., 2020; Zhang et al., 2021). In most cases, the most direct approach is to create a k -nearest neighbor graph. However, the GCN models are sensitive to the input graph, in the sense that a more effective classification depends on the edges between nodes of the same class.

In this paper, we propose a novel GCN-based approach, the Manifold-GCN, for image classification in semi-supervised scenarios, with limited labeled data. Deep features are extracted for image representation employing transfer learning by CNNs and Vision Transformers (ViT) models. Ranking structures are computed and used as input by unsupervised manifold learning algorithms based on these extracted features. In general, manifold Learning approaches aim to capture and exploit the intrinsic manifold structure to compute a more effective distance/similarity measure (Jiang et al., 2011). In this work, we consider recent unsupervised manifold learning methods to provide more effective similarity measures using rank-based formulations.

The manifold learning methods produce more effective ranking results, i.e., improved neighbor sets, which are exploited for building the input graph of the GCN model. In addition to constructing kNN graphs, the use of reciprocal kNN graphs is proposed. The main hypothesis of the paper is that the use of manifold learning to improve the graph structure provided as the input of the Graph Convolutional Network (GCN) can further improve the classification results obtained. This work proposes and validates this hypothesis on different manifold learning and recent GCN approaches.

We can highlight the main contributions of our work as follows: (i) novel ways to learn the graph structures that improve GCN image classification; (ii) the use of reciprocal kNN graph in order to provide a more reliable graph for GCNs. There are very few works that employ kNN graphs (Franceschi et al., 2019) or manifold learning (Chen et al., 2020a) for GCNs. In (Franceschi et al., 2019) the traditional kNN graph is employed and (Chen et al., 2020a) uses manifold learning, but in both works no image data is considered. Other few works have recently employed GCN models on image classification (Chaudhuri et al., 2019; Liu et al., 2019; Yang et al., 2020; Zhang et al., 2021). However, to the best of our knowledge, this is the first work that exploits both manifold learning and reciprocal kNN graphs for GCN-based semi-supervised image classification. In addition, it combines powerful contextual modeling given by GCN models with effective representations given by CNNs and ViT features.

There are many applications of the proposed approach. The improvement of classification results using GCNs may benefit many different areas, especially when there is limited labeled data. For example: person re-identification (Karanam et al., 2019) and diagnosis of diseases (Agarwal and Mostafa, 2011). The Manifold-GCN can be employed in scenarios where the graph data is not previously available by building the graph from the features and employing manifold learning.

A wide experimental evaluation was conducted in order to assess the effectiveness of the proposed approach. The experi-

mental results were obtained on 3 public datasets. We evaluated the impact of different GCN models combined with different manifold learning methods. The experimental results demonstrate the effectiveness of the proposed approach and the gains on combining manifold learning and reciprocal kNN graphs.

This paper is organized as follows. Section 2 presents the related work, while Section 3 presents the formal definition of semi-supervised learning. Section 4 describes our proposed approach, the ManifoldGCN. Section 5 presents the GCNs and manifold learning methods considered. Section 6 reports the experimental evaluation. Finally, Section 7 states conclusions and considers possible future works.

2. Related Work

This section presents an overview of the methods proposed for semi-supervised image classification over recent years and their main ideas, especially regarding deep learning.

Semi-supervised approaches perform training considering both labeled and unlabeled data, which is advantageous in multiple scenarios where there is little labeled data (van Engelen and Hoos, 2019). Some of them rely on the generation of pseudo-labels (Vanyan and Khachatrian, 2021). Among the traditional methods for generating pseudo-labels, we can cite: Label Spreading (Zhou et al., 2004) and Pseudo-label (Lee, 2013). There are also several supervised approaches that later presented semi-supervised variants that do not require the generation of pseudo-labels. For example: Support Vector Machines (Cortes and Vapnik, 1995) (SVM) and Optimum Path Forest (Amorim et al., 2014) (OPF).

The taxonomy and categories of semi-supervised approaches vary in the literature (van Engelen and Hoos, 2019; Vanyan and Khachatrian, 2021). Generally, there is some overlap among categories. In the following subsections, we present them according to 4 research directions (Vanyan and Khachatrian, 2021): category regularization; stronger augmentation; convergence with self-supervised learning; and graph-based approaches.

2.1. Consistency regularization

These methods rely on a concept known as category regularization. The central idea is to force the approach to produce similar results for augmented versions of the same unlabeled image. This is generally done by considering an additional term in the loss function. The first method as far as it is known, to use this concept is called II-Model (Laine and Aila, 2017). In II-Model, they use translation and random horizontal flips as augmentations for unlabeled data, which is often called weak augmentation.

However, the main issue with II-Model is the unstable target, which compromises the algorithm learning procedure. The Mean Teacher (Tarvainen and Valpola, 2017) approach was proposed with the intent to address this issue. For this, they use two separate models: the Student network and the Teacher network. While the Student is trained as usual, the Teacher does not use back-propagation and the weights are updated at each iteration using the weights from the Student network.

2.2. Stronger Augmentation

Data augmentation is of crucial importance for various semi-supervised approaches (Vanyan and Khachatrian, 2021). Some strategies focus on improving the performance of classification by employing different kinds of data augmentation techniques, in such a way that the inputs given to the two branches of the neural model (or, to the two separate networks) are sufficiently distinct. There are many methods that fit in this category, among them: Virtual Adversarial Training and Entropy Minimisation (Miyato et al., 2019) (VAT), Unsupervised Data Augmentation (Xie et al., 2020) (UDA), MixMatch (Berthelot et al., 2019), FixMatch (Sohn et al., 2020), ReMixMatch (Berthelot et al., 2020), AlphaMatch (Gong et al., 2021). Some of them also mix other ideas, such as the concept of consistency regularization.

2.3. Convergence with Self-supervised Learning

Recently, self-supervision has been used by several semi-supervised methods. Self-supervised approaches are a category of representation learning algorithms capable of generating supervision signals without any human annotations. Most approaches in this category use self-supervision to generate a set of pseudo-labels for training. Among the main approaches in this category, we can cite: SimCLR (Chen et al., 2020b), CoMatch (Li et al., 2021), Self-Match (Kim et al., 2021).

2.4. Graph-based Approaches

A promising research direction is methods based on graphs. There are different traditional graph-based approaches, both transductive, and inductive ones (van Engelen and Hoos, 2019). The idea is that the elements of the dataset can be represented as nodes and the edges can be used to propagate or represent some kind of information between these nodes. Graph-based methods are usually based on the manifold assumption (van Engelen and Hoos, 2019): the graphs, constructed based on the local similarity between data points, provide a lower-dimensional representation of the potentially high-dimensional input data. This makes these approaches advantageous for scenarios with data of high dimensionality.

Recently, Graph Convolutional Networks (GCN), have been proposed for semi-supervision. While CNNs are specially built to operate on regular (Euclidean) structured data, the GNNs work on graphs with different numbers of vertexes and unordered nodes (irregular on non-Euclidean structured data). There are many variants of GCNs proposed: GCN-Net (Kipf and Welling, 2017), GCN-SGC (Wu et al., 2019), GCN-GAT (Veličković et al., 2018), GCN-APPNP (Klicpera et al., 2019), GCN-ARMA (Bianchi et al., 2021). Also, variants of GNNs: GNN-LDS (Franceschi et al., 2019), GNN-KNN-LDS (Franceschi et al., 2019).

The GCNs exploit feature vectors and graph-based neighborhood structures to learn more effective representations. Due to these aspects, the GCNs have been recently applied to graph-based data on semi-supervised learning tasks, achieving state-of-the-art results. Several GCN variations have been proposed with relevant results (Bianchi et al., 2021; Klicpera et al., 2019; Veličković et al., 2018; Wu et al., 2019). The use of GCN has

many different applications. There are some recent works that exploit graph learning for question and answer systems (Nie et al., 2020), including conversational image search (Nie et al., 2021).

However, there are still not many approaches for using GCNs in image classification. Among the multiple research topics, there is finding the best approach to model the graph and the features, which are provided as the input for these networks and directly impact their performance and results.

3. Graph-based Semi-Supervised Learning Formulation

In this section, we first discuss a formal definition of the semi-supervised learning setting for classification tasks using GCNs, mostly following the notation from (Kipf and Welling, 2017; Pedronette and Latecki, 2021).

Let $\mathcal{C}=\{o_1, o_2, \dots, o_n\}$ be an object collection, where $o_i \in \mathcal{C}$ denotes an image and n denotes the collection size. The collection is represented by an undirected graph \mathcal{G} . The graph can be formally defined as tuple $\mathcal{G} = (\mathcal{V}, \mathbf{X}, \mathcal{E})$, where \mathcal{V} denotes the node set, \mathbf{X} is a feature matrix, and \mathcal{E} denotes the edge set.

The node set is defined by $\mathcal{V} = \{v_1, v_2, \dots, v_n\}$ where each node $v_i \in \mathcal{V}$ represents an image $o_i \in \mathcal{C}$. Labels can be assigned to nodes $v_i \in \mathcal{V}$, such that a set of labels can be defined as $\mathcal{Y} = \{y_1, y_2, \dots, y_c\}$. According to the labels, the node set can be more specifically defined as $\mathcal{V} = \{v_1, v_2, \dots, v_L, v_{L+1}, \dots, v_n\}$, which denotes a partially labeled data set, where $\mathcal{V}_L = \{v_i\}_{i=1}^L$ is the labeled data items subset and $\mathcal{V}_U = \{v_i\}_{i=L+1}^n$ is the unlabeled data items subset. Formally, the training set can be seen as a labeling function $f_l: \mathcal{V}_L \rightarrow \mathcal{Y}$, where $y_i = f_l(v_i) \forall v_i \in \mathcal{V}_L$. In general, on semi-supervised scenarios, we have $|\mathcal{V}_L| \ll |\mathcal{V}_U|$.

The feature matrix can be defined as $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n]^T \in \mathbb{R}^{n \times d}$, where \mathbf{x}_i is a d -dimensional feature vector which represents the image o_i , or equivalently, the node v_i . The vector \mathbf{x}_i is obtained by a feature extraction approach, which can be defined as function $f_e: \mathcal{C} \rightarrow \mathbb{R}^d$, such that $\mathbf{x}_i = f_e(o_i)$.

The edge set \mathcal{E} is a set of nodes pairs (v_i, v_j) , formally defined as $\mathcal{E} \subseteq \{(v_i, v_j) | (v_i, v_j) \in \mathcal{V}^2 \wedge v_i \neq v_j\}$. For graph-structured content, the set \mathcal{E} is intrinsically defined by the data. For general image data, we propose to define the set \mathcal{E} based on the feature matrix \mathbf{X} . How to define an effective graph is a central challenge addressed by our approach, discussed in the next section.

Once defined the graph \mathcal{G} , a GCN model denoted by a function f_{gcn} can be used to learn an embedded representation \mathbf{z}_i for each node v_i . The learned representation is exploited to perform classification tasks. Formally, the classification goal is to learn a function $\hat{f}_l: \mathcal{V}_U \rightarrow \mathcal{Y}$ to predict the labels of unlabeled nodes in \mathcal{V}_U .

4. Manifold-based Graph Convolutional Network

In this work, we propose the Manifold-based Graph Convolutional Network (Manifold-GCN), a semi-supervised framework based on the use of manifold learning and GCN models for image classification for scenarios with limited labeled

data. The initial representations were obtained by deep features extracted by CNN and ViT models trained on a transfer learning setting. Given the representations, the central idea consists in exploiting contextual similarity measures given by unsupervised manifold learning methods for computing a graph. The similarity information encoded in the graph is exploited by GCN models for learning novel representations used for classification.

Figure 1 illustrates the main steps that compose our strategy. Each step is identified by a number (top of boxes) and a function (bottom of boxes). In (1), a feature vector is extracted for representing each image. In (2), representations are processed in order to obtain ranked lists, which encode the similarity information. Unsupervised manifold learning methods are used to analyze contextual similarity information and compute more effective rankings in (3). In (4), the outputs of the manifold learning methods are modeled as kNN graphs or reciprocal kNN graphs. In (5), the graph and features are jointly provided to the GCN models for semi-supervised training. The embeddings obtained for each of the elements of the dataset can be used for classification, through a softmax operation. Each of the main steps of the framework is described in the next subsections.

4.1. Similarity Measurement and Ranking Model

In the proposed approach, the similarity information is encoded on ranking structures. Let us consider a ranking task in which, given a query image, an ordered list of images from the collection is returned according to the similarity to the query. Formally, given a query image o_q , a ranked list $\tau_q=(o_1, o_2, \dots, o_L)$ in response to the query, where L denotes the length of the list. The ranked list τ_q can be defined as a permutation of a set C_L which contains the L most similar images to image x_q in the collection \mathcal{C} . The permutation τ_q is a bijection from the set C_L onto the set $[L] = \{1, 2, \dots, L\}$. The $\tau_q(o_i)$ notation denotes the position (or rank) of image o_i in the ranked list τ_q .

The ranked list τ_q can be computed based on the comparison between image representations. Let $d: \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$ be a distance function that computes the distance between two images according to their corresponding feature vectors. The Euclidean distance is often used as the distance function. Formally, the distance between two images o_i, o_j is defined by $d(\mathbf{x}_i, \mathbf{x}_j)$.

For a given query, a ranked list can be obtained by sorting images in increasing order of the distance. In terms of ranking positions, we can say that if image o_i is ranked before image o_j in the ranked list of image o_q , that is, $\tau_q(o_i) < \tau_q(o_j)$, then $d(\mathbf{x}_q, \mathbf{x}_i) \leq d(\mathbf{x}_q, \mathbf{x}_j)$. Taking every image in the collection as a query image x_q , a set of ranked lists $\mathcal{T} = \{\tau_1, \tau_2, \dots, \tau_n\}$ can be obtained. In this way, the set \mathcal{T} can be obtained from the feature matrix \mathbf{X} and the ranking task defined by a function f_r , such that $\mathcal{T} = f_r(\mathbf{X})$. Tree-based indexing structures (Omohundro, 1989) and hashing approaches (Gionis et al., 1999) can be exploited in order to provide efficient implementations for the function f_r . In this work, we consider BallTree (Omohundro, 1989; Pedregosa et al., 2011) structures.

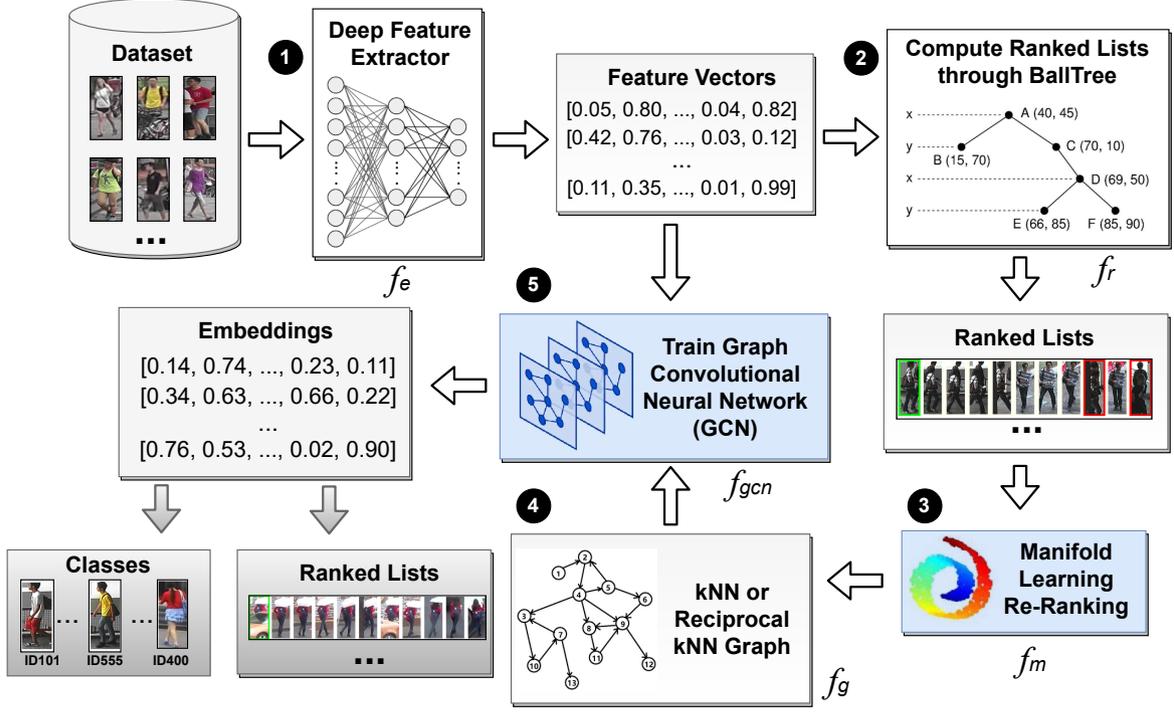


Fig. 1: Workflow of our proposed Manifold-GCN framework for image classification. The steps of the approach are numbered.

4.2. Unsupervised Manifold Learning

How to accurately define distance or similarity among data elements is a challenging and fundamental step in many machine learning tasks. The most common approach is given by pairwise comparisons based on Euclidean-like distance functions. However, pairwise analyses ignore contextual information and complex similarity arrangements encoded in the structural information of the dataset manifold. Aiming at addressing such drawbacks, many contextual similarity approaches take into account the structure of datasets in order to compute more global and effective similarity measures.

Manifold Learning is a wide term that has many different definitions in the literature. In general, manifold Learning approaches aim to capture and exploit the intrinsic manifold structure to compute a more effective distance/similarity measure (Jiang et al., 2011). Recently, unsupervised manifold learning approaches based on ranking information have achieved relevant advances in contextual similarity measurement (Guimarães Pedronette et al., 2021; Pedronette et al., 2019; Pedronette et al., 2021).

In fact, the set of ranked lists \mathcal{T} encodes rich similarity information about the image collection. The main objective of rank-based manifold learning methods is to exploit such information to capture the structure of the dataset manifold. Therefore, this step consists of the use of unsupervised manifold learning methods for processing the original ranked lists, providing more effective ranking results which are subsequently modeled as graphs to be submitted to a GCN model.

Formally, the manifold learning methods can be defined as a function f_m that receives a set of ranked lists \mathcal{T} as input and returns a set of ranked lists \mathcal{T}_m as output, which is expected to be more effective than the original:

$$\mathcal{T}_m = f_m(\mathcal{T}). \quad (1)$$

Once defined under a common formulation, three different manifold learning algorithms were considered to instantiate the proposed approach (described in Section 5).

4.3. Graph Building

The improved set of ranked lists computed by the manifold learning methods is used to build a graph. The motivation is based on the conjecture that more effective similarity information can be extracted and encoded in the graph by exploiting the processed ranked lists. Let $\mathcal{G} = (\mathcal{V}, \mathbf{X}, \mathcal{E})$ be the graph defined in Section 3. We propose to compute the edge set \mathcal{E} as a function of the set of ranked lists \mathcal{T}_m , such that $\mathcal{E} = f_g(\mathcal{T}_m)$.

This work considers two distinct approaches to define the function f_g . The similarity information encoded in the ranked lists is modeled through different neighborhood set formulations. Both approaches are discussed in the following.

- *Traditional kNN Graph*: the kNN graph is based on the natural neighborhood set. Given an element o_q , the natural neighborhood set $\mathcal{N}(o_q, k)$ contains the k most similar elements to o_q , which can be formally defined as:

$$\mathcal{N}(o_q, k) = \{X \subseteq \mathcal{C}, |X| = k \wedge \forall o_i \in X, o_j \in \mathcal{C} - X : \tau_q(o_i) < \tau_q(o_j)\}. \quad (2)$$

Therefore, the edge set \mathcal{E} of the kNN graph can be defined as:

$$\mathcal{E} = \{(o_q, o_j) \mid o_j \in \mathcal{N}(o_q, k)\}. \quad (3)$$

In other words, each element has an edge to the k most similar elements.

• *Reciprocal kNN Graph*: the reciprocal kNN graph is based on the reciprocal neighborhood set (Qin et al., 2011), which requires a stronger bidirectional similarity relationship. Different from the natural neighborhood set, which is not symmetrical, the reciprocal neighborhood set is symmetrically defined as:

$$\mathcal{N}_c(o_q, k) = \{obj_i | obj_i \in \mathcal{N}(o_q, k) \wedge o_q \in \mathcal{N}(o_i, k)\}. \quad (4)$$

The edge set E for the reciprocal kNN set can be defined as:

$$\mathcal{E} = \{(o_q, o_j) \mid o_j \in \mathcal{N}_c(o_q, k)\}. \quad (5)$$

Thus, we can interpret that there are edges between the elements o_q and o_j if they are reciprocal neighbors in the top- k positions of their ranked lists.

For both kNN and reciprocal kNN approaches, the edge set \mathcal{E} can be represented by a non-negative adjacency matrix $\mathbf{A} = [a_{ij}] \in \mathbb{R}^{n \times n}$, which can be defined as:

$$a_{ij} = \begin{cases} 1, & (o_i, o_j) \in \mathcal{E} \\ 0, & \text{otherwise.} \end{cases} \quad (6)$$

The adjacency matrix \mathbf{A} is used as input by GCN models, as discussed in next section.

4.4. Graph Convolutional Networks

Graph Convolutional Networks (GCN), originally introduced in (Kipf and Welling, 2017), aim at learning novel and more effective representations (embeddings) for each graph node. It is done by iteratively aggregating the embeddings of its neighbors, encoding the graph structure directly in a neural network model. The original model proposed in (Kipf and Welling, 2017) is a two-layer GCN model which uses the graph represented by the adjacency matrix \mathbf{A} for semi-supervised node classification.

The network model can be depicted as a function both on the feature data \mathbf{X} and on the adjacency matrix \mathbf{A} , as:

$$\mathbf{Z} = f_{gcn}(\mathbf{X}, \mathbf{A}), \quad (7)$$

where \mathbf{Z} denotes an embedding matrix, such that $\mathbf{Z} = [\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_n]^T \in \mathbb{R}^{n \times c}$ and \mathbf{z}_i is a c -dimensional embedded representation learned for the node v_i ; where n is the dataset size and c corresponds to the number of classes.

The degree matrices are computed as a pre-processing step, defined as $\hat{\mathbf{A}} = \hat{\mathbf{D}}^{-1/2} \hat{\mathbf{A}} \hat{\mathbf{D}}^{-1/2}$, where $\hat{\mathbf{A}} = \mathbf{A} + \mathbf{I}$ and $\hat{\mathbf{D}}$ is the degree matrix of $\hat{\mathbf{A}}$. Then, the function $f_{gcn}(\cdot)$ which represents the two-layer GCN model assumes the form:

$$\mathbf{Z} = f(\mathbf{X}, \mathbf{A}) = \text{softmax}(\hat{\mathbf{A}} \text{ReLU}(\hat{\mathbf{A}} \mathbf{X} \mathbf{W}^{(0)}) \mathbf{W}^{(1)}). \quad (8)$$

The matrix $\mathbf{W}^{(0)} \in \mathbb{R}^{d \times H}$ defines the neural network weights for an input-to-hidden layer with H feature maps, while $\mathbf{W}^{(1)} \in \mathbb{R}^{H \times c}$ is a hidden-to-output matrix. Both matrices $\mathbf{W}^{(0)}$ and $\mathbf{W}^{(1)}$ are trained using gradient descent, considering the cross-entropy error over all labeled nodes. $v_i \in \mathcal{V}_L$.

The activation function is applied row-wise and is defined as $\text{softmax}(z_i) = \frac{\exp(z_i)}{\sum_j \exp(z_j)}$, where z_i is the position i of embedding

\mathbf{z}_i . The softmax yields the probability distribution over the c class labels for each row, i.e., the probability values sum up to 1 for each row. Given an image o_i , the learned embedded representation \mathbf{z}_i is then used for classification tasks by applying an argmax over the output of the softmax.

5. GCNs and Manifold Learning Methods

The proposed approach is flexible in the sense that it can be instantiated by different GCN models and manifold learning methods. This section briefly describes the GCN models and the manifold learning methods considered in this work.

5.1. GCN Models

The original GCN (Kipf and Welling, 2017) model and more 4 variants (Bianchi et al., 2021; Klicpera et al., 2019; Wu et al., 2019) are used in the proposed Manifold-GCN approach. The GCN models employed are:

- **Graph Convolution Network (GCN)** (Kipf and Welling, 2017): the first GCN proposed, introducing the idea of convolutions applied to graph domains, often known as GCN-Net or simply GCN;
- **Simple Graph Convolution (SGC)** (Wu et al., 2019): a simplification of the conventional GCN models which removes the non-linearities and collapses weight matrix between consecutive layers;
- **Graph Attention Networks (GAT)** (Veličković et al., 2018): employs auto-attention layers with the idea of solving the main shortcomings of the previous GCN models. The layers are stacked in a way that it is possible to specify different weights for nodes of the same neighborhood without requiring costly operations;
- **Approximate Personalized Propagation of Neural Predictions (APPNP)** (Klicpera et al., 2019): a model that combines a GCN with the PageRank algorithm, deriving a propagation strategy based on a modified PageRank approach;
- **Auto-Regressive Moving Average (ARMA) Filter Convolutions** (Bianchi et al., 2021): a GCN variant that defines convolutional layers based on filters of Auto-Regressive Moving Average type.

5.2. Manifold Learning Methods

Manifold learning can be broadly understood as the process of non-linear dimensionality reduction by performing distance learning for a set of features. In fact, images are commonly represented as points in a high-dimensional feature space. However, it has been shown that data samples often live in a much lower dimensional intrinsic space (Jiang et al., 2011). Therefore, how to capture and exploit the intrinsic manifold structure to compute a more effective distance/similarity measure becomes a key task in many areas (Jiang et al., 2011). In this work, we consider recent unsupervised manifold learning methods to provide more effective similarity measures using rank-based approaches. Three of them are considered:

- **Log-based Hypergraph of Ranking References (LHRR)** (Pedronette et al., 2019): an algorithm that models the input ranked lists as hypergraphs and exploits the relations between the elements in the dataset.
- **BFS-Tree of Ranking References (BFSTREE)** (Pedronette et al., 2021): it uses a breadth-first tree structure that models the similarity information between the elements in the ranked lists, which is employed with the objective of analyzing the implicit relations between the elements of the dataset. The tree structure allows a representation of the top- k elements such that the weights of the edges are computed based on the correlations among the ranked lists.
- **The Rank-based Diffusion Process with Assured Convergence (RDPAC)** (Guimarães Pedronette et al., 2021): it performs a diffusion process to exploit the information contained in the ranked lists. It also presents formal proof for the convergence of the diffusion process. The asymptotic complexity of the algorithm is low, which allows its use in many different scenarios with a great number of data elements.

6. Experimental Evaluation

This section discusses the experimental evaluation conducted to assess the effectiveness of the proposed Manifold-GCN. Section 6.1 describes the datasets and features considered. Section 6.2 discusses the experimental protocol. The semi-supervised image classification results are presented in Section 6.3. Section 6.4 shows visualizations of feature space improvements, while Section 6.5 reports a comparison with both traditional and recent state-of-the-art methods, Section 6.6 reports the run-time for each step of the proposed approach.

6.1. Datasets and Features

Three public datasets were considered in the experimental evaluation:

- **Flowers17 (Nilsback and Zisserman, 2006)**: traditional dataset composed of 1,360 images of 17 species of flowers;
- **Corel5k (Liu and Yang, 2013)**: 5,000 diverse images (cars, animals, buildings, and others) divided into 100 classes;
- **CUB-200-2011 (Wah et al., 2011)**: a popular benchmark for image classification composed by 11,788 photos of 200 bird species;

A diverse set of deep features was considered in the conducted experiments. For the general purpose datasets (Flowers17, Corel5k, and CUB-200-2011), all CNNs were trained on ImageNet (Deng et al., 2009) dataset through. The extractions were performed with the PyTorch framework¹. In all cases, the Euclidean distance was considered.

We also employ two different types of Vision Transformers: An Image is Worth 16x16 Words Visual Transformer (ViT-B16) (Dosovitskiy et al., 2021)² and Tokens-To-Token Vision Transformer (T2T-ViT) (Yuan et al., 2021)³. Both were trained on ImageNet (Deng et al., 2009) dataset.

6.2. Experimental Protocol

In this work, the training data is the labeled set and the testing data is the unlabeled set. The unlabeled data considered during the training process comes from the test set. This protocol was also adopted for all baselines.

For the manifold learning approach, all the data is used for the distance learning process, which is completely unsupervised; no labels are used. In the second step, the semi-supervised classification by the GCN, we perform cross-validation that, in our case, consists of a 10-fold split where one fold is used for training and the rest is used for testing. For each of 10 executions (one for every fold being considered as training) and 90% is considered as testing data (unlabeled data). We highlight that, since we are running 10 executions by changing the folds, every dataset element will be considered as training or test at least once. Therefore, each reported value corresponds to the mean of 50 executions (number of executions multiplied by the number of folds).

For all the GCNs, the Adam optimizer with a learning rate of 10^{-5} was used, except for Cub200, in which we used a learning rate of 10^{-4} . Regarding the number of neurons, we used 256. The only exceptions are GCN-SGC, which does not have this parameter; and GCN-GAT which has a number of heads, which was set to 32. The training processes consisted of 200 epochs, using input graphs with $k = 40$. In the same way, the manifold learning methods also have a parameter k , which is different from the graph k . For the method k , we also used $k = 40$.

6.3. Classification Results

The proposed approach was evaluated on a wide diversity of semi-supervised classification scenarios, considering 3 distinct datasets (Flowers, Corel5k, and CUB200). For each dataset, 4 to 5 deep learning features trained on a transfer learning setting were used, considering both CNNs and Vision Transformers approaches. For classification, 5 GCNs models are evaluated considering both the traditional and reciprocal kNN graphs. The impact of re-ranking step is also assessed, evaluating the classification results with and without this step, considering 3 distinct rank-based manifold learning methods. In the semi-supervised scenario, the mean of 5 executions for 10 folds was performed.

Tables 1, 2 and 3 present the results for the datasets Flowers, Corel5k, and CUB200, respectively. The best result for each feature/GCN is highlighted in bold. The gray highlight is used to indicate the best result for the corresponding GCN. The blue color indicates the best result for the dataset (the best result in the whole table).

Some interesting observations can be made from the experimental results. In general, it can be noticed that the reciprocal

¹<https://github.com/Cadene/pretrained-models.pytorch>

²<https://github.com/faustomoraes/vit-keras>

³<https://github.com/yitu-opensource/T2T-ViT>

kNN graph outperforms the traditional kNN graph. It can be observed that the use of manifold learning methods outperforms the scenarios without its use. Moreover, the combination of reciprocal kNN graph and manifold learning methods leads to the best results for all GCN models (gray highlight) and datasets (in blue).

Among the features, VIT-B16 yielded the best results. Therefore, there is a correlation that shows that the better the feature, the better the classification result. In this case, the best feature is VIT-B16. For GCN models and manifold learning methods, the diversity is higher, but GCN-SGC and RD-PAC achieved the best results in most of the scenarios. We also can highlight the remarkable gains obtained on all datasets and features from the original (kNN without re-ranking) to the proposed approach (reciprocal kNN with re-ranking). For CUB200, the most challenging dataset, the accuracy of GCN-APPNP was improved from 55.24% to 75.59%.

Our method was also evaluated considering the weighted F-Measure. Figure 2 reports the results for GCN-SGC on the traditional kNN graph (on the left) and the Reciprocal kNN graph (on the right). For every graph, we see that using manifold learning improves the results of the traditional GCN.

6.4. Visualization Results

In order to visualize the effectiveness of our approach, an experiment was conducted showing the distribution of features in a 2D space, after being processed by t-Distributed Stochastic Neighbor Embedding (TSNE) (van der Maaten and Hinton, 2008). Figure 3 shows the results for (a) the original CNN-ResNet features; (b) the GCN output with kNN graph; (c) the GCN output with kNN graph and manifold learning; (d) the GCN output with Reciprocal graph and manifold learning. The Flowers-17 was chosen for this visualization due to the small number of classes, which makes it easier to visualize the improvements. Each class is represented by a different combination of shape and color. Notice that the distribution of classes is further improved when the Manifold-GCN is applied (c and d), which is consistent with our main hypothesis.

6.5. Comparison with Other Approaches

For comparison purposes, a wide variety of supervised and semi-supervised classification approaches were considered, both traditional and more recent ones. A brief description of the employed baselines, the implementations, and the parameters used are presented in the following:

- **k Nearest Neighbor (kNN)**: traditional approach that computes the distance to the other elements in the dataset and selects the k closest ones. The sklearn implementation was used, with $k = 20$.
- **Support Vector Machine (SVM)** (Cortes and Vapnik, 1995): it is a traditional method that consists in finding the hyperplane that best separates the data into the correct classes in a high-dimensional space. The sklearn implementation was used, with default parameters and Radial Basis Function (RBF) kernel.
- **Single-Layer and Multi-Layer Perceptron**: the sklearn implementation was used for both, with the Stochastic Gradient Descent (SGD) optimizer.
- **Optimum-Path Forest (OPF)** (Amorim et al., 2014; Papa et al., 2009): it builds a graph where each node is an element of the dataset and the edges are weighted by their Euclidean distance. The algorithm computes the optimum path between the nodes in order to classify them into a given class. The pyOPF⁴ implementation was used, with the default parameters.
- **Pseudo-label** (Lee, 2013): the method is semi-supervised and is used to assign labels to unlabeled data. In this work, a public implementation⁵ was used along with the Logistic Regression classifier that employed the Stochastic Gradient Descent (SGD) optimizer and Squared Hinge loss with $\alpha = 10^{-5}$ for training.
- **Label Spreading (LS)** (Zhou et al., 2004): a semi-supervised algorithm that attributes labels to elements according to the labels of their neighbors, given a certain degree of similarity. For this process, it uses an affinity matrix based on a normalized graph Laplacian. The sklearn implementation was used, considering a Radial Basis Function (RBF) kernel with $\alpha = 0.4125$, $\gamma = 0.1$, and a maximum of 100 iterations. This method is used to expand the training set and is used along with the other classifiers.
- **Learning Discrete Structures for Graph Neural Networks (GNN-LDS and GNN-KNN-LDS)** (Franceschi et al., 2019): this Graph Neural Network (GNN) learns both a graph and embeddings from the input features. It approximately solves a bilevel program that learns a discrete probability distribution on the edges of the graph. The authors claim that this is the first method that simultaneously learns the graph and the parameters of a GNN for semi-supervised classification. The approach presents two variants: (i) GNN-LDS; and (ii) GNN-KNN-LDS which initializes by computing a kNN graph. Both were used as baselines with their default parameters proposed in the implementation⁶ provided by the original authors. For the kNN graph, $k = 20$ was used.
- **Weakly Supervised Framework Experiments Framework (WSEF)** (Presotto et al., 2021): the method generates pseudo-labels by applying different rank correlation measures (e.g., Jaccard, Spearman). The approach is mainly based on the idea that elements that have ranked lists with a high intersection with others probably belong to the same class. The implementation⁷ provided by the authors was used considering Rank Biased Overlap

⁴<https://github.com/marcoscleison/PyOPF>

⁵https://github.com/anirudhshenoy/pseudo_labeling_small_datasets

⁶<https://github.com/lucfra/LDS-GNN>

⁷<https://github.com/UDLF/WSEF>

Table 1: Impact of manifold learning approaches (LHRR, RDPAC, BFSTREE) and Reciprocal Graph (Rec.) on the classification accuracy (%) of 5 different GCN models on Flowers17 dataset. The best results for each feature and GCN model are highlighted in bold, the best results for each GCN model are marked with a gray background, and the best result for the entire dataset is highlighted in blue. In all the cases, the best results used manifold learning and Reciprocal Graph.

| Classifier Specification | | | Feature | | | | |
|--------------------------|-------|---------|---------------------------------|----------------------------------|--------------------------------|----------------------------------|---------------------------------------|
| GCN | Graph | Re-Rank | CNN-ResNet (He et al., 2016) | CNN-DPNet (Chen et al., 2017) | CNN-SENet (Hu et al., 2018) | T2T-VIT24 (Yuan et al., 2021) | VIT-B16 (Dosovitskiy et al., 2021) |
| GCN-Net | kNN | — | 79.08 ± 0.3039 | 76.94 ± 0.3688 | 72.72 ± 0.2052 | 69.75 ± 0.0827 | 92.72 ± 0.1324 |
| | kNN | LHRR | 84.37 ± 0.3239 | 80.76 ± 0.1372 | 73.89 ± 0.133 | 72.03 ± 0.1131 | 95.88 ± 0.0567 |
| | kNN | RDPAC | 83.91 ± 0.1279 | 81.24 ± 0.2597 | 74.76 ± 0.2245 | 74.60 ± 0.1353 | 96.86 ± 0.0702 |
| | kNN | BFSTREE | 83.12 ± 0.1784 | 81.39 ± 0.1222 | 74.83 ± 0.1284 | 72.49 ± 0.3283 | 96.33 ± 0.0695 |
| | Rec. | — | 83.89 ± 0.1973 | 81.19 ± 0.264 | 76.23 ± 0.1913 | 75.82 ± 0.2096 | 97.07 ± 0.0606 |
| | Rec. | LHRR | 84.67 ± 0.0988 | 80.64 ± 0.1749 | 73.97 ± 0.1383 | 72.40 ± 0.1927 | 95.39 ± 0.1583 |
| | Rec. | RDPAC | 84.20 ± 0.1975 | 82.27 ± 0.1659 | 76.61 ± 0.1968 | 75.87 ± 0.1877 | 97.16 ± 0.0168 |
| | Rec. | BFSTREE | 82.97 ± 0.1623 | 81.20 ± 0.1141 | 74.80 ± 0.2034 | 73.26 ± 0.1008 | 96.52 ± 0.0538 |
| GCN-SGC | kNN | — | 79.64 ± 0.1023 | 77.09 ± 0.1139 | 73.00 ± 0.0941 | 70.05 ± 0.0802 | 92.84 ± 0.0655 |
| | kNN | LHRR | 84.41 ± 0.0835 | 80.36 ± 0.0661 | 74.04 ± 0.0599 | 72.11 ± 0.113 | 95.85 ± 0.0285 |
| | kNN | RDPAC | 84.19 ± 0.0659 | 81.12 ± 0.0645 | 75.06 ± 0.0627 | 75.18 ± 0.0743 | 96.95 ± 0.0133 |
| | kNN | BFSTREE | 83.33 ± 0.0533 | 81.54 ± 0.0410 | 75.08 ± 0.0565 | 72.86 ± 0.0879 | 96.42 ± 0.0396 |
| | Rec. | — | 83.99 ± 0.0478 | 81.32 ± 0.0314 | 76.16 ± 0.0415 | 75.69 ± 0.0828 | 96.93 ± 0.0464 |
| | Rec. | LHRR | 84.91 ± 0.0665 | 80.75 ± 0.0694 | 74.60 ± 0.0510 | 72.95 ± 0.0563 | 95.47 ± 0.0171 |
| | Rec. | RDPAC | 84.53 ± 0.0580 | 82.53 ± 0.1335 | 76.93 ± 0.0376 | 76.43 ± 0.0499 | 97.11 ± 0.0163 |
| | Rec. | BFSTREE | 83.43 ± 0.0200 | 81.58 ± 0.1169 | 75.03 ± 0.0313 | 73.58 ± 0.026 | 96.63 ± 0.0337 |
| GCN-GAT | kNN | — | 80.67 ± 0.2144 | 65.60 ± 0.9961 | 74.64 ± 0.3048 | 67.33 ± 0.9069 | 93.65 ± 0.228 |
| | kNN | LHRR | 84.52 ± 0.3202 | 76.15 ± 1.4547 | 75.48 ± 0.2365 | 73.37 ± 0.2824 | 95.33 ± 0.2522 |
| | kNN | RDPAC | 84.02 ± 0.1058 | 77.39 ± 1.2703 | 75.29 ± 0.3550 | 75.40 ± 0.4316 | 97.09 ± 0.0572 |
| | kNN | BFSTREE | 83.04 ± 0.1844 | 77.19 ± 1.833 | 75.82 ± 0.2086 | 73.26 ± 0.3184 | 96.41 ± 0.0465 |
| | Rec. | — | 83.67 ± 0.1965 | 77.42 ± 0.6762 | 76.74 ± 0.3398 | 74.82 ± 0.2978 | 96.99 ± 0.0558 |
| | Rec. | LHRR | 84.82 ± 0.2194 | 79.63 ± 0.6337 | 75.22 ± 0.2648 | 73.32 ± 0.3684 | 95.21 ± 0.2575 |
| | Rec. | RDPAC | 84.40 ± 0.1488 | 79.69 ± 1.0373 | 77.18 ± 0.2940 | 76.90 ± 0.3418 | 97.22 ± 0.0557 |
| | Rec. | BFSTREE | 82.79 ± 0.2926 | 78.74 ± 0.2682 | 75.94 ± 0.2681 | 73.85 ± 0.2632 | 96.55 ± 0.0881 |
| GCN-APPNP | kNN | — | 77.25 ± 0.1692 | 76.38 ± 0.238 | 71.0 ± 0.4051 | 69.45 ± 0.3072 | 90.24 ± 0.2128 |
| | kNN | LHRR | 84.58 ± 0.2621 | 82.53 ± 0.2443 | 76.83 ± 0.1622 | 74.32 ± 0.2989 | 96.05 ± 0.0421 |
| | kNN | RDPAC | 85.35 ± 0.2205 | 83.32 ± 0.1287 | 76.89 ± 0.3673 | 77.87 ± 0.0660 | 97.28 ± 0.0303 |
| | kNN | BFSTREE | 84.22 ± 0.1638 | 83.34 ± 0.0875 | 77.94 ± 0.3084 | 75.65 ± 0.2505 | 96.73 ± 0.0763 |
| | Rec. | — | 83.91 ± 0.1181 | 82.20 ± 0.2160 | 77.74 ± 0.1645 | 77.11 ± 0.1485 | 97.24 ± 0.0470 |
| | Rec. | LHRR | 85.88 ± 0.1896 | 82.55 ± 0.2138 | 76.60 ± 0.2479 | 75.40 ± 0.2458 | 95.68 ± 0.1083 |
| | Rec. | RDPAC | 85.41 ± 0.2304 | 83.99 ± 0.1276 | 78.82 ± 0.1466 | 78.01 ± 0.1307 | 97.43 ± 0.0699 |
| | Rec. | BFSTREE | 83.75 ± 0.2099 | 83.14 ± 0.1915 | 77.83 ± 0.1826 | 75.85 ± 0.2098 | 96.89 ± 0.0632 |
| GCN-ARMA | kNN | — | 78.69 ± 0.2471 | 76.01 ± 0.295 | 73.18 ± 0.4015 | 70.47 ± 0.2548 | 91.27 ± 0.1731 |
| | kNN | LHRR | 84.64 ± 0.3211 | 81.90 ± 0.4272 | 76.09 ± 0.1451 | 74.26 ± 0.2543 | 95.66 ± 0.1726 |
| | kNN | RDPAC | 85.05 ± 0.1643 | 82.38 ± 0.3741 | 76.18 ± 0.3637 | 76.75 ± 0.2599 | 96.88 ± 0.0698 |
| | kNN | BFSTREE | 83.72 ± 0.0791 | 81.96 ± 0.3477 | 76.81 ± 0.1272 | 75.03 ± 0.1647 | 96.24 ± 0.0812 |
| | Rec. | — | 83.32 ± 0.3713 | 80.86 ± 0.1282 | 76.96 ± 0.3041 | 76.11 ± 0.3851 | 96.66 ± 0.1140 |
| | Rec. | LHRR | 85.36 ± 0.3818 | 82.17 ± 0.3283 | 75.92 ± 0.2516 | 74.64 ± 0.3728 | 95.13 ± 0.2118 |
| | Rec. | RDPAC | 84.97 ± 0.2524 | 83.14 ± 0.3078 | 77.89 ± 0.2358 | 77.81 ± 0.3271 | 97.02 ± 0.0944 |
| | Rec. | BFSTREE | 84.06 ± 0.2612 | 82.21 ± 0.1901 | 76.88 ± 0.1897 | 75.10 ± 0.3000 | 96.47 ± 0.1171 |

(RBO) (Webber et al., 2010) correlation measure, $k = 40$ in combination with SVM.

- **CoMatch** (Li et al., 2021): the method is based on concepts of graph-based self-supervised learning. The approach is trained to produce similar embeddings for the same image with different augmentations. CoMatch jointly optimizes three losses: (i) a supervised classification loss on labeled data, (ii) an unsupervised classification loss on unlabeled data, and (iii) a graph-based contrastive loss on unlabeled data. It takes images as input instead of features. This version employs ResNet (He et al., 2016) as the backbone. We considered the implementa-

tion provided by the authors⁸, with default parameters (the ones used for ImageNet (Deng et al., 2009) in their code). We trained with a batch size of 25 and 400 epochs for all datasets. Except for the CUB200 dataset, which is larger, we used a batch size of 50 and 300 epochs.

We also compared our results with three CNN-based classifiers, considering image data as input. The images were provided with a size of 100x100 pixels in batches of size 32. For the other methods, the input consists of feature vectors obtained from deep features trained through transfer learning. Notice that the CNNs used as baselines require more labeled data in

⁸<https://github.com/salesforce/CoMatch>

Table 2: Impact of manifold learning approaches (LHRR, RDPAC, BFSTREE) and Reciprocal Graph (Rec.) on the classification accuracy (%) of 5 different GCN models on Core5k dataset. The best results for each feature and GCN model are highlighted in bold, the best results for each GCN model are marked with a gray background, and the best result for the entire dataset is highlighted in blue. In all the cases, the best results used manifold learning.

| Classifier Specification | | | Feature | | | | |
|--------------------------|-------|---------|---------------------------------|----------------------------------|--------------------------------|----------------------------------|---------------------------------------|
| GCN | Graph | Re-Rank | CNN-ResNet (He et al., 2016) | CNN-DPNet (Chen et al., 2017) | CNN-SENet (Hu et al., 2018) | T2T-VIT24 (Yuan et al., 2021) | VIT-B16 (Dosovitskiy et al., 2021) |
| GCN-Net | kNN | — | 89.34 ± 0.0950 | 86.49 ± 0.0998 | 89.17 ± 0.0956 | 89.02 ± 0.1452 | 89.93 ± 0.2878 |
| | kNN | LHRR | 91.40 ± 0.0906 | 88.94 ± 0.1958 | 90.19 ± 0.1392 | 90.68 ± 0.0957 | 94.57 ± 0.121 |
| | kNN | RDPAC | 91.46 ± 0.1402 | 89.05 ± 0.1054 | 90.65 ± 0.0483 | 91.77 ± 0.1246 | 94.29 ± 0.139 |
| | kNN | BFSTREE | 92.03 ± 0.1165 | 89.28 ± 0.1858 | 91.19 ± 0.1102 | 91.78 ± 0.0432 | 94.30 ± 0.3362 |
| | Rec. | — | 91.68 ± 0.1064 | 89.62 ± 0.1114 | 91.81 ± 0.1159 | 92.19 ± 0.0908 | 93.42 ± 0.1987 |
| | Rec. | LHRR | 91.68 ± 0.0224 | 88.48 ± 0.1268 | 90.58 ± 0.0901 | 91.50 ± 0.0684 | 94.63 ± 0.139 |
| | Rec. | RDPAC | 92.00 ± 0.1434 | 89.55 ± 0.0944 | 90.93 ± 0.1654 | 91.96 ± 0.0705 | 94.76 ± 0.1577 |
| | Rec. | BFSTREE | 92.00 ± 0.0954 | 89.33 ± 0.1221 | 91.32 ± 0.0833 | 92.43 ± 0.0401 | 94.39 ± 0.2771 |
| GCN-SGC | kNN | — | 89.62 ± 0.0321 | 86.78 ± 0.0256 | 89.81 ± 0.0426 | 88.95 ± 0.0482 | 93.36 ± 0.0401 |
| | kNN | LHRR | 91.19 ± 0.0262 | 88.74 ± 0.0242 | 89.90 ± 0.044 | 90.49 ± 0.0518 | 95.20 ± 0.0219 |
| | kNN | RDPAC | 91.47 ± 0.0216 | 88.95 ± 0.0632 | 90.70 ± 0.0403 | 91.77 ± 0.0521 | 94.76 ± 0.078 |
| | kNN | BFSTREE | 91.98 ± 0.0246 | 89.23 ± 0.0453 | 91.40 ± 0.0061 | 91.71 ± 0.0444 | 95.26 ± 0.0759 |
| | Rec. | — | 91.98 ± 0.0133 | 89.83 ± 0.0415 | 92.15 ± 0.0164 | 92.75 ± 0.0908 | 95.49 ± 0.0107 |
| | Rec. | LHRR | 91.73 ± 0.0508 | 88.70 ± 0.0669 | 90.73 ± 0.0235 | 91.68 ± 0.0305 | 95.57 ± 0.017 |
| | Rec. | RDPAC | 92.00 ± 0.0247 | 89.84 ± 0.1057 | 90.85 ± 0.0396 | 92.31 ± 0.072 | 95.50 ± 0.020 |
| | Rec. | BFSTREE | 92.04 ± 0.009 | 89.49 ± 0.0627 | 91.30 ± 0.0257 | 92.54 ± 0.0591 | 95.30 ± 0.0479 |
| GCN-GAT | kNN | — | 90.48 ± 0.1727 | 83.28 ± 0.33 | 91.13 ± 0.1107 | 90.7 ± 0.1187 | 91.3 ± 0.1764 |
| | kNN | LHRR | 92.21 ± 0.1328 | 88.59 ± 0.4012 | 91.28 ± 0.2208 | 92.2 ± 0.0839 | 94.56 ± 0.1777 |
| | kNN | RDPAC | 91.86 ± 0.1403 | 89.78 ± 0.2723 | 91.41 ± 0.1429 | 92.82 ± 0.0956 | 94.46 ± 0.2555 |
| | kNN | BFSTREE | 92.42 ± 0.1008 | 89.61 ± 0.362 | 91.95 ± 0.1382 | 93.09 ± 0.1337 | 94.58 ± 0.2226 |
| | Rec. | — | 92.02 ± 0.0917 | 89.0 ± 0.2638 | 92.23 ± 0.0844 | 92.81 ± 0.113 | 93.64 ± 0.2373 |
| | Rec. | LHRR | 92.19 ± 0.1057 | 89.17 ± 0.2074 | 91.18 ± 0.1451 | 92.41 ± 0.1456 | 94.55 ± 0.1918 |
| | Rec. | RDPAC | 92.22 ± 0.0858 | 90.48 ± 0.1718 | 91.48 ± 0.1021 | 93.02 ± 0.1334 | 94.89 ± 0.1492 |
| | Rec. | BFSTREE | 92.30 ± 0.1128 | 90.01 ± 0.2374 | 91.88 ± 0.1081 | 93.35 ± 0.1537 | 94.75 ± 0.1385 |
| GCN-APPNP | kNN | — | 89.72 ± 0.2031 | 87.68 ± 0.0785 | 89.92 ± 0.0992 | 89.86 ± 0.0731 | 86.89 ± 0.2487 |
| | kNN | LHRR | 92.6 ± 0.0625 | 90.81 ± 0.1043 | 91.49 ± 0.1307 | 91.83 ± 0.0952 | 94.53 ± 0.1144 |
| | kNN | RDPAC | 92.69 ± 0.1161 | 90.75 ± 0.179 | 91.81 ± 0.098 | 92.58 ± 0.0588 | 94.12 ± 0.2213 |
| | kNN | BFSTREE | 93.04 ± 0.0872 | 91.01 ± 0.1026 | 92.35 ± 0.0535 | 92.83 ± 0.031 | 94.37 ± 0.0855 |
| | Rec. | — | 92.69 ± 0.05 | 90.70 ± 0.1301 | 92.79 ± 0.0429 | 93.56 ± 0.0669 | 93.53 ± 0.1042 |
| | Rec. | LHRR | 92.88 ± 0.1058 | 89.99 ± 0.0869 | 91.78 ± 0.0694 | 92.63 ± 0.0817 | 94.95 ± 0.2116 |
| | Rec. | RDPAC | 92.82 ± 0.046 | 90.95 ± 0.1134 | 91.92 ± 0.0738 | 93.17 ± 0.0804 | 95.13 ± 0.1095 |
| | Rec. | BFSTREE | 93.08 ± 0.0727 | 90.78 ± 0.1317 | 92.39 ± 0.0269 | 93.70 ± 0.0653 | 94.72 ± 0.1564 |
| GCN-ARMA | kNN | — | 88.58 ± 0.312 | 86.47 ± 0.0729 | 89.11 ± 0.1061 | 89.16 ± 0.0571 | 85.48 ± 0.3945 |
| | kNN | LHRR | 91.58 ± 0.1185 | 89.84 ± 0.1565 | 90.98 ± 0.1738 | 91.46 ± 0.0963 | 90.66 ± 0.5051 |
| | kNN | RDPAC | 91.72 ± 0.1775 | 90.09 ± 0.2858 | 91.08 ± 0.1226 | 92.28 ± 0.0545 | 92.62 ± 0.4067 |
| | kNN | BFSTREE | 92.23 ± 0.1447 | 90.31 ± 0.1195 | 91.7 ± 0.0869 | 92.24 ± 0.0682 | 92.28 ± 0.3061 |
| | Rec. | — | 91.14 ± 0.137 | 89.24 ± 0.2139 | 91.31 ± 0.1887 | 91.84 ± 0.0774 | 90.48 ± 0.1707 |
| | Rec. | LHRR | 91.77 ± 0.1541 | 89.24 ± 0.1428 | 91.07 ± 0.126 | 91.78 ± 0.1145 | 92.39 ± 0.2078 |
| | Rec. | RDPAC | 92.05 ± 0.1403 | 90.41 ± 0.1645 | 91.47 ± 0.1202 | 92.49 ± 0.2056 | 92.80 ± 0.1896 |
| | Rec. | BFSTREE | 92.27 ± 0.0377 | 90.14 ± 0.1897 | 91.71 ± 0.1753 | 92.90 ± 0.1446 | 92.74 ± 0.2083 |

comparison to other methods and were evaluated on a supervised cross-validation scenario (9 folds for training, 1 fold for testing). Except for CNN classifiers, all other methods were evaluated on semi-supervised scenarios (1 fold for training, 9 folds for testing).

Table 4 presents the comparison with both traditional and recent state-of-the-art baselines in relation to our approach on Flowers, Core5k, and CUB200 datasets. Most results are the mean of 5 executions of 10 folds, with some exceptions which are indicated in italic text. Some methods require long running times on larger datasets (i.e., LDS and CoMatch). For GNN-KNN-LDS, KNN-LDS, and CoMatch, the results on CUB200 correspond to 1 execution. For CoMatch, the mean of 3 executions is reported for Core5k dataset. The best result for each feature is highlighted in bold and the best for each dataset

is highlighted in red. The gray rows indicate the results that correspond to our method.

The proposed method revealed superior results compared to the baselines in most of the cases. The only exception is Flowers with VIT-B16 features where WSEF shows the best results (97.82% accuracy). However, our ManifoldGCN is very close with 97.43% accuracy.

6.6. Efficiency Results

We conducted an experiment to measure the run-time (in seconds) for running each of the manifold learning methods and GCN models. The experiments were executed on a machine with an Intel(R) Core(TM) i7-10700F CPU @ 2.90GHz, 32 GB RAM, NVIDIA GeForce RTX 3060 GPU with 12GB VRAM

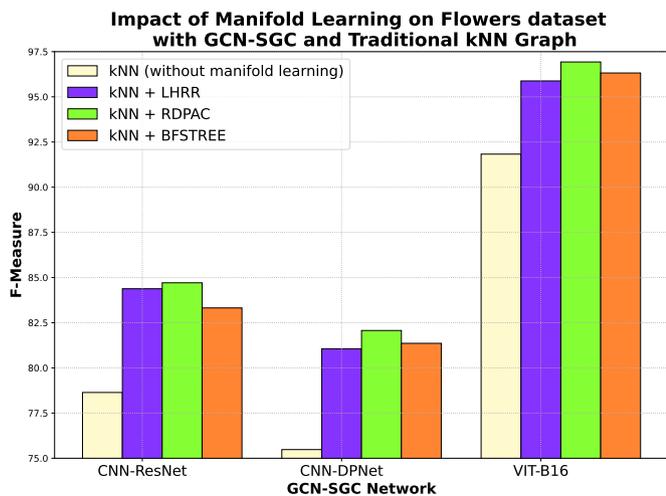
Table 3: Impact of manifold learning approaches (LHRR, RDPAC, BFSTREE) and Reciprocal Graph (Rec.) on the classification accuracy (%) of 5 different GCN models on CUB200 dataset. The best results for each feature and GCN model are highlighted in bold, the best results for each GCN model are marked with a gray background, and the best result for the entire dataset is highlighted in blue. In all the cases, the best results used manifold learning.

| Classifier Specification | | | Feature | | | |
|--------------------------|-------|---------|---------------------------------|----------------------------------|---------------------------------|---------------------------------------|
| GCN | Graph | Re-Rank | CNN-ResNet (He et al., 2016) | CNN-SENet (Chen et al., 2017) | CNN-Xception (Chollet, 2017) | VIT-B16 (Dosovitskiy et al., 2021) |
| GCN-Net | kNN | — | 40.76 ± 0.7467 | 35.8 ± 0.0634 | 46.66 ± 0.019 | 64.39 ± 0.4486 |
| | kNN | LHRR | 49.16 ± 0.3119 | 36.17 ± 0.1153 | 51.13 ± 0.0738 | 70.42 ± 0.671 |
| | kNN | RDPAC | 49.44 ± 0.1092 | 36.84 ± 0.0578 | 51.18 ± 0.0284 | 72.71 ± 0.1506 |
| | kNN | BFSTREE | 49.18 ± 0.1011 | 37.10 ± 0.0482 | 50.62 ± 0.0639 | 71.54 ± 0.1888 |
| | Rec. | — | 49.46 ± 0.3279 | 39.42 ± 0.113 | 50.76 ± 0.0713 | 68.85 ± 0.3055 |
| | Rec. | LHRR | 51.23 ± 0.0788 | 36.5 ± 0.0728 | 51.92 ± 0.0546 | 73.49 ± 0.1879 |
| | Rec. | RDPAC | 51.57 ± 0.0999 | 38.57 ± 0.0712 | 53.12 ± 0.0596 | 74.39 ± 0.3061 |
| | Rec. | BFSTREE | 50.80 ± 0.0291 | 37.8 ± 0.0538 | 51.82 ± 0.0658 | 73.58 ± 0.3939 |
| GCN-SGC | kNN | — | 47.55 ± 0.0329 | 36.48 ± 0.0684 | 48.60 ± 0.0072 | 74.23 ± 0.0385 |
| | kNN | LHRR | 51.22 ± 0.0184 | 35.88 ± 0.0137 | 52.36 ± 0.0125 | 77.84 ± 0.0519 |
| | kNN | RDPAC | 51.88 ± 0.0315 | 37.75 ± 0.0148 | 52.98 ± 0.0103 | 78.16 ± 0.0453 |
| | kNN | BFSTREE | 51.66 ± 0.016 | 37.70 ± 0.01 | 52.21 ± 0.0095 | 77.31 ± 0.0563 |
| | Rec. | — | 53.71 ± 0.0362 | 40.31 ± 0.0255 | 54.0 ± 0.0054 | 78.03 ± 0.0428 |
| | Rec. | LHRR | 51.99 ± 0.0251 | 36.74 ± 0.0162 | 53.12 ± 0.0153 | 78.54 ± 0.0177 |
| | Rec. | RDPAC | 52.85 ± 0.0164 | 38.91 ± 0.0073 | 54.59 ± 0.0036 | 79.27 ± 0.0325 |
| | Rec. | BFSTREE | 52.68 ± 0.0308 | 38.65 ± 0.023 | 53.54 ± 0.0041 | 78.12 ± 0.0344 |
| GCN-GAT | kNN | — | 41.84 ± 0.2901 | 32.5 ± 0.205 | 42.45 ± 0.1848 | 59.53 ± 0.5668 |
| | kNN | LHRR | 48.86 ± 0.1593 | 34.78 ± 0.1155 | 48.8 ± 0.246 | 64.02 ± 0.4082 |
| | kNN | RDPAC | 49.05 ± 0.1145 | 35.9 ± 0.1158 | 49.03 ± 0.1037 | 68.78 ± 0.2495 |
| | kNN | BFSTREE | 48.77 ± 0.1427 | 35.98 ± 0.1457 | 48.3 ± 0.1084 | 68.1 ± 0.2488 |
| | Rec. | — | 45.46 ± 0.1879 | 33.02 ± 0.1206 | 45.88 ± 0.16 | 64.82 ± 0.2582 |
| | Rec. | LHRR | 50.19 ± 0.0904 | 35.28 ± 0.1364 | 50.17 ± 0.1073 | 70.31 ± 0.0762 |
| | Rec. | RDPAC | 50.95 ± 0.0632 | 37.55 ± 0.1087 | 51.29 ± 0.1577 | 72.94 ± 0.1716 |
| | Rec. | BFSTREE | 49.89 ± 0.1871 | 36.67 ± 0.129 | 49.87 ± 0.1245 | 71.73 ± 0.1775 |
| GCN-APPNP | kNN | — | 29.16 ± 0.6867 | 30.27 ± 0.3694 | 42.68 ± 0.0826 | 55.24 ± 0.5689 |
| | kNN | LHRR | 47.0 ± 0.1836 | 34.91 ± 0.1598 | 48.77 ± 0.0979 | 66.57 ± 0.572 |
| | kNN | RDPAC | 47.19 ± 0.0701 | 35.29 ± 0.1195 | 47.72 ± 0.094 | 69.92 ± 0.2262 |
| | kNN | BFSTREE | 46.59 ± 0.2154 | 35.28 ± 0.0718 | 47.14 ± 0.0895 | 70.86 ± 0.2702 |
| | Rec. | — | 48.51 ± 0.1192 | 38.02 ± 0.0461 | 47.51 ± 0.0452 | 68.29 ± 0.0935 |
| | Rec. | LHRR | 51.99 ± 0.0800 | 37.45 ± 0.0768 | 51.43 ± 0.084 | 74.61 ± 0.0991 |
| | Rec. | RDPAC | 51.82 ± 0.1028 | 39.15 ± 0.1601 | 52.17 ± 0.0865 | 75.59 ± 0.2139 |
| | Rec. | BFSTREE | 50.6 ± 0.0848 | 38.21 ± 0.0358 | 50.26 ± 0.1301 | 74.15 ± 0.1837 |
| GCN-ARMA | kNN | — | 38.74 ± 0.4527 | 32.96 ± 0.1626 | 42.91 ± 0.1465 | 60.26 ± 0.4398 |
| | kNN | LHRR | 47.58 ± 0.2387 | 34.56 ± 0.0799 | 49.26 ± 0.2191 | 67.21 ± 0.2825 |
| | kNN | RDPAC | 47.77 ± 0.2075 | 35.4 ± 0.1474 | 49.88 ± 0.1479 | 71.16 ± 0.2337 |
| | kNN | BFSTREE | 47.12 ± 0.3126 | 35.6 ± 0.1385 | 48.78 ± 0.0991 | 70.13 ± 0.4433 |
| | Rec. | — | 44.37 ± 0.1739 | 34.25 ± 0.1559 | 46.95 ± 0.3062 | 64.55 ± 0.3184 |
| | Rec. | LHRR | 49.29 ± 0.0987 | 35.22 ± 0.0891 | 50.38 ± 0.1318 | 70.05 ± 0.653 |
| | Rec. | RDPAC | 49.81 ± 0.2090 | 37.12 ± 0.1276 | 51.63 ± 0.1155 | 73.29 ± 0.34 |
| | Rec. | BFSTREE | 48.92 ± 0.2721 | 36.38 ± 0.1331 | 50.41 ± 0.0777 | 72.17 ± 0.3336 |

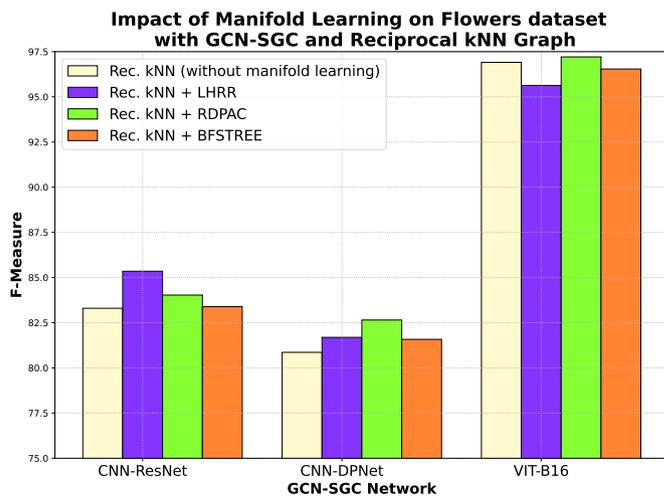
running Ubuntu 20.04 with Linux kernel 5.15.0-52-generic. Table 5 reports the average and standard deviation of 5 executions of 10 folds on each dataset and for the two types of graphs (*kNN* and Reciprocal *kNN*).

Manifold Learning (M.L.) performs the pre-processing of the GCN graph. Since these methods are not currently parallelized, they run all on CPU. Parallelization of these approaches is out of the scope of this paper, but rank-based methods can be parallelized with data parallelism as shown in other papers (Valem et al., 2017, 2015). While the training involves both the GCN initialization and the learning process, testing is responsible for computing the classification of all the queries. Both training and testing are performed on the GPU.

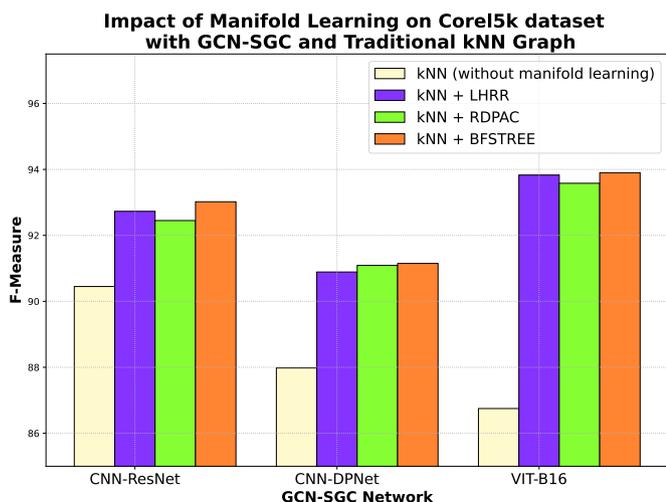
Notice that the execution times are very low, which indicates that the method is fast even for the more robust GCNs. Also, most compared methods have a costly training process. An example is CoMatch (2021) that requires huge training times: 40 minutes on Flowers; 88.4 minutes on Corel5k; 260 minutes on CUB200. These times are the average of executions for 100 epochs. However, CoMatch is generally recommended to be trained for 400 epochs. These values are much higher than our proposed approach.



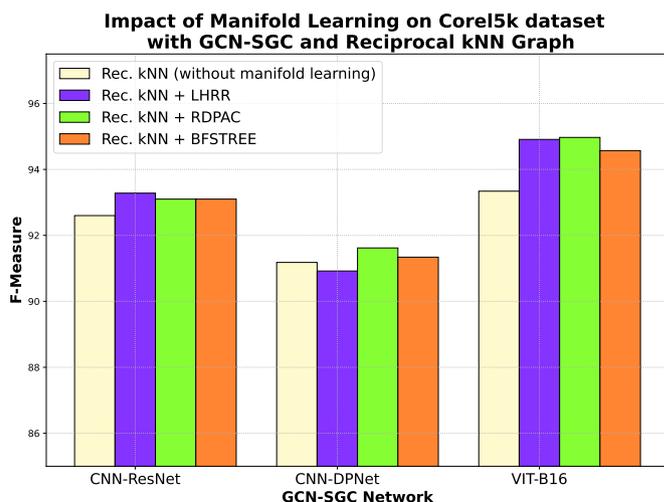
(a) Flowers - Traditional kNN



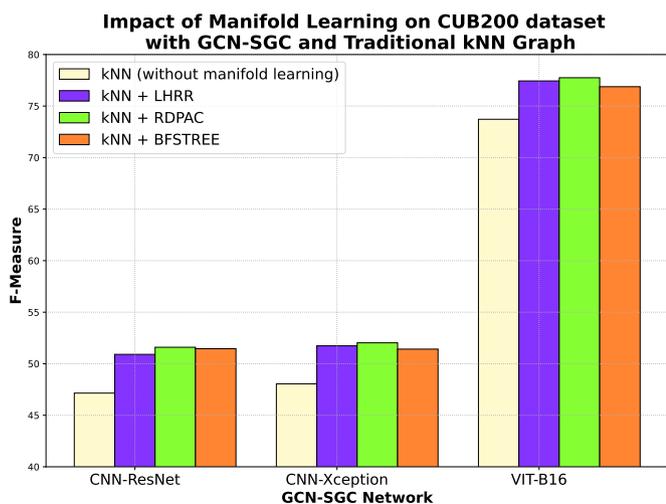
(b) Flowers - Reciprocal kNN



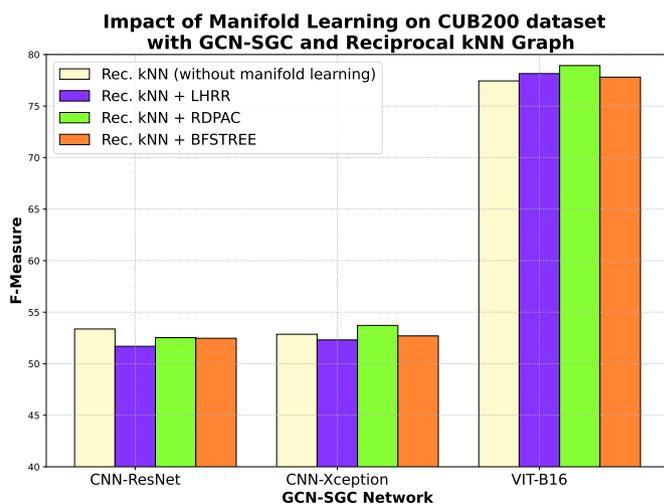
(c) Corel5k - Traditional kNN



(d) Corel5k - Reciprocal kNN



(e) Cub200 - Traditional kNN



(f) Cub200 - Reciprocal kNN

Fig. 2: Impact of manifold learning approaches on F-measure results considering GCN-SGC on different datasets and features.

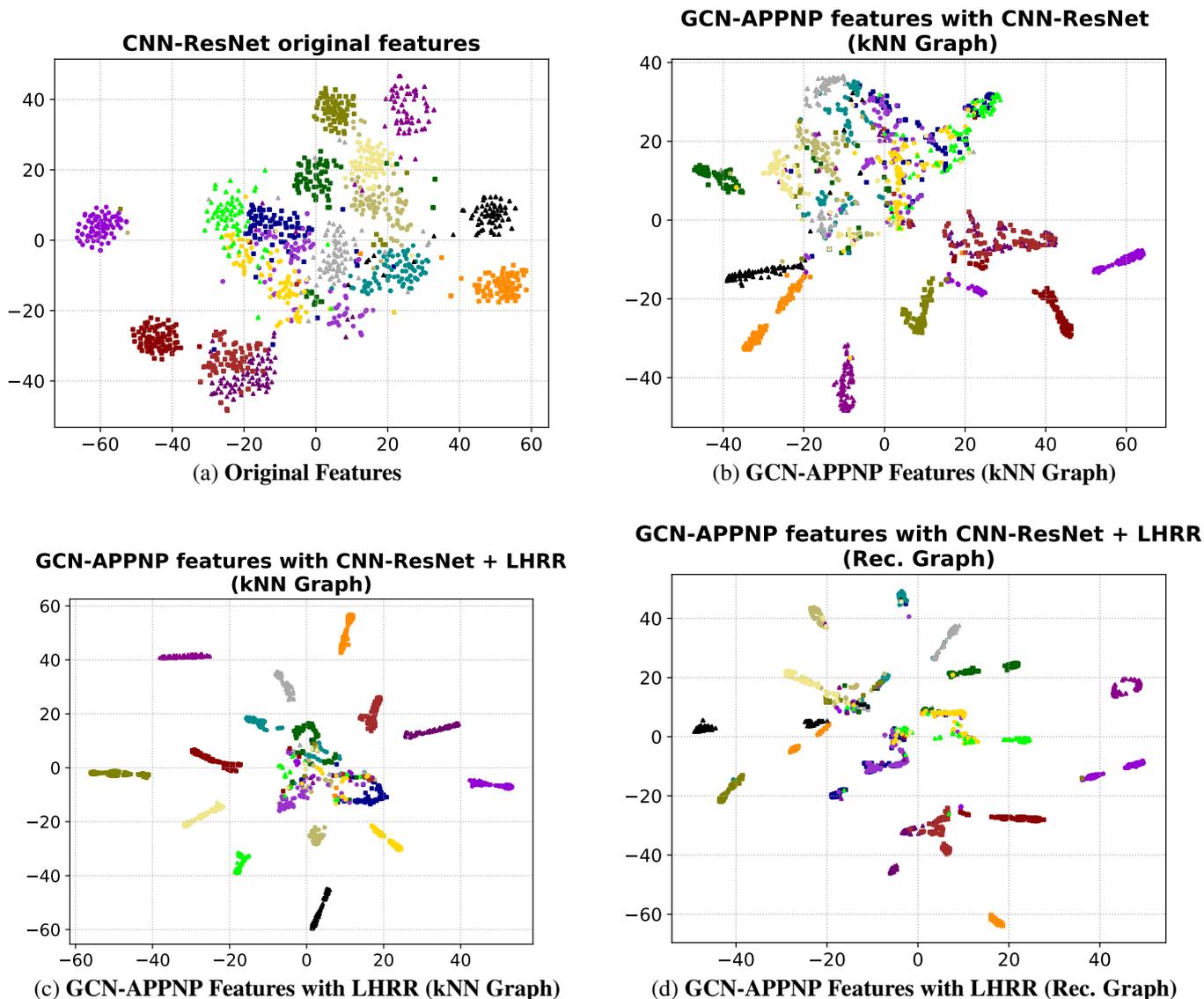


Fig. 3: t-SNE visualizations that show the feature space improvement when manifold learning and reciprocal graph were applied. Experiments were conducted on the Flowers dataset and CNN-ResNet features. Each class is represented by a different shape and color.

7. Conclusions

In this work, we presented a novel framework, the Manifold-GCN, for semi-supervised image classification. The approach is flexible and allows the use of different types of GCNs, graphs, features, and manifold learning. To the best of our knowledge, this is the first framework that allows the combination of GCNs with different types of manifold learning approaches for image classification. In our work, all the manifold learning algorithms are completely unsupervised, which is especially useful for scenarios where labeled data is limited.

An experimental evaluation was conducted on 3 datasets and 5 distinct deep learning features, including features obtained by CNNs and Visual Transformers. The results validated our main hypothesis by revealing that the manifold learning methods were capable of improving the effectiveness (accuracy and F-Measure) in the vast majority of the cases, for different GCNs

and features. The visual analyzes show that the feature space was significantly improved by the proposed approach. In comparison to both traditional and very recent state-of-the-art methods, it is clearly noticeable that Manifold-GCN is among the best in most cases. Our approach is also very efficient and requires very low execution times for training and testing.

In future work, we intend to use other types of graph models in addition to kNN Graph and Reciprocal Graph. For this work, the input feature and the feature used to model the graph are the same. Since complementary features can possibly increase results even further, we intend to evaluate cases where a different feature can be used for the graph. We also intend to employ Manifold-GCN in larger image collections and retrieval scenarios. For efficiency purposes, it is also possible to parallelize the implementations of manifold learning approaches.

Table 4: Accuracy comparison (%) for baselines on Flowers, Corel5k, and CUB200 datasets. For every dataset, we compared our approach with both supervised and semi-supervised baselines. The methods are compared with different input features. The results of our method are highlighted with a gray background; the best results for each pair of features and dataset are marked in bold, and the best results for each dataset are in red.

| Method | Year | Input | Training | Flowers | Corel5k | CUB200 |
|---------------------------|------|---------------------|-----------------|--------------|--------------|--------------|
| MobileNet | 2017 | Images | Supervised | 86.66 | 90.90 | 35.20 |
| ResNet50 | 2015 | | | 85.97 | 91.52 | 31.10 |
| CNN-Xception | 2016 | | | 90.24 | 93.32 | 44.25 |
| CoMatch | 2021 | Images | Semi-Supervised | 82.55 | 85.70 | 38.29 |
| kNN | — | ResNet Features | Semi-Supervised | 63.67 | 76.80 | 36.67 |
| SVM | 1995 | | | 80.54 | 88.73 | 48.84 |
| OPF | 2009 | | | 71.77 | 83.56 | 38.59 |
| SL-Perceptron | — | | | 75.44 | 83.56 | 39.91 |
| ML-Perceptron | — | | | 78.88 | 87.10 | 32.24 |
| PseudoLabel+SGD | 2013 | | | 82.69 | 89.76 | 21.67 |
| LS+kNN | 2004 | | | 73.49 | 83.98 | 36.99 |
| LS+SVM | 2004 | | | 73.53 | 83.26 | 38.70 |
| LS+OPF | 2004 | | | 72.66 | 82.32 | 39.28 |
| LS+SL-Perceptron | 2004 | | | 72.34 | 82.38 | 39.21 |
| LS+ML-Perceptron | 2004 | | | 73.03 | 82.53 | 39.68 |
| GNN-LDS | 2019 | | | 54.98 | 62.69 | — |
| GNN-KNN-LDS | 2019 | | | 79.32 | 88.94 | 37.78 |
| WSEF+SVM+RBO | 2021 | | | 85.12 | 91.68 | 52.17 |
| SGC+Rec.+RDPAC | Ours | | | | | 84.53 |
| ManifoldGCN (best result) | Ours | | | 85.88 | 93.08 | 52.85 |
| kNN | — | SENet Features | Semi-Supervised | 48.71 | 58.78 | 22.23 |
| SVM | 1995 | | | 73.30 | 85.89 | 35.32 |
| OPF | 2009 | | | 64.00 | 81.33 | 30.94 |
| SL-Perceptron | — | | | 71.84 | 82.28 | 36.39 |
| ML-Perceptron | — | | | 72.62 | 86.90 | 32.15 |
| PseudoLabel+SGD | 2013 | | | 76.87 | 89.85 | 20.96 |
| LS+kNN | 2004 | | | 58.05 | 72.16 | 20.00 |
| LS+SVM | 2004 | | | 59.84 | 72.79 | 24.82 |
| LS+OPF | 2004 | | | 59.25 | 72.20 | 25.38 |
| LS+SL-Perceptron | 2004 | | | 59.27 | 72.19 | 25.41 |
| LS+ML-Perceptron | 2004 | | | 59.39 | 72.24 | 25.72 |
| GNN-LDS | 2019 | | | 52.24 | 65.80 | — |
| GNN-KNN-LDS | 2019 | | | 73.69 | 89.95 | — |
| WSEF+SVM+RBO | 2021 | | | 76.16 | 89.74 | 36.49 |
| SGC+Rec.+RDPAC | Ours | | | | | 76.93 |
| ManifoldGCN (best result) | Ours | | | 78.82 | 92.79 | 40.31 |
| kNN | — | ViT-B16 Features | Semi-Supervised | 91.91 | 81.19 | 56.62 |
| SVM | 1995 | | | 96.75 | 91.92 | 75.61 |
| OPF | 2009 | | | 96.50 | 90.02 | 73.27 |
| SL-Perceptron | — | | | 75.79 | 82.15 | 70.84 |
| ML-Perceptron | — | | | 92.59 | 74.41 | 12.02 |
| PseudoLabel+SGD | 2013 | | | 96.84 | 89.07 | 30.19 |
| LS+kNN | 2004 | | | 95.74 | 89.63 | 66.15 |
| LS+SVM | 2004 | | | 94.49 | 87.59 | 66.81 |
| LS+OPF | 2004 | | | 94.22 | 86.14 | 66.68 |
| LS+SL-Perceptron | 2004 | | | 93.71 | 86.31 | 65.45 |
| LS+ML-Perceptron | 2004 | | | 95.13 | 87.68 | 62.81 |
| GNN-LDS | 2019 | | | 72.03 | 56.33 | 22.75 |
| GNN-KNN-LDS | 2019 | | | 96.66 | 88.56 | 52.42 |
| WSEF+SVM+RBO | 2021 | | | 97.82 | 94.00 | 78.64 |
| SGC+Rec.+RDPAC | Ours | | | | | 97.11 |
| ManifoldGCN (best result) | Ours | | | 97.43 | 95.57 | 79.27 |

Table 5: Execution time (in seconds) for manifold learning methods and GCN approaches for both training and testing.

| | | Flowers | Corel5k | CUB200 |
|-------|------------------|---------------|----------------|-----------------|
| M.L. | LHRR | 1.10 ± 0.0012 | 6.21 ± 0.0017 | 20.16 ± 0.0108 |
| | RDPAC | 4.66 ± 0.0195 | 41.74 ± 0.0158 | 104.18 ± 0.5091 |
| | BFSTREE | 9.34 ± 0.0046 | 37.94 ± 0.1712 | 95.09 ± 0.0704 |
| Train | GCN-Net (kNN) | 0.76 ± 0.0187 | 2.23 ± 0.0178 | 6.95 ± 0.0141 |
| | GCN-Net (Rec.) | 0.61 ± 0.0016 | 1.57 ± 0.0018 | 4.40 ± 0.0003 |
| | GCN-SGC (kNN) | 0.15 ± 0.0005 | 0.20 ± 0.0011 | 0.54 ± 0.0006 |
| | GCN-SGC (Rec.) | 0.14 ± 0.0003 | 0.19 ± 0.0022 | 0.51 ± 0.0002 |
| | GCN-GAT (kNN) | 3.41 ± 0.0021 | 11.89 ± 0.0031 | 30.62 ± 0.0095 |
| | GCN-GAT (Rec.) | 2.44 ± 0.0025 | 7.90 ± 0.0029 | 19.35 ± 0.0043 |
| | GCN-APPNP (kNN) | 0.77 ± 0.0088 | 3.49 ± 0.0032 | 27.95 ± 0.0025 |
| | GCN-APPNP (Rec.) | 0.75 ± 0.0002 | 2.44 ± 0.0032 | 17.11 ± 0.0032 |
| | GCN-ARMA (kNN) | 3.84 ± 0.0064 | 14.45 ± 0.0215 | 47.8 ± 0.0146 |
| | GCN-ARMA (Rec.) | 2.80 ± 0.0051 | 9.94 ± 0.0013 | 30.51 ± 0.0113 |
| Test | GCN-Net (kNN) | 0.06 ± 0.0366 | 0.18 ± 0.0382 | 0.40 ± 0.0327 |
| | GCN-Net (Rec.) | 0.05 ± 0.0013 | 0.18 ± 0.002 | 0.44 ± 0.0029 |
| | GCN-SGC (kNN) | 0.04 ± 0.0015 | 0.16 ± 0.0011 | 0.38 ± 0.0051 |
| | GCN-SGC (Rec.) | 0.05 ± 0.0015 | 0.18 ± 0.0018 | 0.44 ± 0.0005 |
| | GCN-GAT (kNN) | 0.04 ± 0.001 | 0.15 ± 0.0009 | 0.38 ± 0.004 |
| | GCN-GAT (Rec.) | 0.05 ± 0.0015 | 0.18 ± 0.002 | 0.44 ± 0.0032 |
| | GCN-APPNP (kNN) | 0.05 ± 0.0015 | 0.15 ± 0.0008 | 0.38 ± 0.0045 |
| | GCN-APPNP (Rec.) | 0.05 ± 0.0015 | 0.18 ± 0.0021 | 0.44 ± 0.0026 |
| | GCN-ARMA (kNN) | 0.04 ± 0.0015 | 0.15 ± 0.0008 | 0.39 ± 0.0036 |
| | GCN-ARMA (Rec.) | 0.04 ± 0.0018 | 0.18 ± 0.0023 | 0.44 ± 0.0004 |

ACKNOWLEDGMENTS

The authors are grateful to Fulbright Commission, São Paulo Research Foundation - FAPESP (grants #2017/25908-6, #2018/15597-6, and #2020/11366-0), Brazilian National Council for Scientific and Technological Development - CNPq (grant #309439/2020-5), Petrobras (grant #2017/00285-6), and Microsoft Research for financial support. We also acknowledge the NSF Grant No. IIS-2107213.

References

Agarwal, M., Mostafa, J., 2011. Content-based image retrieval for alzheimer’s disease detection, in: 2011 9th International Workshop on Content-Based Multimedia Indexing (CBMI), pp. 13–18.

Amorim, W.P., Falcão, A.X., d. Carvalho, M.H., 2014. Semi-supervised pattern classification using optimum-path forest, in: 2014 27th SIBGRAPI Conference on Graphics, Patterns and Images, pp. 111–118.

Bai, S., Zhang, F., Torr, P.H.S., 2019. Hypergraph convolution and hypergraph attention. CoRR abs/1901.08150.

Berthelot, D., Carlini, N., Cubuk, E.D., Kurakin, A., Sohn, K., Zhang, H., Raffel, C., 2020. Remixmatch: Semi-supervised learning with distribution matching and augmentation anchoring, in: International Conference on Learning Representations.

Berthelot, D., Carlini, N., Goodfellow, I.J., Papernot, N., Oliver, A., Raffel, C., 2019. Mixmatch: A holistic approach to semi-supervised learning. CoRR abs/1905.02249. arXiv:1905.02249.

Bianchi, F.M., Grattarola, D., Livi, L., Alippi, C., 2021. Graph neural networks with convolutional arma filters. IEEE Transactions on Pattern Analysis and Machine Intelligence , 1–1.

Chaudhuri, U., Banerjee, B., Bhattacharya, A., 2019. Siamese graph convolutional network for content based remote sensing image retrieval. Computer Vision and Image Understanding 184, 22–30.

Chen, S.B., Tian, X.Z., Ding, C.H.Q., Luo, B., Liu, Y., Huang, H., Li, Q., 2020a. Graph convolutional network based on manifold similarity learning. Cognitive Computation 12, 1144.

Chen, T., Kornblith, S., Norouzi, M., Hinton, G., 2020b. A simple framework for contrastive learning of visual representations, in: Proceedings of the 37th International Conference on Machine Learning, JMLR.org.

Chen, Y., Li, J., Xiao, H., Jin, X., Yan, S., Feng, J., 2017. Dual path networks, in: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (Eds.), Advances in Neural Information Processing Systems 30. Curran Associates, Inc., pp. 4467–4475.

Chollet, F., 2017. Xception: Deep learning with depthwise separable convolutions, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1800–1807.

Cortes, C., Vapnik, V., 1995. Support-vector networks. Mach. Learn. 20, 273–297.

Datta, R., Joshi, D., Li, J., Wang, J.Z., 2008. Image retrieval: Ideas, influences, and trends of the new age. ACM Computing Surveys 40, 5:1–5:60.

Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L., 2009. ImageNet: A Large-Scale Hierarchical Image Database, in: CVPR09.

Deng, L., 2014. A tutorial survey of architectures, algorithms, and applications for deep learning. APSIPA Transactions on Signal and Information Processing 3.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N., 2021. An image is worth 16x16 words: Transformers for image recognition at scale, in: International Conference on Learning Representations.

van Engelen, J.E., Hoos, H.H., 2019. A survey on semi-supervised learning. Machine Learning 109, 373–440.

Franceschi, L., Niepert, M., Pontil, M., He, X., 2019. Learning discrete structures for graph neural networks, in: Proceedings of the 36th International Conference on Machine Learning.

Gionis, A., Indyk, P., Motwani, R., 1999. Similarity search in high dimensions via hashing, in: Proceedings of the 25th International Conference on Very Large Data Bases, p. 518–529.

Gong, C., Wang, D., Liu, Q., 2021. Alphamatch: Improving consistency for semi-supervised learning with alpha-divergence, in: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 13678–13687.

Guimarães Pedronette, D.C., Pascotti Valem, L., Latecki, L.J., 2021. Efficient rank-based diffusion process with assured convergence. Journal of Imaging 7.

He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770–778.

Hino, H., 2020. Active learning: Problem settings and recent developments. arXiv:2012.04225v2 .

Hu, J., Shen, L., Sun, G., 2018. Squeeze-and-excitation networks, in: 2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

Jiang, J., Wang, B., Tu, Z., 2011. Unsupervised metric learning by self-smoothing operator, in: 2011 International Conference on Computer Vision, pp. 794–801.

Karanam, S., Gou, M., Wu, Z., Rates-Borras, A., Camps, O., Radke, R.J., 2019. A systematic evaluation and benchmark for person re-identification: Features, metrics, and datasets. IEEE Transactions on Pattern Analysis and Machine Intelligence 41, 523–536.

Kherfi, M.L., Ziou, D., 2006. Relevance feedback for cbir: a new approach based on probabilistic feature weighting with positive and negative examples. IEEE Transactions on Image Processing 15, 1017–1030.

Kim, B., Choo, J., Kwon, Y., Joe, S., Min, S., Gwon, Y., 2021. Selfmatch: Combining contrastive self-supervision and consistency for semi-supervised learning. CoRR abs/2101.06480. arXiv:2101.06480.

Kipf, T.N., Welling, M., 2017. 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, OpenReview.net.

Klicpera, J., Bojchevski, A., Günnemann, S., 2019. Combining neural networks with personalized pagerank for classification on graphs, in: International Conference on Learning Representations.

Krizhevsky, A., Sutskever, I., Hinton, G.E., 2012. Imagenet classification with deep convolutional neural networks, in: Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1, pp. 1097–1105.

Laine, S., Aila, T., 2017. Temporal ensembling for semi-supervised learning, in: International Conference on Learning Representations.

Lee, D.H., 2013. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks, in: Workshop on challenges in representation learning, ICML.

Li, J., Xiong, C., Hoi, S.C.H., 2021. Comatch: Semi-supervised learning with contrastive graph regularization, in: 2021 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 9455–9464.

Li, M., Zhu, X., Gong, S., 2019. Unsupervised tracklet person re-identification.

- IEEE Transactions on Pattern Analysis and Machine Intelligence .
- Li, Q., Han, Z., Wu, X., 2018. Deeper insights into graph convolutional networks for semi-supervised learning, in: McIlraith, S.A., Weinberger, K.Q. (Eds.), Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), AAAI Press. pp. 3538–3545.
- Li, Y., Wu, Z., Karanam, S., Radke, R., 2014. Real-world re-identification in an airport camera network, pp. 1–6.
- Liu, C., Yu, G., Volkovs, M., Chang, C., Rai, H., Ma, J., Gorti, S.K., 2019. Guided similarity separation for image retrieval, in: Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R. (Eds.), Advances in Neural Information Processing Systems, Curran Associates, Inc.
- Liu, G.H., Yang, J.Y., 2013. Content-based image retrieval using color difference histogram. *Pattern Recognition* 46, 188 – 198.
- van der Maaten, L., Hinton, G., 2008. Visualizing data using t-sne. *Journal of Machine Learning Research* 9, 2579–2605.
- Mandal, D., Rao, P., Biswas, S., 2020. Semi-supervised cross-modal retrieval with label prediction. *IEEE Transactions on Multimedia* 22, 2345–2353.
- Miyato, T., Ichi Maeda, S., Koyama, M., Ishii, S., 2019. Virtual adversarial training: A regularization method for supervised and semi-supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 41, 1979–1993.
- Nie, L., Jiao, F., Wang, W., Wang, Y., Tian, Q., 2021. Conversational image search. *IEEE Transactions on Image Processing* 30, 7732–7743.
- Nie, L., Li, Y., Feng, F., Song, X., Wang, M., Wang, Y., 2020. Large-scale question tagging via joint question-topic embedding learning. *ACM Trans. Inf. Syst.* 38.
- Nilsback, M.E., Zisserman, A., 2006. A visual vocabulary for flower classification, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1447–1454.
- Omohundro, S.M., 1989. Five Balltree Construction Algorithms. Technical Report.
- Papa, J., Falcão, A., Suzuki, C., 2009. Supervised pattern classification based on optimum-path forest. *Int. J. Imaging Syst. Technol.* 19, 120–131.
- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E., 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research* 12, 2825–2830.
- Pedronette, D.C.G., Latecki, L.J., 2021. Rank-based self-training for graph convolutional networks. *Information Processing & Management* 58, 102443.
- Pedronette, D.C.G., Valem, L.P., Almeida, J., da S. Torres, R., 2019. Multimedia retrieval through unsupervised hypergraph-based manifold ranking. *IEEE Transactions on Image Processing* 28, 5824–5838.
- Pedronette, D.C.G., Valem, L.P., da S. Torres, R., 2021. A bfs-tree of ranking references for unsupervised manifold learning. *Pattern Recognition* 111, 107666.
- Piras, L., Giacinto, G., 2017. Information fusion in content based image retrieval. *Inf. Fusion* 37, 50–60.
- Presotto, J.G.C., Valem, L.P., de Sá, N.G., Pedronette, D.C.G., Papa, J.P., 2021. Weakly supervised learning through rank-based contextual measures, in: 2020 25th International Conference on Pattern Recognition (ICPR), pp. 5752–5759.
- Qin, D., Gammeter, S., Bossard, L., Quack, T., van Gool, L., 2011. Hello neighbor: Accurate object retrieval with k-reciprocal nearest neighbors, in: CVPR 2011, pp. 777–784.
- Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L., 2018. MobileNetV2: Inverted residuals and linear bottlenecks, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4510–4520.
- Schober, J.P., Hermes, T., Herzog, O., 2004. Content-based image retrieval by ontology-based object recognition, in: in KI-2004 Workshop on Applications of Description Logics, pp. 61–67.
- Sener, O., Savarese, S., 2018. Active learning for convolutional neural networks: A core-set approach, in: International Conference on Learning Representations.
- Sharma, A., Vans, E., Shigemizu, D., Boroevich, K.A., Tsunoda, T., 2019. Deepinsight: A methodology to transform a non-image data to an image for convolution neural network architecture. *Scientific Reports* 9, 11399.
- Sohn, K., Berthelot, D., Li, C.L., Zhang, Z., Carlini, N., Cubuk, E.D., Kurakin, A., Zhang, H., Raffel, C., 2020. Fixmatch: Simplifying semi-supervised learning with consistency and confidence, in: Proceedings of the 34th International Conference on Neural Information Processing Systems, Curran Associates Inc., Red Hook, NY, USA.
- Sultana, M., Gavrilova, M., 2013. A Content Based Feature Combination Method for Face Recognition. Springer International Publishing, Heidelberg. pp. 197–206.
- Tarvainen, A., Valpola, H., 2017. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results, in: Proceedings of the 31st International Conference on Neural Information Processing Systems, Curran Associates Inc., Red Hook, NY, USA. p. 1195–1204.
- Valem, L.P., Pedronette, D.C.G., Almeida, J., 2017. Unsupervised similarity learning through cartesian product of ranking references. *Pattern Recognition Letters* .
- Valem, L.P., Pedronette, D.C.G.a., Torres, R.d.S., Borin, E., Almeida, J., 2015. Effective, efficient, and scalable unsupervised distance learning in image retrieval tasks, in: Proceedings of the 5th ACM on International Conference on Multimedia Retrieval, ACM, New York, NY, USA. pp. 51–58.
- Vanyan, A., Khachatryan, H., 2021. Deep semi-supervised image classification algorithms: a survey. *JUCS - Journal of Universal Computer Science* 27, 1390–1407.
- Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y., 2018. Graph attention networks, in: International Conference on Learning Representations.
- Wah, C., Branson, S., Welinder, P., Perona, P., Belongie, S., 2011. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001. California Institute of Technology.
- Wang, M., Song, T., 2013. Remote sensing image retrieval by scene semantic matching. *IEEE Transactions on Geoscience and Remote Sensing* 51, 2874–2886.
- Wang, Z., Wang, S., Yang, S., Li, H., Li, J., Li, Z., 2020. Weakly supervised fine-grained image classification via gaussian mixture model oriented discriminative learning, in: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).
- Webber, W., Moffat, A., Zobel, J., 2010. A similarity measure for indefinite rankings. *ACM Trans. Inf. Syst.* 28.
- Wu, F., Souza, A., Zhang, T., Fifty, C., Yu, T., Weinberger, K., 2019. Simplifying graph convolutional networks, in: Chaudhuri, K., Salakhutdinov, R. (Eds.), Proceedings of the 36th International Conference on Machine Learning, PMLR. pp. 6861–6871.
- Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., Yu, P.S., 2021. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems* 32, 4–24.
- Xie, Q., Dai, Z., Hovy, E., Luong, M.T., Le, Q.V., 2020. Unsupervised data augmentation for consistency training, in: Proceedings of the 34th International Conference on Neural Information Processing Systems, Curran Associates Inc., Red Hook, NY, USA.
- Yang, J., Zheng, W.S., Yang, Q., Chen, Y.C., Tian, Q., 2020. Spatial-temporal graph convolutional network for video-based person re-identification, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3286–3296.
- Ye, M., Shen, J., Lin, G., Xiang, T., Shao, L., Hoi, S.C.H., 2020. Deep learning for person re-identification: A survey and outlook. *arXiv:2001.04193*.
- Yuan, L., Chen, Y., Wang, T., Yu, W., Shi, Y., Tay, F.E., Feng, J., Yan, S., 2021. Tokens-to-token vit: Training vision transformers from scratch on imagenet. *arXiv preprint arXiv:2101.11986* .
- Zhang, G., Pan, J., Zhang, Z., Zhang, H., Xing, C., Sun, B., Li, M., 2021. Hybrid graph convolutional network for semi-supervised retinal image classification. *IEEE Access* 9, 35778–35789.
- Zhou, D., Bousquet, O., Lal, T.N., Weston, J., Schölkopf, B., 2004. Learning with local and global consistency, in: Advances in Neural Information Processing Systems 16, MIT Press. pp. 321–328.
- Zhou, Z.H., 2017. A brief introduction to weakly supervised learning. *National Science Review* 5, 44–53.