

Technische Universität Berlin

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



Generative Adversarial Networks for Content-based Retrieval of Multispectral Images

Master of Science in Computer Engineering

September, 2019

Xuelel Chen

Matriculation Number: 396878

Supervisor: Prof. Dr. Begüm Demir

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

.....

Name Surname

Acknowledgements

Firstly, I'd like to express my sincere appreciation to Prof. Begüm Demir. She introduced me to the remote sensing image analysis field. And she suggested that I choose a topic that I was interested in. It was very important because I had the passion to do what I liked and would never give up easily. She emphasized novelty and the value of the research. This always encouraged me to do valuable and challenging research. And her professional knowledge of remote sensing always made me gain a lot after meeting with her.

Secondly, I'd like to thank all the other members of the RSiM group. Their professional ideas and constructive feedback always inspired me and helped me to do better in my thesis research. Yakun Li had regular meetings with me, which ensured the progress of my thesis. Later, Dr. Rubén Fernández Beltrán and Dr. Jian Kang had some meetings with me and provided me with some good methods of processing remote sensing images based on their research experience. Other researchers and thesis students also helped me a lot. I enjoyed the time working with them.

Lastly, I would like to thank my parents and friends. They always support me and helped me both materially and emotionally.

Abstract

With the advances in satellite and sensor technology, a large amount of Earth Observation (EO) data are produced every day. The Earth Observation has inevitably arrived in the Big Data era. The remote sensing image is a typical type of EO data. Remote sensing images can be used for urban area study, climate change analysis, forestry study, etc. To better manage and use large-scale remote sensing image data, it is necessary to develop efficient image retrieval methods. Deep learning-based hashing methods can generate binary codes by using deep hashing neural networks. The problem of these methods is the unavailability of sufficient labeled images, which hinders the performance of deep hashing neural networks. We propose novel methods using GANs to address this problem. GAN is a generative model to model data distributions and is made up of a generator and a discriminator. The two components play a minimax two-player game. This thesis explores two ways to make image retrieval benefit from GAN. One is to use GAN to generate images conditionally, thus enlarging the labeled dataset. Enlarged labeled data can be used to train a supervised deep hashing for image retrieval. The other is to use the discriminator of a GAN for unsupervised representation learning. By adding a hash binary constraint and a semantic constraint, we can obtain an unsupervised adversarial hashing network. This method can generate binary hash codes for multispectral remote sensing images. Experimental results showed that GAN can help improve hashing-based remote sensing image retrieval. The quality of the generated images and its effect on image retrieval are also analyzed. Conditional GAN can generate very realistic remote sensing images. Images generated from unconditional GAN are less realistic, but still keep some basic characteristics of the real images.

Zusammenfassung

Mit den Fortschritten in der Satelliten- und Sensortechnologie werden täglich große Mengen an Erdbeobachtungsdaten (Earth Observation, EO) erstellt. Die Erdbeobachtung ist unvermeidlich in der Big-Data-Ära angekommen. Fernerkundungsbild ist eine typische Art von EO-Daten. Fernerkundungsbilder können für Stadtgebietsstudien, Klimawandel-Analysen, Forststudien usw. verwendet werden. Um umfangreiche Fernerkundungsbilddaten besser verwalten und nutzen zu können, müssen effiziente Bildabrufmethoden entwickelt werden. Auf Deep Learning basierende Hashing-Methoden können mithilfe des Deep Hashing Neuronales Netzwerks Binärcodes generieren. Das Problem dieses Verfahrens ist die Nichtverfügbarkeit von ausreichend beschrifteten Bild Datensätzen, die die Leistung des Deep Hashing Neuronales Netzwerk behindern. Wir schlagen neuartige Methoden, die GANs verwendet, um dieses Problem zu beheben. GAN ist ein generatives Modell zur Modellierung von Datenverteilungen und besteht aus einem Generator und einem Diskriminator. Die beiden Komponenten spielen ein Minimax-Spiel für zwei Spieler. Unsere Forschung untersucht zwei Möglichkeiten, wie der Bildabruf von GAN profitieren kann. Eine Möglichkeit besteht darin, mit GAN Bilder unter bestimmten Bedingungen zu generieren und so den beschrifteten Datensatz zu vergrößern. Mit vergrößerten beschrifteten Daten kann ein überwachtes Deep-Hashing für den Bildabruf trainiert werden. Das andere ist, den Diskriminator von GAN für unüberwachtes Repräsentationslernen zu verwenden. Durch Hinzufügen von binären Hash-Bedingungen und semantischen Bedingungen können wir ein unüberwachtes, kontradiktorisches Hashing-Netzwerk erhalten. Mit diesen Methoden können binäre Hash-Codes für multispektrale Fernerkundungsbilder generiert werden. Experimentelle Ergebnisse zeigen, dass GAN dazu beitragen kann, die auf Hashing basierende Fernerkundungsbildersuche zu verbessern. Die Qualität der generierten Bilder und ihre Auswirkung auf den Bildabruf werden ebenfalls analysiert. Bedingtes GAN kann wirklich realistische Fernerkundungsbilder generieren. Bilder, die aus bedingungslosem GAN generiert wurden, sind weniger realistisch, behalten jedoch einige Eigenschaften realer Bilder bei.

Contents

List of Acronyms	vii
List of Figures	viii
List of Tables	x
1 Introduction	1
1.1 Motivation	1
1.2 Objective	2
1.3 Outline	3
2 Fundamentals of Deep Learning	5
2.1 Convolutional Neural Networks	6
2.1.1 Network Components	8
2.1.2 CNN Architectures	11
2.1.3 Training Method	14
2.2 Generative Adversarial Networks	15
2.2.1 GAN Architecture	16
2.2.2 Training Scheme	17
2.2.3 Evaluation Metrics	17
2.3 Variants of GANs	18
3 Related Work	21
3.1 Image Retrieval Methods	21
3.1.1 Text-based Image Retrieval	21
3.1.2 Content-based Image Retrieval	22
3.2 Remote Sensing Image Retrieval	24
3.3 GANs for Image Retrieval	26
4 Supervised Hashing Boosted by GANs-Generated Images	27
4.1 Methodology	27
4.1.1 GANs for Remote Sensing Image Generation	28
4.1.2 Deep Hashing for Remote Sensing Image Retrieval	30

4.2	Design of Experiments	32
4.2.1	Dataset	32
4.2.2	Implementation Details	34
4.2.3	Evaluation Metrics	35
4.3	Experimental Results	36
4.3.1	Analysis of Image Generation	36
4.3.2	Results of Image Retrieval	39
4.4	Conclusion	40
5	Unsupervised Adversarial Hashing for Multispectral Images	43
5.1	Methodology	44
5.1.1	GAN for Image Retrieval	44
5.1.2	GAN for Multispectral Image Retrieval	45
5.1.3	GAN for Unsupervised Semantic Hashing	46
5.1.4	Semantic Similarity Matrix	49
5.2	Design of Experiments	52
5.2.1	Dataset	52
5.2.2	Implementation Details	56
5.2.3	Evaluation Metrics	58
5.3	Experimental Results	59
5.3.1	Semantic Similarity Matrix Comparison	59
5.3.2	Results of Image Retrieval	60
5.3.3	Analysis of Image Generation	62
5.3.4	Correlation between Image Retrieval and Image Quality	65
5.4	Conclusion	66
6	Conclusion and Future	68
6.1	Conclusion	68
6.2	Future Work	69
	Bibliography	71

List of Acronyms

EO	Earth Observation
ILSVRC	ImageNet Large Scale Visual Recognition Challenge
RS	Remote Sensing
CBIR	Content-Based Image Retrieval
FC	Fully Connected
MLP	MultiLayer Perceptron
CNN	Convolutional Neural Network
NN	Neural Network
GAN	Generative Adversarial Network
IS	Inception Score
FID	Frechet Inception Distance
CGAN	Conditional Generative Adversarial Network
DCGAN	Deep Convolutional Generative Adversarial Network
ACGAN	Auxiliary Classifier Generative Adversarial Network
VAE	Variational AutoEncoder
VAE-GAN	Variational AutoEncoder Generative Adversarial Network
BiGAN	Bidirectional Generative Adversarial Networks
SSM	Semantic Similarity Matrix
ESA	European Space Agency
NIR	Near-InfraRed
SWIR	Short-Wavelength InfraRed
UAHM	Unsupervised Adversarial Hashing for Multispectral Images
MAP	Mean Average Precision
IoU	Intersection over Union
GAN _{UAHM}	GAN for Unsupervised Adversarial Hashing for Multispectral Images

List of Figures

2.1	ImageNet Large Scale Visual Recognition Challenge (ILSVRC) winner methods and their corresponding top-5 error on classification task [22].	5
2.2	An example of CNNs: AlexNet [35].	6
2.3	An simple example of fully connected neural nnetwork.	7
2.4	Flatten operation.	7
2.5	Features extracted in different convolutional layers [38, 73].	8
2.6	The convolution operation.	9
2.7	The padding operation.	9
2.8	The pooling operation.	10
2.9	Different activation functions.	10
2.10	The architecture of LeNet-5 [39].	11
2.11	Comparison between architectures of AlexNet and VGG Net.	12
2.12	Inception Module used in GoogLeNet.	13
2.13	Residual block.	14
2.14	Forwardpass and backwardpass in the neural network [69].	15
2.15	Illustration of a vanilla GAN.	16
2.16	Comparison between the convolution and the transposed convolution [15].	16
2.17	The structure of CGAN and ACGAN	19
2.18	The structure of VAE-GAN and BiGAN	20
3.1	Illustration of content-based image retrieval.	22
3.2	Illustration of Remote Sensing technology.	24
3.3	Illustration of content-based large scale image retrieval system.	25
4.1	The framework of the proposed method: supervised hashing boosted by GANs-generated images.	28
4.2	The architecture of deep hashing network.	31
4.3	Visualization of samples from different classes in EuroSAT. Only RGB bands are shown here.	33
4.4	The detailed architecture of the generator and the discriminator network in Conditonal DCGAN and ACGAN.	34
4.5	Visual comparison of fake images generated from Conditional DCGAN and ACGAN.	36

4.6	The learning process of Conditional DCGAN and ACGAN on EuroSAT remote sensing image dataset.	38
4.7	Visualization of the improvement of image retrieval performance on river and highway class after using GAN-generated images.	41
4.8	Visualization of the image retrieval result. From each class, one image is random selected as the query image. The best model is chosen here: training set includes real images and ACGAN images; Hash code length is 32 bits.	42
5.1	Illustration of image retrieval based on GAN.	45
5.2	GAN architecture used for multispectral images.	46
5.3	Left: Leaky ReLU activation function. Right: Tanh activation function .	47
5.4	GAN architecture used for <i>Unsupervised Adversarial Hashing for Multispectral Images</i>	48
5.5	Illustration of the ablation study. The network on the right is named as DHNN MS.	49
5.6	Visualization of multispectral remote sensing images in EuroSAT. The first column includes RGB images and other columns include single spectral images from different bands. Two images are randomly sampled from classes: Annual Crop, Forest, Herbaceous Vegetation, Highway and Industrial.	53
5.7	Visualization of multispectral remote sensing images in EuroSAT. The first column includes RGB images and other columns include single spectral images from different bands. Two images are randomly sampled from classes: Pasture, Permanent Crop, Residential, River, Sea Lake.	54
5.8	Precision-Recall Curve of image retrieval using different models and different code lengths.	62
5.9	Visualization of image retrieval results using different models. The first column shows the query images. From the second column to the last column, images at 0, 50, 100, 150, 200, 250, 300, 350, 400 of the retrieved ranking list are shown. In each group of three rows, the first row shows the results using GAN RGB, the second row shows the results using GAN MS and the third row shows the results using GAN _{UAHM}	63
5.10	Fake images generated by different models.	64
5.11	A minibatch of real images from the training set.	64
5.12	Visualization of non-RGB bands of generated multispectral images.	65
5.13	Analysis of image quality and image retrieval.	66
5.14	Illustration of the change of image quality and image retrieval during the training process.	67

List of Tables

4.1	Quantitative comparison of fake images generated from Conditional DCGAN and ACGAN.	37
4.2	CIFAR10 experiments using different generative model. IS: higher is better. FID: lower is better. [59]	37
4.3	MAP@100 of image retrieval on testing set using different trained model.	39
4.4	Class-wise MAP@100 of image retrieval on testing set using different trained model.	39
4.5	The change of class-wise MAP@100 of image retrieval on testing set using different trained model.	39
5.1	Different spectral features. h denotes the height of the image. w denotes the width of the image. c denotes the number of channels of the image. n_bins denotes the number of intervals which the whole spectral range is divided into. $n_patches$ denotes the number of patches which a images is divided into.	51
5.2	All 13 bands of multispectral images in EuroSAT. The band name, the purpose, the spatial resolution and the central wave length are listed here.	55
5.3	Illustration of dataset splitting.	55
5.4	The detailed architecture of the generator and the discriminator. l denotes the code length. c denotes the number of images channels.	56
5.5	The Output Part of GAN and GAN_{UAHM}	57
5.6	Semantic Similarity Matrix results comparison.	60
5.7	MAP@100 of image retrieval using different models and different code lengths.	61
5.8	Top20 Precision of image retrieval using different models and different code lengths.	61
5.9	Class-wise MAP of image retrieval using different models. Code length is 128.	62

1 Introduction

1.1 Motivation

Remote sensing is a technology that uses aircraft- or satellite-based instruments to observe the objects on the earth by analyzing propagated signals, e.g., electromagnetic radiation. With the development of the remote sensing technology, more and more remote sensing images with the higher spectral resolution and the higher spatial resolution are produced every day. According to [31], Sentinel satellites which are operated by the European Space Agency can produce approximately 10 TB of Earth Observation (EO) data per day. It is necessary to build an efficient and accurate remote sensing image retrieval system, which will benefit urban area study, forestry research, risk management, etc.

Early remote sensing image retrieval methods mostly depend on manual tags, such as geographical locations, visual descriptions, sensor types, waveband information, etc. These text-based retrieval methods rely on manual annotations, which are expensive and not always available. And the retrieval performance is highly dependent on the quality of the manual annotations. Recent research on remote sensing image retrieval turns to using content-based methods. In the content-based remote sensing image retrieval system, the feature representation of the query image is computed and then compared to the feature representations of all images in the archive. Feature descriptors are usually used to obtain these feature representations. Feature descriptors can be categorized into two groups: handcrafted feature descriptors and data-driven feature descriptors. Handcrafted features include local invariant features [71], morphological features [2], textural features [26], etc. Data-driven features are obtained by utilizing machine learning methods. Hash learning (also known as learning to hash) is based on data-driven features. In this method, a hash function is built to compute for each image the hash codes. Hash lookup can find similar images rapidly using hash tables and hash codes. Therefore, the retrieval accuracy and the retrieval speed can both be improved. A recent study [41] using deep hashing neural networks (DHNNs) has achieved excellent results on large-scale remote sensing image retrieval.

The generative adversarial network (GAN) is firstly proposed in [20]. It is trained using a two-player minimax game. A GAN is composed of a generator and a discriminator. The generator learns to generate fake but realistic images from random low-dimensional embeddings (noise vectors) to deceive the discriminator. The discriminator learns to dis-

1 Introduction

tinguish between real images and generated images. They are alternatively optimized for many iterations. GANs are widely used in image synthesis, data augmentation, and image super-resolution. There is also some research focusing on using a GAN to improve classification. In [52], the authors proposed deep convolutional generative adversarial networks and used the discriminator as a feature extractor for the classification task. Later in [16] and [13], a new structure of the GAN was proposed. They added an encoder to the original GAN, which only has a generator and a discriminator. Three components are trained in an alternative fashion, and the encoder proved to be a competitive feature extractor compared to other weakly supervised learning approaches. Similar to feature learning in the classification task, hash learning can also benefit from the GAN. [6] proposed a method of adding a hashing stream and classification stream to the original GAN to build a better hash function and achieve better image retrieval. [62] used an architecture similar to VAE-GAN [37] and trained it using a binary constraint and a neighborhood similarity constraint on the latent code layer. [11] proposed a method of adding semantic-preserving loss to BiGAN's encoder part [13]. These methods show different ways of applying the GAN to the image retrieval task. Considering the unavailability of sufficient annotations and multispectral characteristics of remote sensing images, this thesis wants to explore efficient and accurate multispectral remote sensing image retrieval methods based on the GAN. .

1.2 Objective

Remote sensing image data are usually very large. Manual annotation of such big data is time-consuming and expensive. And different from natural images, remote sensing images have much lower spatial resolution, which makes the annotation even harder. A lack of sufficient labeled data can be an obstacle in applying deep learning to remote sensing images. The GAN proves to be an effective method for learning the distribution of the data [52] [13]. Then the initial idea is that we can use GANs to generate a large amount of labeled remote sensing images based on a limited number of labeled images. Generated images and real labeled images will then together serve as the training set to learn a deep hashing neural network.

Remote sensing images have multiple bands: if we use transfer learning to extract the features of remote sensing images from CNNs pretrained on other datasets, e.g., ImageNet dataset, only the information stored in RGB bands is used. To make use of all the information stored in all bands, we must train the neural network from scratch. [52] [16] and [13] show that GANs can be used not only for modeling the data distribution (generating images), but also for unsupervised representation learning. Representation learning is closely related to image retrieval because the distance of different representations can be directly the similarity measure. In [11], an empirical study of adding semantic-preserving loss to BiGAN [13] for image search was conducted. In this thesis,

we proposed a novel method called *Unsupervised Adversarial Hashing for Multispectral Images* to achieve efficient and accurate retrieval of multispectral remote sensing images. Our method can generate fake multispectral images and compute hash codes for real multispectral images.

In summary, we will use GANs to increase the amount of labeled data to improve supervised deep hashing for image retrieval as the first step. Then in the second step, we will exploit the representation learning ability of GANs to achieve unsupervised deep hashing for multispectral image retrieval.

1.3 Outline

This thesis is separated into six chapters. A short introduction of each chapter is given below.

Chapter 2 introduces background knowledge from basic CNNs to complicated GANs. The CNN is a widely used deep learning method for computer vision problems. Different layers of CNNs are introduced in this chapter. State-of-the-art CNN architectures are compared. And backpropagation, an efficient weight updating method are also explained in this chapter. Then the details of the GAN are introduced in the similar way. Besides, we list and introduce several evaluation metrics of GANs. Five variants of GANs, which are CGAN [46], DCGAN[52], ACGAN[47], VAE-GAN [37]and Bi-GAN[13][16] are introduced in the last section of this chapter.

Chapter 3 introduces related work on image retrieval. Image retrieval methods can be divided into two groups: text-based image retrieval and content-based image retrieval. Content-based image retrieval (CBIR) makes use of visual information that can be directly derived from the image e.g. color, shape, texture, and other features. The details of CBIR are provided in this chapter. The second section of this chapter introduces some methods that prove to be effective in remote sensing image retrieval. In the last section, some publications using GAN to improve image retrieval are reviewed.

Chapter 4 introduces the basic method of using different GANs to generate fake images, which are aimed at helping the training of supervised deep hashing. The loss functions and network architectures are described in this chapter. We also give an introduction of the dataset used in this thesis: EuroSAT [24]. Following that, the experimental details and final results are presented and analyzed.

Chapter 5 thoroughly describes a newly proposed method called *Unsupervised Adversarial Hashing for Multispectral Images*. We show the difference between this method and other GAN-based hashing methods. We also show how this method is implemented

1 Introduction

and how the network is trained. The multispectral version of the dataset, EuroSAT, is introduced and visualized in this chapter. In the experiment part, a lot of experiments have been done to show how each component in our methods helps to improve the image retrieval performance.

Chapter 6 gives a summary of all the methods and experimental results. Two future research directions are provided: finding the best band combination for image retrieval and using other advanced GAN architectures.

2.1 Convolutional Neural Networks

CNN makes use of the hidden patterns in a hierarchical way. Low-level patterns are recognized in the first several layers and then assembled to form high-level patterns. The most important part of CNN is the convolutional layer.

In the traditional fully connected neural network (also called multilayer perceptron), all the input data are fed into the first layer and do the dot production with weight and bias matrix. After the activation function, all the outputs of one layer are fed into the next layer. These layers are called fully connected layers or dense layers. Figure 2.3 shows a simple example of a fully connected neural network. For image data, a flatten operation is necessary to convert a 2D image to a 1D vector as a preprocessing step. The disadvantage of this method is that it does not make use of spatial dependencies and temporal dependencies (for video data) and treat pixels that are close or far away in completely the same way as illustrated in Figure 2.4.

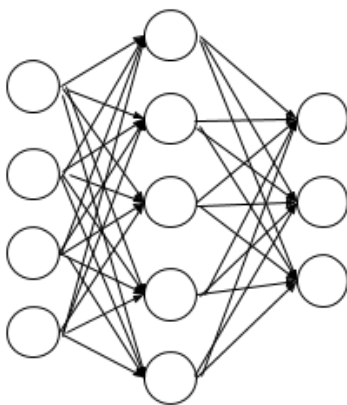


Figure 2.3: An simple example of fully connected neural nnetwork.

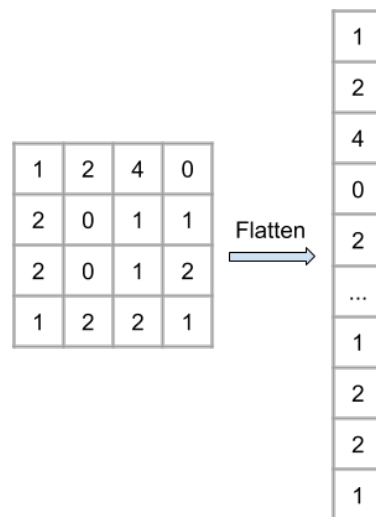


Figure 2.4: Flatten operation.

Unlike the fully connected neural network, the convolutional neural network adds convolutional layers to extract features hierarchically. The main idea of the convolution operation is to use convolutional filters in small restricted areas (also called receptive fields) from the input data. As a result, it can make use of the spatial or temporal structure of the data, thus extracting more informative features. What's more, the reusing of a filter in one layer can reduce the number of parameters drastically.

Nowadays, neural networks go deeper and deeper. Figure 2.2 presents the architecture of AlexNet [35] which is the first winner that uses CNNs in the ImageNet Large Scale Visual Recognition Challenge [12]. There are five convolutional layers in AlexNet. The features extracted in the convolutional layers are getting less abstract with the data

2 Fundamentals of Deep Learning

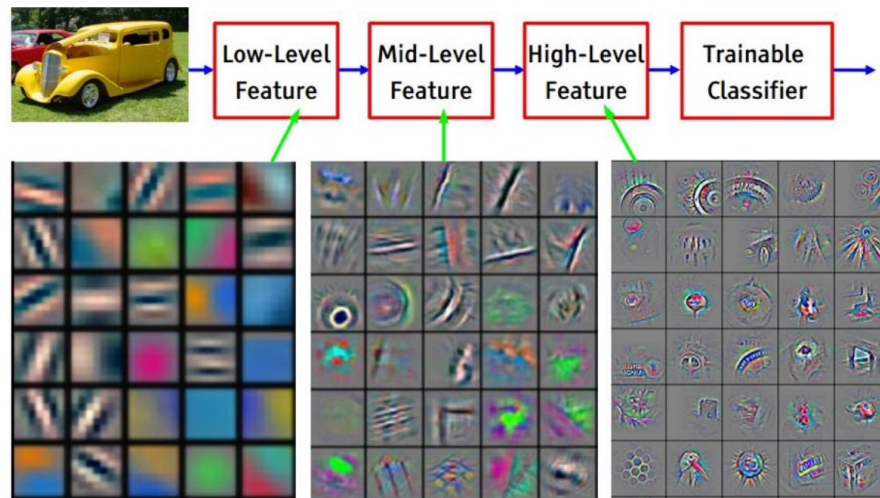


Figure 2.5: Features extracted in different convolutional layers [38, 73].

going toward the output. This phenomenon is illustrated in Figure 2.5. The details of the convolution operation will be introduced in 2.1.1.

The convolutional layer is inspired by how visual information is processed in the human brain. The visual cortex is the part of the cerebral cortex that receives, integrates and processes visual information relayed from the retinas. Neurons in the visual cortex often only respond to the stimuli in a specific receptive field. And neurons in different parts of the visual cortex respond to different stimuli. When the visual information is passed from one area to another area in the visual cortex, the corresponding cortex area is becoming more and more specialized [28].

In the following several subsections, the details of CNN components, different CNN architectures, and training methods will be introduced.

2.1.1 Network Components

The usual CNN is composed of several different layers such as convolutional layers, pooling layers, and fully connected layers. These layers have different functions and are placed according to some fixed rules. It's worth noting that not all CNNs have three above-mentioned layers. The CNN denotes the neural network which has at least one convolutional layer. For example, the fully convolutional neural network which only has convolutional layers is used in [44] to do image semantic segmentation and in [9] to perform object detection. Also, some networks include special operations, such as residual blocks in ResNet [22] and transposed convolutional layers in the GAN [20]. We will discuss the residual block in Section 2.1.2 and the transposed convolutional layer in Section 2.2.1.

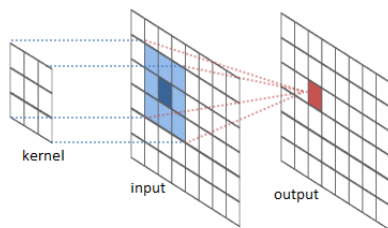


Figure 2.6: The convolution operation.

0	0	0	0	0	0	0
0	1	2	3	2	1	0
0	2	2	2	2	2	0
0	3	2	0	2	2	0
0	2	2	2	2	2	0
0	1	3	3	2	0	0
0	0	0	0	0	0	0

Figure 2.7: The padding operation.

Convolutional Layer

The convolutional kernel (also called the filter) plays an important role in the convolution operation. The convolutional kernel is basically a matrix. The value of each entry of the matrix will be initialized and then updated during the learning process. When doing the convolution operation, the kernel will shift from left to right and from top to bottom on the image. At each shift step, the kernel matrix will be multiplied by the pixel value matrix where the kernel is located at that step. The multiplication result will be the value of the corresponding entry of the output matrix. Figure 2.6 illustrates how the convolution operation is conducted.

The shape of a kernel is usually defined by $width \times height \times depth_{in} \times depth_{out}$. $width$ and $height$ determine the size of the kernel and also determine how large the receptive field is. $depth_{in}$ and $depth_{out}$ denote the number of channels of the tensor data. $depth_{in}$ is the same as the number of channels of the input data. $depth_{out}$ decides the number of channels of the output data. We can also regard $depth_{out}$ as the number of filters of one layer. In AlexNet, kernel size includes 11×11 and 5×5 . Later research [64] proposed to use two 3×3 kernels to replace one 5×5 kernel. This replacement not only reduces the number of parameters but also helps in extracting better features. Usually, the selection of the kernel size depends on the image data.

The stride decides how many pixels the kernel should move through every after a shift step. The bigger the stride is, the smaller the size of the output data is.

Padding is used to adjust the input size to meet our requirements for the output size. Padding will increase the width and height of the input data and assign values to these newly added entries. Zero padding is the most widely used padding method. In some case, nonzero padding or reflection padding is more reasonable and effective. As illustrated in Figure 2.7, when there is a convolution operation between a 5×5 image and a 3×3 kernel using stride 1, the size of the output data will still be 5×5 if we use padding. Otherwise, the size of the output data will be 3×3 . This is because the entries on the edge are less processed than those in the center without padding. As a result, some information may be lost during the forward pass. So the other advantage of

2 Fundamentals of Deep Learning

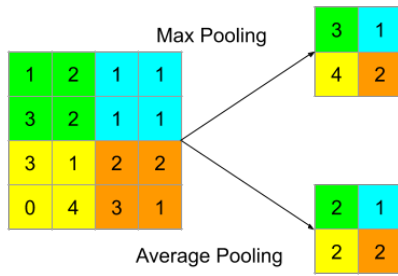


Figure 2.8: The pooling operation.

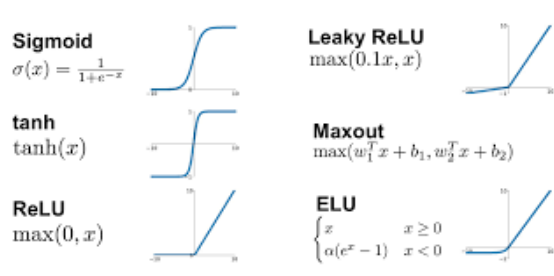


Figure 2.9: Different activation functions.

padding is reducing information loss.

Pooling Layer

Pooling layers are usually placed after convolutional layers. They can reduce the spatial size of the data effectively as shown in Figure 2.8. Therefore, the number of parameters in the network is reduced as well. There are different kinds of pooling layers: max pooling, average pooling, L2-norm pooling, etc. The filter size and stride also apply to pooling operation. The most commonly used pooling layer now is max pooling with stride equaling 2 and filter size equaling 2 or 3.

Fully Connected Layer

The fully connected layer builds connections between all output neurons and all input data as shown in Figure 2.3. The connection here means a calculation involving input neurons, the weight matrix, the bias, and the activation function. In the mathematical form, it can be formulated as Equation 2.1.

$$\mathbf{x}_{k+1} = act(\mathbf{w} * \mathbf{x}_k + \mathbf{b}) \quad (2.1)$$

where $act(\cdot)$ denotes the activation function which will be introduced in the next subsection. \mathbf{w} denotes the weight matrix. And \mathbf{b} denotes the bias.

Fully connected layers are usually placed in the last several layers of a CNN. It will process flattened tensor data and decrease the number of output until finally, the number of entries in the output vector becomes the same as the number of categories in a classification task.

Activation Functions

Activation functions are motivated by the biological phenomenon as well. The dendrites of a neuron will perceive the stimuli and carry the signal to the neuron cell body. When the signals summed at the neuron cell body reach a specific threshold, the neuron will

fire, sending the electrical signal along the axon. We model the firing rate with the activation function.

Each activation function performs a fixed mathematical computation on the input data. Most layers in CNNs include activation function as the last operation before the output, like convolutional layers, fully connected layers and transposed convolutional layers. Some commonly used activation functions are shown in Figure 2.9

2.1.2 CNN Architectures

A convolutional neural network is composed of different layers which are introduced in Section 2.1.1. By permuting the order of these layers, different architectures can be constructed. [39] proposed a convolutional neural network, called LeNet-5, to automatically recognize handwritten characters from document images in 1998. Figure 2.10 shows the architecture of LeNet-5 which is composed of two convolutional layers, two max pooling layers, and three fully connected layers. Recently more and more new architectures have been designed and proposed. Several important architectures are proposed in the ILSVRC [12] competition. There is a tendency that the architectures are getting more complicated, and the networks are getting deeper. In these newly proposed architectures, some innovative modification on the network structure can achieve great improvement compared to the state-of-the-art. We will introduce some of them in the following several paragraphs.

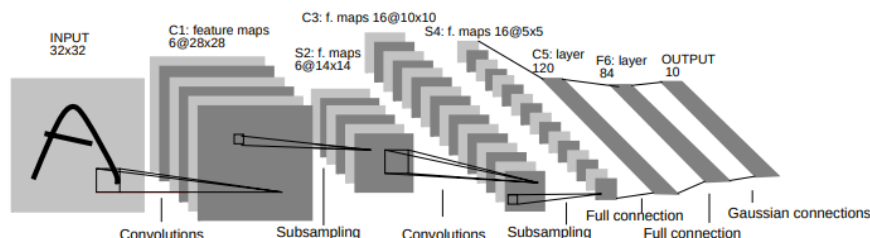


Figure 2.10: The architecture of LeNet-5 [39].

AlexNet

AlexNet [35] was the first winner that used the CNN-based method in ILSVRC. Its architecture is shown in Figure 2.2. The architecture is very similar to LeNet-5. The differences are that AlexNet has more convolutional layers and more filters at each layer than LeNet-5.

It was also the first one to use ReLU as the activation function in the CNN. And it firstly introduced local response normalization(LRN).

2 Fundamentals of Deep Learning

VGG

VGG [60] was the runner-up method in the ILSVRC 2014 competition. It increased the number of layers to 16 (VGG 16) and 19 (VGG 19). In VGG, all convolutional filters have the size of 3×3 and the stride of 1. All pooling layers have the size of 2×2 and the stride of 2. Compared with AlexNet, VGG Net uses a smaller filter size and a deeper network. This design needs fewer parameters with an even deeper network due to the smaller filter size. The advantage of the deeper network is that the deeper network can model more nonlinearities.

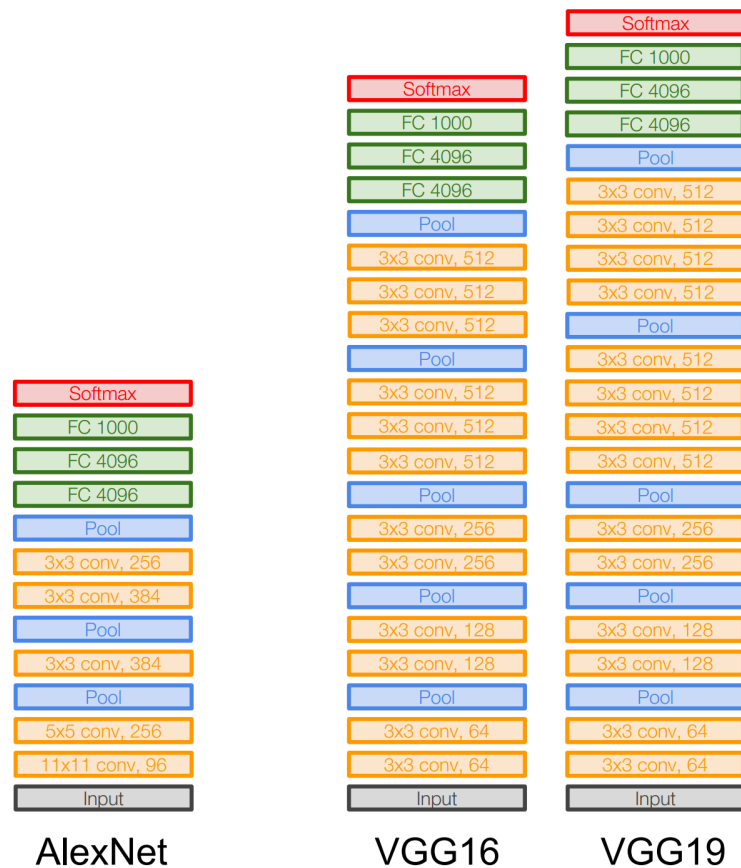


Figure 2.11: Comparison between architectures of AlexNet and VGG Net.

GoogLeNet

GoogLeNet introduced a novel element to the network which is called Inception module. So GoogLeNet is also called Inception v1. In an Inception module, convolutional operations will be applied on the output of the previous layer in a parallel way as shown

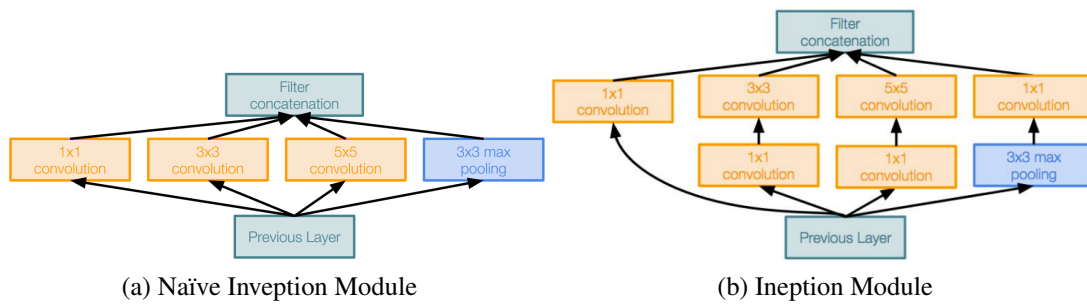


Figure 2.12: Inception Module used in GoogLeNet.

in Figure 2.12a. To reduce the number of parameters, they added several 1×1 convolutional layers which aim to reduce the number of channels. Many Inception Modules are stacked to build the core part of GoogLeNet. There are two auxiliary classification outputs in the middle layer to prevent the gradient from vanishing in lower layers.

There are improved versions of GoogLeNet (Inception v1): Inception v2 [29], Inception v3 [66] and Inception v4 [65]. Inception v2 added batch normalization to accelerate the convergence. Inception v2 also replaced the 5×5 filter in Inception v1 with two 3×3 filters. Inception v3 redesigned the Inception module using the smaller filter size and rearranged the network. Inception v4 added residual connections to increase the learning speed.

ResNet

ResNet[22] is proposed mainly to solve such a problem: in some cases, the result gets worse when the convolutional neural network gets deeper,. But the deeper model should be at least as good as the shallow model because the additional layers can be set as identity mapping. The residual block addressed this problem by learning a residual function with respect to the input. This process can also be regarded as a refinement step to adjust the input feature map rather than build a completely new feature map. When the feature map does not need refinement anymore, the learned residual gradually gets close to zero which produce the identity mapping layer. Residual blocks are stacked to build the whole architecture of the ResNet. When the network is very deep, an improved residual block with 1×1 filter for reducing the number of feature map channels is proposed. Figure 2.13 illustrates the basic residual block and the improved residual block.

2 Fundamentals of Deep Learning

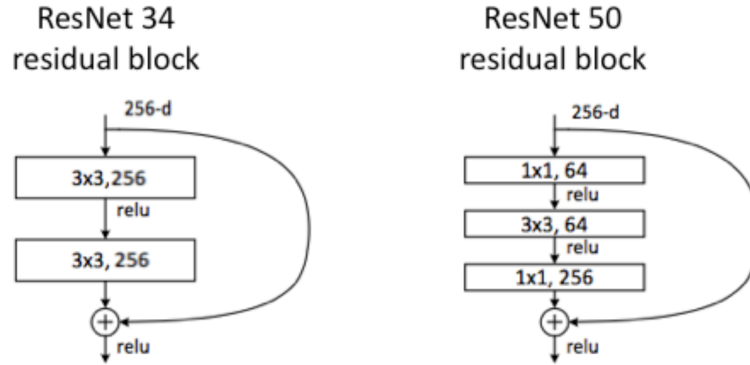


Figure 2.13: Residual block.

2.1.3 Training Method

Loss Function

The loss function is the objective function of the training process. It describes how good your model (CNN) is at the expected task. The aim of the training process is to optimize the loss function.

For the basic regression task, the mean squared error loss and mean absolute error loss are two commonly used loss functions. For the classification task, the most common loss function is the cross entropy loss. The cross entropy loss was initially used for binary classification. By introducing one-hot encoding, the cross entropy loss can also be used in multiclass classification. The cross entropy loss is calculated by:

$$L_{crossentropy} = -\frac{1}{N} \sum_{n=1}^N [y_n \log \hat{y}_n + (1 - y_n) \log (1 - \hat{y}_n)] \quad (2.2)$$

where y_n denotes the real class, and \hat{y}_n denotes the predicted class.

The triplet loss is a new loss function for learning the similarity between images. It was proposed in [8] and applied to face recognition where it achieves a big performance improvement [57]. In this method, all training samples are used to build triplets according to their labels. A triplet is made up of an anchor sample, a positive sample and a negative sample. The triplet loss is calculated by:

$$L_{triplet} = \sum_i^N [\|f(x_i^a) - f(x_i^p)\|_2^2 - \|f(x_i^a) - f(x_i^n)\|_2^2 + \alpha]_+ \quad (2.3)$$

where $f(\cdot)$ denotes the trained model. x_i^a denotes the anchor sample, x_i^p denotes the positive sample and x_i^n denotes the negative sample. α is the margin between positive and negative pairs.

There are also a lot of other loss functions for specific purposes, like the adversarial loss in the GAN and the binary loss in image hashing retrieval. We will discuss these loss functions in the later related part.

Backpropagation

Backpropagation is a gradient descent approach using the chain rule in computing derivatives. As shown in Figure 2.14, the forward pass processes the data hierarchically to obtain a prediction, and the backward pass uses the chain rule to compute the derivatives of the loss with respect to the parameters in each layer.

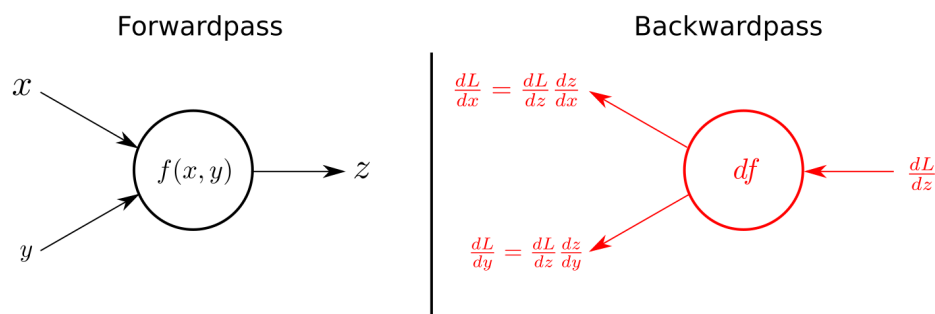


Figure 2.14: Forwardpass and backwardpass in the neural network [69].

2.2 Generative Adversarial Networks

The GAN was initially proposed in [20] as an approach to learning a generative model. The generative model can generate new data from a distribution in which the training data lie. And the generative model tries to address density estimation problem which is important for unsupervised learning. Some generative model methods give an explicit density function: like PixelCNN [48], PixelRNN [49] and VAE [33]. However, the generative adversarial network will not give an explicit density function but directly generate new data from the density function. So we can say that the GAN learns an implicit data distribution.

Even though the GAN is initially proposed as a generative model, some research has also shown it can benefit unsupervised representation learning, semi-supervised learning and supervised learning. GAN has a lot of different applications, e.g., image super resolution, image colorization, text-to-image translation, etc.

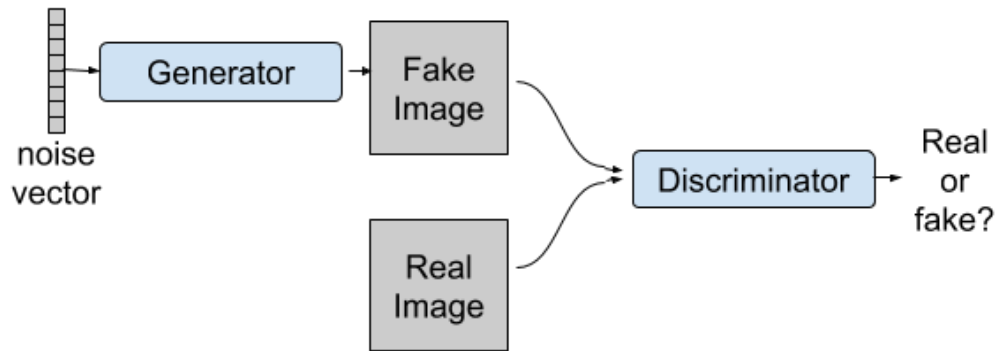


Figure 2.15: Illustration of a vanilla GAN.

2.2.1 GAN Architecture

A vanilla GAN is composed of a generator and a discriminator. Basically, the generator and the discriminator are both deep neural networks.

The input of the generator is a noise vector and often sampled in the range $[0, 1]$ using random uniform distribution. To generate an image from a noise vector, the network of the generator is made up of several transposed convolutional layers (sometimes also called deconvolutional layers, but this name is misleading because this operation is very different from mathematical term deconvolution). Figure 2.16 shows the comparison between the convolution and the transposed convolution.

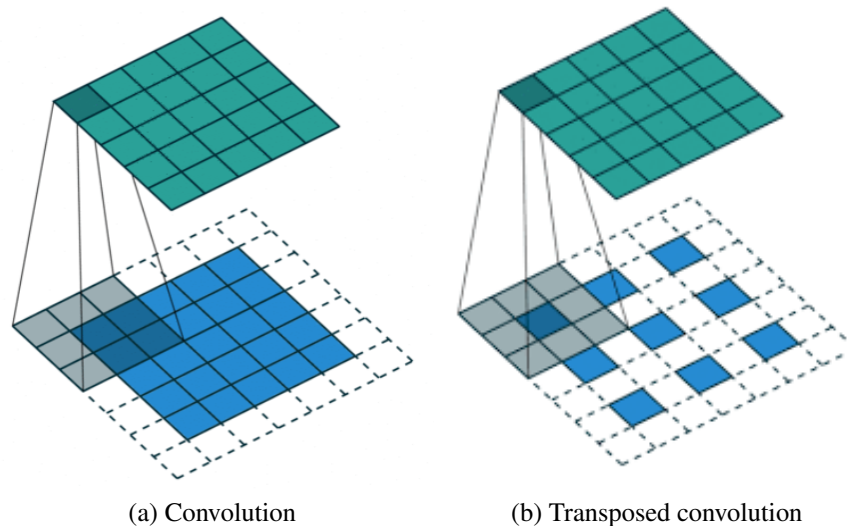


Figure 2.16: Comparison between the convolution and the transposed convolution [15].

The input of the discriminator is images, either real images from the dataset or fake images generated from the generator. The discriminator is essentially a CNN with binary source prediction.

2.2.2 Training Scheme

The training of a generative adversarial network is similar to a minimax two-player game. The objective function can be expressed as Equation 2.4

$$\min_G \max_D V(D, G) = \mathbf{E}_{\mathbf{x} \sim p_{data}(\mathbf{x})} [\log D(\mathbf{x})] + \mathbf{E}_{\mathbf{z} \sim p_{\mathbf{z}}(\mathbf{z})} [\log(1 - D(G(\mathbf{z})))] \quad (2.4)$$

where \mathbf{z} denotes the noise vector, \mathbf{x} denotes the real image, $G(\mathbf{z})$ denotes the image generated by the generator, and $D(\cdot)$ denotes the output of the discriminator which is a likelihood in $(0, 1)$. The discriminator is trained to maximize the objective $V(D, G)$ while the generator is trained to minimize the objective $V(D, G)$

However, experiments show that using the above objective function can not optimize the generator very well. Usually, another training strategy is used when we train the generator. The new strategy tries to maximize the likelihood of the discriminator being wrong instead of minimizing the discriminator being correct. The improved objective functions are shown below:

1. Training the discriminator

$$\max_D \mathbf{E}_{\mathbf{x} \sim p_{data}(\mathbf{x})} [\log D(\mathbf{x})] + \mathbf{E}_{\mathbf{z} \sim p_{\mathbf{z}}(\mathbf{z})} [\log(1 - D(G(\mathbf{z})))] \quad (2.5)$$

2. Training the generator

$$\max_G \mathbf{E}_{\mathbf{z} \sim p_{\mathbf{z}}(\mathbf{z})} [\log(D(G(\mathbf{z})))] \quad (2.6)$$

2.2.3 Evaluation Metrics

To evaluate our GAN and the quality of generated images, visual comparison with real images is not convincing enough. Many metrics for GAN evaluation have been proposed. Here we introduce two commonly used metrics: Inception Score (IS) [56] and Fréchet Inception Distance (FID) [25].

IS evaluates how dispersed generated images are across different classes and how focused the prediction of one specific image is on one class. All generated images will be fed into Inception v1 to get the prediction probability and then to compute the final IS. Inception Score is calculated by:

$$IS(G) = \exp(E_{x \sim p_g} [D_{KL}(p(y|\mathbf{x}) || p(y))]) \quad (2.7)$$

2 Fundamentals of Deep Learning

where D_{KL} denotes Kullback-Leibler divergence which can measure the difference between two probability distributions as expressed in Equation 2.8.

$$D_{KL}(P(x)||Q(x)) = \sum_{x \in X} P(x) \log \frac{P(x)}{Q(x)} \quad (2.8)$$

FID evaluates the distance of feature embedding between real images and fake images by considering their statistical characteristics. The last pooling layer of Inception v3 is used as the coding layer for all generated images. The improvement of FID with respect to IS is that FID takes both real images and generated images into consideration. IS only considers generated images. Fréchet Inception Distance is calculated by:

$$FID(G) = \|\mathbf{m}_r - \mathbf{m}_g\|^2 + Tr(\mathbf{C}_r + \mathbf{C}_g - 2(\mathbf{C}_r \mathbf{C}_g)^{1/2}) \quad (2.9)$$

where $(\mathbf{m}_r, \mathbf{C}_r)$, $(\mathbf{m}_g, \mathbf{C}_g)$ denote the mean and covariance of the feature embedding of the real images and generated images, respectively.

Higher Inception Score and lower Fréchet Inception Distance mean a better GAN and better quantitative results of generated images.

2.3 Variants of GANs

After the first GAN paper [20] was published, GAN attracted more and more attention from researchers around the world. GAN is an insanely active topic in deep learning and the computer vision community. A lot of GAN variants were proposed and used for different purposes. In this section, five representative variants will be introduced and explained in detail. They are CGAN [46], DCGAN [52], ACGAN [47], VAE-GAN [37], BiGAN [13].

CGAN

Compared with the vanilla GAN, CGAN which is short for "conditional generative adversarial network", adds a condition input y to both the generator and the discriminator. This modification is illustrated in Figure 2.17a with a yellow color. The condition input y can be any kind of auxiliary information, such as class labels or data from other modalities.

DCGAN

In vanilla GAN, MLP (Multilayer Perceptron) is used as the architecture of the discriminator and the generator. The DCGAN paper [52] gives some tricks for a stable deep convolutional GAN. In DCGAN, fully connected layers and pooling layers are replaced

with convolutional layers and transposed convolutional layers. They use Batch Normalization [30] to stabilize training by normalizing the input to make it have zero mean and unit variance. In the generator, ReLU activation function is applied to all layers except for the output, which uses Tanh. In the discriminator, Leaky ReLU activation function is applied to all layers.

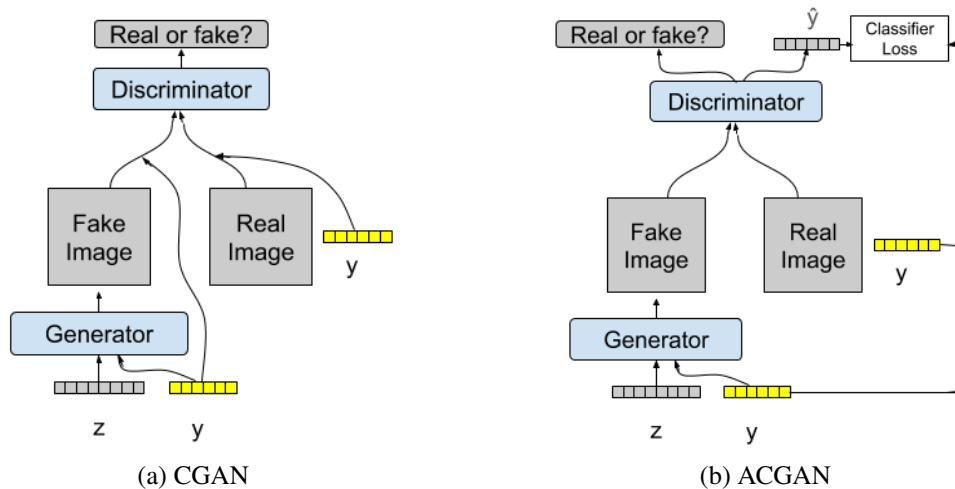


Figure 2.17: The structure of CGAN and ACGAN

ACGAN

Same as CGAN, ACGAN is also doing conditional image generation, in which fake images are generated from a combination of the noise vector z and the condition information y . But different from CGAN which takes y as one of the discriminator's inputs, ACGAN only takes images as the discriminator's input. The output of the discriminator is a combined prediction on the source (real or fake) and class labels. ACGAN is trained by optimizing the joint loss including adversarial loss and classifier loss. The experiment shows that ACGAN-generated images have better global coherence.

VAE-GAN

VAE-GAN is a hybrid model of VAE [33] and GAN [20]. Figure 2.18a illustrates the structure of VAE-GAN. The encoder takes a real image as the input and then outputs its latent representation. The decoder (or the generator) takes the latent representation as the input and then outputs a reconstructed image. The decoder also takes noise vectors as the input and outputs fake images. The discriminator will distinguish between reconstructed (fake) images and real images. The training goal is to make the reconstructed images close to original images, make the discriminator not able to distinguish

between real and fake images, and make the latent representations close to the normal distribution.

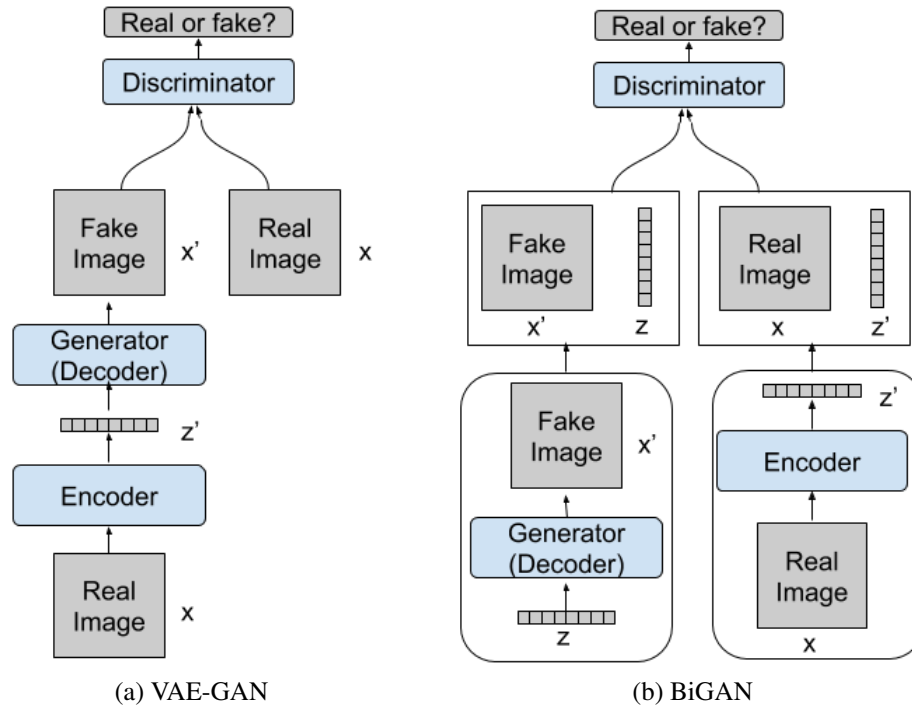


Figure 2.18: The structure of VAE-GAN and BiGAN

BiGAN

BiGAN is similar to VAE-GAN. But it separates the decoder and encoder to two blocks. And the input of the discriminator is also different. Figure 2.18b illustrates the structure of BiGAN. The decoder (or the generator) takes a noise vector z as the input and outputs a generated fake image x' . The encoder takes a real image x as the input and output its latent representation z' . Then the input and the output of both the generator and the encoder will build a pair (x', z) or (x, z') . The pairs serve as the input of the discriminator. The discriminator distinguishes whether the pair is from the encoder or the generator.

3 Related Work

Image retrieval is also called image search, where the system can return similar images when a query term is given. Most early traditional methods make use of metadata, such as keywords, text, description, etc. Image retrieval will be conducted based on metadata associated with each image. A new method called content-based image retrieval (CBIR) has proven to be very efficient and accurate and demands less human work for preprocessing. It makes use of the visual contents that can be derived from the image itself [10]. Among all CBIR methods, hashing learning methods that represent each image as binary hash codes gain a lot of attention. It can not only improve the retrieval accuracy but also speed up the retrieval process.

Image retrieval is a hot topic in two different research communities which are the information management system community and the computer vision community. Early methods based on metadata are mostly used by the former community and belong to a big category: text-based image retrieval. The computer vision community mainly focuses on content-based image retrieval research. Deep learning is a novel method that shows great improvement compared to traditional methods in many computer vision fields. Recently, some researchers proposed that deep learning can be used to extract features for image retrieval and an end-to-end deep learning-based image retrieval network can be trained.

This chapter is divided into three parts: image retrieval methods, remote sensing image retrieval and GANs for image retrieval. In the first part, a review on image retrieval methods will be given. And different methods will be explained in detail and compared with each other. In the second part, some remote sensing images' characteristics are introduced, and a review on remote sensing image retrieval will be given. In the last part, some latest publications using GAN for image hashing or image retrieval will be introduced.

3.1 Image Retrieval Methods

3.1.1 Text-based Image Retrieval

The text-based image retrieval dates back to 1970s. Usually, before the retrieval process, all the images must be manually annotated with keywords, texts, descriptions, etc. And the annotations include not only the content of the image, but also some auxiliary

3 Related Work

information, like image size, image format, shooting time and address, and camera info. Then some text retrieval techniques will be used, e.g., bag of words approach, vector space model, and Boolean search of words.

But there exist limitations in this method. The retrieval accuracy is highly dependent on the quality of the annotation. And with the dataset getting larger and larger, manual annotation becomes time consuming and expensive. Sometimes the dataset is so large that it is impossible to give annotation to all images in it. On the other hand, annotation is never complete. Some things in the image are hard to express or can be expressed in different ways. The text-based image retrieval are not robust enough due to these problems.

In some special cases, text-based image retrieval can also be done without annotations. One case is to make use of the text present in the images [5]. This method involves text detection, extraction, and optical character recognition. The text-based retrieval is done on the extracted text. This case only applies to images with text in it, like book cover images, document images, and TV news images. Another case is to do automatic image caption. And then generated captions can be used for text-based image retrieval.

3.1.2 Content-based Image Retrieval

Due to the rapid advances in information technology and digital imaging devices, numerous images are produced in different fields. To make use of the information stored in these images, an efficient image search system is necessary. Even though there existed a lot of research on text-based image retrieval methods[7, 36], the manual annotation on the large dataset is impractical and it's hard to get the perfect annotation. These drawbacks of text-based image retrieval hindered its popularity in dealing with large image data and make content-based image retrieval (CBIR) gain more attention.

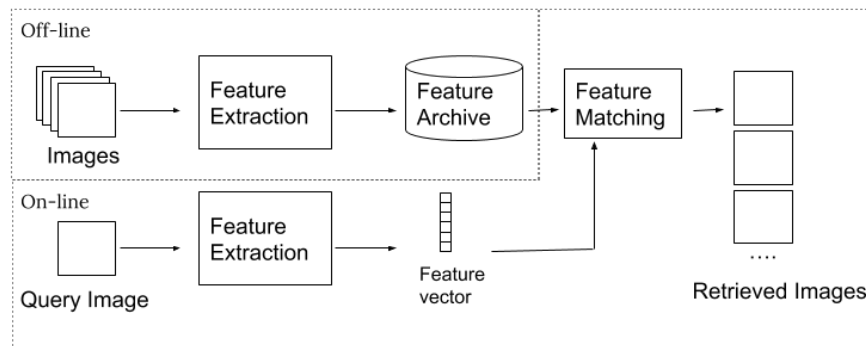


Figure 3.1: Illustration of content-based image retrieval.

In content-based image retrieval, visual contents derived from the image itself are

used for image description and similarity comparison. Visual contents are usually called features which include color, texture, shape, etc.

Color is a widely used feature for image retrieval. There are several different color features: the color histogram, the color moments, and the color coherent vectors. The color histogram method was firstly proposed in [63]. It makes use of the distribution of different colors in an image. In the first step, the whole color range is divided into many small color ranges (usually called bins). Then the number of pixels that fall in each range will be counted. There are two versions of color histogram methods: the global version and the local version. Their differences lie in different scopes to compute the histogram. The global color histogram is the distribution from the whole image. The local color histogram is the distribution in small patches, and then all the distributions of patches are concatenated. Color moments [27] of a color distribution are similar to the central moments of a probability distribution. The mean value, the standard deviation, the skewness, and the kurtosis are computed as the color moments. These low-order moments contain a lot of information that can be used to retrieve similar images. Color coherent vector method classifies each pixel in a given color bucket as either coherent or incoherent, based on whether or not it is part of a large similarly colored region. A color coherence vector stores the number of coherent versus incoherent pixels with each color [50]. The biggest advantage of color feature is its scale, translation, and rotation invariance.

Texture features contain important information on the structural arrangement of a surface and their relationship to the surrounding environment [21]. Early texture features include statistics extracted from the co-occurrence matrix [21] and computational approximation texture features inspired by psychology studies [67]. Later when wavelet transform was introduced in the 1990s, a lot of new texture representations were explored. [61] used statistics of wavelet subbands as the texture representation. The Gabor filter is another widely used texture feature extraction method for image retrieval [45]. The Gabor filter is basically a group of wavelets of which each wavelet captures energy at a specific frequency and a specific direction.

The shape feature is another important image content description method. Various shape descriptors have been proposed in the last three decades. They can be categorized into two groups: the contour-based shape descriptor and the region-based shape descriptor. The former one only uses the contour to extract the features, while the latter one uses the entire shape region [54]. A typical contour-based shape descriptor is the Fourier descriptor. The Fourier descriptor uses the Fourier transformed contour as the feature representation. Common region-based shape descriptors use the moment to describe shapes. In [27], Hu selected seven moments that are invariant to transformations.

Although there exists numerous research on using the above mentioned features to realize CBIR, the semantic gap is still an unsolved big problem. With the advances in deep learning and its increasing popularity in computer vision community, extensive

3 Related Work

research effort has been devoted to bridging the semantic gap between those low-level features and high-level semantic concepts perceived by the human [70]. Distance metric learning for image retrieval is a deep learning based method which aims to learn an optimal distance metric. The optimal metric minimizes the distance of similar images and maximizes the distance of dissimilar images. Two constraints can guide the learning process, the pairwise constraint or the triplet constraint. ImageNet, a large scale image dataset, becomes a benchmark to test deep learning methods in computer vision. And image retrieval can also benefit from those convolutional neural networks pretrained on ImageNet. The first way is to use the output of last several fully connected layers as the feature representation. The second way is to refine the last several layers using metric learning methods. The third way is to retrain the whole network with a similarity loss.

Deep hashing neural network is an extension on deep metric learning for image retrieval. It learns binary hash codes for fast image retrieval [42].

3.2 Remote Sensing Image Retrieval

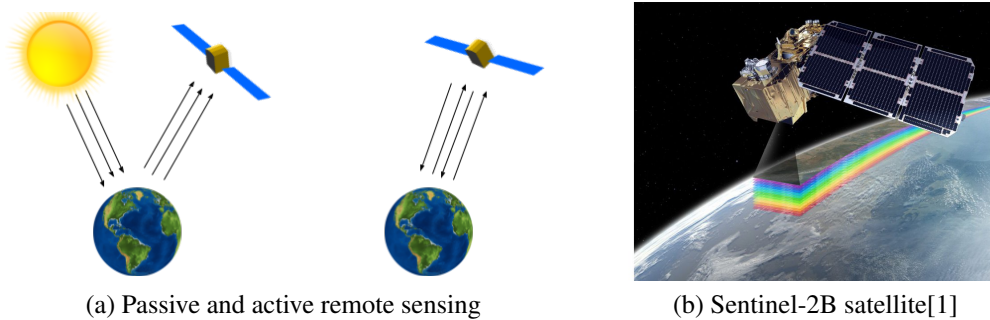


Figure 3.2: Illustration of Remote Sensing technology.

Remote sensing is a term which refers to the use of sensors on the aircraft or satellite to observe the objects on the earth based on propagated signals (e.g., electromagnetic radiation). Depending on the source of the signal, remote sensing can be divided into two categories: passive remote sensing and active remote sensing. As illustrated in Figure 3.2a, passive remote sensing instrument detects the radiation emitted or reflected by the objects. Sunlight is a common source of radiation in passive remote sensing. While active remote sensing instrument can emit radiation towards target objects by itself and then detect the reflected radiation. Some examples of passive remote sensing technology are photography, infrared devices, and radiometers. Some examples of active remote sensing technology are synthetic aperture radar (SAR) and light detection and ranging (LiDAR). Remote sensing image analysis has attracted great interest in the academic community because of its broad application, e.g., climate analysis, urban area

3.2 Remote Sensing Image Retrieval

study, forestry research, risk and damage management, water quality assessment, and crop monitoring.

The advances in sensor development lead to higher spectral resolution and spatial resolution of the remote sensing images. And as time goes by, more and more remote sensing images are produced and stored every day. According to [31], Sentinel satellites which are operated by the European Space Agency can produce approximately 10 *TB* of Earth Observation (EO) data per day. We have already entered an era of remote sensing big data (RSBD). It is necessary to build an efficient image retrieval system to manage such a large amount of remote sensing data.

As we already mentioned, there are two kinds of image retrieval systems: the text-based image retrieval system and the content-based image retrieval system. Remote sensing image retrieval methods can also be categorized into these two groups.

Early remote sensing image retrieval methods mostly depend on manual tags including geographical locations, visual description, sensor types, waveband information, etc. These text-based retrieval methods are time-consuming. The manual annotations are expensive and not always available. In addition, the retrieval performance is highly dependent on the quality of the manual annotation.

Recent research on remote sensing image retrieval turns to using content-based methods. In the content-based remote sensing image retrieval system, the feature representation of the query image is computed and then compared to the feature representations of all images in the archive. Finally, the system will output a ranking list where more similar images lie in the higher position. Figure 3.3 is the flowchart of a content-based remote sensing image retrieval system.

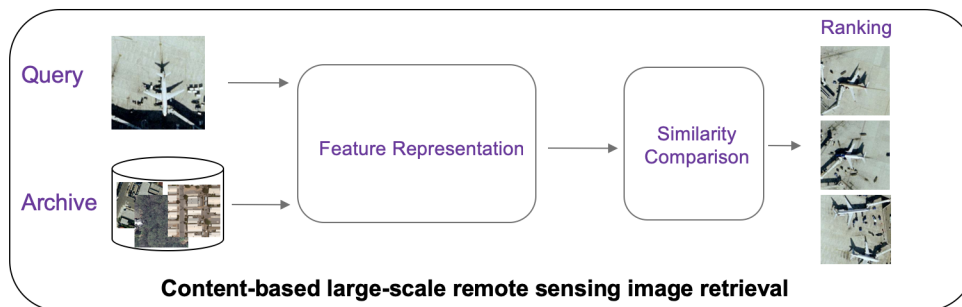


Figure 3.3: Illustration of content-based large scale image retrieval system.

Numerous feature descriptors have been designed for indexing remote sensing images. Feature descriptors can be divided into two types: hand-crafted feature descriptors and data-driven feature descriptors. Hand-crafted features include local invariant features [72], morphological features [3], textural features [26], etc. Data-driven features are obtained by utilizing machine learning methods. This kind of features contains complicated semantic information because of the learning process on the big data set.

3 Related Work

Hash learning method (also known as learning to hash) is based on data-driven features. In hash learning, a hash function is built to compute the index of the hash table for each image. Hash lookup is used to find similar images stored in the hash table rapidly. Therefore, both the retrieval accuracy and the retrieval speed can be improved in hash learning.

Deep learning is a very important member of machine learning family to learn the data representation. Deep neural networks (DNNs) are a commonly used deep architecture of deep learning. Deep learning has shown its potential to extract meaningful features and to do accurate image classification. By making use of deep learning's automatic feature extraction ability, deep neural networks for image hashing can definitely benefit remote sensing image retrieval. [41] proposed to construct a deep hashing neural network, which is composed of feature learning part and hash learning part for remote sensing image retrieval. [53] proposed to train a deep hashing neural network with transfer learning methods. They used ImageNet-pretrained Inception network to extract features from remote sensing images.

3.3 GANs for Image Retrieval

GAN is a newly invented machine learning method. It is a generative model that can learn an implicit data distribution. The vanilla GAN is composed of a generator and a discriminator. A lot of publications [47, 30, 46] have shown the realistic image synthesis ability of the generator. The discriminator's ability in representation learning is also explored in [52, 13, 16]. GAN can help improve image retrieval task in these two aspects.

In [18], the authors proposed a two-stage pipeline to learn deep hashing models. The first stage was to generate fake images. In the second stage a deep hashing network is trained on both real images and generated fake images. [51] proposed a generative adversarial network with three streams on the discriminator's output: an adversarial stream, a classification stream, and a hashing stream. A joint loss was used to train such a network. [19] proposed an unsupervised deep hashing method based on GAN. In this method, the input of the generator is a random binary vector plus a random noise vector. The discriminator tries not only to distinguish between real and fake images, but also to recover the binary vector. [11] proposed a network similar to BiGAN which is composed of a generator, an encoder and a discriminator. An extra semantic constraint and an extra binary constraint are put on the encoder during the training process. The encoder is used to generate semantic-preserving hash codes in the end.

In the following chapters, I will introduce my novel methods on how to apply GAN to supervised hashing and unsupervised hashing for remote sensing image retrieval.

4 Supervised Hashing Boosted by GANs-Generated Images

GAN is initially proposed as a generative model that can learn the data distribution and generate data from noise vectors. Nowadays, remote sensing image dataset is so big that we can not give annotation to every image. Deep learning has proven to be effective at feature extraction and accurate at object detection and classification. Deep learning can also benefit hash learning methods for image retrieval. To overcome the problem of a limited number of labeled data, we proposed a method using GAN to augment the data by generating a large number of labeled fake images. And the generated fake images will be used to train a deep hashing neural network together with real images.

In this part of the thesis work, I mainly made the following contributions:

- I implemented two GAN architectures for conditional image synthesis: Conditional DCGAN and ACGAN. And thorough experiments were conducted to generate realistic fake remote sensing images.
- I implemented Inception Score and Fréchet Inception Distance to evaluate remote sensing images generated from Conditional DCGAN and ACGAN.
- I used the transfer learning method to extract features from ImageNet-pretrained Inception v3. The hashing neural network is trained to map the high-dimensional features to low-dimensional hash codes. The training set is a combination of real images and generated images.

4.1 Methodology

As shown in Figure 4.1, the proposed method has a two-stage pipeline. The first stage is RSGAN (Remote Sensing image GAN) for fake remote sensing generation. The second stage is deep hashing for remote sensing image retrieval. Two GAN architectures are used and compared in RSGAN. Deep Hashing uses Inception Net pretrained on ImageNet to extract features and then uses a deep hashing neural network to construct binary hash codes.

I will first introduce RSGAN by showing their architectures and training methods. Then I will introduce the deep hashing part and how deep hashing is used for image retrieval.

4 Supervised Hashing Boosted by GANs-Generated Images

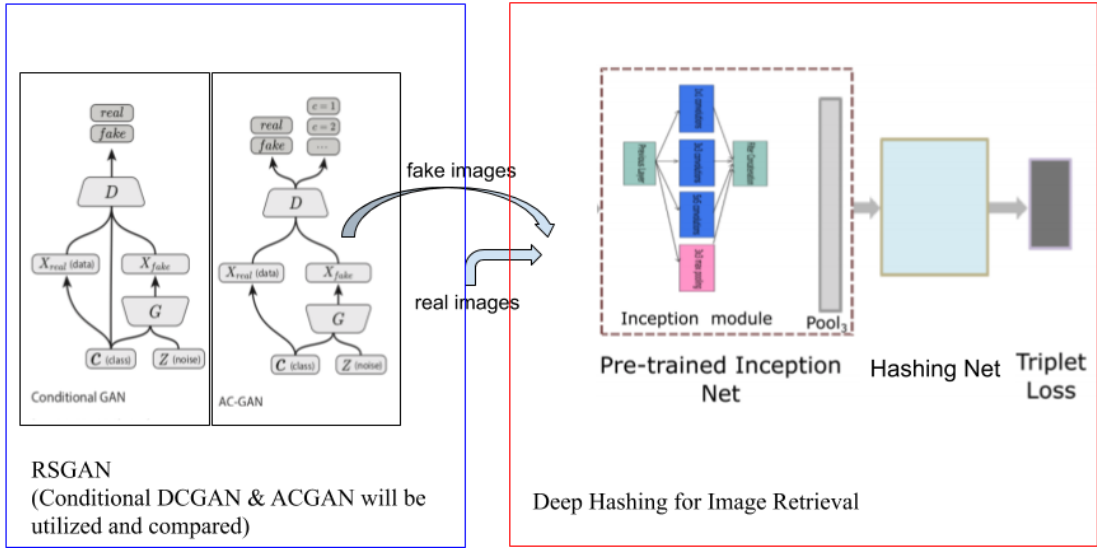


Figure 4.1: The framework of the proposed method: supervised hashing boosted by GANs-generated images.

4.1.1 GANs for Remote Sensing Image Generation

Conditional GAN can generate class-specific images when the condition information is given. The motivation here is to use conditional GAN to generate more labeled images, which will solve the problem of a limited number of labeled data. Conditional DCGAN and ACGAN are two important methods for conditional image synthesis and are adopted in this thesis to generate labeled remote sensing images.

Conditional DCGAN

Conditional DCGAN is short for Conditional Deep Convolutional Generative Adversarial Network. It is composed of a generator and a discriminator. The generator takes random noise vectors plus the condition information, which is labels in our case, as the input. It tries to generate images that can fool the discriminator. The discriminator takes images and labels as input. It tries to distinguish between real images and fake images generated from the generator.

Conditional DCGAN follows all guidelines proposed in [52] to make the training process more stable. All pooling layers are replaced with convolutional layers (in the discriminator) and transposed convolutional layers (in the generator). Batch normalization is used to reduce internal covariant shift between layers. All fully connected hidden layers are removed. The ReLU activation function is used in all layers of the generator except for the output which uses Tanh. The Leaky ReLU activation function is used in all layers of the discriminator.

Suppose that we have the prior input noise distribution $p_z(\mathbf{z})$ and the condition information y . The generator builds a mapping from the noise distribution $p_z(\mathbf{z})$ to data space $G(\mathbf{z}|\mathbf{y})$. The discriminator $D(\mathbf{x}|\mathbf{y})$ or $D(G(\mathbf{z}|\mathbf{y})|\mathbf{y})$ outputs a probability that the input is from real data. The objective function of this two-player minimax game is:

$$\min_G \max_D V(D, G) = \mathbf{E}_{\mathbf{x} \sim p_{data}(\mathbf{x})} [\log D(\mathbf{x}|\mathbf{y})] + \mathbf{E}_{\mathbf{z} \sim p_z(\mathbf{z})} [\log(1 - D(G(\mathbf{z}|\mathbf{y})|\mathbf{y}))] \quad (4.1)$$

A lot of experiments show that using an inverse objective function brings better optimization for the generator. The improved loss functions used in the training process of Conditional DCGAN are as follows:

$$L_D = \mathbf{E}_{\mathbf{x} \sim p_{data}(\mathbf{x})} [\log D(\mathbf{x}|\mathbf{y})] + \mathbf{E}_{\mathbf{z} \sim p_z(\mathbf{z})} [\log(1 - D(G(\mathbf{z}|\mathbf{y})|\mathbf{y}))] \quad (4.2)$$

$$L_G = \mathbf{E}_{\mathbf{z} \sim p_z(\mathbf{z})} [\log D(G(\mathbf{z}|\mathbf{y})|\mathbf{y})] \quad (4.3)$$

The discriminator and the generator are trained to maximize L_D and L_G simultaneously in each iteration. The 'simultaneously' here means that they are both optimized one after the other in a specific iteration rather than one of them is trained for many iterations to achieve the optimal state and then the other is trained.

ACGAN

ACGAN is short for Auxiliary Classifier Generative Adversarial Network. It is also composed of a generator and a discriminator. The generator also takes random noise vectors plus condition information as the input. It tries to generate realistic fake images that can fool the discriminator. The discriminator takes only images as the input which is different from Conditional DCGAN whose discriminator's input is images plus labels. The discriminator in ACGAN has two goals: one is to distinguish between real images and generated images, the other is to predict the labels of the input image.

In ACGAN, we adopt almost the same architecture as Conditional DCGAN. All the tricks that make GAN training stable are also used here. The input and the output of the discriminator in ACGAN need to be adapted due to the added auxiliary classifier. Figure 2.17 illustrates this difference clearly.

Suppose that we have the prior input noise distribution $p_z(\mathbf{z})$ and the condition information y . The generator builds a mapping from the noise distribution $p_z(\mathbf{z})$ to data space $G(\mathbf{z}|\mathbf{y})$. The discriminator $D(\mathbf{x})$ or $D(G(\mathbf{z}|\mathbf{y}))$ outputs a probability that the input is from real data and the probability that the input belongs to different classes. They can be formulated in the following way:

$$D(\mathbf{x}) = (D(\mathbf{x})_{adversarial}, D(\mathbf{x})_{classifier}) \quad (4.4)$$

4 Supervised Hashing Boosted by GANs-Generated Images

$$D(G(\mathbf{z}|\mathbf{y})) = (D(G(\mathbf{z}|\mathbf{y}))_{adversarial}, D(G(\mathbf{z}|\mathbf{y}))_{classifier}) \quad (4.5)$$

where (\cdot) denotes concatenation. Therefore the loss functions are as follows:

$$L_{DA} = \mathbf{E}_{\mathbf{x} \sim p_{data}(\mathbf{x})} [\log D(\mathbf{x})_{adversarial}] + \mathbf{E}_{\mathbf{z} \sim p_{\mathbf{z}}(\mathbf{z})} [\log(1 - D(G(\mathbf{z}|\mathbf{y}))_{adversarial})] \quad (4.6)$$

$$L_{GA} = \mathbf{E}_{\mathbf{z} \sim p_{\mathbf{z}}(\mathbf{z})} [\log D(G(\mathbf{z}|\mathbf{y}))_{adversarial}] \quad (4.7)$$

$$L_C = \mathbf{E}_{\mathbf{x} \sim p_{data}(\mathbf{x})} \left[\sum_{c=1}^M y_c \log D(\mathbf{x})_{classifier,c} \right] + \mathbf{E}_{\mathbf{z} \sim p_{\mathbf{z}}(\mathbf{z})} \left[\sum_{c=1}^M y_c \log D(G(\mathbf{z}|\mathbf{y}))_{classifier,c} \right] \quad (4.8)$$

where L_{DA} and L_{GA} are standard adversarial losses, and L_C is the categorical cross entropy loss for the classifier. y_c which is 1 or 0 denotes whether or not the image belongs to class c . And $D(\cdot)_{classifier,c}$ denotes the category prediction on class c . D is trained to maximize $L_{DA} + L_C$ while G is trained to maximize $L_{GA} + L_C$.

Training Methods

Label smoothing, mini-batch training and Adam optimizer are used in the training process.

Label smoothing is an effective strategy that can reduce the vulnerability of neural networks to adversarial examples [56]. It replaces the 0 and 1 targets with smoothed values 0.1 and 0.9. This strategy is only applied to the training of the discriminator.

Mini batch gradient decent is a trade-off between robustness to the noise and computation efficiency. It splits all the data into a lot of small batches. In each iteration, a small batch of data is used to train the neural network.

Adam optimizer is used to optimize the loss functions in each training step. Adam is an algorithm for first-order gradient-based optimization of stochastic objective functions, based on adaptive estimates of lower-order moments [32]. It combines the advantages of AdaGrad [14] and RMSProp [68].

4.1.2 Deep Hashing for Remote Sensing Image Retrieval

After the training of GAN for remote sensing image generation is finished, GAN will be used to perform conditional remote sensing image synthesis. On the one hand, the labeled remote sensing image dataset can be enlarged. On the other hand, typical features for each land-use class will be explicitly presented in the generated images. Then generated images together with real images, both of which are labeled, will be fed into the deep hashing network for hash learning.

Network Architecture

Transfer learning is used to extract semantic features of remote sensing images. In this thesis, Inception v3, which is pretrained on ImageNet, is used to extract features. We design a three-layer fully connected neural network as deep hashing network. It can map the high-dimensional semantic features to low-dimensional compact binary hash codes.

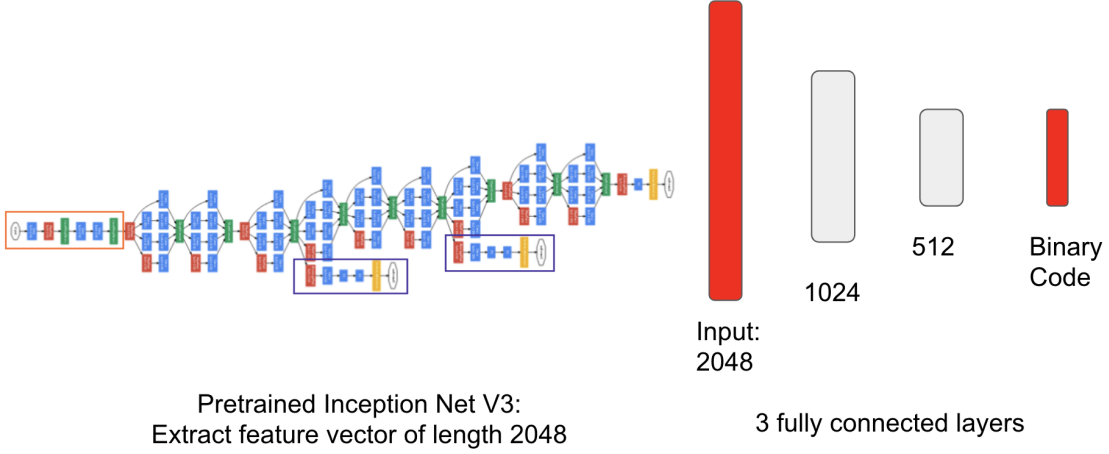


Figure 4.2: The architecture of deep hashing network.

Loss Function

Three constraints are utilized to train our deep hashing network. Triplet loss is used to ensure the short distance of similar images and the large distance of dissimilar images in hash space. Binary loss is used to push each entry of hash codes close to 0 and 1. Balance loss is used to ensure that each entry of hash codes have equal chances to be 0 or 1. This can avoid only part of the hash codes being really used in the optimization.

Triplet loss is formulated in Equation 4.9:

$$L_{triplet} = \sum_i^N [||f(x_i^a) - f(x_i^p)||_2^2 - ||f(x_i^a) - f(x_i^n)||_2^2 + \alpha]_+ \quad (4.9)$$

where $f(\cdot)$ denotes the trained model. x_i^a denotes the anchor sample, x_i^p denotes the positive sample and x_i^n denotes the negative sample. α is the margin between positive and negative pairs. N denotes N group of triplets.

Binary loss is formulated in Equation 4.10:

4 Supervised Hashing Boosted by GANs-Generated Images

$$L_{binary} = -\frac{1}{M} \sum_{i=1}^M \|f(x_i) - 0.5\mathbf{1}\|^2 \quad (4.10)$$

where M denotes the number of images in a training batch. x_i denotes the i -th image. $f(x_i)$ denotes the output of the model which is the corresponding hash codes of the i -th image. $\mathbf{1}$ denotes a vector with the same length as hash codes and all entries being 1.

Balance loss is formulated in Equation 4.11:

$$L_{balance} = \frac{1}{M} \sum_{i=1}^M \left(\frac{1}{L} \sum_{j=1}^L f(x_i)_j - 0.5 \right)^2 \quad (4.11)$$

where L denotes the length of the hash codes. $f(x_i)_j$ denotes the j -th entry of the hash codes of the i -th image.

The joint loss is formulated in Equation 4.12:

$$L = L_{triplet} + \beta L_{binary} + \gamma L_{balance} \quad (4.12)$$

where β and γ denote the weights of different loss functions. These two hyperparameters can be obtained by cross validation.

Image Retrieval

After the training of the proposed deep hashing network, the hash codes of the images in the archive are computed in advance. Each time a query image is given, it will firstly be fed into Inception v3 to extract features and then the final hash codes are obtained by feeding the extracted features to deep hashing network.

The Hamming distance is taken as the similarity measure between the hash codes of different images. It is the number of bits at which two hash codes are different. The smaller distance corresponds to the higher similarity between the original images.

The retrieval result is a ranking list in which the more similar images are in the higher positions of the ranking list.

4.2 Design of Experiments

4.2.1 Dataset

EuroSAT [24] is a recently published remote sensing image dataset. This dataset gathers multispectral images acquired by Sentinel-2A satellite over cities in 34 European countries. The images have a total of 13 bands with the spatial resolution from 10m to 60m per pixel. The images are the cropped small patches with the size of 64×64 . They

4.2 Design of Experiments



Figure 4.3: Visualization of samples from different classes in EuroSAT. Only RGB bands are shown here.

cover 10 land-use classes: Annual Crop, Forest, Herbaceous Vegetation, Highway, Industrial, Pasture, Permanent Crop, Residential, River and Sea Lake. Each class contains 2000-3000 images. In total the dataset has 27000 images.

Two different versions of this dataset are provided by the authors: RGB version and multispectral version. In this chapter, *Supervised Hashing Boosted by GANs-generated Images*, only RGB bands of the remote sensing images are considered. So, the RGB version of this dataset is used in the experiments.

I randomly sampled 100 images from each class to build the *testing set* which has 1000 images in total. Each image itself is a query image whose searching archive is the remaining 999 images.

The remaining 26,000 images are taken as the *training set*. The training set is used to train Conditional DCGAN and ACGAN. For the training set of the deep hashing network, there are three cases:

4 Supervised Hashing Boosted by GANs-Generated Images

- The remaining 26000 images.
- The remaining 26000 images and 5000 images generated by Conditional DCGAN.
- The remaining 26000 images and 5000 images generated by ACGAN.

These three cases are compared to show the effect of adding GANs-generated images to deep hash learning.

4.2.2 Implementation Details

GAN

The detailed network architectures of conditional DCGAN and ACGAN are shown in Figure 4.4. The input of the generator is composed of a random noise vector and the condition information. The noise vector is sampled from a uniform distribution in the range $[-1, 1]^{100}$. The condition information is a one-hot vector with the shape (10,) which represents the class label. For the discriminator in Conditional DCGAN, one-hot condition vector is concatenated to the channel dimension of each pixel of the image. For the discriminator in ACGAN, the output is a real or fake prediction and a class prediction. The α in Leaky ReLU is 0.2.

Input Z: (100,1)+(10,1)		Input: C-DCGAN: (64,64,3+10) ACGAN: (64,64,3)	
Linear & Reshape	(4, 4, 512)	conv2d, filter 5, stride 2, LReLU	(32, 32, 16)
deconv2d, filter 5, stride 2, ReLU	(8, 8, 256)	conv2d, filter 5, stride 1, LReLU	(32, 32, 32)
deconv2d, filter 5, stride 2, ReLU	(16, 16, 128)	conv2d, filter 5, stride 2, LReLU	(16, 16, 64)
deconv2d, filter 5, stride 2, ReLU	(32, 32, 64)	conv2d, filter 5, stride 1, LReLU	(16, 16, 128)
deconv2d, filter 5, stride 2, Tanh	Output: (64, 64, 3)	conv2d, filter 5, stride 2, LReLU	(8, 8, 256)
		Flatten	(8*8*256,)
		fully_connect, LReLU	(1024,)
		Fully_connect, sigmoid	Output: C-DCGAN: (1,) ACGAN: (10+1,)
Generator		Discriminator	

Figure 4.4: The detailed architecture of the generator and the discriminator network in Conditional DCGAN and ACGAN.

We set batch size as 64, learning rate as 0.0002, β_1 and β_2 in Adam optimizer as 0.5 and 0.999.

Deep Hashing

For deep hashing network, we set the batch size as 96. In a mini batch, images are from two different land use classes. 64 images are from the first class to build the anchor data and the positive data. 32 images are from the second class to build the negative data. The learning rate is set as 0.0001. The β_1 and β_2 in Adam optimizer are set as 0.5 and 0.9 respectively.

4.2.3 Evaluation Metrics

GAN/Fake Image Quality

We evaluate Conditional DCGAN and ACGAN by evaluating the quality of fake images generated from them. Two commonly used metrics for fake image quality evaluation are Inception Score (IS) and Fréchet Inception Distance (FID). They can be calculated by Equation 4.13 and Equation 4.14. They are both classifier-based evaluation metrics, which means they feed fake images (plus real images if using FID) into a classifier and compute the metric values based on the classifier’s outputs. The classifiers they use are both Inception v2 pretrained on ImageNet. For a detailed explanation, please refer to Section 2.2.3.

$$IS(G) = \exp(E_{x \sim p_g}[D_{KL}(p(y|\mathbf{x})||p(y))]) \quad (4.13)$$

$$FID(G) = \|\mathbf{m}_r - \mathbf{m}_g\|^2 + Tr(\mathbf{C}_r + \mathbf{C}_g - 2(\mathbf{C}_r \mathbf{C}_g)^{1/2}) \quad (4.14)$$

For both Conditional DCGAN and ACGAN, 5000 generated images are used to compute the IS and FID.

Image Retrieval

Mean average precision (MAP) is a widely used criterion to evaluate the performance of the image retrieval task. MAP can be computed from Equation 5.18.

$$MAP = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{n_i} \sum_{j=1}^{N_i} precision(R_i^j) \delta(j) \quad (4.15)$$

In this equation, Q denotes the set of all query images. n_i is the number of images similar to $q_i \in Q$. N_i is the size of the ranking list. R_i^j is the subset of ranked results, which only includes the result from the 1-st to the j -th position in the ranking list.

4 Supervised Hashing Boosted by GANs-Generated Images

$\delta(j) = 0$ when the j -th retrieved image is not relevant to the query image. $\delta(j) = 1$ when the j -th retrieved image is relevant to the query image .

4.3 Experimental Results

The results will be analyzed in two aspects: image generation and image retrieval.

4.3.1 Analysis of Image Generation

We first visually compare the images generated from Conditional DCGAN and ACGAN after they are trained for 100 000 steps. The results are shown in Figure 4.5. We can see that they are both able to generate realistic images of a specific class when the condition information is given. Visually, it's hard to say which GAN definitely performs better. Images generated from ACGAN look less blurry and have more details like textures or shapes.

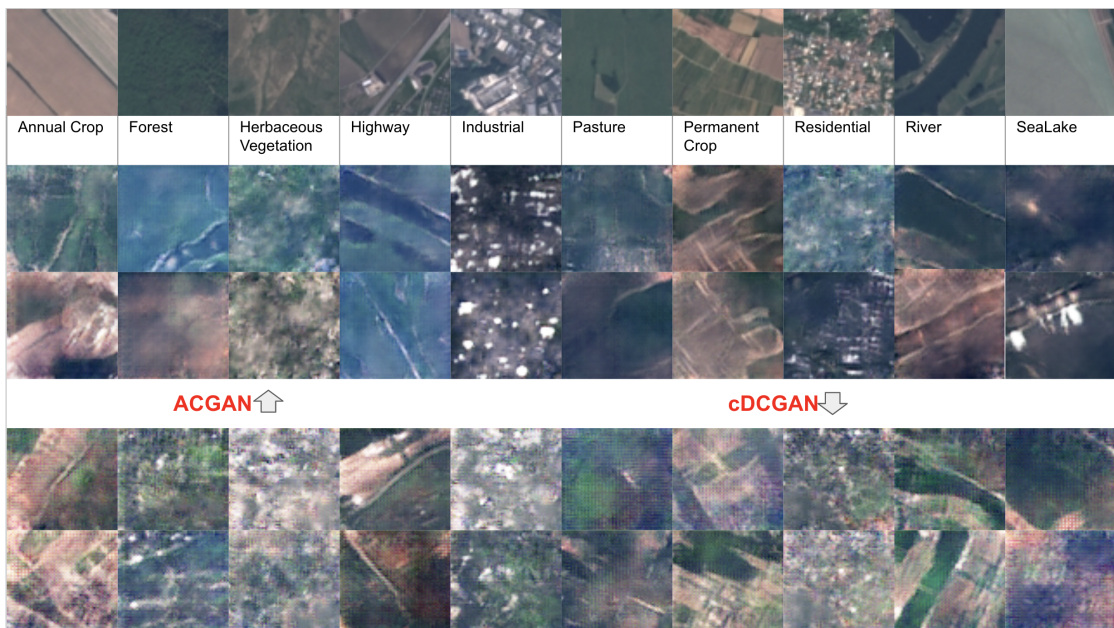


Figure 4.5: Visual comparison of fake images generated from Conditional DCGAN and ACGAN.

Then we visualize the training process of Conditional DCGAN and ACGAN as illustrated in Figure 4.6. These images are generated by the corresponding model from the beginning to after 50 000 training steps. We can see that Conditional GAN is faster to

find the correct optimization direction and begin generating meaningful images. ACGAN is not stable in the beginning and only generate noise but can start to generate good images after many training steps. And the final generation results have better visual effects.

To fairly compare these two GAN architectures, we also give quantitative evaluations using two commonly used evaluation metrics: Inception Score (IS) and Fréchet Inception Distance (FID). For IS, the higher value is better. For FID, the lower value is better. We can see that ACGAN beat Conditional DCGAN in both criteria. This difference comes from how the label information is used in both GAN. The label information in Conditional DCGAN serves as the input of the discriminator. The label information in ACGAN is used in the classifier loss function. This makes fake images more discriminative and realistic.

Table 4.1: Quantitative comparison of fake images generated from Conditional DCGAN and ACGAN.

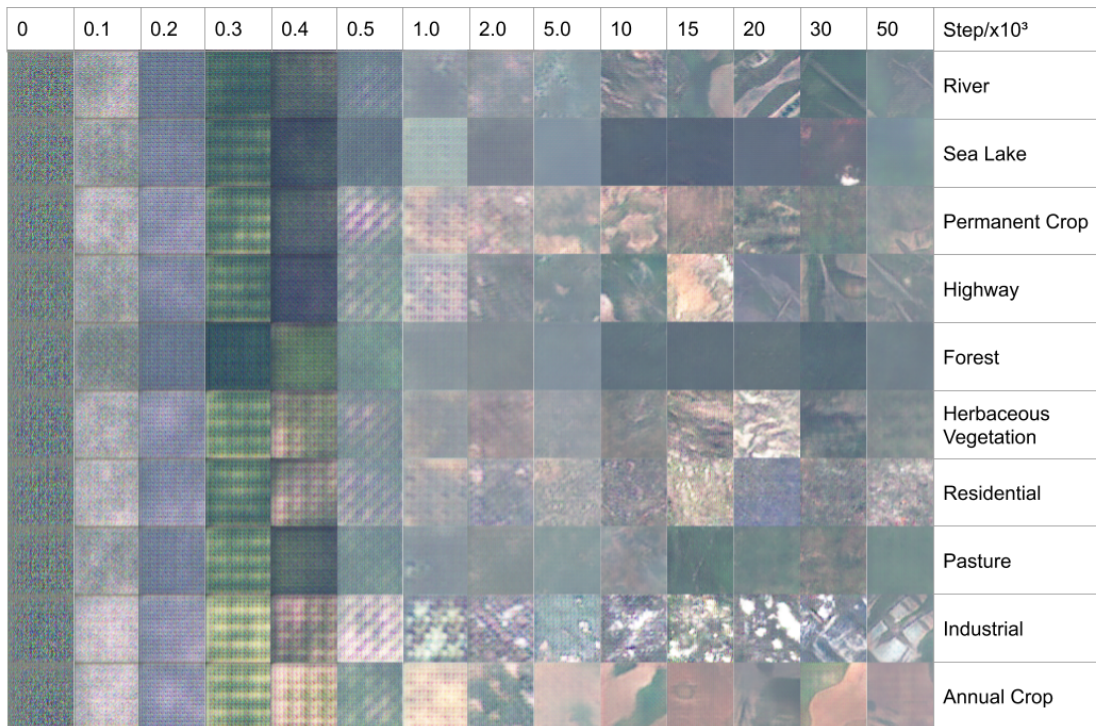
	IS	FID
Conditional DCGAN	2.3521	188.9619
ACGAN	2.6371	170.8508

It's worth noting that IS and FID posted here are much worse than those from the state-of-the-art methods. [59] listed the IS and FID results of several different generative methods. But it's normal because IS and FID are classifier-based evaluation metrics. That means they feed generated fake images into a classifier and calculate the metric value based on the output of the classifier. And the classifier is Inception v2 pretrained on ImageNet. This classifier is aimed for natural image visual recognition. Natural images are quite different from remote sensing images. And the final output of the classifier is the class of natural images which even may not include classes of remote sensing images like forest and pasture.

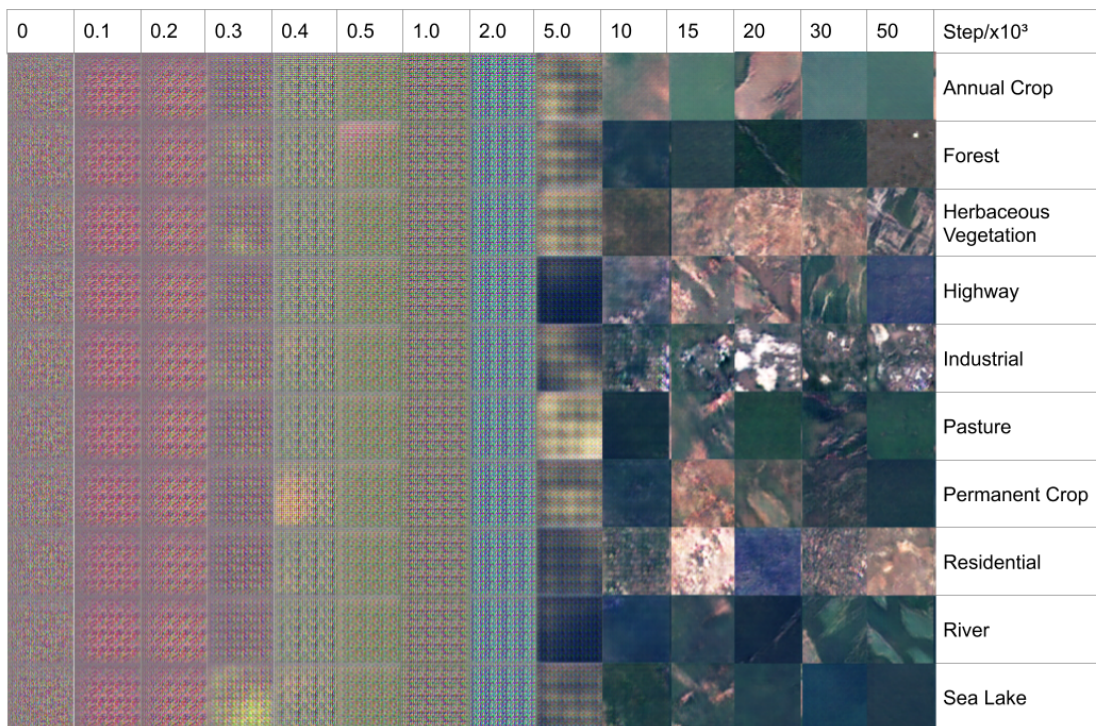
Table 4.2: CIFAR10 experiments using different generative model. IS: higher is better. FID: lower is better. [59]

model	IS	FID-5K	FID
real images	11.33	9.4	2.1
SNGAN	8.43	18.8	11.8
WGAN-GP (10M)	8.21	21.5	14.1
WGAN-GP (2.5M)	8.29	22.1	15.0
DCGAN	6.69	42.5	35.6
PixelCNN++	5.36	121.3	119.5

4 Supervised Hashing Boosted by GANs-Generated Images



(a) Conditional DCGAN



(b) ACGAN

Figure 4.6: The learning process of Conditional DCGAN and ACGAN on EuroSAT remote sensing image dataset.

4.3.2 Results of Image Retrieval

For image retrieval, MAP is used to evaluate the results. Table 4.3 shows the MAP of top 100 results using the model trained on different training sets. When the hash code length is 32 bits or 64 bits, GAN images can help improve the overall retrieval result. Among all the experiments, the best result of 0.9404 is obtained when the model is trained on real images plus ACGAN images.

Table 4.3: MAP@100 of image retrieval on testing set using different trained model.

Training Set	16bits	32bits	64bits
Only real images	0.9338	0.9357	0.9334
Real + C-DCGAN images	0.9189	0.9387	0.9372
Real + ACGAN images	0.9217	0.9404	0.9313

To have a closer look at how GAN images influence the retrieval, we also compare the class-wise MAP results using the model trained on the different training sets. Table 4.4 and Table 4.5 show class-wise results and class-wise MAP changes using the model trained on the different training sets with hash code length equaling 32. From Table 4.4 we can see that the lowest two MAP values exist in Highway and River these two categories when the model is trained only on real images. From Table 4.5 we can see that MAP values of these two categories are obviously improved when GAN images are added to the training set. In other categories, MAP values get a relatively small increase or decrease. The increase in these two categories leads to an increase on the whole MAP.

Table 4.4: Class-wise MAP@100 of image retrieval on testing set using different trained model.

	Ann.	For.	Her.	Higa.	Ind.	Pas.	Per.	Res.	Riv.	Sea.	ALL
Real	0.9844	0.9678	0.9033	0.8110	0.9984	0.9665	0.8893	0.9991	0.8566	0.9801	0.9357
Real+C.	0.9519	0.9690	0.9123	0.8542	0.9676	0.9427	0.8854	0.9719	0.9415	0.0.9705	0.9387
Real+A.	0.9591	0.9674	0.9015	0.8378	0.9895	0.9636	0.8995	0.9805	0.9140	0.9908	0.9404

Table 4.5: The change of class-wise MAP@100 of image retrieval on testing set using different trained model.

	Ann.	For.	Her.	Higa.	Ind.	Pas.	Per.	Res.	Riv.	Sea.	ALL
Real	-	-	-	-	-	-	-	-	-	-	-
Real+C.	-3.29%	+0.12%	+0.99%	+5.32%	-1.09%	-2.46%	-0.44%	-2.72%	+9.92%	-0.98%	+0.33%
Real+A.	-2.56%	-0.04%	-0.20%	+3.29%	-0.89%	-0.30%	+1.15%	-1.86%	+6.71%	+1.09%	+0.51%

We also provide the visualization of image retrieval on the river and the highway class in Figure 4.7. We can see that the river and the highway images are easily confused by

4 Supervised Hashing Boosted by GANs-Generated Images

the model. No matter if the query image is from the river or highway class, the top-6 retrieved images are a mixture of river and highway images. On the other hand, we can see that GAN-generated images really help the model achieve better retrieval results on these two classes.

The explanation for this is that the GAN generates images by focusing on several characteristics of that class. Most classes in EuroSAT have uniform colors or uniform textures in the images. The river and the highway classes are more complicated. A river or highway can be surrounded by different land use classes, like the agricultural land, the residential area or the industrial area. When the GAN generates river or highway images, the surrounding land use class may be neglected and only the parallel lines' shape is emphasized. So, when the deep hashing model is trained on these shape emphasized fake images, it can learn better features and therefore generating more representative hash codes for image retrieval.

In the end, we show the retrieval results of the best model on all classes in Figure 4.8.

4.4 Conclusion

In this chapter, I address the problem of a limited number of labeled images with GAN-based data augmentation. Two GAN architectures are used for conditional image synthesis. ACGAN proves to be better than Conditional DCGAN on remote sensing image generation. The generated images are very realistic with a lot of similar characteristics to real images.

Because conditional GAN is used here, generated images can be regarded as labeled images. The training set can be enlarged by combining real images and generated images. During the experiments, we found that GAN focuses more on generating distinct characteristics of images for each class. Thus, these fake images can help deep hash learning neglect unimportant parts in the images. From all the experimental results and all the analysis, we can conclude that images generated by GAN can improve deep hash learning and improve image retrieval performance. The improvement mainly comes from the improvement on hard images (images with more complicated contents, like river and highway remote sensing images in our case).

Results from training with real images



Results from training with real + C-DCGAN images



Results from training with real + ACGAN images



(a) Visualization of the retrieval result on a random selected River image.

Results from training with real images



Results from training with real + C-DCGAN images



Results from training with real + ACGAN images



(b) Visualization of the retrieval result on a random selected Highway image.

Figure 4.7: Visualization of the improvement of image retrieval performance on river and highway class after using GAN-generated images.

4 Supervised Hashing Boosted by GANs-Generated Images

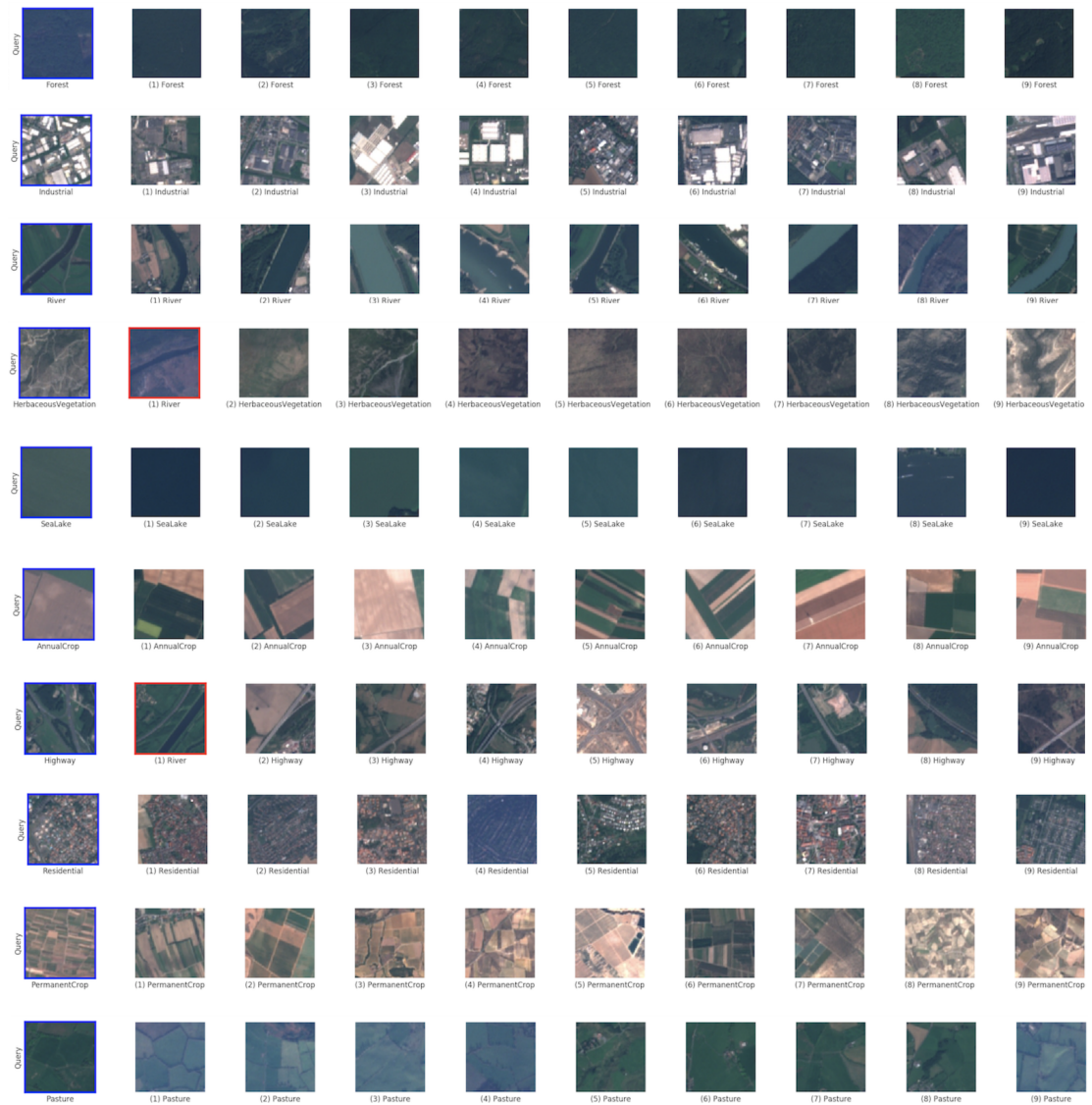


Figure 4.8: Visualization of the image retrieval result. From each class, one image is random selected as the query image. The best model is chosen here: training set includes real images and ACGAN images; Hash code length is 32 bits.

5 Unsupervised Adversarial Hashing for Multispectral Images

With the advances in imaging technology and internet technology, more and more images are produced and stored. Image search and indexing are important in data management to efficiently make use of the information stored in these images. Hash-based approximate nearest neighbor search attracts a lot of attention. It can not only reduce the storage cost but also speed up the retrieval process.

Traditional hashing methods use hand-crafted features and learn a mapping from the high dimensional space to the binary hash space but also maintains the similarity. Deep learning is changing the whole computer vision community with its incredible power of bridging the gap between low-level features and high-level semantic concepts. Deep learning also brings learn-to-hash research to a new stage.

Supervised hashing uses pairwise similarity loss or triplet loss to guide the hash function learning process. But it requires label information of the data. Manual labeling is time consuming and impractical when the dataset is very large. Unsupervised hashing methods are proposed to solve these problems. Some unsupervised hashing methods use the deep autoencoder to learn hash codes [34, 55]. These methods build an encoder neural network as the hash function. And the decoder neural network tries to reconstruct the original images from encoder's output, the hash codes. Other methods learn hash codes by maximizing their representation capacity [17] or enforcing rotation similarity [43].

Remote sensing images have far more bands than natural images. If we apply hashing methods designed for natural images directly to remote sensing images, all other bands except RGB bands are neglected. Most information in remote sensing images is lost, which will have a side effect on the retrieval performance.

Different from Chapter 4 where GAN is used only for image generation, we explore how to directly use GAN for hash learning in this chapter. Research in [52, 13, 16] already shows that GAN is useful for representation learning. Representation learning is closely related to semantic hash learning. In this chapter a novel method called *Unsupervised Adversarial Hashing for Multispectral Images* is proposed to address the problems of unavailability of a large amount of labeled data and lost information of neglected bands in multispectral images.

In this part of the thesis work, I mainly made the following contributions:

5 *Unsupervised Adversarial Hashing for Multispectral Images*

- I implemented a GAN for multispectral remote sensing image generation. And I used the learned discriminator to extract low-dimensional representations for multispectral remote sensing image retrieval.
- I designed a novel GAN-based unsupervised semantic hashing method. This method adds a semantic constraint and a hash constraint to the vanilla GAN for multispectral remote sensing image hashing.
- I conducted thorough experiments to prove the effectiveness of three strategies: considering multiple bands of multispectral images, using semantic similarity information, using GAN for representation learning.
- I analyzed the quality of generated images and its influence on the retrieval performance.

5.1 Methodology

In this section, I will introduce my methods step by step. In the first part, I will introduce how to use the unsupervised representation learning ability of the GAN for RGB image retrieval. In the second part, I will introduce how to make use of multiple bands of multispectral images in GAN-based image retrieval. In the third part, I will introduce my novel method for unsupervised semantic hashing on multispectral remote sensing images. In the last part, I will introduce how to build the semantic similarity matrix to guide the learning process of unsupervised semantic hashing.

5.1.1 GAN for Image Retrieval

We use a slightly modified GAN as the starting point to show how GAN can be used for image retrieval.

A vanilla GAN is made up of a generator and a discriminator. The generator generates fake images from random noise vectors to fool the discriminator. The discriminator learns to distinguish between real images and images generated from the GAN. The generator in the vanilla GAN generates images in an unconditional way, which means images are generated only from noise vectors without the assistance of label information.

The discriminator is basically a convolutional neural network for binary source classification. The last several layers of the discriminator are fully connected layers. We take the output of the last but one layer as the image feature representation. As shown in Figure 5.1, in the training process, the discriminator can learn a useful representation before the final binary output, and in the retrieval process, the discriminator is used

as a feature extractor. The similarity measure for the extracted features is the cosine distance.

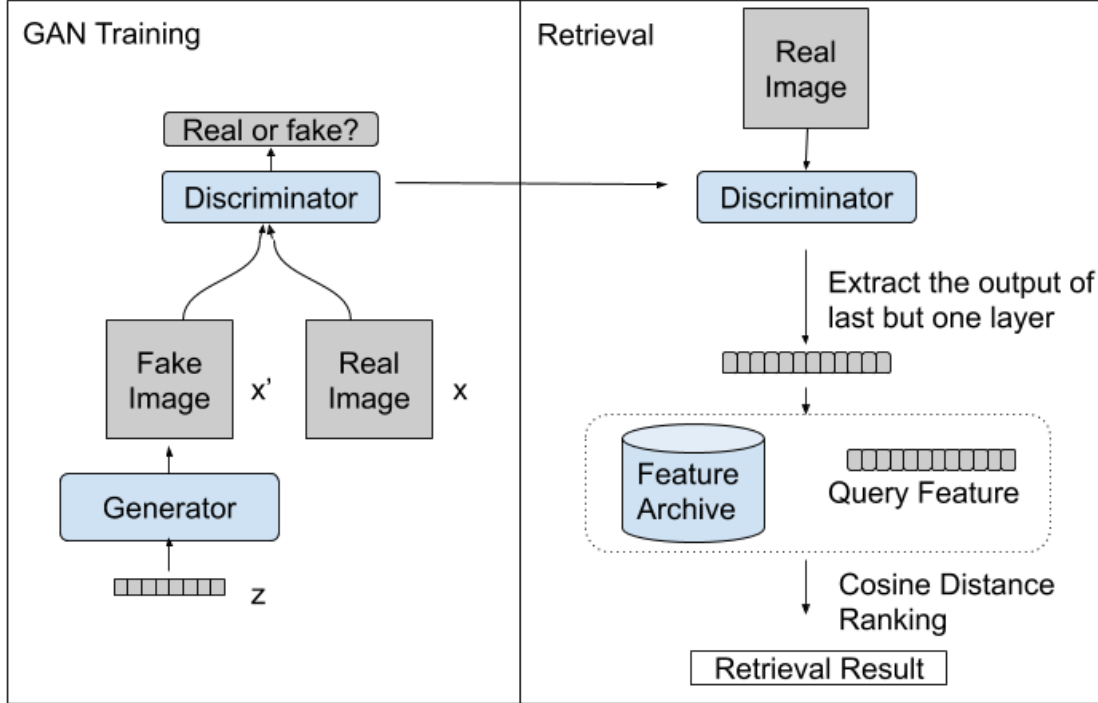


Figure 5.1: Illustration of image retrieval based on GAN.

The loss functions of this GAN are:

1. Training the discriminator

$$L_D = \mathbf{E}_{\mathbf{x} \sim p_{data}(\mathbf{x})} [\log D(\mathbf{x})] + \mathbf{E}_{\mathbf{z} \sim p_z(\mathbf{z})} [\log(1 - D(G(\mathbf{z})))] \quad (5.1)$$

2. Training the generator

$$L_G = \mathbf{E}_{\mathbf{z} \sim p_z(\mathbf{z})} [\log(D(G(\mathbf{z})))] \quad (5.2)$$

The generator and the discriminator are optimized to maximize these two loss functions.

5.1.2 GAN for Multispectral Image Retrieval

When taking multiple bands of remote sensing images into consideration, the GAN follows almost the same architecture as the GAN used for RGB images. The only difference is that the output layer of the generator and the input layer of the discriminator need to be adapted to multiple bands. The output of the last but one layer will be used as the feature representation for later image retrieval.

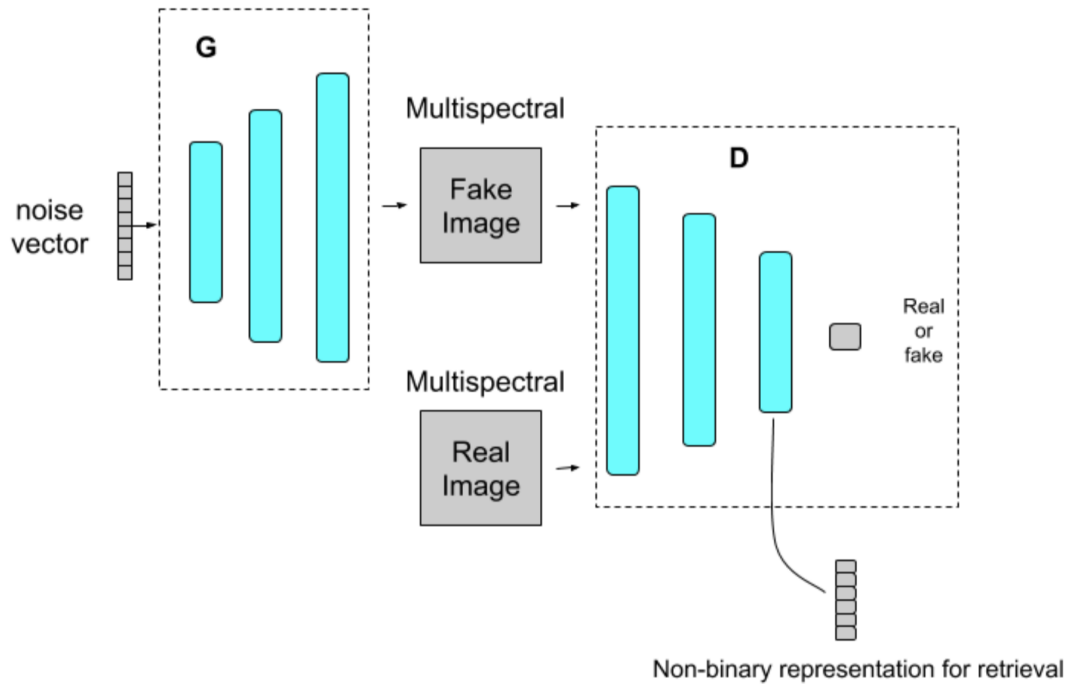


Figure 5.2: GAN architecture used for multispectral images.

5.1.3 GAN for Unsupervised Semantic Hashing

Network Architecture

As shown in Figure 5.2, the feature representation obtained in the last but one layer of the discriminator is non-binary. The exact values of this representation depend on which activation function is used in that layer. Usually, Leaky ReLU is used as the activation function of all the layers in the discriminator. As shown in Figure 5.3, output values can be very large in Leaky ReLU. The storage of such a feature representation takes much space. And the similarity computation is also slow, which will influence the retrieval speed.

To overcome this problem, we propose a novel method by redesigning the GAN architecture as shown in Figure 5.4. The redesigned network adds a new stream after the last but one layer of the discriminator. That means the last but one layer will have two data flow directions: one is to a binary source prediction through a fully connected layer, the other one is to a hash binary representation through another fully connected layer. The hash stream uses Tanh as the activation function for the output layer, because Tanh can restrict the output values between -1 and 1. And the binary constraint will further push the hash codes close to -1 and 1.

It's worth noting that we use $(-1, 1)^l$ to replace so-called binary codes $(0, 1)^l$ in the

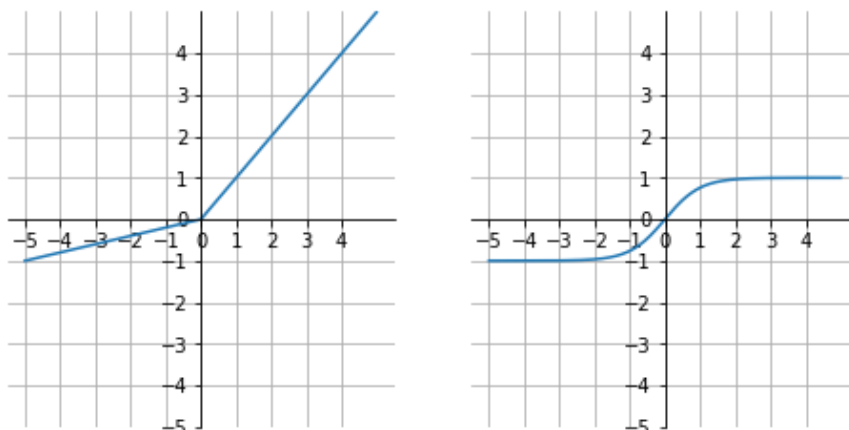


Figure 5.3: Left: Leaky ReLU activation function. Right: Tanh activation function

training process. It would never be a problem because $(-1, 1)^l$ and $(0, 1)^l$ can be easily transformed to each other.

To further improve the method, we add a semantic constraint on the output hash codes. The semantic constraint is used to keep the high-level semantic similarity in the hash space. The way to apply this constraint is optimizing a semantic loss based on semantic similarity matrix (SSM). More details of the semantic similarity matrix (SSM) are introduced in Section 5.1.4.

Training Method

The generator and the discriminator are trained simultaneously. They are both updated at every iteration step. Adam optimizer is used to perform gradient-based optimization. Three constraints form the objective functions for the generator and the discriminator.

Adversarial Loss: As mentioned before, the discriminator and the generator play a two-player minimax game. Adversarial loss makes the generator generate realistic images and the discriminator accurately distinguish between generated images and real images. The generator and the discriminator will finally reach a Nash equilibrium. Even though the discriminator is designed for source prediction, it can extract meaningful representations in the last several layers. The initially proposed loss functions for the generator and the discriminator have the same form of expression, but with completely opposite optimization direction. The improved version has different forms, but with the same optimization direction. The adversarial loss for the generator and the discriminator are shown in Equation 5.3 and Equation 5.4.

$$L_{G_A} = -\mathbf{E}_{\mathbf{z} \sim p_{\mathbf{z}}(\mathbf{z})}[\log(D(G(\mathbf{z})))] \quad (5.3)$$

5 Unsupervised Adversarial Hashing for Multispectral Images

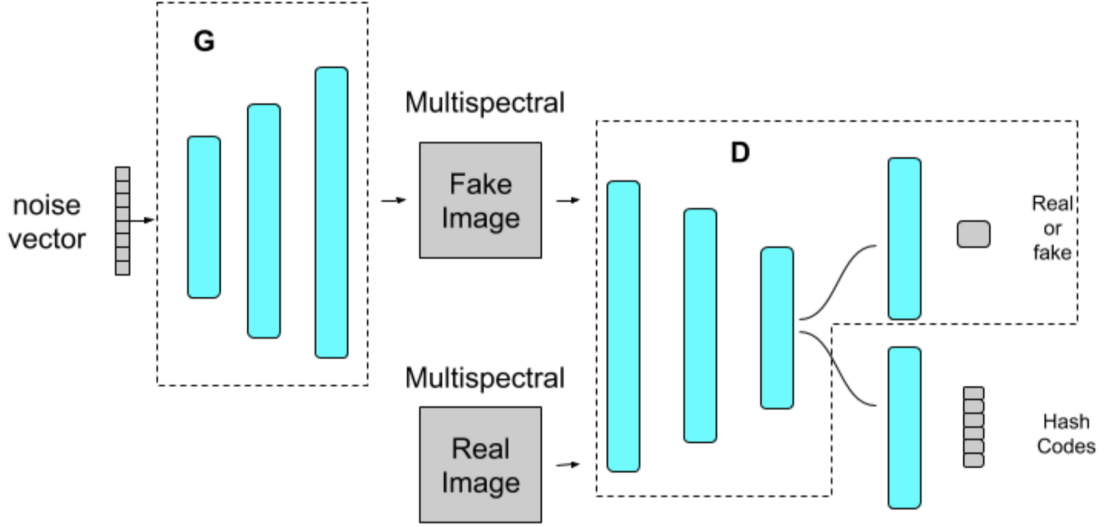


Figure 5.4: GAN architecture used for *Unsupervised Adversarial Hashing for Multispectral Images*.

$$L_{DA} = -\mathbf{E}_{\mathbf{x} \sim p_{data}(\mathbf{x})}[\log D(\mathbf{x})] - \mathbf{E}_{\mathbf{z} \sim p_{\mathbf{z}}(\mathbf{z})}[\log(1 - D(G(\mathbf{z})))] \quad (5.4)$$

where \mathbf{x} denotes the real image, \mathbf{z} denotes the random noise vector, $G(\mathbf{z})$ denotes the generated image from the random noise vector \mathbf{z} , and $D(\cdot)$ denotes the image source prediction, from the real data or the generated data.

Binary Loss: Even though Tanh activation function can restrict each entry of the hash codes in the range $(-1, 1)$, some entries may lie in the middle of this range, which can hinder the accuracy. Binary loss will push the codes close to -1 and 1. This loss is formulated in Equation 5.5.

$$L_b = -\mathbf{E}_{\mathbf{x} \sim p_{data}(\mathbf{x})} \|H(\mathbf{x})\|^2 \quad (5.5)$$

where \mathbf{x} denotes the real image and $H(\mathbf{x})$ denotes the hash codes of \mathbf{x} . $H(\cdot)$ is made up of part of the discriminator and a hash-specific fully connected layer.

Semantic Similarity Loss: From the unsupervised semantic similarity matrix (which will be introduced in Section 5.1.4), we can know the estimated similarity value between any two images from a high-level semantic perspective. We use semantic similarity loss to enforce the hash codes to preserve semantic information by keeping semantic similarity. The semantic similarity loss is formulated in Equation 5.6.

$$L_s = \mathbf{E}_{\mathbf{x}_i \sim p_{data}(\mathbf{x}), \mathbf{x}_j \sim p_{data}(\mathbf{x})} \left\| \frac{H(\mathbf{x}_i) * H(\mathbf{x}_j)}{l} - S_{i,j} \right\|^2 \quad (5.6)$$

where l denotes the hash code length, \mathbf{x}_i and \mathbf{x}_j denotes random two real images, $H(\mathbf{x}_i)$ and $H(\mathbf{x}_j)$ denotes the hash codes of \mathbf{x}_i and \mathbf{x}_j , and $S_{i,j}$ denotes the semantic similarity value of \mathbf{x}_i and \mathbf{x}_j .

Gathering all these three constraints, the joint loss functions for the generator and the discriminator are shown in Equation 5.7 and Equation 5.8. The binary constraint and the semantic constraint are only applied to real images when training the discriminator. In each training step, Adam optimizer minimizes L_D and L_G one after the other.

$$L_G = L_{G_A} \quad (5.7)$$

$$L_D = L_{D_A} + w_1 * L_b + w_2 * L_s \quad (5.8)$$

Ablation Study

Ablation study can help us know the importance of the adversarial part in our network architecture. We removed the adversarial structure from the proposed method. The remaining part is a deep hashing neural network (DHNN) which takes multispectral images in and outputs hash representations. The three loss functions mentioned in Section 5.1.3 will be reduced to two loss functions: the semantic loss and the binary loss.

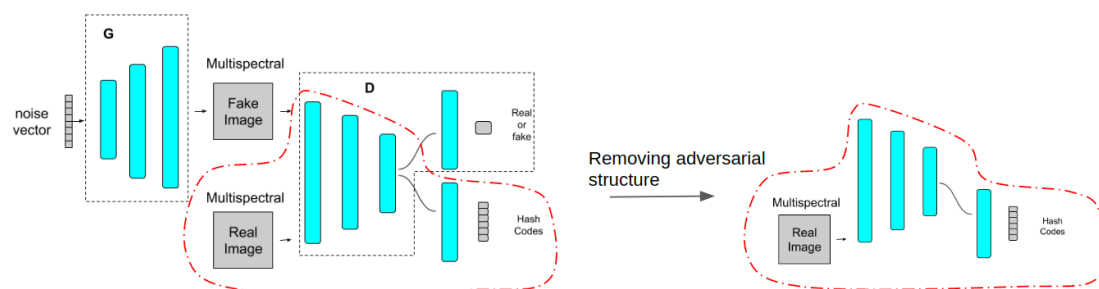


Figure 5.5: Illustration of the ablation study. The network on the right is named as DHNN MS.

In the results analysis section, we name the method with adversarial structure removed as DHNN MS.

5.1.4 Semantic Similarity Matrix

[11] proposed building an unsupervised semantic similarity matrix (SSM) to guide the hash learning. The motivation behind this is that a lot of very deep neural networks do pretty well in visual recognition on natural images. And these neural networks pre-trained on large-scale image dataset ImageNet are off-the-shelf. We can directly use

5 Unsupervised Adversarial Hashing for Multispectral Images

transfer learning to extract high-level semantic features from remote sensing images. These semantic features are of high dimension, so it's not suitable to do image retrieval directly based on these features. But they can guide the hash learning process and make the hash codes become more representative and preserve semantic information.

The first step of building the SSM is to extract features. [11] used the outputs of the last but one layer of ResNet v2-152 as semantic feature representations of images. I improved this by proposing a method combining semantic spatial features and non-semantic spectral features. Even though ResNet v2 ranks first in ILSVRC competition, it's not sure if it can extract good features from remote sensing images as well. I try another very deep neural network pretrained on ImageNet which is called Inception v3 to extract semantic features.

Non-semantic spectral features are also called color features. I listed three color feature extraction methods in Table 5.1. The raw color method directly flattens the image into a feature vector. The color histogram method uses the distribution of colors as the feature. In the global color histogram method, each channel will get its histogram by counting the number of pixels whose value falls in a specific spectral interval. In the local color histogram method, each channel will be divided into many small patches, and color histogram is calculated in these small patches. Color histograms from all patches from all channels are concatenated to build the feature vector. The local color histogram method can store not only spectral information but also some spatial information.

When fusing the semantic spatial feature and non-semantic spectral feature, a weight value is necessary to balance possible different scales of these two features as shown in Equation 5.9.

$$feature_{fuse} = (feature_{spectral}, \eta * feature_{spatial}) \quad (5.9)$$

where $(,)$ denotes the concatenating operation, η is the weight value to balance scale difference between spectral features and spatial features.

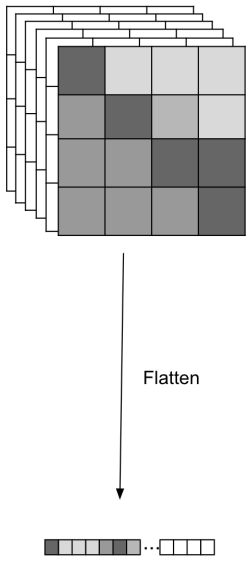
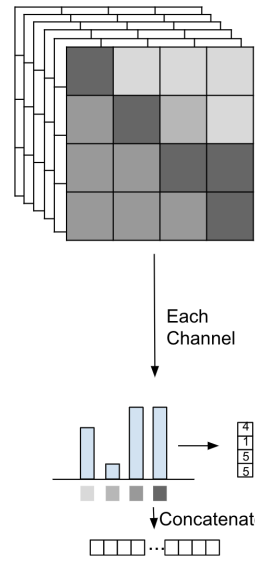

The second step of building SSM is calculating similarity values based on features obtained in the first step. For this step, we adopt the same method proposed in [11]. This method is based on k-nearest neighbor (kNN) search. Two-stage kNN searches are conducted to refine the similarity matrix building result.

The first kNN search uses the cosine distance as the distance metric. The first K_1 neighbors are regarded as similar images. The initial similarity matrix is built from Equation 5.10.

$$(S_1)_{i,j} = \begin{cases} 1, & \text{if } x_j \text{ is in } K_1\text{-NN of } x_i \\ -1, & \text{otherwise} \end{cases} \quad (5.10)$$

After the initial similarity matrix is built, the other kNN search is used to refine it. The idea behind the second kNN is that similar images have more same neighbors. The distance metric here is the number of the different neighbors. $(S_1)_i =$

Table 5.1: Different spectral features. h denotes the height of the image. w denotes the width of the image. c denotes the number of channels of the image. n_bins denotes the number of intervals which the whole spectral range is divided into. $n_patches$ denotes the number of patches which a images is divided into.

Method	Raw Color	Global Color Histogram	Local Color Histogram
			
Feature Size	$h * w * c$	$n_bins * c$	$n_patches * n_bins * c$

$((S_1)_{i,1}, (S_1)_{i,2}, \dots, (S_1)_{i,N})$ is the similarity vector which represents the similarity of x_i and $x_n|_{n=1}^N$. Then the number of different neighbors of x_i and x_j is also the Hamming distance between $(S_1)_i$ and $(S_1)_j$. The first K_2 neighbors are regarded as similar images. The second refined similarity matrix is built from Equation 5.11.

$$(S_2)_{i,j} = \begin{cases} 1, & \text{if } x_j \text{ is in } K_2\text{-NN of } x_i \\ -1, & \text{otherwise} \end{cases} \quad (5.11)$$

The final semantic similarity matrix is obtained by doing element-wise AND operation on S_1 and S_2 as shown in Equation 5.12. When two images are regarded as similar in both similarity matrices, they can be set as similar in the final similarity matrix.

$$S_{i,j} = \begin{cases} 1, & \text{if } (S_1)_{i,j} = 1 \text{ and } (S_2)_{i,j} = 1 \\ -1, & \text{otherwise} \end{cases} \quad (5.12)$$

5.2 Design of Experiments

5.2.1 Dataset

Multispectral Remote Sensing Image Dataset

As mentioned in Chapter 4, EuroSAT [24] is a recently published remote sensing image dataset whose images come from the Sentinel-2A satellite launched by the European Space Agency (ESA). Two different versions of this dataset are provided by the authors: the RGB version and the multispectral version. The RGB version is used in the experiments described in Chapter 4. However, in this chapter, the multispectral version is used to test the proposed novel image retrieval method.

The dataset covers 10 land use classes: Annual Crop, Forest, Herbaceous Vegetation, Highway, Industrial, Pasture, Permanent Crop, Residential, River and Sea Lake. Each class contains 2000-3000 images. In total, the dataset has 27 000 images.

The spectral bands of images in EuroSAT include visible, near-infrared (NIR), and short-wavelength infrared (SWIR) spectrum. The details of spectral bands are shown in Table 5.2. All the bands are acquired in three different spatial resolutions: 10m per pixel, 20m per pixel and 60m per pixel. Bands with lower resolution can be upsampled to 10m per pixel using cubic-spline interpolation. The public multispectral version of the EuroSAT dataset is already upsampled using this interpolation method.

Visualization of RGB bands of EuroSAT images is already shown in Figure 4.3, Section 4.2.1. Here I visualize all bands of multispectral images from EuroSAT in Figure 5.6 and Figure 5.7. The first image in each row is of the RGB visible bands. The remaining 13 images in that row come from B01, B02, B03, B04, B05, B06, B07, B08, B08A, B09, B10, B11 and B12. Two images are randomly selected from each class to be visualized.

Preprocessing

Radiometric resolution is a measure of the ability of an imaging system to record different levels of brightness or tone. According to [58], the radiometric resolution of Sentinel-2 is 12-bit. This gives a potential range of brightness levels from 0 – 4095. But in experiments, I found there exist some pixels in some images whose pixel values are greater than 4095. This is probably due to the noise or other artefacts. The **normalization** on the dataset is necessary and can be done following Equation 5.13. Those values greater than 4095 will be set as 4095 in the preprocessing step.

$$P_{i,j,c} = \frac{\min(p_{i,j,c}, 4095)}{4095} \quad (5.13)$$

where i denotes the horizontal coordinate, j denotes the vertical coordinate and c denotes the channel/band. $p_{i,j,c}$ denotes the pixel value before normalization in position

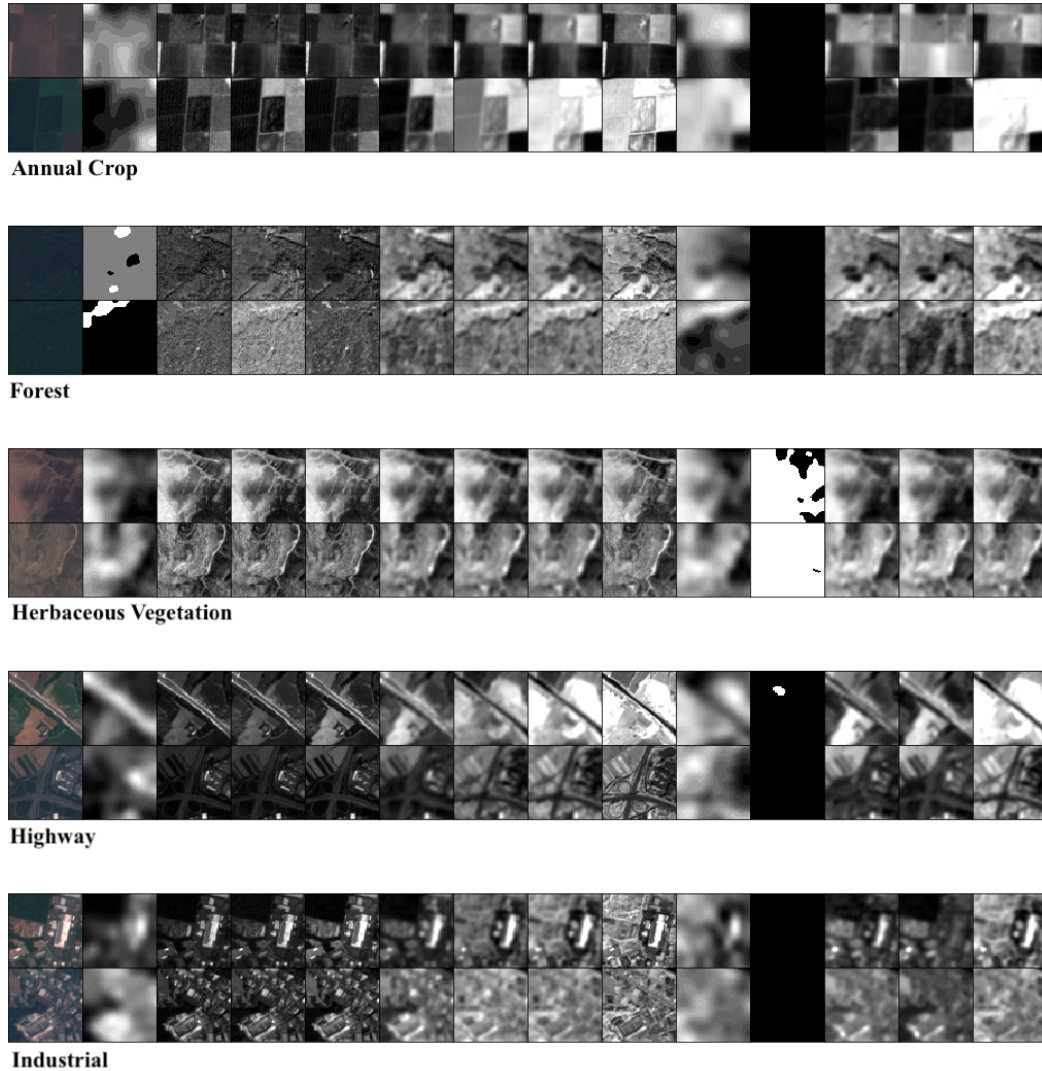


Figure 5.6: Visualization of multispectral remote sensing images in EuroSAT. The first column includes RGB images and other columns include single spectral images from different bands. Two images are randomly sampled from classes: Annual Crop, Forest, Herbaceous Vegetation, Highway and Industrial.

5 Unsupervised Adversarial Hashing for Multispectral Images

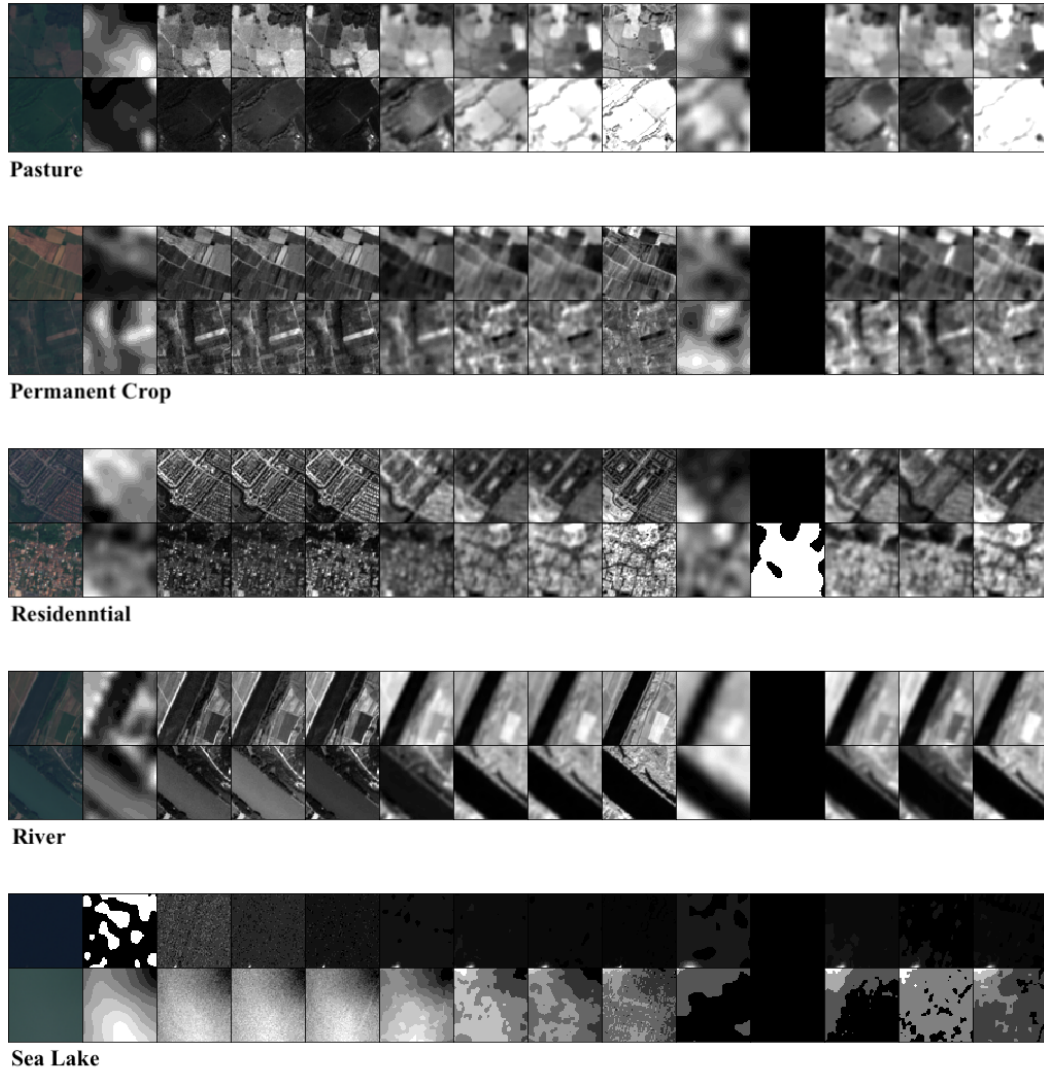


Figure 5.7: Visualization of multispectral remote sensing images in EuroSAT. The first column includes RGB images and other columns include single spectral images from different bands. Two images are randomly sampled from classes: Pasture, Permanent Crop, Residential, River, Sea Lake.

Table 5.2: All 13 bands of multispectral images in EuroSAT. The band name, the purpose, the spatial resolution and the central wave length are listed here.

Band	Purpose	Spatial Resolution	Central Wavelength
		m	nm
B01	Aerosol detection	60	443
B02	Blue	10	490
B03	Green	10	560
B04	Red	10	665
B05	Vegetation classification	20	705
B06	Vegetation classification	20	740
B07	Vegetation classification	20	783
B08	Near infrared	10	842
B08A	Vegetation classification	20	865
B09	Water vapor	60	945
B10	Cirrus	60	1375
B11	Snow / ice / cloud discrimination	20	1610
B12	Snow / ice / cloud discrimination	20	2190

(i, j, c) . $P_{i,j,c}$ denotes the pixel value after normalization in position (i, j, c)

Only 10 bands whose spatial resolution is $10m$ per pixel or $20m$ per pixel are taken into consideration in our experiments. Because other three bands' spatial resolution is too low, which is only $60m$ per pixel. And the other reason is that these low-resolution bands are used for detecting aerosols, water vapor, or cirrus in the atmosphere.

Different from experiments in Chapter 4 whose testing set (as query set and archive set) is very small with only 1000 images, the dataset splitting here is more reasonable, and more close to the real situation of large scale remote sensing image retrieval. 100 images are randomly sampled from each class to form the query set. The remaining images form the archive set. Meanwhile, 500 images per class are randomly sampled in the archive set to build the training set. Table 5.3 clearly shows how the dataset is split.

Table 5.3: Illustration of dataset splitting.

Query Set	1000 (100*10)
Training Set	5000 (500*10)
Archive Set	26000 (Training set is also included)

5 Unsupervised Adversarial Hashing for Multispectral Images

Table 5.4: The detailed architecture of the generator and the discriminator. l denotes the code length. c denotes the number of images channels.

Operation	Kernel	Strides	#Kernel	BN	Nonlinearity	Output
G - ($l, 1$) input						
Linear + Reshape			512	✓	ReLU	(4, 4, 512)
Transposed Convolution	5×5	2×2	256	✓	ReLU	(8, 8, 256)
Transposed Convolution	5×5	2×2	128	✓	ReLU	(16, 16, 128)
Transposed Convolution	5×5	2×2	64	✓	ReLU	(32, 32, 64)
Transposed Convolution	5×5	2×2	c	✗	Tanh	(64, 64, c)
D - (64, 64, c) input						
Convolution	3×3	2×2	32	✓	Leaky ReLU	(32, 32, 32)
Convolution	3×3	1×1	64	✓	Leaky ReLU	(32, 32, 64)
Convolution	3×3	2×2	128	✓	Leaky ReLU	(16, 16, 128)
Convolution	3×3	1×1	256	✓	Leaky ReLU	(16, 16, 256)
Convolution	3×3	2×2	256	✓	Leaky ReLU	(8, 8, 256)
Flatten						($8 \times 8 \times 256,$)
Output Part	refer Table 5.5					

5.2.2 Implementation Details

Network Architecture

In the experiments, three GANs were used and compared. The main bodies of these GANs have the same architecture as shown in Table 5.4. The input of the generator is the noise vector sampled from a uniform distribution in the range $[-1, 1]^l$. If the input of the discriminator is a RGB image, then $c = 3$. If the input of the discriminator is a multispectral image, then $c = 10$.

The difference between these three GANs lies in the different architectures of the discriminator’s output part, as shown in Table 5.5. The GAN for RGB images and the GAN for Multispectral images actually have the same architecture. Their output parts are also the same and are made up of three consecutive fully connected layers. The output of the last but one layer is taken as the non-binary feature representation for image retrieval task. The GAN for *Unsupervised Adversarial Hashing for Multispectral Images* also has three fully connected layers, but not in a consecutive way. The first fully connected layer’s output which has the shape (1024,) will be fed into two parallel fully connected layers. One is for generating binary hash codes for retrieval. The other is for source prediction.

Table 5.5: The **Output Part** of GAN and GAN_{UAHM}.

Model	Input	Operation	BN	Nonlinearity	Output
GAN RGB / GAN MS	(8 × 8 × 256,)	Linear	✓	Leaky ReLU	(1024,)
	(1024,)	Linear	✓	Leaky ReLU	(<i>l</i> ,) non-binary
	(<i>l</i> ,)	Linear	✗	Sigmoid	(1,)
GAN _{UAHM}	(8 × 8 × 256,)	Linear	✓	Leaky ReLU	(1024,)
	(1024,)	Linear	✓	Tanh	(<i>l</i> ,) binary
	(1024,)	Linear	✗	Sigmoid	(1,)

Hyperparameters

I set batch size as 64, learning rate as 0.0001, β_1 in Adam optimizer as 0.5 and β_2 in Adam optimizer as 0.999. The α in Leaky ReLU is 0.2. w_1 and w_2 in joint loss Equation 5.8 are 0.1 and 0.1. η in feature fusion Equation 5.9 is 6. K_1 and K_2 in SSM building Equations 5.10 and 5.11 are 500 and 300.

Image Retrieval

In GAN RGB and GAN MS, the feature representation is non-binary. When a query image is given, its feature representation is firstly obtained by using the trained discriminator of the GAN. Then the cosine distances between this feature representation and all feature representations in the archive are calculated using Equation 5.14. Finally, a ranking list will be given as the retrieval result.

$$cDis_{i,j} = \frac{\mathbf{F}(\mathbf{x}_i)\mathbf{F}(\mathbf{x}_j)}{|\mathbf{F}(\mathbf{x}_i)||\mathbf{F}(\mathbf{x}_j)|} \quad (5.14)$$

where $\mathbf{F}(\mathbf{x}_i)$ denotes the feature representation of image \mathbf{x}_i .

For GAN_{UAHM}, the feature representation is binary because Tanh is used as the activation function and a binary constraint is used during the training. During the retrieval process, query image's binary representation is obtained using GAN_{UAHM} firstly. Then $(-1, 1)^l$ representation is turned into real binary code $(0, 1)^l$. The Hamming distance, which can be calculated in Equation 5.15, is used as the distance metric. Finally a ranking list will be given as the retrieval result.

$$hDis_{i,j} = \sum_{k=1}^l \delta(\mathbf{F}(\mathbf{x}_i)_k, \mathbf{F}(\mathbf{x}_j)_k), \text{ where } \delta(a, b) = \begin{cases} 0 & a = b \\ 1 & a \neq b \end{cases} \quad (5.15)$$

5.2.3 Evaluation Metrics

Semantic Similarity Matrix

As mentioned before, the semantic similarity matrix for the training set is built in an unsupervised way. SSM is evaluated on two metrics: accuracy and IoU (Intersection over Union).

The accuracy evaluates the percentage of entries in SSM which give the correct similarity value. It can be calculated as Equation 5.16.

$$Acc = \frac{\sum_{i=0, j=0}^N [(S_{i,j} * (S_{gt})_{i,j} + 1)/2]}{N^2} \quad (5.16)$$

where $S_{i,j}$ denotes the similarity value of the i -th image and the j -th image from SSM, $(S_{gt})_{i,j}$ denotes the similarity value of the i -th image and the j -th image from ground truth.

IoU evaluates the similarity between estimated SSM and ground truth SSM. It can be calculated as Equation 5.17.

$$IoU = \frac{|set(S = 1) \cap set(S_{gt} = 1)|}{|set(S = 1) \cup set(S_{gt} = 1)|} \quad (5.17)$$

where $set(S = 1)$ denotes the set of SSM's entries where the similarity value is 1.

Image Retrieval

For image retrieval, I also use the evaluation metric, MAP, that is used in Chapter 4. MAP, which is short for Mean Average Precision, is a commonly used metric to evaluate the image retrieval performance. It can be computed from Equation 5.18.

$$MAP = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{n_i} \sum_{j=1}^{N_i} precision(R_i^j) \delta(j) \quad (5.18)$$

In this equation, Q denotes the set of all query images. n_i is the number of images similar to $q_i \in Q$. N_i is the size of the ranking list. R_i^j is the subset of ranked results, which only includes the result from the 1-st to the j -th position in the ranking list. $\delta(j) = 0$ when the j -th retrieved image is not relevant to the query image. $\delta(j) = 1$ when the j -th retrieved image is relevant to the query image. In the experimental results part, we use MAP@100. This means $N_i = 100$. MAP is calculated only on the first retrieved 100 results.

TopN-Precision is another commonly used evaluation metric for image retrieval. It is just the precision of the first N retrieved images and can be calculated from Equation 5.19. In the experimental results part, Top20-Precision is used.

$$\text{TopN-Precision} = \frac{\# \text{Relevant Retrieved Images}}{\# \text{All Retrieved Images}} = \frac{\# \text{Relevant Retrieved images}}{N} \quad (5.19)$$

Different from MAP or TopN-Precision these numerical metrics, Precision-Recall Curve is a commonly used graphical metric of image retrieval systems. The precision and the recall are calculated as Equation 5.20 and Equation 5.21. The curve shows how the precision and the recall change when retrieving different number of images.

$$\text{Precision} = \frac{\# \text{Relevant Retrieved Images}}{\# \text{All Retrieved Images}} \quad (5.20)$$

$$\text{Recall} = \frac{\# \text{Relevant Retrieved Images}}{\# \text{All Relevant Images}} \quad (5.21)$$

In our experiments, we calculate the precision values and recall values every after 500 retrieved images and then plot the final Precision-Recall Curve.

5.3 Experimental Results

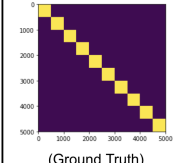
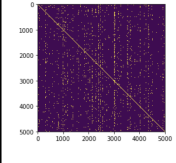
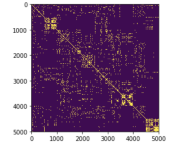
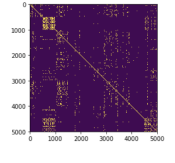
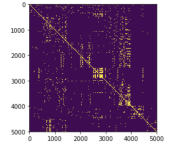
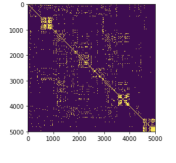
5.3.1 Semantic Similarity Matrix Comparison

Here the semantic similarity matrix building results using the original method [11] and the improved method proposed in this thesis are compared. The training set has 5000 images with 500 images from each class. To better visualize the SSM building results, images in the training set are arranged according to their class. Images from the same class will be placed in consecutive 500 entries of the list. The visualization of the ground truth SSM is displayed in the lower left of Table 5.6. The originally proposed method used Resnet v2 to extract high-dimensional features for SSM building. Its results are shown in the second column of Table 5.6. The improved method uses high-dimensional features extracted from Inception v3 combined with local color histogram features to build the SSM. Its results are shown in the last column of Table 5.6. The last row of the table shows the visualization of the built SSMs using different methods.

We can see that for semantic spatial feature extraction, Inception v3 achieves better results than Resnet v2. It's interesting because Resnet v2 beats Inception v3 in natural image recognition task. I attribute this phenomenon to two possible reasons. The first possible reason is that remote sensing images and natural images are very different. Some remote sensing images have just uniform colors like Sea Lake images in EuroSAT. Some remote sensing images have uniform distributed textures like Herbaceous Vegetation images or Residential images in EuroSAT. The second possible reason is

5 Unsupervised Adversarial Hashing for Multispectral Images

Table 5.6: Semantic Similarity Matrix results comparison.

Method	Resnet_v2	Inception_v3	Raw Color	Local Histogram	Inception_v3+Local Histogram	
	Spatial		Spectral			
Accuracy	0.8761	0.8899	0.8884	0.8894	0.8948	
IoU	0.0331	0.1626	0.0853	0.1071	0.1628	
Visualization						
	(Ground Truth)					

that features extracted from those pretrained deep neural networks are perfect for classification. Perfect classification features don't promise perfect features for similarity evaluation or image retrieval.

For spectral features, the local color histogram performs much better in similarity estimation as shown in the table. When we fuse features extracted from Inception v3 and color histogram features, we can build a much better semantic similarity matrix than only considering one type of the feature.

5.3.2 Results of Image Retrieval

Overall Results

Image retrieval is evaluated using MAP as the metric. The experiments are conducted on three different feature lengths: 32, 64, 128. Table 5.7 and Table 5.8 shows the MAP values using different models and different code lengths. In the table, GAN RGB or GAN MS denotes the method using the discriminator of GAN to extract features from RGB images or multispectral images. GAN_{UAHM} denotes the proposed method of Unsupervised Adversarial Hashing for Multispectral Images. DHNN MS denotes the network from GAN_{UAHM} with adversarial structure removed, which is used for ablation study to know the importance of adversarial learning in the proposed method. It's worth noting that GAN_{UAHM} outputs approximate hash codes which lie in the range $[-1, 1]$ and are not binary actually. Using sign function can make the outputs binary. Image retrieval results on both approximate non-binary codes and binarized hash codes are tested and listed in Table 5.7 and Table 5.8.

We can see that in all three tested code lengths there exists a tendency for both MAP@100 and Top-20 Precision to increase from GAN RGB to GAN MS and from

Table 5.7: MAP@100 of image retrieval using different models and different code lengths.

Code length	32	64	128
DHNN MS (bin)	0.2752	0.2842	0.2720
GAN RGB (non-bin)	0.5184	0.5287	0.5480
GAN MS (non-bin)	0.5655	0.5915	0.6002
GAN _{UAHM} (non-bin)	0.6510	0.6575	0.6810
GAN _{UAHM} (bin)	<u>0.6470</u>	<u>0.6556</u>	<u>0.6785</u>

Table 5.8: Top20 Precision of image retrieval using different models and different code lengths.

Code length	32	64	128
DHNN MS (bin)	0.1542	0.1808	0.2283
GAN RGB (non-bin)	0.5113	0.5217	0.5398
GAN MS (non-bin)	0.5603	0.5221	0.5980
GAN _{UAHM} (non-bin)	0.6393	0.6452	0.6734
GAN _{UAHM} (bin)	<u>0.6367</u>	<u>0.6432</u>	<u>0.6705</u>

GAN MS to GAN_{UAHM}. It proves the effectiveness of two strategies, considering multiple bands and adding the semantic constraint. The proposed GAN_{UAHM} always achieves the best image retrieval result.

In the ablation study, we compare the results from DHNN MS and GAN_{UAHM}. And in both MAP@100 and Top20 Precision criteria, GAN_{UAHM} achieves much better results than DHNN MS. We can conclude that adversarial loss helps the hash learning.

We also plot the Precision-Recall Curve of these methods in Figure 5.8. The proposed GAN_{UAHM} achieved the best results. And the results of GAN MS is slightly better than that of GAN RGB, which shows the effectiveness of considering multiple bands of remote sensing images again.

Class-wise Results

To better understand the improvement of image retrieval in each specific class. We show the class-wise MAP results using different models in Table 5.9. MAP values are calculated when query images are from only one class. These classes are Annual Crop, Forest, Herbaceous Vegetation, Highway, Industrial, Pasture, Permanent Crop, Residential, River and Sea Lake.

We can see that the proposed GAN_{UAHM} achieves the best results in all classes. We can also find that the improvement in some class are very big like Annual Crop and Industrial. In some classes like Highway or Pasture, there are just small increases.

5 Unsupervised Adversarial Hashing for Multispectral Images

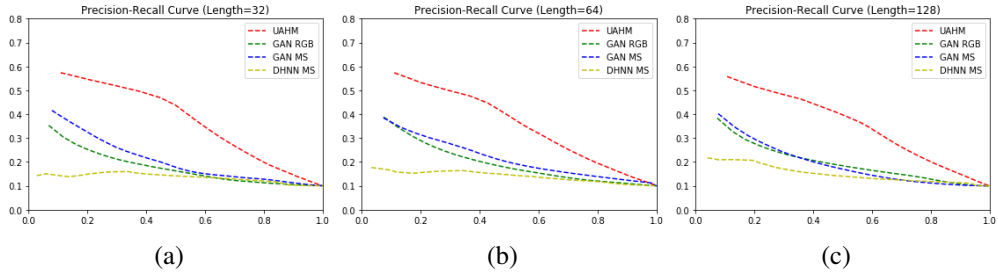


Figure 5.8: Precision-Recall Curve of image retrieval using different models and different code lengths.

Table 5.9: Class-wise MAP of image retrieval using different models. Code length is 128.

	Ann.	For.	Her.	Higa.	Ind.	Pas.	Per.	Res.	Riv.	Sea.
GAN RGB	0.4325	0.8267	0.4940	0.2528	0.8161	0.5655	0.4348	0.5367	0.3418	0.7793
GAN MS	0.5201	0.8557	0.5256	0.2554	0.5960	0.6543	0.5393	0.5078	0.5724	0.9747
GAN _{UAHM}	0.6377	0.9245	0.5730	0.2713	0.8887	0.6557	0.5816	0.5551	0.7103	0.9873

When we analyze Table 5.6 and Table 5.9 together, this phenomenon can be attributed to the incorrectness of SSM in some classes.

Visualization of the Retrieved Images

To explicitly show the image retrieval results using different models. Figure 5.9 is the visualization of image retrieval results. Three images were randomly selected as the query images. 9 retrieved images at 0, 50, 100, 150, 200, 250, 300, 350, and 400 of the ranking list are demonstrated. We can also see the improvement from GAN RGB to GAN MS and from GAN MS to GAN_{UAHM}. It's worth noting that some images in the retrieved list look a little different, e.g., the last several images in the third row when the query image is industrial. Even though they look different from the query image, they are still correctly retrieved. This is because the third row shows the result using GAN_{UAHM} and semantic guidance is introduced in this method. The semantic similarity matrix helps the proposed method achieve better results in the semantic level.

5.3.3 Analysis of Image Generation

The images generated by three different unconditional models from this chapter and one conditional model from Chapter 4 are shown in Figure 5.10. These images are 36 images from a 64-image batch generated by GAN's generator. We can easily see that conditional GAN generates much better and more realistic fake images. Unconditional

5.3 Experimental Results

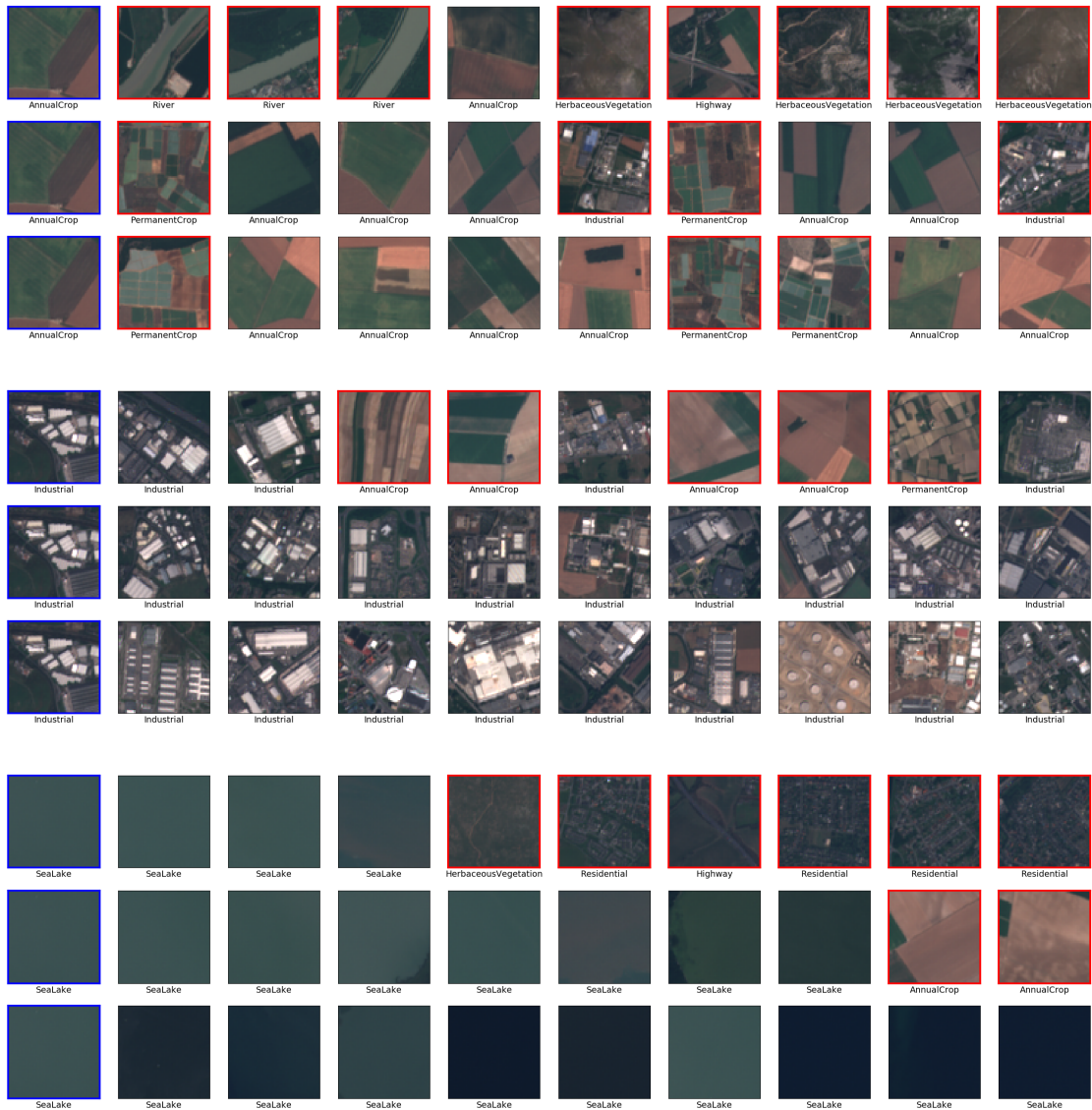


Figure 5.9: Visualization of image retrieval results using different models. The first column shows the query images. From the second column to the last column, images at 0, 50, 100, 150, 200, 250, 300, 350, 400 of the retrieved ranking list are shown. In each group of three rows, the first row shows the results using GAN RGB, the second row shows the results using GAN MS and the third row shows the results using GAN_{UAHM} .

5 Unsupervised Adversarial Hashing for Multispectral Images

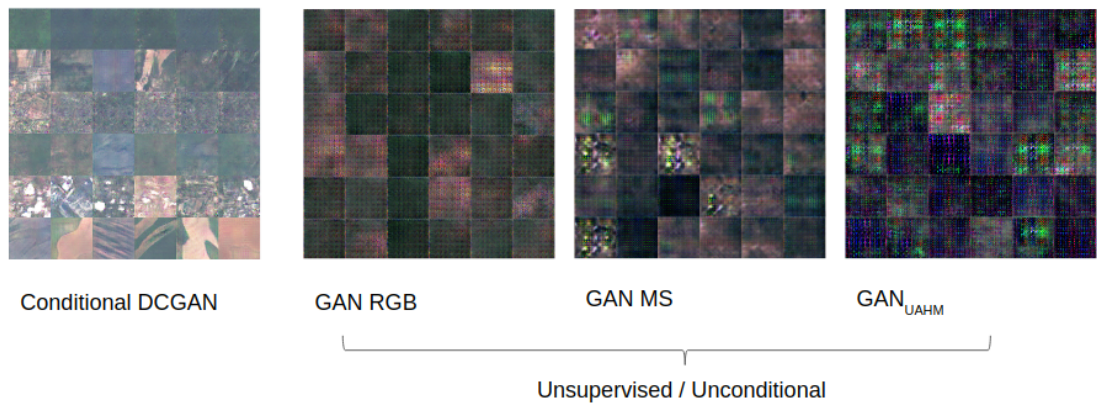


Figure 5.10: Fake images generated by different models.

GAN can only generate approximate colors, textures and basic shapes. We can not recognize which class each image generated by unconditional GAN belongs to. It's worth noting that GAN MS and GAN_{UAHM} can generate multispectral images which will be visualized later. Shown here are just RGB bands of generated multispectral images to make them comparable with images generated from the conditional DCGAN and GAN RGB.

This image quality difference between conditional GAN and unconditional GAN is reasonable. The bad quality of remote sensing images generated by unconditional GAN can be attributed to two reasons: big internal variance and unavailable condition information. EuroSAT images have a variety of different appearances as shown in Figure 5.11. We can see that some images have almost no texture and have just one color. The examples are images from Sea Lake and Forest classes. Meanwhile, some images have more complicated textures and shape characteristics. The examples are images from Highway and Residential classes. Unavailable condition information (which are labels in conditional DCGAN and ACGAN) means learning a universal model to model the whole distribution of remote sensing images. It is more difficult than learning class-specific model in conditional DCGAN and ACGAN.



Figure 5.11: A minibatch of real images from the training set.

GAN MS and GAN_{UAHM} can generate fake multispectral images and we already see

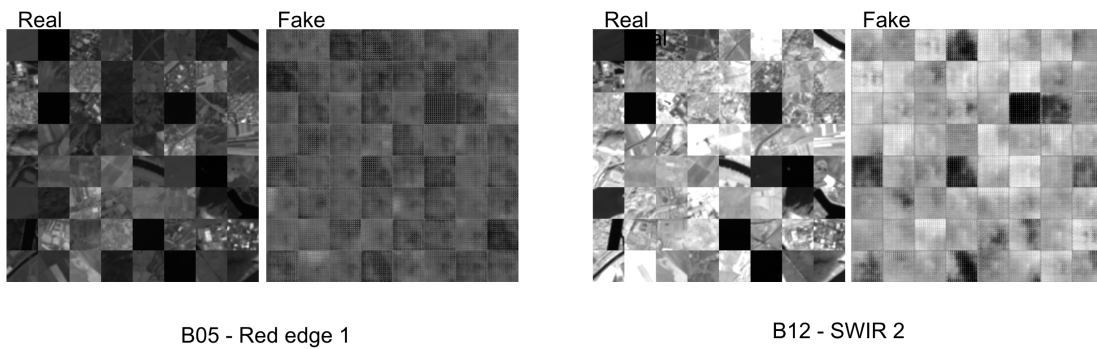


Figure 5.12: Visualization of non-RGB bands of generated multispectral images.

the RGB bands of the generated multispectral images in Figure 5.10. The visualization of other bands of generated multispectral images is shown in Figure 5.12. We can see that these generated bands are not so realistic and do not have too much recognizable things in the images. Nevertheless, they still learn some basic characteristics like colors or basic textures. We can see from Figure 5.12 that GAN does well in modeling the real images in the aspect of colors.

All the results show that images generated from unsupervised GAN are not very realistic, no matter in RGB bands or other bands. This is caused by two reasons: internal variance of the dataset and unavailable condition on the model. But generated images at least capture some basic characteristics of real images: color ranges, basic textures and basic shapes.

5.3.4 Correlation between Image Retrieval and Image Quality

Image generation and representation learning are two different tasks that a GAN can do. And they are done in different parts of the GAN. Image generation is the generator's task and representation learning is the discriminator's task. In this chapter, we address the image retrieval problem with the GAN's representation learning ability. Here we want to explore the correlation between fake image quality and image retrieval performance.

Firstly, the training process of GAN MS is shown in Figure 5.14. The generated images (RGB bands) and image retrieval performance evaluated on MAP are explicitly demonstrated in these two figures. And the evaluation is done every 2500 training steps. It's worth noting that MAP here are MAP@ALL or MAP@26000.

Mode Collapse: Mode collapse means the generator can only generate limited diversity of modes even with various inputs.

We can see that the training will fail after some steps. Before the training fails, the generated images after 5000 training steps are better than those after 2500 training steps but lead to worse image retrieval performance (lower MAP). After 7500 training steps, the generator can not generate realistic images. To some degree, the generator just

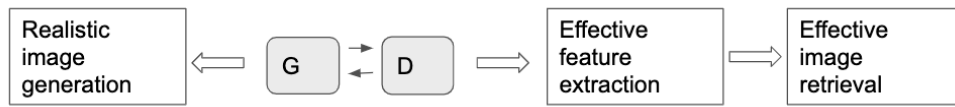


Figure 5.13: Analysis of image quality and image retrieval.

generates noise. But the image retrieval performance doesn't decrease too much and increases again after some training steps. These two facts show that image retrieval and image quality have no obvious correlation.

GAN is made up of a generator and a discriminator. Figure 5.13 shows the analysis of the correlation between image quality and image retrieval. There exists a two-player game between G and D. The game is around real or fake images. There is no direct correlation between image quality and image retrieval performance. On the other hand, the ideal final state of GAN training is a Nash equilibrium. This state means the generator can generate very realistic images. The best feature for retrieval is different from the best feature for source prediction. Therefore better image quality doesn't mean better features for retrieval. Besides, even when training fails, the discriminator can still be able to learn to extract good features and achieve better image retrieval performance.

5.4 Conclusion

In this chapter, I addressed unsupervised hashing for image retrieval with a novel GAN-based method. Even though there exists some research using GAN for hashing, my work is the first to make use of multiple bands from multispectral remote sensing images. Thorough experiments proved the effectiveness of three strategies of the proposed method: using GAN for representation learning, taking multiple bands of remote sensing images into consideration, and adding a semantic similarity constraint.

Image generation is the basic function of a GAN. The proposed GAN-based hashing can generate fake remote sensing images as well. And what's more, the proposed method can generate multispectral images. Through image quality analysis, we find that images generated from unsupervised GAN are less realistic than images generated from conditional GAN. Nevertheless, they still capture some characteristics not only in RGB bands but also in other bands of multispectral images. These less realistic fake images can still help the representation learning of the discriminator.

GAN is suitable for unsupervised representation learning. The discriminator in GAN can be used to extract features for image retrieval. The quality of generated images and the performance of image retrieval have no direct correlation.

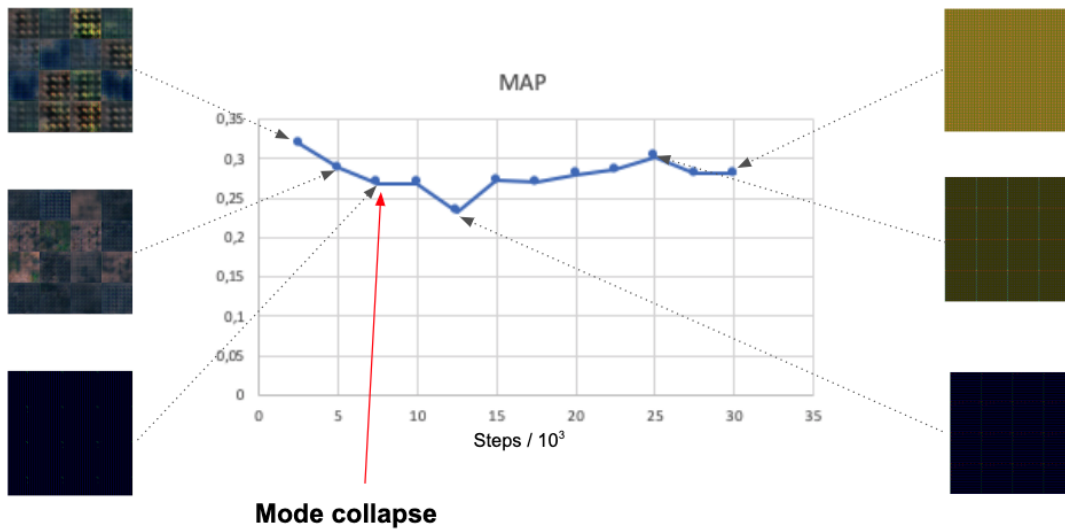
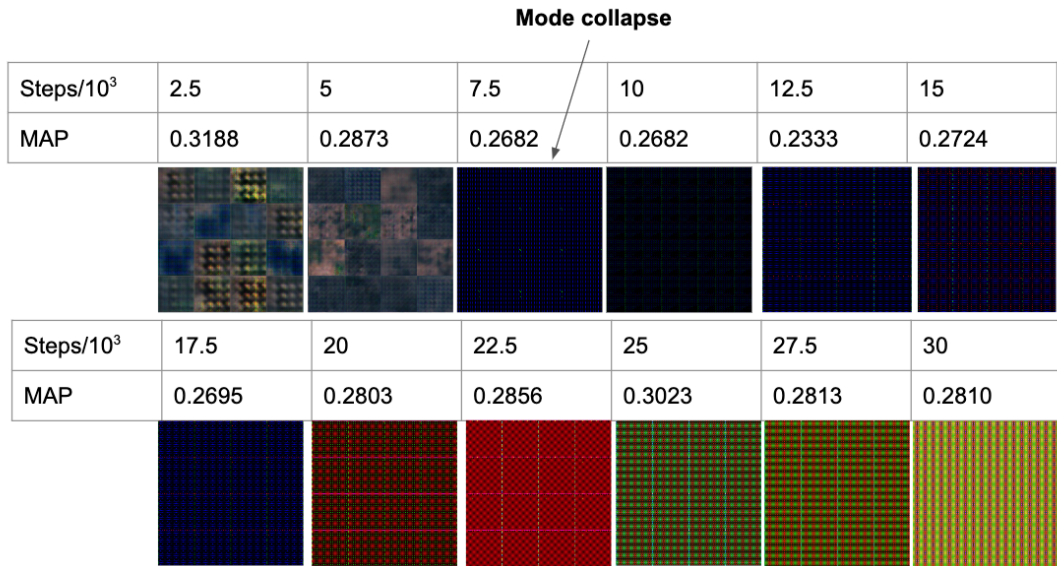


Figure 5.14: Illustration of the change of image quality and image retrieval during the training process.

6 Conclusion and Future

6.1 Conclusion

In this thesis, we try to address the problems of a large amount of data with a limited number of labels in the remote sensing image retrieval task. We proposed GAN-based methods as solutions. The GAN, which is short for generative adversarial network, is a recently proposed generative model to model the data distribution in an implicit way. A GAN is made up of a discriminator and a generator. The GAN can help us with the problems in remote sensing data in two different directions. The first direction is to use conditional GAN to generate more remote sensing images with labels. The generated images can help the training of the deep hashing neural network for image retrieval. The second direction is to use the unsupervised representation learning ability of the discriminator for hash learning. These two directions are based on two different components of the GAN: the generator and the discriminator. For this reason, the research of this thesis is divided into two sub-topics:

- Supervised Hashing Boosted by GANs-generated Images
- Unsupervised Adversarial Hashing for Multispectral Images

In the first direction, we successfully built conditional DCGAN and ACGAN for remote sensing image generation. The generated images resembled real images. We evaluated the images generated from the two GANs mentioned. The images generated from ACGAN beat images generated from conditional DCGAN in two widely used metrics, IS and FID. And we combined generated images and real images to build a larger labeled training set. The best image retrieval performance was achieved when adding images generated from ACGAN to the training set.

In the second direction, we built unsupervised GAN to do representation learning for image retrieval. Firstly, we tested using the discriminator of the GAN to extract feature representations directly. The cosine distances were calculated based on extracted features for the similarity ranking. We improved the vanilla GAN by considering multiple bands of multispectral remote sensing images and adding a semantic constraint. These adjustments proved to be effective on image retrieval performance. In the end, we analyzed the quality of the images generated from the unconditional/unsupervised GAN model. They were worse than images generated from conditional GAN, but still

captured some characteristics not only in RGB bands but also in other bands of multispectral images. We also analyzed the correlation between fake image quality and image retrieval performance. We concluded that they have no direct correlation.

6.2 Future Work

Even though we show that GAN can really help us with the problems in remote sensing image retrieval, there is still some improvement space. We provide two directions for future research in this section.

Our experiments use all bands with the spatial resolution of $10m$ per pixel and $20m$ per pixel. Future work can be focused on exploring different band combinations for image retrieval. Exploring the instance-aware band combination for image retrieval is also a meaningful topic.

We designed our GAN based on the vanilla GAN architecture. A lot of more complicated GAN architectures have been proposed nowadays. Future work can also be focused on applying those advanced GAN architectures like BiGAN or VAE-GAN to remote sensing image retrieval.

Bibliography

- [1] Airbus: Successfully launched Sentinel-2B to complete Europe's colour vision mission of Earth - Space - Airbus. <https://www.airbus.com/newsroom/press-releases/en/2017/03/airbus-successfully-launched-sentinel-2b-to-complete-europes-colour-vision-mission-of-earth.html>. (Accessed on 08/24/2019).
- [2] Erchan Aptoula. "Remote sensing image retrieval with global morphological texture descriptors". In: *IEEE transactions on geoscience and remote sensing* 52.5 (2013), pp. 3023–3034.
- [3] Erchan Aptoula. "Remote sensing image retrieval with global morphological texture descriptors". In: *IEEE transactions on geoscience and remote sensing* 52.5 (2014), pp. 3023–3034.
- [4] Pierre Baldi. "Autoencoders, unsupervised learning, and deep architectures". In: *Proceedings of ICML workshop on unsupervised and transfer learning*. 2012, pp. 37–49.
- [5] Avinash N Bhute and BB Meshram. "Text Based Approach For Indexing And Retrieval Of Image And Video: A Review". In: *arXiv preprint arXiv:1404.1514* (2014).
- [6] Yue Cao et al. "Hashgan: Deep learning to hash with pair conditional wasserstein gan". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 1287–1296.
- [7] Ning-San Chang and King Sun Fu. "A relational database system for images". In: *Pictorial Information Systems*. Springer, 1980, pp. 288–321.
- [8] Gal Chechik et al. "Large scale online learning of image similarity through ranking". In: *Journal of Machine Learning Research* 11.Mar (2010), pp. 1109–1135.
- [9] Jifeng Dai et al. "R-fcn: Object detection via region-based fully convolutional networks". In: *Advances in neural information processing systems*. 2016, pp. 379–387.
- [10] Ritendra Datta, Jia Li, and James Z Wang. "Content-based image retrieval: approaches and trends of the new age". In: *Proceedings of the 7th ACM SIGMM international workshop on Multimedia information retrieval*. ACM. 2005, pp. 253–262.

Bibliography

- [11] Cheng Deng et al. “Unsupervised semantic-preserving adversarial hashing for image search”. In: *IEEE Transactions on Image Processing* (2019).
- [12] Jia Deng et al. “Imagenet: A large-scale hierarchical image database”. In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, pp. 248–255.
- [13] Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. “Adversarial feature learning”. In: *arXiv preprint arXiv:1605.09782* (2016).
- [14] John Duchi, Elad Hazan, and Yoram Singer. “Adaptive subgradient methods for online learning and stochastic optimization”. In: *Journal of Machine Learning Research* 12:Jul (2011), pp. 2121–2159.
- [15] Vincent Dumoulin and Francesco Visin. “A guide to convolution arithmetic for deep learning”. In: *arXiv preprint arXiv:1603.07285* (2016).
- [16] Vincent Dumoulin et al. “Adversarially learned inference”. In: *arXiv preprint arXiv:1606.00704* (2016).
- [17] Venice Erin Liong et al. “Deep hashing for compact binary codes learning”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 2475–2483.
- [18] Libing Geng et al. “Regularizing Deep Hashing Networks Using GAN Generated Fake Images”. In: *arXiv preprint arXiv:1803.09466* (2018).
- [19] Kamran Ghasedi Dizaji et al. “Unsupervised deep generative adversarial hashing network”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 3664–3673.
- [20] Ian Goodfellow et al. “Generative adversarial nets”. In: *Advances in neural information processing systems*. 2014, pp. 2672–2680.
- [21] Robert M Haralick, Karthikeyan Shanmugam, et al. “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] Kaiming He et al. “Identity mappings in deep residual networks”. In: *European conference on computer vision*. Springer. 2016, pp. 630–645.
- [24] Patrick Helber et al. “Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification”. In: *arXiv preprint arXiv:1709.00029* (2017).

- [25] Martin Heusel et al. “Gans trained by a two time-scale update rule converge to a local nash equilibrium”. In: *Advances in Neural Information Processing Systems*. 2017, pp. 6626–6637.
- [26] Yao Hongyu, Li Bicheng, and Cao Wen. “Remote sensing imagery retrieval based-on Gabor texture feature classification”. In: *Proceedings 7th International Conference on Signal Processing, 2004. Proceedings. ICSP’04. 2004*. Vol. 1. IEEE. 2004, pp. 733–736.
- [27] Ming-Kuei Hu. “Visual pattern recognition by moment invariants”. In: *IRE transactions on information theory* 8.2 (1962), pp. 179–187.
- [28] Trevor Huff and Prasanna Tadi. “Neuroanatomy, visual cortex”. In: *StatPearls [Internet]*. StatPearls Publishing, 2019.
- [29] Sergey Ioffe and Christian Szegedy. “Batch normalization: Accelerating deep network training by reducing internal covariate shift”. In: *arXiv preprint arXiv:1502.03167* (2015).
- [30] Phillip Isola et al. “Image-to-image translation with conditional adversarial networks”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 1125–1134.
- [31] Pieter Kempeneers and Pierre Soille. “Optimizing Sentinel-2 image selection in a Big Data context”. In: *Big Earth Data* 1.1-2 (2017), pp. 145–158.
- [32] Diederik P Kingma and Jimmy Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014).
- [33] Diederik P Kingma and Max Welling. “Auto-encoding variational bayes”. In: *arXiv preprint arXiv:1312.6114* (2013).
- [34] Alex Krizhevsky and Geoffrey E Hinton. “Using very deep autoencoders for content-based image retrieval.” In: *ESANN*. Vol. 1. 2011, p. 2.
- [35] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Advances in neural information processing systems*. 2012, pp. 1097–1105.
- [36] TL Kunil et al. “Pictorial data-base systems”. In: *Computer* 11 (1981), pp. 13–21.
- [37] Anders Boesen Lindbo Larsen et al. “Autoencoding beyond pixels using a learned similarity metric”. In: *arXiv preprint arXiv:1512.09300* (2015).
- [38] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. “Deep learning”. In: *nature* 521.7553 (2015), p. 436.
- [39] Yann LeCun et al. “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

Bibliography

- [40] Christian Ledig et al. “Photo-realistic single image super-resolution using a generative adversarial network”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 4681–4690.
- [41] Yansheng Li et al. “Large-scale remote sensing image retrieval by deep hashing neural networks”. In: *IEEE Transactions on Geoscience and Remote Sensing* 56.2 (2017), pp. 950–965.
- [42] 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.
- [43] Kevin Lin et al. “Learning compact binary descriptors with unsupervised deep neural networks”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 1183–1192.
- [44] Jonathan Long, Evan Shelhamer, and Trevor Darrell. “Fully convolutional networks for semantic segmentation”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 3431–3440.
- [45] 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.
- [46] Mehdi Mirza and Simon Osindero. “Conditional generative adversarial nets”. In: *arXiv preprint arXiv:1411.1784* (2014).
- [47] Augustus Odena, Christopher Olah, and Jonathon Shlens. “Conditional image synthesis with auxiliary classifier gans”. In: *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org. 2017, pp. 2642–2651.
- [48] Aaron Van den Oord et al. “Conditional image generation with pixelcnn decoders”. In: *Advances in neural information processing systems*. 2016, pp. 4790–4798.
- [49] Aaron van den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. “Pixel recurrent neural networks”. In: *arXiv preprint arXiv:1601.06759* (2016).
- [50] Greg Pass, Ramin Zabih, and Justin Miller. “Comparing Images Using Color Coherence Vectors.” In: *ACM multimedia*. Vol. 96. Citeseer. 1996, pp. 65–73.
- [51] Zhaofan Qiu et al. “Deep semantic hashing with generative adversarial networks”. In: *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM. 2017, pp. 225–234.
- [52] Alec Radford, Luke Metz, and Soumith Chintala. “Unsupervised representation learning with deep convolutional generative adversarial networks”. In: *arXiv preprint arXiv:1511.06434* (2015).

- [53] 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.
- [54] Yong Rui, Alfred C She, and Thomas S Huang. “Modified Fourier descriptors for shape representation-a practical approach”. In: *Proc of First International Workshop on Image Databases and Multi Media Search*. Citeseer. 1996, pp. 22–23.
- [55] Ruslan Salakhutdinov and Geoffrey Hinton. “Semantic hashing”. In: *International Journal of Approximate Reasoning* 50.7 (2009), pp. 969–978.
- [56] Tim Salimans et al. “Improved techniques for training gans”. In: *Advances in neural information processing systems*. 2016, pp. 2234–2242.
- [57] Florian Schroff, Dmitry Kalenichenko, and James Philbin. “Facenet: A unified embedding for face recognition and clustering”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 815–823.
- [58] *Sentinel-2 - Missions - Resolution and Swath - Sentinel Handbook*. <https://sentinel.esa.int/web/sentinel/missions/sentinel-2/instrument-payload/resolution-and-swath>. (Accessed on 09/08/2019).
- [59] Konstantin Shmelkov, Cordelia Schmid, and Karteek Alahari. “How good is my GAN?” In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 213–229.
- [60] Karen Simonyan and Andrew Zisserman. “Very deep convolutional networks for large-scale image recognition”. In: *arXiv preprint arXiv:1409.1556* (2014).
- [61] John R Smith and Shih-Fu Chang. “Transform features for texture classification and discrimination in large image databases”. In: *Proceedings of 1st International Conference on Image Processing*. Vol. 3. IEEE. 1994, pp. 407–411.
- [62] Jingkuan Song et al. “Binary generative adversarial networks for image retrieval”. In: *Thirty-Second AAAI Conference on Artificial Intelligence*. 2018.
- [63] Michael J Swain and Dana H Ballard. “Color indexing”. In: *International journal of computer vision* 7.1 (1991), pp. 11–32.
- [64] Christian Szegedy et al. “Going deeper with convolutions”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 1–9.
- [65] Christian Szegedy et al. “Inception-v4, inception-resnet and the impact of residual connections on learning”. In: *Thirty-First AAAI Conference on Artificial Intelligence*. 2017.
- [66] Christian Szegedy et al. “Rethinking the inception architecture for computer vision”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 2818–2826.

Bibliography

- [67] Hideyuki Tamura, Shunji Mori, and Takashi Yamawaki. “Textural features corresponding to visual perception”. In: *IEEE Transactions on Systems, man, and cybernetics* 8.6 (1978), pp. 460–473.
- [68] Tijmen Tieleman and Geoffrey Hinton. “Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude”. In: *COURSERA: Neural networks for machine learning* 4.2 (2012), pp. 26–31.
- [69] *Understanding the backward pass through Batch Normalization Layer*. <https://kratzert.github.io/2016/02/12/understanding-the-gradient-flow-through-the-batch-normalization-layer.html>. (Accessed on 08/23/2019).
- [70] Ji Wan et al. “Deep learning for content-based image retrieval: A comprehensive study”. In: *Proceedings of the 22nd ACM international conference on Multimedia*. ACM. 2014, pp. 157–166.
- [71] Yi Yang and Shawn Newsam. “Geographic image retrieval using local invariant features”. In: *IEEE Transactions on Geoscience and Remote Sensing* 51.2 (2012), pp. 818–832.
- [72] Yi Yang and Shawn Newsam. “Geographic image retrieval using local invariant features”. In: *IEEE Transactions on Geoscience and Remote Sensing* 51.2 (2013), pp. 818–832.
- [73] Matthew D Zeiler and Rob Fergus. “Visualizing and understanding convolutional networks”. In: *European conference on computer vision*. Springer. 2014, pp. 818–833.