Technische Universität Berlin

Faculty of Electrical Engineering and Computer Science
Dept. of Computer Engineering and Microelectronics
Remote Sensing Image Analysis Group

Semi-Supervised Learning for Remote Sensing Image Retrieval Based on Graph CNNs

Master of Science in Electrical Engineering

January, 2021

Xiaoxin Feng

Matriculation Number: 406088

Supervisor: Prof. Dr. Begüm Demir

Advisor: Tristan Kreuziger, Dr. Mahdyar Ravanbakhsh

Eidesstattliche Erklärung

Hiermit erkläre ich an Eides statt, dass ich die vorliegende Arbeit selbstständig und ohne fremde Hilfe angefertigt habe. Sämtliche benutzten Informationsquellen sowie das Gedankengut Dritter wurden im Text als solche kenntlich gemacht und im Literaturverzeichnis angeführt. Die Arbeit wurde bisher nicht veröffentlicht und keiner Prüfungsbehörde vorgelegt.

Hereby I declare that I wrote this thesis myself with the help of no more than the mentioned literature and auxiliary means.

Berlin, Date 16.01.2021

Xiaoxin Feng

....

Xiaoxin Feng

Acknowledgements

Upon the completion of this thesis, I am grateful to those who have offered me encouragement and support during my study.

First of all, special acknowledgment is given to my respectable supervisor Prof. Dr. Begüm Demir whose patient instruction and constructive suggestions are beneficial to me a lot. She introduced me to explore the field of remote sensing, and I really enjoy doing research in this field. Her insightful comments on my thesis and experiments provided me many enlightening ideas. The thesis could not be finished without her patient guidance.

Secondly, particular thanks go to my advisors Tristan Kreuziger and Dr. Mahdyar Ravanbakhsh. In the past half year, they have given me a lot of help. They gave me patient and meticulous guidance through many meetings and emails. Regarding my drafts, they carefully revised it sentence by sentence and put forward many opinions. I am very grateful for their help and am very happy to work with them.

Lastly, I would like to thank my friends and family. They always support me and give me as much help as they can.

Abstract

The development of various earth observation missions in recent years has led to a large amount of remote sensing (RS) image archives. The resulting massive volumes of RS images require new methods for content-based image retrieval (CBIR) to enable accurate search and retrieval of relevant images. Deep learning based methods have been widely applied for CBIR due to their prominent capability of characterizing complex RS images. However, deep learning based methods require a large amount of annotated RS images, which is difficult because the manual annotation is time-consuming and laborious. To address this problem, we first propose novel semi-supervised methods based on graph convolutional neural networks (Graph CCNs) for content-based image retrieval (CBIR) method from remote sensing (RS) image archives. The propagation of labels through the graph allows the model to use the unlabeled images as an additional input. Secondly, we propose a novel Triplet Graph Convolutional Network (TGCN) with a Graph-based Triplet Sampling (GTS) strategy. The TGCN consists of three parallel graph models with shared weights and learns a representation from triplets of images suitable for image retrieval. The GTS relies on the graph representation to select triplets considering their similarity learned by the graph. The TGCN shows superior performance in learning the graph representationsâ metric space. The proposed GTS enables exploiting the implicit similarity information within the graph structure to select hard triplets, which are beneficial for the efficiency of the training. Thirdly, we generalize the semi-supervised methods based on Graph CNNs to multi-label RS image retrieval scenarios. Experimental results show the effectiveness of the proposed methods for semi-supervised RS image retrieval.

Zusammenfassung

Die Entwicklung verschiedener Erdbeobachtungsmissionen in den letzten Jahren hat zu einer Vielzahl von Fernerkundungsbildarchiven (RS) geführt. Die daraus resultierenden massiven Mengen an RS-Bildern erfordern neue Methoden für das inhaltsbasierte Abrufen von Bildern (CBIR), um eine genaue Suche und das Abrufen relevanter Bilder zu ermöglichen. Deep-Learning-basierte Methoden wurden für CBIR aufgrund seiner herausragenden Fähigkeit zur Charakterisierung komplexer RS-Bilder in großem Umfang angewendet. Deep-Learning-basierte Methoden erfordern jedoch eine große Anzahl kommentierter RS-Bilder, was schwierig ist, da die manuelle Annotation zeitaufwändig und mühsam ist. Um dieses Problem anzugehen, schlagen wir zunächst neuartige halbüberwachte Methoden vor, die auf Graph Convolutional Neural Networks (Graph CCNs) für die inhaltsbasierte Bildabrufmethode (CBIR) aus Fernerkundungsbildern (RS) basieren. Durch die Weitergabe von Beschriftungen durch das Diagramm kann das Modell die unbeschrifteten Bilder als zusätzliche Eingabe verwenden. Zweitens schlagen wir ein neuartiges Triplet Graph Convolutional Network (TGCN) mit einer graphbasierten Triplet Sampling (GTS) -Strategie vor. Das TGCN besteht aus drei parallelen Diagrammmodellen mit gemeinsamen Gewichten und lernt eine Darstellung aus Tripletts von Bildern, die zum Abrufen von Bildern geeignet sind. Das GTS stützt sich auf die Diagrammdarstellung, um Tripletts unter Berücksichtigung ihrer durch das Diagramm erlernten Ähnlichkeit auszuwählen. Das TGCN zeigt eine überlegene Leistung beim Lernen des metrischen Raums der Diagrammdarstellungen. Das vorgeschlagene GTS ermöglicht die Nutzung der impliziten Ähnlichkeitsinformationen innerhalb der Diagrammstruktur, um harte Tripletts auszuwählen, die für die Effizienz des Trainings von Vorteil sind. Drittens verallgemeinern wir die halbüberwachten Methoden, die auf Graph CNNs basieren, auf RS-Bildabrufszenarien mit mehreren Etiketten. Die experimentellen Ergebnisse zeigen die Wirksamkeit der vorgeschlagenen Methoden zur halbüberwachten RS-Bildwiederherstellung.

Contents

List of Acronyms

RS	Remote Sensing
DL	Deep Learning
TBIR	Text-Based Image Retrieval
CBIR	Content-Based Image Retrieval
CNN	Convolutional Neural Network
GNN	Graph Neural Network
GCNN	Graph Convolutional Neural Network
ChebNet	Chebyshev Spectral CNN
GCN	Graph Convolutional Network
DCNN	Diffusion Convolutional Neural Network
MoNet	Mixture Model Networks
MPNN	Message Passing Neural Networks
GAT	Graph Attention Network
SAGE	Sample and Aggregate
ConfGCN	Confidence-based Graph Convolutional Networks
BoVW	Bag-of-Visual-Words
SIFT	Scale-invariant Feature Transform
LDA	Linear Discriminative Analysis
SVM	Support Vector Machine
KNN	K-Nearest Neighbor
PCA	Principle Component Analysis
CCA	Canonical Component Analysis
LLE	Locally Linear Embedding
LLP	Locally Preserving Projections
NLP	Neural Language Process
SGD	Stochastic Gradient Descent
GPU	Graphics Processing Unit
GTS	Graph-based Triplet Sampling
TGCN	Triplet Graph Convolutional Networks
RTS	Random Triplet Sampling
AID	Aerial Image Dataset
BATM	Batch All Triplet Mining
MAP	Mean Average Precision
WMAP	Weighted Mean Average Precision
ACG	Average Cumulative Gain
BCE	Binary Cross Entropy
CE	Cross Entropy

HOG	Histogram of Oriented Gradient
LBP	Local Binary Pattern
GLCM	Gray Level Co-occurrence Matrices
VLAD	Vector of Locally Aggregated Descriptors

List of Figures

List of Tables

1 Introduction

1.1 Motivation

With the development of various earth observation missions, a large amount of remote sensing (RS) images are available for many important applications, including disaster monitoring, land-cover and land-use classification, and urban mapping. Such massive volumes of RS image archives require vigorous blooming of content-based image retrieval (CBIR) methods for efficiently and accurately searching and retrieving relevant images, given the query images.

Deep learning (DL) based methods have recently seen a huge raise in popularity and due to their powerful capabilities for extracting features from semantically complex RS images they have been been applied to many CBIR tasks. Most DL-based CBIR methods in the literature exploit supervised information (e.g., semantic labels) to guide the learning of the underlying DL model [71, 6, 58]. To achieve scalable and accurate image retrieval, Roy et al. [50] introduce the metric learning based deep hashing network that learns hash codes for RS images. The network learns an embedding space, which induces a metric by receiving triplets of images as input. However, such supervised learning requires large-scale labeled RS images, which require huge amount of human efforts. Compared with supervised learning, semi-supervised learning can effectively overcome such limitation by using an approach of learning the global structure of the dataset from both the labeled and the unlabeled datasets.

Semi-supervised learning methods are divided into two aspects, transductive learning and inductive learning. In the transductive learning methods, the classifier is constructed only for those images that have been utilized in the training phase. In the inductive learning methods, the classifier can predict the new images which are previously unseen when training the classifier. There are various graph-based semi-supervised learning methods, the key idea of the existing methods is building a graph whose nodes represent the labeled and unlabeled images, the nodes with unlabeled images can be labeled by propagation from the labeled nodes through the graph.

In particular, one of the state-of-art DL methods for image retrieval is based on triplet loss, which takes two similar images and one dissimilar image as input. Provided with such triplets, the loss function drives the learning of a feature embedding space, which is suited for image search and retrieval applications. It does so by forcing the distance between similar images to be smaller than the distance between dissimilar image by at least a margin [53]. However, using triplets that are obtained from random sampling may contain many easy triplets, which already satisfy the optimization constraint. This reduces the training efficiency and stalls the training progress. It is therefore important to efficiently mine hard triplets to help training and improve performance.

Graph neural networks (GNNs) are first proposed by Gori et al. [19] and Scarselli et al. [52] as recurrent neural networks. In the GNN, the information on each node can be passed to the neighbor nodes according to the topological structure of the GNN and the node representations reach a stable state after training. Duvenaud et al. [16] propose a convolutional neural network based on graph-structured data to achieve end-to-end learning for predictions leading to graph neural networks (GNNs). Bruna et al. [5] use smooth spectral multiplier for the spectral construction of deep neural network on graphs. Kipf and Welling [29] present the first adoption of a GNN for semi-supervised classification. Recently a Siamese graph convolution network is proposed to learn the discriminative feature space for CBIR in RS [9]. This network consists of two GCN models for measuring the similarity between a pair of graphs. Such an architecture can learn the metric space and pull similar images closer, while pushing dissimilar images away. However, it cannot capture the potentially high-order correlations between samples and correctly model the inter- and intra-class similarities of fine-grained datasets with large numbers of classes.

Due to the sufficiency of the Graph CNNs applied in semi-supervised learning, we can implement Graph CNNs to address the issues mentioned above by using only few annotated RS images and we want to investigate the performance of Graph CNNs methods for RS image retrieval.

1.2 Objective

Supervised learning relies on a large amount of labeled RS images, which require a huge amount of human effort and are expensive. In contrast to this, semi-supervised learning can effectively overcome such limitations by learning the global data structure from both labeled and unlabeled images. Various graph-based semi-supervised learning methods are proposed in the literature. The key idea of the existing methods is to build a graph, with the nodes representing the images and the edges expressing relations between images. Labels can be assigned to the nodes of unlabeled images by propagating the labels from nodes through the graph.

Therefore, we plan to construct a Graph Convolutional Neural Network, which implements label propagation method on the inherent graph structure among the images based on the learned CNN models. The Graph CNNs methods can be beneficial for discovering the inherent data structure among all the images in a scalable archive, such as BigEarthNet [59], with only a small subset of annotated images.

In this thesis, we aim to show how GCN contributes to semi-supervised learning by propagating label information to unlabeled data and sampling suitable hard triplets. The semi-supervised nature of the method allows it to benefit of the inherent data structure among all the images in a large-scale archive with only a small subset of annotated images. Moreover, GCN can help to explore the similarity relations between images from same or different classes, so that hard triplet can be selected. Compared with the easy triplet mentioned above, each hard triplet consists of a dissimilar positive and a similar negative image with regard to query image. It can improve the training efficiency of the metric learning.

We address the lack of huge amounts of labeled images and difficulty in semi-supervised RS image retrieval in this thesis by proposing novel semi-supervised RS image retrieval frameworks based on Graph CNNs. First, we develop a semi-supervised GCN model for RS image retrieval to learn image similarities by relying only on a few annotated images. Second, we propose a new Triplet Graph Convolutional Networks with a novel GCN-based triplet sampling strategy to improve the efficiency of graph-based metric learning. Third, we propose a new semi-supervised GCN for multi-label RS image retrieval to generalize the GCN-based methods into multi-label scenarios. This work shows the effectiveness of GCN-based methods on semi-supervised RS image retrieval, which can achieve a better performance with only few annotated images compared with state-of-art CNN-based methods.

1.3 Outline

This example thesis is separated into 7 chapters. A brief introduction is given bellow.

Chapter 2 introduces the related works with regard to graph convolutional neural networks and image retrieval. In the first section, image retrieval methods are introduced, which are divided into two categories: 1) text-based image retrieval (TBIR); 2) content-based image retrieval (CBIR). In the second section, the development and state-of-art remote sensing image retrieval methods are presented.

Chapter 3 introduces the background knowledge of graph convolutional neural networks (GCNN) and semi-supervised learning. First, basic knowledge and assumptions for semi-supervised learning are introduced. Then several representative semi-supervised methods are presented including self-training, co-training, generative methods, and graph-based methods. After that, the details of GCNN are introduced. GCNN are divided into two streams, spectral-based GCNN and spatial-based GCNN. Moreover, several representative GCNN methods are introduced, such as Spectral Convolutional Neural Network (Spectral CNN) [5], Chebyshev Spectral CNN (ChebNet) [13], Graph Convolution Network (GCN) [29], Diffusion Convolutional Neural Network (DCNN) [1], Message Passing Neural Networks (MPNN) [17], Graph Attention Network (GAT) [67].

Chapter 4 introduces the proposed semi-supervised GCN model for RS image retrieval. The architecture of the proposed model is presented, which consists of 1) an embedding CNN for feature extraction and graph construction; 2) a GCN which can propagate the label information from labeled data to unlabeled data by means of graph convolution. The loss functions and the metrics for image retrieval are introduced. In addition, three RS benchmarks are introduced, i.e. AID [69], NWPU-RESISC45 [10], and EuroSAT [23]. The proposed models are evaluated by comparison with CNN-based methods in a semi-supervised setup. The experimental results are presented and analyzed.

Chapter 5 introduces the proposed TGCN model and GTS strategy. The architecture of TGCN and the loss function are presented. Specifically, the detailed derivation of GTS is provided. Experiments are conducted on three RS benchmarks in comparison with state-of-art BATM method to evaluate the effectiveness of the proposed methods. Afterwards, the experimental results are presented and analyzed.

Chapter 6 introduces the proposed semi-supervised GCN model for multi-label RS image retrieval. The network architecture and loss function are introduced. Besides, the introduction to the multi-label RS image dataset, i.e. BigEarthNet [59] is provided. Afterwards, experimental results and analysis are presented.

Chapter 7 summarizes all of the proposed methods and analyzes the experimental results. The effectiveness and novelty of the proposed methods are discussed. In addition, several directions of future work are presented, such as generalize the proposed TGCN and GTS into multi-label scenario for RS image retrieval.

2 Related Work

In this chapter, some related work is introduced and discussed. Section 2.1 introduces the categories of image retrieval methods including text-based image retrieval and content-based image retrieval. Section 2.2 presents the methods related to remote sensing image retrieval. Section 2.3 introduces some of the methods based on graph neural network which are effective on image retrieval.

2.1 Image Retrieval Methods

In recent years, a huge number of images have been created and collected in many areas including media, academy, education, finance, etc. However, how to make use of the images and how to access the desired information from them such as effectively and accurately search for the needed or interested image from the large image datasets has become a hot topic in the field of image process, information retrieval, remote sensing, etc.

Image retrieval methods can be divided into two categories according to the different ways of describing the image, which are text-based image retrieval (TBIR) and content-based image retrieval (CBIR).

2.1.1 Text-based Image Retrieval

The studies on the text-based image retrieval (TBIR) method started in the 1970s. TBIR describes the content of image using text annotations, and generates keywords or descriptions for each image to describe the image content, such as the objects or scenes in the image. The image can also be annotated by some metadata such as the image format, the image size and image name. TBIR can be implemented by manually annotating or annotating with the help of image recognition techniques.

Users can provide query keywords for retrieval according to their interests, then the retrieval system can find the images annotated with the query keywords, and returns the query result to the user.

The text-based retrieval method is easy to implement, and normally can achieve a high accuracy due to the manual annotation. Therefore it is usually used in some small-scale image retrieval scenarios. However, the text-based method has some well-known drawbacks. First, the method is highly dependent on manual annotation, which is time-consuming and also takes a lot of financial resources on large-scale image datasets. Therefore, it is only practicable on small-scale image data. Second, the images can consist of much content and details, therefore, it is difficult for a user to accurately describe the desired image by some keywords. Moreover, the manual annotation is limited or affected by the cognition level and language of the annotator. Different annotators may give different descriptions to the same image.

Figure 2.1 shows the framework of a typical text-based image retrieval system. The images are annotated manually by the database manager with text-based annotations, then the user can retrieve all the images according to the given text annotations.

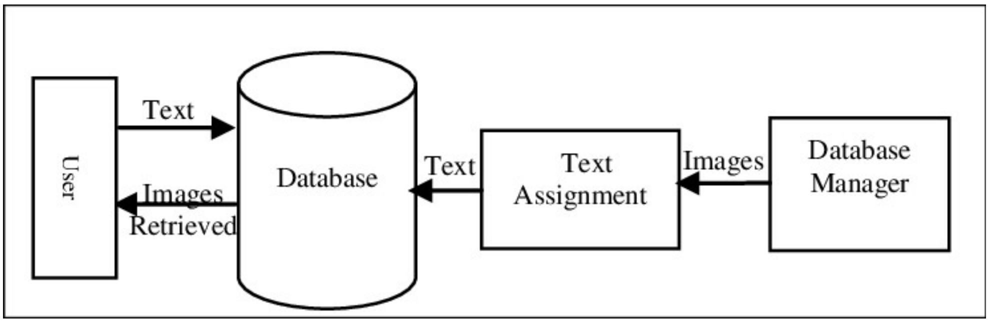


Figure 2.1: Text-based Image Retrieval

2.1.2 Content-based Image Retrieval

Along with the rapidly increased size of image archives, the drawbacks of TBIR have become increasingly serious. In 1992, the National Science Foundation of the United States has discussed the development of the image database management system and stated that the most effective way for image retrieval should be based on the image content.

A typical framework of a CBIR system is illustrated in Figure 2.2. The system analyses the images in the database and extracts the image features, then constructs the vector descriptions of the image features and stores the descriptions into the image features database. When the user inputs a query image, the same feature extraction method is used to process the image to obtain the query vector. Then a certain similarity measure is used to compute the similarity between the query vector and each feature in the feature database. Finally, the images are sorted according to the similarity and output in order.

CBIR methods are first proposed in early 1990s by Kato [28], which can be used to retrieve images from a database based on the color and shape. Before 2003, the global feature based on texture and color has been widely used by the traditional CBIR methods as the feature representation of the image, such as color histogram, color correlogram, Wavelet transform, GIST [41], Edgel [7], ect. Color features and texture features are simple to calculate and can provide reasonable representations of the image. However, the ability of these kind of global features to represent the image content is limited, they are mainly used for image copy retrieval.

Similar to the Bag-of-Word model in NLP but uses image features as the words, the Bag-of-Visual-Words (BoVW) model [55] is proposed in 2003. Thanks to the Scale-invariant feature transform (SIFT) [35], the BoVW can be widely used for image retrieval. SIFT can detect the local areas with significant visual characteristics in the image, and generate feature representations

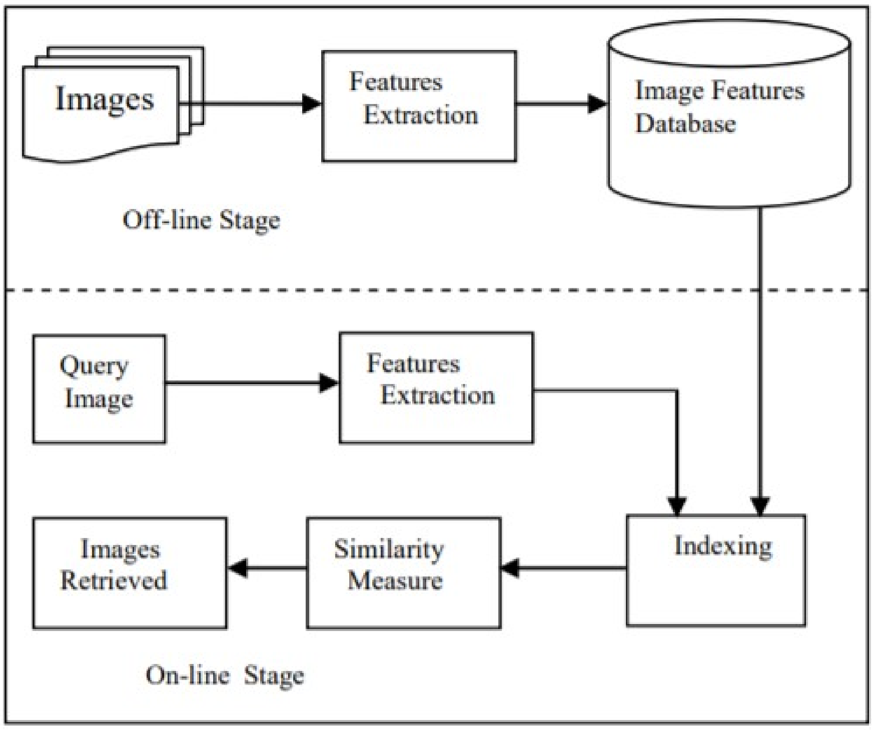


Figure 2.2: Content-based Image Retrieval

with stable description capabilities for these areas. Therefore, it can be effectively used for constructing visual words to describe the image. The local visual features represented by SIFT have good resistance to geometric transformations such as translation, scaling and rotation. Therefore, the visual retrieval methods can be effectively applied to more application scenarios such as similar image retrieval, instance retrieval, etc. After the BoVW and SIFT, many methods which based on local features and visual words are proposed. For example, clustering methods such as hierarchical k-Means [40] and approximate k-Means [46] are used to generate visual vocabulary from a large set of image descriptors, Fisher Vector [45] and VLAD [27] can be used to aggregating the local image descriptors, Hamming embedding [25], product quantization [26] and scalar quantization [74] are used to match or quantize the descriptors. These local descriptor-based methods can significantly benefits the image feature extraction, and thus are widely used in image retrieval.

With the development of deep learning, the image retrieval techniques has accordingly changed in recent years. Deep neural networks can can simulate the neural mechanisms of humans and extract the high-level features to produce an abstract representation of the image. Convolutional Neural Networks (CNN) are the most popular model in visual representation. Thanks to the local respective field and weight sharing, CNN can map the original image into an abstract semantic representation with limited parameters. A number of representative CNN models have been proposed, such as VGGNet [54], GoogleNet [60], ResNet [22], AlexNet [30], etc. The deep learning techniques have also been widely used in image segmentation, image classification, object recognition and many other computer vision tasks.

Training a deep convolutional neural network requires a large amount of labeled data, which is difficult to access for many specific computer vision tasks. However, the response of the middle layers of a CNN trained with a large amount of samples already has the capability to effectively extract the image features and represent the image content [42]. Therefore, it is possible to fine-tune the pre-trained CNN to apply the model in different tasks via transfer learning. The off-the-shelf CNN trained on benchmark datasets is also easily available or applicable to different specific image retrieval scenarios such as clothes retrieval, vehicle retrieval and remote sensing retrieval. Many studies also focus on distance metric learning for the image retrieval task. A proper metric can be learned from the image dataset to effectively represent the similarity of the images by distance.

Many studies focus on how to utilize the features obtained from the upper layers of the deep CNN as a descriptor for image retrieval. Babenko et al. [2] uses neural codes as the high-level descriptors and retrieves images based on the feature distance. Gong et al. [18] developed the idea that the global deep CNN contains too much spatial information, which leads to its lack of invariance to the geometric transformation when performing image retrieval. Therefore, the multi-scale orderless pooling (MOP-CNN) is proposed to extract the CNN activation from multiple scales and multiple local areas on the image, then VLAD pooling is performed to the obtained feature representation. In addition, the robustness of global descriptors provided by CNN can also be improved by utilizing integral max-pooling method on the convolutional layers [63].

2.2 Remote Sensing Image Retrieval

In recent decades, with the rapid development of aerospace technology and sensor technology, remote sensing and earth observation technology has greatly improved. A large number of remote sensing images are acquired by the satellite sensors and stored in large-scale image databases. Thanks to the explosive growth of remote sensing data, remote sensing has been widely used in many field such as urban planning, disaster management, environment monitoring. How to quickly and effectively acquire target information and retrieve required images has attracted more and more research interests.

The remote sensing image retrieval methods are mainly divided into two categories, which are text-based RS image retrieval and content-based RS image retrieval.

Text-based RS image retrieval is commonly used for remote sensing image retrieval in the early years. It requires manual annotations for each RS image and describe the image with relevant text information such as geographic regions, acquisition time and image name, then match the query image with the images in the database according to the descriptions. As we mentioned in Section 2.1.1, manual annotating requires a lot of time and financial resources, and can easily be affected by the subjective recognition of the annotators. Therefore, it has gradually been replaced by content-based RS image retrieval.

Content-based RS image retrieval can extract the image feature and measure the similarity between query feature and features in the image feature database, then retrieve the image according to the similarity. A content-based RS image retrieval system consists of feature extraction, feature indexing and feature similarity measuring.

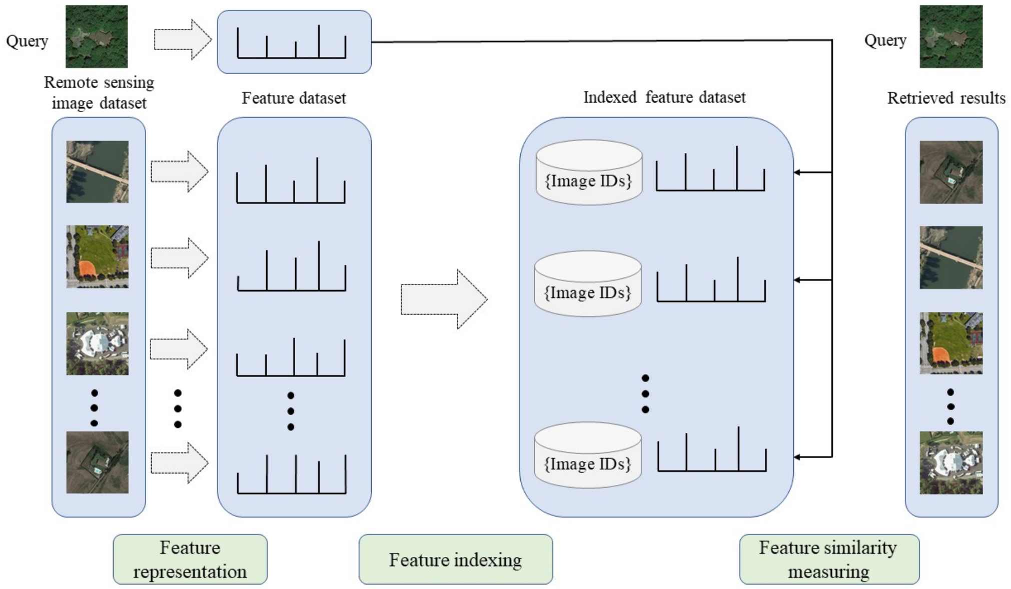


Figure 2.3: Illustration of conventional content-based RS retrieval

The conventional content-based RS image retrieval system mainly extracts low-level features including spectral features, texture features, shape features to describe the semantic content of the remote sensing image. Compared with natural images, remote sensing images normally have multiple or even hundreds of bands. As one of the simplest features, the spectral can describe the most intuitive feature information of the RS images [44]. Texture is understood as the ordered structure of repeated pixels, which is normally adopted in the RS image retrieval as a single feature or the combination of many features. The common texture descriptors include gray level co-occurrence matrices (GLCM) [21], wavelet [37, 3], Gabor filters [12, 38], etc. Shape feature is used to depict the outline or region information of the object, which is usually adopted in RS object recognition or image retrieval [14, 36]. However, it does not have the ability to capture the spatial relation information. In addition to the above-mentioned low-level features, there are some other local and global descriptors including SIFT [35], Histogram of Oriented Gradient (HOG) [11], and Local Binary Pattern (LBP) [47, 62]. As an effective local feature descriptor, SIFT can maintain the invariance to the scaling and rotation transformation. HOG is a global descriptor which extract the feature by compute and count the histogram of oriented gradient in the portion of the image. LBP encode and extract feature representations by comparing the pixel value with neighboring pixels in a local window, and has gray scale and rotation invariance.

Compared with low-level features, middle-level features capture the semantic content of the RS images by embedding the low-level feature descriptors into the visual vocabulary space. The common encoding methods include Bag-of-Words (BoW) [55], Fisher Vector [45] and vector of locally aggregated descriptors (VLAD) [27]. Compared with low-level features, BoW and VLAD show the effectiveness in depicting the image content [61] and can achieve a high precision in RS image retrieval [4]. According to [43], VLAD can reach a higher retrieval accuracy while BoW has better computational speed. VLAD can be further improved by aggregating deep local convolutional features to produce a global descriptor [24].

However, there is still a ”semantic gap” between low- and mid-level features and high-level semantics. Due to the limitation of the handcraft descriptors, the semantic content of the RS images can not be effectively depicted. In recent years, deep learning has been widely used and achieved great success in many field including RS image retrieval. CNN is adopted by Zhou et al. [73] in RS image classification and generate the low-dimensional feature representation of the high-resolution RS images through global average pooling. Kumar et al. [31] use CNN to extract the features of buildings in RS images, train neural networks through classification, and retrieve RS images based on the features extracted from the network. Xiong et al. [70] introduce attention mechanism to CNN and cause more attention to salient features to generate discriminative for RS image retrieval. Imbriaco et al. [24] use VLAD to aggregate attentive, local convolutional features to produce a global feature representation and achieve a faster and more accurate image retrieval.

Due to the massive amount of RS images, the regular indexing methods are not able to meet the requirements of large-scale RS image retrieval. The Approximate Nearest Neighbor (ANN) can greatly improve retrieval efficiency with a relative high image retrieval accuracy. Hashing methods have been widely adopted in large-scale RS image retrieval due to its advantages in time-efficiency and storage capability [15]. Hashing methods initially find the mapping function from the original feature space to the Hamming space. Therefore, similar data in original feature space have similar binary hash codes in the Hamming space. Hashing methods are mainly divided into supervised hashing and unsupervised hashing which learns the hash functions from unlabeled data. Among them, Locality Sensitive Hashing (LSH) [56] is the most representative unsupervised hashing method, which implements the mapping through random binary projection and can effectively fast the retrieval process. However, LSH can only achieve a high retrieval accuracy with long hash codes, which leads to larger storage requirements and worse retrieval efficiency.

In recent years, deep learning methods are adopted into hash coding to achieve a better retrieval performance. Lin et al. [34] proposed Deep Learning of Binary Hash Codes (DLBHC) method, which inserts a coding layer between the fully connected layer and the classification layer of the CNN to obtain the binary hash codes which contains semantic information. Li et al. [33] proposed a novel hashing method named partial randomness hashing (PRH) for the large-scale RS image retrieval. PRH first generates random projections to map the image into Hamming space to obtain the low-dimensional representation and train the feature extraction model. Reao et al. [49] introduce a unsupervised strategy which use multi-hash codes to represent the RS image and improve the performance of large-scale RS image retrieval. Roy et al. [51] proposed a deep metric and hash-code learning network to learn the metric space for RS image retrieval and simultaneously produce binary hash codes to perform a efficient archive search.

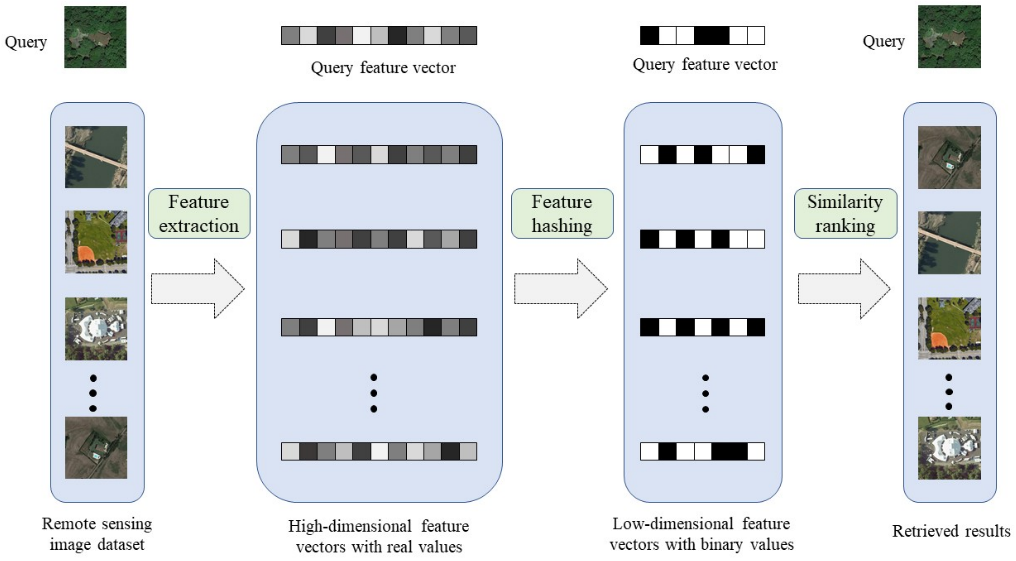

Query
Query feature vector
Query feature vector
Query
Feature extraction
Feature hashing
Similarity ranking
Remote sensing image dataset
High-dimensional feature vectors with real values
Low-dimensional feature vectors with binary values
Retrieved results

Figure 2.4: Illustration of hashing-based RS retrieval

3 Fundamentals of Graph Convolutional Neural Networks

Attributed to the rapidly developing computational capability and a large amount of training data, deep learning has recently been applied to many machine learning tasks. Compared to traditional machine learning techniques which usually rely on handcrafted features, deep learning can effectively extract informative features from Euclidean structure data. In section 3.1 fundamentals and representative algorithms of semi-supervised learning are presented. The graph convolutional neural networks including spectral-based and spatial-based graph convolutional neural networks are introduced in section 3.2.

3.1 Semi-supervised Learning

In machine learning, there are two major learning approaches, supervised learning and unsupervised learning. Supervised learning performs the learning task using a set of data in which each data has a certain corresponding labels. Representative methods that can work in a supervised setup include LDA(Linear Discriminative Analysis), PLS(Partial Least square), SVM(Support Vector Machine), KNN(K-Nearest Neighbor), Naive Bayes, Logistic Regression, Decision Tree and Neural Network. Unsupervised learning methods aim at automatically classifying or grouping the input data without specific labels by inferring the underlying similarity between the input data. Representative methods that can work under a unsupervised setup include K-Means, Hierarchical Clustering, PCA(Principle Component Analysis), CCA(Canonical Component Analysis), ISOMAP(Isometric Feature Mapping), LLE(Locally Linear Embedding) and LLP(Locally Preserving Projections).

Usually, deep learning models are data-hungry Nevertheless. manual labeling is an expensive and time-consuming process. On the other hand, only using unsupervised methods with unlabeled data usually can not achieve desirable performance Semi-supervised learning is a branch of Machine learning between unsupervised learning and supervised learning Semi-supervised learning methods aim at using both labeled and unlabeled data. This is applicable for many Machine Learning cases, such as Text Classification Computer Vision, NLP(Neural Language Process) where a large amount of unlabeled data are easily accessible as well as some annotated samples. Therefore, researchers try to combine the large amounts of unlabeled with the limited number of labeled data to train together for learning, hoping to improve the learning performance. Semi-supervised learning can make full use of data, and at the same time improves the generalization ability of supervised learning and the learning accuracy of unsupervised learning.

3.1.1 Assumptions for Semi-supervised Learning

In order to make use of the unlabeled data and improve the learning accuracy, the underlying distributed data must satisfy the condition that the data distribution contains the information of the posterior distribution. Therefore, semi-supervised learning relies on three assumptions: smoothness assumption, cluster assumption and manifold assumption [65].

(1) Smoothness Assumption: The labels of two closely spaced samples in the high-density regions are similar, that is, when two samples are connected by edges in the high-density regions, they have the same label with a high probability; Conversely, when two samples are separated by low-density regions, they tend to be in different classes.

(2) Cluster Assumption: When two samples are in the same cluster, they share a label with a high probability. The equivalent of this assumption is defined as Low Density Separation Assumption [8], that is, the decision boundary should pass through the low-density regions, and avoid dividing the samples in the high-density regions on both sides of the decision boundary.

(3) Manifold Assumption: The high-dimensional data is embedded in the low-dimensional manifold. When two samples are located in a small local neighborhood in the low-dimensional manifold, they have similar class labels.

3.1.2 Semi-supervised Learning Methods

Semi-supervised learning includes transductive learning and inductive learning. Transductive learning only processes the given training data of the sample space and trains the model to predict the label of the unlabeled sample only based on the labeled and unlabeled sample of the training data, whereas inductive learning processes all the given and unknown samples in the entire sample space. The labeled and unlabeled samples of training data are trained together with the unknown test samples to predict the labels of the samples in training and test data.

Self-training

Self-training methods is the most fundamental pseudo label method [64]. The basic assumption of self-training methods is that the samples with high confidence are more likely to be classified correctly when the classifier predicts the labels of the samples. In the beginning, the supervised classifier is trained only based on the labeled data. The trained classifier is used to predict the unlabeled data and produce pseudo labels. The obtained pseudo-labeled data with high confidence is added into the labeled training set. Then the classifier is iteratively trained based on the original labeled and new pseudo-labeled data until there is no more unlabeled data.

Co-training

In co-training, two classifiers are trained iteratively based on the labeled data. In each iteration, each trained classifier is used to predict the unlabeled data. The most confident predictions are added into the labeled data sets of the other classifier. Then the labeled and unlabeled data sets are updated. The classifiers are trained iteratively until the data sets do not change. Based on disagreement-based methods, different classifier provide different predictions for the unlabeled data. Therefore, both classifiers providing useful information for each other and by exchanging learned information through unlabeled data. Co-training has been successfully applied in many fields such as neural language processing [68], semi-supervised image recognition [48] and classification [72].

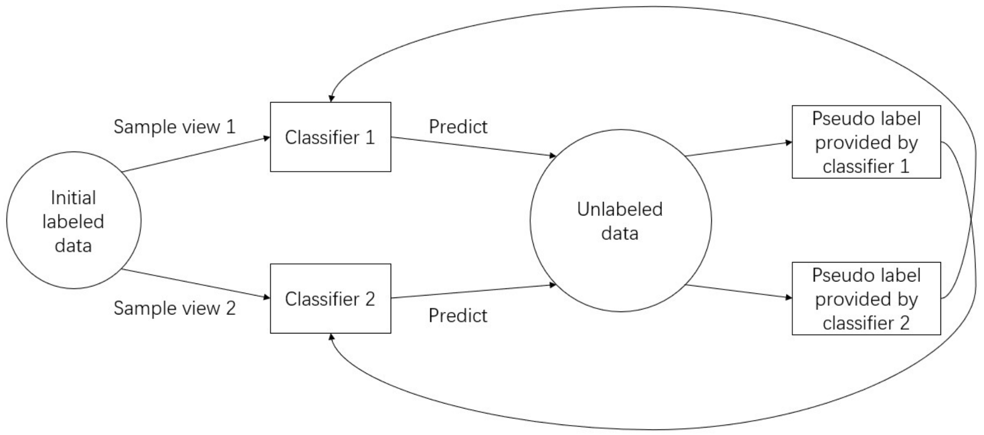


Figure 3.1: Co-training

Generative Methods

The generative methods assume that samples and labels are generated by a certain set of probability distributions or structural relationship. We can obtain the prior distribution $p(x)$ and the conditional distribution $p(x|y)$. Then we can compute the posterior probability $p(y|x)$ according to the Bayes' theorem, and label the sample x by the corresponding y with the maximum $p(y|x)$.

There are many models to generate the samples, such as Gaussian model, Bayesian Network, Sigmoidal Belief Networks (SBN), Gaussian Mixture Model (GMM) and Hidden Markov Model (HMM) [65].

(1)The sample distribution of the Gaussian model is:

$$p(x|y) = N(x|\mu, \sum) = \frac{1}{(2\pi)^{D/2}|\sum|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu)^T \sum^{-1}(x-\mu)\right) \tag{3.1}$$

(2)The sample distribution of the Bayesian Network is as Figure 2.2:

(3)The sample distribution of the SBN is:

$$p(x_i|pa(x_i)) = \frac{\exp((\sum_j J_{ij}x_j + h_i)x_i)}{1 + exp(\sum_j J_{ij}x_j + h_i)} \tag{3.2}$$

(4)The sample distribution of the GMM is:

$$p(x|y) = \sum_i \pi_i p_i(x|y) \tag{3.3}$$

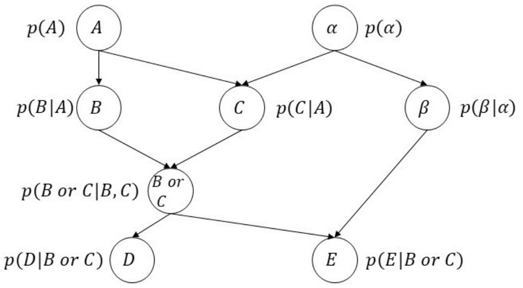


Figure 3.2: Bayesian Network

Where $\sum_i \pi_i = 1$, $p_i(x|y)$ is the Gaussian distribution as in (1).

(5)The samples in HMM are generated by the hidden state of the HMM, and the conditional distribution of the state is GMM as (4).

For the generated samples $x_i = x_{i1}, x_{i2}, \cdots, x_{im}, i = 1, \cdots, n$, Naive Bayes classifier can be used to compute the posterior probability of the label $y_i \in c_1, c_2, \cdots, c_C$, and label x_i as the label with highest posterior probability.

$$y_i = \text{argmax}_{y_i=c_i}^{c_C} p(y_i|x_i) = \text{argmax} p(x_i|y_i)p(y_i) \tag{3.4}$$

Graph-based Methods

The essence of graph-based methods is label propagation. Based on the manifold assumption, the graph is constructed according to the geometric structure between the samples, and the samples are represented by the vertices of the graph. The labels are propagated from the labeled samples to the unlabeled samples according the adjacency relation which describes the correlations between the vertices of the graph.

The training procedure of the graph-based methods is shown in Figure 3.3.

(1) Choose an appropriate distance function to compute the sample distance such as Euclidean Distance, Manhattan Distance, Chebyshev Distance, Minnesota Distance, Mahalanobis Distance and Normalized Euclidean Distance.

(2) Select the appropriate connection method according to the distance. Construct the graph by connecting the samples with edges. The graph includes dense graph such as complete graph shown in Figure 3.4, in which every pair of distinct nodes is connected by a unique edge, and sparse graph as shown in Figure 3.5, in which the nearest nodes are connected according to certain criteria.

(3) Use kernel function to assign a weight to the edges of the graph, which can reflect the similarity between the two edges. In other words, when the two nodes are close to each other, the weight of the edge is large, which means the two samples are very likely to share a same label. Commonly used kernel functions include Linear kernel, Polynomial kernel, Gaussian kernel, RBF(Radial Basis Function) kernel, Hyperbolic Tangent kernel, Neural Network kernel, Fisher kernel and Spline kernel [57].

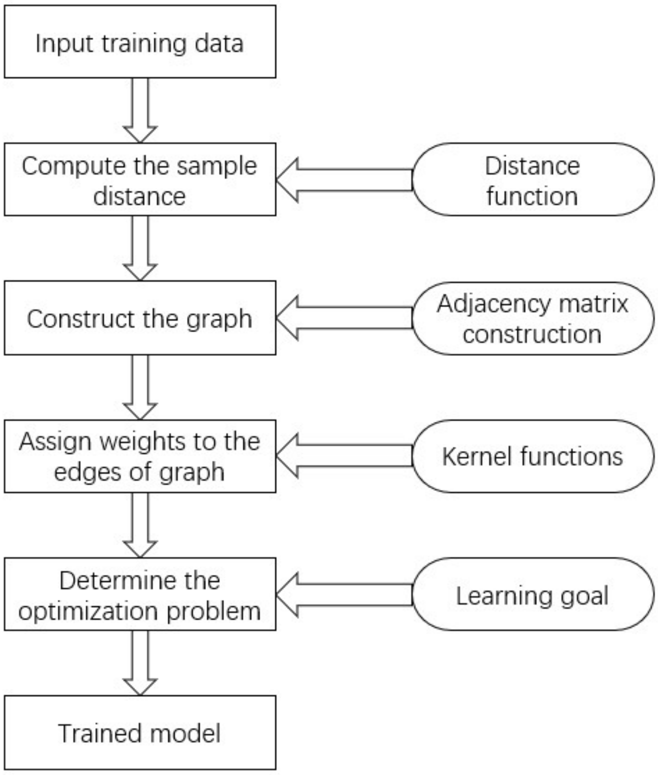

Input training data
Compute the sample distance
Distance function
Construct the graph
Adjacency matrix construction
Assign weights to the edges of graph
Kernel functions
Determine the optimization problem
Learning goal
Trained model

Figure 3.3: Graph based methods

(4) Determine and solve the optimization problem according to the learning goal. The goal of the semi-supervised learning is to find a prediction function which minimizes the objective function, which can be regarded as a regularized risk minimization problem of a composite objective function composed of a loss function and a regularization function. Therefore, the objective function of the graph-based method for the semi-supervised learning problem is generally expressed as following.

$$\min_{f(x)} V(y, f(x)) + \lambda\Omega(f) \tag{3.5}$$

The loss function $V(y, f(x))$ is used to penalise the case when the predicted label dose not match the given label, the regularization function $\Omega(f)$ is used to ensure the smoothness of the prediction function, so that the predicted label of the neighboring samples can be the same. Different loss functions and regularization functions can be selected according to specific learning tasks including Square Error function, Absolute Value function, Logarithmic function, Exponential function and Hinge function.

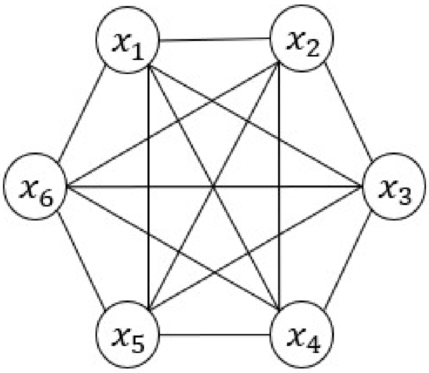


Figure 3.4: Complete graph

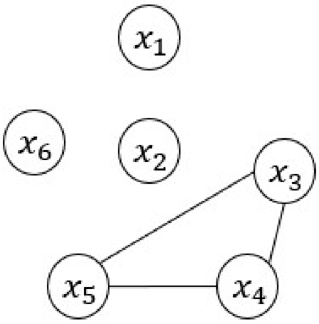


Figure 3.5: Sparse graph

3.2 Graph Convolutional Neural Networks

3.2.1 Graph Neural Networks

Graph Neural Networks (GNN) was first proposed by Gori et al. [19] and further elaborated by Scarselli et al. [52] as recurrent neural networks. These early studies mainly focus on iteratively propagating the node information to update the state of a target node until reaching a stable fixed point. However, the approaches mentioned above suffering from the high computational complexity for training the recurrent neural network. To overcome such limitations, Li et al. [33] improved the framework to use modern practices around recurrent neural network to overcome the above limitations. Kipf and Welling [29] present the first adoption of a GNN for semi-supervised classification.

In this paper, a graph represents as $G = (V, E)$, where V is the set of vertices or nodes, and E is the set of edges. $v_i \in V$ represents an node and $e_{ij} = (v_i, v_j) \in E$ represent a edge pointing from v_i to v_j. $N(v) = u \in V|(u, v) \in E$ denotes the neighborhood of node v. A is the adjacency matrix, which is a $n \times n$ matrix with $A_{ij} = 1$ if $e_{ij} \in E$ and $A_{ij} = 0$ if $e_{ij} \notin E$. For an attributed graph, the node attributes are represented by a node feature matrix $X \in R^{n \times d}$ where $x_v \in R^d$ representing the feature vector of node v. The edge attributes of a graph are represented by an edge feature matrix $X^e \in R^{m \times c}$ where $x^e_{uv} \in R^c$ representing the feature vector of edge (u, v). Table 3.1 shows the commonly used notations.

A directed graph is a graph where all the edges are directed from one node or vertices to another. An undirected graph is a graph where all the edges are bidirectional. The adjacency matrix of a graph is symmetric when the graph is an undirected graph.

Table 3.1: Commonly used notations

Notations	Descriptions
$\lvert \cdot \rvert$	The length of a set.
$\cdot$	Element-wise product.
G	A graph.
V	The set of nodes in a graph.
v	A node of V.
E	The set of edge in a graph.
e_{ij}	A edge of E.
$N(v)$	The neighborhood of node v.
A	The adjacency matrix.
D	The degree matrix of A, $D_{ii} \sum_{j=1}^{n} A_{ij}$.
L	The Laplacian matrix, $L = D - A$.
n	The number of nodes.
m	The number of edges.
d	The dimension of a node feature vector.
b	The dimension of a hidden node feature vector.
c	The dimension of an edge feature vector.
$X \in R^{n \times d}$	The feature matrix of a graph.
$x \in R^n$	The feature vector of a graph in the case of d = 1.
$x_v \in R^d$	The feature vector of the node v.
$X^e \in R^{m \times c}$	The edge feature matrix of a graph.
$x^e_{(u,v)} \in R^c$	The edge feature vector of the edge (u, v).
$H \in R^{n \times b}$	The node hidden feature matrix.
$h_v \in R^b$	The hidden feature vector of node v.
k	The index of the layer.
$\sigma(\cdot)$	The sigmoid activation function.
$W, \theta, \omega, \Theta$	Learnable parameters.

GNNs [52] is the first proposed model which build the neural network on graph. In GNNs, the aggregation function is defined as a recurrent function, the state of each node is updated based on the neighboring nodes and edge information:

$$h_x = f_w(l_x, l_{c0[x]}, l_{ne[x]}, h_{ne[x]}) \tag{3.6}$$

Where $l_x, l_{c0[x]}, l_{ne[x]}, h_{ne[x]}$ respectively denotes the label of node x, label of edge connected to x, label of neighboring nodes of x and the state of neighboring nodes of x at the previous time step. The f_w is the aggregation function which is defined as a recurrent function. The state of x is updated recurrently according to f_w until convergence. In addition, the global output function is defined by GNNs and applied to the converged state of each node to obtain the final output:

$$o_x = g_w(h_x, l_x) \tag{3.7}$$

Where g_w denotes the global output function.

3.2.2 Spectral-based Graph Convolutional Neural Networks

The spectral-based methods define the graph convolution in spectral space based on convolution theorem.

Convolution Theorem: The Fourier transform of signal convolution is equivalent to the product of Fourier transform of the signal:

$$F(f \star g) = F(f) \cdot F(g) \tag{3.8}$$

Where f, g denote the original signal, $F(f)$ denotes the Fourier transform of f. $\cdot$ and $\star$ indicate the pointwise product and the convolution, respectively.

According to the inverse Fourier transform:

$$f \star g = F^{-1}(F(f) \cdot F(g)) \tag{3.9}$$

Where $F^{-1}(f)$ denotes the inverse Fourier transform of f.

We can implement graph convolution by inverse transform the product of the spectral signal into original space according to Convolution theorem. Therefore, the convolution on graph can be implemented without the translation invariance on graph.

The undirected graph can be represented by the symmetric normalized graph Laplacian matrix: $L = I - D^{-1/2}\Lambda D^{-1/2}$, where D is the degree matrix and A is the adjacency matrix of the graph. L is a symmetric positive-semidefinite matrix, which can be decomposed with $L = U\lambda U^T$, where U is the matrix of eigenvectors and Λ is the diagonal matrix of eigenvalues. Taking U as the basis of the spectral space, the graph Fourier transform of signal x is:

$$\hat{x} = U^T x \tag{3.10}$$

Where x denotes the original graph signal in vertex space. $\hat{x}$ denotes the signal transformed into spectral space. U^T denotes the transpose of the eigenvector matrix for the Fourier transform. The inverse Fourier transform of signal x is:

$$x = U\hat{x} \tag{3.11}$$

Based on the Fourier transform and inverse Fourier transform on graph, the graph convolution operator can be defined as:

$$x_G^* y = U((U^T x) \odot (U^T y)) \tag{3.12}$$

Where $^*_G$ denotes the graph convolution operator, x, y denote the signal in vertex space, $\odot$ denotes Hadamard product, which can be transformed from element-wise product to matrix product by replace the vector $U^T y$ with a diagonal matrix g_θ. Therefore, the graph convolution can be represented as:

$$U g_\theta U^T x \tag{3.13}$$

Spectral CNN

Spectral Convolutional Neural Network (Spectral CNN) [5] defines the graph convolution operator on each layer based on convolution theorem. With the help of the loss function, the convolution kernel is learned through the gradient back-propagation, and multiple graph convolutional layers are stacked to build a neural network. The structure of the m-th layer of the Spectral CNN is as follows.

$$X_j^{m+1} = h(U \sum_{i=1}^{p} \{F_{i,j}^m U^T X_i^m\}), j = 1, \cdots, q \tag{3.14}$$

Where, p, q are the dimensions of input and output feature, $X_i^m \in R^n$ denotes the i-th input feature on the m-th layer of the graph nodes, $F_{i,j}^m$ denotes the convolutional kernel in spectral space, h denotes the activation function. In this layer a p dimensional feature is transformed into q dimensional by graph convolution.

ChebNet

The traditional Spectral CNN has two shortcomings:

(1) The graph convolution kernel is global with a large amount of parameters.

(2) The computational complexity of the graph convolution is very high because of the eigendecomposition.

Chebyshev Spectral CNN (ChebNet) [13] uses the polynomial expansion to approximate the graph convolution, that is, the parameterized convolution kernel is:

$$g_\theta = \sum_{i=0}^{K-1} \theta_k T_k(\hat{\Lambda}) \tag{3.15}$$

Where θ_k is the learning coefficient, $\hat{\Lambda} = 2\Lambda/\lambda_{max} - I_n$. The recursive expression of Chebyshev polynomial is:

$$T_k(x) = 2xT_{k-1}(x) - T_{k-2}(x) \tag{3.16}$$

Where $T_0(x) = 1, T_1(x) = x$.

Let $\hat{L} = 2L/\lambda_{max} - I_n$, the structure of the m-th layer of the ChebNet is defined as:

$$X_j^{m+1} = h\left(U \sum_{i=1}^{p} \left(\sum_{k=0}^{K-1} \theta_k T_k(\hat{\Lambda})\right) U^T X_i^m\right) = h\left(\sum_{i=1}^{p} \sum_{k=0}^{K-1} \theta_k T_k(\hat{L}) X_i^m\right),\ j = 1, \cdots, q \tag{3.17}$$

ChebNet implements the spectral CNN based on the polynomial parameterized convolutional kernel of the eigenvalue matrix. The Laplacian matrix introduced by $L = U\Lambda U^T$ to avoid the eigendecomposition of the Laplacian matrix. Therefore, the parameter complexity is reduced from $O(n \times p \times q)$ to $O(K \times p \times q)$. In addition, in the Laplacian matrix, if and only if the nodes i, j satisfy K-hop reachability, $L^K_{i,j} \neq 0$. Therefore, the ChebNet has locality when K is small.

CayleyNets

The CayleyNets [32] is proposed based on ChebNet, which builds a new spectral convolution filter based on Cayley polynomials. Cayley polynomials are real-valued functions with complex coefficients:

$$g_{c,h}(\lambda) = c_0 + 2\mathrm{Re}\left\{\sum_{j=1}^{r} c_j(h\lambda - i)^j(h\lambda + i)^{-j}\right\} \tag{3.18}$$

Cayley filter is a spectral filter defined on real-valued signal f:

$$Gf = g_{c,h}(\Delta)f = c_0 f + 2\mathrm{Re}\left\{\sum_{j=1}^{r} c_j(h\Delta - iI)^j(h\Delta + iI)^{-j} f\right\} \tag{3.19}$$

Where c and h are the training parameters.

With the analytical properties of Cayley filter that any smooth spectral filter can be represented as Cayley polynomials, which has better spatial locality. Compared with the Chebyshev based ChebNet, the CayleyNets also has better locality and linear complexity. In addition, the spectral scaling factor h can be adaptively adjusted to detect the narrow frequency bands.

Graph Convolution Network

In order to use the graph convolutional neural network in a semi-supervised setup on graph, Graph Convolution Network (GCN) [29] is proposed as the first-order approximation of the simplified ChebNet. In GCN, assume $K = 2$ and $\lambda_{max} = 2$, the equation can be simplified as:

$$X_j^{m+1} = h\left(\sum_{i=1}^{p}(\theta_0 - \theta_1(L - I_n))X_i^m\right),\ j = 1, \cdots, q \tag{3.20}$$

There is only a limited number of labeled data in the graph semi-supervised learning scenario. Therefore, to avoid overfitting of the model caused by the limited labeled training data, the parameters are reduced by constraining $\theta = \theta_0 = -\theta_1$, and the weights matrix is normalized. Therefore, the following first-order graph convolutional neural network is obtained:

$$X_j^{m+1} = h\left(\sum_{i=1}^{p} \theta \hat{D}^{-\frac{1}{2}}\hat{A}\hat{D}^{-\frac{1}{2}}X_i^m\right),\ j = 1, \cdots, q \tag{3.21}$$

Where $\hat{A} = A + I_n,\ \hat{D}_{ii} = \sum_j \hat{A}_{i,j}$.

3.2.3 Spatial-based Graph Convolutional Neural Networks

Compared with the spectral-based graph convolution methods based on convolution theorem, the spatial-based methods implement the graph convolution by defining the aggregation function. Such aggregation function aggregates each node with its neighboring nodes based on nodes spatial relations. For example, ChebNet and the first-order GCN can be regarded as the graph convolution methods which use Laplacian matrix as the aggregation function. Here we introduce two general frameworks of spatial-based convolutional neural networks.

Diffusion Convolutional Neural Network

Diffusion Convolutional Neural Network (DCNN) [1] computes a degree-normalized transition matrix which provides the probability of the node information transferring from node i to node j in one step. The weights between nodes are defined based on the transition matrix. The diffusion graph convolution is defined by DCNN as follows:

$$H^k = f(W^k \odot P^k X) \tag{3.22}$$

Where $P \in R^{n \times n}$ is the transition matrix, $P = D^{-1}A$, P^k denotes the probability of the node information can be transferred to another neighboring node in k steps, $f(\cdot)$ is the activation function, H^k is the hidden representation matrix which has the same dimension with the input feature matrix X, W is the learning weights. DCNN can represent the high-order information between the nodes, but is hard to apply on large-scale graph due to the high computational complexity.

Mixture Model Networks

Mixture Model Networks (MoNet) [39] focuses on the lack of translation invariance on the graph. It maps the local structure of each node to the vectors with same size through a defined mapping function and then learns the shared convolutional kernel from the mapping.

MoNet defines the coordinate system on graph, and represents the node relation as a low-dimensional vector in the new coordinate system. In addition, MoNet defines a series of weighting functions, which are applied on all neighboring nodes of the central node. The input of the weighting function is the node relation and the output is a scalar value. MoNet can obtain the vector representation with same size for each node according to the weighting functions:

$$D_j(x)f = \sum_{y \in N(x)} w_j(u(x,y))f(y), \ j = 1, \cdots, J \tag{3.23}$$

Where $N(x)$ denotes the set of neighboring nodes of x, $f(y)$ denotes the value of node y on signal f, $u(x,y)$ denotes the low-dimensional vector representation of node x,y in coordinate system u, w_j denotes the j-th weighting function, and J denotes the number of weighting functions. Therefore, each node can obtain a J-dimensional representation which consists of the local structure information of the node. Accordingly, MoNet defines the shared convolutional kernel:

$$(f_G^{\star} g)(x) = \sum_{j=1}^{J} g(j) D_j(x) f \tag{3.24}$$

Where $g(j)_{j=1}^{J}$ is the convolutional kernel.

Message Passing Neural Networks

Compared with MoNet, Message Passing Neural Networks (MPNN) [17] focuses on defining the aggregation function between the nodes. Each node can be expressed as the superposition of the information from the neigboring nodes and the central node itself. Therefore, MPNN proposes the framework of graph convolutional neural networks by defining the general aggregation function. MPNN consists of two steps. First, the aggregation function is applied to each central node and its neighboring nodes to obtain the local structure expression. Then the update function is applied on the expression of the central node and the local structure to obtain the updated expression of the current node:

$$m_x^{t+1} = \sum_{y \in N(x)} M_t(h_x^t, h_y^t, e_{x,y}), h_x^{t+1} = U_t(h_x^t, m_x^{t+1}) \tag{3.25}$$

Where h_x^t denotes the hidden representation of the node x at the t-th step, $x_{x,y}$ denotes the edge feature of nodes x, y, M_t denotes the aggregation function at the t-th step, m_x^{t+1} denotes the local structure expression of node x according to aggregation function M_t, U_t denotes the update function at the t-th step. Each layer of the graph neural networks can be defined by the abovementioned aggregation and update functions, and thus each node can be updated based on the information from itself and the neighboring nodes to obtain the new expression based on the node local structure.

Graph Attention Network

Graph Attention Network (GAT) [67] defines the aggregation function based on the attention mechanisms, which is used to learn the relative weights between the pair of connected nodes. However, unlike the models which focus on the edge information, the adjacency matrix in GAT is only used to define the relevant nodes, and the relative weights can be learned from the node feature expression based on attention mechanisms. The structure of each graph attention layer is illustrated as Figure 3.6.

The graph attention layer takes the feature vector of node i, j as input and compute the normalized attention weight between i, j, then aggregate the feature of the neighboring nodes to the central node by weighted sum according to the attention weight. The attention weight and graph convolution of GAT is computed as follows:

$$\alpha_{i,j} = \frac{\exp(LeaklyReLU(a[Wh_i \| Wh_j]))}{\sum_{k \in N(i)} \alpha_{i,j} Wh_j)} \tag{3.26}$$

$$h_i = \sigma(\sum_{j \in N(i)} \alpha_{i,j} Wh_j) \tag{3.27}$$

Where W is used for the feature dimension transformation of each node, a is for the attention weight computation, $\|$ represents the concatenation of vectors, $\alpha_{i,j}$ represents the weight between node i, j according to a, σ denotes the nonlinear activation function.

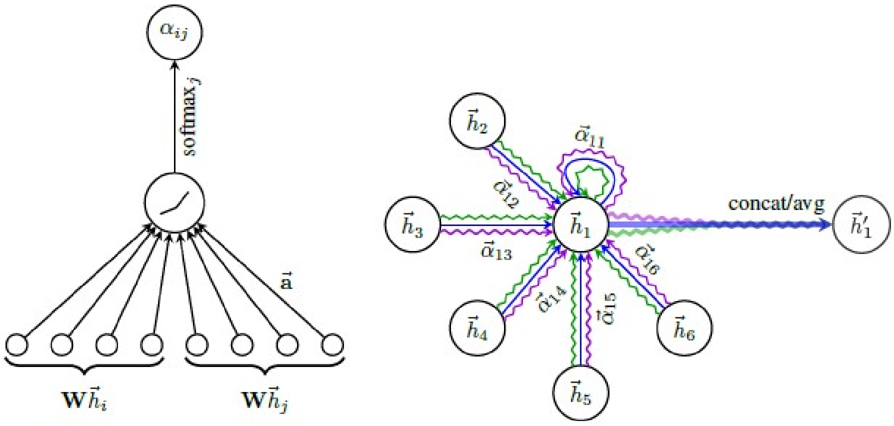


Figure 3.6: GAT

GraphSAGE

GCN models such as GAT compute the weights between nodes based on the node features. Therefore, the model needs to load the node features of the whole network, which has high computation complexity and high memory requirement. Accordingly, GraphSAGE (Sample and Aggregate) [20] was proposed. Different from the previous model that considers all neighbors, GraphSAGE randomly samples neighboring nodes from the neighbors so that the neighboring nodes of each central node are less than the given sample number. The structure of GraphSAGE is shown in Figure 3.7.

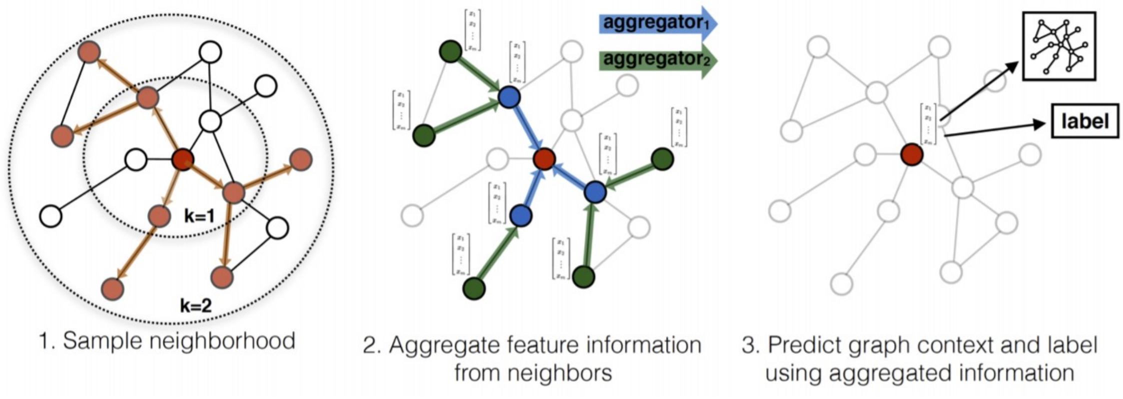

k=1
k=2
1. Sample neighborhood
aggregator$_1$
aggregator$_2$
2. Aggregate feature information from neighbors
label
3. Predict graph context and label using aggregated information

Figure 3.7: GraphSAGE

Taking the red node as the target node, GraphSAGE randomly samples from the first and second-order neighbors and uses the sampled nodes as related nodes. Then the aggregator function is applied to the related node features, and update the state of the red node according to the aggregation result.

There are a variety of aggregator functions, such as mean aggregator, LSTM aggregator and Pooling aggregator. The mean aggregator takes the element-wise mean of the node feature vectors as the aggregation result. The LSTM aggregator can aggregate the nodes with larger expressive capability due to the LSTM architecture. The pooling aggregator fed the node feature vectors into a full-connected neural network and a following max-pooling operator is applied to the output to obtain the aggregation result:

$$\mathrm{AGGREGATE}_{\mathrm{k}}^{\mathrm{pool}} = \text{ma`x}(\{\sigma(W_{pool}h_{u_i}^{k} + b), \forall u_i \in n(V)\}) \tag{3.28}$$

Where *max* denotes the element-wise max operator, σ is a nonlinear activation function.

GraphSAGE can train the model with mini-batch, which only need to load the local structure of the corresponding node instead of the whole network. This makes it possible to build a graph convolutional neural network on a large-scale data set.

Confidence-based Graph Convolutional Networks

Confidence-based Graph Convolutional Networks (ConfGCN) [66] assume each pair of two nodes has a corresponding confident label score. ConfGCN estimate the influence between of each node on the neighboring nodes based on the label confidence of the corresponding nodes. Figure 3.8 illustrates the structure of ConfGCN applied to node classification task. According to the estimation of ConfGCN, node *b* and *c* have higher influence on estimating the label of node *a* due to the high confident label scores.

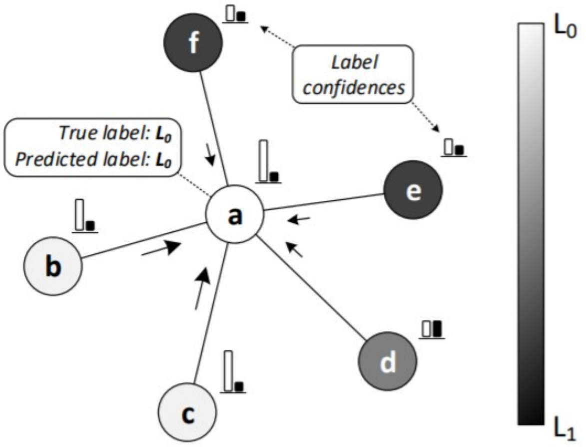


Figure 3.8: ConfGCN [66]

In ConfGCN, the distance between two nodes is defined based on the label distributions μ_u, μ_v and co-variance matrices$\sum_u, \sum_v$:

$$d_M(u,v) = (\mu_u - \mu_v)^T (\sum\limits_u^{-1} + \sum\limits_v^{-1})(\mu_u - \mu_v) \tag{3.29}$$

Accordingly the influence score of node u on node v is defined as r_{uv}:

$$r_{uv} = \frac{1}{d_M(u,v)} \tag{3.30}$$

Therefore, the graph convolution defined by ConfGCN for updating the node state at the k-th layer is as follows:

$$h_v^{k+1} = f\left(\sum_{u \in N(v)} r_{uv} \times (W^k h_u^k + b^k)\right), \forall v \in V \tag{3.31}$$

Where W denotes the learning parameters, h_v^{k+1} is the representation of node v at the $k+1$-th layer. b_k denotes bias, $N(v)$ is the set of neighboring nodes of v including v itself.

4 Semi-Supervised Remote Sensing Image Retrieval

In this chapter, the proposed end-to-end semi-supervised GCN model is introduced and the entire experimental process for RS image retrieval is presented. In section 4.1, the architecture of the semi-supervised GCN for RS image retrieval is presented, which includes the embedding networks and graph convolutional neural networks. In section 4.2, the three RS benchmarks and the experimental setup are introduced. The evaluation results are presented and analyzed in section 4.3.

4.1 Methodology

4.1.1 Network Architecture

Embedding Networks

In the proposed semi-supervised GCN, ResNet18 [22] is used as the embedding networks. ResNet18 is a form of ResNet [22], which is a residual learning framework proposed on the basis of the existing training deep neural network, which has the advantages of easy optimization and low computational burden. The residual is used to solve the degradation and gradient problems such as vanishing or exploding gradient problem so that the performance of the network can be improved as the depth increases. The structure of ResNet18 is shown in Figure 4.1.

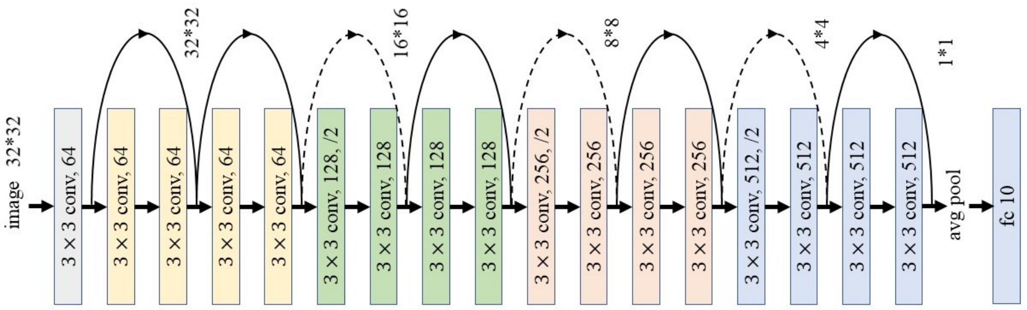


Figure 4.1: Structure of ResNet18

It consists of 17 convolutional layers and a fully-connected layer. More specifically, ResNet18 contains 4 residual blocks (shown as yellow, green, orange, and blue in Figure 4.1, respectively), and each block contains 4 convolutional layers. The first-layer residual structure of each residual block (broken lines) need to adjust the shape of the input feature matrix by reducing the size of the feature matrix to half of the original and adjusting the depth of the channel to meet the needs of the residual structure of the next layer. The images are resized to 224 × 224 before inputting to ResNet18. As shown in Figure 4.2, each residual block of ResNet18 contains 2 convolutional layers. Each convolutional layer is followed by a Batch Norm layer and a ReLU activation function. The residual structure contains skip connections, which transfer the input across layers through shortcut connections, and then adds it to the output after convolution. By means of the structure, the underlying networks can be fully trained, so that the accuracy is significantly improved with the increase in depth. In many cases, ResNet is used for image classification by means of the fully connected layer after the residual blocks and the average pooling layer. However, the fully connected layer is not required here, because it is used as a feature extractor in the semi-supervised GCN. Specifically, the features are extracted from average pooling layer.

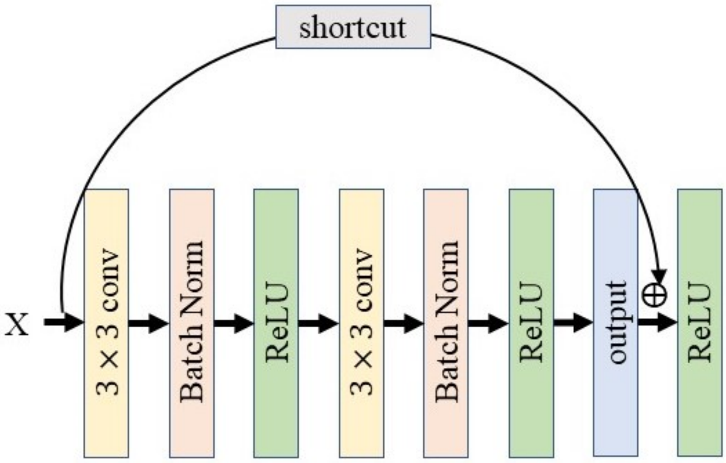


Figure 4.2: Structure of residual block of ResNet18

Graph Convolutional Neural Networks

In this chapter, we present a semi-supervised RS image retrieval approach based on GCN. The input of the end-to-end GCN framework is a collection of labeled or unlabeled RS images. Utilizing embedding networks ResNet introduced above, the input images can be embedded into feature vectors. To propagate the label information from the labeled data to unlabeled data with the help of graph convolution, a fully-connected graph $G = (V,E)$ is constructed according to the feature vectors and the labels of the input images. As shown in Figure 4.3, each node $v_i \in V$ is a concatenation of feature embedding of the image and one-hot encoding of the corresponding label.

Let a graph be denoted as $G = (V,E)$, where V is the set of nodes and E is the set of edges. $v_i, v_j \in V$ represent two nodes and $e_{ij} = (v_i, v_j) \in E$ represent an edge pointing from v_i to v_j. Given an image set $X = \{x_1, \dots, x_n\}$, we obtain the feature embedding $\phi^{(q)}(x_i)$ of the image

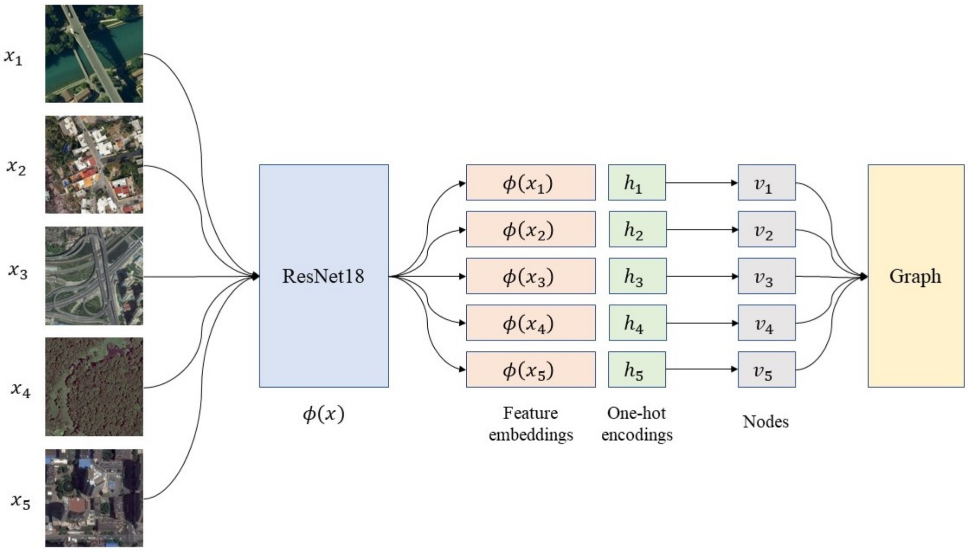


Figure 4.3: Graph construction

$x_i \in X$ by means of the embedding function $\phi^{(q)}(\cdot)$. The initial node $v_i \in V$ of the graph G is constructed as the concatenation of the image embedding feature $\phi^{(q)}(x_i)$ and the one-hot encoding h_i of the image label l_i, i.e. $v_i^{(0)} = (\phi^{(0)}(x_i), h_i)$. The node attributes of the graph are represented by the node feature matrix $Z \in R^{n \times d}$, where $z_v \in R^d$ represents the feature vector of node v. A is the adjacency matrix, which is a $n \times n$ matrix, where A_{ij} denotes the connectivity of v_i and v_j. $\hat{A} = A + I_n$ denotes the symmetric normalization of A with a self-loop, where I_n is the identity matrix. The graph convolution [29] is defined as:

$$Z_j^{l+1} = \sigma\left(\sum_{i=1}^{p} \theta \hat{D}^{-\frac{1}{2}} \hat{A} \hat{D}^{-\frac{1}{2}} Z_i^l\right), \ j = 1, \cdots, q, \tag{4.1}$$

where $\hat{A} = A + I_n$ denotes the symmetric normalization of A with a self-loop. I_n is the identity matrix, $\hat{D}_{ii} = \sum_j \hat{A}_{i,j}$ denotes a diagonal degree matrix, and σ is a non-linear activation function.

Each GCN layer is a function $f : \mathbb{R}^{n \times n} \times \mathbb{R}^{n \times d} \to \mathbb{R}^{n \times l}$ which receives input signal $Z^{(k)} \in \mathbb{R}^{n \times d}$ and produces $Z^{(k+1)} \in \mathbb{R}^{n \times l}$ denotes as

$$Z^{(k+1)} = f(\hat{A}^{(k)}, Z_i^{(k)}) = \rho(\hat{A}^{(k)} Z^{(k)} \Theta_i), \tag{4.2}$$

where $\hat{A}$ indicates the adjacency matrix, $\Theta_i \in \mathbb{R}^{d \times l}$ contains the parameters to be learned in the convolutional layer, and ρ denotes the leaky-ReLU activation function. Before each graph convolutional layer, $\hat{A}$ is learned from the current hidden states of the nodes by the symmetric function ψ_θ parameterized by a 3-layer neural network F for similarity computing based on the absolute difference of two nodes as

$$\hat{A}_{i,j}^{(k)} = \psi_\theta(v_i^{(k)}, v_j^{(k)}) = F_\theta(\|v_i^{(k)} - v_j^{(k)}\|) \tag{4.3}$$

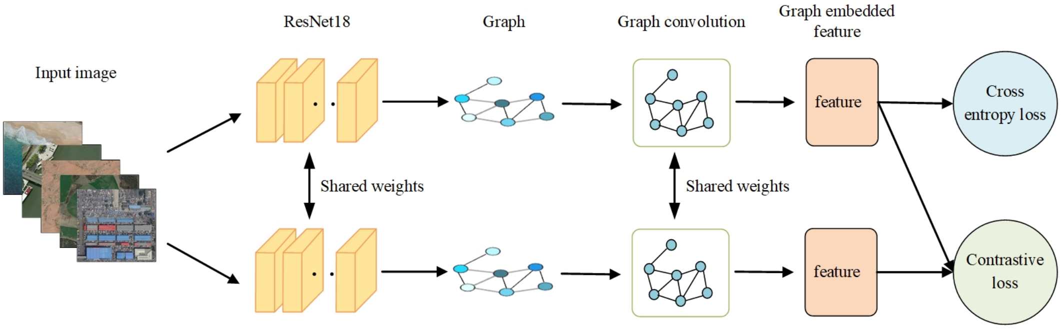


Figure 4.4: Illustration of the training procedure of the proposed GCN

As shown in Figure 4.4, with regard to cross-entropy loss and Contrastive loss, two different architectures of end-to-end GCN are proposed. The first one optimized by cross-entropy loss takes input images from the archive and extracts features through a ResNet18 to produce the feature embeddings. The graph is constructed using the feature embeddings and one-hot encodings of the corresponding labels. Afterward the graph is entered into a graph convolutional neural network to propagate the labeled information to unlabeled data and extract node features by means of graph convolution. The model is optimized by cross-entropy loss based on the labels and graph embedded features.

The second model optimized by Contrastive loss contains two parallel ResNet18 and graph convolution neural networks with shared weights.

4.1.2 Loss Function

Contrastive Loss

Contrastive loss can effectively deal with the relationship of paired data in the siamese network. This loss function was originally used in dimensionality reduction. Originally similar samples are still similar in the feature space after feature extraction. Accordingly, the original dissimilar samples are still not similar after dimensionality reduction.

Contrastive loss is defined as

$$L = \sum_{i,j} l_{ij} \|f_i - f_j\|_2^2 + (1 - l_{ij}) h(m - \|f_i - f_j\|_2)^2 \tag{4.4}$$

where $h(\cdot)$ denotes the hinge loss function, i.e. $h(x) = \max(0, x)$, m is the margin, l_{ij} denotes the label indicator function which is formulated as

$$l_{ij} = \begin{cases} 1 & \text{if } y_i = y_j \\ 0 & \text{otherwise.} \end{cases} \tag{4.5}$$

Observing the expression of the above-mentioned contrastive loss, it can be found that this loss function can effectively represent the matching degree of the paired data, and can also be used to train the model for extracting features. When $l_{ij} = 1$ (that is, the samples are similar), the loss function only leaves $\sum = l_{ij} \|f_i - f_j\|_2^2$, which is the distance of the original similar samples. A large Euclidean distance in the feature space means that the current model needs to be optimized and increase the penalization. When $l_{ij} = 0$ (that is, the samples are not similar), the loss function is $\sum (1 - l_{ij}) \max(m - \|f_i - f_j\|_2, 0)^2$. When the samples are not similar, a small Euclidean distance of the feature space corresponds to a large loss and an increasing penalization.

Cross-entropy Loss

Cross-entropy loss compares the difference between the predicted class probability and the actual class label and penalizes the probability according to the distance between the probability and the desired value.

During the training, cross-entropy loss is used to adjust the weights of the model, and the model is therefore optimized by minimizing the cross-entropy loss.

Cross-entropy loss is defined as

$$L = -\frac{1}{N} \sum_{i=1}^{N} \sum_{c=1}^{C} y_i^c log(p_i^c) \tag{4.6}$$

where p_i^c denotes the softmax probability that the sample x_i is classified into class c

$$p_i^c = \frac{exp(w_c^T v_i)}{\sum_j exp(w_j^T v_i)} \tag{4.7}$$

where w_c, w_j denote the learning parameters. v_i is the feature of x_i.

4.2 Description and Design of Experiments

4.2.1 EuroSAT

EuroSAT is a dataset for land use and land cover classification, which consists of 10 categories with in total 27,000 annotated Sentinel-2 satellite images. The 10 land-use classes are Annual Crop, Forest, Herbaceous Vegetation, Highway, Industrial, Pasture, Permanent Crop, Residential, River and Sea Lake. There are 2000-3000 images in each land-use class.

The EuroSAT is consist of two datasets, one contains only the R, G, B frequency bands and the other contains all 13 spectral bands. In this experiment, we use the first dataset which contains only RGB bands. Example of images in EuroSAT are shown in Figure 4.5.

4.2.2 NWPU-RESISC45

NWPU-RESISC45 is a large-scale benchmark for remote sensing scene classification. It contains 31,500 images, which cover 45 scene types with 700 images in each type. Each image has a size of 256×256 pixels with a spatial resolution from about 30m to 0.2m per pixel. The scene types covers airplane, airport, baseball diamond, basketball court, beach, bridge, chaparral, church, etc.

The dataset contains complex background and illustration conditions and holds a huge difference in resolution, viewpoint, translation, object pose, etc. Due to the between-class similarity and within-class diversity, NWPU-RESISC45 becomes one of the most difficult benchmarks for RS image classification and retrieval. Example of images in NWPU-RESISC45 are shown in Figure 4.6.

4.2.3 AID

AID is a large-scale aerial dataset constructed based on the image samples collected from Google Earth imagery. AID is designed for aerial scene classification and retrieval, which consists of 10,000 RGB images distributed in 30 scene classes including: airport, bare land, baseball field, beach, bridge, church, dense residential, desert, farmland, forest, industrial, meadow, mountain, parking, playground, pond, port, railway station, river, school, sparse residential, etc.

The number of sample images in each aerial scene class ranges from 220 to 420. Each image has a size of 600×600 pixels, and the image resolution varies from about 8m to 0.5m per pixel. The images are acquired from different countries and regions around the world in different seasons and illustration conditions, which increases the intra-class diversity of the dataset. Example of images in AID are shown in Figure 4.7.

4.2.4 Experimental Setup

The feature embedding of the input image can be obtained by the trained CNN and Graph CNNs models. The Euclidean distance between the obtained feature embeddings is used to obtain the closest neighbor of the out-of-sample images from a set of samples with known classes. Image retrieval searches for the most similar image in the archive by measuring the distance of the feature embedding with the query image in the metric space. In this paper, the performance of the image retrieval is evaluated by mean average precision (MAP), which is formulated as

$$\mathrm{mAP} = \frac{1}{N}\sum_{i=1}^{N} \mathrm{AP}_i \tag{4.8}$$

$$\mathrm{AP@}k = \frac{1}{\mathrm{GTP}}\sum_{k}^{n} \mathrm{P@} \times \mathrm{rel@}k \tag{4.9}$$

where N denotes the number of all the queries, GTP denotes the number of the ground truth positives, P@k denotes the precision@k and *rel*@k denotes the indicator function that equals to 1 if the retrieved data at rank k is relevant to the query and equals to 0 otherwise.

For the task in this chapter, 70% of the images in each class are randomly sampled to build the training set, 10% for validation, and 20% of the images are used for the test set. For the RS image retrieval, the training set is used as the archive while the validation set and test set are used as the query. In the task, ResNet18 is selected as the backbone embedding network architecture of the proposed GCN model. However, other CNN models such as ResNet50 can also be implemented in this task and may have better performance for feature extracting. To keep it simple, we only choose ResNet18 for this task.

The input images from the above-mentioned three benchmarks are resized to 256 × 256 pixels. A series of data augmentation methods are implemented before training including RandomGrayscale, ColorJitter, and RandomHorizontalFlip. The margin of the contrastive loss is set to 0.5. Stochastic gradient descent (SGD) optimizer is used to update the gradients. The initial learning rate is set to 0.001 and decayed by 0.5 for every 30 epochs. The batch size is set to 16, and the model is trained for 130 epochs in total.

In this chapter, we compare several GCN model with several CNN methods, including: 1) CNN-based triplet loss, 2) CNN-based contrastive loss, 3) CNN-based cross-entropy loss, 4) GCN-based contrastive loss, 5) GCN-based cross-entropy loss. All the experiments are conducted on the NVIDIA Tesla P100 graphics processing unit (GPU).

4.3 Experimental Results

Table 4.3-4.2 shows the mAP obtained by CNN-Triplet, CNN-Contrastive, CNN-CE, GCN-Contrastive, and GCN-CE for the top-40 results on EuroSAT, NWPU-RESISC45, and AID benchmarks, respectively. The results show that the GCN-based methods generally outperform CNN-based methods.

Table 4.1: MAP@40 of semi-supervised image retrieval based on different methods on NWPU-RESISC45

Methods	5% labeled	20% labeled
CNN-Triplet	0.6792	0.8111
CNN-Contrastive	0.3747	0.5712
CNN-CE	0.8086	0.8884
GCN-Contrastive	0.9184	0.9703
GCN-CE	0.9307	0.9822

In Table 4.1, CNN-Triplet yields 0.6792 mAP on NWPU-RESISC45 with 5% labeled data, CNN-Contrasive provides only 0.3747 mAP, namely the lowest mAP in contrast to other methods, while the GCN-Contrastive provides a 35.17% and 19.62% higher mAP for 5% and 20% labeled scenarios, respectively, compared to CNN-Triplet. Moreover, GCN-CE provides the highest mAP in semi-supervised scenarios on NWPU-RESISC45, namely 0.9307 and 0.9822 with 5% and 20% labeled data respectively, which is increased by 1.33% and 1.22% compared with GCN-Contrastive respectively.

In Table 4.2, GCN-Contrastive and GCN-CE also outperform the CNN-based methods including CNN-Triplet, CNN-Contrastive, and CNN-CE on AID. Among the CNN-based methods, CNN-CE provides the highest mAP for both 5% labeled and 20% labeled scenarios, i.e. 0.7326 and 0.9461 respectively, while CNN-Contrastive yields the lowest mAP with correspondingly only 0.5033 and 0.5889. By comparison, for the 5% labeled scenario, for example, GCN-Contrastive and GCN-CE increase the mAP to 0.9013 and 0.9176 respectively.

In Table 4.3, different from the above results, CNN-CE yields the highest mAP on EuroSAT for both 5% and 20% labeled scenarios, i.e. 0.9738 and 0.9832 respectively. Next is

Table 4.2: MAP@40 of semi-supervised image retrieval based on different methods on AID

Methods	5% labeled	20% labeled
CNN-Triplet	0.6840	0.8779
CNN-Contrastive	0.5033	0.5889
CNN-CE	0.7326	0.9461
GCN-Contrastive	0.9013	0.9766
GCN-CE	0.9176	0.9794

Table 4.3: MAP@40 of semi-supervised image retrieval based on different methods on EuroSAT

Methods	5% labeled	20% labeled
CNN-Triplet	0.9403	0.9699
CNN-Contrastive	0.7295	0.9524
CNN-CE	0.9738	0.9832
GCN-Contrastive	0.9576	0.9766
GCN-CE	0.9539	0.9752

GCN-Contrastive which provides a slightly higher mAP than GCN-CE. For example, GCN-Contrastive increases the mAP by 0.38% compared with GCN-CE with 5% labeled data.

We compare the computational complexity of CNN and GCN-based methods for the AID dataset. Tabel 4.4 shows the number of required model parameters (NP) and floating-point operations (FLOPS) associated to CNN-CE and GCN-CE for the AID dataset. From Tabel 4.4 one can see that the GCN model requires slightly more parameters than the CNN model. However, the FLOPs of the proposed GCN model greatly increases 47.87% compared with the CNN model, i.e. from 2.3752×10^9 to 3.5124×10^9. The introduction of GCN significantly increases the computational complexity of the model.

Table 4.4: Number of required model parameters (NP) and floating-point operations (FLOPS) associated to different methods

Methods	NP($\times 10^6$)	FLOPS($\times 10^9$)
CNN-CE	11.2093	2.3752
GCN-CE	11.7279	3.5124

Figure 4.8 shows examples of the retrieved images by CNN-Triplet, CNN-Contrastive, CNN-CE, GCN-Contrastive and GCN-CE for the AID dataset. For a query image sampled from the airport class of the test set, we present the 1st, 5th, 9th, 13th and 17th retrieved image from the archive. In general, GCN-Contrastive and GCN-CE outperform the CNN-Triplet, CNN-Contrastive and CNN-CE. As shown in Figure 4.8, GCN-based methods can retrieve more similar images, which belong to the same class as the query image. For example, in Figure 4.8(b), the 13th and 17th retrieved images are from the commercial and railway station classes respectively. In Figure 4.8(c) the 9th, 13th and 17th retrieved images are from the pond, school and industrial

classes respectively. However, the performance of CNN-CE is close to the GCN-based methods. In Figure 4.8(d), the 1^{st}, 5^{th}, 9^{th} and 13^{th} images are correctly retrieved from the same class with the query, while the 17^{th} image is from the school class.

4.4 Conclusion

In this chapter, A semi-supervised GCN framework is constructed for remote sensing image retrieval. The framework consists of: 1) an embedding convolutional neural network which can extract the feature of the RS images and generate the feature embedding for graph construction, 2) a graph convolution neural network that takes the constructed graph as input and propagates the label information from the labeled nodes to unlabeled nodes by means of graph convolution. GCN-Contrastive and GCN-CE are proposed to explore the semi-supervised learning capability of GCN based on Contrastive loss and cross-entropy loss, respectively. To evaluate the performance on semi-supervised RS image retrieval of GCN-based methods, several CNN-based methods are implemented including CNN-Triplet, CNN-Contrastive, and CNN-CE. The GCN and CNN-based methods are evaluated on three challenging RS benchmarks, i.e. EuroSAT, NWPU-RESISC45, and AID. Experimental results show the effectiveness of the proposed GCN framework. In most semi-supervised scenarios, the GCN-based methods obviously outperform CNN-based methods. However, the GCN model also significantly increases the computational complexity with a limited increase of model parameters.

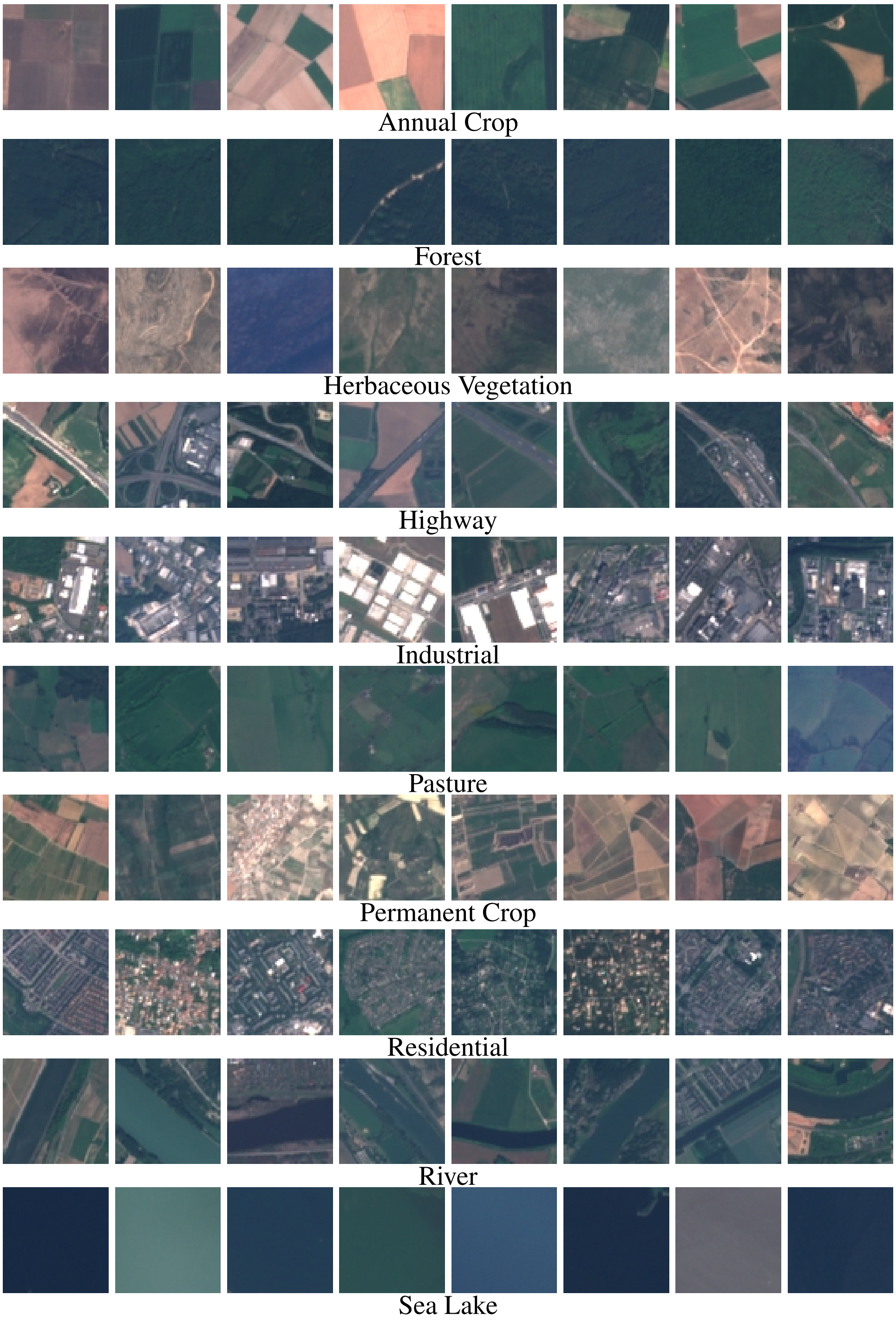


Figure 4.5: Example of images in EuroSAT and their labels.

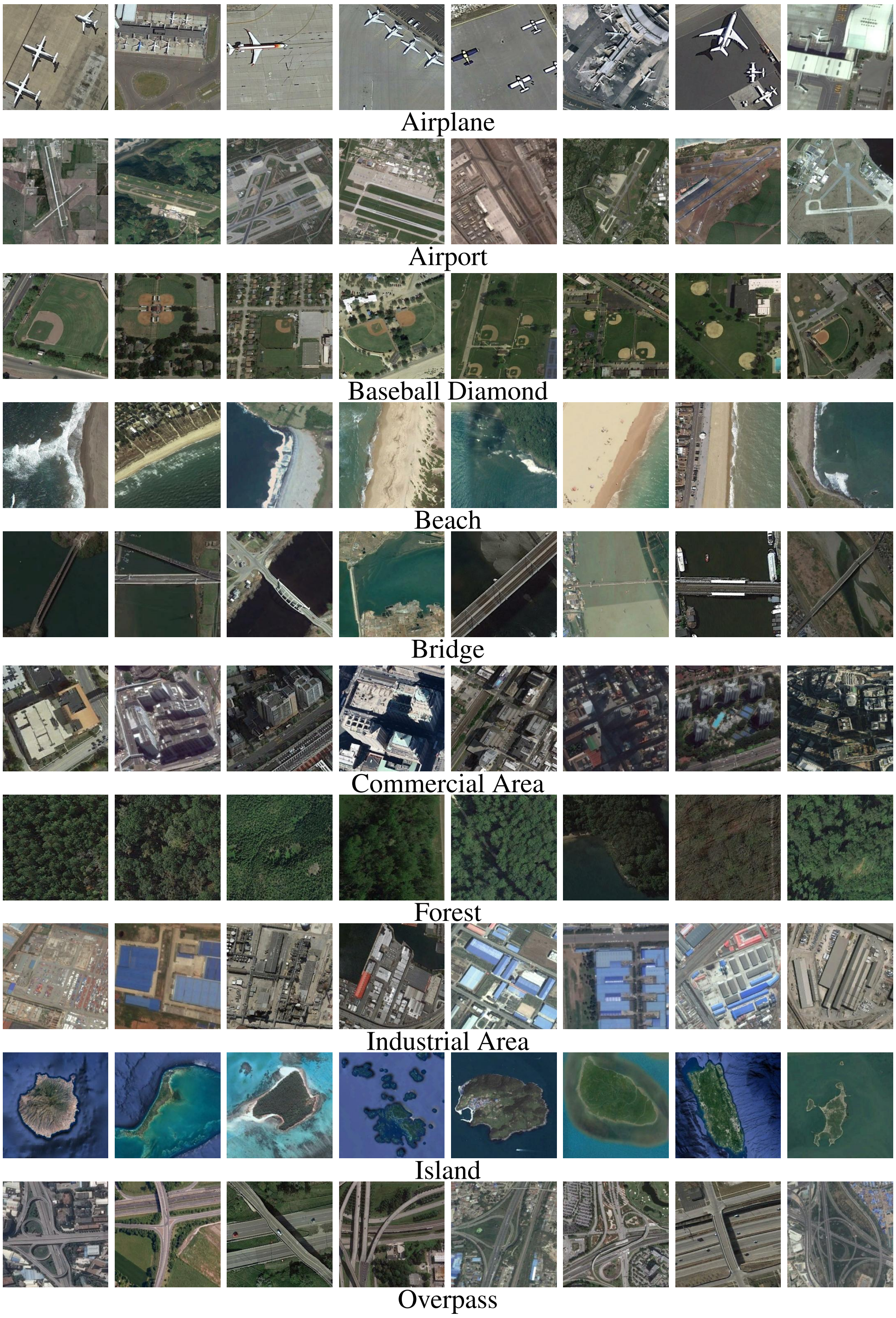


Figure 4.6: Example of images in NWPU-RESISC45 and their labels.

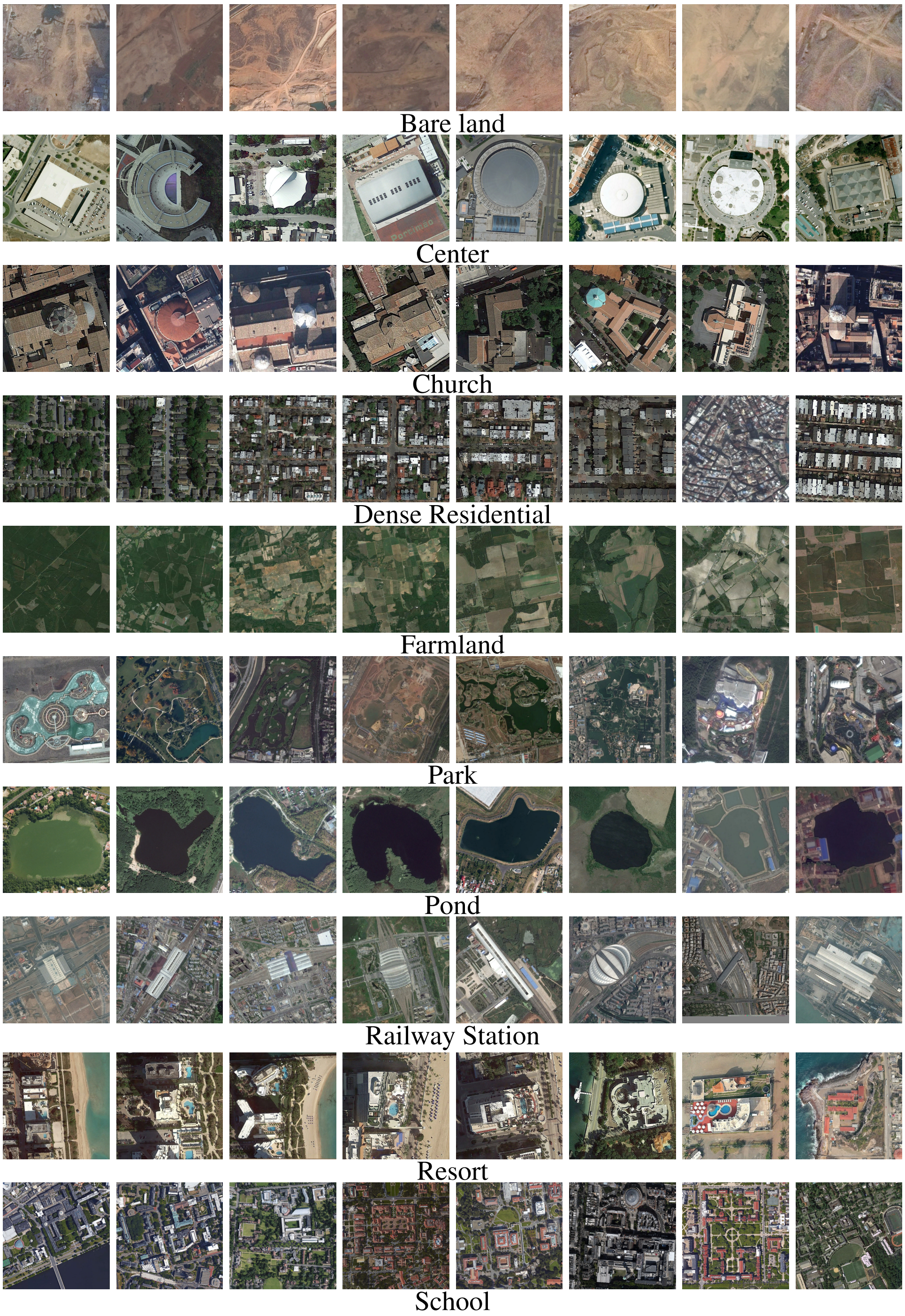


Figure 4.7: Example of images in AID and their labels.

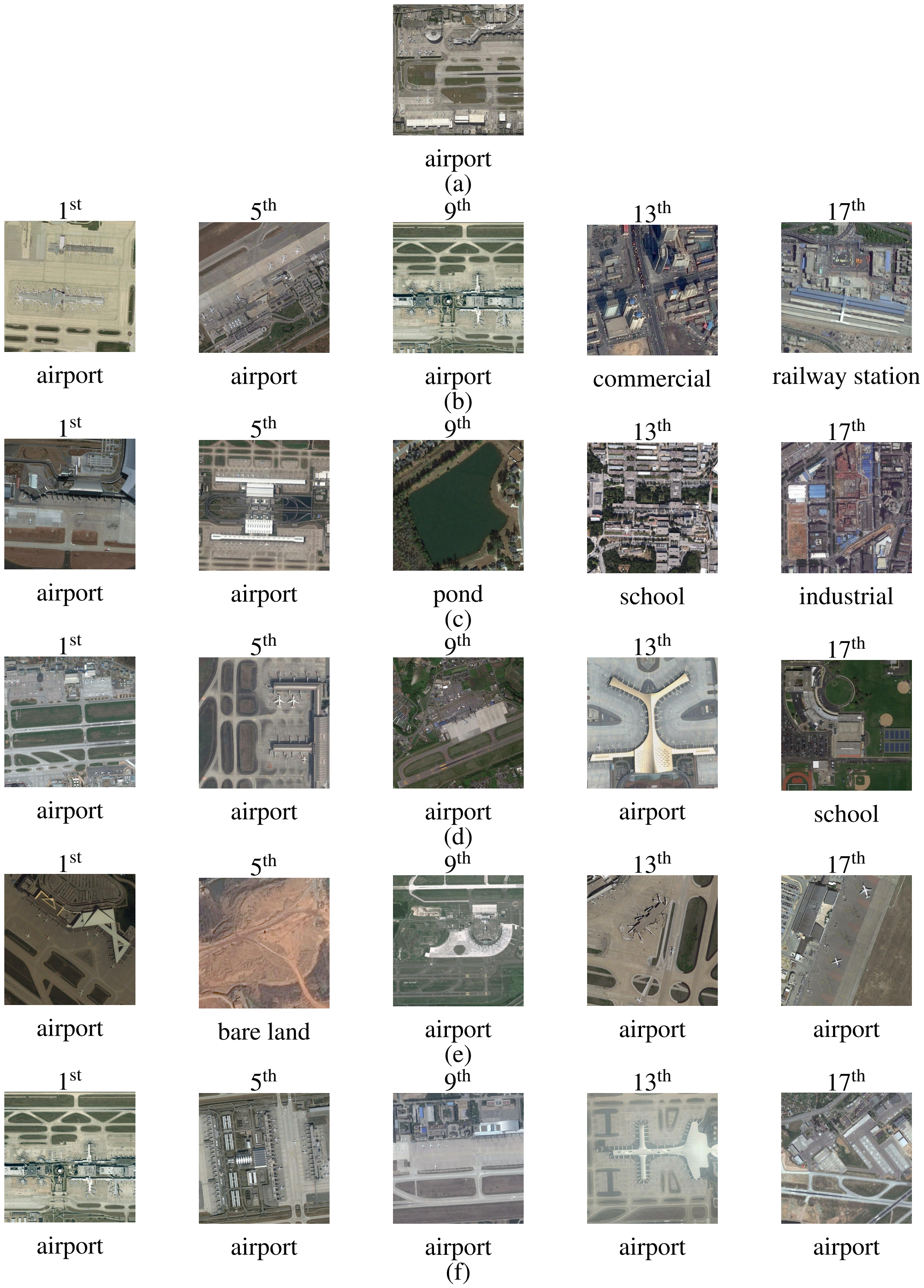


Figure 4.8: (a) Query image, (b) image retrieved by CNN-Tiplet, (c) image retrieved by CNN-Contrastive, (d) image retrieved by CNN-CE, (e) image retrieved by GCN-Contrastive, (f) image retrieved by GCN-CE.

5 Graph CNNs-based Triplet Sampling for Semi-supervised Image Retrieval

In this chapter, a triplet GCN (TGCN) for semi-supervised learning is introduced and a graph-based triplet sampling (GTS) strategy is proposed. In section 5.1, the architecture of the TGCN for RS image retrieval is presented, and the methodology of GTS is explained. The used benchmarks and experimental setup are introduced in section 5.2. In section 5.3, we provide the evaluation results and the analysis.

5.1 Methodology

5.1.1 Triplet Graph Convolutional Neural Networks

As shown in Figure 5.1, the TGCN consists of three parallel CNNs (i.e., ResNet-based architecture [22]) for extracting initial features followed by three parallel GCNs for learning the graph embedding of the images. The parallel models share their network parameters. The proposed TGCN learns the feature extraction and graph convolution simultaneously to find a graph structure that is fitted for image retrieval in a semi-supervised scenario driven by the triplet loss.

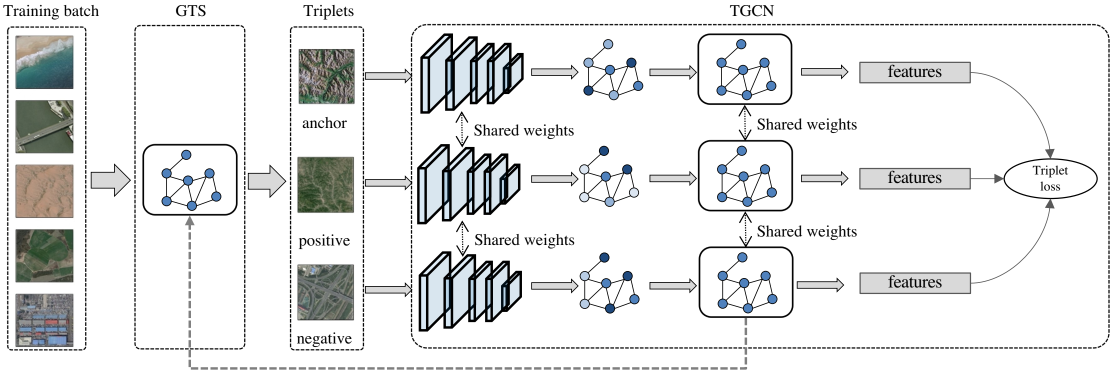


Figure 5.1: Illustration of the training procedure of the proposed TGCN-GTS.

Let a graph be denoted as $G = (V, E)$, where V is the set of nodes and E is the set of edges. $v_i, v_j \in V$ represent two nodes and $e_{ij} = (v_i, v_j) \in E$ represent an edge pointing from v_i to v_j. Given an image archive $X = \{x_1, \ldots, x_n\}$, we obtain the feature embedding $\phi^{(q)}(x_i)$ of the image $x_i \in X$ by means of the embedding function $\phi^{(q)}(\cdot)$. The initial node $v_i \in V$ of the graph G is constructed as the concatenation of the image embedding feature $\phi^{(q)}(x_i)$ and the one-hot encoding h_i of the image label l_i, i.e., $v_i^{(0)} = (\phi^{(0)}(x_i), h_i)$. The node attributes of the graph are represented by the node feature matrix $Z \in R^{n \times d}$, where $z_v \in R^d$ represents the feature vector of node v. $A \in \mathbb{R}^{n \times n}$ is the adjacency matrix, where A_{ij} denotes the connectivity of v_i and v_j. $\hat{A} = A + I_n$ denotes the symmetric normalization of A with a self-loop, where I_n is the identity matrix. The edge attributes of the graph G are represented by the edge feature matrix $Z^e \in R^{m \times c}$, where $z^e_{uv} \in R^c$ represents the feature vector of the edge (u,v). The graph convolution [29] is defined as:

$$Z_j^{l+1} = \sigma\left(\sum_{i=1}^{p} \theta \hat{D}^{-\frac{1}{2}} \hat{A} \hat{D}^{-\frac{1}{2}} Z_i^l\right), \ j = 1, \cdots, q, \tag{5.1}$$

where $\hat{D}_{ii} = \sum_j \hat{A}_{i,j}$ denotes a diagonal degree matrix, and σ is a non-linear activation function. Each GCN layer is a function $f : \mathbb{R}^{n \times n} \times \mathbb{R}^{n \times d} \to \mathbb{R}^{n \times l}$ which receives input signal $\mathrm{Z}^{(k)} \in \mathbb{R}^{n \times d}$ and produces $\mathrm{Z}^{(k+1)} \in \mathbb{R}^{n \times l}$ denoted as:

$$\mathrm{Z}^{(k+1)} = f(\hat{A}^{(k)}, Z_i^{(k)}) = \rho(\hat{A}^{(k)} \mathrm{Z}^{(k)} \Theta_i), \tag{5.2}$$

where $\Theta_i \in \mathbb{R}^{d \times l}$ contains the learnable parameters in the convolutional layer and ρ denotes the leaky-ReLU activation function. Before each GCN layer, $\hat{A}$ is learned from the current hidden states of the nodes by the symmetric function ψ_θ parameterized by a 3-layer network F. The similarity of two nodes is obtained based on the absolute difference of the nodes and denoted as:

$$\hat{A}_{i,j}^{(k)} = \psi_\theta(v_i^{(k)}, v_j^{(k)}) = F_\theta(\|v_i^{(k)} - v_j^{(k)}\|) \tag{5.3}$$

The proposed TGCN consists of three parallel ResNets and GCNs that share the network parameters. As shown in Figure 5.1, the input of the model is a triplet sampled from the archive X. The triplet consists of an anchor image x_a, a positive image x_p from the same class, and a negative image x_n from a different class. The ResNet works as the embedding network to produce the feature embedding $\phi^{(q)}(x_i)$ of the image $x_i \in X$. The graph G is the input to the 3-layer GCN, which performs the feature learning and label propagation through the graph convolution and obtains the output graph representation. Afterwards, the graph-structured data is mapped into an l-dimensional embedding space.

The model is optimized by an adapted triplet loss, which is applied to the embeddings of the images that are provided by the GCN. The triplet loss function is originally proposed by [53] to ensure that the distance in the feature space between the anchor image and the positive image is smaller than the distance between the negative image by at least the margin m. The formulation of triplet loss is given in Eq. 5.4:

$$\mathscr{L}_{Triplet} = \sum_{a,p,n} [\|f(v_a) - f(v_p)\|_2^2 - \|f(v_a) - f(v_n)\|_2^2 + m]_+, \tag{5.4}$$

where $f(\cdot)$ denotes the GCN function. This loss function ensures that the distance in the GCN-induced metric space between the anchor image and the positive image is smaller than between and the negative image by at least the margin m.

5.1.2 Graph CNNs-based Triplet Sampling

A common way to select triplets is to apply random selection based on the class label similarity among the images. Such a random triplet sampling (RTS) strategy may select several easy triplets. To avoid this issue, we propose the GTS strategy that aims at considering similarity in the embedding space, in which negative images are closer to the anchor, while positive images are farther apart. The proposed sampling strategy forms hard triplets that can make the model better understand the essential difference between different categories [53]. For each anchor image $x_a \in X$ we first randomly sample a number of positive images $X_p = \{x_p^1, \ldots, x_p^n\}$ from X. For each selected image $x_i \in X_p \cup \{x_a\}$ with label l_i we obtain the feature embedding $\phi^{(q)}(x_i)$. Therefore, the initial nodes of the graph G_p^a are represented by $V_p^{a(0)} = \{v_1^{(0)}, \ldots, v_k^{(0)}\}$ where $v_i^{(0)} = \phi^{(0)}(x_i)$. The graph G_p^a is given as an input to a neural network stacked after the 3-layer GCN and optimized by the triplet loss to map the graph into metric space and get the index of a hard positive image. The corresponding node of the hard positive image v_{hp} is denoted as:

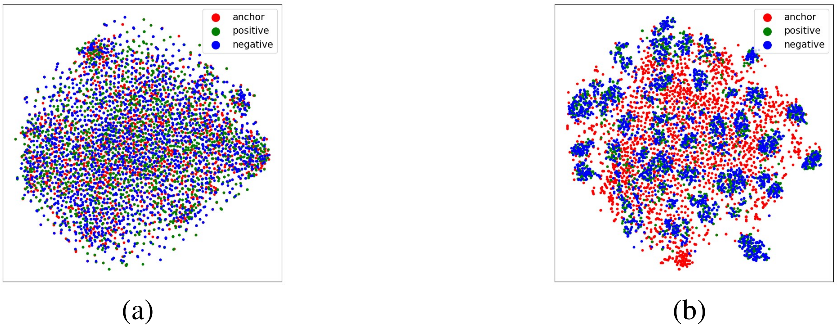
$$v_{hp} = \underset{v_i^{(k)} \in V_p^{a(k)}}{\operatorname{argmax}} \, \psi_\theta(g(v_a^{(k)}), g(v_i^{(k)})), \tag{5.5}$$

where $g(\cdot)$ denotes the GCN function, and ψ_θ is the metric function. By this way, we can construct the graph by selecting hard positive images. In the same way, we can select hard negative images. From the hard triplets the loss term is computed as:

$$\begin{aligned}\mathscr{L}_{GTS} = \sum_{a=1}^{A} [\max_{v_p \in V_p^a \setminus \{v_a\}} \|g(v_a) - g(v_p)\|_2^2 - \\ \min_{v_n \in V_n^a \setminus \{v_a\}} \|g(v_a) - g(v_n)\|_2^2 + m]_+,\end{aligned} \tag{5.6}$$

where v_a denotes the node corresponding to the a^{th} anchor image, V_p^a denotes the nodes of the graph G_p^a constructed by v_a and a number of positive images v_p. Accordingly, V_n^a denotes the nodes of the graph G_n^a constructed by v_a and a number of negative images v_n.

The image retrieval is conducted by a k-nearest neighbors search in the graph embedding space that is generated by the proposed TGCN. For a given query image the most similar images are retrieved by calculating the Euclidean distance between the image embeddings in the feature space. The k images with the lowest distance from the query image are the result of the image retrieval.



(a) (b)

Figure 5.2: t-SNE 2D scatter plots obtained by: (a) the TGCN-RTS; and (b) the TGCN-GTS.

To visualize the difference between RTS and the proposed GTS strategy, the selected triplets were projected into a two-dimensional space by the t-distributed stochastic neighbor embedding (t-SNE). From Figure 5.2 one can see that for a set of anchor images (denoted by red points), the positive and negative images (denoted by green and blue points, respectively) sampled by GTS are more closely distributed in the metric space and are covering each other visually. More specifically, for each anchor image, the GTS provides a dissimilar positive image and a similar negative image in the metric space to form a hard triplet.

5.2 Description and Design of Experiments

5.2.1 Experimental Setup

The experiments were conducted on two different RS benchmark archives. The first archive is the Aerial Image Dataset (AID) that consists of 10,000 images grouped into 30 classes. The second archive is NWPU-RESISC45, that is a large-scale RS dataset containing 31,500 images grouped into 45 classes. For both datasets the images were split into training, validation, and testing with a ratio of 70%, 10%, and 30%, respectively. For evaluating the image retrieval performance each image in the test set is selected as a query image and image are retrieved from the training set.

A ResNet18 [22] model pretrained on ImageNet was used to extract features from the images, which are used by the GCN to learn the graph representation. The input images from the beforementioned two archives are resized to 256 × 256 pixels. A series of data augmentation methods are implemented before training including RandomGrayscale, ColorJitter, and RandomHorizontalFlip. The batch size is set to 16. The margin m of the triplet loss is set to 0.2. The stochastic gradient descent optimizer is used to update the gradients with an initial learning rate of 0.001, which is decayed by 0.5 for every 30 epochs.

In order to evaluate our TGCN we select deep metric learning based on triplet loss with batch all triplet mining (BATM) [6] as a baseline to compare with our TGCN under two triplet sampling strategies: i) RTS and ii) GTS.

5.3 Experimental Results

Table 5.1: mAP obtained by the BATM, the proposed TGCN-RTS, and the proposed TGCN-GTS for the AID archive.

Method		TDR 5%	TDR 10%	TDR 20%
BATM [6]		0.6841	0.7913	0.8779
Proposed TGCN	RTS	0.8886	0.9323	0.9493
	GTS	**0.9448**	**0.9678**	**0.9879**

Tables 5.1-5.3 show the mAP obtained by BATM, the proposed TGCN-RTS, and TGCN-GTS for the top-40 retrieved images from AID, NWPU-RESISC45 and EuroSAT, respectively. The

Table 5.2: mAP obtained by the BATM, the proposed TGCN-RTS and the proposed TGCN-GTS for the NWPU-RESISC45 archive.

Method		TDR 5%	TDR 10%	TDR 20%
BATM [6]		0.6792	0.7594	0.8111
Proposed	RTS	0.8489	0.8804	0.8920
TGCN	GTS	**0.9294**	**0.9632**	**0.9839**

Table 5.3: mAP obtained by the BATM, the proposed TGCN-RTS and the proposed TGCN-GTS for the EuroSAT archive.

Method		TDR 5%	TDR 10%	TDR 20%
BATM [6]		0.9403	0.9573	0.9699
Proposed	RTS	0.9415	0.9539	0.9922
TGCN	GTS	**0.9854**	**0.9961**	**0.9958**

results demonstrate that TGCN outperforms BATM in a semi-supervised learning setup independent of the triplet sampling strategy.. The results demonstrate that Triplet GCN outperforms BATM in semi-supervised learning setup. In Table 5.1, Triplet GCN provides a 16.97% and 12.10% higher mAP for 5% and 10% labeled scenarios respectively compared to BATM. As the amount of labeled training data increases, the performance of BATM gradually approaches Triplet GCN. From these results one can see that the proposed GTS can further improve the performance of semi-supervised image retrieval with regards to BATM and Triplet GCN. For example, in Table 5.2 GTS yields 0.9294 mAP on NWPU-RESISC45 with only 5% labeled data, which is increased by 8.05% compared with Triplet GCN and by 25.02% compared with BATM.

We evaluate the computational complexity of BATM and proposed TGCN for AID dataset. From Tabel 5.4 one can see the NP and FLOPS associated to BATM and TGCN for AID. By analysing the results in Table 5.4 one can observe that the TGCN only slightly increases the computational complexity. Specifically, FLOPS and NP of TGCN are increased by 2.42% and 4.62% compared with BATM, respectively. Therefore, the introduction of GCN will not significantly increase the computational complexity and training time consumption.

Table 5.4: Number of required model parameters (NP) and floating-point operations (FLOPS) associated to different methods

Methods	NP($\times 10^6$)	FLOPS($\times 10^9$)
BATM	33.6280	7.1258
TGCN	35.1840	7.2989

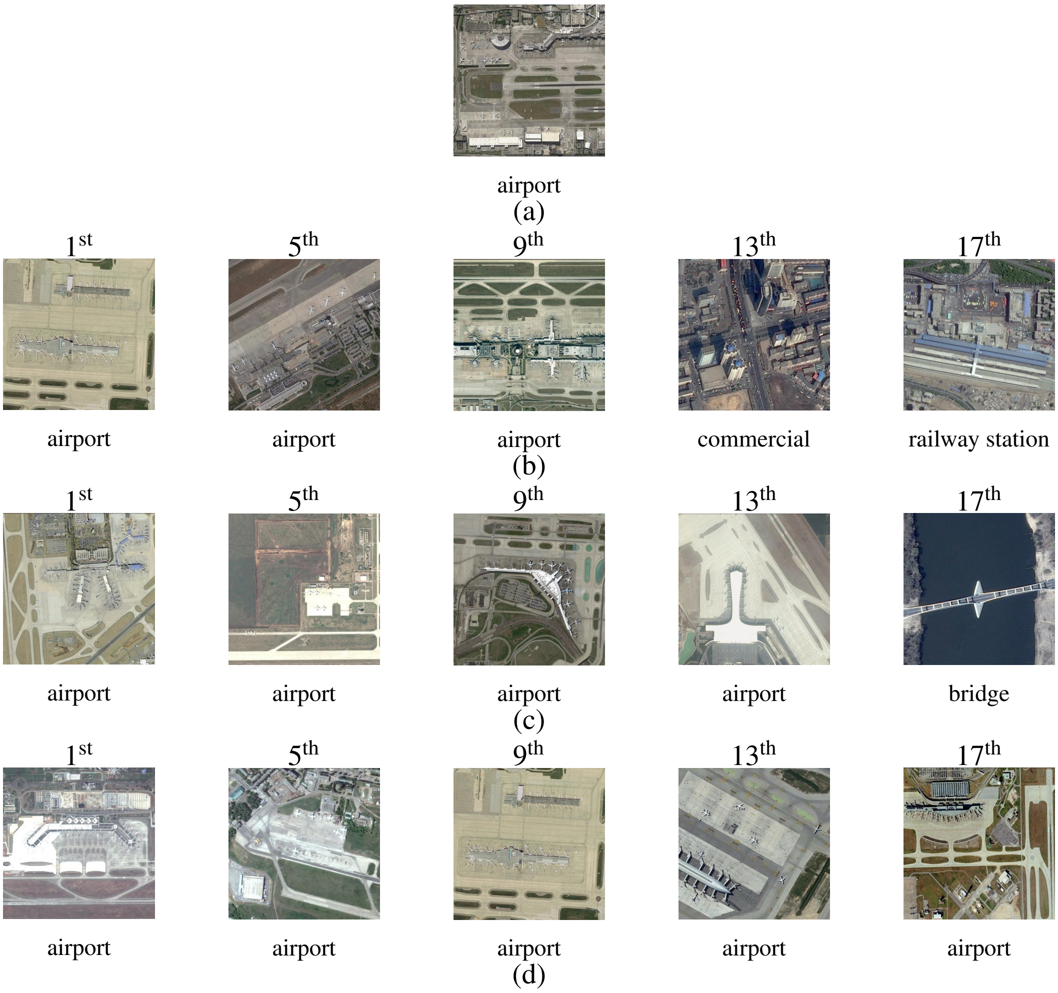


Figure 5.3: (a) Query image, (b) image retrieved by BATM, (c) image retrieved by TGCN-RTS, (d) image retrieved by TGCN-GTS.

Figure 5.3 shows examples of the retrieved images by BATM and the proposed methods for the AID dataset. For a query image sampled from the airport class of the test set, we present the 1st, 5th, 9th, 13th and 17th retrieved image from the archive. As shown in Figure 5.3, the proposed Triplet GCN and GTS methods can retrieve more similar images, which belong to the same class as the query image. For example, in Figure 5.3(b), the 13th and 17th retrieved images are from the commercial and railway station classes respectively. In Figure 5.3(c), only the 17th retrieved image belongs to a different class than the query image. In Figure 5.3(d) all retrieved images share the class of the query image.

5.4 Conclusion

In this work, we introduced a contrastive learning-based deep metric GCN for semi-supervised image retrieval from RS image archives. The GCN can propagate the label information from the labeled data to the unlabeled data and learn the implicit information from the graph-structured data. We first constructed a Triplet GCN model to characterize the remote sensing images and learn the metric space of the graph representations. We then proposed a novel GCN-based triplet sampling method that can explore the underlying similarity information among the graph structure and select hard triplets for efficient contrastive learning and model optimization. Experiments on the AID, NWPU-RESISC45, and EuroSAT RS datasets show the effectiveness of our proposed method. In addition, the proposed TGCN can achieve a computational complexity close to BATM. Moreover, the TGCN can reach the convergence in less training iterations by means of proposed GTS.

6 Semi-supervised Image Retrieval for Multi-label Remote Sensing Image

In this chapter, the proposed end-to-end semi-supervised GCN model for multi-label RS image retrieval based on Binary cross-entropy (BCE) loss is introduced and the experiments conducted on BigEarthNet are presented. In section 6.1, the architecture of the proposed GCN model is presented. The model consists of an embedding network and a GCN. In section 6.2, the multi-label RS benchmark BigEarthNet and the experimental setup is introduced. The experiment results are presented and analyzed in section 6.3.

6.1 Methodology

6.1.1 Network Architecture

The GCN model presented in this chapter is similar to the GCN model proposed for single label RS image retrieval in Chapter 4. As shown in Figure 6.1, the proposed GCN consists of an embedding network and a graph convolutional network. The ResNet18 [22] introduce in Chapter 4 is utilized as the embedding network and the graph convolutional neural network shares the same architecture as the model in Chapter 4. Different from the model for single label image retrieval, the GCN model for multi-label scenario is optimized by Binary cross-entropy (BCE) loss.

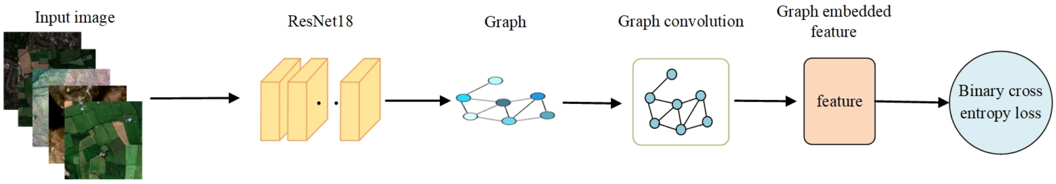


Figure 6.1: Illustration of the training procedure of the proposed GCN

6.1.2 Binary Cross-entropy Loss

Binary cross-entropy loss is combined by Sigmoid activation and Cross-entropy loss. Different from Softmax loss, each class is independent of another, that is, the loss value for each class is independently computed and will not influence the loss value for other classes when it is utilized in the multi-label scenario. The BCE loss is defined as follows

$$L_{BCE} = -\sum_i \sum_c l_{y_i}(c) log(p_i^c) - (1 - l_{y_i}(c)) log(1 - p_i^c) \tag{6.1}$$

where p_i^c denotes the probability of the existence of class c, l_{y_i} denotes the target label y_i of the i$^{\text{th}}$ image with regard to class c. The c$^{\text{th}}$ element of y_i is set to 1 if the class c is annotated and set to 0 otherwise.

6.2 Description and Design of Experiments

6.2.1 BigEarthNet

BigEarthNet is a large-scale multi-label remote sensing benchmark archive. The images are acquired by 125 Sentinel-2 tiles from 10 different countries in Europe including Austria, Belgium, Finland, Ireland, Kosovo, Lithuania, Luxembourg, Portugal, Serbia, and Switzerland. All the tiles were atmospherically corrected by the Sentinel-2 Level 2A product generation and formatting tool (sen2cor).

The dataset contains 590,326 images covering 43 imbalanced labels. The ground image patch size is 1.2 × 1.2 km, and the image size is variable due to the different resolutions in different spectral bands. BigEarthNet contains 12 of 13 Sentinel-2 spectral bands except for the 10^{th} band due to the lack of surface information. Therefore, the image size in the dataset includes 120 × 120 pixels with 10m channel resolution, 60 × 60 with 20m channel resolution and 20 × 20 with 60m channel resolution.

The number of the label correlated with each image varies from 1 to 12. Among the dataset, about 95% of the images have at most 5 labels, and only 15 images have more than 9 labels. In addition, images with approximately close numbers are acquired from different seasons. Due to the high cloud cover percentage of the Sentinel-2 images in winter, the images acquired in winter occupy the smallest proportion in the dataset. Example of images in BigEarthNet is shown in Figure 6.2.

6.2.2 Experimental Setup

The feature embedding of the input image can be obtained by the trained CNN and Graph CNNs models. The Euclidean distance between the obtained feature embeddings is used to obtain the closest neighbor of the out-of-sample images from a set of samples with known classes. Image retrieval searches for the most similar image in the archive by measuring the distance of the feature embedding with the query image in the metric space. In this chapter, the performance of the image retrieval is evaluated by mean average precision (MAP), weighted mean average precision (WMAP) and average cumulative gain (ACG).

ACG represents the average number of the shared labels between the top-r retrieved images and the query image, which is formulated as

$$ACG@r = \frac{1}{r} \sum_i^r C(q, i) \tag{6.2}$$

where $C(q,i)$ denotes the number of shared labels between the ith retrieved image with the query image.

MAP represents the mean of the average precision of the retrieved images for each query image, which is formulated as

$$\mathrm{MAP} = \frac{1}{Q}\sum_{q}^{Q}\mathrm{AP}(q) \tag{6.3}$$

$$\mathrm{AP@}k = \frac{1}{\mathrm{N_{rel}(q)@R}}\sum_{r}^{R}(\delta(q,r) \times \frac{N_{rel}(q)@R}{r}) \tag{6.4}$$

where Q denotes the number of all the queries, $N_{rel}(q)@R$ denotes the number of relevant images which share at least one label with the query image in the top R retrieved images. $\delta(q,r)$ is a indicator function that equals to 1 if the rth retrieved image shares at least one label with the query and equals to 0 otherwise.

Different from MAP, WMAP is computed based on the ACG for each top r retrieved images instead of average precision. WMAP is formulated as

$$WMAP = \frac{1}{Q}\sum_{q}^{Q}(\frac{1}{N_{rel}(q)@R}\sum_{r}^{R}(\delta(q,r) \times ACG@r)) \tag{6.5}$$

For the task in this chapter, 70% of the images in each class are randomly sampled to build the training set, 10% for validation, and 20% of the images are used for the test set. For the RS image retrieval, the training set is used as the archive while the validation set and test set are used as the query. In the task, ResNet18 is selected as the backbone embedding network architecture of the proposed GCN model. However, other CNN models such as ResNet50 can also be implemented in this task and may have better performance for feature extracting. To keep it simple, we only choose ResNet18 for this task.

The input images from the BigEarthNet are resized to 256 × 256 pixels. A series of data augmentation methods are implemented before training including RandomGrayscale, ColorJitter and RandomHorizontalFlip. Stochastic gradient descent (SGD) optimizer is used to update the gradients. The initial learning rate is set to 0.001 and decayed by 0.5 for every 30 epochs. The batch size is set to 16, and the model is trained for 130 epochs in total.

In this chapter, we compare several GCN model with several CNN methods, including: 1) CNN-based binary cross-entropy loss, referred to simply as CNN-BCE; 2) GCN-based binary cross-entropy loss, referred to simply as GCN-BCE. All the experiments are conducted on the NVIDIA Tesla P100 graphics processing unit (GPU).

6.3 Experimental Results

Table 6.1 - 6.3 show the ACG, mAP and WMAP obtained by CNN-BCE and GCN-BCE for top-100 retrieved images from BigEarthNet respectively. More specifically, each table consists of the results from three semi-supervised scenarios in which the models are trained with 5%, 10% and 20% labeled data respectively. The results demonstrate that GCN-BCE outperforms CNN-BCE in a semi-supervised learning setup.

Table 6.1: ACG@100 of semi-supervised image retrieval based on different methods on BigEarthNet

Methods	5% labeled	10% labeled	20% labeled
CNN+BCE	1.2401	1.2732	1.3259
GCN+BCE	1.2858	1.3236	1.4014

In Table 6.1, GCN-BCE yields 1.2858, 1.3236 and 1.4014 ACG for 5%, 10% and 20% labeled scenarios respectively. Compared with CNN-BCE, GCN-BCE increase ACG by 3.68%, 3.95% and 5.69% for each semi-supervised scenario respectively. From this results one can see that the GCN-based method can improve the performance of retrieving images from the archive with more shared labels with the query image.

Table 6.2: MAP@100 of semi-supervised image retrieval based on different methods on BigEarthNet

Methods	5% labeled	10% labeled	20% labeled
CNN+BCE	0.9618	0.9725	0.9787
GCN+BCE	0.9572	0.9685	0.9815

In Table 6.2, GCN-BCE generally achieves a similar performance with CNN-BCE in 5%, 10% and 20% labeled scenarios. To be more specific, CNN-BCE provides a 0.9618 and 0.9672 mAP with 5% and 10% labeled data respectively, which improve by 0.48% and 0.41% compared with the results provided by GCN-BCE. However, GCN-BCE also yields a higher mAP of 0.9815 compared with the mAP of 0.9787 provided by CNN-BCE for 20% labeled scenario. The results in Table 6.2 indicates that the GCN-based and CNN-based methods provide the similar performance on precision accuracy.

Table 6.3: WMAP@100 of semi-supervised image retrieval based on different methods on BigEarthNet

Methods	5% labeled	10% labeled	20% labeled
CNN+BCE	1.2386	1.2715	1.3226
GCN+BCE	1.2784	1.3157	1.3915

Table 6.3 evaluates the performance of GCN-BCE and CNN-BCE from a more comprehensive perspective. It shows clearly that GCN-BCE outperforms CNN-BCE in the semi-supervised learning scenarios. For example, GCN-BCE yields a 1.2784, 1.3157 and 1.3915 WMAP for 5%, 10% and 20% labeled scenarios respectively, which improve the WMAP by 3.21%, 3.47% and 5.21% compared with CNN-BCE. Different from mAP, WMAP can indicate the degree of shared labels of the retrieved images. Therefore, the results in Table 6.3 indicates the effectiveness of GCN-based method, which can learn more complex semantic relation between images.

We present the computational complexity of the CNN model and the proposed GCN model for the BigEarthNet dataset. Table 6.4 provides the NP and FLOPS of CNN-BCE and GCN-

Table 6.4: Number of required model parameters (NP) and floating-point operations (FLOPS) associated to different methods

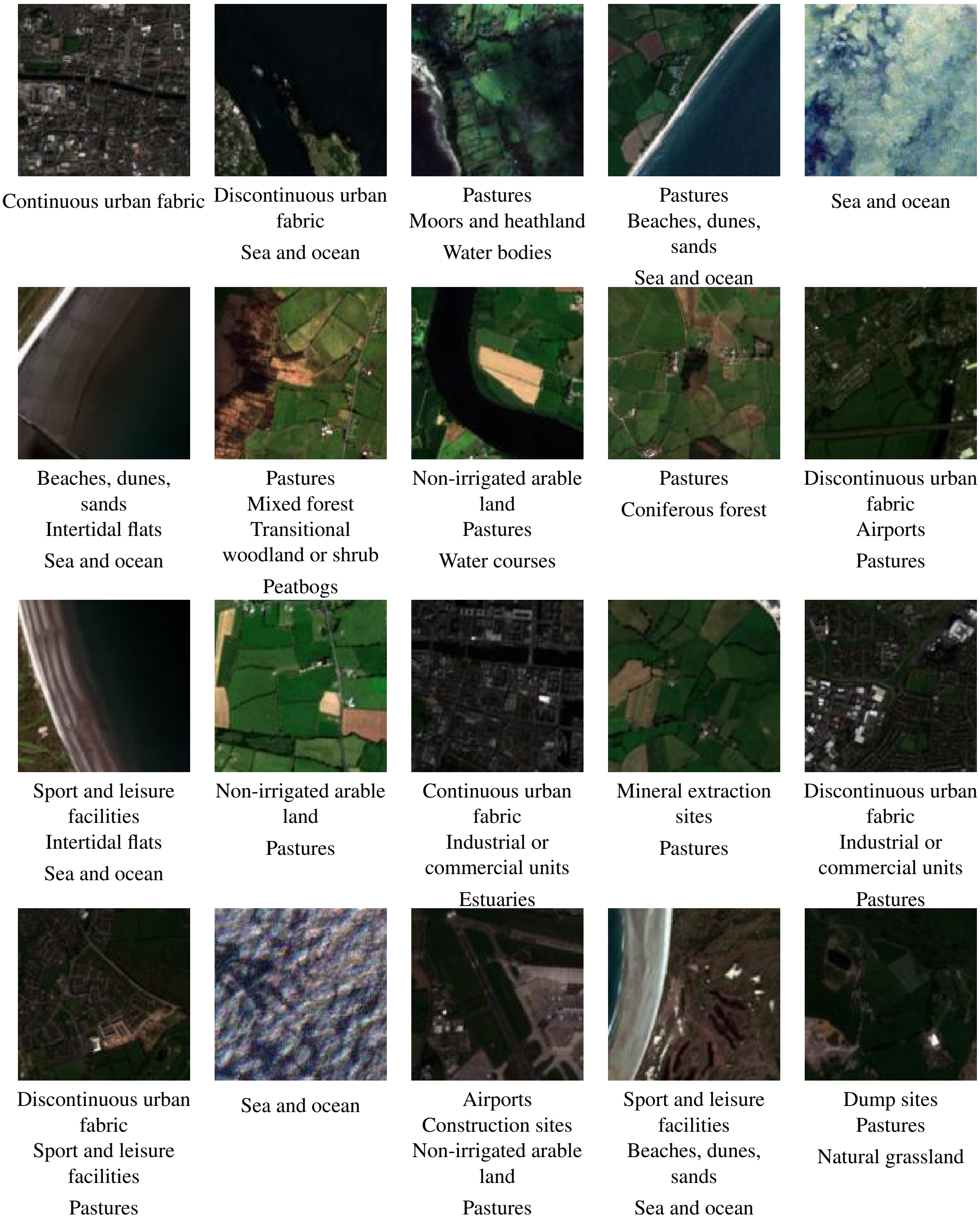
Methods	NP($\times 10^6$)	FLOPS($\times 10^9$)
CNN-BCE	11.2278	0.7012
GCN-BCE	11.7639	1.4757

BCE for BigEarthNet. From Table 6.4 one can see that the proposed GCN-BCE significantly increases FLOPS by 110.45% compared with CNN-BCE, i.e. from 0.7012 to 1.4757. However, the required model parameters of GCN-BCE is only slightly increased by 4.77% compared with CNN-BCE. By analyzing the results one can observe that the introduction of GCN significantly increases the computational complexity of the model.

Figure 6.3 shows the examples of retrieved images by CNN-BCE and GCN-BCE for the BigEarthNet dataset. For a query image sample from the test set, which has the label of Pastures, Coniferous forest and Peatbogs, we present the 1st, 2nd, 5th, 10th, 15th, 20th, 25th, 30th, 35th, and 40th retrieved images from the archive. As shown in 6.3, GCN-BCE can retrieve more similar images compared with CNN-BCE. From Figure 6.3(a) one can see that the query image has the labels including Pastures, Coniferous forest and Peatbogs. By analysing Figure 6.3(b) one can observe that most of retrieved images by CNN-BCE only share one label with query image, i.e. Pastures. In contrast, the Figure 6.3(c) shows that GCN-BCE retrieves more images which have more shared labels with query image. For example, the 1st, 2nd, 5th, 10th, 20th, 25th and 30th images share the labels Pastures and Coniferous forest with query image, while the 40th image shares the labels Pastures and Peatbogs with query image. Furthermore, the 15th retrieved image has the same labels with query image.

6.4 Conclusion

In this chapter, we introduced a multi-label RS image retrieval framework driven by a semi-supervised GCN-BCE model. The GCN-BCE model can be trained in a semi-supervised manner and propagate the label information from the labeled data to the unlabeled data utilizing the graph convolution. By means of the capability of GCN-BCE for learning the complex semantic relations between multi-label RS images, we generalize the RS image retrieval to the multi-label scenario. Experimental results on the BigEarthNet dataset show the effectiveness of the proposed GCN-BCE compared with CNN-based methods. However, the proposed GCN model significantly increases the computational complexity compared with the CNN model.



Continuous urban fabric

Discontinuous urban fabric
Sea and ocean

Pastures
Moors and heathland
Water bodies

Pastures
Beaches, dunes, sands
Sea and ocean

Sea and ocean

Beaches, dunes, sands
Intertidal flats
Sea and ocean

Pastures
Mixed forest
Transitional woodland or shrub
Peatbogs

Non-irrigated arable land
Pastures
Water courses

Pastures
Coniferous forest

Discontinuous urban fabric
Airports
Pastures

Sport and leisure facilities
Intertidal flats
Sea and ocean

Non-irrigated arable land
Pastures

Continuous urban fabric
Industrial or commercial units
Estuaries

Mineral extraction sites
Pastures

Discontinuous urban fabric
Industrial or commercial units
Pastures

Discontinuous urban fabric
Sport and leisure facilities
Pastures

Sea and ocean

Airports
Construction sites
Non-irrigated arable land
Pastures

Sport and leisure facilities
Beaches, dunes, sands
Sea and ocean

Dump sites
Pastures
Natural grassland

Figure 6.2: Examples of images in BigEarthNet and their labels.

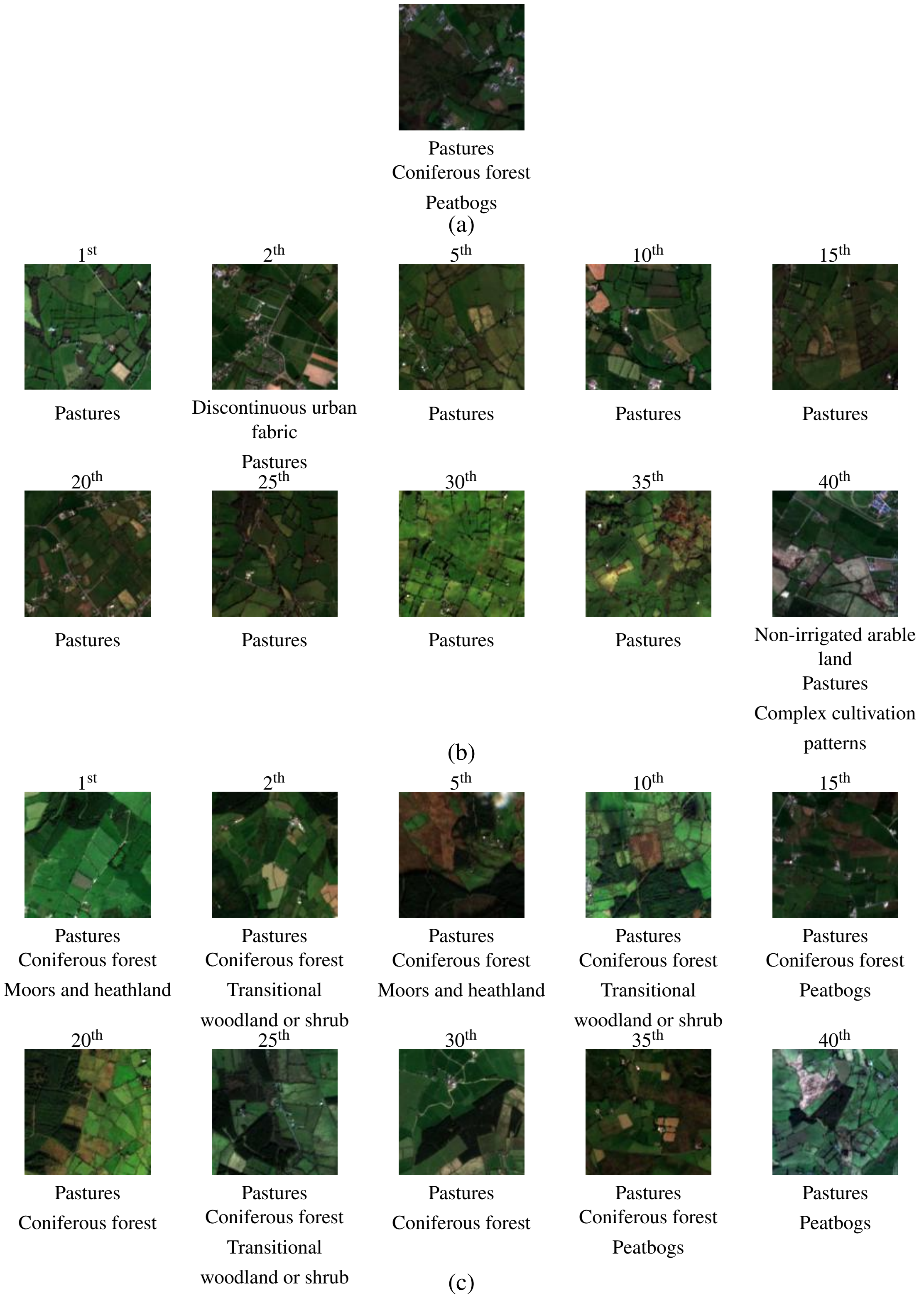


Figure 6.3: (a) Query image, (b) image retrieved by CNN-BCE, (c) image retrieved by GCN-BCE.

7 Conclusion and Discussion

In this work, aiming to address the problem of the lack of a large amount of annotated images in remote sensing image retrieval tasks, GCN-based methods are introduced. GCN can propagate the label information from labeled data to unlabeled data employing the graph convolution on graph-structured data.

Based on GCN, we first proposed a new semi-supervised GCN model for RS image retrieval by means of two different components: 1) Contrastive loss and 2) Cross entropy loss. The proposed model can effectively learn from few annotated images with a large number of unlabeled images. Experiments conducted on three RS benchmarks in different semi-supervised image retrieval scenarios validate the effectiveness of the proposed models. Compared with the CNN-based methods including: 1) deep metric CNN based on triplet loss; 2) deep metric CNN based on Contrastive loss; 3) CNN based on Cross entropy loss, the proposed GCN-based models can accurately retrieve more semantically similar images from the archive.

Afterward, aiming to utilize the explicit information about similarity and dissimilarity provided by triplet loss for metric learning, we proposed a Triplet Graph Convolutional Network (TGCN) which consists of three parallel graph models with shared weights and learns a representation from triplets of images suitable for image retrieval. Additionally, we proposed a novel GCN-based triplet sampling strategy exploring the underlying similarity information among the graph structure and selecting hard triplets for efficient metric learning and model optimization. Experiments conducted on three RS benchmarks validate the effectiveness of the proposed TGCN. Compared with TGCN with random sampling strategy and the state-of-art method BATM, the proposed GTS can further effectively improve the image retrieval performance in the semi-supervised scenario.

Moreover, we proposed a semi-supervised GCN model for multi-label RS image retrieval by means of the binary cross-entropy loss. The proposed model consists of a ResNet18 and a GCN. The ResNet18 is used to extract the multi-label RS images and produce the feature embeddings. The GCN is utilized to explore the inherent correlation between multiple labels with image features and further propagate the labels to unlabeled images. Experiments conducted on BigEarthNet benchmarks show the effectiveness of the proposed GCN model compared with the CNN model in semi-supervised scenarios. Specifically, the GCN model can effectively retrieve similar images with more shared labels.

In general, the proposed GCN frameworks have a more complex network structure and more parameters due to the additional graph convolutional neural networks. Therefore, for each training iteration, the proposed GCN needs a longer time to extract features and perform convolution operations. However, compared with the RTS strategy, the proposed GTS strategy can significantly speed up the training process due to the selection of a more effective triplet. The model can thus reach convergence with fewer iterations.

As future work, we provide several directions based on this work. One is to generalize the deep metric GCN for the multi-label scenario and explore the capability for learning the complex semantic relations between multi-label RS images. The proposed triplet deep metric GCN has already validated the effectiveness of GCN in characterizing the RS images and generating metric space that can better measure the feature similarity between RS images. Therefore, we can explore the possibility of generating the feature embeddings of multi-label RS images by the GCN-based metric learning methods by means of triplet loss. More specifically, for multi-label images, using only a single label to determine the relations between two images has limitations as in a single-label scenario, because many highly similar pictures may have multiple shared labels. GCN can be used to provide a criterion for judging whether a multi-label image is a positive or negative image for the anchor images based on the similarity.

In addition, different labels in the multi-label images may be associated. For example, beaches and ocean generally appear together. Therefore, we can construct the graph in another way, using nodes to represent each label, and edges with weights to describe the correlation between these labels. The correlation of labels can be extracted by GCN and combined with extracted features of images to better characterize the RS images.

Bibliography

[1] James Atwood and Don Towsley. “Diffusion-convolutional neural networks”. In: *Advances in neural information processing systems*. 2016, pp. 1993–2001.

[2] Artem Babenko et al. “Neural codes for image retrieval”. In: *European conference on computer vision*. Springer. 2014, pp. 584–599.

[3] Albert Boggess and Francis J Narcowich. *A first course in wavelets with Fourier analysis*. John Wiley & Sons, 2015.

[4] Petra Bosilj et al. “Retrieval of remote sensing images with pattern spectra descriptors”. In: *ISPRS International Journal of Geo-Information* 5.12 (2016), p. 228.

[5] Joan Bruna et al. “Spectral networks and locally connected networks on graphs”. In: *International Conference on Learning Representations*. 2014.

[6] Rui Cao et al. “Enhancing remote sensing image retrieval using a triplet deep metric learning network”. In: *International Journal of Remote Sensing* 41.2 (2020), pp. 740–751.

[7] Yang Cao et al. “Edgel index for large-scale sketch-based image search”. In: *CVPR 2011*. IEEE. 2011, pp. 761–768.

[8] Olivier Chapelle and Alexander Zien. “Semi-supervised classification by low density separation.” In: *AISTATS*. Vol. 2005. Citeseer. 2005, pp. 57–64.

[9] Ushasi Chaudhuri, Biplab Banerjee, and Avik Bhattacharya. “Siamese graph convolutional network for content based remote sensing image retrieval”. In: *Computer Vision and Image Understanding* 184 (2019), pp. 22–30.

[10] Gong Cheng, Junwei Han, and Xiaoqiang Lu. “Remote sensing image scene classification: Benchmark and state of the art”. In: *Proceedings of the IEEE* 105.10 (2017), pp. 1865–1883.

[11] Navneet Dalal and Bill Triggs. “Histograms of oriented gradients for human detection”. In: *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR’05)*. Vol. 1. IEEE. 2005, pp. 886–893.

[12] John G Daugman. “Complete discrete 2-D Gabor transforms by neural networks for image analysis and compression”. In: *IEEE Transactions on acoustics, speech, and signal processing* 36.7 (1988), pp. 1169–1179.

[13] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. “Convolutional neural networks on graphs with fast localized spectral filtering”. In: *arXiv preprint arXiv:1606.09375* (2016).

[14] Fabio Dell'Acqua and Paolo Gamba. "Query-by-shape in meteorological image archives using the point diffusion technique". In: *IEEE transactions on geoscience and remote sensing* 39.9 (2001), pp. 1834–1843.

[15] Begüm Demir and Lorenzo Bruzzone. "Hashing-based scalable remote sensing image search and retrieval in large archives". In: *IEEE transactions on geoscience and remote sensing* 54.2 (2015), pp. 892–904.

[16] David K Duvenaud et al. "Convolutional networks on graphs for learning molecular fingerprints". In: *Advances in neural information processing systems*. 2015, pp. 2224–2232.

[17] Justin Gilmer et al. "Neural message passing for quantum chemistry". In: *arXiv preprint arXiv:1704.01212* (2017).

[18] Yunchao Gong et al. "Multi-scale orderless pooling of deep convolutional activation features". In: *European conference on computer vision*. Springer. 2014, pp. 392–407.

[19] Marco Gori, Gabriele Monfardini, and Franco Scarselli. "A new model for learning in graph domains". In: *IEEE International Joint Conference on Neural Networks*. Vol. 2. 2005, pp. 729–734.

[20] Will Hamilton, Zhitao Ying, and Jure Leskovec. "Inductive representation learning on large graphs". In: *Advances in neural information processing systems*. 2017, pp. 1024–1034.

[21] Robert M Haralick, Karthikeyan Shanmugam, and Its' Hak Dinstein. "Textural features for image classification". In: *IEEE Transactions on systems, man, and cybernetics* 6 (1973), pp. 610–621.

[22] Kaiming He et al. "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[23] Patrick Helber et al. "Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification". In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 12.7 (2019), pp. 2217–2226.

[24] Raffaele Imbriaco, Clint Sebastian, Egor Bondarev, et al. "Aggregated deep local features for remote sensing image retrieval". In: *Remote Sensing* 11.5 (2019), p. 493.

[25] Herve Jegou, Matthijs Douze, and Cordelia Schmid. "Hamming embedding and weak geometric consistency for large scale image search". In: *European conference on computer vision*. Springer. 2008, pp. 304–317.

[26] Herve Jegou, Matthijs Douze, and Cordelia Schmid. "Product quantization for nearest neighbor search". In: *IEEE transactions on pattern analysis and machine intelligence* 33.1 (2010), pp. 117–128.

[27] Hervé Jégou et al. "Aggregating local descriptors into a compact image representation". In: *2010 IEEE computer society conference on computer vision and pattern recognition*. IEEE. 2010, pp. 3304–3311.

[28] Toshikazu Kato. "Database architecture for content-based image retrieval". In: *image storage and retrieval systems*. Vol. 1662. International Society for Optics and Photonics. 1992, pp. 112–123.

[29] Thomas N Kipf and Max Welling. “Semi-supervised classification with graph convolutional networks”. In: *arXiv preprint arXiv:1609.02907* (2016).

[30] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Communications of the ACM* 60.6 (2017), pp. 84–90.

[31] N Suresh Kumar, M Arun, and Mukesh Kumar Dangi. “Remote sensing image retrieval using object-based, semantic classifier techniques”. In: *International Journal of Information and Communication Technology* 13.1 (2018), pp. 68–82.

[32] Ron Levie et al. “Cayleynets: Graph convolutional neural networks with complex rational spectral filters”. In: *IEEE Transactions on Signal Processing* 67.1 (2018), pp. 97–109.

[33] Peng Li and Peng Ren. “Partial randomness hashing for large-scale remote sensing image retrieval”. In: *IEEE Geoscience and Remote Sensing Letters* 14.3 (2017), pp. 464–468.

[34] Kevin Lin et al. “Deep learning of binary hash codes for fast image retrieval”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*. 2015, pp. 27–35.

[35] David G Lowe. “Distinctive image features from scale-invariant keypoints”. In: *International journal of computer vision* 60.2 (2004), pp. 91–110.

[36] Aiyesha Ma and Ishwar K Sethi. “Local shape association based retrieval of infrared satellite images”. In: *Seventh IEEE International Symposium on Multimedia (ISM’05)*. IEEE. 2005, 7–pp.

[37] Stephane G Mallat. “A theory for multiresolution signal decomposition: the wavelet representation”. In: *IEEE transactions on pattern analysis and machine intelligence* 11.7 (1989), pp. 674–693.

[38] Bangalore S Manjunath and Wei-Ying Ma. “Texture features for browsing and retrieval of image data”. In: *IEEE Transactions on pattern analysis and machine intelligence* 18.8 (1996), pp. 837–842.

[39] Federico Monti et al. “Geometric deep learning on graphs and manifolds using mixture model cnns”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 5115–5124.

[40] David Nister and Henrik Stewenius. “Scalable recognition with a vocabulary tree”. In: *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’06)*. Vol. 2. Ieee. 2006, pp. 2161–2168.

[41] Aude Oliva and Antonio Torralba. “Modeling the shape of the scene: A holistic representation of the spatial envelope”. In: *International journal of computer vision* 42.3 (2001), pp. 145–175.

[42] Maxime Oquab et al. “Learning and transferring mid-level image representations using convolutional neural networks”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014, pp. 1717–1724.

[43] Savaş Özkan et al. “Performance analysis of state-of-the-art representation methods for geographical image retrieval and categorization”. In: *IEEE Geoscience and Remote Sensing Letters* 11.11 (2014), pp. 1996–2000.

[44] DU Peijun et al. “Study on content-based remote sensing image retrieval”. In: *Proceedings. 2005 IEEE International Geoscience and Remote Sensing Symposium, 2005. IGARSS’05.* Vol. 2. IEEE. 2005, 4–pp.

[45] Florent Perronnin, Jorge Sánchez, and Thomas Mensink. “Improving the fisher kernel for large-scale image classification”. In: *European conference on computer vision*. Springer. 2010, pp. 143–156.

[46] James Philbin et al. “Object retrieval with large vocabularies and fast spatial matching”. In: *2007 IEEE conference on computer vision and pattern recognition*. IEEE. 2007, pp. 1–8.

[47] Matti Pietikäinen, Timo Ojala, and Zelin Xu. “Rotation-invariant texture classification using feature distributions”. In: *Pattern recognition* 33.1 (2000), pp. 43–52.

[48] Siyuan Qiao et al. “Deep co-training for semi-supervised image recognition”. In: *Proceedings of the european conference on computer vision (eccv)*. 2018, pp. 135–152.

[49] Thomas Reato, Begüm Demir, and Lorenzo Bruzzone. “An unsupervised multicode hashing method for accurate and scalable remote sensing image retrieval”. In: *IEEE Geoscience and Remote Sensing Letters* 16.2 (2018), pp. 276–280.

[50] S. Roy et al. “Metric-Learning-Based Deep Hashing Network for Content-Based Retrieval of Remote Sensing Images”. In: *IEEE Geoscience and Remote Sensing Letters* (in press, 2020). DOI: `10.1109/LGRS.2020.2974629`.

[51] Subhankar Roy et al. “Deep metric and hash-code learning for content-based retrieval of remote sensing images”. In: *IGARSS 2018-2018 IEEE International Geoscience and Remote Sensing Symposium*. IEEE. 2018, pp. 4539–4542.

[52] Franco Scarselli et al. “The graph neural network model”. In: *IEEE Transactions on Neural Networks* 20.1 (2008), pp. 61–80.

[53] Florian Schroff, Dmitry Kalenichenko, and James Philbin. “Facenet: A unified embedding for face recognition and clustering”. In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 815–823.

[54] Karen Simonyan and Andrew Zisserman. “Very deep convolutional networks for large-scale image recognition”. In: *arXiv preprint arXiv:1409.1556* (2014).

[55] Josef Sivic and Andrew Zisserman. “Video Google: A text retrieval approach to object matching in videos”. In: *null*. IEEE. 2003, p. 1470.

[56] Malcolm Slaney and Michael Casey. “Locality-Sensitive Hashing for Finding Nearest Neighbors”. In: *IEEE SIGNAL PROCESSING MAGAZINE* 1053.5888/08 (2008).

[57] Celso André R de Sousa, Solange O Rezende, and Gustavo EAPA Batista. “Influence of graph construction on semi-supervised learning”. In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer. 2013, pp. 160–175.

[58] Gencer Sumbul, Jian Kang, and BegÃ$\frac{1}{4}$m Demir. “Deep Learning for Image Search and Retrieval in Large Remote Sensing Archives”. In: *arXiv preprint arXiv:2004.01613* (2020).

[59] Gencer Sumbul et al. “Bigearthnet: A large-scale benchmark archive for remote sensing image understanding”. In: *IGARSS 2019-2019 IEEE International Geoscience and Remote Sensing Symposium*. IEEE. 2019, pp. 5901–5904.

[60] Christian Szegedy et al. “Going deeper with convolutions”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 1–9.

[61] Xu Tang et al. “Unsupervised deep feature learning for remote sensing image retrieval”. In: *Remote Sensing* 10.8 (2018), p. 1243.

[62] Issayas Tekeste and Begüm Demir. “Advanced local binary patterns for remote sensing image retrieval”. In: *IGARSS 2018-2018 IEEE International Geoscience and Remote Sensing Symposium*. IEEE. 2018, pp. 6855–6858.

[63] Giorgos Tolias, Ronan Sicre, and Hervé Jégou. “Particular object retrieval with integral max-pooling of CNN activations”. In: *arXiv preprint arXiv:1511.05879* (2015).

[64] Isaac Triguero, Salvador García, and Francisco Herrera. “Self-labeled techniques for semi-supervised learning: taxonomy, software and empirical study”. In: *Knowledge and Information systems* 42.2 (2015), pp. 245–284.

[65] Jesper E Van Engelen and Holger H Hoos. “A survey on semi-supervised learning”. In: *Machine Learning* 109.2 (2020), pp. 373–440.

[66] Shikhar Vashishth et al. “Confidence-based graph convolutional networks for semi-supervised learning”. In: *arXiv preprint arXiv:1901.08255* (2019).

[67] Petar Veličković et al. “Graph attention networks”. In: *arXiv preprint arXiv:1710.10903* (2017).

[68] Xiaojun Wan. “Co-training for cross-lingual sentiment classification”. In: *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*. 2009, pp. 235–243.

[69] Gui-Song Xia et al. “AID: A benchmark data set for performance evaluation of aerial scene classification”. In: *IEEE Transactions on Geoscience and Remote Sensing* 55.7 (2017), pp. 3965–3981.

[70] Wei Xiong et al. “A discriminative feature learning approach for remote sensing image retrieval”. In: *Remote Sensing* 11.3 (2019), p. 281.

[71] F. Ye et al. “Remote sensing image retrieval using convolutional neural network features and weighted distance”. In: *IEEE Geoscience and Remote Sensing Letters* 15.10 (2018), pp. 1535–1539.

[72] Xiangrong Zhang et al. “Modified co-training with spectral and spatial views for semisupervised hyperspectral image classification”. In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 7.6 (2014), pp. 2044–2055.

[73] Weixun Zhou et al. “Learning low dimensional convolutional neural networks for high-resolution remote sensing image retrieval”. In: *Remote Sensing* 9.5 (2017), p. 489.

[74] Wengang Zhou et al. “Scalable feature matching by dual cascaded scalar quantization for image retrieval”. In: *IEEE transactions on pattern analysis and machine intelligence* 38.1 (2015), pp. 159–171.